Yazılım Kalitesi Faktörü Olarak Yazılım Güvenliğinin Öğrenci Bilgi Sisteminde RIPS Testi ile Değerlendirilmesi

Araştırma Makalesi
Alınış Tarihi: 22 Temmuz 2019
Kabul Tarihi: 04 Ekim 2019

Sibkat KAÇTIOĞLU [*]
Mustafa KESKİNKILIÇ [**]
Ferhat KAHVECİ [***]

Öz: Öğrenci bilgi sistemleri (ÖBS) üniversitelerin eğitim süreçlerinin vazgeçilmezi olan birer yazılımdırlar. Her bilgi sisteminin olduğu gibi ÖBS'nin de yazılım kalite faktörlerinden biri olan yazılım güvenliğine yeterince sahip olması çok önemlidir. Çünkü ÖBS türü yazılımlar ciddi şekilde siber saldırılara maruz kalmaktadırlar. Bu çalışmada yazılım güvenliğini sağlamak için yazılım sistemlerindeki güvenlik açıklarını tarayan RIPS aracı ele alınmış ve kullanılmıştır. RIPS belirlenen yazılımın kaynak kodlarını statik olarak çalıştırmadan analiz edebilen ve bu özelliği ile testin tarafsızlığına katkı sunan bir araçtır. Statik kod analizörü olan RIPS, bu araştırma modelinde kullanılan ÖBS yazılımına uygunluğu ve bu konuda önceden yapılan çalışmalar dikkate alınarak seçilmiştir. Bu çalışmayla, üniversitelere kendi ÖBS yazılımlarını bu tür yazılım güvenliği testleri ile yapmaları konusunda yol göstermek amaçlanmıştır. Ayrıca burada yazılım güvenliği, bir yazılım kalitesi faktörü olarak ele alınmaktadır. Bu bağlamda açık kaynak kodlu bir ÖBS olan OpenSIS 7.1'in güvenliğini benzer birkaç popüler PHP tabanlı uygulamayla karşılaştıran deneysel bir çalışma sunulmaktadır. Bu çalışmanın literatürdeki doldurduğu boşluk: bir yazılımın diğer yazılımlar arasındaki kalite ve güvenlikle ilgili konumunun görünmesini sağlamasıdır. Yazılım güvenliği noktasında yazılım kalitesi sıralaması, testlerden elde edilen veriler üzerinden Katkı Oranı Değerlendirme (Additive Ratio Assessment) yöntemi ile yapılmıştır. Belirtilen yazılımlar üzerinde yapılan yazılım güvenliği testleri sonucunda; birinci Moodle 3.6.4, ikinci Joomla 3.9.5, üçüncü osCommerce 2.3.4, sonuncu ise OpenSIS 7.1. bulunmuştur.

Anahtar Kelimeler: Yönetim Bilişim Sistemleri, Yazılım Kalitesi, Yazılım Güvenliği, Yazılım Testi, Öğrenci Bilgi Sistemi, ARAS Yöntemi, RIPS, OpenSIS

Evaluation of Software Security as a Factor of Software Quality with RIPS Test in Student Information System

Abstract: *Student information systems (SIS) are software that is indispensable to the educational processes of universities. As with any information system, it is very important that the student information system (SIS) has enough software security which is one of the software quality factors. This is because software such as SIS is exposed to serious cyber attacks. In this study, the RIPS tool that scans security vulnerabilities in software systems to ensure software security is discussed and used. RIPS is a tool that can analyze the source code of the specified software without running it statically and contributes to the objectivity of the test with this feature. RIPS, a static code analyzer, has been selected considering the compatibility of the SIS software used in this*

[*] Prof. Dr., İstanbul Ticaret Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği Bölümü, ORCİD-ID: 0000-0002-8529-3775
[**] Dr. Öğr. Üyesi, Atatürk Üniversitesi, İktisadi ve İdari Bilimler Fakültesi, Yönetim Bilişim Sistemleri Bölümü, ORCİD-ID: 0000-0002-3394-5575
[***] Doktora Öğrencisi, Atatürk Üniversitesi, Yönetim Bilişim Sistemleri Anabilim Dalı, Yönetim Bilişim Sistemleri Bilim Dalı, ORCİD-ID: 0000-0002-7692-1266

research model and the previous studies on this subject. In this study, it is aimed to guide universities to conduct their own SIS software with such software security tests. In addition, software security is considered here as a software quality factor. In this context, an experimental study comparing the security of OpenSIS 7.1, an open source SIS, with a few popular PHP-based applications is presented. The gap that this study fills in the literature is that one software makes it possible to see the position of quality and safety among other softwares. The software quality ranking at the point of software security was done by using the Additive Ratio Assessment method based on the data obtained from the tests. As a result of software security tests performed on the specified software; first Moodle 3.6.4, second Joomla 3.9.5, third osCommerce 2.3.4, last one OpenSIS 7.1. It was found.

Keywords: *Management Information Systems, Software Quality, Software Security, Software Testing, Student Information System, ARAS Method, RIPS, OpenSIS*

I.Giriş

Yazılım geliştirme süreci etkinliklerinin yürütülmesindeki asıl hareket noktası müşteri memnuniyetini sağlayacak kalitede yazılım geliştirilmesidir. Bu da kaliteli yazılım geliştirmenin önemini artırmaktadır (Kurtel ve Eren, 2008: 1).

Yazılım testi yazılımın hem işlevselliği ve kullanılabilirliği açısından hem de yazılımın güvenliği bakımından ayrı ayrı yapılmalıdır. Yazılım kalitesinin nicel olarak değerlendirilebilmesi için yazılımın diğer testleri yanında güvenlik testinin de yapılması gereklidir (IEEE, 1998: 3). Bu nedenle yazılım testi yazılım kalitesinin çok önemli bir parçasıdır.

Bundan başka yazılımın güvenlik noktasında daha kaliteli olabilmesi için, güvenli yazılım geliştirme süreçlerine uygun biçimde ve siber güvenlik ilkelerine dikkat ederek yazılım geliştirmek gereklidir.

Bu çalışmanın kavramsal çerçeve bölümünde; önce ÖBS ve işlevleri ile yazılım kalitesi ve faktörleri açıklanmaktadır. Ardından yazılım güvenliği ve güvenli yazılım geliştirme konuları, sonra da yazılım testinden söz edilmektedir. Daha sonra test aracı RIPS anlatılıp, elde edilen sonuçların değerlendirilmesinde kullanılan ARAS yönteminden bahsedilmektedir. Son olarak literatürden taranan konuyla ilgili çalışmalar tanıtılmaktadır.

İkinci aşama olan yöntembilim bölümünde; çalışmanın amaç, kapsam ve yönteminden bahsedilmektedir. Böylece araştırmada hangi yazılımların test edileceği, test için hangi aracın kullanılacağı, test için hangi test kriterleri ve ölçütlerinin kullanılacağı, test aracıyla üretilen verilerin sınıflandırılmasında ve analiz edilmesinde hangi yöntemin kullanılacağı anlatılmıştır. Daha sonra araştırmanın modeli ve testin nasıl gerçekleştirileceği anlatılıp, yazılım testi süreci sonunda elde edilen bulgulara yer verilmektedir.

Sonuç bölümünde ise, RIPS 3.1 test aracı kullanılarak yapılan yazılım güvenliği taramasından elde edilen sonuçlarına göre, testi yapılan yazılımlarda kullanılan kod satır sayısı, XSS ve SQLi açıkları kriter olarak kullanılarak ARAS yöntemi ile güvenlik noktasında yazılımların kalite düzeyleri sıralanmaktadır.

II.Kavramsal Çerçeve

A. Öğrenci Bilgi Sistemi

ÖBS, öğrenci işleri ile ilgili verilerin girilip işlendiği ve sonuçların depolanıp ilgililere gönderildiği operasyonel düzeyde bir bilgi sistemidir. Önceden tanımlanan verilerin belli süreçler içerisinde ÖBS tarafından gerekli zamanlarda yetkili personel ile girilmesini sağlayan ve ayrıca öğrenci ile ilgili işlemlerde kurum adına resmi evrak formatında not dökümü, öğrenci belgesi gibi raporlar sunan bir tür raporlama yazılımıdır. Üniversitelerin farklı ihtiyaçlarına göre tasarlanmış farklı ÖBS'ler olduğundan bu tanımın kapsamı, üniversitede kurulu ÖBS'nin sağladığı hizmetlere göre ölçeklendirilebilir. Bu kapsam çerçevesinde ÖBS, Kumar (2011: 13)'ın tanımına göre, lise ve üniversitelerde operasyonel seviyede hizmet sağlayan bir operasyonel işlem sistemi (transaction processing system) olarak da tanımlanmaktadır.

ÖBS'ler ilk zamanlarda ana bilgisayar (mainframe) uygulamaları olarak kullanılıyordu. Daha sonra internetin, kişisel bilgisayarların ve dolayısıyla gelişmiş PC sunucuların yaygınlaşmaya başlamasıyla 1990'ların sonuna doğru Web ortamına hızlı bir şekilde adapte edilmişlerdir (Kumar, 2011: 13). Günümüzde gerek masaüstü bilgisayarlarda web tabanlı ara yüz ile gerekse akıllı cep telefonları üzerinden mobil uygulamalarla kullanıcılara erişim kolaylığı sağlanmasıyla üniversitelerde yaygın olarak kullanılmaktadırlar.

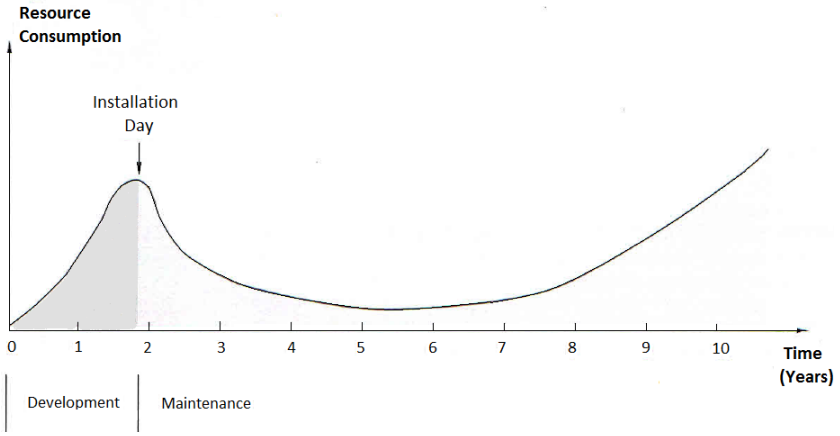
Öğrenci bilgi sistemine olan ihtiyaç öğrenci sayısıyla doğru orantılıdır. Öğrenci sayısı arttıkça akademisyen, öğrenci ve bunlara ait ders, şube, yoklama, sınav, not, vb. verilerin basit tablolar veya evrak üzerinde takibi ve raporlanması zorlaşmakta ve kullanıcıların hata yapma ihtimali artmaktadır. Ergin ve Akseki (2012: 365)'ye göre ÖBS, kullanıcıların gerekli bilgiye istedikleri zaman istedikleri yerden erişme imkânı sağlama, gerektiğinde bu bilgileri uygun formatlarda raporlama ve en önemlisi verilerin güvenliğini sağlama da dahil olmak üzere komple çözüm sunmalıdır. Bu nedenlerden dolayı üniversitelerde öğrenci işlerinin temel omurgasını oluşturması nedeniyle ÖBS'nin önemi büyüktür.

Bilgi sistemleri her ne kadar çeşitli donanım birimleriyle desteklenmiş olsalar da tek başlarına ele alındıklarında birer yazılımdırlar. Üniversitelerde öğrenci, akademisyen, personel sayısının çokluğu yapılan işlerin belli bir sistematiğe göre yapılmasını gerektirir. Bu nedenle ÖBS üniversitelerde vazgeçilmez bir şekilde ve yaygın bir biçimde kullanılan yazılımdır.

Genellikle ÖBS'nin kurulum işlemi tamamlandıktan sonra üniversite kendi bünyesi içinde ÖBS'yi geliştirmeye yine devam etmektedir. Bu geliştirme zaman içerisinde bazı güvenlik açıklarına sebep olmakta ve yazılım kalitesini olumsuz olarak etkileyebilmektedir. Çünkü yazılım güvenliği, yazılım kalitesinin bir faktörüdür ve kaliteyi doğrudan etkileyen unsurlardan biridir.

Şekil 1'de görüldüğü gibi yazılımların kurulum sonrası bakım-onarım ve yeniden geliştirme maliyetleri ilk proje maliyetlerinin üzerine çıkabilmektedir. Bunu engellemenin en önemli yolu proje dönemi içerisinde yazılım süreçlerinin belli kalite standartları ve modellerine uygun biçimde düzenlenmesi ve sisteminin yazılım kalite testlerinin yapılmasıdır. ÖBS'de yaşanabilecek aksaklıklar, eğitimin gecikmesine hatta durmasına sebebiyet verebilmektedir. Bu nedenle proje sonrasında, kullanım aşamasında da ÖBS üzerinde yazılım kalite ölçüm çalışmaları düzenli olarak yapılmalıdır.

Bu çalışmada bir ÖBS yazılımına güvenlik açığı testinin nasıl yapılabileceği yaygın kullanılan bir ÖBS yazılımı olan ve belirli kriterlere göre seçilen OpenSIS 7.1 üzerinde gösterilmektedir.



Şekil 1. Bakım Maliyet Grafiği (Flatten vd., 1992).

Üzerinde yazılım güvenliği testi yapılan ÖBS seçiminde kullanılan bazı kriterler şunlardır. Öncelikle RIPS 3.1 yazılımı ile test edilebilmesi için PHP koduyla yazılmış olması gerekmektedir. Testin asıl amacı; üniversitelerin kendi ÖBS yazılımı üzerinde nasıl yazılım güvenliği testi gerçekleştirebilecekleri konusunda yol gösterici bir çalışma olmasıdır. Bu nedenle çalışmanın maliyetini minimize etmek açısından açık kaynak kodlu olması gerekir. Ayrıca yazılımın tüm kaynak kodlarına erişilebilmelidir. Bir başka önemli unsur ise kullanılacak ÖBS'nin yaygın olarak kullanılmasıdır. Burger (2015)'in araştırmasına göre en çok kullanılan açık kaynak kodlu yazılımlar Tablo 3'de test kriterlerine göre belirtilmektedir. Tablo 1'de görüldüğü üzere RIPS 3.1 yazılımı ile testi gerçekleştirilebilecek en uygun ÖBS, kodlama dili PHP olması ve kaynak kodlara erişim kriterlerine göre OpenSIS'dir.

Tablo 1. *Test için uygun ÖBS yazılımı seçimi için kriter tablosu*

Yazılım	Open Admin	Schooltool	Ascend	Fedena	TS School
Kodlama dili	Perl	Python	Asp	Ruby	Exe
Kaynak kodlara erişim	var	var	yok	var	yok

OpenSIS yazılımı sayesinde öğrenci bilgileri, yoklamaları, planlama, sağlık durumu ve değişik türde raporlar alabilme gibi temel ÖBS görevlerini yerine getirmektedir. Aktif dizin entegrasyonu, Moodle entegrasyonu ve online öğrenci kaydı gibi bazı özellikler için ücretli versiyonu da bulunmaktadır. 100.000 öğrenciye kadar kapasitesi bulunmaktadır. 16.000'e yakın eğitim kurumunda kullanılmaktadır (OpenSIS, 2016). Bu çalışmada test için en son sürüm olan OpenSIS 7.1 kullanılmıştır.

B. Yazılım Kalitesi

Yazılım kalitesi, yazılım üretim süreci boyunca ara ürünlerin ve tamamının standartlara uygunluğunu denetleme ve kalitesinin geliştirilmesidir. Kalite sağlama çalışmalarının genel amacı yazılım maliyetini düşürme, üretimini ve yönetimini kolaylaştırma, belgeleme ve genel sorunların giderilmesi gibi genel düzeni sağlamaya yönelik faaliyetleri içerir (Arifoğlu ve Doğru, 2001: 10).

Yazılım kalitesi; yazılım mühendisliği altında tanımlanmış, yazılım isterler analizi, yazılım tasarımı, yazılım gerçekleştirimi, yazılım testi, sistem entegrasyonu, yazılım bakımı, düzenleşim yönetimi, yazılım proje yönetimi gibi teknik çalışmalar ve yazılım geliştirme yöntembilimleri ile birlikte ele alınıp uygulanırsa başarıya ulaşılacak bir konudur. Bu yüzden yazılım kalitesi konusu tıpkı yukarıda sayılan konular gibi yazılım mühendisliği uygulamalarının vazgeçilmezi olarak tanımlanmış ve yazılım mühendisliği başlığı altına yerleştirilmiştir. Yazılım kalitesi konusunda tecrübe ve pratiğe dayalı birçok önlem ve öneri ileri sürülmekle beraber; aşağıda sıralandığı gibi birçok yöntem, teknik ve uygulama da sınıflandırılarak literatüre kazandırılmıştır (Sarıdoğan, 2004; Atbaş, 2012).

Yazılım kalite etmenlerinden bazıları şunlardır: Kalite metrikleri, kalite güvence ve yöntemleri, süreç modelleri, yazılım süreç iyileştirme modelleri, kalite sistemi standartları ve uygulamaları, gözden geçirme uygulamaları, yazılım güvenilirliği, güvenli yazılım geliştirme, sistem aktarımı ve entegrasyon, yazılım kalite ölçümü (Sarıdoğan, 2004).

C. Yazılım Güvenliği

Bilgisayara bir iş yaptırmak istenildiğinde mutlaka bir yazılım geliştirilmelidir. Geliştirilen yazılım çok küçük ölçekte olabileceği gibi birçok bilgisayar üzerinde çalışan büyük ölçekte yazılımlar grubu da olabilmektedir. Yazılımın karmaşıklığı arttıkça geliştirmesi de zorlaşmakta ve bu noktada güvenli yazılım geliştirmenin önemi daha iyi anlaşılmaktadır (Sarıdoğan, 2008: 114).

Donanım ve yazılım ortamlarındaki gelişmeler ile birlikte birçok kurum kendi iş süreçlerinin yönetilmesinde çeşitli yazılımlar kullanmaktadırlar. Bu yazılımların sağladığı avantajlar hiç şüphesiz tercih edilme sebeplerindendir. Ancak kullanılan yazılımlardaki güvenlik sorunları kurum için de bir tehdit olmaktadır. Bu yüzden kurumun iş süreçlerinin sürekliliğinin sağlanmasında yazılım güvenliği büyük önem arz etmektedir (Şahinaslan vd., 2011: 599).

Web uygulamalarının daha güvenli bir hale getirmek için kâr amacı gütmeden çalışan bir kuruluş olan Açık Kaynak Kodlu Web Uygulamaları Güvenliği Projesi (The Open Web Application Security Project-OWASP)'nin en son 2017'de yayınladığı raporda en yüksek ilk 10 tehdit sıralaması Tablo 2'de verilmiştir. Bu çalışmada yazılım kalite faktörlerinden olan yazılım güvenliği boyutu için, Tablo 2'de yer alan Araya Kod Girme (SQLi) ve Çapraz Siteden Betik Çalıştırma (XSS) testleri yapılacaktır.

Tablo 2. *OWASP-2017 Raporuna Göre Yazılımlarda En Yüksek İlk 10 Tehdit.*

Kod	**Risk**
A1	Araya kod girme-Injection
A2	Kırık Kimlik Doğrulama-Broken Authentication
A3	Hassas Veri Teşhiri-Sensitive Data Exposure
A4	Dış XML nesneleri-XML External Entities (XXE)
A5	Kırık Erişim Kontrolü-Broken Access Control
A6	Yanlış Güvenlik Yapılandırılması-Security Misconfiguration
A7	Çapraz Siteden Betik Çalıştırma-Cross-Site Scripting (XSS)
A8	Güvensiz Seri Ayrışımı-Insecure Deserializtion
A9	Açıkları Bilinen Bileşenleri Kullanma-Using Components with Known Vulnerabilities
A10	Yetersiz Günlük Kaydı ve İzleme-Insufficient Logging & Monitoring

1. Araya kod girme (SQLi): Birçok web uygulaması, kullanıcı bilgilerini ve tercihlerini bir veri tabanında tutmaktadır. Bu veri tabanı ile haberleşme SQL sorguları ile gerçekleştirilir (Jovanovic vd., 2010: 864). Saldırganlar, SQL sorgusunun "WHERE" komutundan sonra gelen "AND", "OR" ve "NOT" gibi mantıksal komutlarının sağladığı veri süzme zincirini kırabilecek bir değişken kullanmaktadırlar.

```
$id = $_GET['ogr_no'];
$query= "SELECT * FROM kullanicilar WHERE ogr_no = $id";
```

Yukarıdaki sorgu örneğinde görüldüğü üzere formdan gelen ogr_no bilgisi hiçbir filtre ve kontrolden geçirilmeden sorguya dâhil edilmektedir. Saldırganlar bu ogr_no değişkeni olarak sorguyu bütün öğrenci numaralarını görüntüleyecek değişken kullanabilirler. ogr_no değişkeni "105 or 1=1" şeklinde kod girildiğinde "WHERE 1=1" durumu her zaman sağlanacağından saldırgan aşağıdaki gibi SQL dizilimini elde ederek tüm kullanıcı bilgilerini ele geçirir.

```
"SELECT * FROM kullanicilar WHERE ogr_no = 105 or 1=1";
```

2. Çapraz Siteden Betik Çalıştırma (XSS): XSS saldırılarının amacı, kullanıcı bilgilerini ele geçirmektir. Kimlik doğrulama bilgisi çerezi (cookie) içeren bir web talebi, kullanıcı sistemden çıkış yapmadığı sürece sunucu tarafından ilgili kullanıcının talebi olarak değerlendirilir. Bu nedenle, bu çerezi bir şekilde elde eden kişi mevcut oturumun sahibi olur (Jovanovic vd., 2010: 863). Saldırganlar çoğunlukla Javascript betikleri yardımıyla bu cookie dosyasını ele geçirmektedirler. Aşağıdaki örnekte saldırganlar link olarak Javascript betiği ekleyebilirler ve çerez dosyalarını kendi bilgisayarına aktarırlar.

```
$sayfa = $ _GET ['sayfa'];
echo "<a tıklayın </a> href='$sayfa'>";
```

D. Yazılım Testi

Yazılım testi, kalite hedeflerini yakalama, yazılım değişikliğinde kalitenin izlenmesi, karmaşıklığın kontrolü, hata analizi, yazılımın anlaşılabilirliği, yazılım ile ilgili tahmin yapabilme ve yazılım geliştirme sürecinin kontrolü konularında yardımcı olmaktadır (IEEE, 1998: 3; Ebert vd., 2005: 2; Zuse, 1998: 23). Yazılım testi, yazılım geliştirme yaşam döngüsü içerisinde kalite, güvenlik, doğruluk ve tamlık gibi iyi bir yazılımın olmazsa olmaz özelliklerini doğrudan etkileyen önemli süreçlerden biridir (Gürbüz, 2010: 36).

Yazılım testlerinin en çok bilineni kutu yaklaşımıdır. Beyaz Kutu ve Kara Kutu olmak üzere iki kutu yaklaşımı vardır (Jovanovic. 2009: 31). Kara kutu yaklaşımı yazılımı kapalı bir kutu olarak ele alır. Yazılımın istenen girdilerle istenen çıktıları sağlayıp sağlamadığı kontrol edilmektedir. Beyaz kutu yaklaşımında ise yazılımın algoritmalarının ve değişkenlerinin uygun olup olmadığına bakılır.

3'de Camoğlu (2010) tarafından gerçekleştirilen yazılım test çeşitleri görülmektedir. Yazılım testlerinin kodlar çalıştırılarak ve kodlar çalıştırılmadan olmak üzere iki ana başlık altında sınıflandırıldığı görülmektedir. Kodlar çalıştırılmadan statik test olarak gerçekleştirilen yazılım testinde ise yazılım ve kod inceleme uygulamaları bulunmaktadır. Bu bilgilere göre, yazılım açıklarındaki statik kod analizi uygulamaları yazılım kalitesi ölçümlerinden biridir.

Tablo 3. *Yazılım Test Çeşitleri (Camoğlu, 2010)*.

Kodlar çalıştırılarak (dinamik olarak)		**Kodlar çalıştırılmadan (statik test)**
Yazılım geliştirme sürecinin her aşamasının sonunda yazılım geliştirme düzeyine göre	*Performans, güvenlik, kullanım kolaylığı gibi işlevsel olmayan testler*	
• Birim Testi-Unit Test • Tümleştirme Testi-Integration Test • Sistem Testi-System Test • Sistem Tümleştirme Testi-System Integration Test • Regresyon Testi-Regression Test • Kabul Testi-Acceptance Test • Alfa Test-Alpha Test • Beta Test	• Başarım Testi-Performance Test • Yük Testi-Load Test • Kararlılık Testi-Stability Test • Kullanışlılık Testi-Usability Test • Güvenlik Testi-Security Test • Uluslararası ve Yerel Kullanıma Uygunluk Testi-Internationalization and Localization Test	• Yazılım İnceleme-Software Inspection • Kod İnceleme-Code review

E. Test Aracı RIPS

RIPS 3.1 yazılımı, PHP ile yazılan bir statik kod analizi aracıdır. Yazılımın tamamını analiz ederek bir program modeline dönüştürmektedir ve bu şekilde hassas güvenlik açıklarını tespit etmektedir. Sonuçları grafikler ve istatistikî bilgiler olarak görüntülemektedir. PHP ile kodlanmış dosyaların kaynak kodlarını çalıştırmadan birer statik olarak kontrol etmektedir. Bu nedenle PHP ile kodlanan modüler bir yapı olarak tasarlanmış olan ÖBS ile içerik yönetim sistemi için kullanılan Moodle, osCommerce ve Joomla gibi yazılımları XSS ve SQLi gibi açıkları yönüyle taramada başarıyla kullanılabilmektedir (Ripstech, 2019).

F. Test Edilen Yazılımlar

Joomla 3.9.5, osCommerce 2.3.4 ve Moodle 3.6.4 ÖBS yazılımı ile aynı kodlama dili olan PHP kullanılarak oluşturulmuş çok yaygın kullanılan uygulamalardır.

Joomla, web sitesi kurmaya yönelik bir içerik yönetim sistemidir. Açık kaynak kodlu olması ve sürekli güncellenmesi en büyük avantajlarındandır. Web site yönetimi için fazla teknik bilgi ve deneyim gerektirmemektedir. Şirketler, okullar, e-ticaret siteleri, kişisel siteler, vb. birçok sitenin Joomla ile yönetildiği görülmektedir (Joomla, 2019). Bu çalışmada test için en son sürüm olan Joomla 3.9.5 kullanılmıştır.

osCommerce, e-ticaret yazılımıdır ve açık kaynak kodludur. Ürünleri kategorilere ayırma, ürün listesi hazırlama, fatura çıktısı alma, döviz kuru dönüşümü ve çeşitli istatistikî bilgilerin raporlanması gibi birçok e-ticaret içeriğini kolayca yönetilmesini sağlar (osCommerce, 2019). Bu çalışmada test için en son sürüm olan osCommerce 2.3.4 kullanılmıştır.

Moodle, açık kaynak kodlu uzatan eğitim sistemi ve ders içeriği yönetim sistemidir. Modular Object Oriented Dynamic Learning Environment kısaltması olan Moodle (Esnek Nesne Yönelimli Dinamik Öğrenme Ortamı), ÖBS yazılımına entegre olarak üniversitelerin çoğunda kullanılmaktadır (Moodle, 2019). Bu çalışmada test için en son sürüm olan Moodle 3.6.4 kullanılmıştır.

G. ARAS Yöntemi

Zavadskas ve Zenonas (2010: 159) tarafından çok kriterli karar vermede yeni bir yaklaşım olarak ortaya konan ve çok kriterli sonuçları sıralamada kullanılan bir yöntemdir. Katkı Oranı Değerlendirme (Additive Ratio Assessment-ARAS) yönteminde her kriter optimal kriterlerle karşılaştırılmaktadır. Örneğin optimal kriter değerinin 100 olduğu bir sıralamada alternatif kriter değerlerinin tümünün bu değerin altında olduğu ve en büyük değerin 70 olduğu durumda en iyi değer 70 olur ve diğer kriterler bu değer üzerinden sıralanır. Bu durumda en iyi kriter değerinin yüzdelik değeri 100 olur. Bulgular bölümünde 6 adımda karar matrisinin oluşturulması, normalize katsayısının oluşturulması, normalize karar matrisinin oluşturulması, ağırlık katsayısının hesaplanması, ağırlıklı normalize karar matrisi, optimallik fonksiyon değerlerinin hesaplanması olmak üzere RIPS 3.1 test aracından elde edilen sonuçlar ARAS yöntemi ile sıralanmaktadır.

1. Karar Matrisinin Oluşturulması: Optimal değerlerin hesaplanmasında eğer kriter değerin büyük olması daha iyi ise kritere ait en büyük değer (1) ve eğer kriter değerin düşük olması daha iyi ise kritere ait en küçük değer (2) alınarak bir optimal değer satırı oluşturulur. Böylece Kriterlere ait değerler matrisi oluşturulur.

$$x_{0j} = \max_i x_{ij} \qquad (1)$$

$$x_{0j} = \min_i x_{ij} \qquad (2)$$

2. Normalize Katsayısının Oluşturulması: Farklı veri toplama araçlarından elde edilen veriler farklılık gösterdiğinden dolayı değerler normalize edilir. Normalize işlemine başlamadan önce her sütun için normalize katsayısı hesaplanır. Normalize katsayısı hesaplanırken eğer kriter değerin büyük olması daha iyi ise kritere ait değerlerin her biri bölme işlemine göre tersi alınarak (3) toplamları (4) alınır. Eğer değerin düşük olması daha iyi ise kritere ait değerler toplanarak (5) normalize katsayıları oluşturulur.

$$x_{ij}^{\bullet} = \frac{1}{x_{ij}} \qquad (3)$$

$$\bar{x}_k = \sum_{i=0}^{m} x_{ij}^{\bullet} \quad (4)$$

$$\bar{x}_k = \sum_{i=0}^{m} x_{ij} \quad (5)$$

3. Normalize Karar Matrisinin Oluşturulması: Normalize karar matrisi oluşturulurken eğer kriter değerinin büyük olması daha iyi ise kritere ait değerlerin her biri normalize katsayısına bölünür (6), eğer değerin düşük olması daha iyi ise kritere ait değerlerin her biri normalize katsayısı ile çarpılır ve tersi alınarak (7) normalize karar matrisi oluşturulur.

$$\bar{x}_{ij} = \frac{x_{ij}}{\bar{x}_k} \quad (6)$$

$$\bar{x}_{ij} = \frac{1}{x_{ij}.\bar{x}_k} \quad (7)$$

4. Ağırlık Katsayısının Hesaplanması: Kriterlerin ağırlıklarını belirlemek için $0<w_j<1$ arasında ve ağırlık katsayılarının toplamı bir (8) olacak şekilde her bir kriter için ağırlık katsayısı seçilir. Yazılımların kalite sıralaması ağırlıklı olarak XSS ve SQLi açıklarına göre yapıldı.

$$\sum_{j=1}^{n} w_j = 1 \quad (8)$$

5. Ağırlıklı Normalize Karar Matrisi: Normalize karar matrisindeki tüm değerler kendi kriterinin ağırlık katsayısı ile çarpılarak (9) ağırlıklı normalize karar matrisi oluşturulur.

$$\hat{x}_{ij} = \bar{x}_{ij}.w_{ij} \quad (9)$$

6. Optimallik Fonksiyon Değerlerinin Hesaplanması: Ağırlıklı normalize karar matrisinin her bir kriterinin değerleri toplanır (10) ve her kriterin optimal fonksiyon değeri Si hesaplanır. Matrisin ilk satırı optimal değer satırı olduğundan ilk satırdan elde edilen değer So olarak adlandırılır. Elde edilen Si, So ile oranlanır ve 0 ila 1 arasında Ki değerleri (11) elde edilir. Ki değeri 100 ile çarpıldığında yüzdelik bir değer elde edilir.

$$S_i = \sum_{j=1}^{n} \hat{x}_{ij}, \qquad i = 0,1,\dots,m \quad (10)$$

$$K_i = \frac{S_i}{S_0} \qquad i = 0,1,\dots,m \quad (11)$$

H. İlgili Çalışmalar

Dahse ve Holz (2014: 1) çalışmasında NewsPro 1.1.4, NewsPro 1.1.5, myBloggie 2.1.3b ve myBloggie 2.1.4 gibi PHP'nin en çok kullanılan yazılımları RIPS 0.55 ile test etmiştir. Elde ettikleri sonuçları, farklı test araçları kullanarak aynı yazılımları test eden Jovanovic, Kruegel ve Kirda (2010: 330), Xie ve Aiken (2006: 1)'in yaptığı testlerle karşılaştırmaktadır. Aynı şekilde bu çalışmada RIPS yazılımı ile en çok bilinen PHP tabanlı yazılımlar kullanılarak test işlemi yapılmıştır.

Yıldırım (2015: 285) ARAS yönteminin literatürde kullanım alanlarını ele almaktadır. Özellikle Stanujkic ve Jovanovic (2012: 545)'in fakülte web sayfasını belli kriterlerle ARAS yöntemi sıralamasını ayrıntılı bir şekilde açıklamakta ve RIPS 0.55'den elde ettiği sonuçları ARAS ile sıralamaktadır.

Jovanovic vd. (2010: 861)'nin geliştirdikleri Java ile yazılan PHP statik kod analizi yapan açık kaynak kodlu Pixy yazılımıyla, çapraz siteden betik çalıştırma ve araya SQL kodu girilebilen açıkları bulabilmektedir. Pixy'in biraz daha gelişmiş versiyonu olan Saner, manuel olarak oluşturulan ve ön tanımlı test durumlarını kullanmaktadır (Balzarotti vd., 2008: 387).

Bussieck vd. (2004: 267) yazılım testi için GAMS (The General Algebraic Modeling System) aracını kullanmaktadırlar. GAMS matematiksel programlama ve optimizasyon için tasarlanan yüksek seviyeli bir dildir. Çalışmalarında, GAMS ile nasıl bir yazılım testi yapılacağı anlatılmaktadır. Ancak makalelerinde uygulamalı olarak yapılan bir çalışma yoktur.

Hills vd. (2013: 325) PHP kodlarının dinamik yapısını inceleyen bir çalışmayı ele almışlardır. Çalışmalarında Joomla, Moodle, phpBB, phpMyAdmin ve benzeri PHP tabanlı yazılımların değişken yapılarını, statik ve dinamik fonksiyonlarını analiz etmişlerdir.

Huang vd. (2004: 40) çalışmalarında web uygulamalarının güvenliğini test eden WebSSARI aracını kullanmışlardır. Bu araç başta araya kod girme ve çapraz siteden betik çalıştırma olmak üzere çeşitli güvenlik açıklarını tespit edebilmektedir. 230 adet açık kaynak kodlu Web uygulamasının testini gerçekleşirmişlerdir. Safana ve Ibrahim (2010: 447) çalışmalarında etkin bir yazılım testi için yazılım test yönetim aracı kullanmanın önemi vurgulamakta ve SpriraTeam yazılım test aracı ile örnek bir uygulama gerçekleştirmişlerdir. Kirda vd. (2006: 330) makalesinde XSS saldırılarını önleyen sunucu taraflı Noxes adında aracı uygulamalı olarak tanıtmıştır.

Bu çalışmalarda görüldüğü gibi, birçok araştırmacı teknolojideki yeniliklere ayak uydurmak için Bilgi Sistemleri Başarı ve Memnuniyet modelleri geliştirmektedirler. Yukarıda belirtildiği gibi, bazıları yazılım kalitesini ölçmek için kendi araçlarını geliştiriyorlar; Noxes, WebSSARI, GAMS, Pixy, vb. Ancak, bu çalışmada olduğu gibi herhangi bir yazılım kalite modelinin veya genel yazılım kalitesi öğesi olarak kullanılabilecek bir değişkenden (örneğin yazılım güvenliği gibi) bahseden bir çalışmayla sıkça karşılaşmak zordur. Bu çalışmanın

literatürdeki doldurduğu boşluk; sahip olunan yazılımın diğer yazılımlar arasındaki konumunun görülmesini sağlamaktır.

III.Yöntembilim

A. Araştırmanın Amacı

Bu çalışmada ÖBS ve yazılım test süreçleri hakkında genel bilgiler vererek ve literatürden benzer çalışmaları da örnekleyerek, yazılım güvenlik açıklarının belirlenmesinde statik kod analizi çalışmalarının yazılım kalite belirleme uygulamalarından biri olduğunu göstermek amaçlanmaktadır. Ayrıca yazılım güvenliği bağlamında yazılım kalitesini belirleme süreci çalışmalarından olan XSS ve SQLi açıklarının tespitinin uygulamalı olarak incelenmesi amaçlamaktadır.

RIPS aracı, XSS ve SQLi açıkları ve ARAS yöntemi çalışmanın temelini oluşturmaktadır. Bu temellerin belirli bir test yöntemi olarak seçilmesinin nedeni, üzerinde test yapılacak olan ÖBS yazılımının diğer yazılımlar arasındaki kalite ve güvenlik bağlamında konumunun görülmesinin sağlanmasıdır. Bir vaka çalışması olarak, küçük bir çerçevede, sistemlerin seçimi ve testteki ağırlık katsayıları değişkendir ve varsayımdır. Yazılım sürümleri, deney yapıldığı sırada mevcut olan sürümlerdir. Ağırlık katsayıları 20/40/40 olarak verilmiştir. Seçilen sistemler popüler PHP tabanlı sistemlerdir. Aynı zamanda başka popüler PHP sistemleri de vardır (Örneğin: Drupal, WordPress, MediaWiki). Bu çalışmada asıl amaç, üniversitelere kendi SIS yazılımlarında bu tür bir testi gerçekleştirmek için pratik bir rehber oluşturmaktır. Bu nedenle, bu sistemler, versiyonlar ve ağırlıklandırma katsayıları, örnek olarak seçilmiştir.

B. Araştırmanın Kapsamı ve Yöntemi

Bu araştırma üniversitelerde sıklıkla kullanılan açık kaynak kodlu ve PHP tabanlı yazılımlar olan OpenSIS 7.1, osCommerce 2.3.4, Joomla 3.9.5 ve Moodle 3.6.4'ü kapsamaktadır. Bunların yazılım kalite faktörlerinden olan yazılım güvenliği boyutunun analiz edilmesi için bu yazılımlar üzerinde RIPS 3.1 yazılım test aracı kullanılarak; OWASP'ın 2017'de yayınladığı raporda en yüksek ilk 10 tehdit sıralaması içinde yer alan, Araya Kod Girme (SQLi) ve Çapraz Siteden Betik Çalıştırma (XSS) testleri yapılacaktır. Bu yazılımların bu iki tehdit konusundaki açıkları sayısal olarak ortaya konup, elde edilen sonuçlar ARAS yöntemiyle değerlendirilecektir. Sonra da bu yazılımlar arasında yazılım güvenliği noktasında bir yazılım kalite sıralaması yapılacaktır.

C. Araştırma Modeli

"İlgili Çalışmalar" bölümünde, RIPS 3.1 ile test edilen önde gelen PHP yazılımlarından bazıları belirtilmiştir. Burger'in (2015) araştırmasında en popüler açık kaynaklı yazılım olarak OpenSIS 7.1 seçilmiştir. Bu seçim, programlama dilinin PHP olması ve kaynak kodun erişilebilir olması kriterlerine göre yapılmıştır. RIPS 3.1 sonuçlarının her bir bölümü detaylı olarak incelenerek her bölümün sonuçları, görüntüleri ve şekilleri ile birlikte kaydedilmiştir. Özellikle araştırmaya konu olan XSS ve SQLi açıklarından elde edilen sayısal

veriler tablo halinde verilmiştir. Yazılımların satır sayıları yine RIPS 3.1 test aracı tarafından belirlenmektedir.

Kullanılan yazılım kodları RIPS test aracının bulut hizmet sağlayıcısı servisi olan "https://saas-3.ripstech.com" adresine yüklenerek test gerçekleştirilmiştir. RIPS teknoloji şirketi test yazılımını araştırmacılara 15 günlük bir deneme hesabı üzerinden ücretsiz olarak kullanmalarına izin vermektedir. RIPS 3.1 statik test programı ile OpenSIS 7.1 yazılımı taranarak; OWASP'ın 2017 listesinde yer alan XSS ve SQLi açıkları konusunda elde edilen test sonuçları nicel olarak karşılaştırılmıştır. PHP ile kodlanmış en çok bilinen yazılımlar olan Joomla 3.9.5, osCommerce 2.3.4 ve Moodle 3.6.4 da aynı şekilde test ile elde edilen veriler kullanılarak ARAS yöntemiyle yazılım kalite sıralaması yapılmaktadır.

D. Bulgular

Tablo 4'de görüldüğü üzere her yazılım için dosya sayısı, kod satırı, XSS sayısı ve SQLi açıkları tarama sonuçları olarak verilmiştir. Testin uygulandığı OpenSIS 7.1 yazılımı 4363 adet dosya ve 334773 kod satırından oluşmaktadır.

Tablo 1. *Test sonuçlarına göre yazılımların XSS ve SQLi açıkları.*

Yazılım	Dosya	Satır Sayısı	XSS	SQLi
OpenSIS 7.1	4363	334773	10	10
Joomla 3.9.5	7057	568326	3	1
osCommerce	702	86723	7	9
Moodle 3.6.4	5039	3349944	20	0

ARAS yöntemi ile hesaplamalara başlamadan önce, hangi kriterlerin rakam olarak fazla olmasının veya az olmasının iyi olduğunun belirlenmesi gerekmektedir. Testte kullanılan satır sayısı kriteri yazılım projesinin büyüklüğünü göstermektedir. Bu nedenle bu kriterin rakam olarak büyük olması iyidir. XSS ve SQLi kriterleri yazılım açıklarındaki sayıyı gösterdiğinden az olması iyidir.

Hesaplamada dosya sayısı yerine satır sayısının baz alınmasının sebebi, bazen bir dosya içerisinde 10 satırlık bir kod olabildiği gibi 1000 satırı aşkın kod da bulunmaktadır. Bu yüzden daha sağlıklı bir değerlendirme için satır sayıları dikkate alınmıştır.

1. Karar Matrisinin Oluşturulması: Tablo 4'deki test sonuçları (1) ve (2) numaralı denklemler kullanılarak aşağıdaki gibi bir karar matrisi oluşturmaktadır (12). Bu matrisin ilk satırını kriterlere ait optimal değerler oluşturur.

$$X = \begin{bmatrix} 4339766 & 0 & 0 \\ 334773 & 10 & 10 \\ 568326 & 3 & 1 \\ 86723 & 7 & 9 \\ 3349944 & 20 & 0 \end{bmatrix} \qquad (12)$$

2. Normalize Katsayısının Oluşturulması: Tablo 5'te normalize katsayısı, satır sayısı için eşitlik (3) ve (4) kullanılarak, XSS ve SQLi için eşitlik (5) kullanılarak hesaplanmıştır.

Tablo 2. *Kriterlerin her biri için hesaplanan normalize katsayıları*

	Satır Sayısı	**XSS**	**SQLi**
Normalize katsayıları	7689710	0.95	20000001

3. Normalize Karar Matrisinin Oluşturulması: Normalize karar matrisi (13) oluşturulurken, satır sayısı için eşitlik (6) kullanılarak, XSS ve SQLi için eşitlik (7) kullanılarak hesaplanmıştır.

$$\bar{X} = \begin{bmatrix} 0.435 & 0.347 & 0.5 \\ 0.043 & 0.104 & 0 \\ 0.073 & 0.347 & 0 \\ 0.011 & 0.148 & 0 \\ 0.435 & 0.052 & 0.5 \end{bmatrix} \quad (13)$$

4. Ağırlık Katsayısının Hesaplanması: Tablo 6'de sırasıyla 0.20, 0.40 ve 0.40 ağırlık katsayıları eşitlik (8) kuralına göre toplamları bir olacak şekilde verilmiştir.

Tablo 3. *Kriterlerin her biri için hesaplanan ağırlıklı normalize katsayıları*

	Satır Sayısı	**XSS**	**SQLi**
Ağırlıklı Normalize katsayıları	0.20	0.40	0.40

5. Ağırlıklı Normalize Karar Matrisi: (9) numaralı denkleme göre ağırlıklı normalize karar matrisi (14) oluşturulmuştur.

$$\hat{X} = \begin{bmatrix} 0.087 & 0.139 & 0.20 \\ 0.008 & 0.041 & 0 \\ 0.014 & 0.139 & 0 \\ 0.002 & 0.059 & 0 \\ 0.087 & 0.020 & 0.20 \end{bmatrix} \quad (14)$$

6. Optimallik Fonksiyon Değerlerinin Hesaplanması: (10) ve (11) numaralı denklemler kullanılarak elde edilen bu yüzdelik değer üzerinden yazılımlar sıralanır (Tablo 7).

Tablo 4. *Optimallik fonksiyon değerleri ve alternatif sıralamaları*

	Si	**%Ki**	**Sıra**
Optimal	0.426		
OpenSIS 7.1	0.050	11.83	4
Joomla 3.9.5	0.153	36.08	2
osCommerce 2.3.4	0.061	14.51	3
Moodle 3.6.4	0.307	72.28	1

Tablo 7'de, Moodle 3.6.4, yazılım kalitesinde ilk sırada yer alırken, OpenSIS 7.1, sonuncusudur. Joomla 3.9.5, osCommerce 2.3.4'den ikinci daha güvenli bir yazılımdır.

IV.Sonuç

Bu çalışmada ÖBS kavramını, yazılım kalitesinin önemini, yazılım güvenliği ile ilgili kavramları ve yazılım kalitesini güvenlik bağlamında ölçen yazılım testi kullanımı gösterilmektedir. Yazılım güvenliği, yazılım kalitesinin önemli bir parçasıdır. Yazılım güvenliğinin bir parçası olan statik kod analizi aynı zamanda yazılım kalitesinin ölçüm uygulamalarındandır. Kalite ölçüm süreci açısından bu çalışmada statik kod analizi yapılmıştır. Böylece bu çalışmanın temel amacı olan, ÖBS yazılımının belirli saldırılara karşı kırılganlığını ölçmek için nicel bir yaklaşım kullanılmış ve yazılımları genel kalitesi de ortaya konmuştur.

Statik kod analizörü araştırma modelinde kullanılan ÖBS yazılımına uygunluğuna ve önceki çalışmalarda kullanılmasına göre seçilmiştir. RIPS seçilen tüm yazılımların kaynak kodlarını ayırım yapmadan statik olarak çalıştırmaksızın analiz edebildiğinden, araştırmada önemli ölçüde yazılım testinin tarafsızlığına katkıda bulunmaktadır. Bu araştırmada RIPS'in, yazılım temelli sitemlere yazılım testini, özellikle iki yaygın saldırı türü olan, SQLi ve XSS konusunda kolaylıkla yapılabileceği gösterilmiştir. Tablo 4'te her yazılım için dosya ve kod satırı sayısı ile XSS ve SQLi açıkları tarama sonuçları olarak verilmiştir. OpenSIS 7.1 yazılımında tespit edilen XSS ve SQLi açıkları, aynı programlama dili (PHP) ile kodlanmış öncü yazılımlar olan Joomla 3.9.5, osCommerce 2.3.4, Moodle 3.6.4 karşılaştırılmıştır.

Bu çalışmada, Dahse ve Holz (2014: 1), Jovanovic vd. (2010: 861), Xie ve Aiken (2006: 1)'nin araştırmalarına ek olarak, ARAS yöntemini kullanarak yazılım kalitesine göre sıralama gerçekleştirilmiştir. Bu çalışmanın varsayımına göre, yazılımın satır sayısı kriteri için yüksek değere sahip olması ve XSS ve SQLi güvenlik açıkları için düşük değere sahip olması daha iyi olarak kabul edilir. Tablo 7'de, Moodle 3.6.4, yazılım kalitesinde ilk sırada yer alırken, OpenSIS 7.1, sonuncusudur. Joomla 3.9.5, osCommerce 2.3.4'den ikinci daha güvenli bir yazılımdır.

Bir vaka çalışması olan bu çalışma, bir değişkenin, herhangi bir yazılım kalite modelinin bir öğesi olarak nasıl kullanılabileceğini de göstermektedir. Bu çalışma literatüre, kişiye bir yazılımın diğer yazılımlar arasındaki yazılım kalitesi ve güvenliği noktasında konumunu göstermede katkı sağlamaktadır. Ayrıca, bu çalışma üniversitelere bu tür bir testi kendi ÖBS yazılımları üzerinde gerçekleştirmeleri için pratik bir kılavuz sağlar.

ÖBS yazılım açıkları için, benzer programlama dili kullanan iki veya daha fazla ÖBS yazılımı karşılaştırılarak daha geniş bir çalışma yapılabilir. Bir sonraki çalışmada RIPS 3.1 ile belirlenen XSS açıklarından korunmak için Kirda vd. (2006: 330)'nin yazısında ele aldığı Noxes aracı kullanılarak bir yazılım testi

gerçekleştirilmesi planlanmaktadır. Böylece gelecekte o çalışmanın XSS açıklarını önlemede ne kadar başarılı olunduğuna bağlı olarak, aynı türden ÖBS yazılımlarını XSS açıklarından korumak için kullanılabilecektir.

Kaynaklar

Arifoğlu, A. ve Doğru, A. (2001), *Yazılım Mühendisliği*, Birinci Baskı, SAS Bilişim Yayınları, Ankara, ISBN 975-97197-2-1.

Atbaş, H. (2012), *Kaliteli Yazılım Nasıl Geliştirilir*, Birinci Baskı, Pusula Yayıncılık, Ankara, ISBN 978-9944-711-79-1.

Balzarotti, D., Cova, M., Felmetsger, V., Jovanovic, N., Kırda, E., Kruegel, C., ve diğ. (2008), "Saner: Composing Static and Dynamic Analysis to Validate Sanitization in Web Applications", *IEEE Symposium on Security and Privacy*, 18-21 Mayıs, ss. 387-401, Oakland, California, ABD, ISBN 978-0-7695-3168-7.

Burger, R. (2015), *The Top 6 Free and Open Source School Administration Software*, http://blog.capterra.com/the-top-6-free-school-administration-software/, (Erişim Tarihi: 10.03.2019).

Bussieck, M. R., Dirkse, S. P., Meeraus, A. ve Pruessner, A. (2004), "Software Quality Assurance For Mathematical Modeling System", *The Next Wave in Computing, Optimization, and Decision Technologies*, (29), ss. 267-284, ISBN 978-0-387-23529-5.

Camoğlu, K. (2010), *Yazılım Kalitesi ve Test*, http://www.chip.com.tr/blog/kadircamoglu/yazilim-kalitesi-ve-test_5582.html, (Erişim Tarihi: 10.03.2019).

Dahse, J. ve Holz, T. (2014), "Simulation of Built-in PHP Features for Precise Static Code Analysis", *Network and Distributed System Security (NDSS) Symposium*, 23-26 Şubat, ss. 1-15, San Diego, California, ISBN 1-891562-35-5.

Ebert, C., Dumke, R., Bundschuh, M. ve Schmietendorf, A. (2005), Best Practices in Software Measurement: How to Use Metrics to Improve Project and Process Performance, *Springer Science & Business Media, Berlin*, ISBN 978-3-540-26734-8

Ergin, İ. ve Akseki, B. (2012), "Lisansüstü Eğitimde Kullanılan Öğrenci Bilgi Sistemi", *Eğitim ve Öğretim Araştırmaları Dergisi*, 1(2), ss. 364-380, ISNN 2146-9199.

Flatten, O. P., Mccubbrey, D. L., O'riordan, D. P., ve Burges, K. (1992), *Fundations of Bussines Systems*, Dryden Press, New York, ISBN-13 978-0030764813

Gürbüz, Ali (2010), *Yazılım Test Mühendisliği*, Papatya Yayınevi, İstanbul, ISBN 9786054220090.

Hills, M., Klingt, P. ve Vinju, J. (2013), "An Empirical Study of PHP Feature Usage: A Static Analysis Perspective", *International Symposium on Software Testing and Analysis (ISSTA)*, 15-20 Temmuz, ss. 325-335, Lugano, İsviçre, ISBN 978-1-4503-2159-4.

HUANG, Y. W., Yu, F., Hang, C., Tsia, C. H., Lee, D. T. ve Kuo, S. Y. (2004), "Securing Web Application Code by Static Analysis and Runtime Protection", *13th international conference on World Wide Web*, 17-22 Mayıs, ss. 40-52, New York, ABD, ACM 1-58113-844-X/04/0005.

IEEE (1998), *1061-1992 - IEEE Standard for a Software Quality Metrics Methodology*. ABD: Institute of Electrical and Electronics Engineers, Inc, Piscataway, New Jersey, ABD, ISBN 1-55937-529-9.

Joomla. (2019), *About Joomla*, https://www.joomla.org/about-joomla.html, (Erişim Tarihi: 08 03, 2019).

Jovanovic, Irena (2009), "Software Testing Methods and Techniques", *IPSI BgD Transactions on Internet Research*, 5(1), ss. 31-41, ISSN 1820- 4503.

Jovanovic, N., Kruegel, C. ve Kırda, E. (2010), "Static Analysis for Detecting Taint-Style Vulnerabilities in Web Applications", *Journal of Computer Security*, 18(5), ss. 861–907, DOI 10.3233/JCS-2009-0385.

Kirda, E., Kruegel, C., Vigna, G. ve Jovanovic, N. (2006), "Noxes: A Client-Side Solution for Mitigating Cross-Site Scripting Attacks", *The 31st ACM/SIGAPP Symposium on Applied Computing*, 23-27 Nisan, ss. 330-337, New York, ABD, ACM 1-59593-108-2/06/0004.

Kumar, B. A. (2011), "Thin Client Web-Based Campus Information", *International Journal of Software Engineering & Applications (IJSEA)*, 2(1), 13-26, DOI 10.5121/ijsea.2011.2102.

Kurtel, K. ve Eren, Ş. (2008), "Yazılım ölçümü: Genel Bir Bakış", *Yazılım Kalitesi ve Yazılım Geliştirme Araçları Sempozyumu*, 9-10 Ekim, Hava Harp Okulu, İstanbul Kültür Üniversitesi.

Moodle. (2019), *About Moodle*, https://docs.moodle.org/36/en/About_Moodle, (Erişim Tarihi: 10.03.2019).

osCommerce. (2019), *Sell Online*, https://www.oscommerce.com/, (Erişim Tarihi: 10.03.2019).

OWASP. (2017), *OWASP Top 10*, http://owasptop10.googlecode.com/files/OWASP_Top_10_2017_(en).pdf, (Erişim Tarihi: 10.03.2019).

Ripstech. (2019), *Features*, https://www.ripstech.com/features/, (Erişim Tarihi: 10.03.2019).

Safana, A. I. ve Ibrahim, S. (2010), "Implementing Software Test Management Using SpiraTeam Tool", *Advance Software Engineering Center 2010 Fifth International Conference on Software Engineering Advances*, 22-27 Ağustos, ss. 447-452, Nice, Fransa, DOI 10.1109/ICSEA.2010.76.

Sarıdoğan, Erhan (2004), *Yazılım Mühendisliği*, Birinci Baskı, Papatya Yayıncılık, İstanbul, ISBN 978-975-6797-57-0.

Sarıdoğan, Erhan (2008), Yazılım mühendisliği: *Profesyonel Yazılım Geliştirmeyi Öğrenmek İsteyenler İçin,* İkinci Baskı, Papatya Yayıncılık, İstanbul, ISBN 978-975-6797-57-0.

Stanujkic, D. ve Jovanovic, R. (2012), "Measuring a Quality of Faculty Website Using ARAS Method", *International Scientific Conference on Contemporary Issues In Business, Management And Education*, 9-10 Mayıs, ss. 545-554, Vilnius, Litvanya, ISBN 978-609-457-323-1.

Şahinaslan, Ö., Emin Borandağ Ve Şemseddin Aksoy (2011), "Web Tabanlı Uygulamalarda Performansı Etkileyen Unsurlar", *XIII. Akademik Bilişim Konferansı Bildirileri*, 2-4 Şubat, 599-604. Malatya, İnönü Üniversitesi Akademik Bilişim.

Xie, Y. ve Aiken, A. (2006), "Static Detection of Security Vulnerabilities in Scripting Languages", *15th USENIX Security Symposium*, 31 Temmuz-4 Ağustos, ss. 1-14, Vancouver, Kanada, ISBN 1931971455.

Yıldırım, B. F. (2015), "Çok Kriterli Karar Verme Problemlerinde Aras Yöntemi", *Kafkas Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 6(9), ss. 285-296, ISSN 1309 - 4289.

Zavadskas, E. K. ve Zenonas, T. (2010), "A New Additive Ratio Assessment (ARAS) Method in Multicriteria Decision-Making", *Technological and Economic Development of Economy*, 16(2), ss. 159-172, ISSN 1322 3613.

Zuse, Horst (1998), *A Framework of Software Measurement*. Walter de Gruyter, New York, ISBN 3-11-015587-7.