Denetci.org

REFERANS İLKELERİ

Güvenli Yazılım Geliştirme İlkeleri

Rİ-YAZ-1

Sürüm 1.0 · 11.08.2026

www.denetci.org

Güvenli Yazılım Geliştirme İlkeleri

Güvenli yazılım geliştirme, yazılımın güvenlik özelliklerinin yalnızca sonradan bulunan hataların düzeltilmesine değil; tasarımından üretimine, dağıtımından kullanım ömrünün sonuna kadar verilen mühendislik kararlarına dayanmasını ele alır. Amaç belirli bir geliştirme yöntemi, araç seti veya güvenlik testi zorunluluğu tanımlamak değil; yazılımın güven sınırlarının yaşam döngüsü boyunca korunmasını sağlayan kalıcı bir geliştirme anlayışı oluşturmaktır.

İlke 1: Güvenlik yazılım tamamlandıktan sonra eklenen ayrı bir özellik olmamalıdır.

Kimlik, yetki, veri akışı, hata davranışı ve güven sınırları mimari kararlarla şekillenir. Yazılımın temel yapısı bu kararları desteklemiyorsa sonradan eklenen kontroller sınırlı veya kırılgan bir güvence sağlayabilir.

İlke 2: Yazılım yalnızca amaçlanan kullanıma göre tasarlanmamalıdır.

Kullanıcılar hata yapabilir, istemciler değiştirilebilir ve saldırganlar işlevleri tasarım amacının dışında kullanabilir. Makul biçimde öngörülebilen kötüye kullanım ve hata yolları yazılımın güvenlik modelinin parçası olmalıdır.

İlke 3: Güven sınırları yazılımın mimarisinde anlaşılabilir olmalıdır.

Hangi bileşenin hangi kimliğe, veriye veya başka bileşene neden güvendiği görünmez bağlantılar ve varsayımlar arasında kaybolmamalıdır. Kritik güven ilişkileri tasarımın açıklanabilir parçaları olmalıdır.

İlke 4: Değiştirilebilir bileşenlerin uyguladığı güvenlik kararlarına tek başına güvenilmemelidir.

Tarayıcılar, mobil istemciler, kullanıcı cihazları ve benzeri bileşenler saldırgan tarafından değiştirilebilir. Kritik güvenlik kararları, ilgili sınırı gerçekten kontrol eden güvenilir bileşende korunmalıdır.

İlke 5: Dış veri kaynağına göre değil kullanılacağı bağlama göre değerlendirilmelidir.

Veri kullanıcıdan, başka bir iç sistemden veya güvenilir görülen üçüncü taraftan gelebilir; buna rağmen hatalı veya saldırgan kontrollü olabilir. Doğrulama, yalnızca teknik biçimi değil verinin kullanıldığı iş ve yürütme bağlamındaki anlamını da kapsamalıdır.

İlke 6: Veri ile çalıştırılabilir davranış arasındaki ayrım korunmalıdır.

Bir değer sorgu, komut, şablon, dosya yolu veya başka bir yürütme bağlamına girdiğinde uygulamanın sahip olduğu daha geniş gücü devralabilir. Dış veri, yalnızca belirli bir biçimde yazılarak uygulamanın komutuna dönüşmemelidir.

İlke 7: Yetkilendirme gerçek kaynak ve işlem etkisine kadar anlamını korumalıdır.

Bir kullanıcının bir işleve erişebilmesi bütün nesneler, alanlar veya sonuçlar üzerinde aynı yetkiye sahip olması anlamına gelmez. Yetki kontrolü, uygulamanın gerçekte neyi okuyabildiği veya değiştirebildiğine göre tasarlanmalıdır.

İlke 8: Güvenlik açısından başarısızlık erişimi veya güveni fark edilmeden genişletmemelidir.

Bir kimlik, politika, bağımlılık veya güvenlik mekanizması çalışmadığında sistemin davranışı tesadüfen daha izin verici hale gelmemelidir. Hata durumu, kullanılabilirlik ve güvenlik arasındaki bilinçli bir tasarım kararını yansıtmalıdır.

İlke 9: Hata işleme yeni bir bilgi veya yetki kanalı oluşturmamalıdır.

Hatalar sistemin iç yapısı, sırları veya başka hassas bilgileri gereksiz biçimde açığa çıkarabilir; aynı zamanda normal güvenlik akışından farklı çalışma yolları oluşturabilir. Hata davranışı normal çalışma kadar güvenlik modelinin parçasıdır.

İlke 10: Sırlar kaynak kodun veya yazılım paketinin doğal bir parçası olmamalıdır.

Kodun kopyalanması, incelenmesi ve farklı ortamlara taşınması yazılım geliştirme yaşam döngüsünün olağan parçasıdır. Bir sırrın kodla birlikte hareket etmesi bağımsız güven alanlarını gereksiz biçimde birbirine bağlamamalıdır.

İlke 11: Kriptografik mekanizmalar açık ve sınırlandırılmış güven amaçları için kullanılmalıdır.

Kriptografi kullanılması tek başına güvenli tasarım anlamına gelmez. Kullanılan mekanizmanın hangi güvenlik özelliğini sağladığı, anahtarlarının nasıl korunduğu ve başarısızlığının hangi sonuçları doğurduğu anlaşılabilir olmalıdır.

İlke 12: Güvenliğin rastgeleliğe dayandığı durumlarda öngörülebilirlik güven varsayımı olmamalıdır.

Belirteçler, anahtarlar veya benzeri değerlerin güvenliği tahmin edilememelerine dayanıyorsa bunların üretimi olağan amaçlı veya öngörülebilir değerlerden türetilmemelidir. Rastgele görünen değer ile güvenli rastgelelik aynı şey değildir.

İlke 13: Yazılım güvenliği yalnızca tek bir isteğin doğru işlenmesine indirgenmemelidir.

İşlemlerin tekrarı, eşzamanlı gerçekleşmesi, farklı sırada yürütülmesi veya yarıda kalması güvenlik ve veri bütünlüğü sonuçlarını değiştirebilir. Durum taşıyan yazılımlar gerçek işlem yaşam döngüsünü dikkate almalıdır.

İlke 14: Bir işlevin tüketebildiği kaynak da güvenlik etkisinin parçasıdır.

Küçük bir girdi büyük hesaplama, depolama, veri aktarımı veya başka hizmetlerde yoğun çalışma oluşturabilir. Yazılım, tek bir kullanıcı veya bileşenin ortak kaynaklar üzerinde ölçüsüz etki oluşturmasını doğal çalışma biçimi haline getirmemelidir.

İlke 15: Karmaşık veri işleyen bileşenler güven sınırının parçası olarak görülmelidir.

Dosya ayrıştırıcıları, dönüştürücüler, serileştirme mekanizmaları ve benzeri bileşenler dış veriyi daha güçlü iç yapılara dönüştürür. Veri işleme kolaylığı, saldırgan kontrollü girdinin uygulamanın yürütme gücünü devralmasına yol açmamalıdır.

İlke 16: Yazılımın dış sistemlere erişim gücü kullanıcı girdisine sınırsız biçimde devredilmemelidir.

Sunucu tarafındaki yazılım dosyalara, ağlara veya başka hizmetlere erişebilir. İstemcinin sağladığı hedef veya parametre, bu teknik yetkiyi istemcinin serbestçe kullanabileceği genel bir erişim aracına dönüştürmemelidir.

İlke 17: Bağımlılıklar yazılımın gerçek güvenlik modelinin parçasıdır.

Kütüphaneler, çerçeveler, servisler ve araçlar uygulamanın kendi kodu gibi veri işleyebilir ve yetki kullanabilir. Bir bağımlılığın dışarıdan sağlanması onun güvenlik sonuçlarını yazılım ekibinin alanı dışına çıkarmaz.

İlke 18: Bağımlılık eklemek ve sürdürmek sürekli bir güven kararıdır.

Bir bileşenin geçmişte uygun olması gelecekte aynı sürümün veya aynı bağımlılığın gerekli ve güvenilir kalacağını göstermez. Kullanılmayan, desteklenmeyen veya artık anlamlı fayda üretmeyen bağımlılıklar görünmez saldırı yüzeyi olarak yaşamaya devam etmemelidir.

İlke 19: Üretim yazılımının kökeni ve bütünlüğü açıklanabilir olmalıdır.

Üretimde çalışan artefaktın hangi kaynak, bağımlılık ve derleme sürecinden ortaya çıktığı gerektiğinde ilişkilendirilebilmelidir. Geliştiricinin kişisel ortamından veya doğrulanamayan bir süreçten çıkan yazılım, kaynak kodunun güvenilirliğini tek başına miras almamalıdır.

İlke 20: Derleme ve dağıtım zinciri üretim kodu kadar güvenlik açısından önemlidir.

Kaynak kod doğru olsa bile derleme, paketleme veya dağıtım aşamasının ele geçirilmesi farklı bir yazılımın üretime ulaşmasına neden olabilir. Yazılımın güveni yalnızca kaynak deposunun güvenliğine bağlı olmamalıdır.

İlke 21: İnceleme ve test güvenlik düşüncesinin yerine geçen biçimsel aşamalar olmamalıdır.

Kod incelemesi, otomatik tarama veya başka doğrulama yöntemlerinin yapılmış olması tasarımın güvenli olduğunu otomatik olarak göstermez. Bu faaliyetler yazılımın gerçek güvenlik varsayımlarını sorguladıkları ölçüde değer üretir.

İlke 22: Güvenlik bulguları yalnızca kapatılacak kayıtlar olarak görülmemelidir.

Bir hata daha temel bir mimari varsayımın, tekrar eden kod deseninin veya geliştirme yaklaşımının sonucu olabilir. Düzeltmenin amacı yalnızca belirli saldırı örneğini engellemek değil, hatayı doğuran güvenlik zayıflığını gidermek olmalıdır.

İlke 23: Güvenlik düzeltmesi yeni veya daha geniş bir zayıflık oluşturmamalıdır.

Bir saldırıyı engellemek için eklenen karmaşıklık, yetki veya istisna başka güvenlik sınırlarını zayıflatabilir. Düzeltmenin doğruluğu yalnızca eski saldırının artık çalışmamasına göre değerlendirilmemelidir.

İlke 24: Geliştirme kolaylıkları üretimin kalıcı güven modeline dönüşmemelidir.

Tanılama özellikleri, test hesapları, örnek veriler ve geçici güvenlik gevşetmeleri geliştirme sırasında yararlı olabilir. Bu özelliklerin üretim sisteminde kalması geçmiş geliştirme ihtiyacını kalıcı saldırı yüzeyine dönüştürmemelidir.

İlke 25: Yazılımın güvenli biçimde güncellenebilmesi tasarımın parçası olmalıdır.

Yazılımlar zaman içinde yeni zayıflıklar ve değişen bağımlılıklarla karşılaşır. Güvenlik yalnızca ilk sürümün özelliklerine değil, güvenilir bir değişikliğin gerçek kullanıcılara ve sistemlere ulaştırılabilmesine de dayanmalıdır.

İlke 26: Güncelleme mekanizması yazılımın güven sınırlarını aşan bir kod çalıştırma yolu olmamalıdır.

Güncelleme sistemi yeni kodu mevcut yazılımın yetkileriyle çalıştırabilir. Bu nedenle güncellemenin kaynağı ve bütünlüğü, normal üretim yazılımından daha düşük güvenle kabul edilmemelidir.

İlke 27: Destek ömrünün sona ermesi güvenlik durumunu belirsiz bırakmamalıdır.

Güncellenmeyen veya artık desteklenmeyen bir yazılımın süresiz biçimde aynı güven düzeyinde kabul edilmesi gerçekçi değildir. Yazılım yaşam döngüsünün sonu, ona bağlı sistemlerin hangi güven koşullarında devam edebileceğiyle birlikte ele alınmalıdır.

İlke 28: Güvenli yazılım geliştirme tek bir ekip, araç veya yaşam döngüsü aşamasına bağlı olmamalıdır.

Güvenlik gereksinimleri, mimari, kod, bağımlılıklar, derleme, dağıtım ve işletim birbirini etkiler. Tek bir güvenlik taraması veya ayrı bir güvenlik ekibi bütün bu kararların sorumluluğunu tek başına taşıyamaz.

İlke 29: Güvenli geliştirme güvencesi gerçek ürün davranışı üzerinden doğrulanabilmelidir.

Geliştirme standardı, kod inceleme kaydı, güvenlik aracı veya test sonucu tek başına güvence oluşturmaz. Yazılımın gerçek kimlik, yetki, veri, hata, bağımlılık ve güncelleme davranışlarının tasarlanan güven sınırlarını fiilen koruduğu gösterilebilmelidir.