Map Application Architecture (Harita Uygulama Mimarisi)

Summary (Özet)

Birbirine bağlı ve heterojen web altyapısının karmaşıklığı yüzlerce web uygulamasını içerebilir ve yapılandırma yönetimini yapabilir ve her bir uygulamayı test etme ve dağıtmada temel bir adımdır. Aslında, tüm altyapının güvenliğini baltalamak için sadece tek bir güvenlik açığı gerekir ve küçük ve görünüşte önemsiz sorunlar bile aynı altyapıda başka bir uygulama için ciddi risklere dönüşebilir.

Bu sorunları ele almak için, yapılandırma ve bilinen güvenlik sorunlarının derinlemesine bir incelemesini yapmak son derece önemlidir. Derinlemesine bir inceleme yapmadan önce, ağ ve uygulama mimarisinin haritasını çıkarmak gerekir. Altyapıyı oluşturan farklı unsurların, bir web uygulamasıyla nasıl etkileşime girdiklerini ve güvenliği nasıl etkilediklerini anlamak için belirlenmesi gerekir.

Test Objectives (Test Hedefleri)

- Yapılan araştırmaya dayanarak eldeki uygulamanın bir haritasını oluşturun.

How to Test (Nasıl Test Edilir)

Uygulama Mimarisini Haritalayın

Web uygulamasını oluşturmak için hangi farklı bileşenlerin kullanıldığını belirlemek için bazı testler yoluyla uygulama mimarisinin haritalandırılması gerekir. Basit bir PHP uygulaması gibi küçük kurulumlarda, PHP uygulamasına ve belki de kimlik doğrulama mekanizmasına hizmet eden tek bir sunucu kullanılabilir.

Çevrimiçi bir banka sistemi gibi daha karmaşık kurulumlarda, birden fazla sunucu dahil olabilir. Bunlar bir ters proxy, bir ön uç web sunucusu, bir uygulama sunucusu ve bir veritabanı sunucusu veya LDAP sunucusu içerebilir. Bu sunucuların her biri farklı amaçlar için kullanılacak ve hatta aralarında güvenlik duvarları olan farklı ağlarda ayrılabilir. Bu, web sunucusuna erişimin mutlaka kimlik doğrulama mekanizmasının kendisine uzaktan kullanıcı erişimini sağlamayacağı ve

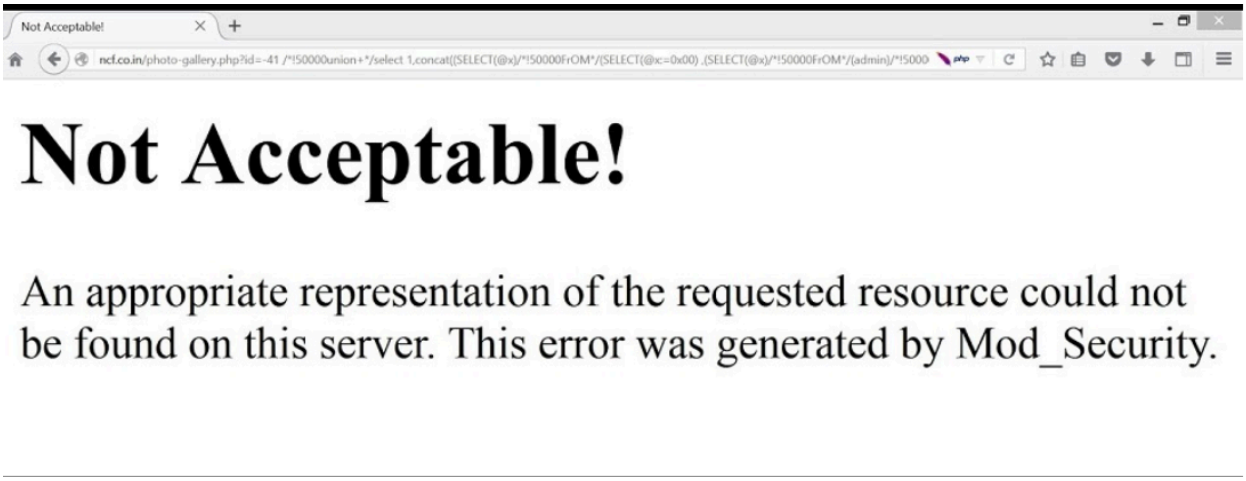
böylece mimarinin farklı unsurlarının ödünlerinin tüm mimariyi tehlikeye atmaması için izole edilebilmesi için farklı ağ bölgeleri oluşturur.

Uygulama mimarisi hakkında bilgi edinmek, bu bilgi uygulama geliştiricileri tarafından belge formunda veya mülakat yoluyla test ekibine sağlanırsa kolay olabilir, ancak aynı zamanda kör bir penetrasyon testi yapılırsa çok zor olduğunu da kanıtlayabilir.

İkinci durumda, bir test cihazı ilk önce basit bir kurulum (tek bir sunucu) olduğu varsayımıyla başlayacaktır. Daha sonra diğer testlerden bilgi alacak ve farklı unsurları elde edecekler, bu varsayımı sorgulayacak ve mimari haritasını genişletecekler. Testçi, “Web sunucusunu koruyan bir güvenlik duvarı var mı?” gibi basit sorular sorarak başlayacaktır. Bu soru, web sunucusuna yönelik ağ taramalarının sonuçlarına ve web sunucusunun ağ bağlantısındaki ağ bağlantılarında filtrelenip filtrelenmediğine (cevap veya ICMP ulaşılamaz maddeler alınmıyor) veya sunucunun doğrudan İnternet'e bağlı olup olmadığına (yani, tüm listelemeyen bağlantı noktaları için RST paketleri iade edip etmediği) analizine göre cevaplanacaktır. Bu analiz, ağ paketi testlerine dayalı olarak kullanılan güvenlik duvarı türünü belirlemek için geliştirilmiştir. Diyafralık bir güvenlik duvarı mı yoksa bir yönlendiricide bir erişim listesi filtresi mi? Nasıl yapılandırılır? Atlayarak atlanabilir mi? Tam bir web uygulaması güvenlik duvarı mı?

Web sunucusunun önünde ters bir proxy tespit etmek, ters bir vekilin varlığını doğrudan açıklayabilecek web sunucusu afişinin analizi ile yapılabilir. Web sunucusunun isteklerine verdiği cevapları elde ederek ve bunları beklenen cevaplarla karşılaştırarak da belirlenebilir. Örneğin, bazı ters proxyler, web sunucusunda hedeflenen bilinen saldırıları engelleyerek İntimsasyon Önleme Sistemleri (IPS) görevi görür. Web sunucusunun, mevcut olmayan bir sayfayı hedef alan ve güvenlik açığı tarayıcıları tarafından yapılanlar gibi bazı yaygın web saldırıları için farklı bir hata mesajı veren bir talebe 404 mesajıyla cevap verdiği biliniyorsa, istekleri filtreleyen ve beklenenden farklı bir hata sayfasını iade eden bir ters proxy'nin (veya uygulama düzeyinde bir güvenlik duvarının) bir göstergesi olabilir. Başka bir örnek: web sunucusu bir dizi mevcut HTTP yöntemini (TRACE dahil) döndürürse, ancak beklenen yöntemler hataları geri döndürürse, muhtemelen onları engellemek arasında bir şey vardır.

Bazı durumlarda, koruma sistemi bile kendini ortadan kaldırır. İşte mod_security benliği tanımlayıcı bir örnek:



Şekil 4.1.10-1: Örnek mod_security Hata Sayfası

Ters proxyler, arka uç uygulama sunucularının performansını hızlandırmak için vekil-kazanç olarak da tanıtılabilir. Bu proxyleri tespit etmek, sunucu başlığına göre yapılabilir. Ayrıca, sunucu tarafından önbelleğe alınması gereken zamanlama istekleri ve ilk isteğin sunucuya alınması gereken süreyi sonraki isteklerle karşılaştırarak algılanabilirler.

Tespit edilebilecek bir diğer unsur ise ağ yük dengeleyicileridir. Tipik olarak, bu sistemler belirli bir TCP / IP bağlantı noktasını farklı algoritmalara (yuvarlak robin, web sunucusu yükü, istek sayısı vb.) dayalı birden fazla sunucuya dengeleyecektir. Böylece, bu mimari elemanının tespit edilmesi, taleplerin aynı veya farklı web sunucularına gidip gitmediğini belirlemek için birden fazla isteği inceleyerek ve sonuçları karşılaştırarak yapılmalıdır. Örneğin, sunucu saatleri senkronize edilmezse Tarih başlığına göre. Bazı durumlarda, ağ yük dengesi süreci, başlıklara, F5 BIG-IP yük dengeleyicileri tarafından tanıtılan BIGipServer ön tarifeli çerez gibi belirgin bir şekilde öne çıkmasını sağlayacak yeni bilgiler enjekte edebilir.

Uygulama web sunucularının tespit edilmesi genellikle kolaydır. Birkaç kaynak talebi uygulama sunucusunun kendisi tarafından ele alınır (web sunucusu değil) ve yanıt başlığı önemli ölçüde değişecektir (cevap başlığındaki farklı veya ek değerler dahil). Bunları tespit etmenin bir başka yolu, web sunucusunun bir uygulama web sunucusunun (çeşitli J2EE sunucuları tarafından sağlanan JSESSIONID gibi) bir uygulama web sunucusunun göstergesi olan çerezleri ayarlamaya çalışıp çalışmadığını veya oturum takibi yapmak için URL'leri otomatik olarak yeniden yazmaya çalışıp çalışmadığını görmektir.

Bununla birlikte, doğrulama geri uçları (LDAP dizinleri, ilişkisel veritabanları veya RADIUS sunucuları gibi), harici bir bakış açısıyla derhal tespit edilmesi kolay değildir, çünkü uygulamanın kendisi tarafından gizleneceklerdir.

Bir arka uç veritabanının kullanımı, bir uygulamada gezinmek suretiyle belirlenebilir. “Flanında” üretilen son derece dinamik içerik varsa, muhtemelen uygulamanın kendisi tarafından bir tür veritabanından çıkarılmaktadır. Bazen bilginin talep edilme şekli, bir veritabanının geri ucunun varlığına dair bir fikir verebilir. Örneğin, sayısal tanımlayıcıları kullanan bir çevrimiçi alışveriş uygulaması (`id`) Dükkandaki farklı makalelerde gezinirken. Bununla birlikte, kör bir uygulama testi yaparken, altta yatan veritabanının bilgisi genellikle yalnızca bir güvenlik açığı uygulamada, kötü istisna işleme veya SQL enjeksiyonuna duyarlılık gibi yüzeye çıktığında kullanılabilir.