Tarifname

BİR YAZILIMIN DAYANIKLILIĞINI ÖLÇMEYE YÖNELİK BİR YÖNTEM

Teknik Alan

Buluş, ölçme yöntemi ile ilgilidir.

Buluş özellikle, bir yazılımın girdilere karşı dayanıklılığını ölçmeye yönelik bir yöntem ile ilgilidir.

Tekniğin Bilinen Durumu

Günümüzde, bir yazılımın girdilere karşı dayanıklılığını ölçmeye yönelik bir yöntem bulunmamaktadır. Mevcut yöntemler, bir yazılım kaynak kod içerisinde zayıf olmasına neden olan veya zafiyet içeren kod parçalarını ve bulundukları yeri göstermek üzerinedir, bir başka deyişle hata/zafiyet tespiti yapmak üzerinedir. Ancak bu yöntemler geliştiricilere hiçbir şekilde ölçüm verisi sunmamaktadır. Bu nedenle geliştiriciler yazılım kaynak kod içerisinde girdilere karşı zayıf olmasına neden olan zafiyetlerin ne kadar kritik olduğunu görememektedir.

Yine mevcut teknikte bununla beraber geliştiricilerin yönlendirildiği zafiyet ve zayıf noktalar üzerinde gerekli düzeltmeleri ve iyileştirmeleri uyguladıktan sonra, yapılan değişikliklerinin yazılım üzerinde dayanıklılık açısından ne kadar etkili olduğunu gözlemlenememektedir. Bu tür ölçümler gösterilemediğinden geliştiriciler geliştirdikleri yazılımlar üzerinden tam bir kontrol sağlayamamaktadırlar.

Yine mevcut teknikte yer alan TR2011/06312 no'lu, "Bir test performans sistemi" başlıklı ve "G06F 11/36" tasnif sınıflı buluş, yazılım parçalarının testlerinin gerçeklendiği test sistemlerinde tanımlanan test dizayn örüntülerinin kullanıldığı bir test performans sistemi ile ilgilidir. Bilgisayar programlarının performans testlerinin gerçeklendiği buluş konusu sistem, performans testinin uygulanacağı kurumda kullanılmakta olan bilgisayar programlarını ve bilgisayar programları üzerinde yapılan değişikliklerin saklandığı en az bir versiyonlama birimi, bilgisayar programları üzerinde uygulanan performans testlerinde ortaya çıkan sorunlara karşı sunulan

çözümlerin örüntüye dönüştürülmesi ile oluşturulan en az bir test dizayn örüntüleri birimi ve bilgisayar programı üzerinde test dizayn örüntüleri ile performans testinin koşulduğu en az bir performans birimi içermektedir.

Yine, TR2011/03060no'lu, "Bir test otomasyon sistemi ve yöntemi" başlıklı ve "G06F 11/36" tasnif sınıflı buluş, gerçek test mühendislerinin yardımıyla ayarlanan (configure) ve alınan aksiyonları tarihsel veritabanında saklayıp değerlendirerek regresyon testlerini kendiliğinden koşabilen sistem ve yöntem ile ilgilidir. Regresyon testlerini otomasyon senaryolarıyla uygulamalar üzerinde tek başına koşan sistem, kurumda geliştirilen ve test ortamına yüklenen projelerin kaynak kodlarının bulunduğu en az bir kurumsal test ortamı kaynak kod deposu, kod deposunda bulunan kodların, sorgulaması yapılan uygulama ile ilişkisini kontrol eden, ilişkili olanlarda hazırlanan GUI otomasyon test senaryolarını koşan, var olan kod parçalarıyla kaynak kod deposundaki değişiklikleri kontrol ederek farklılaşan GUI'lerle ilgili doğrulama fonksiyonlarını çalıştıran ve tutarsız noktalarda doğrulama tavsiye motorunu devreye sokan en az bir test otomasyon ve takip birimi, kod ilişkilerinin ve test otomasyon senaryo kod parçalarının saklandığı en az bir test otomasyon ve takip veri tabanı, hata kayıtlarının açıldığı en az bir kurumsal test senaryo ve hata giriş birimi ve tutarsız noktaların tespiti için kullanılan en az bir ayırıcı içermektedir.

Yine, TR 2012/07563 no'lu, "Bir hata tanımlama ve çözüm önerisi sunma sistemi ve yöntemi" başlıklı ve "G06F 11/36" tasnif sınıflı buluş, yazılım yaşam döngüsünde oluşan hataların önceden saklanan hatalar yardımı ile tanımlanıp uygun çözüm önerilerinin sunulduğu bir hata tanımlama ve çözüm önerisi sunma sistemi ve yöntemi ile ilgilidir. Yazılım yaşam döngüsünde ortaya çıkan hataları tanımlayan ve hatalara ilişkin çözüm önerisi sunan sistem, yazılım yaşam döngüsünde ortaya çıkan hataların ve hatalara ilişkim çözüm önerilerinin saklandığı en az bir veri tabanı, yazılım geliştirme esnasında, test aşamasında veya yazılımın entegre olduğu sistemlerde ortaya çıkan hataları veri tabanında yer alan hatalar ile karşılaştırarak hataları tanımlayan en az bir eşleştirme birimi ve hataların tanımlanması ile söz konusu hataya ilişkin en uygun çözüm önerisini sunan en az bir öneri birimi içermektedir.

Yine, TR 2011/09708 no'lu, "Test otomasyon senaryo (case) sayısı öneri sistemi" başlıklı ve "G06F 11/36" tasnif sınıflı buluş, test aşamasında olan yazılım projeleri için yazılması ve/veya koşulması gereken test otomasyon senaryo (case) sayısının belirlenmesinde kullanılan bir senaryo (case) sayısı öneri sistemi ile ilgilidir. Yazılacak ve/veya koşulacak test otomasyon senaryo sayısının önerilmesini sağlayan sistem, projede görev alan kişilere sorulacak soruların saklandığı en az bir soru veri tabanı, soruların projede görev alan kişilere yöneltildiği en az bir arayüz, arayüz aracılığı ile sorulara alınan cevapların saklandığı en az bir kayıt birimi, her bir soruya alınan cevapların hesaplanan ağırlıklarının saklandığı en az bir ağırlık veri tabanı, her bir soru için önceden belirlenen senaryo sayılarının saklandığı en az bir baz senaryo veri tabanı ve her soru için ağırlık veri tabanından alınan cevapların hesaplanan ağırlıkları ile baz senaryo biriminde saklanan baz senaryo değerlerinin eşleştirilerek çarpılması sonucu yazılacak ve/veya koşulacak test otomasyon senaryo sayısının belirlendiği en az bir önerme birimi içermektedir.

Buluşun Amacı

Tekniğin bilinen durumunda yer alan dezavantajları ortadan kaldırmak üzere buluşun bir amacı, bir yazılımın girdilere karşı dayanıklılığını ölçmesidir.

Buluşun bir diğer amacı, geliştiricilere ölçüm verisi sunmasıdır.

Buluşun bir diğer amacı, geliştiricilerin yönlendirildiği zafiyet ve zayıf noktalar üzerinde gerekli düzeltmeleri ve iyileştirmeleri uyguladıktan sonra, yapılan değişikliklerinin yazılım üzerinde dayanıklılık açısından ne kadar etkili olduğunu gözlemleme imkanı sunmasıdır.

Yukarıdaki avantajları sağlamak üzere buluş, bir yazılımın dayanıklılığını ölçmeye yönelik; FIPS”(Fonksiyon Girdi Parametresi Durumu) düğümleri ile, kod içerisinde birbirinden bağımsız olarak zafiyetler için gerekli önlemlerin alınıp alınmadığını kontrol edilerek sayısal değerlerin alınması, FIPS düğümlerinde alınan bu değerlerin incelenen zafiyetlere ilişkin zafiyet düğümlerine aktarılması ve bilgilerin işlenmesi, zafiyet düğümlerinde işlenen bilgilerin uygulama düğümüne aktarılması, uygulama

düğümünün gelen bilgileri değerlendirerek, yazılımın genel olarak dayanıklılığına dair çıkarsama yaparak bir ölçüm vermesi işlem adımlarını içeren bir yöntemdir.

Şekillerin Açıklaması

Şekil 1 jQuery Kütüphanesinin Bayes ağları ile Dayanıklılığının Ölçülmesi diyagramı görünümüdür,

Şekil 2 buluş konusu yazılım dayanıklılığını Bayes ağları ile ölçme akış diyagramı görünümüdür.

Referans Numaralarının Açıklaması

Ref. No	Ref. Adı
A	FIPS (Function Input Parameter Status/Fonksiyon Girdi Parametresi Durumu) düğümleri
B	Zafiyet düğümleri
C	Yazılım dayanıklılık uygulaması düğümü
10	Başla adımı
11	yazılım kaynak kodunun statik analiz aracına gönderilmesi
12	Statik analiz aracı ile yazılım kaynak kodu üzerinden veri akışı analiz gerçekleştirilmesi
13	kodun işleyişini çizgesel olarak ifade eden bir akış çizgesi çıktısı elde edilmesi
20	Girdi doğrulama tespit işlemi
21	akış çizgesi çıktısı analiz edilerek, fonksiyon ve parametreleri hedef alınarak bilgiler çıkarılması
22	Fonksiyon ve parametrelere ait bilgilerin elde edilmesi
23	parametrelerin kullanılmadan önce doğrulanıp doğrulanmadıkları analiz edilmesi
24	girdiler hakkında doğrulanmış veya doğrulanmamış bilgileri elde edilmesi
25	girdi doğrulama zafiyetleri için doğrulanma kuralları çıkarılması

26	24 ve 25 no'lu adımlardan gelen girdilerin doğrulama denetiminin yapılması
27	26 no'lu adımdan gelen girdilerin doğrulama denetiminin yapılması
30	Dayanıklılık hesaplanması işlemi
31	önceden hesaplanmış ve incelenen 6 girdi doğrulama zafiyeti için zafiyet puanları hesaplanması
32	Zafiyet istatistiklerinin elde edilmesi
33	30., 31. ve 32. adımlardan toplanılan veriler oluşturulan Bayes ağı modeline aktarılması
34	Bayes ağı dosyası oluşturulması
35	yazılan ya da geliştirilen yazılım kaynak kodunun dayanıklılık hesaplaması yapılması
36	olasılıksal bir dayanıklılık değeri elde edilmesi
37	Sonlanma adımı

Buluşun Detaylı Açıklaması

Buluş, bir yazılımın girdilere karşı dayanıklılığını ölçmeye yönelik bir yöntemdir. Buluş konusu yöntem, popüler bir yazılım kütüphanesi olan “jQuery” üzerinde uygulanmıştır. Şekil 1’deki “FIPS”(Function Input Parameter Status/Fonksiyon Girdi Parametresi Durumu) düğümleri (A), kod içerisinde birbirinden bağımsız olarak zafiyetler için gerekli önlemlerin alınıp alınmadığını kontrol ederek, sayısal değerlerini alır.

Söz konusu “FIPS” düğümlerinden (A) gelen sayısal değerler yazılım kaynak kod içerisinde incelenen zafiyetler için önlem alma kodları içerme değerlerini ifade eder. Bu değerler incelenen zafiyetlere ilişkin zafiyet düğümlerine (B) aktarılır. Aktarılan değerler bu zafiyet düğümlerinde (B) CWSS (Common Weakness Scoring System/Genel Zafiyet Puanlandırma Sistemi) ve NVD (National Vulnerability Database/Ulusal Zafiyet Veri Tabanı)’den gelen bilgiler ile işlenir.

Son olarak zafiyet düğümlerindeki (B) bilgiler uygulama düğümüne yani genel olarak yazılımın dayanıklılığını ölçecek düğüme (C) aktarılır. Uygulama düğümü (C) gelen

bilgileri değerlendirerek, yazılımın genel olarak dayanıklılığına dair çıkarsama yaparak bir ölçüm verir.

Buluş konusu yazılımın dayanıklılığını ölçecek düğümde (C) gerçekleşen işlem adımları aşağıdaki gibidir;

10. başla adımı,

11. Yazılan ya da geliştirilen yazılım kaynak kodu statik analiz aracına gönderilir.

12. Statik analiz aracı ile yazılım kaynak kodu üzerinden veri akışı analiz gerçekleştirilir. Analiz yazılım kaynak kodu içerisinde verinin kod içerisinde nasıl ilerlediğini ve işlendiğinin bilgisini verilir.

13. Statik analiz aracı ile analiz edilen kaynak kodun sonucunda doğan, kodun işleyişini çizgesel olarak ifade eden bir akış çizgesi çıktısı elde edilir.

20. Girdi doğrulama tespit işlemi

21. 13. Aşamada elde edilen akış çizgesi çıktısı analiz edilerek, fonksiyon ve parametreleri hedef alınarak bilgiler çıkarılır.

22. 21. Aşamadan sonra aşağıdaki bilgiler elde edilir.

Fonksiyonların

a. İsimlerini
b. Kaç parametre içerdiğini
c. Tanımlandığı satır sayısı

Parametrelerin

a. İsimlerini
b. Bulunduğu fonksiyon
c. İlk kullanıldığı satır ve sütun bilgisi.
d. İlk doğrulandığı satır ve sütun bilgisi.

23. 22. Aşamadan çıkarılan bilgiler doğrultusunda, parametrelerin kullanılmadan önce doğrulanıp doğrulanmadıkları analiz edilir. Eğer kullanılmadan önce doğrulanmışsa, parametre doğrulanmış, kullanıldıktan sonra doğrulanmış ya da doğrulanmamışsa parametre doğrulanmamış olarak kabul edilir.

24. 23. Aşamada yapılan girdi doğrulama analizi sonucunda girdiler hakkında doğrulanmış veya doğrulanmamış bilgileri elde edilir.

25. Konfigürasyon dosyasında tanımlanmış ve incelenen "6 girdi doğrulama zafiyetleri" için doğrulanma kuralları çıkarılır. Bu kurallar geliştirici tarafından güncellenebilir, eklenilebilir ve çıkarılabilir. Tanımlanan "6 girdi doğrulama zafiyeti" (XSS/Siteler Arası Betik Çalıştırma, SQL Injection/SQL Enjeksiyonu, Operating System Command Injection/İşletim Sistemi Komut Enjeksiyonu, Path Traversal/Dizin Gezinme, Uncontrolled String Format/Kontol Edilmemiş Metin Biçimi, Buffer Errors/Arabellek Hataları), CWE (Common Weakness Enumeration/Genel Zafiyet Sayımı) tarafından tanımlanmış, doğrulanmamış fonksiyon parametreleri veya girdiler üzerinden yapılan saldırılar kategorisinde bulunan 6 tip girdi doğrulama zafiyetidir.

26. 24. Aşamadaki analiz sonucundan gelen doğrulanmış parametre bilgileri ile 25.aşamadan gelen ön tanımlı doğrulama zafiyetleri kuralları kullanılır. Girdi doğrulama zafiyetlerine karşı kuralların uyulup uyulmadığı kontrol edilerek, incelenen 6 girdi doğrulama zafiyetine karşı önlem alınıp alınmadığının analizi gerçekleştirilir.

27. 9. Aşamadaki analiz sonucundan gelen girdi doğrulama denetim bilgisi çıkarılır.

30. Dayanıklılık hesaplanması işlemi;

31. Common Weakness Scoring System/Genel Zafiyet Puanlandırma Sistemi (CWSS) tarafından önceden hesaplanmış ve incelenen 6 girdi doğrulama zafiyeti için hesaplanan zafiyet puanları, 13. aşamada kullanılmak üzere çıkarılır.

32. National Vulnerability Database/Ulusal Zafiyet Veri Tabanı (NVD)'den 15 yıllık incelenen 6 girdi doğrulama zafiyetleri hakkında rapor edilme sıklıklarının istatistikleri elde edilerek, 13. Aşamada kullanılmak üzere çıkarılır.

33. 10., 11. ve 12. aşamalardan toplanılan veriler oluşturulan Bayes ağı modeline aktarılır.

34. 13. Aşamada oluşturulan Bayes ağı modeli, ileride dayanıklılık hesabı yapabilmek ve çalıştırmak için, Bayes ağının düğüm, ayrıt ve olasılıksal bilgilerinin tutulduğu bir Bayes ağı dosyası oluşturulur.

35. 14. aşamada oluşturulan Bayes ağı dosyasındaki verilerden çıkarsama yaparak, yazılan ya da geliştirilen yazılım kaynak kodunun dayanıklılık hesaplaması yapılır.

36. Yazılım kaynak kodunun dayanıklılık hesabı sonucundan olasılıksal bir dayanıklılık değeri elde edilir.

37. Sonlanma adımı.

Buluş konusu yöntem uygulamasında; incelenen jQuery yazılım kütüphanesi yöntemimizle sadece “Buffer Overflow/Arabellek Aşımı” zafiyetine karşı çok az önlem aldığını, incelenen diğer zafiyetlere karşı ise hiç önlem almadığını tespit etmiştir. Bu bilgiler doğrultusunda jQuery yazılım kütüphanesinin dayanıklılığı %5.039 olarak ölçülmüştür.

İSTEMLER

1. Buluş, bir yazılımın dayanıklılığını ölçmeye yönelik bir yöntem olup, özelliği,
   - FIPS düğümleri (A) ile, kod içerisinde birbirinden bağımsız olarak zafiyetler için gerekli önlemlerin alınıp alınmadığını kontrol edilerek sayısal değerlerin alınması,
   - FIPS düğümlerinde (A) alınan bu değerlerin incelenen zafiyetlere ilişkin zafiyet düğümlerine (B) aktarılması ve bilgilerin işlenmesi,
   - zafiyet düğümlerinde (B) işlenen bilgilerin uygulama düğümüne (C) aktarılması, uygulama düğümünün (C) gelen bilgileri değerlendirerek, yazılımın genel olarak dayanıklılığına dair çıkarsama yaparak bir ölçüm vermesi,
   işlem adımlarını içermesidir.

2. İstem 1'e uygun bir yöntem olup, özelliği, jQuery” üzerinde uygulanmasıdır.

3. İstem 1 ve 2'ye uygun bir yöntem olup, özelliği, aktarılan değerler bu zafiyet düğümlerinde (B) CWSS ve NVD’den gelen bilgiler ile işlenmesidir.

4. İstem 1, 2 ve 3'e uygun bir yöntem olup, özelliği, bahsedilen uygulama düğümünde (C) aşağıdaki işlem adımlarının gerçekleşmesidir;
   - başla adımı (10),
   - yazılım kaynak kodunun statik analiz aracına gönderilmesi (11),
   - statik analiz aracı ile yazılım kaynak kodu üzerinden veri akışı analiz gerçekleştirilmesi (12),
   - kodun işleyişini çizgesel olarak ifade eden bir akış çizgesi çıktısı elde edilmesi (13),
   - girdi doğrulama tespit işlemi (20),
   - dayanıklılık hesaplanması işlemi (30),
   - sonlanma adımı (37).

5. İstem 4'e uygun bir yöntem olup, özelliği, girdi doğrulama tespit işlemi (20) aşağıdaki işlem adımlarını içermesidir;

— akış çizgesi çıktısı analiz edilerek, fonksiyon ve parametreleri hedef alınarak bilgiler çıkarılması (21),
— fonksiyon ve parametrelere ait bilgilerin elde edilmesi (22),
— parametrelerin kullanılmadan önce doğrulanıp doğrulanmadıkları analiz edilmesi (23),
— girdiler hakkında doğrulanmış veya doğrulanmamış bilgileri elde edilmesi (24),
— girdi doğrulama zafiyetleri için doğrulanma kuralları çıkarılması (25),
— 24 ve 25 no'lu adımlardan gelen girdilerin doğrulama denetiminin yapılması (26),
— 26 no'lu adımdan gelen girdilerin doğrulama denetiminin yapılması (27).

6. İstem 5'e uygun bir yöntem olup, özelliği, fonksiyon ve parametrelere ait bilgilerin elde edilmesi (22) işlem adımında; 21. Aşamadan sonra aşağıdaki bilgiler elde edilmesidir; fonksiyonların isimlerini, kaç parametre içerdiğini, tanımlandığı satır sayısı; Parametrelerin, İsimlerini, bulunduğu fonksiyon, ilk kullanıldığı satır ve sütun bilgisi, ilk doğrulandığı satır ve sütun bilgisi.

7. İstem 4'e uygun bir yöntem olup, özelliği, dayanıklılık hesaplanması işleminin (30) aşağıdaki işlem adımlarını içermesidir;
— önceden hesaplanmış ve incelenen "6 girdi doğrulama zafiyeti" için zafiyet puanları hesaplanması (31),
— zafiyet istatistiklerinin elde edilmesi (32),
— 30., 31. ve 32. adımlardan toplanılan veriler ile oluşturulan bayes ağı modeline aktarılması (33),
— bayes ağı dosyası oluşturulması (34),
— yazılan ya da geliştirilen yazılım kaynak kodunun dayanıklılık hesaplaması yapılması (35),
— olasılıksal bir dayanıklılık değeri elde edilmesi (36).

ÖZET

BİR YAZILIMIN DAYANIKLILIĞINI ÖLÇMEYE YÖNELİK BİR YÖNTEM

Buluş, bir yazılımın girdilere karşı dayanıklılığını ölçmeye yönelik; FIPS (Fonksiyon Girdi Parametresi Durumu) düğümleri (A) ile, kod içerisinde birbirinden bağımsız olarak girdilere karşı zafiyetler için gerekli önlemlerin alınıp alınmadığını kontrol edilerek sayısal değerlerin alınması, FIPS düğümlerinde (A) alınan bu değerlerin incelenen zafiyetlere ilişkin zafiyet düğümlerine (B) aktarılması ve bilgilerin işlenmesi, zafiyet düğümlerinde (B) işlenen bilgilerin uygulama düğümüne (C) aktarılması, uygulama düğümünün (C) gelen bilgileri değerlendirerek, yazılımın genel olarak dayanıklılığına dair çıkarsama yaparak bir ölçüm vermesi işlem adımlarını içeren bir yöntem ile ilgilidir.

Şekil 2