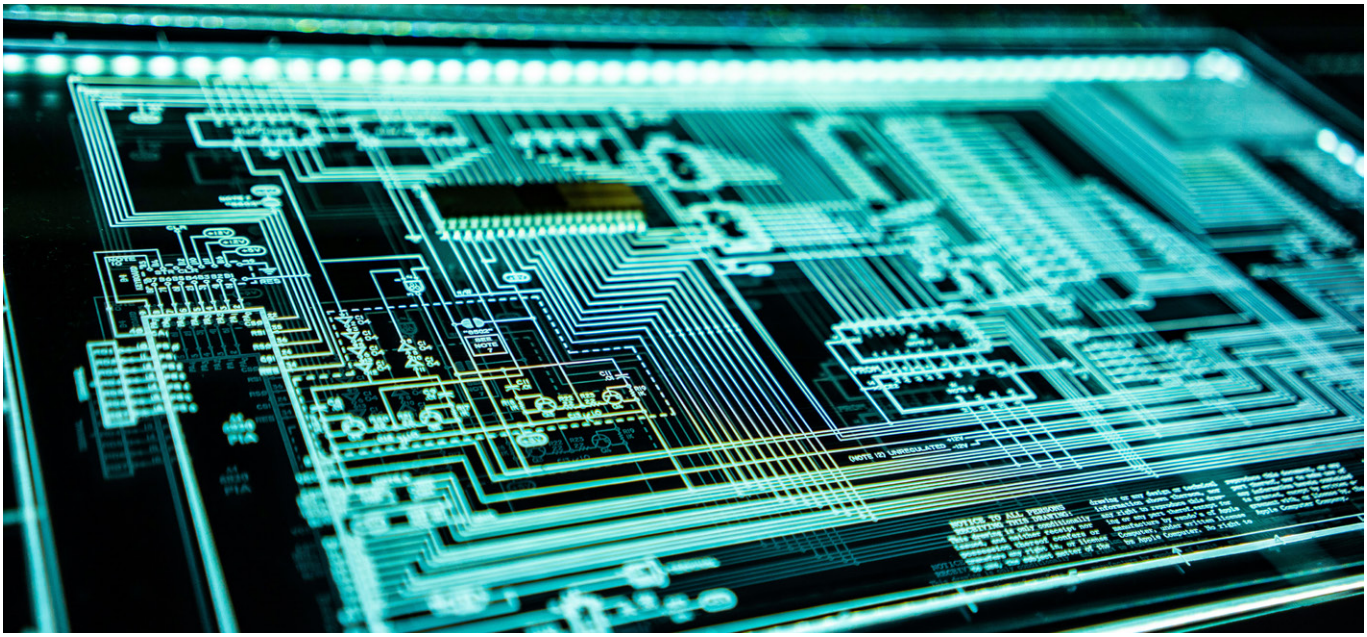
STM Cyber

OCAK-MART 2025

# SİBER TEHDİT DURUM RAPORU

STM MÜHENDİSLİK TEKNOLOJİ DANIŞMANLIK

thinktech STM Teknolojik Düşünce Merkezi



# İÇİNDEKİLER

## ŞEKİLLER



## TABLO LİSTESİ

# GİRİŞ

2025 yılının birinci çeyreğinde Siber Güvenlik Müdürlüğü tarafından hazırlanan raporumuzda yine birbirinden ilginç konularla karşınızdayız. Bu bağlamda ilk olarak, IT yapıların güvenlik operasyonlarını otomatikleştirmek amacıyla geliştirilen bir yazılım olan SOAR'ın önemine odaklanıyoruz. Ardından, tarihin şahit olduğu en büyük kripto para hırsızlığı olan "Bybit Hack Vakası"nı inceliyoruz. Sonrasında, "DEVSECOPS: Yazılım Güvenliğinin Sürekli Entegrasyonu" başlığı altında yazılım güvenliğini sağlamak için DEVSECOPS sürecinin prensiplerini ele alıyoruz. Ayrıca, insan faktörünün siber güvenliğe olan etkisi ve siber güvenlikte insan faktörünü azaltmak için dikkat edilmesi gereken noktaları sizlere aktarıyoruz.

Ekim-Aralık 2024 Siber Tehdit Raporu'nun dönem konusu olan "Hareket Eden, Değişen Hedefler -Automated Moving Target Defence (AMTD)" konusunda bahsedilen Aldatma (Deception) teknikleri konusuna bu çeyrek raporumuzda yer veriyoruz. Son olarak "Zararlı Yazılım Analizinde Fuzzy Hashing" bölümünde fuzzy hash kavramına ve geleneksel hash'e karşı sağladığı avantajlara odaklanıyoruz.

"Siber Güvenlik Profesyonelleri için Prompt Mühendisliğinden En İyi Şekilde Faydalanma" konusunu ise, bu çeyreğin dönem konusu olarak belirledik. Bu bölümde günümüzde çokça kullanılan yapay zekâ sistemlerinin anlamlı ve etkili yanıtlar verebilmesi için kullanılan prompt mühendisliği tekniklerinden ve çeşitli siber güvenlik işlevlerini desteklemeye başlayan Dil Modellerinin kullanımından bahsedeceğiz.

Son olarak, her raporumuzda güncellediğimiz honeypot verilerimizi inceleyeceğiz. Bilişim dünyasındaki güvenlik tehditlerini ve korunma stratejilerini anlamak isteyen herkes için önemli bir kaynak olan bu raporumuzu, güvenlik bilincinizi artırmak ve siber tehditlere karşı daha hazırlıklı olmanız için incelemenizi tavsiye ederiz. Güvende kalın!

# TEKNOLOJİK GELİŞMELER VE SİBER GÜVENLİK

## 1. Kurumsal BT Yapılarının SOAR (Security Automation And Orchestration Response) ile Güvenlik Otomasyonu

### SOAR Teknolojilerinin Temel Özellikleri

SOAR, kurumsal IT yapılarının güvenlik operasyonlarını hızlandırmak ve otomatikleştirmek amacıyla geliştirilen bir yazılım platformudur. Temel işlevleri sayesinde, güvenlik olaylarının orkestrasyonu ve otomasyonu sağlanarak hızlı bir şekilde eyleme geçilebilir. Bu yetenek sayesinde, güvenlik ekibi olaylara hızlı bir şekilde müdahale etme kabiliyeti kazanır. Güvenlik ekibinin manuel müdahaleleri azalır ve SOAR rutin güvenlik görevlerini otomatikleştirerek insan hatası riskini ortadan kaldırır.

Kısaca, farklı güvenlik ürünlerinden gelen uyarıları analiz eder, ilgili verileri toplayarak olayı anlamlı hale getirir ve ardından otomatik olarak bir yanıt planı uygular. Bu sayede kurumlar, daha düşük yanıt süresi ve daha etkin bir güvenlik operasyonu elde ederler.

### SOAR Üzerinde Çevre Güvenlik Yazılımların Rolü

Çevre güvenlik yazılımları, bir kurumun dış ya da iç ortamdan gelen tehditlere karşı savunmasını güçlendiren yazılımlar bütünüdür. Güvenlik bilgi ve olay yönetimi (SIEM), Ağ güvenliği (NAC), uç nokta güvenliği (EDR), virüs taraması (AV), güvenlik duvarları (Firewall), e-posta filtreleme sistemleri (Mail Gateway) ve veri kaybı önleme (DLP) yazılımları gibi araçlar, kurumların siber güvenlik stratejilerinin temel bileşenlerini oluşturur. Bu yazılımlar, güvenlik tehditlerini algılayıp izler, veri akışını denetler ve herhangi bir anormal davranışa karşı uyarılar verir.

Ancak çevre güvenlik yazılımları tek başına yeterli olmayabilir. Bu yazılımlar genelde birbirinden bağımsız olarak çalıştığı için siber güvenlik ekibi tarafından manuel müdahalelerle saldırının anlamlı hale getirilmesi gereklidir. Bu güvenlik yazılımları SOAR teknolojileriyle entegre edildiklerinde veriler bir noktada toplanarak analiz edilir. SOAR platformu ile bu veriler işlenerek güvenlik olaylarına daha hızlı ve etkin bir yanıt verilmesi sağlanır. Anormal bir etkinlik algılandığında ise SAOR üzerindeki güvenlik prosedürleri devreye girerek manuel ya da otomatik aksiyonlarla olay döngüsü hızlı bir şekilde sonlandırılmış olur.

### Çevre Güvenlik Yazılımlarının SOAR'a Katkıları ve Siber Olay Yönetimi

Çevre güvenlik yazılımlarının SOAR'a sağladığı en önemli katkılardan biri, saldırı yüzeyinin genişletilmiş ve çeşitlendirilmiş olmasıdır. Çevre güvenlik yazılımları, farklı güvenlik tehditlerine odaklanır ve her biri belirli bir güvenlik katmanını korur. Bu yazılımlar, SOAR platformuna, her katmanda tespit edilen tehditlere dair veri akışı sunar. Bu veriler, SOAR tarafından işlenerek daha doğru bir tehdit değerlendirmesi yapılmasını sağlar.

Örneğin, bir e-posta güvenlik yazılımı oltalama (phishing) saldırılarını tespit edebilirken, bir uç nokta güvenliği yazılımı zararlı yazılımların sisteme girmesini engelleyebilir.

SOAR, farklı kaynaklardan gelen bu bilgileri birleştirerek saldırıyı daha kapsamlı bir şekilde değerlendirir ve bu sayede olayların daha verimli bir şekilde ele alınması sağlanmış olur.

### Sonuç ve Değerlendirme

SOAR ve çevre güvenlik yazılımları, bir arada kullanıldıklarında kurumların güvenlik stratejilerini önemli ölçüde güçlendirir. Çevre güvenlik yazılımları, tehditlerin tespit edilmesinde ve önlenmesinde önemli bir rol oynarken, SOAR sistemleri bu tehditlere karşı hızlı ve otomatik yanıtlar geliştirir.

Sonuç olarak, oluşturulan bu bütünleşik yapı sayesinde, kurumsal BT yapılarına hem zaman kazandırılır hem de bu yapı üzerinde insan hatası riski azaltılmış olur. Kurumlar, bu bütünleşik yapı sayesinde güvenlik teknolojilerini uyumlu bir şekilde kullanarak, güvenlik operasyonlarını daha verimli ve etkili hale getirebilirler.

## 2. Bybit Hack Vakası: Tarihin En Büyük Kripto Para Hırsızlığı

### Saldırının Gelişimi

2025'in Şubat ayında kripto dünyası, Dubai merkezli önde gelen kripto para borsası Bybit'e yapılan büyük çaplı siber saldırıyla sarsıldı. Yaklaşık 1,5 milyar dolar değerindeki 400.000'e yakın Ethereum token'ının çalındığı bu saldırı, tarihin en büyük kripto para hırsızlığı olarak kayıtlara geçti[1].

21 Şubat 2025'te Bybit, platformunda yapılan rutin bir işlem olan, soğuk cüzdandan (offline depolama) sıcak cüzdana (online depolama) gerçekleştirilen bir transfer sırasında bir güvenlik açığının kötüye kullanıldığını tespit etti. Saldırganlar, bu süreci manipüle ederek fonları kendi adreslerine yönlendirdi. Yapılan saldırının ardından Bybit CEO'su Ben Zhou, platformun güvenliğini sağlamak için derhal harekete geçtiklerini ve kullanıcıları bilgilendirdiklerini açıkladı[2].

### Saldırının Arkasındaki Grup: Lazarus

ABD Federal Soruşturma Bürosuna (Federal Bureau of Investigation -FBI) göre, olayın ardından başlatılan soruşturmalar, saldırının Kuzey Kore destekli ünlü hacker grubu TraderTraitor olarak da bilinen Lazarus tarafından gerçekleştirildiğini ortaya koydu[3]. APT38 olarak da bilinen Lazarus Grubu daha önce de Sony Pictures saldırısı, çeşitli bankalara yönelik SWIFT hırsızlıkları ve farklı kripto para borsalarına yapılan saldırılarla adını duyurmuş bir Gelişmiş Kalıcı Tehdit (APT) oluşumudur. FBI tarafından yürütülen araştırmalar, çalınan Ethereum'ların hızla Bitcoin ve diğer dijital varlıklara dönüştürüldüğünü ve binlerce farklı blockchain adresine dağıtıldığını gösterdi. Bu yöntem, Lazarus Grubu'nun çalıntı fonların kaynağını gizlemek için kullandığı tipik bir strateji olarak biliniyor.

Saldırganlar çalınan fonları özellikle Tornado Cash gibi gizlilik sağlayan mixer platformları aracılığıyla aklamaya çalıştı. Ancak blockchain analiz şirketleri, bu fonların hareketlerini takip ederek borsalar ve kolluk kuvvetlerine bildirdi. Birçok büyük borsa, bu adresleri kara listeye alarak şüpheli işlemleri dondurdu[1].

### Kuzey Kore Rejimi ve Siber Suçlar

Blockchain istihbarat şirketi TRM Labs'e göre, Kuzey Koreli hacker'lar 2017'den bu yana toplam 5 milyar dolardan fazla kripto para çaldı. Bu saldırılar genellikle kripto para piyasalarındaki regülasyon açıklarını hedef alarak büyük ölçekli fon aktarımlarını mümkün kılıyor. TRM Labs ayrıca, Lazarus Grubu'nun çalınan fonları aklamak amacıyla giderek daha sofistike yöntemler geliştirdiğini ve farklı blockchain platformlarında izlerini kaybettirmek için DeFi protokollerini de kullandığını belirtiyor[1].

### Bybit'in Yanıtı ve Kripto Para Sektörüne Etkileri

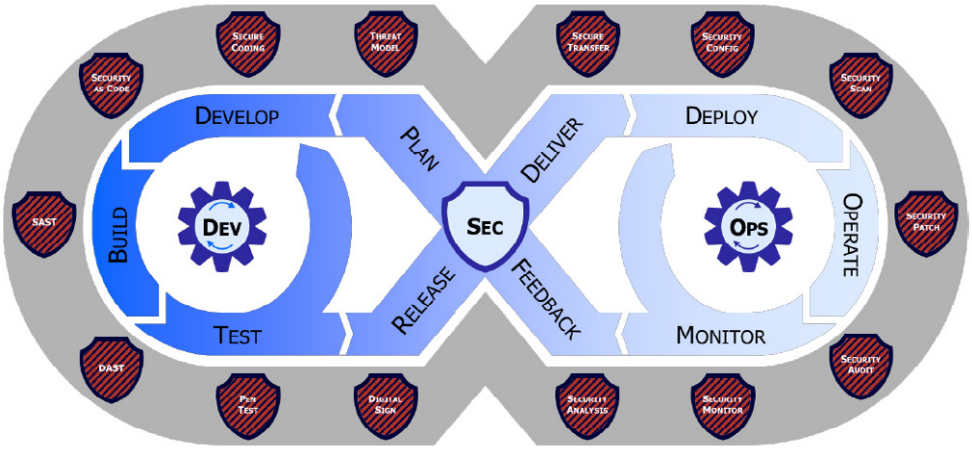
Bybit, saldırının hemen ardından platformunun güvenliğini artırmaya yönelik önlemler aldı ve etkilenen kullanıcıların zararlarını tamamen tazmin edeceğini duyurdu. Ayrıca, çalınan fonların geri alınmasına yardımcı olacak bilgiler sağlayanlara 140 milyon dolarlık ödül teklif etti. Bu süreçte Bybit, müşterilerine hesaplarının güvende olduğunu göstermek amacıyla, ek güvenlik doğrulamaları ve platformdaki çekim işlemleri için daha sıkı kontroller uygulamaya başladı[4].

Bu olay, kripto para dünyasında güvenlik önlemlerinin artırılması gerektiği konusunda yeni bir tartışma başlattı. Uzmanlar, çoklu imza gerektiren cüzdanlar, düzenli güvenlik denetimleri ve kullanıcı farkındalığını artırmaya yönelik eğitimlerin yaygınlaştırılması gerektiğine dikkat çekiyor. Aynı zamanda, borsaların soğuk cüzdan kullanımlarını daha güvenli hale getirmek ve sistemlerini sıkılaştırmak için yeni protokoller geliştirmesi öneriliyor.

### Kripto Dünyasında Güvenlik

Bybit'in hack'lenmesi, dünya genelinde düzenleyicilerin ve kolluk kuvvetlerinin dikkatini kripto para borsalarına daha fazla çevirmesine neden oldu. Daha sıkı güvenlik önlemleri ve standartlaştırılmış düzenlemeler konusunda çağrılar artarken, uluslararası işbirliğinin siber suçlarla mücadelede hayati bir rol oynayacağı belirtiliyor.

FBI ve diğer kolluk kuvvetleri, çalınan fonların izini sürmeye devam ederken, uluslararası finans kurumlarıyla işbirliği yaparak hacker'ların platformlara erişimini engellemeye çalışıyor. Kripto endüstrisinin geleceği için bu tür saldırılara karşı daha güçlü önlemler alınması gerekecek[5].




**Şekil 1:** DevSecOps süreci[6].

## 3. DEVSECOPS: Yazılım Güvenliğinin Devamlı Entegrasyonu

Günümüzde yazılım geliştirme süreçleri hızla değişmekte ve bu değişimle birlikte güvenlik anlayışı da evrilmektedir. Mikro servisler, konteynerler ve bulut teknolojilerine geçiş, siber güvenliğe yönelik yeni bir yaklaşımı zorunlu kılmaktadır. DevOps'un yükselişiyle birlikte, yazılım geliştirme ve operasyon süreçlerinin daha verimli hale gelmesi sağlanmışken, bu süreçlerin güvenlik unsurlarıyla da uyumlu hale getirilmesi gerektiği fark edilmiştir. İşte bu noktada DevSecOps devreye girmektedir. DevSecOps farkı, güvenlik ve fonksiyonel yeteneklerin, yaşam döngüsünün her adımında inşa edilmesi, test edilmesi ve izlenmesiyle tam olarak gerçekleştirilir; böylece güvenlik ve fonksiyonel sorunların üretime ulaşması önlenmiş olur. DevSecOps sürecinin genel akışı Şekil 1'de gösterilmektedir.

### DevSecOps Nedir?

**DevSecOps** (Development, Security, and Operations), yazılım geliştirme, güvenlik ve operasyon ekiplerinin bir arada çalışmasını sağlayan, güvenlik uygulamalarının sürekli entegrasyonuyla güvenli yazılım geliştirmeyi hedefleyen bir yaklaşımdır. Geleneksel yazılım geliştirme süreçlerinde güvenlik, genellikle son aşamada kontrol edilen bir unsurken, DevSecOps ile yazılım geliştirme yaşam döngüsünün her aşamasına entegre edilir.

DevSecOps'un temel amacı, güvenliği, yazılım geliştirme sürecine dahil etmek ve güvenlik tehditlerini daha erken tespit ederek minimize etmektir. Böylece yazılımın üretime geçmeden önce güvenliği sağlanmış olur.

### DevSecOps'un Temel Prensipleri

DevSecOps, güvenliği otomatikleştirmek ve her aşamada dikkate almak üzerine kurulu bir yaklaşımdır. Temel prensipleri şu şekilde sıralanabilir:

1. **Güvenli Kod Yazımı**: Yazılım geliştiriciler, güvenlik açıklarını önceden engellemek için güvenli yazılım geliştirme tekniklerini öğrenmeli ve kullanmalıdır. Bu teknikler, şifreleme, kimlik doğrulama ve hata ayıklama yöntemlerini kapsar.
2. **Sürekli Güvenlik İzleme**: DevSecOps, güvenlik tehditlerini her aşamada izler ve tespit eder. Sürekli Entegrasyon ve Sürekli Teslimat (CI/CD) süreçleri, güvenlik testlerinin otomatik olarak yapılmasını sağlar.
3. **Otomatik Güvenlik Testleri**: Güvenlik testleri, manuel süreçlerden ziyade otomatik hale getirilerek yazılımın her güncel sürümünde güvenlik açığı taramaları yapılır. Bu testler genellikle statik analiz, dinamik analiz ve güvenlik tarama araçlarını içerir.
4. **Ekip İşbirliği ve Eğitim**: Güvenlik, sadece güvenlik ekibinin sorumluluğunda değildir. Tüm ekip üyeleri (geliştiriciler, operasyon ekipleri ve güvenlik uzmanları) birlikte çalışmalı ve güvenlik konusunda eğitilmelidir.
5. **Güvenlik Zafiyetlerinin Hızlı Tespiti ve Müdahalesi**: Potansiyel güvenlik açıkları ve zafiyetler hızlıca tespit edilmeli, böylece daha büyük sorunlara yol açmadan müdahale edilmelidir. Bu süreç, doğru araçlar ve izleme sistemleriyle desteklenmelidir.

### DevSecOps Yaşam Döngüsü Aşamaları

DevSecOps'un sürekli bir süreç döngüsü vardır ve bu döngü, güvenlik önlemlerinin yazılım geliştirme yaşam döngüsüne sürekli olarak entegre edilmesini sağlayan bir modeldir. Bu döngü, yazılımın her aşamasında güvenliği sürekli olarak sağlamak için geribildirim ve iyileştirme süreçlerini içerir. DevSecOps'un bu döngüsünü aşağıdaki gibi açıklayabiliriz:

#### *Planlama (Plan)*

DevSecOps döngüsü, yazılım geliştirme sürecinin başında, planlama aşamasında başlar. Bu aşamada, yazılımın

gereksinimleri belirlenir ve güvenlik gereksinimleri de göz önünde bulundurulur. Plan aşaması, ekibin zaman, maliyet, kalite, risk ve sorunları yönetmesine yardımcı olan faaliyetleri içerir.

### *Geliştirme (Develop)*

Geliştirme aşamasında, yazılım geliştirme ekibi, belirlenen gereksinimlere ve güvenlik önlemlerine uygun şekilde yazılımı kodlar. Bu süreçte, güvenlik önlemleri (örneğin, güvenlik kod incelemeleri, statik güvenlik analiz araçları kullanma) sürekli olarak entegre edilir. Geliştiriciler, güvenlik açıklarını en erken aşamalarda tespit etmek için otomatikleştirilmiş güvenlik testleri uygularlar.

### *Sürekli Entegrasyon ve Sürekli Teslimat (CI/CD) – Build & Test*

Yazılım geliştirme sürecinde, her kod değişikliği otomatik olarak test edilir ve entegre edilir. Bu aşamada, güvenlik testleri de otomatikleştirilir. Kodun her yeni sürümü, güvenlik testlerine tabi tutulur ve her hatalı kod değişikliği hızlıca tespit edilip geri alınır. Sürekli entegrasyon (CI) ve sürekli teslimat (CD) ile boru hatlarında (pipeline) güvenlik taramaları otomatikleştirilmiştir.

Statik Uygulama Güvenlik Testi (SAST): Kaynak kodu analiz edilir.

Dinamik Uygulama Güvenlik Testi (DAST): Çalışan uygulama üzerinde güvenlik testleri yapılır.

Bağımlılık Yönetimi: Kullanılan üçüncü parti kütüphaneler ve araçlar kontrol edilir.

### *Yayınlama ve Dağıtım (Release & Deploy)*

Yazılımın üretim ortamına dağıtılması aşamasında; tüm güvenlik testlerinden geçmiş ve onaylanmış yazılım, kullanıcılarla buluşturulmak üzere canlıya alınır. Ancak bu aşama, yalnızca yazılımın güvenli olduğuna dair sağlam bir güvence verildikten sonra gerçekleşir. Yine de bu aşamada, herhangi bir güvenlik açığının gözden kaçmış olma ihtimali göz önünde bulundurularak, sistemler ve araçlar üzerinden sürekli izleme yapılır.

### *İzleme ve Geribildirim (Monitor & Feedback)*

Yazılım canlıya alındıktan sonra, güvenlik açısından sürekli izleme yapılır. Bu aşamada, güvenlik açıkları, saldırılar veya anormal etkinlikler tespit edilir. Güvenlik duvarları, izleme araçları ve log yönetimi gibi çözümler kullanılarak, yazılımın güvenliği sürekli olarak izlenir. Herhangi bir güvenlik ihlali veya açık tespit edildiğinde, sistem anında geribildirim sağlar ve bu bilgi, geliştirme ekibine iletilir.

### *İyileştirme (Improve)*

Geribildirim aldıktan sonra, yazılım geliştirme ekibi, tespit edilen güvenlik açıklarını düzeltir ve iyileştirmeler yapar. Hatalar hızlı bir şekilde düzeltilir, yeni güvenlik önlemleri eklenir ve sürekli geliştirme yapılır. Bu aşama, yazılımın güvenliğini artırmak için yeni stratejilerin ve önlemlerin geliştirilmesi sürecidir. Bu döngü, DevSecOps'un sürekli iyileştirme felsefesini yansıtır.

### *Yineleme (Iterate)*

Bu süreç, her yeni yazılım sürümünde devam eder. Geliştirilen yazılımın güvenliği sürekli izlenir, yeni tehditler ve zafiyetler dikkate alınarak yazılımda iyileştirmeler yapılır. Bu, bir geribildirim döngüsü olup, her aşamanın güvenlik açısından değerlendirilmesi ve sürekli geliştirilmesi gerektiğini ifade eder.

### DevSecOps ve Geleneksel Güvenlik Yaklaşımları Arasındaki Farklar

DevSecOps, geleneksel güvenlik yaklaşımlarından farklıdır çünkü:

- Geleneksel güvenlikte, güvenlik yalnızca yazılımın son aşamasında, üretime girmeden önce dikkate alınır. DevSecOps'ta ise güvenlik her aşamada yer alır ve yazılımın her güncel sürümünde otomatik olarak kontrol edilir.
- Geleneksel yöntemlerde güvenlik, çoğu zaman ayrı bir departman tarafından sağlanır. DevSecOps ise tüm yazılım geliştirme ekiplerinin güvenlik konusunda sorumluluk üstlenmesini gerektirir.
- DevSecOps, yazılım geliştirme ve operasyon süreçlerinin hızlı ve güvenli bir şekilde ilerlemesini sağlayarak güvenlik açıklarının erken tespitini sağlar. Geleneksel güvenlik süreçlerinde ise açıklar daha geç fark edilir ve bu da daha yüksek risklere yol açabilir.

### DevSecOps'un Faydaları

1. **Erken Güvenlik Tespiti**: Yazılım geliştirme sürecinde güvenlik tehditlerinin erken tespit edilmesi, potansiyel riskleri ve maliyetleri azaltır.
2. **Hızlı Yanıt ve Düzeltme**: Güvenlik açıkları hızlı bir şekilde tespit edilip düzeltilerek yazılımın üretime alınması daha güvenli hale gelir.
3. **Daha Az Hata**: Otomatik testler ve güvenlik kontrol mekanizmaları sayesinde insan hatası riski azalır ve güvenlik seviyeleri artırılır.
4. **Sürekli Güvenlik Güncellemeleri**: DevSecOps, yazılım geliştirme sürecine sürekli güvenlik güncellemeleri ve yamaları entegre etme imkânı verir, bu sayede güvenlik tehditlerine karşı daha dirençli bir yazılım elde edilir.

5. **Daha Az İnsani Hata**: Otomatik güvenlik testleri ve sürekli entegrasyon süreçleri, insan hatasını azaltır ve manuel müdahale gereksinimini minimuma indirir.
6. **Daha Hızlı Yazılım Teslimatı:** DevSecOps, yazılım geliştirme ve güvenlik süreçlerini entegre ederek, yazılımın daha hızlı ve güvenli bir şekilde teslim edilmesini sağlar.
7. **Uyum ve Regülasyonlar:** DevSecOps, güvenlik standartları ve düzenlemelere uyum sağlamayı kolaylaştırır. Güvenlik kontrolleri sürekli olarak izlenebilir, raporlanabilir ve denetlenebilir.

### DevSecOps'un Zorlukları

DevSecOps, birçok avantajı beraberinde getiriyor olsa da bazı zorlukları vardır:

- **Kültürel Değişim**: DevSecOps, organizasyonel kültürde değişiklikler gerektirir. Güvenliği sadece güvenlik ekibinden beklemek yerine, tüm ekiplerin bu sorumluluğa sahip olması gerektiği bilinci oluşturulmalıdır.
- **Eğitim ve Kaynaklar**: DevSecOps uygulamaları, tüm ekip üyelerinin güvenlik hakkında bilgi sahibi olmasını gerektirir. Bu da ek eğitimler ve kaynaklar gerektirir.
- **Araç Entegrasyonu**: DevSecOps süreçlerinde kullanılan araç ve teknolojilerin entegrasyonu bazen karmaşık olabilir. Bu, doğru araç seçimi ve yapılandırma gereksinimlerini ortaya çıkarır.
- **Zaman ve Kaynak Yönetimi**: Güvenlik testlerinin sürekli hale getirilmesi zaman alıcı olabilir. Bu nedenle uygun zaman ve kaynak yönetimi yapılmalıdır.

### Sonuç

DevSecOps, güvenliği yazılım geliştirme yaşam döngüsüne entegre etmek için güçlü bir yaklaşım sunar. Hem güvenlik hem de operasyonel verimlilik için kritik bir model olan DevSecOps, modern yazılım geliştirme dünyasında giderek daha fazla benimsenmektedir. Ancak, başarılı bir DevSecOps uygulaması için doğru araçlar, eğitim ve ekipler arası işbirliği büyük önem taşımaktadır. Bu süreçlerin etkili bir şekilde yönetilmesi, güvenlik tehditlerine karşı yazılımların daha dirençli olmasını sağlar.

## 4. İnsan Faktörü Siber Tehditleri Nasıl Artırıyor?

Siber güvenlik, sadece teknolojik önlemlerden ibaret değildir; aynı zamanda insan faktörü de büyük bir rol oynar. En gelişmiş güvenlik sistemlerine sahip olsanız bile, kullanıcı hataları, dikkatsizlik veya bilinçsizlik nedeniyle büyük güvenlik açıkları oluşabilir. Siber saldırganlar, teknik güvenlik önlemlerini atlatmak yerine, insan zaaflarını istismar ederek sistemlere sızmayı tercih eder.

İnsanlar, siber güvenlik zincirinin en zayıf halkası olarak kabul edilir. Kullanıcıların bilinçsiz veya hatalı davranışları, güvenlik mekanizmalarını aşmak için saldırganlar


**Şekil 2:** Siber saldırılarda insan faktörü.

tarafından aktif olarak kullanılmaktadır. Siber Saldırılarda İnsan Faktörü Şekil 2'de görselleştirilmiştir. Buna göre, insan faktörünün siber tehditleri artırdığı bazı temel noktalar şunlardır:

- **Dikkatsizlik ve Bilinçsiz Kullanım:** Çalışanlar veya bireyler, güvenlik protokollerine uymadığında sistemler açık hale gelir.
- **Zayıf Parola Kullanımı:** Kolay tahmin edilebilen veya tekrar kullanılan parolalar, hesapların ele geçirilmesini kolaylaştırır.
- **Kimlik Avı (Phishing) Saldırılarına Duyarlılık:** Kullanıcıların sahte e-postalara veya web sitelerine inanması, kimlik bilgilerinin çalınmasına neden olabilir.
- **Sosyal Mühendislik Saldırıları:** İnsan psikolojisini hedef alan saldırılar, kullanıcılardan hassas bilgilerin elde edilmesine olanak tanır.
- **Güvenlik Farkındalığının Düşüklüğü:** Çoğu kullanıcı, tehditleri algılayamaz ve kötü amaçlı yazılım bulaşmasına sebep olabilir.

### İnsan Kaynaklı En Yaygın Siber Güvenlik Tehditleri

#### *Zayıf Parolalar ve Kimlik Bilgisi Hırsızlığı*

Birçok kullanıcı, "123456", "password" gibi tahmin edilmesi kolay parolalar kullanmaktadır. Ayrıca, aynı parolanın farklı platformlarda da kullanılması, bir hesabın ele geçirilmesi durumunda diğer hesapların da tehlikeye girmesine neden olur. Bu tür tehditleri önlemek için:

- Çok faktörlü kimlik doğrulama (MFA) kullanılmalıdır.
- Parola yöneticileri ile güçlü ve benzersiz şifreler oluşturulmalıdır.
- Parola değiştirme politikaları düzenli olarak uygulanmalıdır.

#### *Kimlik Avı (Phishing) ve Sosyal Mühendislik Saldırıları*

Kimlik avı saldırıları, sahte e-postalar, mesajlar veya web siteleri kullanılarak kullanıcıları kandırma yöntemidir. Bu tür saldırıları engellemek için:

- Çalışanlara düzenli olarak siber güvenlik farkındalık eğitimleri verilmelidir.
- E-posta güvenlik çözümleri (DMARC, SPF, DKIM) aktif hale getirilmelidir.
- Kullanıcılara, bağlantılara tıklamadan önce doğrulama yapmaları gerektiği öğretilmelidir.

#### *USB ve Taşınabilir Medya Kullanımı*

Zararlı yazılım içeren USB bellekler, birçok organizasyonda ciddi güvenlik tehditleri oluşturur. Bu riskin önüne geçebilmek için:

- USB bellek kullanımına dair politikalar belirlenmelidir.
- Çalışanlar, bilinmeyen USB bellekleri bilgisayarlara takmamaları konusunda eğitilmelidir.
- Bilinmeyen cihazların sisteme bağlanmasını engelleyen güvenlik çözümleri uygulanmalıdır.

#### *Çalışanların Yetki Aşımı ve İç Tehditler*

Bazı durumlarda, şirket içinden bir çalışan, kötü niyetli veya bilinçsiz hareketlerle kurumun sistemlerine zarar verebilir. Bu tür iç tehditleri önlemek için:

- Zero Trust modeli benimsenerek her erişim için kimlik doğrulama yapılmalıdır.
- Erişim yönetimi ve kullanıcı hakları düzenli olarak denetlenmelidir.

### İnsan Kaynaklı Siber Tehditleri Azaltmak İçin En İyi Uygulamalar

1. Düzenli güvenlik farkındalığı eğitimleri verilmelidir.
2. Çalışanlar saldırı simülasyonları ile test edilmelidir.
3. Zero Trust modeli uygulanarak erişim kontrolü sıkılaştırılmalıdır.

## 5. Aldatma Teknolojileri (Deception Technologies)

Aldatma Teknolojilerini (Deception Technologies) detaylı şekilde incelemeye çalışacağız.

Günümüz siber güvenlik dünyasında, yapay zekâ teknolojileri, Siber Silah (Offensive Cyber Weapons) olarak kullanılacak uygulamaların oluşturulmasında kaldıraç olarak etkin bir rol oynayacaktır. Bu silahlara karşı koymak için yine yapay zekâ kullanılarak oluşturulan savunma ve erken uyarı sistem ve teknolojileri ortaya çıkmaktadır Bununla ilgili olarak, Ekim-Aralık 2024 Siber Tehdit Raporunda[7] "Hareket Eden, Değişen Hedefler -Automated Moving Target Defence (AMTD)" başlığı altında Aldatma Tekniklerinin (Deception), AMTD uygulamalarının kullandığı temel bileşenlerden biri olduğuna değinmiştik.

Aldatma Teknolojileri, ağ ve sistemler üzerinde saldırganların önceden tespit edilmesini, sisteme erişme çabalarını boşa çıkarmayı ve kullanılan tekniklerle ilgili bilgi toplamayı amaçlayan (Tactics, Techniques and Procedures -TTPs), bunun için ağ üzerinde tuzakların (decoy) ve gerçek gibi görünen ancak doğru olmayan bilgilerin (misinformation) oluşturulmasını ve bunlar üzerindeki erişimlerin, hareketlerin sürekli izlenmesini, yetkisiz erişimler olduğunda organizasyonun uyarılmasını sağlayan teknolojilerdir. Bu teknolojiler kurgulanırken MITRE ATT&CK®[8],[9] tarafından, gerçek dünyada gerçekleşen siber saldırılarda saldırganların kullandığı taktik ve tekniklerin gözlemlenmesiyle oluşturulan herkese açık bir

veritabanı kullanılmaktadır. Organizasyonların siber güvenlik bölümleri, siber güvenlik uzmanları, üreticiler ve karar vericiler için; saldırganlarla etkileşimi, aldatma ve erişim engelleme (denial) faaliyetlerinin nasıl planlanıp yürütüleceğini belirleyen MITRE Engage™[10] adı verilen çerçeve MITRE tarafından yayınlanmıştır.

Bu bilgilerden sonra Aldatma Teknolojileri ve Sistemlerini açıklayınca, hemen akla Bal Küpü sistemleri gelecektir. Hatta bu teknolojilerin geleneksel Bal Küpü sistemlerinden farkının ne olduğu sıklıkla merak edilmektedir. Aldatma Teknolojilerinin öne çıkan farkını;

> *"Gerçek ağ ve sistem varlıklarının (kullanıcı, hafıza alanı, token, dosya, registry girdileri, çerez, servis, ağ iletişimi, sunucu, ağ anahtarı gibi) taklit edilmesi (mimic) ile kurumsal ağların içinde dinamik bir ortamın yaratılması, bu varlıklara olan erişimin izlenmesi, uyarı ve alarmların üretilmesi, kullanılan teknoloji ve sisteme göre bunlara erişim gerçekleştiğinde müdahale edilerek daha fazla ilerlenmesinin engellenmesi ve detaylı iz kayıtlarının tutulması"*

olarak belirtebiliriz. Günümüzde Aldatma Teknolojileri ve Sistemleri, Balküpü sistemlerini de kullanan ve kapsayan, daha geniş bir tanım, ürün, teknoloji ve uygulama başlığıdır[11].

Bu teknoloji ve sistemler son dönemde dikkat çekmeye başlamış ve giderek daha fazla sayıda firma bu alanda çeşitli ürünler piyasaya çıkarmaya başlamıştır. Ekim 2024 tarihinde Gartner tarafından yayınlanan, "Emerging Tech: Build Preemptive Security Solutions to Improve Threat Detection (Part 1)[12]" başlıklı rapor, bu teknoloji ve sistemlerin ortaya çıkmasının nedenini şöyle açıklamaktadır: "Tehdit aktörleri tarafından yapay zekânın giderek silahlandırılması, hâlihazırda geride kalan tespit ve yanıt stratejileri üzerindeki baskıyı attırmakta, siber güvenlik firmalarının ürünlerine önleyici siber savunma yeteneklerini dahil etmesi gerekmektedir. Aksi takdirde bu firmalar, müşterilerinin gelecekteki tehditlerle mücadele etmesi konusunda başarısız olacaktır"

Yapay zekâ ve makine öğrenmesinin sağladığı hızlı işlem yapma ve öğrenme olanaklarıyla siber saldırıların daha karmaşık, hızlı ve organize hale geldiği, bu nedenle "Önleyici Siber Güvenlik" kabiliyetlerinin artırılması ihtiyacı ortaya çıkmaktadır. Yapay zekâ ile donatılan siber saldırılar için kullanılan araç ve uygulamamaların devlet destekli ve bağımsız kişi ve grupları, kısaca saldırganları daha yetenekli ve hızlı hale getirdiğini söylemek yanlış olmayacaktır.

Ek olarak, Aldatma (Deception), "NIST 800-160 Volume 2[13]" dokümanında, "yanlış yönlendirmek, karıştırmak, kritik varlıkları saldırganlardan gizlemek ya da gizlice kirletilmiş varlıkların saldırganlara ifşa edilmesi olarak tanımlanmaktadır. Aldatma Teknolojilerinin kullanılması, "A Decade of Security Assessments: Security Issues That Refuse to Die"[14] makalesinde önerilmektedir.

Organizasyonlarda siber tehditlere karşı önlem almak, siber tehditleri tespit etmek ve tespit edilen siber tehditlere müdahale etmek için imza tabanlı tespit, davranış analizi, makine öğrenmesi ve kum havuzu gibi metot ve teknolojiler kullanılmaktadır. Dolayısıyla saldırganları tespit edebilmek için sistemler sürekli izlenmektedir. Proaktif bir yaklaşımla Aldatma Teknolojilerini kullanarak ağda varlığını sürdüren ve henüz tespit edilmemiş saldırganların, ağ ve sistem varlıkları olarak oluşturulmuş tuzaklara eriştiklerinde tespit edilmeleri ve bunlara karşı gereken müdahalelerin yapılması için ilgili birim ve sistemlerin uyarılması gereksiniminin de ortaya çıktığını söyleyebiliriz.

Aldatma Teknolojileri sadece Bilgi Teknolojilerinde (BT) değil, aynı zamanda Operasyonel Teknolojilerde (OT), IoT ağlarında ve endüstriyel kontrol sistemlerinde de kurgulanmakta ve bu alanda firmalar ürün geliştirmektedirler.

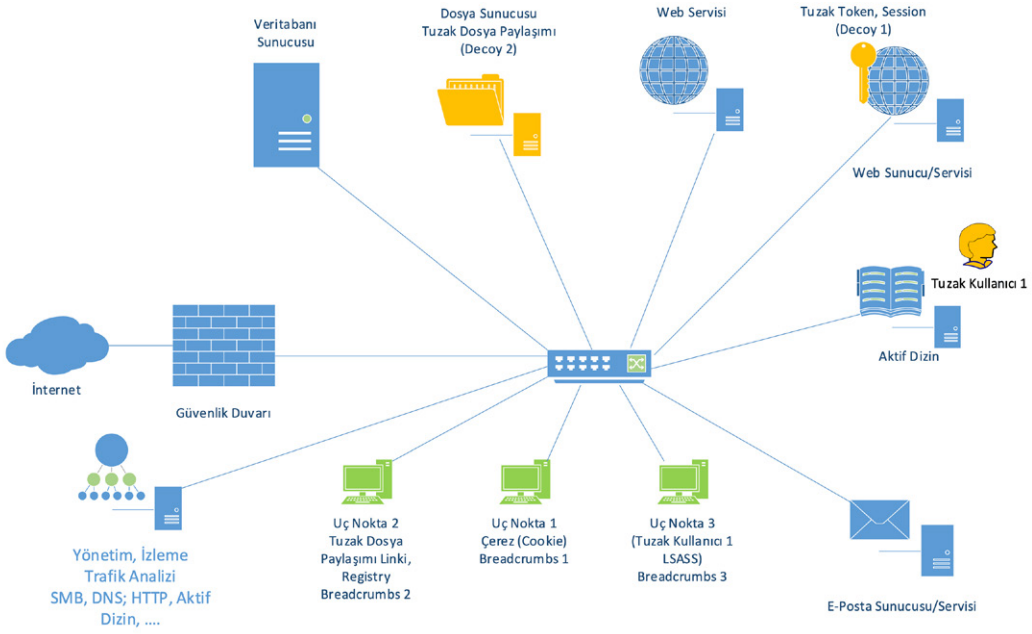
Uygulamanın hafızadaki alanlarının gizlenmesi ve bu alanların yerine sürekli izlenen tuzak alanların gösterilmesi, bir sunucudaki dosyaların sanki gerçekmiş gibi oluşturulması ve izinlerinin tuzak kullanıcılarla konfigüre edilmesi, gerçek bir varlığa ait olmayan kimlik bilgilerinin oluşturulması ve bu kullanıcı ya da bir servise ait bilgilerle oturum açılması, sunucu üzerinde yanıltıcı ağ trafiği, tuzak RDP oturumları, veritabanı bağlantıları, e-posta, script dosyaları, çerezler bu teknoloji ya da yöntemleri uygulayan sistemler tarafından ağda oluşturulur. Aldatma Teknolojileri ve Sistemlerinin örnek bir kurgusu Şekil 3'te görülebilir.

Örnek bir senaryoda, tuzak olarak hazırlanan bir kullanıcı kimliğinin (Decoy), LSASS işlemi ile hafızaya yüklenmesi (breadcrumbs, hedeflerin yerlerine dair ipuçları), saldırganların bu kullanıcının parolasını elde etmeleri (cezbetmek -lure) ve bu kullanıcı ile diğer sistem ve servislere erişmeyi denemeleriyle ilgili sistem ve kişilere bildirim yapılır. Eşzamanlı detaylı iz kayıtları alınır ve olay müdahale süreçleri başlar.

Tuzak olarak ağ anahtarı, güvenlik duvarı, ağ yönlendiricileri (router), yazıcı, SIP servisi, IoT cihazı/servisi, PLC, RTU, SSH, Telnet, SMB gibi davranan servisler de taklit edilerek kurumsal ağlarda Aldatma Teknolojileri, üreticisinin yeteneklerine göre oluşturulabilir ve izlenir.

Organizasyonlar, sistemlerini sürekli izleyerek saldırganları tespit etmeye çalışmaktadır. Ancak yapay zekâ desteğiyle daha yetenekli hale gelen saldırganlara karşı, organizasyonların da siber güvenlik uygulamalarında proaktif davranarak ve Sıfırıncı Gün (Zero Day) ve gelişmiş saldırıları gerçek zamanlı olarak tespit edebilmesi gerekmektedir. Önümüzdeki dönemde, aldatma teknolojilerinin kurumsal ağlarda yaygınlaşmasıyla birlikte, organizasyonların sistemleri izlemenin yanında, saldırganları açığa çıkarmak için bu teknolojileri de kullanmaları gerekecektir.

Aldatma Teknolojileri ve Sistemleri alanında önde gelen üreticiler şunlardır: Acalvio, Aves, Bytes, CounterCraft, Deceptive, DefusedCyber, Fortinet, Fidelis Security, Labrinth, Lupovis, Netsec, PacketViper, Sectrio, SentinelOne; Seedata.io, Sectrio, SentinelOne, Sigularity Identity.




**Şekil 3:** Aldatma Teknolojileri/Sistemleri örnek ağ ortamı.

## 6. Zararlı Yazılım Analizinde Fuzzy Hashing

Siber güvenlik tehditleri giderek daha karmaşık hale gelirken, zararlı yazılım analiz yöntemleri de bu dinamik tehdit ortamına ayak uydurmak zorundadır. Geleneksel kriptografik hash algoritmaları (cryptographic hash algorithms), örneğin MD5 ve SHA-256 bir dosyanın benzersiz bir özetini oluşturur. Ancak, bu algoritmaların doğası gereği, dosyada yapılan en ufak bir değişiklik bile tamamen farklı bir hash değeri üretir. Bu özellik, veri bütünlüğünü doğrulamak için mükemmel bir yöntem olsa da zararlı yazılım analistleri için bazen dezavantaj oluşturur.

Zararlı yazılım geliştiricileri, polimorfik (polymorphic) ve metamorfik (metamorphic) teknikler kullanarak her saldırıda zararlı yazılım kodlarını farklı şekilde paketleyebilir, küçük değişiklikler yaparak geleneksel hash tabanlı tespit sistemlerini atlatabilir. İşte tam da bu noktada, bulanık özetleme (fuzzy hashing) devreye girer.

Fuzzy hashing, dosyalar arasındaki benzerlik oranlarını hesaplayarak küçük değişiklikler taşısa da aynı zararlı yazılım ailesine ait varyantların (variants) tespit edilmesini sağlar. Bu teknik, yalnızca bire bir aynı dosyaları değil, ilişkili ve türetilmiş varyantları da belirleyerek, tehdit avcılığı (threat hunting), dijital adli bilişim (digital forensics) ve siber tehdit istihbaratı (cyber threat intelligence) süreçlerinde kritik bir rol oynar.

### Fuzzy Hashing Nedir?

Fuzzy hashing, dosyalar veya veri kümeleri arasındaki benzerliği ölçmeye yönelik özel bir hash tekniğidir. Geleneksel kriptografik hash algoritmalarının aksine, fuzzy hashing, bir dosyada yapılan küçük değişiklikleri tamamen farklı bir özet değeriyle sonuçlandırmaz. Bunun yerine, dosyalar arasındaki benzerlik yüzdesini hesaplayarak ilişkili içerikleri belirleyebilir. Bu özellik, kötü amaçlı yazılım analizinde varyantların ve farklı sürümlerin tespit edilmesi açısından kritik bir rol oynar.

### Geleneksel Hash Algoritmalarının Yetersiz Kaldığı Durumlar

Geleneksel kriptografik hash algoritmaları bir dosyanın içeriğini benzersiz bir özetleme fonksiyonuna tabi tutarak tekil bir hash değeri üretir. Bu algoritmalar, veri bütünlüğünü doğrulamak için oldukça güvenilirdir, ancak zararlı yazılım analizinde bazı kritik dezavantajları vardır:

- **Küçük Değişiklikler Büyük Farklar Yaratır**

  Kriptografik hash algoritmaları, bir dosyada yapılan en küçük değişiklik sonucunda bile tamamen farklı bir hash değeri üretir. Örneğin, bir zararlı yazılım geliştiricisi bir dosyanın sadece birkaç baytını değiştirdiğinde veya dosyanın yürütme akışını (execution flow) farklı hale getirdiğinde, geleneksel hash fonksiyonları bu dosyayı tamamen farklı bir nesne olarak tanımlar. Bu durum, aynı zararlı yazılım ailesine ait varyantları veya yeniden paketlenmiş sürümleri tespit etmeyi zorlaştırır.

  Bir endüstriyel kontrol sistemini (ICS) hedef alan zararlı yazılımın farklı sürümlerini tespit etmek için fuzzy hashing kullanımı, geleneksel hash'leme yöntemlerine göre daha etkili olabilir. Örneğin, bir SCADA sistemini manipüle eden bir kötü amaçlı yazılımın

ilk sürümünün SHA-256 hash değeri belirlenmişken, saldırganlar bu yazılımın tespit edilmesini zorlaştırmak için kod obfuscation (bulanıklaştırma), log temizleme mekanizmaları ekleme ve ICS protokol komutlarını değiştirme gibi yöntemlerle varyantını oluşturduğunda, SHA-256 değeri tamamen değişir ve geleneksel hash yöntemleri bu iki sürümün bağlantılı olduğunu belirleyemez. Ancak, fuzzy hashing algoritmaları dosyayı belirli bloklara bölerek entropi tabanlı fingerprint'ler (hash imzaları) oluşturur ve iki sürüm arasındaki benzerliği hesaplar. Örneğin, benzerlik skoru yüzde 92 olabilir. Bu da yeni zararlı yazılımın önceki sürümle ilişkili olduğunu gösterir ve tehdit avcılarının (threat hunters) varyantları tespit etmesini sağlar, böylece ICS güvenlik sistemleri proaktif bir şekilde güncellenebilir.

- **Polimorfik ve Metamorfik Zararlı Yazılımlara Karşı Yetersizlik**

  Polimorfik zararlı yazılımlar her çalıştırıldığında kendi kod yapısını değiştirebilen tehditlerdir. Metamorfik zararlı yazılımlar ise çalışma mantığını koruyarak kendini baştan aşağı yeniden yazabilir.

  Bu tür zararlı yazılımlar, geleneksel hash algoritmalarını kolayca geçersiz kılar çünkü her yeni sürüm, tamamen farklı bir hash değeri üretir. Fuzzy hashing ise dosya içeriğinde meydana gelen değişikliklere rağmen benzerlikleri tespit edebilir ve varyantları ilişkilendirebilir.

- **Kriptografik Amaçlarla Tasarlanmış Olmaları**

  MD5 ve SHA-256 gibi hash algoritmaları, veri bütünlüğünü sağlamak amacıyla tasarlanmıştır. Yani, bir dosyanın değişip değişmediğini doğrulamak için mükemmel araçlardır. Ancak, benzer dosyaları ilişkilendirmek veya ortak noktaları belirlemek için geliştirilmemişlerdir.

### Fuzzy Hash Algoritmaları

Günümüzde en yaygın kullanılan bazı fuzzy hashing algoritmaları şunlardır:

- **SSDEEP**

  Ssdeep algoritması, Spamsum algoritmasına dayanır ve dosyaları farklı uzunluktaki bloklara ayırarak her blok için ayrı hash değerleri üretir. Bu süreçte, kayan pencere (rolling hash) ve yerel duyarlıklı hash'leme (Locality-Sensitive Hashing -LSH) tekniklerini birleştirir. Kayan pencere yöntemi, dosyanın her bir parçasını sürekli kaydırarak ve hash değerini güncelleyerek çalışır. Bu, dosya içeriğinde yapılan küçük değişikliklere tolerans göstermesini sağlar ve benzerlikleri belirler.

  Ssdeep'in çalışma mekanizması, dosya içeriğini parçalara bölmek ve her parçayı ayrı bir hash değeri ile özetlemek üzerine kuruludur. Her bir segment, dosya içeriğinde yapılan küçük değişikliklere rağmen benzerlik oranlarını koruyacak şekilde tasarlanmıştır. Bu sayede, saldırganlar tarafından ufak değişiklikler yapılmış dosyalar dahi aynı zararlı yazılım ailesine ait olarak tespit edilebilir. Ssdeep, segmentleri karşılaştırarak benzerlik yüzdesini hesaplar ve bu veriler, güvenlik analistlerine zararlı yazılım analizinde rehberlik eder.

  Yerel duyarlıklı hash'leme (LSH) tekniği, benzer veri noktalarının aynı veya yakın hash değerlerine sahip olmasını sağlayarak verilerin benzerliklerini keşfetmeye yardımcı olur. Bu, özellikle büyük veri kümelerinde analiz yaparken performans optimizasyonu sağlar. LSH, farklı dosya segmentlerinin benzerlik oranlarını hesaplayarak dosyaların ilişkisini belirler ve bu sayede zararlı yazılım kampanyalarının ardındaki bağlantıları ortaya çıkarabilir.

  Paralel işlem ve önbellekleme stratejileri, ssdeep algoritmasının performansını artırmak için kullanılır. Büyük ölçekli analizlerde, dosya parçalarının paralel olarak işlenmesi ve geçici sonuçların önbelleğe alınması, hesaplama yükünü optimize eder. Bu yöntemler, ssdeep'in büyük veri kümeleri üzerinde daha verimli çalışmasını sağlar ve tehdit tespiti süreçlerini hızlandırır.

  **Avantajlar:**

  Ssdeep, dosya içeriğinde yapılan küçük değişikliklere tolerans gösterir ve benzerlikleri belirler. Bu, saldırganlar tarafından yapılan ufak değişikliklerin tespit edilmesini sağlar.

  Yerel duyarlıklı hash'leme (Locality-Sensitive Hashing -LSH) sayesinde ssdeep, büyük veri kümelerinde hızlı ve verimli analiz yapma imkânı sunar.

  **Dezavantajlar:**

  Ssdeep, büyük dosyalar üzerinde çalışırken yanlış pozitif oranı yüksek olabilir. Bu, tamamen farklı içeriklere sahip dosyaların benzer olarak değerlendirilmesine neden olabilir.

  Büyük ölçekli dosya kümelerinde lineer karşılaştırma gerektirdiğinden ssdeep'in işlem maliyeti yüksektir. Bu, hesaplama yükünü artırabilir ve performans sorunlarına yol açabilir.

- **TLSH (Trend Micro Locality Sensitive Hashing)**

  TLSH (Trend Micro Locality Sensitive Hashing), büyük dosyaların benzerliğini hesaplamak amacıyla geliştirilmiş bir fuzzy hashing algoritmasıdır. Bu algoritma, dosya boyutuna duyarlı çalışması ve büyük veri kümeleri üzerinde optimize edilmiş olmasıyla dikkat çeker. TLSH'nin temel amacı, dosyalar arasındaki benzerlikleri tespit ederek, zararlı yazılım analizi ve tehdit tespiti gibi alanlarda kullanılabilecek güçlü bir araç sunmaktır.

TLSH'nin çalışma prensibi, öncelikle özellik vektörü çıkarma yöntemine dayanır. Bu süreçte, dosyanın içeriği belirli özelliklere dayanarak özetlenir ve bu özellikler bir vektör halinde temsil edilir. Özellik vektörü, dosyanın belirli bölümlerinin özetlenmesiyle elde edilen bilgi parçacıklarını içerir. Bu vektörler, dosya içeriğindeki küçük değişikliklere karşı toleranslı olup, benzerliklerin tespit edilmesine yardımcı olur.

Algoritmanın bir diğer önemli bileşeni ise kademeli özetleme yöntemidir. Bu yöntem, dosyanın çeşitli bölümlerini aşamalı olarak özetleyerek çalışır. Kademeli özetleme, dosyanın her bir parçasını belirli aralıklarla özetler ve bu özetleri birleştirerek genel bir hash değeri oluşturur. Bu, dosya içeriğinde yapılan ufak değişikliklerin tespit edilmesine olanak tanır ve benzer dosyaların belirlenmesini sağlar.

TLSH'nin en kritik bileşenlerinden biri olan yerel duyarlıklı hash'leme algoritması, benzer veri noktalarının aynı veya yakın hash değerlerine sahip olmasını sağlar. LSH, dosya parçaları arasında benzerlik oranlarını hesaplayarak, dosyaların ilişkisini belirler. Bu teknik, özellikle büyük veri kümeleri üzerinde analiz yaparken performans optimizasyonu sağlar ve zararlı yazılım kampanyalarının ardındaki bağlantıları ortaya çıkarabilir.

TLSH, büyük ölçekli veri analizlerinde yüksek performans ve doğruluk sağlar. Özellik vektörü çıkarma, kademeli özetleme ve yerel duyarlıklı hash'leme tekniklerinin kombinasyonu, TLSH'yi güçlü ve etkili bir fuzzy hashing algoritması haline getirir. Bu sayede, siber güvenlik analistleri, zararlı yazılım varyantlarını ve tehdit aktörlerinin saldırı modellerini tespit ederek, proaktif savunma stratejileri geliştirebilirler.

**Avantajlar:**

TLSH algoritması, küçük değişiklikleri göz ardı ederek geniş çapta benzerlik analizi yapabilme yeteneğine sahiptir. Bu özellik, dosya içeriğindeki ufak değişikliklerin analiz edilmesini ve benzerliklerin tespit edilmesini kolaylaştırır. Ayrıca, büyük veri kümeleri üzerinde çalışırken düşük yanlış pozitif oranı ile yüksek doğruluk sağlar. Yerel duyarlıklı hash'leme tekniği, benzer veri noktalarının aynı veya yakın hash değerlerine sahip olmasını sağlayarak performans optimizasyonu sunar.

**Dezavantajlar**:

Ancak, TLSH algoritmasının küçük dosyalar üzerinde performansı, ssdeep algoritmasına kıyasla daha düşük olabilir. Kademeli özetleme yöntemi, küçük dosya boyutlarında etkin çalışmayabilir ve bu durum performans kayıplarına yol açabilir. Ayrıca, yüksek işlem gücü ve bellek tüketimi gerektirebilir, bu da büyük veri kümeleri üzerinde hesaplama maliyetini artırabilir.

- **SDHASH**

SDHASH (Similarity Digest Hashing), dosya içeriğine dayalı analiz yaparak benzerlikleri belirleyen ve adli bilişimde tercih edilen bir algoritmadır. Bu algoritma, dosya içeriklerinin belirli bölümlerini analiz ederek her bölümün entropi değerini hesaplar. Entropi, bir dosyanın veri yoğunluğunu ve rasgeleliğini ölçen bir kavramdır. Yüksek entropiye sahip bölümler, genellikle farklı veri içerir. Entropi, bir veri bloğunun rasgeleliğini ve veri yoğunluğunu ölçer. Yüksek entropiye sahip bölümler, daha karmaşık ve çeşitli verileri barındırır ve bu bölümler seçilerek hash çıktısı oluşturulur. Bu sayede, dosyanın daha bilgilendirici ve önemli kısımları temsil edilerek benzerlik analizlerinde daha hassas sonuçlar elde edilebilir.

Yüksek entropiye sahip olan bölümler hash çıktısına dahil edilir. Bu, dosya içeriğinde yapılan küçük değişikliklerin tespit edilmesini sağlar ve benzer dosyaların belirlenmesini kolaylaştırır. SDHASH, fuzzy hashing karşılaştırmalarında Jaccard benzerlik metriği kullanır. Jaccard metriği, iki veri kümesinin benzerliğini ölçen bir metrik olup, kesişim ve birleşim oranına dayanır. Bu metrik, dosya bölümlerinin benzerlik oranlarını hesaplayarak, iki dosyanın ne kadar benzer olduğunu belirler.

**Avantajlar**:

SDHASH algoritması, yüksek doğruluk oranına sahiptir ve içerik tabanlı karşılaştırmalar için oldukça uygundur. Entropi tabanlı analiz yöntemi sayesinde, dosya içeriğindeki küçük değişiklikler tespit edilebilir ve benzerlikler belirlenebilir. Bu, adli bilişimde ve zararlı yazılım analizinde güvenilir sonuçlar elde edilmesini sağlar. Ayrıca, Jaccard benzerlik metriği kullanarak, dosya bölümlerinin benzerlik oranlarını hassas bir şekilde ölçer.

**Dezavantajlar**:

Büyük dosyalar için karşılaştırma süresi daha uzun olabilir, çünkü her bölümün entropi değeri hesaplanmalı ve yüksek entropiye sahip bölümler seçilmelidir. Bu, hesaplama yükünü artırır ve performans sorunlarına yol açabilir. Ayrıca, SDHASH algoritması, ssdeep ve TLSH algoritmalarına kıyasla daha fazla işlem gücü gerektirir. Bu nedenle, büyük ölçekli veri analizlerinde ve yüksek performans gerektiren uygulamalarda dikkatli bir şekilde kullanılmalıdır.

- **LZJD (Lempel-Ziv Jaccard Distance)**

LZJD, Lempel-Ziv sıkıştırma yöntemine dayanan ve Jaccard benzerlik metriğini kullanarak dosya benzerliklerini hesaplayan bir fuzzy hashing algoritmasıdır. Bu algoritma, dosya içeriklerini analiz ederek benzerlik oranlarını belirler.

LZJD'nin temel çalışma prensibi, dosya içeriğini Lempel-Ziv tabanlı segmentasyon kullanarak analiz etmektir. Bu süreçte, dosya belirli segmentlere bölünür ve bu segmentler Lempel-Ziv sıkıştırma algoritması ile özetlenir. Bu algoritma, verinin tekrar eden desenlerini tanıyıp özetlemekte etkilidir. Ardından,

örüntü tabanlı kümeleme yöntemi kullanılarak belirli veri blokları çıkarılır ve dosya içeriklerinde benzer örüntüler tespit edilir. Bu yöntem, dosya segmentleri arasındaki benzerlik oranlarını hassas bir şekilde ölçerek daha doğru bir benzerlik analizi yapılmasını sağlar. LZJD, Jaccard benzerlik metriği kullanarak iki dosyanın benzerlik oranını hesaplar.

**Avantajlar**:

LZJD algoritması, büyük veri kümeleri üzerinde yüksek doğrulukla çalışır ve dosya içeriklerinde yapılan küçük değişiklikleri göz ardı edebilir. Bu özellikler, LZJD'yi zararlı yazılım analizi ve tehdit tespiti gibi alanlarda etkili bir araç haline getirir.

**Dezavantajlar**:

Küçük veri kümelerinde yanlış pozitif oranı yüksek olabilir, bu da farklı içeriklere sahip dosyaların benzer olarak değerlendirilmesine neden olabilir. Ayrıca, hesaplama süresi diğer yöntemlere kıyasla daha uzun olabilir, bu da büyük ölçekli veri analizlerinde performans sorunlarına yol açabilir.

### Zararlı Yazılım Analizinde Fuzzy Hashing Kullanımı

Fuzzy hashing, zararlı yazılım analizinde farklı senaryolarda kullanılan güçlü bir tekniktir. Zararlı yazılım sınıflandırma, benzerlik analizi, adli bilişim ve tehdit avcılığı gibi birçok alanda güvenlik analistlerine önemli avantajlar sunar. Fuzzy hashing, aynı zararlı yazılım ailesine ait varyantları tespit etmek için kullanılabilir. Bu varyantlar arasında benzerlik oranlarını belirleyerek aynı aileye ait örneklerin tespit edilmesine yardımcı olur.

- **Siber Tehdit İstihbaratı**

  - Güvenlik analistleri potansiyel saldırıları önceden tespit etmek, zararlı yazılım ailelerini sınıflandırmak ve tehdit aktörlerinin saldırı modellerini analiz etmek için fuzzy hashing yöntemlerini kullanır.
  - APT (Advanced Persistent Threat) grupları veya fidye yazılım kampanyaları gibi büyük ölçekli saldırıların ilişkilerini analiz etmek için bu teknik farklı bir öneme sahiptir.
  - Fuzzy hashing sayesinde, zararlı yazılım kampanyalarının ardındaki bağlantılar ortaya çıkarılabilir ve siber güvenlik önlemleri proaktif bir şekilde geliştirilebilir.

  Örnek bir fidye yazılımı analizinde güvenlik analisti, fuzzy hashing kullanarak fidye yazılımı ailesine ait farklı varyantları tespit etmek isteyebilir. Örneğin, bir saldırgan aynı fidye yazılımını farklı yollarla yayıp küçük değişiklikler yaparak tespit edilmesini zorlaştırabilir. MD5 veya SHA-256 gibi geleneksel hash'leme yöntemleri, bu dosyaları tamamen farklı olarak tanımlayacaktır. Fuzzy hashing algoritmaları (ssdeep, TLSH vb.), dosyaların benzerlik oranlarını hesaplayarak bunların benzer zararlı yazılım ailesine ait olduğunu ortaya çıkarabilir.

- **Adli Bilişim ve Tehdit Avcılığı**

  Fuzzy hashing, dijital adli analiz süreçlerinde güvenlik araştırmacıları tarafından şüpheli dosyaları gruplandırmak ve önceden tespit edilen zararlı yazılım örnekleriyle karşılaştırmak için de kullanılır.

  - Bilgisayar korsanları (hackers) tarafından değiştirilmiş zararlı yazılım örneklerinin tespit edilmesini sağlar.
  - Güvenlik araştırmacıları, büyük veri kümeleri içinde ilişkili zararlı yazılımları belirlemek için fuzzy hashing algoritmalarını kullanabilir.

  Örneğin bir saldırgan, kötü amaçlı bir belgeye küçük kod değişiklikleri ekleyerek yeni bir varyant oluşturursa, sdhash veya LZJD gibi fuzzy hashing yöntemleri dosyanın hâlâ aynı temel yapıya sahip olduğunu gösterebilir.

- **Tehdit Avcılığı Kullanımı**

  Tehdit avcıları, bilinen zararlı yazılım örneklerine benzeyen yeni tehditleri tespit etmek için fuzzy hashing ile veritabanlarını tarayabilir.

  Büyük ölçekli güvenlik operasyon merkezlerinde (Security Operations Center -SOC), tehdit istihbaratı sistemleri fuzzy hashing teknikleri kullanarak anormal ve şüpheli dosyaları filtreleyebilir.

- **APT (Advanced Persistent Threat) Analizleri**

  Sofistike siber tehdit grupları tarafından kullanılan gelişmiş zararlı yazılımları izlemek için fuzzy hashing kritik bir yöntem olabilmektedir. APT grupları, geleneksel hash'leme tekniklerinden kaçınmak için kodlarını değiştirerek saldırılarını gizler. Fuzzy hashing algoritmaları, benzerlik analizi yaparak varyantlar arasındaki bağlantıları ortaya çıkarabilir. Örneğin, bir güvenlik ekibi, önceden tespit edilen bir APT zararlı yazılımının yeni bir versiyonunu tespit etmek için TLSH kullanabilir. Saldırının aynı tehdide ait olup olmadığı belirlenebilir.

### Fuzzy Hashing'in Avantaj ve Dezavantajları

Fuzzy hashing, zararlı yazılım analizi, tehdit istihbaratı ve adli bilişim gibi alanlarda geleneksel hash'leme yöntemlerinin yetersiz kaldığı noktalarda büyük avantajlar sunar. Ancak, bu yöntemin de bazı sınırlamaları vardır. Aşağıda, fuzzy hashing'in avantaj ve dezavantajları detaylı bir şekilde ele alınmaktadır.

**Avantajlar:**

- **Varyantları Tespit Edebilir**

  Polimorfik zararlı yazılımlar, her çalıştırıldığında kendini değiştirebilir. Metamorfik zararlı yazılımlar, kodlarını tamamen yeniden yazarak analiz süreçlerini zorlaştırır. Geleneksel hash'leme yöntemleri, bu varyantları tamamen farklı olarak görürken, ssdeep, TLSH ve

sdhash gibi fuzzy hashing algoritmaları benzerlikleri belirleyerek aynı zararlı yazılım ailesine ait olup olmadığını tespit edebilir. Böylece, tehdit analistleri yeniden paketlenmiş veya küçük değişikliklerle yapılan saldırıları daha etkili bir şekilde ilişkilendirebilir.

- **Kapsamlı Analiz Sunar**

  Fuzzy hashing, zararlı yazılım varyantlarını, yeniden paketlenmiş kötü amaçlı yazılımları ve gelişmiş sürekli tehdit (APT) gruplarının saldırı yöntemlerini analiz etmek için geniş kapsamlı bir araç sunar. Fidye yazılımı ailelerinin farklı sürümlerini analiz etmekte kullanılabilir ve APT gruplarının ufak değişikliklerle yeniden paketlenmiş veya şifrelenmiş kötü amaçlı yazılımlarını tespit edebilir. Adli bilişim süreçlerinde, sahte veya değiştirilmiş belgelerin orijinal versiyonlarıyla ilişkisini ortaya çıkarmak için de kullanılabilecek olan fuzzy hashing, özellikle büyük ölçekli tehdit analizlerinde zararlı yazılım kampanyaları arasındaki benzerlikleri tespit ederek siber saldırıları önceden tahmin etme imkânı sunar.

- **Tehdit İstihbaratına Katkı Sağlar**

  Siber tehdit istihbaratı ekipleri, saldırganların değiştirerek yaydığı zararlı yazılım örneklerini analiz etmek için fuzzy hashing yöntemini kullanabilirler. Geleneksel hash'leme yöntemleri, fidye yazılımının farklı varyantlarını ayrı ayrı değerlendirirken, fuzzy hashing varyantlar arasındaki benzerlikleri belirleyerek saldırganın tekniklerini ortaya çıkarabilir. Böylece, siber güvenlik analistleri tehdit aktörlerinin yeniden paketleme, kod şifreleme ve polimorfik teknikleri gibi stratejilerini tespit ederek takip edebilir. Güvenlik ekipleri, fuzzy hashing ile saldırganın geçmişteki saldırılarıyla mevcut saldırılar arasında bağlantılar kurarak daha proaktif savunma mekanizmaları geliştirebilirler.

**Dezavantajlar:**

- **Yanlış Pozitifler Üretebilir**

  Fuzzy hashing, dosya benzerliğini belirlemek için özetleme ve eşleşme algoritmalarını kullanır, bu da bazen tamamen farklı içeriklere sahip dosyaların yanlışlıkla benzer olarak işaretlenmesine neden olabilir. Özellikle büyük veri kümelerinde, küçük benzerlikler nedeniyle farklı dosyalar yanlış eşleşmeler üretebilir ve bu durum, otomatik tehdit avcılığı sistemlerinde yanlış pozitif tespitlere yol açabilir. Önemsiz değişiklikler içeren dosyalar, sistem tarafından zararlı yazılım varyantı olarak algılanabilir ve gereksiz analiz yükü oluşturabilir. Bu yanlış pozitifleri en aza indirmek için, fuzzy hashing genellikle diğer tehdit istihbaratı teknikleriyle (dinamik analiz, imza tabanlı tespit, davranışsal analiz) birlikte kullanılmalıdır.

- **Hesaplama Yükü Fazladır**

  Fuzzy hashing, dosya benzerliğini belirlemek için özetleme ve eşleştirme algoritmalarını kullanır ve geleneksel hash'leme yöntemlerine kıyasla daha fazla işlem gücü ve bellek tüketimi gerektirir. Analiz sürecinde, dosyalar belirli bölümlere ayrılarak karşılaştırılır; ssdeep ve TLSH gibi algoritmalar rolling hash ve blok tabanlı eşleştirme tekniklerini kullandıkları için performans sorunlarına neden olabilir. Ssdeep, her dosya parçası için bir hash değeri oluşturur ve bu değerleri karşılaştırarak benzerlik oranını belirler. TLSH ise, dosyaların içerik özelliklerini özetleyerek benzerlikleri belirler. Hash değerlerinin boyutları geleneksel hash algoritmalarına kıyasla daha büyük olduğundan bellek tüketimi artar. Bu nedenle, büyük ölçekli analizlerde paralel işlem ve önbellekleme stratejileri kullanılarak hesaplama yükü optimize edilmelidir. Hibrit analiz sistemleri, fuzzy hashing'i davranışsal analiz ve makine öğrenmesi tabanlı tehdit tespit teknikleriyle birleştirerek daha verimli hale getirebilir, böylece siber güvenlik analistlerine gelişen saldırı stratejilerini takip etme imkânı sunar.

**Sonuç**

Fuzzy hashing, geleneksel hash'leme yöntemlerinin sınırlamalarını aşarak zararlı yazılım analizi, tehdit avcılığı ve dijital adli bilişim gibi alanlarda kritik bir rol oynar. Dinamik tehditlerin tespiti ve varyant analizi konularında etkili bir araç olan fuzzy hashing, polimorfik ve metamorfik zararlı yazılımlar gibi gelişmiş tehditlere karşı daha hassas analiz yapma imkânı sunar. Geleneksel kriptografik hash algoritmalarının aksine, fuzzy hashing dosyalar arasındaki benzerlikleri hesaplayarak tehditleri tespit etmeye yardımcı olur. Bu özellik, siber tehdit istihbaratında saldırganların kullandığı küçük kod modifikasyonlarını ortaya çıkarmak için oldukça değerlidir. APT grupları ve fidye yazılım saldırıları gibi karmaşık tehdit aktörlerinin geliştirdiği zararlı yazılım varyantlarının belirlenmesinde, fuzzy hashing otomatik tehdit avcılığı sistemlerinin bileşenlerinden biri haline gelmiştir.

Ancak, fuzzy hashing'in bazı sınırlamaları vardır. Yanlış pozitif oranları, bazen tamamen farklı içeriklere sahip dosyaların benzer olarak değerlendirilmesine neden olabilir. Büyük veri kümeleri üzerinde analiz yaparken hesaplama yükü ve bellek tüketimi artabilir. Bu nedenle, fuzzy hashing tek başına bir güvenlik çözümü olarak kullanılmamalıdır. Daha doğru ve kapsamlı tehdit analizi için dinamik analiz, davranışsal tespit ve makine öğrenmesi tabanlı analizlerle entegre edilmelidir.

Fuzzy hashing'in makine öğrenmesi ve büyük veri analitiği ile daha güçlü hale getirilmesi beklenmektedir. Makine öğrenmesi tabanlı fuzzy hashing yöntemleri, yanlış pozitifleri azaltarak daha hassas eşleşmeler yapabilir. Büyük veri analitiği ile entegre edilmiş fuzzy hashing sistemleri, zararlı yazılım varyantlarını gerçek zamanlı olarak daha verimli bir şekilde tespit edebilir. Bulut tabanlı tehdit istihbarat sistemleri, fuzzy hashing tekniklerini küresel ölçekteki saldırıları takip etmek için kullanabilir.

Sonuç olarak, fuzzy hashing, günümüz ve gelecekteki siber güvenlik tehditleriyle mücadelede önemli bir analiz yöntemi olmaya devam edecektir. Geleneksel hash'leme yöntemlerinden farklı olarak, varyant tespiti ve tehdit analizi için güçlü bir çözüm sunmaktadır. Ancak, en iyi sonuçları almak için diğer güvenlik analiz yöntemleriyle birlikte kullanılması gerekmektedir.

# DÖNEM KONUSU

## 7. Siber Güvenlik Profesyonelleri İçin Prompt Mühendisliğinden En İyi Şekilde Faydalanma

GPT-4 gibi Büyük Dil Modelleri (LLM'ler), siber güvenlik profesyonellerinin karmaşık sorunları ele alma şeklini dönüştürüyor. Prompt mühendisliği -LLM'ler için etkili girdiler tasarlama sanatı- savunma (mavi takım), saldırı (kırmızı takım) ve hibrit (mor takım) güvenlik görevlerinde bu modelleri verimli bir şekilde kullanabilmek için kritik bir beceri haline geldi. Bu genişletilmiş rehberde, gelişmiş prompt mühendisliği tekniklerini daha derinlemesine inceleyecek, güvenlik iş akışlarında gerçek dünya uygulamalarını keşfedecek, karşılaşılan zorluklar ve etik hususları ele alacak ve gelecek trendlerini tartışacağız. Tüm süreç boyunca en iyi uygulamaları vurgulayarak, bu bilgileri siber güvenlik operasyonlarınızı güçlendirmek için nasıl kullanabileceğinizi göstereceğiz.

### Gelişmiş Prompt Mühendisliği Teknikleri

LLM'ler güvenlik araçlarına entegre oldukça, profesyoneller de daha sofistike prompt'lar oluşturmayı öğreniyor. Temel tek seferlik (one-shot) soruların ötesinde, gelişmiş teknikler olan **prompt zincirleme (prompt chaining), çok aşamalı prompt'lama (multi-turn prompting) ve bağlamsal bellek yönetimi (contextual memory management)** gibi yöntemler, daha karmaşık ve doğru çıktılar elde etmeyi sağlıyor. Bu teknikleri ustalıkla kullanmak, tek bir prompt'un çözemeyeceği siber güvenlik problemlerine çözüm getirebilir.

#### *Prompt Zincirleme (Prompt Chaining)*

Prompt zincirleme, karmaşık bir görevi daha küçük adımlara bölerek bir dizi ardışık prompt oluşturma yöntemidir. Her prompt'un çıktısı bir sonraki prompt'a giriş olarak kullanılır. Bunun yerine LLM'den tek bir seferde kapsamlı bir analiz yapmasını istemek yerine, adım adım yönlendirme sağlanır. Bu yöntem, modelin alt görevleri bireysel olarak ele almasına olanak tanıdığı için güvenilirliği artırır ve mantıklı bir akış sağlar.

Örneğin, bir tehdit avcılığı (threat hunting) sürecini aşağıdaki gibi aşamalara bölebilirsiniz:

1. İlk prompt ile LLM'den ham günlük (log) verilerinden şüpheli göstergeleri çıkarmasını isteyin.
2. Daha sonra bu göstergeleri ikinci bir prompt'a besleyerek, bilinen tehditlerle analiz ve korelasyon yapmasını sağlayın.
3. Son olarak, üçüncü bir prompt ile modelin olası önleme adımları önermesini sağlayın.

Bu şekilde, modelin önceki yanıtları sonraki adımlara bağlamsal bir temel oluşturur ve daha tutarlı, detaylı bir sonuç elde edilir. Prompt zincirleme yalnızca çok adımlı sorunları çözmekle kalmaz, aynı zamanda modelin muhakeme sürecini daha şeffaf ve hatadan ayıklanabilir (debuggable) hale getirir, çünkü her ara çıktıyı inceleyerek gerekli yerlerde düzeltmeler yapabilirsiniz[15].

#### *Çok Aşamalı Promptlama (Multi-Turn Prompting / Etkileşimli Diyaloglar)*

Çok aşamalı prompt'lama, LLM ile birden fazla değişimde (turn-based) iletişim kurmayı ve her adımda yanıtı daha fazla detaylandırmayı ifade eder. Uzun ve kapsamlı tek bir prompt yazmak yerine, bir analistin bir soruşturmayı konuşarak yürütmesi gibi modelle etkileşimli bir soru-cevap süreci oluşturulur.

Her yeni soru, önceki cevapların bağlamını kullanarak daha ayrıntılı ve anlamlı bilgiler ortaya çıkmasına yardımcı olur. Çok aşamalı diyaloglar, bilgi akışını sürekli hale getirir ve önceki girdilere dayalı olarak modelin bağlamsal hafızasını kullanmasını sağlar.

Örneğin, bir güvenlik analisti şu şekilde bir diyalog kurabilir:

1. **Soru:** "Son 24 saat içinde herhangi bir oturum açma anomalisine rastlandı mı?"
2. **Yanıt:** (LLM analiz sonucunu sunar)
3. **Takip Sorusu:** "Bu anomalilerde yer alan IP adreslerini açıkla ve bilinen kötü amaçlı adreslerle karşılaştır."

Bu iteratif (yinelemeli) yöntem, LLM'in sürekli olarak diyalog bağlamına sadık kalmasını ve yeni veriler ışığında değerlendirme yapmasını sağlar. Çok aşamalı prompt'lama ve prompt zincirleme genellikle bir arada kullanılır -örneğin, bir olayın metodik olarak ele alınmasını sağlamak için yapılandırılmış prompt'ları çok aşamalı bir diyalog içerisinde zincirleme yöntemiyle birleştirebilirsiniz[16].

#### *Bağlamsal Bellek Yönetimi*

LLM'lerin karşılaştığı zorluklardan biri, sınırlı bellek penceresidir (bağlam uzunluğu). Prompt mühendisliğinden en iyi şekilde yararlanmak için profesyonellerin modelin "hatırladığı" bağlamı ustalıkla yönetmesi gerekir. Bağlamsal bellek teknikleri, modelin önemli bilgileri önceki istemlerden veya harici bilgi kaynaklarından korumasını sağlar. Pratikte bu, önceki konuşma noktalarını özetlemek veya modele kritik ayrıntıları unutturmamak için ilgili gerçekleri isteme eklemek anlamına gelebilir. Örneğin bir LLM, birden fazla adımda kötü amaçlı yazılım

analizi yapıyorsa, her yeni istemin başına "şimdiye kadar bilinen gerçeklerin" kısa bir özetini ekleyebilirsiniz. Bazı gelişmiş uygulamalar, olay veritabanlarından veya tehdit istihbarat havuzlarından veri çekmek için vektör aramasıyla geri getirme artırımı (retrieval augmentation) kullanır ve bu verileri isteme ekler. Böylece LLM her zaman doğru bağlama sahip olur. Bu, modele kısa süreli bir bellek veya referans kütüphanesi vermeye benzer. Bu tür yaklaşımlar, modelin uzun oturumlarda tutarsız yanıtlar vermesini önler ve karmaşık konular hakkında tutarlılığı korumasına yardımcı olur[17],[18]. Siber güvenlik ekipleri ayrıca, bağlamın (örneğin, ağ temel istatistikleri veya kullanıcı rolleri gibi) LLM destekli analiz boyunca korunmasını sağlamak için LangChain veya Semantic Kernel gibi bellek modülleri içeren framework'ler kullanır. Bağlamsal belleği proaktif bir şekilde yöneterek, modelin konuya bağlı kalmasını, doğru ve önceki girdilere duyarlı olmasını sağlayabilirsiniz.

### *Few-Shot ve Rol Belirleme İstemleri*

Bahsedilmeye değer bir diğer gelişmiş teknik, modele istenen formatı veya tarzı göstermek için istem içinde giriş-çıkış örnekleri sağladığınız few-shot prompting yöntemidir. Örneğin, bir günlük kaydı girdisini ve doğru yorumlamasını göstererek modelden yeni bir günlük kaydını analiz etmesini isteyebilirsiniz. Bu yöntem, "bağlam içi öğrenmeyi (in-context learning)" kullanarak doğruluğu artırır[16]. Benzer şekilde, "rol belirleme istemleri (role prompting)" modelin belirli bir rol veya kişiliği üstlenmesini sağlamaktır (Örn. "Sen bir kötü amaçlı yazılım analisti yapay zekâsın…"). Bu, modelin yanıtlarını beklenen alan uzmanlığına ve üsluba uygun hale getirebilir. Bu yöntemler, zincirleme (chaining) ve çok adımlı diyaloglarla birleştirildiğinde, siber güvenlik profesyonellerine yapay zekâdan yüksek kaliteli ve ilgili çıktılar almak için güçlü bir araç seti sunar.

### *En İyi Uygulamalar*

Bu gelişmiş teknikleri iş akışlarınıza kademeli olarak dahil edin. Geniş kapsamlı sorular sormak yerine görevleri alt görevlere ayırarak ve LLM ile iteratif şekilde ilerleyerek başlayın. Önemli bulguları özetleyerek ve bunları takip istemlerine yeniden dahil ederek bağlamı koruyun. Sektörde birçok uzman, netlik ve yapılandırılmış istemlerin önemini vurgular -gelişmiş yöntemler kullanılsa bile açık talimatlar ve mantıklı akış kritik önemdedir. Bir kaynağın belirttiği gibi, bir istem genellikle bir giriş (veri veya soru), bağlam (talimatlar veya arka plan bilgisi) ve mümkünse örnekler içermelidir[16]. Prompt engineering stratejilerini (örneğin, zincirleme ve bağlam yönetimi) ustalıkla kullanarak, LLM'leri siber güvenlik cephaneliğinizde gerçekten etkili asistanlar haline getirebilirsiniz.

## Gerçek Dünyada LLM'lerin Siber Güvenlikte Kullanım Alanları

LLM'ler deney aşamasını geride bırakarak gerçek kuruluşlarda çeşitli siber güvenlik işlevlerini desteklemeye başladı. Aşağıda, güvenlik operasyonları, tehdit istihbaratı ve otomatik güvenlik değerlendirmeleri gibi alanlardaki gerçek dünya uygulamalarını vurgulayarak, bu yeteneklerin nasıl açığa çıkarılabileceğini gösteriyoruz. Siber güvenlik ekipleri, iyi tasarlanmış komutlarla (prompt engineering) LLM'lerin zaman alan analiz görevlerini yerine getirebildiğini, içgörüler oluşturabildiğini ve hatta savunma içerikleri üretebildiğini keşfediyor -böylece insanlar daha üst düzey karar alma süreçlerine odaklanabiliyor.

### *Güvenlik Operasyonları ve Olay Müdahalesi (Mavi Takım)*

En etkili kullanım alanlarından biri, Güvenlik Operasyon Merkezi (SOC) analistlerine olayların önceliklendirilmesi ve soruşturulmasında yardımcı olmaktır. LLM'ler uyarıları özetleyip önceliklendirebilir, log'ları tarayabilir ve sonraki adımları önerebilir. Örneğin, Microsoft'un güvenlik odaklı yapay zekâ asistanı **Security Copilot**, bir analistin, "Son 48 saat içinde sunucu X'te şüpheli oturum açma kalıplarını araştır" gibi doğal dilde bir komut girmesine olanak tanır ve organizasyonun güvenlik telemetrisine dayanarak anlamlı bir analiz sunar[19]. Security Copilot, OpenAI'ın GPT-4 modelini Microsoft'un güvenlik özelinde eğitilmiş modeli ve tehdit istihbaratıyla birleştirerek, basit bir komutu analist için sorgular ve içgörüler içeren kapsamlı bir araştırmaya dönüştürür[19]. Bu tür araçlar rutin kontrolleri otomatikleştirebilir, anormallikleri belirleyebilir ve soruşturma sonuçlarına dayalı olay raporları oluşturabilir. Bir başka kullanım örneği, LLM'lerin tehdit algılama kuralları üretmesidir: Karmaşık SIEM sorgularını sıfırdan yazmak yerine, bir analist senaryoyu ("brute-force giriş denemelerini ve ardından yönetici işlemlerini tespit et") tanımlayarak LLM'in bir **Sigma kuralı** veya **Splunk sorgusu** önermesini sağlayabilir. McKinsey'e göre, üretken yapay zekâ SIEM için tespit kuralları önerip yazabiliyor ve analistlerin tehditleri belirlemek için harcadığı zamanı yüzde 20–25 oranında azaltabiliyor[20]. Komut mühendisliğini kullanarak, mavi takımlar büyük veri yığınlarından hızlı (ve gerekçeli) yanıtlar alabilir, olay soruşturmalarını hızlandırabilir ve tehdit algılama süreçlerinin manuel olarak yapılan bölümlerini otomatikleştirebilir.

### *Tehdit İstihbaratı ve İzleme*

LLM'ler, tehdit istihbarat ekiplerinin bilgileri işleme biçimini büyük ölçüde geliştiriyor. Bu modeller, metin okuma ve özetleme konusunda son derece başarılıdır; bu da tehdit istihbaratı analistlerinin raporlar, kötü amaçlı yazılım açıklamaları ve tehdit akışları içinde boğulmasını önler. Bir LLM, doğru komutlarla, 50 sayfalık bir tehdit

raporunu temel çıkarımlara indirgeme, yabancı dilde yazılmış bir tehdit grubunun taktiklerini çevirme veya yapılandırılmamış olay verilerinden **güvenlik ihlali göstergelerini (IoC)** çıkarma görevlerini yerine getirebilir. Bazı kuruluşlar, günlük haberleri veya ham istihbarat verilerini modele girerek **sürekli güncellenen bir "tehdit özeti"** oluşturmak için LLM'leri kullanıyor. Tehdit istihbaratı platformlarıyla entegrasyonlar da gelişiyor -örneğin, **Google'ın Mandiant Threat Intelligence yapay zekâ** aracı, Mandiant'ın devasa tehdit veritabanını kullanarak tehditleri hızlı bir şekilde bulmak ve özetlemek için özel bir LLM (Sec-PaLM) kullanıyor[21]. Analistler, **"Bu hafta sağlık sektörünü hedef alan yeni fidye yazılımı kampanyaları var mı?"** gibi basit bir İngilizce soru sorarak, binlerce veri noktasından damıtılmış, ilgili aktörleri ve yöntemleri içeren bir LLM özetine ulaşabiliyor. Ayrıca, LLM'ler açık kaynak kanallarını (sosyal medya, forumlar, dark web gönderileri) izleyerek saldırganların kullandığı şifreli dili veya argo ifadeleri anlamlandırabiliyor -bu, geleneksel olarak insan dilbilimciler gerektiren bir görevdi. **VirusTotal Code Insight** özelliği, LLM kullanarak potansiyel kötü amaçlı betikleri analiz edip, insan tarafından okunabilir bir açıklama sunarak tehdit araştırmacılarının yeni kötü amaçlı yazılımları daha hızlı anlamalarına olanak tanıyor[22]. Komut mühendisliğini kullanarak, tehdit istihbarat ekipleri verileri **saatler yerine dakikalar içinde eyleme dönüştürülebilir içgörülere** çevirebilir.

### *Otomatik Güvenlik Değerlendirmeleri ve Red Team Desteği (Red/Purple Team)*

Saldırı tarafında, büyük dil modelleri (LLM'ler) red team'lere zafiyet araştırması, exploit geliştirme ve sosyal mühendislik senaryoları gibi görevlerde yardımcı olmaktadır. Etik hususlar geçerli olmakla birlikte (bunları daha sonra ele alacağız), bir red team uzmanının keşif sürecini hızlandırmak için bir prompt kullanabileceğini düşünelim: "2025 itibarıyla Apache Struts için bilinen CVE'leri ve exploit tekniklerini özetle." LLM, ilgili bilgileri hızla derleyerek manuel araştırma süresinden tasarruf sağlayabilir. Prompt mühendisliği ayrıca gerçekçi oltalama e-postaları veya güvenlik farkındalığı testleri için dolandırıcılık senaryoları oluşturmakta da yardımcı olabilir. Örneğin, "BT destek ekibi gibi davranarak, şifre sıfırlamaya teşvik eden ve hafif dilbilgisi hataları içeren bir e-posta hazırla" şeklindeki bir prompt, LLM'in saniyeler içinde ikna edici bir oltalama şablonu üretmesini sağlayabilir. Bu, red team'lerin (kontrollü koşullar altında) düşman tuzaklarını simüle ederek bir kuruluşun savunmasını test etmesine yardımcı olur. Ek olarak, LLM'ler kod inceleme ve exploit yazımı konusunda destek sağlayabilir: Bir purple team, bir uygulama kod parçasını LLM'e "Bu kodda güvenlik açıklarını bul ve olası exploit öner" prompt'u ile besleyebilir. Model, tehlikeli işlevleri veya mantık hatalarını vurgulayabilir hatta bir exploit'in nasıl çalışabileceğini önererek ekibin bunu doğrulamasını sağlayabilir. Bazı güvenlik araştırmacıları, "otomatik bir sızma testi aracı" oluşturmak için prompt zincirleme yöntemlerini keşfetmiştir; örneğin, bir prompt açık servisleri listelemek için, bir diğeri potansiyel exploit'leri belirlemek için ve sonraki prompt'lar kontrollü bir ortamda sömürü denemek için (adımları raporlayarak ancak fiili uygulamadan kaçınarak) kullanılabilir. Yapay zekâ ile tam otonom red-teaming hâlâ deneysel aşamada olsa da, bu kullanım örnekleri yönü göstermektedir: LLM'ler saldırı yüzeylerini analiz etme veya hedef bilgilerini bir insandan çok daha hızlı inceleme konusunda bir çarpan etkisi yaratabilir. Hem saldırı hem de savunmayı birleştiren purple team'ler, LLM'leri saldırı senaryoları oluşturmak ve ardından ilgili tespit mantığını anında geliştirmek için kullanarak, prompt mühendisliğini her iki açıdan güvenliği iyileştiren bir işbirliği aracı haline getirebilir.

### *Güvenlik Dokümantasyonu ve Politikası*

Daha az teknik ama oldukça pratik bir kullanım alanı, LLM'lerin güvenlik dokümantasyonu oluşturma ve inceleme süreçlerinde kullanılmasıdır. Prompt mühendisliği, olay müdahale planları, uyumluluk raporları veya en iyi uygulamalara dayalı yapılandırma kılavuzları gibi belgelerin ilk taslaklarını oluşturmak için yardımcı olabilir. Örneğin, "Orta ölçekli bir finans şirketi için ISO 27001 uyumlu bir erişim kontrol politikası taslağı hazırla" prompt'u, güvenlik yöneticisinin daha sonra üzerinde çalışabileceği sağlam bir başlangıç dokümanı üretebilir. LLM'ler ayrıca, kurumsal politikalarla düzgün şekilde ayarlandığında, çalışanların güvenlikle ilgili sıkça sorulan sorularını yanıtlayan bir iç chatbot olarak da kullanılabilir. Bazı kuruluşlar, çalışanların "Bir oltalama e-postasından şüphelenirsem ne yapmalıyım?" gibi sorular sormasını ve anında standartlaştırılmış rehberlik almasını sağlayan yapay zekâ asistanları uygulamaya koymuştur (bu da güvenlik yardım bölümüne olan yükü azaltır). Hatta güvenlik anketlerini doldurma veya denetim süreçlerini hızlandırma gibi karmaşık görevler de hızlandırılabilir: Üretken yapay zekâ, kuruluşun mevcut kontrolleriyle ilgili prompt'lar kullanılarak tekrar eden uyumluluk formlarını otomatik olarak doldurabilir. McKinsey, bu tür otomasyonun önemli ölçüde zaman tasarrufu sağlayabileceğini belirtmektedir. Bazı durumlarda yapay zekâ destekli güvenlik anketi doldurma süreçlerinde yüzde 80'e varan zaman tasarrufu gözlemlenmiştir[20]. Bu tür bir kullanım, güvenlik profesyonellerinin yazım gibi tekrarlayan işler yerine, daha stratejik iyileştirmelere odaklanmalarını sağlar. Buradaki kritik nokta, prompt mühendisliğinin modeli, doğru politika detaylarını ve tonu içerecek şekilde yönlendirmesi ve nihai çıktının doğruluğunu sağlamak için her zaman bir insan tarafından kontrol edilmesidir.

Bu gerçek dünya örnekleri, prompt mühendisliğinin sadece teorik bir egzersiz olmadığını -bugün siber savunma ve saldırıda somut iyileştirmeler sağladığını gösteriyor. Sahadan gelen en iyi uygulamalar arasında pilot projelerle başlamak yer alıyor: belirli bir sorun noktası belirleyin (örneğin, "özetlenmesi gereken çok fazla uyarı"

veya "zaman alıcı rapor yazımı") ve bunu çözmek için bir LLM istemi veya zinciri ile deneyler yapın. Deneyimli analistlerinizi istemlerin oluşturulmasına dahil edin -alan bilgileri, yapay zekânın doğru ayrıntılara odaklanmasını sağlayacaktır[23]. Birçok ekip, yapay zekâ çıktılarının kalitesini değerlendirip kelime seçimi veya yaklaşımda titiz davranarak istemleri sürekli olarak yinelemektedir (tıpkı bir arama sorgusunu veya kod parçasını rafine etmek gibi). Unutulmamalıdır ki, mevcut LLM'lerin en iyi çalışma biçimi, onları bağımsız karar vericiler olarak değil, asistanlar olarak kullanmaktır. O nedenle LLM'leri insan yargısının yerine koymak için değil, insan yargısını tamamlamak için kullanın. Dikkatli bir şekilde entegre edildiğinde, LLM'ler analiz ve içerik oluşturma süreçlerinde ağır iş yükünü üstlenerek siber güvenlik profesyonellerinin daha yüksek bir verimlilik ve içgörü seviyesinde çalışmasını sağlayabilir.

## Zorluklar ve Etik Hususlar

LLM'lerin siber güvenlikteki avantajları açık olsa da ele alınması gereken önemli zorluklar ve etik meseleler de vardır. Yapay zekâ çift taraflı bir kılıçtır: Savunucuları güçlendiren aynı araçlar, kötü niyetli aktörler tarafından da kötüye kullanılabilir ve teknolojinin kendisi belirli sınırlamalar ve riskler içerir. Siber güvenlik uzmanları, LLM'leri sorumlu ve güvenli bir şekilde kullanabilmek için bu faktörleri dikkate almalıdır.

### *Halüsinasyonlar ve Doğruluk*

LLM'ler, kendinden emin ve otoriter görünen ancak gerçekte yanlış veya alakasız olan çıktılar üretebilir -bu durum genellikle halüsinasyon olarak adlandırılır. Güvenlik bağlamında bu, bir yapay zekânın zararsız bir dosyayı yanlışlıkla kötü amaçlı yazılım olarak tanımlaması veya gerçekte var olmayan bir tehdidi uydurması anlamına gelebilir. Bu tür çıktılara körü körüne güvenmek analistleri yanıltabilir ve daha kötüsü, hatalı müdahalelere yol açabilir. Sektör uzmanları, LLM çıktılarının test edilmesini ve doğrulanmasını tavsiye etmektedir. Örneğin Microsoft, **Security Copilot** aracının "her şeyi her zaman doğru yapamayabileceğini" ve hata yapabileceğini açıkça belirtmektedir; bu nedenle, kullanıcıların yapay zekâyı düzeltebilmesi için geribildirim mekanizmaları sunulmaktadır[19]. En iyi uygulama olarak, yapay zekâdan gelen önerileri hipotezler veya taslak içgörüler olarak değerlendirin -kritik ayrıntıları her zaman diğer araçlar veya insan uzmanlarla doğrulayın. Bir LLM eğer bir tespit kuralı oluşturduysa, bunu bir insanın gözden geçirmesi ve dağıtımdan önce güvenli bir ortamda test etmesi en iyisidir. Geribildirim döngüleri (modele ne zaman yanlış yaptığını söylemek veya doğru yaptığında onaylamak) zamanla performansı iyileştirebilir, ancak insan denetimi her zaman birinci öncelik olmalıdır[19].

### *Önyargı ve Adillik*

LLM'ler büyük miktarda metin verisinden öğrenir ve ne yazık ki bu veriler kültürel, ırka yönelik veya cinsiyetle ilgili önyargılar içerebilir. Bu önyargılar da modelin çıktılarına yansıyabilir. Siber güvenlik kullanımında önyargı, ince ama etkili yollarla ortaya çıkabilir. Örneğin, bir yapay zekâ belirli etnik kökenlere ait isimleri orantısız şekilde dolandırıcılıkla ilişkilendirebilir veya İngilizce raporlanan tehditleri, diğer dillerde raporlanan eşdeğer tehditlere göre daha öncelikli görebilir. **Prompt mühendisleri**, bu tuzakların farkında olmalıdır. Prompt'ları tasarlarken ve yanıtları değerlendirirken adil olmayan veya önyargılı çıkarımlara karşı dikkatli olunmalıdır. Bir rehber, çeşitli bakış açılarını aktif olarak dikkate almayı ve ayrımcı sonuçlara yol açabilecek dil kullanımından kaçınmayı önermektedir[23]. Örneğin, olay etkisi analizi oluştururken, prompt'un yapay zekâyı kanıt olmadan bireyleri veya grupları suçlamaya yönlendirmediğinden emin olun. Ayrıca, kritik çıktıları çapraz kontrol etmek için birden fazla prompt veya araç kullanın (eğer farklı şekilde ifade edilmiş iki prompt aynı olay hakkında tamamen farklı yanıtlar veriyorsa, bu bir uyarı işaretidir). Siber güvenlikte etik yapay zekâ kullanımını sürdürmek, olayları aşırı genelleştirmemeyi de gerektirir -her olay, kendi özel verileri ve bağlamı içinde analiz edilmelidir.

### *Veri Güvenliği ve Gizlilik*

Siber güvenlik uzmanları için en acil endişelerden biri, LLM'lerle paylaşılan verilerin hassasiyetidir. Birçok popüler LLM hizmeti bulut tabanlıdır, yani gönderdiğiniz herhangi bir prompt (ve içeriği) harici sunucularda saklanabilir ve hatta modelleri eğitmek için kullanılabilir. Bu durum, bariz gizlilik riskleri taşır. Bunun ünlü bir örneği, Samsung mühendislerinin, hataları ayıklamak için ChatGPT'ye hassas kaynak kodunu yapıştırarak yanlışlıkla sızdırmalarıdır[24]. Bu olay sonucunda, Samsung ve birçok şirket, yapay zekâ hizmetlerinin dahili kullanımını yasaklamış veya sıkı şekilde kısıtlamıştır.

Bu riski azaltmak için, **herhangi bir LLM'e gizli veya kişisel verileri yetkisiz bir şekilde girmeyin** ve uygun kontroller olmadan paylaşmayın. Alternatifler arasında, bazı kuruluşların verileri kendi bünyesinde tutmak için **yerel veya kendi kendine barındıran LLM çözümlerini kullanması** yer almaktadır. Başka bir strateji, yapay zekâya beslemeden önce verileri anonimleştirmek veya temizlemektir. Örneğin, modelden bir log analiz etmesini isterken gerçek IP adresleri veya kullanıcı adları yerine yer tutucular kullanabilir ve sonuçları model dışı bir ortamda gerçek değerlere geri eşleyebilirsiniz. Ayrıca, **çalışanlara yapay zekâ ile hangi verilerin paylaşılmasının uygun olduğu konusunda eğitim verilmelidir**. Açık politikalar oluşturmak ve teknik kontroller (örneğin, dışa giden prompt'larda hassas verileri tespit eden **DLP sistemleri**) kullanmak, yanlışlıkla gerçekleşen veri

sızıntılarını önleyebilir. Sonuç olarak, bir LLM'i **harici bir yüklenici gibi ele alın**: Ona yalnızca görev için gerekli bilgileri verin ve bundan fazlasını paylaşmayın.

### *Düşmanca Kullanım ve Yanlış Bilgilendirme*

Güvenlik ekipleri, saldırganların da Büyük Dil Modellerine (LLM) erişimi olduğunu göz önünde bulundurmalıdır. Tehdit aktörlerinin üretken yapay zekâyı daha inandırıcı oltalama e-postaları hazırlamak, sahte haberler oluşturmak hatta kötü amaçlı yazılım kodu yazımında yardımcı olarak kullandığına dair giderek artan kanıtlar bulunmaktadır[25], [26]. Bu durum savunucular için riski artırmaktadır -daha sık ve sofistike yapay zekâ destekli saldırıları öngörmek zorundayız. Etik açıdan bakıldığında bu, saldırganlara yardımcı olabilecek yapay zekâ içeriklerinin yayınlanmaması gerektiği anlamına da gelir. Örneğin, bir kırmızı takım tatbikatının parçası olarak bir yapay zekâdan bir kavram kanıtı (proof-of-concept) güvenlik açığı oluşturmasını isterseniz, bu çıktıyı gerçek bir güvenlik açığı gibi dikkatle ele almalı, kamuya açık olarak paylaşmamalı ve amaçlanan testin ötesinde kullanmamalısınız.

Prompt mühendisliği de düşmanca manipülasyona maruz kalabilir. Prompt enjeksiyonu saldırıları, harici bir tarafın yapay zekâ asistanınızı, talimatlarını görmezden gelmesi veya gizli bilgileri açıklaması için kandırmaya çalışma yöntemleridir. Yapay zekâ destekli bir sohbet botu dağıttığınızda (örneğin, çalışan güvenlik soruları için), bir saldırgan, onu iç bilgileri açıklamaya veya istenmeyen eylemler gerçekleştirmeye yönlendiren girdiler oluşturabilir. Bunun önüne geçmek için kullanıcı istemlerinden gizlenmiş sistem veya geliştirici düzeyinde talimatlar kullanmak ve kullanıcı girdilerini sanitize etmek gereklidir. Yapay zekâ sistemlerinizi her zaman kırmızı takım zihniyetiyle test edin: onları karmaşık istemlerle bozmayı veya yanıltmayı deneyin ve zayıf noktaları düzeltin (bu süreç bazen yapay zekâ kırmızı takım testleri olarak adlandırılır)[27], [28]. Bu risklerin farkında olmak, LLM'leri yeni güvenlik açıkları oluşturmadan kullanmanızı sağlar.

### *Etik Kullanım ve Uyumluluk*

Siber güvenlikte yapay zekâ kullanımı aynı zamanda yasal ve etik uyumlulukla da kesişir. GDPR veya farklı ulusal veri koruma yasaları, belirli koşullarda yapay zekâ çıktılarının veya yapay zekâ için kullanılan verilerin kişisel veri olarak sınıflandırılmasına neden olabilir (örneğin, bir yapay zekâ özeti yanlışlıkla kişisel bilgileri açığa çıkarırsa). Siber güvenlik profesyonelleri, yapay zekâ çözümlerini uygularken hukuk ve uyumluluk ekipleriyle koordineli çalışmalıdır.

Ayrıca, şeffaflık, temel bir etik ilkedir -bir olay raporunda yapay zekâ tarafından üretilmiş bir analiz kullanıyorsanız, bunu belirtmek ve içeriğin doğruluğunu teyit etmek için bir süreç oluşturmak akıllıca olabilir. Örneğin CISA, yapay zekâyı kurum içinde nasıl benimsediğine dair "radikal şeffaflık ve hesap verebilirlik" ilkesini vurgulamaktadır[29]. Benzer şekilde, güvenlik ekipleri de yapay zekâ araçlarının karar alma süreçlerinde nasıl kullanıldığını belgelemelidir.

Buna ek olarak, yapay zekânın hatalarını yönetme sürecini belirlemek önemlidir (Örneğin, yapay zekâ yanlış bir öneri sunduğunda nasıl bir yol izleneceği belirlenmelidir). Yapay zekâya aşırı bağımlılığı önlemek de kritik bir önem taşımaktadır. Özellikle, veri ihlaline maruz kalan kişilerle iletişim kurma veya risk kabul kararları verme gibi insan empatisinin ve sezgisinin gerekli olduğu senaryolarda yapay zekâyı tek başına kullanmaktan kaçınılmalıdır.

Prompt mühendisliğinden güvenli bir şekilde yararlanabilmek için güvenlik profesyonellerinin yapay zekâ kullanımına çeşitli sınırlamalar getirmesi gerekir. Bu, LLM çıktılarını sürekli doğrulamak ve önyargılarını analiz etmek, güçlü veri yönetimi uygulamaları benimsemek ve yapay zekânın sınırlarını anlamak anlamına gelir. Bu zorluklarla doğrudan yüzleşerek -politikalar, kullanıcı eğitimi, teknik önlemler ve etik yönergeler yoluyla- kuruluşlar, LLM'lerin verimlilik avantajlarından yararlanırken gereksiz risklerden kaçınabilir.

### Sonuç

Bu makalede, siber güvenlik profesyonellerinin yapay zekâ destekli büyük dil modellerinden (LLM) en verimli şekilde yararlanabilmesi için gelişmiş prompt mühendisliği teknikleri ele alınmıştır. Özellikle prompt zincirleme, çok aşamalı prompt'lama, bağlamsal bellek yönetimi ve rol belirleme gibi stratejilerin, güvenlik operasyonları, tehdit istihbaratı ve otomatik güvenlik değerlendirmeleri gibi alanlardaki pratik uygulamaları incelenmiştir. Bununla birlikte, LLM'lerin doğruluk, önyargı, veri güvenliği ve etik kullanım açısından doğurabileceği riskler de vurgulanmıştır. Sonuç olarak, siber güvenlik uzmanlarının yapay zekâyı güvenli ve etik bir şekilde kullanabilmesi için güçlü bir prompt mühendisliği yaklaşımı geliştirmesi, doğrulama süreçleriyle desteklemesi ve insan denetimini ön planda tutması gerektiği ortaya konmuştur.

## Honeypot Verileri

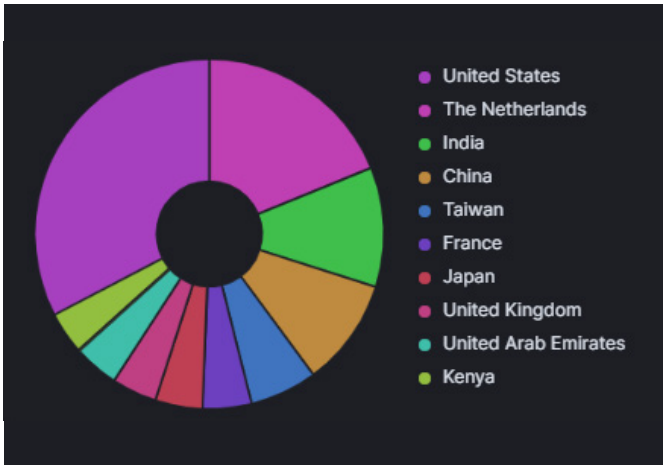
Bu rapor üç ay içinde Honeypot sensörlerimizden topladığımız verilerle oluşturulmuştur. Saldırıların en çok toplandığı ülkeler, portlar, en çok denenen kullanıcı adları ve parolalar, veriler azalan sırada listelenerek Tablo 1'de inceleme için sunulmuştur.

Şekil 4 ve Tablo 1'deki veriler incelendiğinde, en çok saldırı gelen 10 ülke listesinin ilk sırasında ABD (yüzde 26,39) yer alırken, ardından sırasıyla Hollanda (yüzde 15,25), Hindistan (yüzde 8,95), Çin (yüzde 8,06) ve Tayvan (yüzde 5,06) gelmektedir.

Tablo 2'de de görüldüğü üzere en çok saldırı 445 portuna gelmiştir. 445 portunda sunucuların yazıcı ve paylaşılan

| Saldırıların Geldiği Ülke | Saldırı Miktarı (%) |
|---|---|
| ABD | 26,39 |
| Hollanda | 15,25 |
| Hindistan | 8,95 |
| Çin | 8,06 |
| Tayvan | 5,06 |
| Fransa | 3,66 |
| Japonya | 3,54 |
| İngiltere | 3,37 |
| BAE | 3,37 |
| Kenya | 3,24 |

**Tablo 1:** En çok saldırı gelen 10 ülke ve saldırıların yüzdelik dağılımı.




**Şekil 4:** Gelen saldırıların ülkelere göre dağılımı.

| Saldırılan Port | Saldırı Miktarı (%) |
|---|---|
| 445 | 7,82 |
| 443 | 7,63 |
| 22 | 4,13 |
| 135 | 2,84 |
| 64295 | 2,70 |
| 53 | 2,12 |
| 23 | 1,71 |
| 80 | 1,61 |
| 64297 | 1,27 |
| 3001 | 1,24 |

**Tablo 2:** En çok saldırı gelen portlar ve yüzdelik dağılımı.

| Denenen Parola | Deneme Miktarı (%) |
|---|---|
| 123456 | 12,58 |
| (boş) | 3,57 |
| 123 | 2,14 |
| admin | 1,81 |
| abc123 | 1,76 |
| password | 1,21 |
| 1 | 0,93 |
| 1234 | 0,88 |
| 111111 | 0,71 |
| 12345678 | 0,71 |

**Tablo 3:** SSH ve RDP honeypot'larımız üzerinde en çok denenen parolalar.

dosyalar için kullandığı SMB servisi çalışmaktadır. Bu yüzden SMB servisinin diğer servislerden daha çok saldırı alması beklenen bir durum olarak kabul edilebilir.

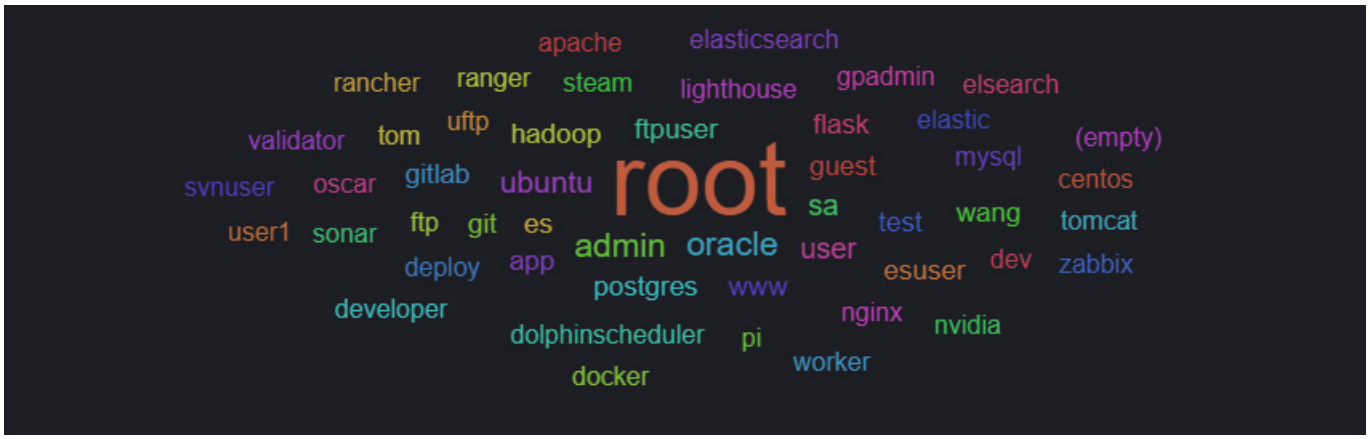
İkinci sırada 443 numaralı port yer almaktadır. 443 portunda sunucuların web üzerinden güvenli iletişim sağlamak amacıyla kullandığı HTTPS servisi çalışmaktadır. Web sunucularındaki güncel güvenlik açıklarının tespiti, kimlik doğrulama hatalarının istismarı ve veri sızıntısı yaratma hedefleriyle saldırganlar tarafından tercih edilmesi gibi nedenlerle HTTPS servisinin diğer servislerden daha çok saldırı alması beklenen bir durum olarak kabul edilebilir.

Üçüncü sırada 22 numaralı port yer almaktadır. 22 portunda, sunucuların uzaktan erişim ve yönetim için kullandığı SSH servisi çalışmaktadır. Saldırganların brute-force denemeleri, zayıf parola kullanımı ve oturum açma hatalarını hedef alarak yetkisiz erişim sağlama amacıyla saldırılar gerçekleştirmesi gibi sebeplerden ötürü SSH servisinin diğer servislerden daha çok saldırı alması beklenen bir durum olarak değerlendirilebilir.

| Denenen Kullanıcı Adı | Deneme Miktarı (%) |
|---|---|
| root | 25,76 |
| admin | 3,62 |
| oracle | 3,4 |
| ubuntu | 2,47 |
| sa | 2,03 |
| user | 1,76 |
| guest | 1,54 |
| es | 1,48 |
| test | 1,48 |
| hadoop | 1,48 |

**Tablo 4:** SSH ve RDP honeypot'larımız üzerinde en çok denenen kullanıcı adları.


**Şekil 5:** Parola etiket bulutu.




**Şekil 6:** Kullanıcı adı etiket bulutu.

Denenen parolalar incelendiğinde, birçok yönetim arayüzünün standart olarak kullandığı parolalar olan 123, password, 123456 gibi terimler gözlemlenmektedir. En çok denenen parolalar Tablo 3'te görülmektedir. Bu parolaların test veya deneme süreçleri tamamlanır tamamlanmaz değiştirilmesi ve karmaşık, 12-16 karakterli, özel karakter içeren parolalarla değiştirilmesi analistlerimiz tarafından tavsiye edilmektedir. Ayrıca Şekil 5'te yer alan parola etiket bulutunda görüldüğü üzere, kolay hatırlanması için herhangi bir harf ve özel karakter içermeyen, sadece sıralı sayılarla oluşturulmuş parolalar kullanmaktan kaçınılmalıdır.

Denenen kullanıcı adları incelendiğinde, yeni sistemlerde sıklıkla kullanılan root, admin, oracle gibi kullanıcı adlarının saldırganlar tarafından tercih edildiği görülmektedir. Tablo 4'te en çok denenen kullanıcı adları görülmektedir. Kurulumu tamamlanan servislerin ve yönetim panellerinin kullanıcı adlarının en kısa zamanda değiştirilmesi ve kurulan sistemlerin kendi isimlerinin (örn. ubuntu, postgres, oracle, testuser) kullanılmaması tavsiye edilmektedir.

# KAYNAKÇA:

[1] "TRM labs," 27 February 2025.[Çevrimiçi]. Available: https://www.trmlabs.com/post/the-bybit-hack-following-north-koreas-largest-exploit.

[2] "Kaiko Research," 24 February 2025.[Çevrimiçi]. Available: https://research.kaiko.com/insights/bybit-hack-by-the-numbers.

[3] " Internet Crime Complaint Center," 26 February 2025.[Çevrimiçi]. Available: https://www.ic3.gov/PSA/2025/PSA250226.

[4] "Infosecurity Magazine," 24 February 2025.[Çevrimiçi]. Available: https://www.infosecurity-magazine.com/news/bybit-140m-bounty-recover-mega/.

[5] "Office of Public Affairs," 6 February 2025.[Çevrimiçi]. Available: https://www.justice.gov/archives/opa/pr/three-north-korean-military-hackers-indicted-wide-ranging-scheme-commit-cyberattacks-and.

[6] "DoD Enterprise DevSecOps Fundamentals," Mart 2021.

[7] [Çevrimiçi]. Available: https://thinktech.stm.com.tr/tr/siber-tehdit-durum-raporu-ekim-aralik-2024. [Erişildi: 03 03 2025].

[8] [Çevrimiçi]. Available: https://attack.mitre.org/. [Erişildi: 06 03 2025].

[9] [Çevrimiçi]. Available: https://www.mitre.org/news-insights/impact-story/mitre-engage-framework-and-community-cyber-deception. [Erişildi: 06 03 2025].

[10] [Çevrimiçi]. Available: https://engage.mitre.org/. [Erişildi: 06 03 2025].

[11] [Çevrimiçi]. Available: https://en.wikipedia.org/wiki/Deception_technology, .[Erişildi: 06 03 2025].

[12] "Gartner, "Emerging Tech: Build Preemptive Security Solutions to Improve Threat Detection (Part 1) ID G00810682,"," 2024.

[13] [Çevrimiçi]. Available: https://csrc.nist.gov/pubs/sp/800/160/v2/r1/final. [Erişildi: 03 03 2025].

[14] [Çevrimiçi]. Available: https://www.sans.org/blog/a-decade-of-security-assessments-security-issues-that-refuse-to-die/. [Erişildi: 04 03 2024].

[15] DAIR.AI, "Prompt Chaining | Prompt Engineering Guide,"[Çevrimiçi]. Available: https://www.promptingguide.ai/techniques/prompt_chaining. [Erişildi: 04 03 2025].

[16] mercity, "Advanced Prompt Engineering Techniques,"[Çevrimiçi]. Available: https://www.mercity.ai/blog-post/advanced-prompt-engineering-techniques. [Erişildi: 04 03 2025].

[17] A. D. Blasi, "Top 20 Prompting Techniques In Use Today: A Real LLM Prompting Guide For Professional Results Using an Interface or API," AI Weekly, 03 09 2024.[Çevrimiçi]. Available: https://ai-weekly.ai/top-20-prompting-techniques-in-use-today/. [Erişildi: 04 03 2025].

[18] S. M, "Dynamic Prompt Adaptation in Generative Models," Analytics Vidhya, 27 12 2024.[Çevrimiçi]. Available: https://www.analyticsvidhya.com/blog/2024/12/dynamic-prompt-adaptation-in-generative-models/. [Erişildi: 04 03 2025].

[19] V. Jakkal, "Introducing Microsoft Security Copilot: Empowering defenders at the speed of AI," Microsoft, 28 03 2023.[Çevrimiçi]. Available: https://blogs.microsoft.com/blog/2023/03/28/introducing-microsoft-security-copilot-empowering-defenders-at-the-speed-of-ai/. [Erişildi: 04 03 2025].

[20] J. Greis ve M. Sorel, "The cybersecurity provider's next opportunity: Making AI safer," McKinsey & Company, 14 11 2024.[Çevrimiçi]. Available: https://www.mckinsey.com/capabilities/risk-and-resilience/our-insights/the-cybersecurity-providers-next-opportunity-making-ai-safer. [Erişildi: 04 03 2025].

[21] S. Potti, "Supercharging security with generative AI," Google, 23 04 2023.[Çevrimiçi]. Available: https://cloud.google.com/blog/products/identity-security/rsa-google-cloud-security-ai-workbench-generative-ai. [Erişildi: 04 03 2025].

[22] B. Quintero, "Introducing VirusTotal Code Insight: Empowering threat analysis with generative AI," VirusTotal, 23 04 2023.[Çevrimiçi]. Available: https://blog.virustotal.com/2023/04/introducing-virustotal-code-insight.html. [Erişildi: 04 03 2025].

[23] Threatshare.ai, "Cracking the Code: Unraveling Prompt Engineering Best Practices," Threatshare.ai, 15 05 2024.[Çevrimiçi]. Available: https://threatshare.ai/prompteng/prompt-engineering-best-practices-2/. [Erişildi: 04 03 2025].

[24] J. Vijayan, "Samsung Engineers Feed Sensitive Data to ChatGPT, Sparking Workplace AI Warnings," Darkreading, 11 04 2023.[Çevrimiçi]. Available: https://www.darkreading.com/vulnerabilities-threats/samsung-engineers-sensitive-data-chatgpt-warnings-ai-use-workplace. [Erişildi: 04 03 2025].

[25] S. Sabin, "Hackers are already abusing ChatGPT to write malware," Axios, 10 01 2023.[Çevrimiçi]. Available: https://www.axios.com/2023/01/10/hackers-chatgpt-malware-cybercrime-ai. [Erişildi: 04 03 2025].

[26] L. Laws ve M. Luallen, "Can ChatGPT write malware?," The Grainger College of Engineering Information Trust Institute, 02 05 2023.[Çevrimiçi]. Available: https://iti.illinois.edu/news/chatgpt-malware. [Erişildi: 04 03 2025].

[27] K. Vongthongsri, "Red Teaming LLMs: The Ultimate Step-by-Step LLM Red Teaming Guide," Confident AI, 02 02 2025.[Çevrimiçi]. Available: https://www.confident-ai.com/blog/red-teaming-llms-a-step-by-step-guide. [Erişildi: 04 03 2025].

[28] mrbullwinkle ve eric-urban, "Planning red teaming for large language models (LLMs) and their applications," Microsoft, 30 11 2024.[Çevrimiçi]. Available: https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/red-teaming. [Erişildi: 04 03 2025].

[29] CISA, "CISA Artificial Intelligence Use Cases," CISA,[Çevrimiçi]. Available: https://www.cisa.gov/ai/cisa-use-cases. [Erişildi: 04 03 2025].