ZAYIF WEB UYGULAMASI

Oğuzhan YILMAZ Dilara TAŞ

Bilgisayar Mühendisliği Bölümü

Kocaeli Üniversitesi

oguzhanyilmaz1996@hotmail.com ve ddilaratas28@gmail.com

Özet

Projemizi geliştirirken farklı internet sayfalarını kaynak olarak kullandık. Daha öncesinden internet sayfası yapma deneyimine sahip olduğumuz için bizim açımızdan daha avantajlı olduğunu söyleyebiliriz.

Tasarım için template yöntemini kullandık. Birer adet header ve footer php dosyası oluşturup her sayfaya bunu implemente ettik. Her zaafiyet için farklı sayfa oluşturduk. Her zaafiyetin kullanabileceği ve sömürülebileceği formlar oluşturduk. Web uygulamamız her zaafiyet tarafından sömürülebildi. Ve bunu zaafiyet sayfalarında info butonuyla açılan yeni sekmede açıkladık.

# Giriş

Web uygulamasını oluştururken html,css,php dillerini kullandık.

Editör olarak notepad ++ ı tercih ettik. Sql için mysql i, sql sorgusu yapmak ve veri tabanını kontrol etmek için SQLyog Enterprise ı kullandık.

# Temel Bilgiler

Sunucu kurulumu için xammp içerisinde apache yi tercih ettik. Bunun haricinde başka bir teknolojiden yararlanma gereksinimi duymadık.

# Geliştirilen Mimari

İndex.php ana sayfamızda template yöntemiyle header ve footer ile tasarımı elde ettik ve web uygulaması hakkında kısa bir açıklama yaptık. Tasarımda her zafiyet için zafiyetler sekmesi içerisinde bir açılır menüde tüm zafiyetler için birer sayfa oluşturduk. Bunun dışında kurulum ve ana sayfa sekmelerini de ekledik.

Kurulum kısmında sadece bilgisayar kullanmayı bilen bir kişi için bile kolay bir şekilde uygulamayı kurmasına yardımcı olacak bir dökümantasyon yaptık.

Web uygulamasının açılabilmesi için gerekecek sunucu programının ve sql bileşenlerinin kurulumunu basit bir şekilde izah ettik.

Sql enjeksionu sayfasında kullanıcı adı ve parolayla gizlenen harici bir sayfa için giriş formu oluşturduk. Sql enjeksiyonu ile ilgili kısa bir açıklama yaptık. Bir info butonuyla açılan pencerede zaafiyetin ve sömürüsünün web uygulamamızda nasıl kullanılabileceğini detaylı bir şekilde anlattık. Web uygulamamız için zararlı olabilecek meta karakterlerin ve komutların çalışabildiğini test ettik. Sonuç olarak web uygulamamızda sql enjeksiyonu zafiyetinin var olduğu konusunda karar kıldık.

Xss sayfamız için için bir ziyaretçi defteri dizayn ettik. Bu ziyaretçi defterinde kullanıcıdan isim ve mesaj alıp bunları veri tabanına kaydettirdik. Ve sayfa her güncellendiğinde veri tabanından isim mesaj ve tarih parametrelerini çekip tablo şeklinde ekrana bastık. Ve burada kullanıcı tarafından isim veya mesaj textbox una bir javascript kodu veya site sahibi tarafından istenmeyecek zararlı komutla yerleştirildiğinde sayfaya giren başka bir kullanıcının daha önce yazılmış olan zararlı kodlarla karşılaştığını gözlemledik.

Örneğin sayfaya giren kullanıcıya bildirim veren bir kodu mesaj kısmına bıraktığımızda daha sonra sayfaya uğrayan herkesin bu bildirimlerle karşılaştığını gördük ve bunları göz önünde bulundurarak web uygulamamızın xss zaafiyetine sahip olduğunu gördük. Bunun çözümünü ve programımızda nasıl kullanıldığını, buna hangi kod bloklarımızın sebep olduğunu sayfa içerisindeki info butonuyla açılan pencerede açıkladık.

Komut enjeksiyonu sayfamızda iki farklı yöntem kullandık. Bunlardan biri kullanıcıdan bir yazı ve dosya ismi alan textboxlarla kaydet butonuna basıldığında sistem klasörüne kullanıcının doldurduğu bir dosya kaydedilir. Diğer yöntemde ise kullanıcı tarafından girilen bir IP adresi veya web sitesine pıng gönderme işlemi yapılır.

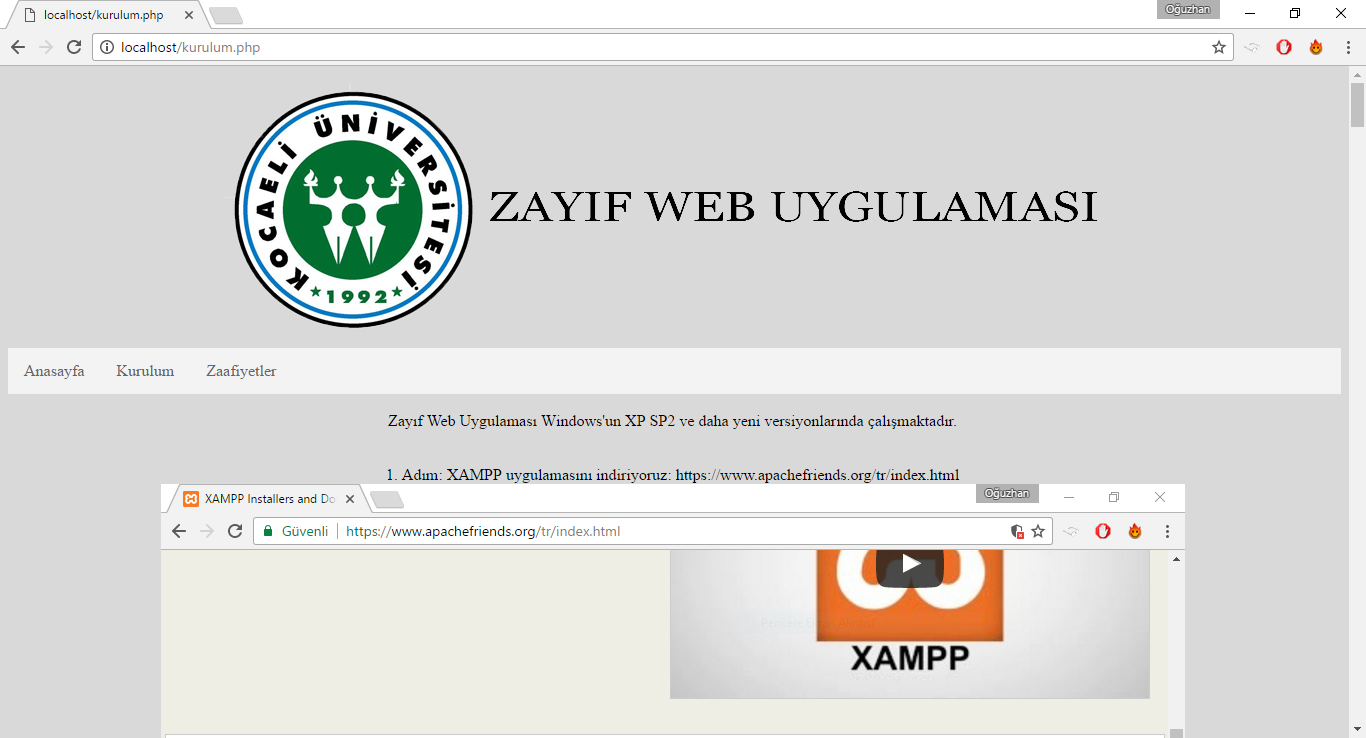
Birinci yöntemde dosya ismi ikinci yöntemde ise pıng gönderilecek adresin bulunduğu textboxlara asıl verilerden sonra komut ekranı için bir meta karakter olan ’ | ’ işareti ve ardından komut ekranında kullanılabilecek herhangi bir komut yazdıktan sonra sunucuya istenilen her şeyi yaptırabildiğini gözlemledik. Çünkü kod içerisinde kullandığımız exec komutu istemci tarafında değil sunucu tarafında çalışmaktadır. İki form için de komutların sonucunu ekrana yazdırdığımız için üçüncül bir kullanıcı sunucumuza istediği işlemi yaptırıp sonucunu ekranda görebilmektedir.

Komut enjeksiyonu sayfasında bu işlemleri yaptıktan sonra uygulamamızın komut enjeksiyonu zafiyetine de sahip olduğunu gördük. Bir açılır pencerede daha bu zafiyetle ve uygulamamız içerisinde nasıl çalıştığıyla ilgili açıklama yaptık. Bu zafiyetin çözümünden de açıklamamızda bahsettik.

# Kullanıcı Kataloğu



1. Uygulamamızın anasayfası.



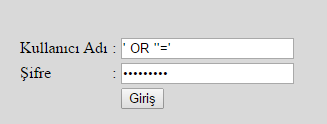
1. Uygulamamızın detaylı kurulum sayfası



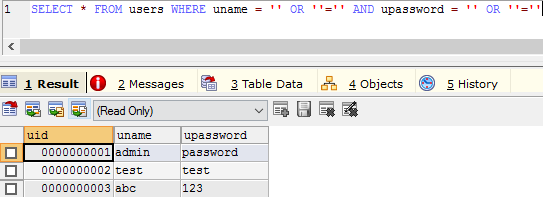
* 1. SQL Injection’ı test edebildiğimiz örnek bir login ekranı



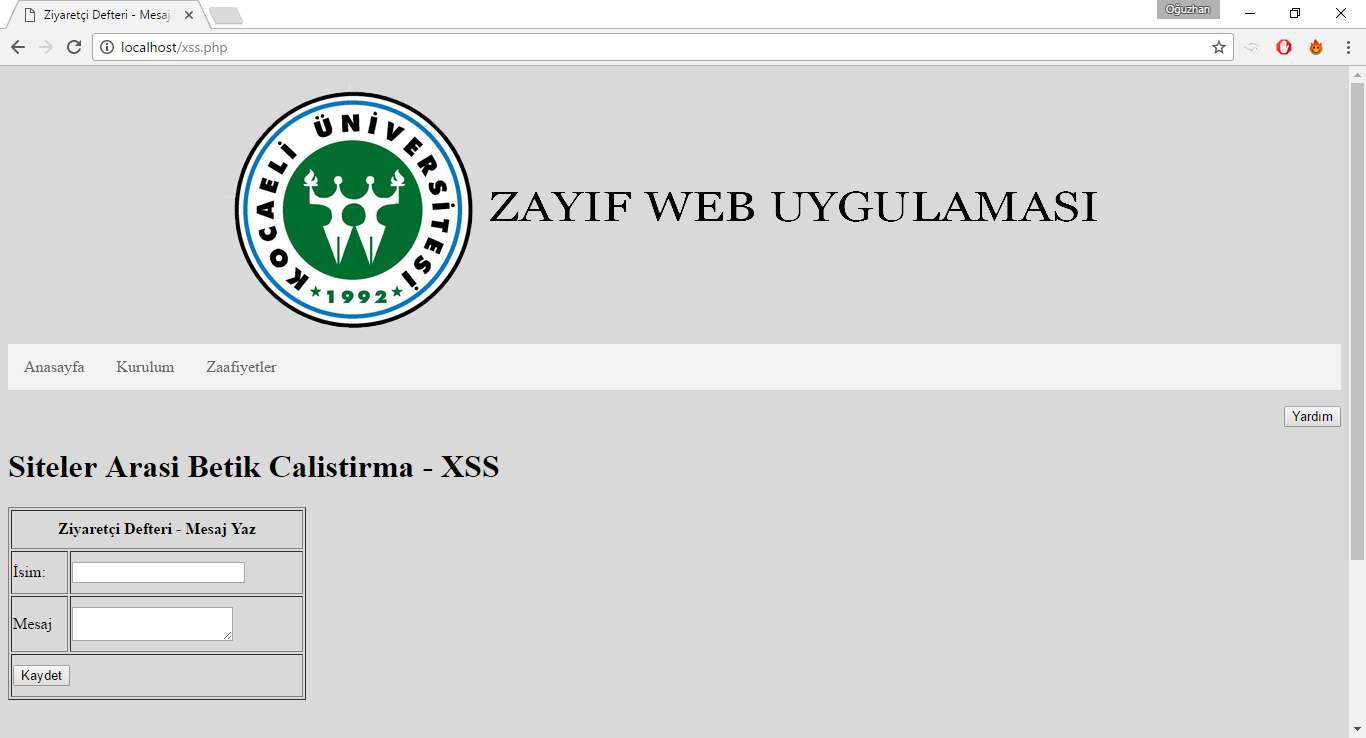
* 1. Login bilgilerinin kontrol edildiği kod bloğu



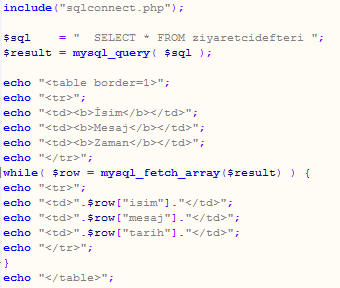
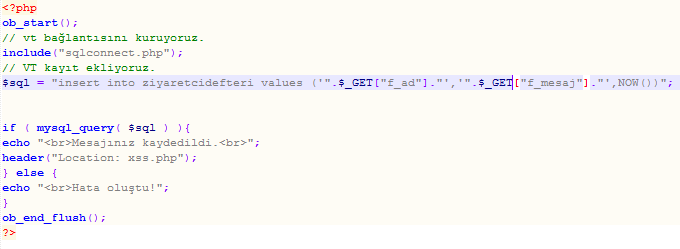
* 1. SQL Injection sömürü yöntemi (Parola kısmına da aynı ifade yazılacak)



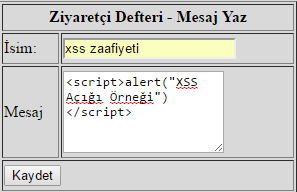
3.4 Giriş butonuna bastıktan sonra oluşan sql sorgusu ve sonuçları



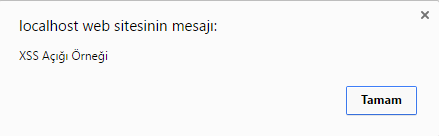
* 1. Stored XSS zafiyetinin test edilebileceği bir ziyaretçi defteri aracı



* 1. Textbox’ları kontrol edip yazdıran kod blokları



* 1. XSS Zaafiyetini sömürme yöntemi



* 1. XSS Sömürüsü sonrası ortaya çıkan sonuç



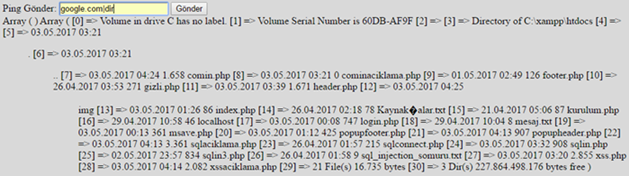
5.1 Komut Enjeksiyonu zafiyetini test edebileceğimiz bir ping gönderme aracı



5.2 google.com’a ping gönderme sonucu uygulamanın döndürdüğü sonuç



5.3 Ping gönderme işleminin yapıldığı kod bloğu



5.4 Komut Enjeksiyonu sömürü örneği ve döndürülen sonu

**Sonuçlar**

Projeyi geliştirirken zorlanmamızın sebebi daha önce konuyla alakalı eğitimimizin olmaması ve fazla Türkçe kaynak bulunmaması idi. Geliştirme sürecinde her aşamada yapacağımız şeylerle ilgili araştırmalar yapıp edindiğimiz bilgilerle sonuca ulaştık.

Geliştirme aşaması yaklaşık olarak 30 saat sürdü. Bunun dışında kaynakların yetersiz olması ve projeye fazla uygun olmaması idi.

Proje geliştirme aşamasında bize hml, css, php teknolojilerinde katkısı oldu.

Projenin tam anlamıyla amacına ulaştığını söyleyebiliriz. Bizden istenen şeylerin tamamını yerine getirdik.

Böylece projemizi tamamlamış olduk.

# Kaynakça

* <http://www.gurelahmet.com/tag/command-execution/>
* <http://www.recepkizilarslan.com/os-command-injection-ile-sistemlere-sizma/>
* <https://www.exploit-db.com/papers/13597/>
* <http://www.turkhackteam.org/bilgisayar-teknolojileri/1424360-os-command-injection-nedir-exploit-hacknology.html>
* <http://gavinmiller.io/2015/fixing-command-injection-vulnerabilities/>
* <http://www.includekarabuk.com/kategoriler/DVWAUygulamasi/Ders-4---Command-Injection-Low-Level.php>
* <http://www.beyaz.net/tr/ipucu/entry/850/>
* <http://www.gurelahmet.com/xsscross-site-scripting-nedir-ve-xss-lab-uygulamas%C4%B1-web-for-pentester/>
* <https://www.w3schools.com/>
* <https://www.formget.com/login-form-in-php/>
* <https://dev.mysql.com/doc/refman/5.7/en/resetting-permissions.html>
* <https://www.chip.com.tr/forum/dos-komutlari-detayliaciklamali_t63272.html>
* <http://www.mylmz.in/php>
* <http://www.turkhackteam.org/bilgisayar-guvenligi/819606-bilgisayar-cmd-komutlari.html>
* <http://php.net/manual/tr/function.system.php>
* <http://php.net/manual/tr/ref.exec.php>
* <http://www.linedesigner.net/blog/bir-dizindeki-dosyalarin-isimlerini-txt-dosyasina-yazdirmak.html>
* <http://bilalbaraz.com/phpde-exec-fonksiyonu-ve-kullanimi/>
* <https://security.stackexchange.com/questions/26047/command-injection-windows-platform>
* <https://www.owasp.org/>
* <https://www.olympos.net/howto-nasil/sql-injection/adim-adim-sql-injection-93419.html>