1. Yansıtılan Siteler Arası Script Çalıştırma / XSS (OWASP-DV-001)

Önem Derecesi: Yüksek
Açıklığın Etkisi: Yetkisiz Erişim, Bilgi İfşası
Erişim Noktası: İnternet
Kullanıcı Profili: Anonim Kullanıcı
Bulgu Kategorisi: Web
Bulgu Sebebi: Uygulama Geliştirmedeki Eksiklikler/Hatalar

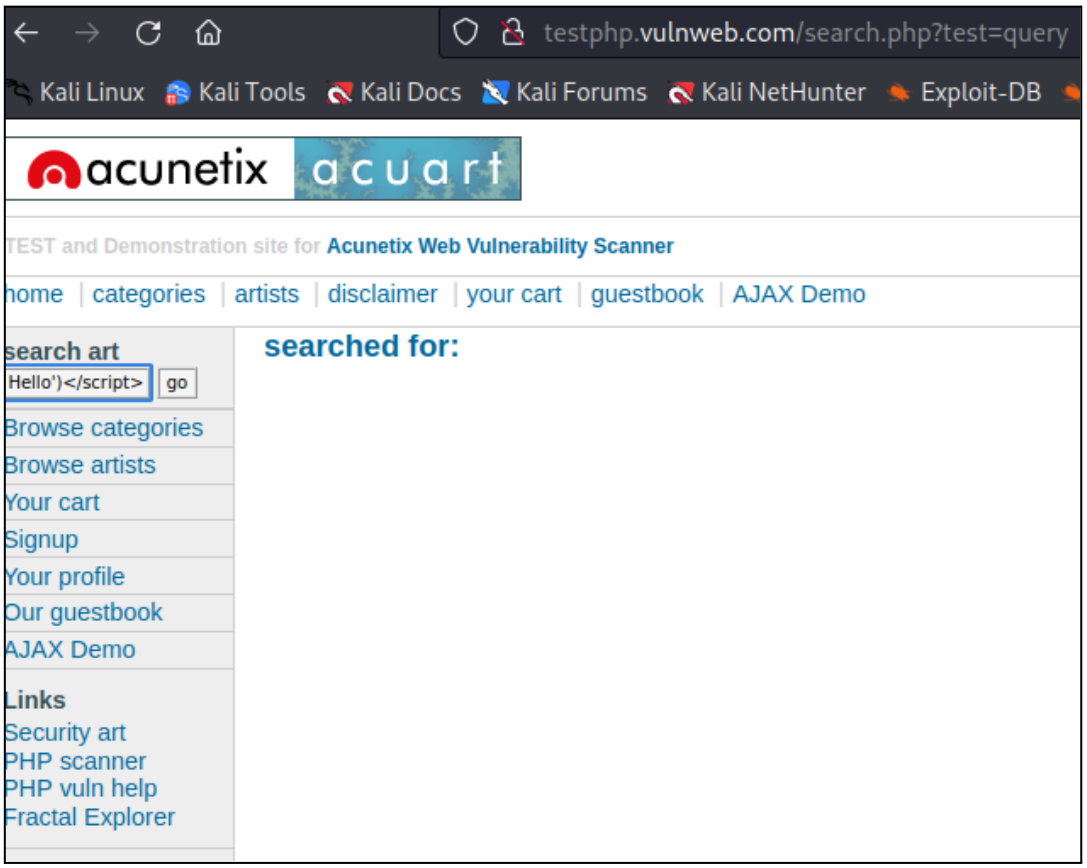
Bulgu Açıklaması:
Reflected XSS: Kalıcı olmayan XSS olarak da bilinen reflected XSS siber saldırısında, bilgisayar korsanları kötü amaçlı komut dosyasını doğrudan bir HTTP isteğine enjekte eder. Ardından, web sunucusundan yürütüldüğü kullanıcının tarayıcısına yansıtır. Bilgisayar korsanı sıklıkla hedeflenen kişilere, onları savunmasız bir sayfaya getiren özelleştirilmiş bağlantılar gönderir.

Reflected XSS saldırıları kalıcı değildir. Bir kullanıcı kötü niyetli bir bağlantıyı tıkladığında, özel olarak hazırlanmış bir formun göndermesi veya kötü niyetli bir siteye göz atması için kandırıldığında, enjekte edilen kod savunmasız web sitesine gider. Web sunucusu, sırayla, enjekte edilen komut dosyasını kullanıcının tarayıcısına döndürür veya yansıtır. Bu aldatma, bir hata mesajında, arama sonucunda veya isteğin bir parçası olarak sunucuya gönderilen verileri içeren başka bir yanıt türünde olabilir. Tarayıcı, yanıtın, kullanıcının zaten etkileşimde bulunduğu “güvenilir” bir sunucudan geldiğini varsaydığı için kodu yürütür.

Bulgu 1:

URL: http://testphp.vulnweb.com/
HTTP Talep Türü: POST
Parametre:
Payload: <script>alert('hello')</script>

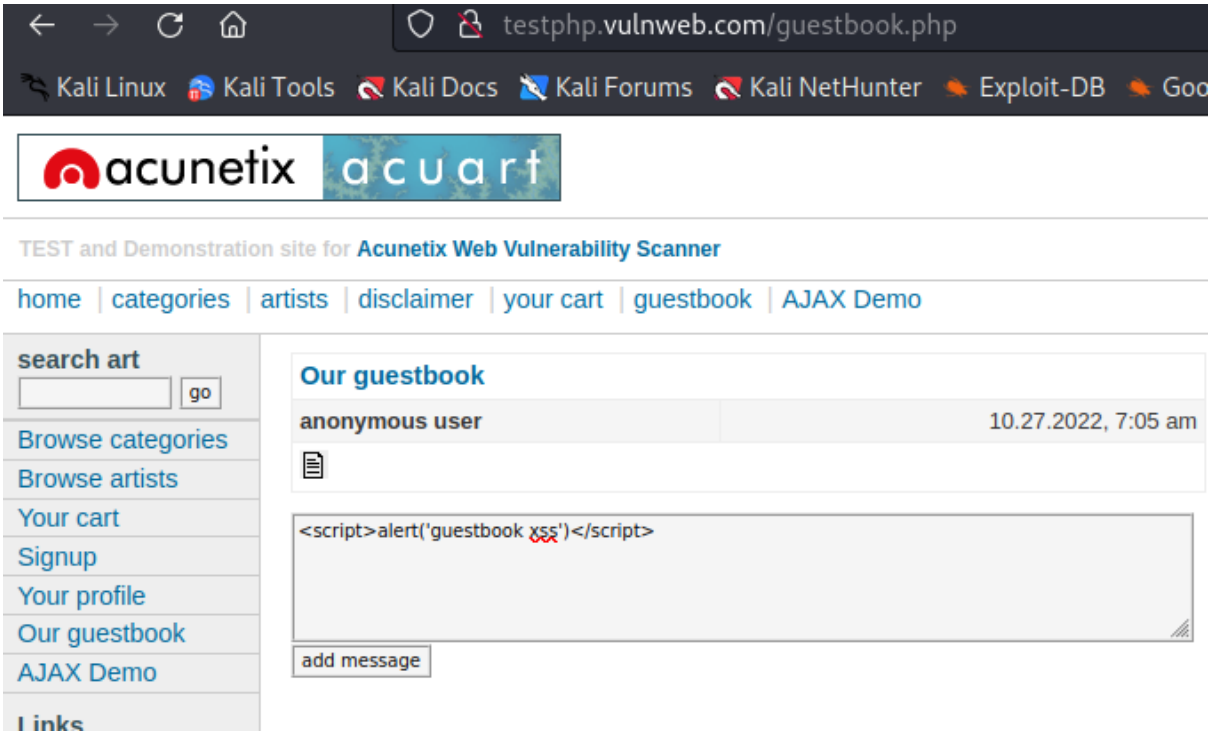
Ekran Görüntüleri:

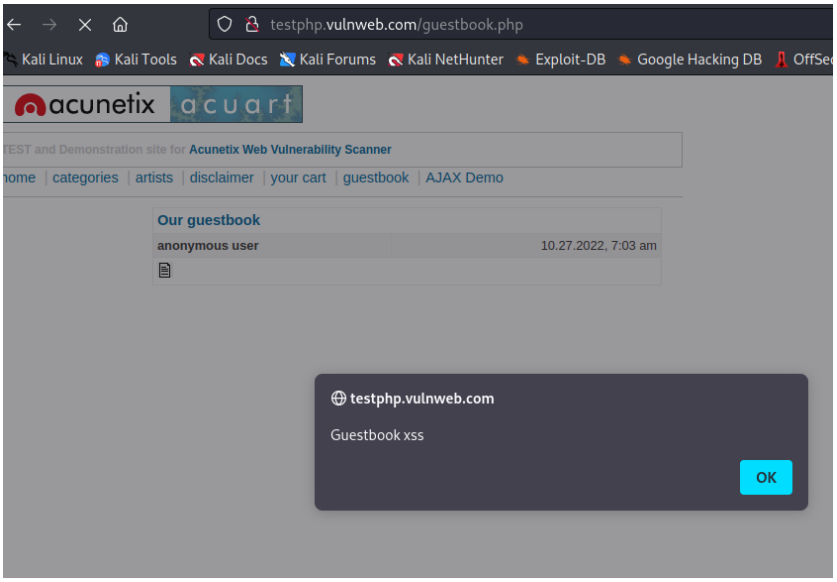


Bulgu 2:

URL: http://testphp.vulnweb.com/guestbook.php
HTTP Talep Türü: POST
Parametre:
Payload: <script>alert(guestbook xss)</script>

Ekran Görüntüleri:

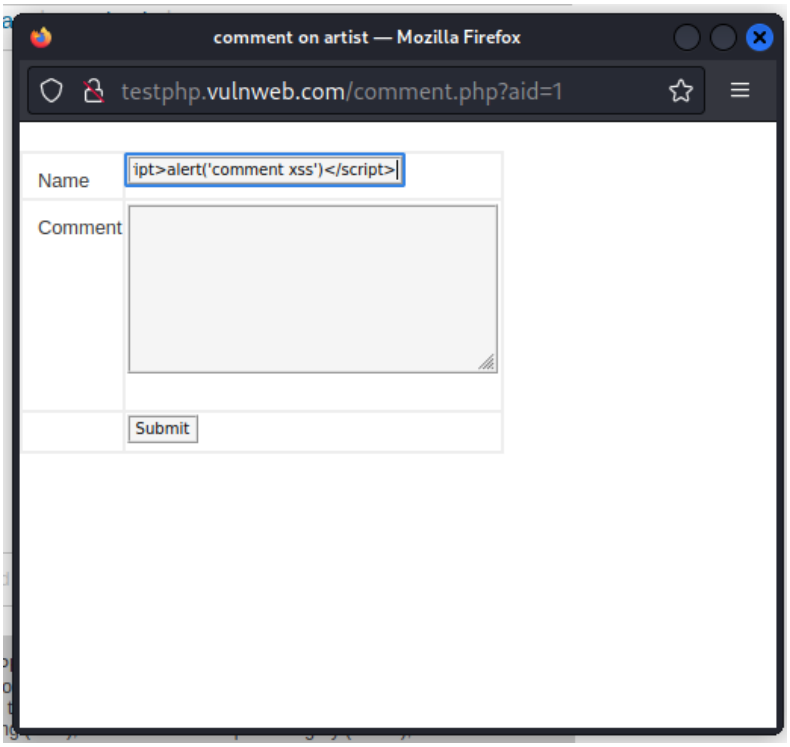


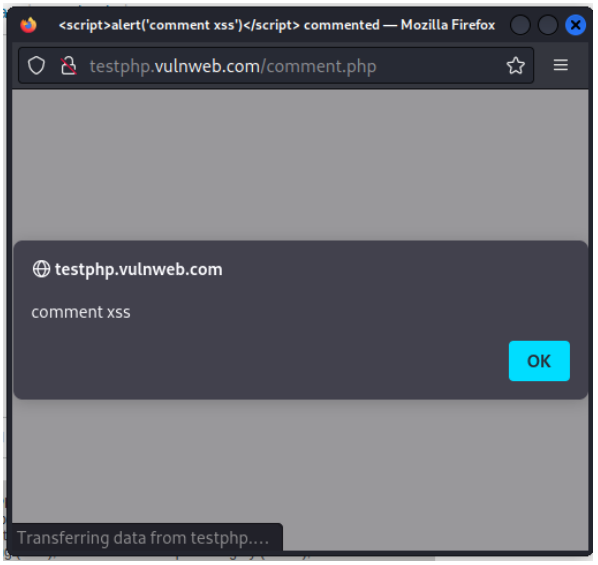


Bulgu 3:

URL: http://testphp.vulnweb.com/comment.php
HTTP Talep Türü: POST
Parametre:
Payload: <script>alert(comment xss)</script>

Ekran Görüntüleri:





Açıklığı Barındıran Sistemler:
http://testphp.vulnweb.com/

http://testphp.vulnweb.com/guestbook.php

http://testphp.vulnweb.com/comment.php?aid=1
http://testphp.vulnweb.com/comment.php?aid=2
http://testphp.vulnweb.com/comment.php?aid=3
… (aid için herhangi bir sayı verilebilir.)

Çözüm Önerileri:
Uygulama kodları gözden geçirilerek parametreler ve HTTP başlığındaki diğer alanlar vasıtası ile yollanan her türlü bilginin kullanılmadan önce zararlı karakterlerden filtrelenmesi önerilmektedir. Bütün girdi ve çıktı noktaları kontrol edilmelidir ve meta karakterler filtrelenmelidir.

Referanslar:
https://bulutistan.com/blog/xss-cross-site-scripting-nedir/
https://owasp.org/www-project-web-security-testing-guide/assets/archive/OWASP_Testing_Guide_v3.pdf
https://portswigger.net/web-security/cross-site-scripting/preventing

2. SQL Injection Zafiyeti (OWASP-DV-005)

Önem Derecesi: Acil
Açıklığın Etkisi: Yetkisiz Erişim, Bilgi İfşası
Erişim Noktası: İnternet
Kullanıcı Profili: Anonim Kullanıcı
Bulgu Kategorisi: Web
Bulgu Sebebi: Uygulama Geliştirmedeki Eksiklikler/Hatalar

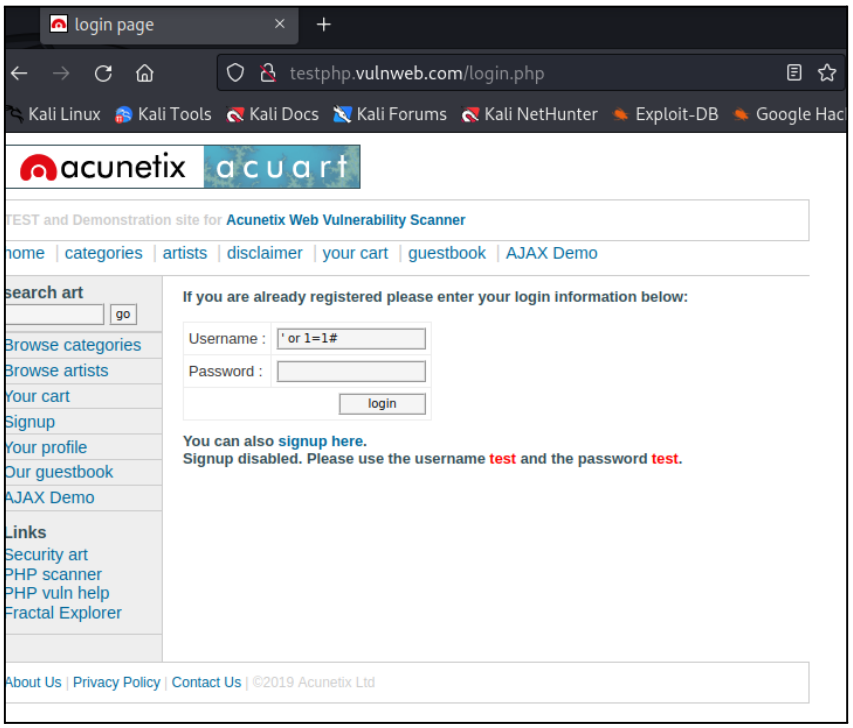
Bulgu Açıklaması:
SQL enjeksiyonu, veri tabanına dayalı uygulamalara saldırmak için kullanılan bir atak tekniğidir; burada saldırgan SQL dili özelliklerinden faydalanarak standart uygulama ekranındaki ilgili alana yeni SQL ifadelerini ekler. (Örneğin saldırgan, veritabanı içeriğini kendisine aktarabilir).

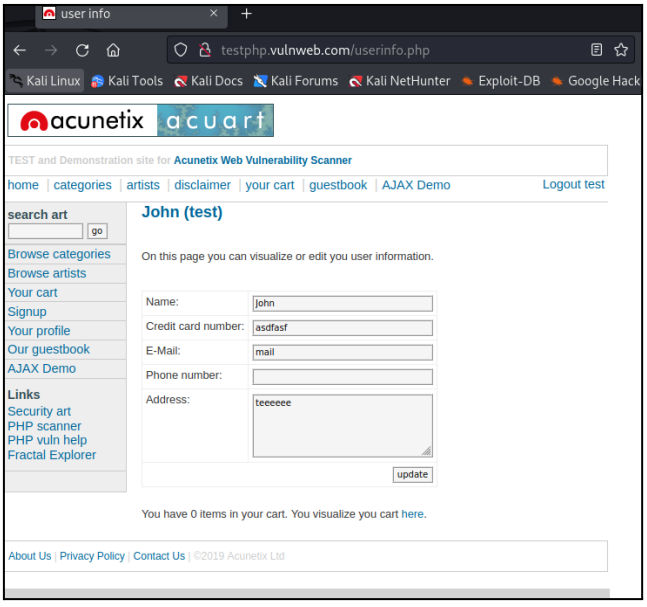
SQL enjeksiyonu, uygulamaların yazılımları içindeki bir güvenlik açığından faydalanır, örneğin, uygulamanın kullanıcı giriş bilgileri beklediği kısma SQL ifadeleri gömülür, eğer gelen verinin içeriği uygulama içerisinde filtrelenmiyorsa veya hatalı şekilde filtreleniyorsa, uygulamanın, içine gömülmüş olan kodla beraber hiçbir hata vermeden çalıştığı görülür. SQL enjeksiyonu, çoğunlukla web siteleri için kullanılan bir saldırı türü olarak bilinse de SQL veri tabanına dayalı tüm uygulamalarda gerçeklenebilir.

BULGU 1:

URL: http://testphp.vulnweb.com/login.php
HTTP Talep Türü: POST
Parametre:
Payload: ‘or 1=1#

Ekran Görüntüleri:

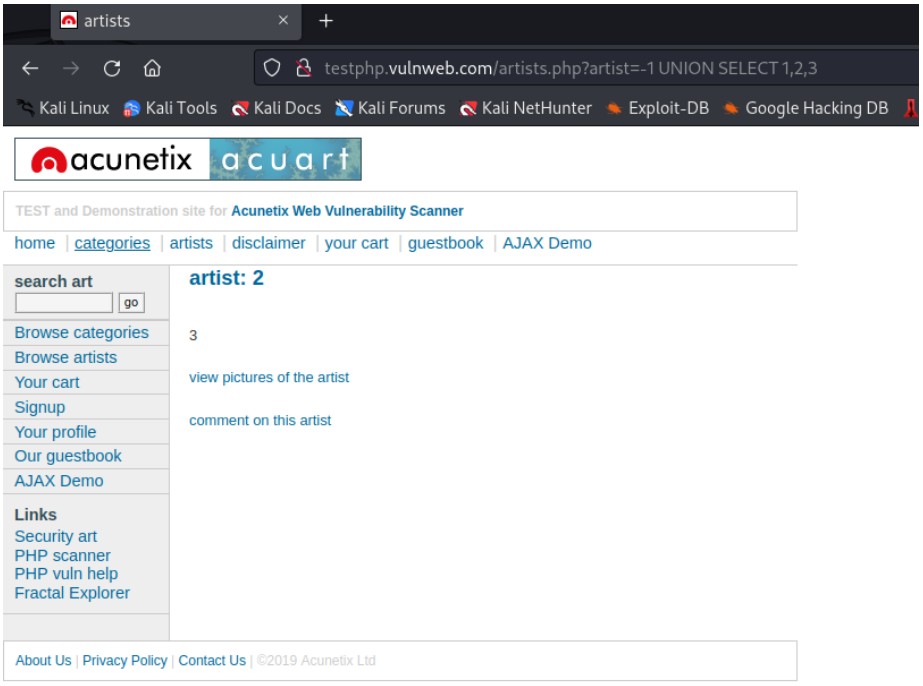


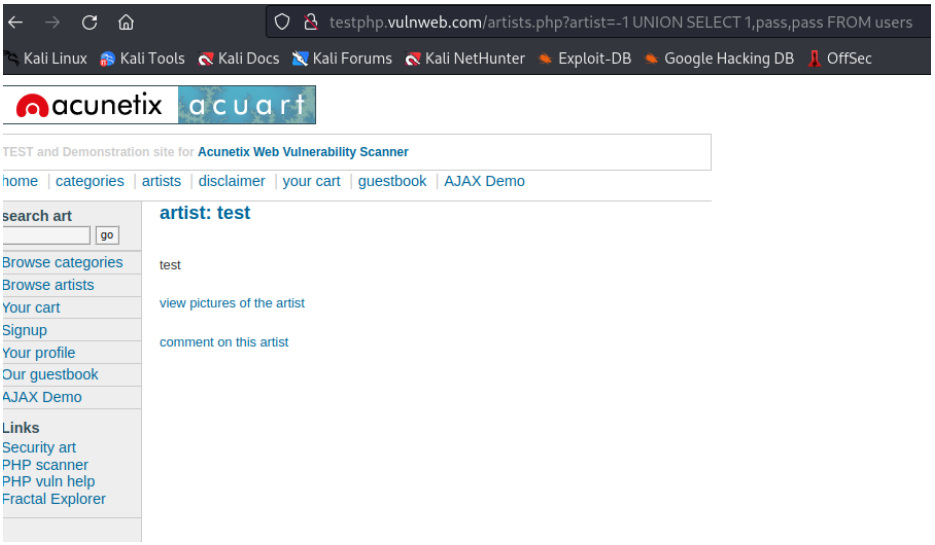


BULGU 2 (Referanslara SQLi engellemek için gerekenleri eklerken karşılaştım bununla. https://www.acunetix.com/websitesecurity/sql-injection/ bu sitede.):

URL: http://testphp.vulnweb.com/artists.php?artist=2
HTTP Talep Türü: GET
Parametre:
Payload1: -1 UNION SELECT 1,2,3
Payload2: -1 UNION SELECT 1,pass,pass FROM users

Ekran Görüntüleri:





Açıklığı Barındıran Sistemler:
http://testphp.vulnweb.com/login.php

http://testphp.vulnweb.com/artists.php?artist=2
(3 farklı artist ile de çalışıyor (1-2-3))

Çözüm Önerileri:
Uygulama kodlarının gözden geçirilerek parametreler ve http başlığındaki diğer alanlar vasıtası ile yollanan her türlü bilginin kullanılmadan önce zararlı karakterlerden filtrelenmesi önerilmektedir.
Uygulamalardaki bütün girdi noktalarından gelen değişkenler girdi kontrolüne sokulmalı ve bu girdilerdeki bütün meta karakterlerin filtrelenmesi önerilmektedir. Detaylı SQL enjeksiyonu önleme yöntemleri için aşağıda belirtilen referanslar incelenebilir.

Referanslar:
https://tr.wikipedia.org/wiki/SQL_Enjeksiyonu
https://owasp.org/www-project-web-security-testing-guide/assets/archive/OWASP_Testing_Guide_v3.pdf
https://www.acunetix.com/websitesecurity/sql-injection/

3. IDOR Zafiyeti

Önem Derecesi: Kritik
Açıklığın Etkisi: Yetkisiz Erişim, Maddi Zarar
Erişim Noktası: İnternet
Kullanıcı Profili: Herhangi Bir Kullanıcı
Bulgu Kategorisi: Web
Bulgu Sebebi: Uygulama Geliştirmedeki Eksiklikler/Hatalar

Bulgu Açıklaması:
Bir websitesi ziyaret edildiğinde içeriğinde bulunan uygulamalara nesneler üzerinden erişim sağlanır. Bu nesneler; veritabanı, dosyalar ve dizinlere erişim gibi önemli durumları da tanımlamakta kullanılmaktadır. Saldırganlar bir başka kullanıcının sahip olduğu nesne değerlerlerini taklit veya manipüle edebilmektedir. Böylelikle hedeflediği kişinin uygulama üzerindeki kimlik bilgilerini elde etmiş olurlar.

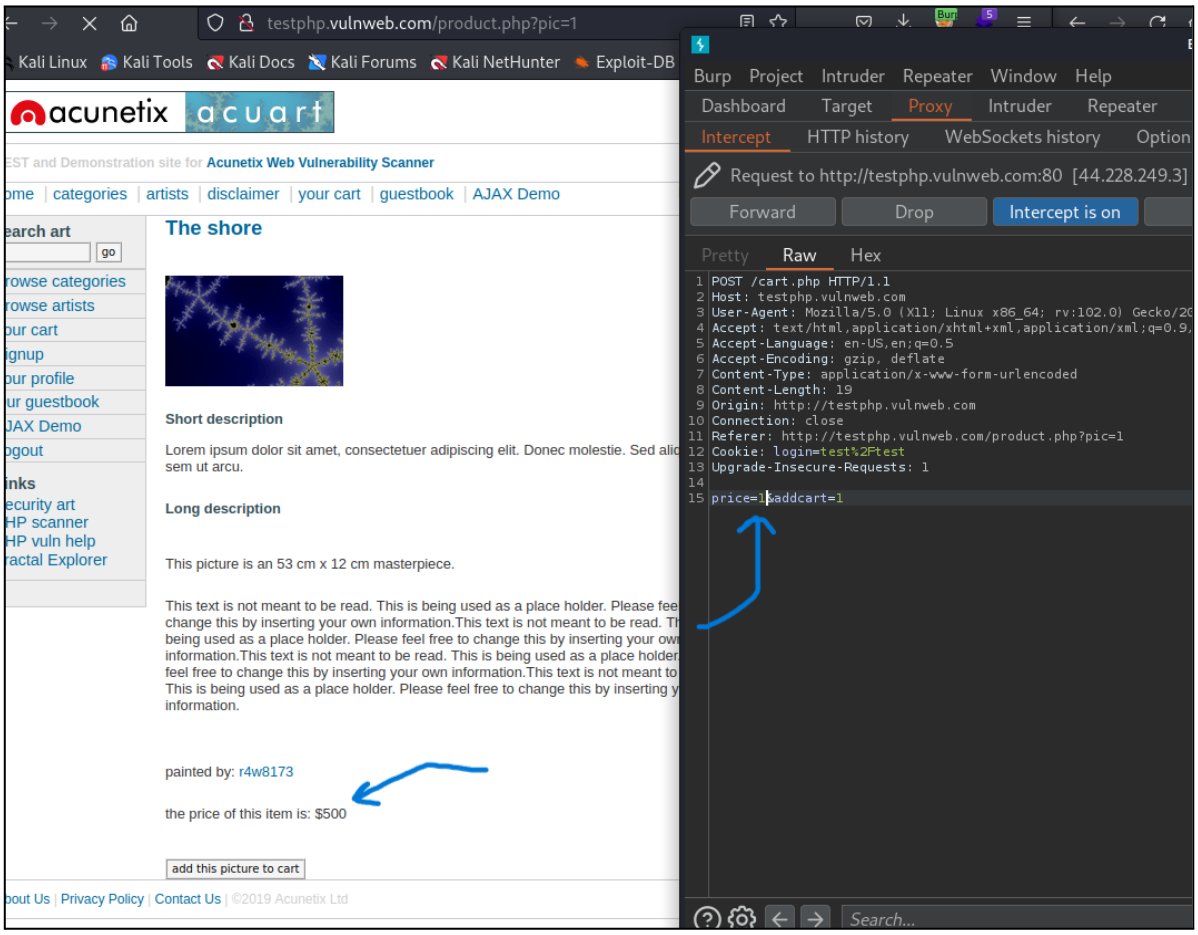
Bu saldırı yöntemi IDOR (Insecure Direct Object References) olarak tanımlanır. IDOR zafiyetinin Türkçe karşılığını “Güvensiz Nesnelere Yönelim” olarakta çevirmek mümkündür. IDOR zafiyeti, OWASP’ın ilk olarak 2013 yılında açıkladığı en sık görülen Top 10 zafiyet listesinde 4. sırada yerini almıştır. 2017 yılındaki açıklamada ise A5 kategorisinde kendisine yer bularak, Broken Access Control zafiyet türünde kendine yer edinmiştir ve sızma testi çalışmaları dahilinde sıklıkla test edilen bir güvenlik açıklığıdır.

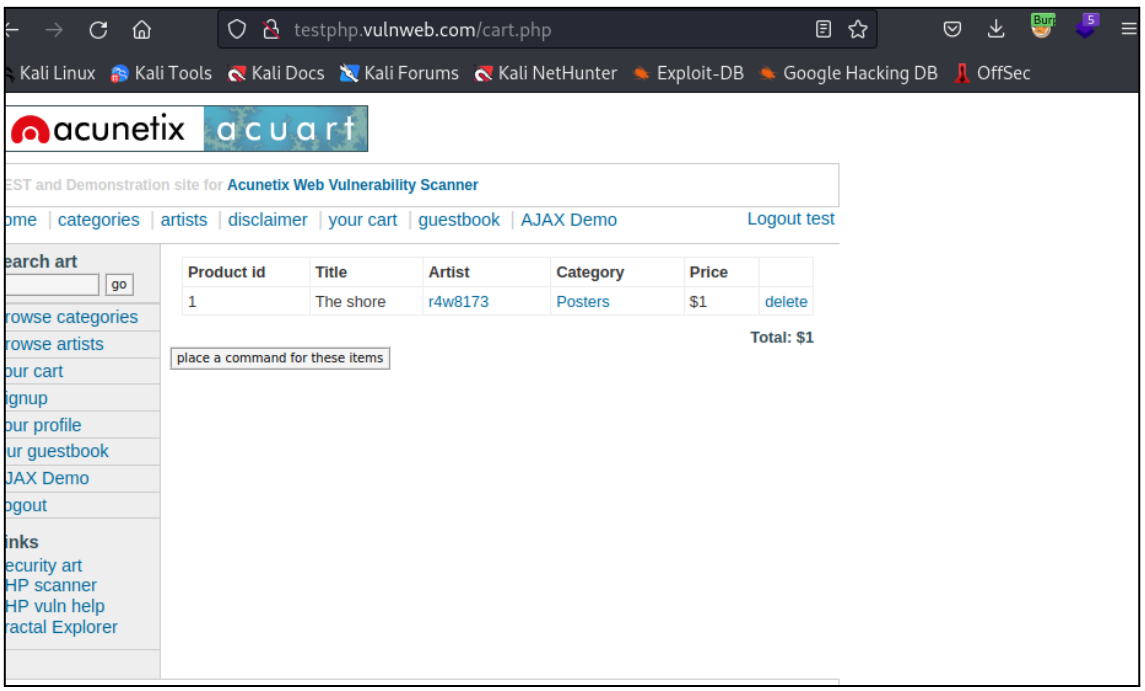
BULGU:

URL: http://testphp.vulnweb.com/product.php?pic=1
HTTP Talep Türü: POST
Parametre:
Payload: Changing the price from Burp Suite.

Bu sitedeki örneğe göre, bu yöntemle normalde pahalı olan ürünleri bedavaya ya da çok ucuza alma şansımız oluyor.

Ekran Görüntüleri:





Açıklığı Barındıran Sistemler:
http://testphp.vulnweb.com/product.php?pic=1
(Herhangi bir product ile de çalışıyor.)

Çözüm Önerileri:
Web uygulamaları, her HTTP isteğiyle alınan tüm güvenilmeyen girdileri doğrulamalıdır. Uygulamalarınız, gelen değerin uygulamalarınızın aşağıdakilerle ilgili beklentilerini karşıladığını doğrulamak için her girişte "beyaz liste doğrulaması" gerçekleştirmelidir:

- Minimum veya maksimum uzunluk
- Sayısal değerler için minimum veya maksimum sınırlar
- Kabul edilebilir karakterler
- Dize, tarih, tamsayı veya rasyonel gibi Veri Türleri
- Üyelik ayarla
- Telefon numarası, sosyal güvenlik veya işveren kimliği gibi kalıplar

IDOR'larla savaşmanın başka bir yolu da, kimlikler, adlar ve anahtarlar gibi kaynakları kriptografik olarak güçlü rastgele değerlerle değiştirilecek şekilde tasarlamaktır. Bu değerler orijinal değerlere karşılık gelir ve her ikisi de sunucuda bulunur, böylece bir uygulama doğrudan bir referans gösteremez. Bu dolaylı referanslar, mantıksal doğrulamadan daha karmaşık bir karşı saldırı metodolojisi sağlayarak, bilgisayar korsanlarının referanslar için anlamlı değerleri değiştirmesini zorlaştırır.

Daha fazlası için referanslar incelenebilir.

Referanslar:
https://www.infinitumit.com.tr/idor-insecure-direct-object-references-zafiyeti-nedir-ve-nasil-onlenir/
https://owasp.org/www-project-web-security-testing-guide/assets/archive/OWASP_Testing_Guide_v3.pdf
https://cheatsheetseries.owasp.org/cheatsheets/Insecure_Direct_Object_Reference_Prevention_Cheat_Sheet.html
https://avatao.com/blog-best-practices-to-prevent-idor-vulnerabilities/

4. Depolanan Siteler Arası Script Çalıştırma (Stored XSS)

Önem Derecesi: Kritik
Açıklığın Etkisi: Yetkisiz Erişim, Bilgi İfşası
Erişim Noktası: İnternet
Kullanıcı Profili: Herhangi Bir Kullanıcı
Bulgu Kategorisi: Web
Bulgu Sebebi: Yapılandırma Eksikliği/Hatası

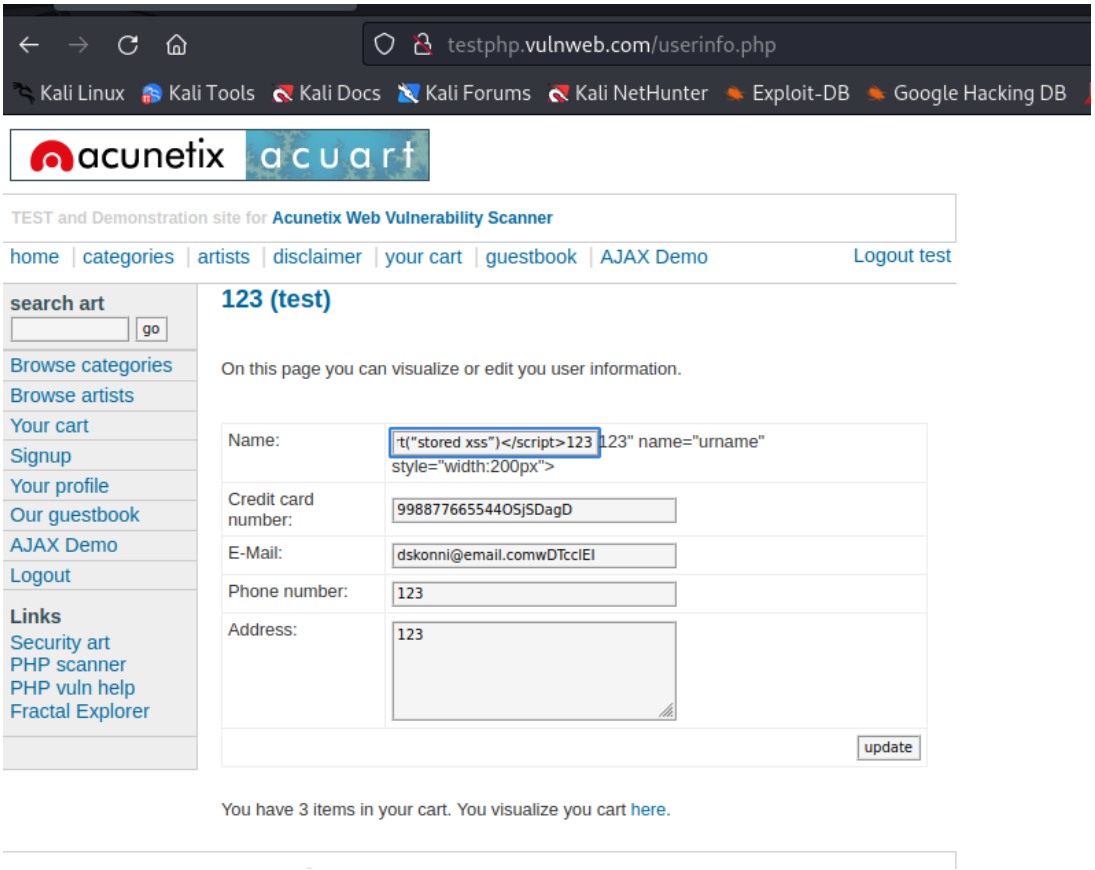
Bulgu Açıklaması:
Bilgisayar korsanları yüklerini güvenliği ihlal edilmiş bir sunucuda depoladığında saldırılar gerçekleşir. Genellikle zarar veren bir XSS saldırı yöntemidir. Saldırgan, yüklerini hedef uygulamaya enjekte etmek için bu yaklaşımı kullanır. Uygulamanın giriş doğrulaması yoksa, kötü amaçlı kod, uygulama tarafından veri tabanı gibi bir konumda kalıcı olarak depolanır veya kalıcı olur. Pratikte bu, saldırganın bir blog veya forum gönderisindeki yorum bölümleri gibi kullanıcı giriş alanlarına kötü amaçlı bir komut dosyası girmesine olanak tanır.

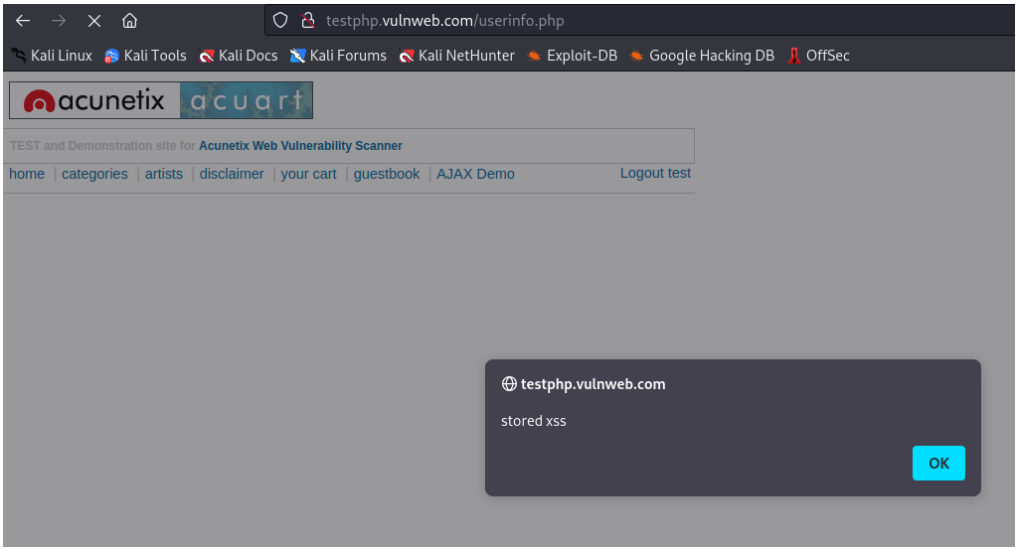
Saldırganın yükü, virüslü sayfayı açtığında, tarayıcısında meşru bir yorumun görünmesiyle aynı şekilde, kullanıcının tarayıcısına sunulur. Hedeflenen kişiler, sayfayı tarayıcılarında görüntülediklerinde yanlışlıkla kötü amaçlı komut dosyasını yürütürler.

Bulgu:

URL: http://testphp.vulnweb.com/userinfo.php
HTTP Talep Türü: POST
Parametre:
Payload: <script>alert(“stored xss”)</script>123

Ekran Görüntüleri:





Açıklığı Barındıran Sistemler:
http://testphp.vulnweb.com/userinfo.php

Çözüm Önerileri:
Uygulama kodları gözden geçirilerek parametreler ve HTTP başlığındaki diğer alanlar vasıtası ile yollanan her türlü bilginin kullanılmadan önce zararlı karakterlerden filtrelenmesi önerilmektedir. Bütün girdi ve çıktı noktaları kontrol edilmelidir ve meta karakterler filtrelenmelidir.

Referanslar:
https://bulutistan.com/blog/xss-cross-site-scripting-nedir/
https://owasp.org/www-project-web-security-testing-guide/assets/archive/OWASP_Testing_Guide_v3.pdf
https://portswigger.net/web-security/cross-site-scripting/preventing