Önemli Güvenlik Gereksinimleri

www.sourceflake.com
Bedirhan Urgun

Ocak 4

2016

Bu doküman özellikle kurum içi geliştirilen web servislerin yazılım ve entegrasyon sürecinde dikkat edilmesi gereken temel güvenlik maddelerini detaylandırır. Bu anlamda doküman, minimum gereksinim dokümanı olarak kullanılabilir.

Web Servisleri

Önemli Güvenlik Gereksinimleri - Web Servisleri

Giriş

HTTP üzerinden işleyen web servisleri genel olarak klasik ve RESTful olarak ikiye ayrılırlar. RESTful servisler HTTP istek-cevap süreçleri ile oluşturulur ve JSON gibi standartları kullanır. Dolayısıyla uygulanması gereken güvenlik prosedürleri web uygulamaları güvenlik prosedürleri ile aynıdır [2].

Klasik web servisleri genellikle SOAP, XML-RPC gibi standartları kullanırlar. Yine HTTP protokolü ile istekler ve cevaplar oluşturulur ancak bilgi taşıma yapısı için temelde XML kullanılmaktadır. Uygulanması gereken güvenlik standartları, örneğin SOAP protokolünde, belirlenmiş olup bu standartların uygulanması gerçek hayatta her zaman kolay olmayabilir.

Bu nedenle günümüz web servisleri klasik ve RESTful tiplerinin ortasında yer alabilirler ve hiçbir zaman belli bir standarda tamamen uymayabilirler. Önemli olan web servislerin servis odaklı mimari prensiplerine uygun olması ve tekrar tekrar kullanılabilmeleridir. Ancak uygulanması gereken güvenlik politikaları bu standart problemlerinden bağımsızdır.

Güvenlik Politikaları

Uygulama

Arka uç çalışan bütün uygulamaların, önemli uygulama geliştirme güvenlik problemlerine karşı bağışık olması gerekmektedir [3]. Bu konu başlı başına kapsamlı bir konu olduğundan detaylarına bu dokümanda değinmek imkânsızdır. Ancak genellikle web servislerinin Internet'e açılan web uygulamaları tarafından kullanıldığı öngörülürse özellikle IDOR gibi zafiyetler tamamıyla yok edilmelidir.

Kimlik Doğrulama

Web servislerini başarılı bir şekilde çağırabilecek kullanıcıları belirleyebilmek amacı ile web servislerinde kimlik doğrulama mekanizması uygulanmalıdır. Web servislerini çağırabilecek kullanıcılar Active Directory/LDAP üzerinde oluşturulabilir. Bu uygulama kullanıcıları "never expire" özellikleri ile operasyon kontağının sorumluluğunda oluşturulur.

Kimlik bilgilerinin çağırılan web servislerinde hangi alanda tutuldukları önemsizdir ancak bu bilgiler URL'de taşınmamalıdırlar.

Kimlik Doğrulama Anti Otomasyon

Kimlik doğrulama isteklerinde saldırganların kullanıcı adı ve şifre denemelerine engel olabilmek amacı ile bazı kontroller uygulanmalıdır.

Anti brute-force kontrolleri, IP/istek sayısı eşik değeri kontrolü olabileceği gibi belirli bir kullanıcı adı ile yapılan hatalı denemeden sonra kullanıcı hesabının kilitlenmesi da olabilir. İkinci kontrol tipi bir DoS saldırısına yol açabileceğinden genel olarak kullanıcı adları belirli bir algoritmaya göre üretilmemelidir. Örneğin CRMUSER01, CRMUSER02, CRMUSER05.

Kullanıcı adları **olabildiğince** birbirinden bağımsız, tahmin edilemez olmalıdır.

Oturum Yönetimi

Kimlik doğrulama işlemi gerçekleştirildikten sonra, web servisleri çağırılan operasyonlarda da kullanıcıların kimliklerini doğrulamak zorundadırlar. Bunu gerçekleştirmek için, kullanıcı adı ve şifre her web servis isteğine konulabilir.

Diğer bir yöntem ise kimlik doğrulama sonrası istemci tarafına bir güvenli bir (Güvenli rasgele algoritmaları ile üretilmiş, yeterince karmaşık ve uzun v.b.) oturum token'ı dönülmesidir [4]. Çıkış web servisi çağırılmadıkça ve örneğin 30 dk'lık eylemsizlik süresi geçmedikçe bu token geçerli olacaktır ve web servis isteklerinde çağrıldıkça geçerli olacaktır. Uygulama tarafında, kullanıcı adı üretilen bu token ile ilişkilendirilmelidir. Çıkış işleminde veya geçerlilik zamanı dolunca geçersiz kılınmalıdır.

Akan Veri Güvenliği

Web servisleri **HTTPS** üzerinden çalışmalıdır.

Yetkilendirme

Yetkilendirme, IP kısıtlaması gibi temel ve **en önemlisi uygulanması basit** bir şekilde olabileceği gibi daha karmaşık durumları yönetebilmek adına rol tabanlı veya izin tabanlı şekilde de gerçekleştirilebilir.

IP kısıtlaması detaylı yetkilendirme gerektiren durumlarda yetersiz kalır. Özellikle aynı IP'den gelen iki kullanıcının çağırabileceği web servisleri farklılaşırsa, bu durumda kullanıcılara atanan roller veya izinler yardımı ile detaylı ayrım yapılabilir.

Rol bilgileri de kimlik doğrulamasında olduğu gibi AD kullanılarak, IDM entegrasyonu ile oluşturulabilir. Her web servisi yetkilendirme kontrolü için kendisini çağıran kullanıcının gerekli hakkı olup olmadığı kontrol eder her seferinde kontrol eder.

Yapılandırma – Hassas Bilgi Yönetimi

Uygulama üzerinde AD/LDAP kullanıcısı ve AD/LDAP link bilgileri gibi hassas bilgiler bir yapılandırma dosyasında mümkünse şifreli halde tutulurlar. Şifreleme işlemi için kullanılan anahtar ise yine aynı uygulama üzerinde ama sadece uygulama kullanıcısı tarafından erişecek şekilde saklanabilir [4].

Yukarıdaki madde olabildiğince temel bir gereksinimden bahseder, anahtar saklama ve şifreleme gereksinimleri arttıkça hardware security module (HSM) gibi cihazların kullanılması kaçınılmazdır.

Hata Yönetimi

Web servislerinin çalışması esnasında oluşabilecek beklendik veya beklenmedik hatalar sonucu, kullanıcılara sistem hakkında detaylı bilgiler dönülmemelidir. Özellikle hata sonucu oluşabilecek .NET ve Java stacktrace'leri sistem hakkında çok detaylı bilgiler içerirler.

Bu bilgilerin kullanıcılara sızmaması için hem container'ların hata yönetim özellikleri (***web.xml*** içerisindeki *error-page*, ***Web.config*** içerisindeki *customErrors* direktifleri) hem de kullanılan dillerin hata yönetim API'ları (try/catch/finally) etkin bir şekilde kullanılmalıdır [3].

Kayıt Tutma

Kurumlarda kayıt işlemi sırasında alınması gereken aksiyonlar, kayıt altına alınması gereken bilgiler farklılık gösterebilir. Web servisleri tarafından yapılan işlemlerin kayıtları kurum içi kayıt standardına uygun gerçekleştirilmelidir.

Ancak hassas verilerin açık bir şekilde kayıt altına alınmaması, kayıtlar tutulurken CR/LF ve kayıtlar işlenirken Second Order Injection zafiyetlerine karşı önlemlerin alınmış olması gerekmektedir [5,6].

Girdi Denetimi

Web servislerinin güvensiz kaynaklardan aldığı bütün girdiler beyaz listelere göre kontrol edilmelidir. Güvensiz kaynaklar kullanıcıların kimlik doğrulanmış veya doğrulanmamış olabildiği gibi, bu kaynak bir güvenilmeyen veritabanı veya başka bir web servisi de olabilir.

Beyaz liste mantığı kara liste mantığının tersi olup sadece doğru ve güvenli bilinen girdilerin kabul edilmesidir. Örneğin bir kredi kartı numarasının yapısı (bir şablon veya bir algoritma ile) bellidir ve bu yapı dışında bir girdi kredi kartı numarası olarak kabul edilemez. Aynı şey E-Mail adresleri, URL'ler, dosya isimleri, v.b. için de geçerlidir.

İletişim

bedirhan.urgun@sourceflake.com

Referanslar

1. https://www.owasp.org/index.php/Web_Service_Security_Cheat_Sheet
2. https://wiki.mozilla.org/WebAppSec/Secure_Coding_Guidelines
3. https://github.com/owasp/devguide
4. http://code.google.com/p/owasp-esapi-java/
5. http://www.veracode.com/security/crlf-injection
6. https://haiderm.com/second-order-sql-injection-explained-example/