Kimlik Doğrulama

Amaç

- Web uygulamalarında güvenli kimlik doğrulama servisleri sağlayabilmek
- Uygulanın riski oranında makul kimlik doğrulama kontrolleri sağlanması
- Kimlik doğrulama sistemine çeşitli methodlarıyla saldıran saldırganların erişiminin engellenmesi.

Etkilenen Platformlar

Hepsi.

İlgili COBIT Konuları

DS5-Tüm bölümler incelenmeli. Bu bölüm COBIT tarafından belirlenen kontrol hedeflerinin hemen hemen hepsini kapsar.

En İyi Uygulamalar

- **Kimlik doğrulama işlemi kullanıcı yönetim işlemi kadar güçlüdür**; ve özel olarak kullanıcının yayımı ve kimlik tanımlama politikaları kadar.
- **Kendinize en uygun kimlik doğrulama formunu kullanın**; örneğin kullanıcı adı ve şifre ile kimlik doğrulama bloglar ve forumlar gibi az değerli sistemler için uygun iken; az değerli e-ticaret için SMS Soru Cevap - Challenge Response- (2005'te); yüksek miktarlarda para dönen e-ticaret (2007 itibariyle tüm e-ticaret siteleri) sistemleri, bankacılık ve ticaret sistemleri için işlem imzalama uygundur.
- **Önemli/Kritik işlemler için kullanıcıdan tekrar kimlik doğrulama isteyin** (örneğin kullanıcı normal düzeyinden yönetici düzeyinde işlem yapma hakkı verirken).
- **Kullanıcıyı değil; işlemi doğrulatın**.Oltacılar genellikle zayıf şekilde gerçekleştirilmiş kimlik doğrulama mekanizmalarından faydalanırlar.
- **Parolalar çok ciddi güvenlik önlemi değildirler**; çok önemli sistemlerin korunmasında uygun değiller.Onun için; bazı kontroller bunu yansıtabilir.16 karakterden kısa uzunlukta herhangi bir parola kaba kuvvet kullanılarak iki haftadan az bir süre içerisinde kırılabilir; bundan dolayı parola politikanız makul olmalıdır.
    1. Kullanıcılarınızı iyi parola üretmeleri için eğitin.
    2. Kullanıcılarınızın parolalarını güvenli şekilde tutabildikleri sürece not almalarına müsade ediniz.

3. Kullanıcılarınızı parolaları yerine söz öbekleri kullanmaları konusunda cesaretlendirin.
4. Parola yenileme süresini parolanın dayanıklılığına göre ayarlayın.8-16 karakter arası parolaların kırılması çok kolay değil; bundan dolayı parolanın kullanım süresinin dolması 30 günden fazla olmamalıdır; haricinde söz öbeklerinden oluşan parolalar 16 karakter parolalar gibi sıkı bir kullanım süresi kontrolüne ihtiyaç duymamakla birlikte 90 gün sonra parolanın yenilenmesi gerektiğine dair bir uyarı yeterli olur.

Genel Web Kimlik Doğrulama Yöntemleri

Temel ve Özet Kimlik Doğrulama

Hemen hemen tüm web ve uygulama sunucuları temel ve özet kimlik doğrulama kullamını destekler Bu web tarayıcının bir diyalog kutusu yardımıyla kullanıcıdan kullanıcı adı ve parola alıp; onları web sunucuya yollaması ve bu bilgiyi veri tabanına işlemesi ya da Active Directory'e işlemesi.

- Temel kimlik doğrulama kimlik bilgilerini şifresiz – okunabilecek – şekilde yollar; SSL olmaksızın yollanmamalı.
- HTTP 1.0 Özet Kimlik doğrulaması sadece kimlik bilgilerini biraz daha karmaşık hale getirir; -yeterli güvenliği sağlayamaz – kullanılmamalı.
- HTTP 1.1 Özet Kimlik doğrulama mekanizması bir "challenge response" mekanizması, kullanır; az önemdeki uygulamalar için kullanılabilir.

Temel ya da özet kimlik doğrulama mekanizmasının kullanılmasının temel sebebi :

- Kimlik bilgilerinin güvensiz dolaşımı.
- Kimlik doğrulama formlarının hem tekrar gönderme saldırıları hem de aradaki adam saldırılarından kötü etkilenmesi
- Bilginin güvenliği ve bütünlüğünün sağlanması için SSL'e ihtiyaç duyulması.
- Kullanıcı arayüzünün kötü olması.
- Tüm bu sebepler temel ve özet kimlik doğrulamasının kullanışsız bir şey olduğu göstermez.Bu yöntem geliştirme sitelerinin ya da çok kritik olmayan yönetimsel arayüz/panellerin korunmasında kullanılabilir,fakat bunun haricinde bu kimlik doğrulama biçimi tavsiye edilmemektedir.

Form Tabanlı Kimlik Doğrulama

Form tabanlı kimlik doğrulama web uygulama yazılımcısına kullanıcı paneli üzerinde çok daha fazla kontrol hakkı verir; bundan dolayı geniş bir çevre tarafından kullanılmaktadır.

Form tabanlı kimlik doğrulama; uygulamanın işini iyi şekilde yapmasına ; kimlik doğrulama ve yetkilendirmeyi iyi şekilde gerçekleştirmesine ihtiyaç duyar.Nadiren web uygulamaları bunu doğru şekilde yapar.Sisteminizde açık olup olmadığını kontrol etmek için 15 özel kontrol adımı gerekir ve bu güvenli şekilde kimlik doğrulama yapmak için minimum rakamdır.

Her ne şekilde mümkün olursa olsun;eğer form tabanlı kimlik doğrulama yapacaksanız; bir erişim kontrol bileşeni yazmaktansa ; güvenilen bileşeni tekrar kullanmaya çalışın.

Form tabanlı kimlik doğrulama şu saldırılardan etkilenmektedir:

- Yeniden gönderme saldırıları
- Araya giren adam saldırıları
- Şifresiz biçimde gönderilen – okunabilir- kimlik bilgileri
- Daha çok yetki kapma saldırıları
- Zayıf parola kontrolleri

Ve “Sömürülebilir olduğunuzu nasıl anlarsınız” isimli dökümanda belirtilen saldırıların çoğu.

Kimlik bilgilerinin değişimi sırasında SSL kullanmak ve kontrollerin SSL ile gerçeklenmesi çok önemlidir.Web uygulama tasarımcıları için en önemli konu korunacak verinin çok önemli olmaması durumunda bu kontrolleri gerçekleştirmek için gereken maliyettir.Güvensizlik kaygılarından kurtulunması amacıyla kurulacak karmaşık kimlik doğrulama yapısının maliyetiyle; kaygılarımız arasında belli bir denge olmalıdır.

Bütünleşik Kimlik Doğrulama

Bütünleşik kimlik doğrulama genellikle Microsoft IIS web sunucusu ve ASP.NET uygulamaları ile yazılmış intranet uygulamalarında kullanılır. Diğer çoğu web sunucuları bu kimlik doğrulama biçimini önermez.Güvenli olmasına karşın[1] –kullanıcı taraflı sertifika kimlik doğrulamasının Kerberos ile kullanılmasıyla eşit – Active Directory tabanlı integrasyonu(hiçbir kimlik bilgisinin uygulama tarafından saklanmaması) pek internet düzeyinde uygulamalarda kullanılmaz.

Eğer bir intranet uygulaması geliştiriyor ve geliştirme ortamınız bütünleşik kimlik doğrulamayı destekliyorsa; kullanın.Bu sizin için kimlik doğrulama ve yetkilendirme kontrollerini yazarken daha az iş, kullanıcılar için daha az kimlik bilgisi bilme/hatırlama ve aynı kimlik doğrulama ve yetkilendirme alt yapısının tekrar kullanılması demektir.

Sertifika Tabanlı Kimlik Doğrulama

Sertifika tabanlı kimlik doğrulama bir çok web ve uygulama sunucusunda kullanılmaktadır.Web sitesi sertifika yayınlar (ya da başkaları tarafından yayınlanmış bir sertifikaya güvenir).Bu sertifikalar web sunucusunun kimlik doğrulama veritabanına yüklenir; ve sonraki web tarayıcı oturumlarında sunucuya önerilen sertifikalar ile karşılaştırılır.Eğer sertifikalar tutuyorsa; kullanıcının kimliği doğrulanmıştır.

Kimlik doğrulamanın kalitesi/güvenilirliği kesin olarak sertifikaları yayınlarken kullanılan açık anahtarın kalitesine bağlıdır.

Sertifika tabanlı kimlik oturum açmanın bazı dezavantajları vardır :

- Bir çok kullanıcı bilgisayarını paylaşır ve başkasının bilgisayarına oturduğu an sertifikalarını da yanında getirmesi gereklidir.Eğer kullanıcı sertifikayı kurmak zorunda ise bu önemsiz bir şey değildir.Bir çok kullanıcı sertifikaları nasıl kurup/kaldıracağını bilmemektedir.
- Bir web tarayıcı üzerinde sertifikaların yönetimi bir çok örnekte kolay değildir.
- Sertifika tabanli kimlik doğrulama extranet ortamlarında da kullanımı mümkündür.
- Özel sertifika sunucularına güven son kullanıcı güven kararlarına bağlıdır; örneğin güvenlik sertifikası verme yetkisine sahip şirketlerin verdiği sertifikaların son kullanıcı tarafından yüklenmesi, ve genellikle son kullanıcılar bu kararı verecek düzeyde değillerdir.
- Sertifikaların tutarları ve açık sertifika şirketleri iş modelindeki yerleri; tedarik edilmeleri için gereken tutar ile çok ilgili değildir, bu sonuçla çok sayıda kullanıcılı bir açık sertifika veritabanı oluşturmak pahalıdır.
- CA ‘lerin özellikle sertifika yenileme işlemlerindeki kötü yönetimiyle birlikte;sertifika tabanlı oturum açma genellikle başarısız olur.Telstra’nın online ödeme servisi bunun iyi bir örneğidir.Bir evrede; sadece sayısal sertifikalar Kabul edilebilirdi; fakat şimdi bu seçenek de pek tercih edilmemeye başlandı.

Güçlü Kimlik Doğrulama

Güçlü kimlik doğrulama(kartlar,sertifikalar) kullanıcı adı ve parolalardan daha yüksek seviyede güvenlik sağlar.Güçlü kimlik doğrulamanın genel formu “bildiğiniz bir şey,sizde bulunan bir şey” dir.Bu nedenle ;gizli bir bilgi (“bildiğiniz bir şey”) ve bir doğrulayıcı; örneğin kart, bir usb aygıt ya da sertifika (“sizde bulunan bir şey”); bu ikisinin birden kontrolu kullanıcı adı-parola (“sadece bildiğiniz bir şey”) ya da biometrik kontrollerinden daha güvenlidir

Ne Zaman Güçlü Kimlik Doğrulama Kullanılmalıdır :

Güçlü kimlik doğrulamayı muhakkak kullanması gereken uygulamalar :

- Çok fazla değer arzeden işlemler.
- Gizliliğin çok önemli olduğu ya da yasal zorunlulukların bulunduğu uygulamalar (sağlık-hastane kayıtları,hükümet kayıtları,vs)
- Birtakım işlemler sırasında kayıt tutulması zorunlu ve bu kayıtlarla insanlar arasında güçlü bir bağ gerektiği zaman; örneğin bankacılık uygulamarında.
- Çok değerli ya da çok kritik sistemlere yönetimsel erişim yapılacağı zaman.

Risk Ne Anlam İfade Eder?

Her organizasyonun belli bir risk eşiği vardır; bu tüm riskin tümden yoksayılmasından; paranoya seviyelerine kadar çıkabilir.

Örneğin;forumlar güçlü kimlik doğrulamaya ihtiyaç duymazlar;ama günlük milyonlarca doların döndüğü mali bir işlem sırasında güçlü kimlik doğrulama gerekir.

Biometrikler Kendi Başlarında Güçlü Bir Kimlik Doğrulama Değillerdir

Biometrikler “sizde bulunan bir şey” kontrolünü yapabilseler de “sizin bildiğiniz bir şey” kontrolünü içermezler.Biometrik kontrolünü her zaman kullanıcı adı-parola kontrolü ile birlikte yapmalısınız.Aksik takdirde kimlik doğrulama mekanizması önemli ölçüde zayıflar.

Biometrikler uzaktan erişilecek web uygulamaları için diğer güçlü kimlik doğrulama mekanizmaları kadar güçlü değillerdir; çünkü :

Aletler saldırganın kendi kontrolündedir – ve düşük biometrik aygıtlarının güçlü bir tekrar koruması yoktur.

Uzaktan kullanıcı kayıt etmeye güvenilemez – kullanıcılar kendi aralarında değişebilir, lens takabilir; ya da herhangi bir gazeteden buldukları resimle kayıt olmaya kalkabilir.

Sadece iki göz; 10 parmak ve bir yüzümüz var; ve buda çok önemli sistemler için ölümcül bir kombinasyondur.Zira daha once saldırganlar bir arabayı ele geçirmek için gerekirse parmak keseceklerini daha once göstermişlerdir. O açıdan biometrikler çok önemli sistemler için risklidir.

Biometrik özellikler değişmezler – içlerinde kripto motorları bulunan USB anahtarları ve diğer fobların 30 saniyede bir değişen “pseudo-random” çıktıları bulunmaktadır.

Yanlış tespit olasılığı vardır;yani tam doğru olmayan biometrikler doğru Kabul edilip oturum açılabilir; diğer kimlik doğrulama mekanizmalarında böyle bir şey söz konusu değildir.

Biometrik cihazların çoğu kandırılabilir ya da tekrar gönderme saldırılarına açıktır. Fiyatları çok yüksek olan cihazlar; fiyatı makul olanlardan çok daha iyi değillerdir, ama önemli bir biometrik cihazın fiyatıyla 50-60 tane fob ya da 1000'e kadar akıll kart alınabilir.

Eğer tek bir kimlik doğrulama mekanizması kullanılacaksa ; biometrikler bunun en zayıf ve riskleri itibariyle en uygunsuz formudur.

Güçlü kimlik doğrulamanın kullanımı ve kuvveti

Tek Kullanımlık Parolalar

Tek kullanımlık foblar 5-10 $ gibi düşük ücretlere temin edilebilmesine karşın sadece parolanın denenerek bulunmasına karşı koyabilirler.Tek kullanımlık foblar genellikle ekranda görüntülenen bir numaraya sahiptirler ve kullanıcı kullanıcı adını,geçiş kelimesini ve tek kullanımlık parolasını girer.

Tek kullanımlık parolalar araya grime saldırılarına karşı bir koruma sağlamaz ve kullanıcıya kullanımla ilgili hiçbir detay sunmaz, bundan dolayı kötü niyetli – kendisini başka bir siteymiş gibi gösteren- web siteleri tek kullanımlık şifreler toplayabilir ve kullanıcı adıyla oturum açıp işlem gerçekleştirebilir.

Yumuşak sertifikalar

Yumuşak sertifikalar (istemci-tarafındakı sertifika kimlik doğrulaması) parolalardan biraz daha kuvvetli olsa da parolalarla ve kimlik bilgilerini işleyen diğer kimlik doğrulama metodlarıyla aynı problemlere sahiptir.

Bağlı sert sertifikalar

USB,PC Card ya da diğer bağlı tokenler gibi programlı şekilde kullanıcı sistemi tarafından sorguya çekilebilen ekipmanlar kimlik bilgilerini tutmanın en iyi yoludur.Buna karşın aygıt güvenilmeyen bir hosta bağlandığında; sert sertifikalar saldırganın sitesi tarafından doğrudan kullanılabilir ve bu şekilde sağlam kimlik doğrulama mekanızmasının aşılması sağlanmış olur.

Çoğu token kullanıcıdan kimlik bilgilerini tedarik etmek için izin isteyen bir pencere çıkarır; saldırgan da benzer bir pencere hazırlayıp ,kimlik doğrulama panelinin aynısını hazırlar ve onu gerçek sistemin farklı işlemler yaptığı bir anda gönderir.Bu atak iki sebeple çalışır :

- Pop up penceresinin uygulama ve kimlik doğrulama ile temiz/açık bir ilişkisi yoktur; bu problem tüm javascript alarmlarının problemidir ve bu tokenin fonksiyonalitesine özgü bir şey değildir.

- Kullanıcı beyni aşılır – çoğu kullanıcı uygulamanın her gördüğü dialogunu kabul eder(program kurarken eula'yı okumamak gibi); bundan dolayı saldırganın güzel bir kimlik doğrulama penceresi hazırlaması durumunda kullanıcılar gerekli bilgileri oraya girebilir.

Bağlı aygıtlar güvenilen iç erişimler ve güvenilen kullanıcı grupları için uygundur.

Challenge Response

CR tokenleri sistemden bir değer alıp o değeri kriptografik şekilde işleyerek bir sonuca ulaşır.

CR hesaplayıcıları bir tuş takımına sahiptir ve haliyle hesaplayıcıya erişmek için bir PIN girmek gereklidir.Kullanıcılar kullanıcı adları ve response'ları sisteme girerler ve kimlik doğrulama sunucusu tarafından onaylanır.

Tekrar ataklarına karşı korumasına karşın;CR tokenleri kimlik doğrulama – bağlantı kesilme saldırılarından etkilenmektedir.

SMS Challenge Response

SMS CR metin mesajları yollayabilen mobil telefonlarda çokça kullanılmaktadır.Tipik metod kullanıcıyı güvenilir bir şekilde üye ettikten sonra; telefon numarasını kayıt etmektir.Kimlik doğrulama ya da işlem imzalama gerektiğinde uygulama mobil telefona bir işlem numarası ve imzalandığına dair bilgiler yollar(işlemlerdeki referans numarası gibi)

SMS CR'nin problemleri :

- Bu halka açık bir yol; hassas bilgileri challenge ile yollamayın.
- Eğer işlem tutarını yolluyorsa; kullanıcı bu figure guvenebilir fakat saldırgan kullanıcıya bir figür yollayıp başka bir figür onaylayabilir.
- SMS mesajlarının tek kaynağı siz değilsiniz; kullanıcı mesajın sizden gelip gelmediğini doğrulayamaz.

Buna karşın SMS CR kullanıcı adı parola kimlik doğrulama şeklinden daha güvenlidir.

İşlem İmzalama

İşlem imzalama offline CR hesaplayıcıları ile gerçekleştirilir.Kullanıcının hesaplayıcıya girebileceği bir çok farklı şey vardır; ve bu girişe göre bir şey hesaplanacaktır.Bu kullanıcının işlem bilgi detaylarını girmesiyle en güçlü kimlik doğrulama şeklidir diğer işlemler uygun bir

cevap uretilebilmesi için yapılmayacaktır.Bu kimlik doğrulama şekli yadsınamayacak güçlü özelliklere sahiptir,araya sızma saldırılarına ,tekrar deneme saldırılarına ve işlem limitlerinin farklılaştırılmasına karşı sağlamdır.

En iyi şekilde hesaplama yapabilmek için; aşağıdaki bilgiler hesaplamaya katılmalıdır :

- Reference ID
- Referans Numarası
- From account
- Hesap
- Amount of the transaction
- İşlem tutarı

Tokenler genellikle tarih ve zaman tabanlıdır ; bundan dolayı çok kolay şekilde işlem saati kullanılabilir; tokenlerin kötü tarafları ise :

- İşlemin tamamlanması için 20 ile 40 arasında tuş girişi almalıdır; ve bunun her işlem için yapılması kullanıcı için problem yaratır.
- Eğer token kullanıcı bilgisayarına bağlı olması durumuna göre insani faktörleri daha iyidir(detay gerekmez),yadsınamaz özelliği ise kullanıcının tüm işlem için düşünmemesi; sadece imzalama penceresinde onay vermesi.

Bu nedenle; işlem imzalama için çoğu hesaplayıcı istemci bilgisayarına bağlantıyı sağlar; bu özellik kullanılmamalıdır.

Güçlü kimlik doğrulamanın sorunları

Çoğu uygulama iskeletleri J2EE ve .NET tarafından desteklenen sertifika tabanlı oturum açma haricinde güçlü kimlik doğrulama mekanizmaları ile bütünleşememektetir

Kodunuz kimlik doğrulama sunucusu ile; sunucunun sonuçlarına güvenmek için bütünleşik olarak çalışmalıdır.Uygulamanızı seçtiğiniz mekanizma ile nasıl bütünleştireceğinizi iyi planlamalı ve enjeksiyon,tekrar ve kurcalama saldırılarına karşı sağlam olmasına dikkat etmelisiniz.

Çoğu organizasyon güçlü kimlik doğrulama seçeneklerini çok pahalı görmektedir.Kullanıcı yönetimi maliyeti genel olarak kimlik doğrulama alt yapısına bağlı değil; kullanıcı kayıtlarının nasıl tutulduğuyla alakalıdır.Eğer güçlü bir system istiyorsanız ; kullanıcı yönetiminin en korkunç ve pahalı tarafı kayıt; bakım ve üyelikten çıkarma olacaktır.Basitçe hesap açtırmak isteyen herkese bir token ya da sertifika yollayın.Güvenli ve sağlam bir kayıt yolu; kimlik doğrulama sisteminin gücünden geçer

Birleşik Kimlik Doğrulama

Birleşik kimlik doğrulama size kullanıcı veri tabanınızın üçüncü parti unsurlara devredebilmeniziya da bir imza ile bir çok sitenin çalıştırılabilmesi kolaylığını sunar.Birleşik güvenliğin kullanılmasının birincil nedeni kullanıcıların sadece bir kez imzalaması ve aynı kimlik doğrulama mekanizmasını kullanan diğer sitelerin imzalanmış tokene güvenip kullanıcıya güvenmesi ve kişisel hizmetleri vermesidir.

Birleşik kimlik doğrulamanın avantajları :

- Kullanıcıların hatırlaması gereken toplam bilgi sayısını azaltmak.
- Sitenizin geniş bir partnerlik anlaşmasının parçası olması; örneğin bir satın alma ağının.
- Diğer anonim kullanıcıları kişiselleştirilmiş servisler sağlamak isteyebilirsiniz.

Şunlar olmaksızın birleşik kimlik doğrulamayı kullanamazsınız :

- Kimlik doğrulama sağlayıcısına güvenmelisiniz.
- Gizlilik gereksinimleriniz kimlik doğrulama sağlayıcısı tarafından karşılanmalıdır.

Kimlik (Bilgisi) Kanunları (Kuralları)

Kim Cameron,Microsoft Kimlik Mimari birleşik kimlik çevresinde dolanan riskler üzerine çalışan bir blog grup kurdu.Grup yedi kimlik kanuna ilişkin bazı yazılar çıkarttı.

1. **Kullanıcı kontrolü ve izni :** Dijital kimlik sistemleri sadece kullanıcı izni ve bir tanımlama bilgisini göstermelidir.
2. **Sınırlı kullanım için sınırlı ifşa** : En az kimlik bilgisini ifşa edip en iyi limitlerle çalışan çözüm uzun vadelidir.

3. **En Az Topluluk Kanunu:** Dijital kimlik tanımlama sistemleri kimlik tanımlama bilgilerinin açıklanmasını limitlemelidir.

4. **Direk Kimlik Tanımlama** :Genel bir kimlik tanımlama sistemi hem "çokyönlü" kimlik tanımlayıcıları halka açık kayıtlar için ve "tekyönlü" kimlik doğrulayıcıları özel kayılar için kullanmalıdır.

5. **Operatör ve Teknolojilerin Çoğulculupu** : Genel bir kimlik tanımlama sistemi birden çok teknoloji ve kimlik tanımlama sisteminin beraber çalışmasını desteklemelidir.

6. **İnsan Entegrasyonu** : Bir kimlik tanımlama sistemi insanı insan-makine iletişimlerinde şühesiz ve korunaklı bir bileşen olarak tanımlamalıdır.

Bu kanunlar yazıldığı zaman kimlik tanımlama çercevesindeki sorunları bitirip bitirmeyeceği bilinmiyordu; ama bu kanunlarda geçen konulardan çoğu birleşik kimlik doğrulama mekanizması gerçekleyicileri tarafından dikkate alınmalıdır.

Microsoft Passport

Microsoft Passport birleşik kimlik doğrulamanın bir örneğidir;Hotmail ,anlık yazışmalar,yazılım ulaştırma için kullanılır ve bir süre eBay gibi iş ortakları tarafından da kullanılmıştır.

İstemci Taraflı kimlik doğrulama kontrolleri

İstemci-taraflı doğrulama(genellikle JavaScript ile yazılan) kullanıcılara,kural dışı bir şey yaptıklarında, hızlı bir şekilde geri bildirim yapmak için kullanılan ve web sunucuya az yük bindiren güzel bir kontroldür.Ama istemci taraflı doğrulamalar çok kolay şekilde aşılabilir.

Saldırıya açık olup olmadığınızı anlama (yolları)

Testler için; oturum açma sayfasını durgun bir html sayfasına indirgeyin ve bir POST metoduyla sunucuya yollanmasını sağlayın.

Şimdi istemci-taraflı kontrolleri aşabilirsiniz;ayrıca bu forma otomatik saldırı araçları kullanarak saldırılması daha kolaydır.

Kendinizi koruma (yolları)

Uygulamanızı korumak için; tüm doğrulama ve hesap politikaları/kurallara uygunluk sınamalarının sunucu tarafında yapıldığına emin olun.

Mesela;boş parolalara izin vermeyecekseniz bunun testini sunucu tarafında yapmalısınız; opsiyonel olarak da istemci-tarafında yapabilirsiniz.Aynı şey "şifre değiştir" için de geçerlidir.

Daha fazla bilgi için; kitabin "Onaylama" kısmını okuyunuz.

Positif Kimlik Doğrulama

Maalesef; kimlik doğrulama için genel-güzel bir tasarım her durum için elverişli olmaz.Bununla birlikte bazı tasarımlar diğerlerinden daha iyidir.Eğer bir uygulama aşağıdaki sözde kodla kimlik doğrulama yapıyorsa;herhangi bir terslik durumunda kimlik doğrulama yapabilir.

```
bAuthenticated := true
try {
userrecord := fetch_record(username)
if userrecord[username].password != sPassword then
bAuthenticated := false
end if
if userrecord[username].locked == true then
      bAuthenticated := false
end if
      …
}
catch {
      // perform exception handling, but continue
}
```

Saldırıya açık olup olmadıgınızı anlama (yolları)

Kimlik doğrulama mekanizmasının başarısız olması için uğraşın.

Eğer olumlu kimlik doğrulama algoritması söz konusuysa;herhangi bir hata uygulamanın diğer kısımlarına erişim verecektir.Özellikle çerezleri,başlık bilgilerini, formları ya da gizli form alanlarını test edin.Tiple,uzunlukla,yazım tarzıyla oynayın.Boşluk enjeksiyonu,CRLF atakları,XSS ve SQL Enjeksiyonu denemeleri yapın.İki tarayıcıyla JavaScript hata ayıklayıcısı kullanarak yarış durumlarının sömürülüp sömürülemeyeceğini araştırın.

Kendinizi koruma (yolları)

Herhangi bir olağandışı durumda kimlik doğrulama işlemini başarısız olarak sona erdirin.

```
bAuthenticated := false
securityRole := null
try {
userrecord := fetch_record(username)
if userrecord[username].password != sPassword then
      throw noAuthentication
end if
if userrecord[username].locked == true then
      throw noAuthentication
end if
if userrecord[username].securityRole == null or banned then
      throw noAuthentication
end if

      … other checks …
bAuthenticated := true
securityRole := userrecord[username].securityRole
}
catch {
bAuthenticated := false
securityRole := null

// perform error handling, and stop
}
return bAuthenticated
```

Hata durumlarında yetki vermeyin; ve “try” blokunun sonunda yetkiyi verin.

Birden Fazla Anahtar ile Bakmak

Kullanıcı kayıtlarına birden fazla anahtar ile bakılması SQL ya da LDAP Enjeksiyon problemi doğmasına sebep olabilir.Örneğin hem kullanıcı adı hem de parola kısımları kullanıcı kayıtlarını bulmak üzere kullanılır ve SQL & LDAP Enjeksiyon kontrolleri yapılmazsa;iki alan da suistimal edilebilir.

Örneğin; kullanıcıyı parolasından bulmak isteyin ve kullanıcı adı kısmına dikkat etmeyin.Çoğu SQL kayıtlara bakma sorguları "select * from table where username = username and password = password" şeklinde yazılır ki bu bilgi saldırgana parolasız oturum açma imkanı verebilir(örneğin "select * from username='username'; -- and password = 'don't care'"").Eğer kullanıcı adları tekilse;onlar anahtar olmalıdır.

Saldırıya açık olup olmadığınızı anlama (yolları)

Programınız :

- Arama sorgusunda sadece kullanıcı adı kullanılıyorsa
- Arama sorgusunda kullanılan alanlar özel karakterlere karşı korunmamış,SQL ve LDAP Enjeksiyonu saldırılarına karşı korunaksız ise.

Bunu şu şekilde test edebilirsiniz.

- Oturum açma sayfasına bir SQL(ya da LDAP) Enjeksiyonu saldırısı yapın;örneğin bir alanı "true" yapın.

```
KullanıcıAdı: a' or '1'='1
Parola: parola
KullanıcıAdı: a)(|(objectclass=*)
Parola: parola
```

Eğer üstteki madde çalışırsa; parola olarak "parola" kullanan ilk hesap bilgisiyle oturum açacaksınız; ya da bir hata oluşacak.Ne sıklıkla çalıştığını görünce şaşıracaksınız.

Kendinizi koruma (yolları)

SQL ve LDAP'in meta karakterlerinden kaçınmak için girdileri test edin. Sorgularda anahtar olarak sadece kullanıcı adların kullanın.

Kaç kaydın geri döndüğünü kontrol edin.

Java

```
public static bool isUsernameValid(string username) {
RegEx r = new Regex("^[A-Za-z0-9]{16}$");
return r.isMatch(username);
```

```
}


// java.sql.Connection conn is set elsewhere for brevity.

PreparedStatement ps = null;
RecordSet rs = null;

try {
isSafe(pUsername);
ps = conn.prepareStatement(“SELECT * FROM user_table WHERE username = ‘?’”);
ps.setString(1, pUsername);
rs = ps.execute();
if ( rs.next() ) {
// do the work of making the user record active in some way
}
}
catch (…) {
…
}
```

.NET (C#)

```
public static bool isUsernameValid(string username) {
RegEx r = new Regex(“^[A-Za-z0-9]{16}$”);
Return r.isMatch(username);
}


…
try {
string selectString = " SELECT * FROM user_table WHERE username = @userID”;
// SqlConnection conn is set and opened elsewhere for brevity.
```

```
SqlCommand cmd = new SqlCommand(selectString, conn);
if ( isUsernameValid(pUsername) ) {
cmd.Parameters.Add("@userID", SqlDbType.VarChar, 16).Value = pUsername;

SqlDataReader myReader = cmd.ExecuteReader();
If ( myReader.
// do the work of making the user record active in some way.
myReader.Close();
}
catch (…) {
…
}
```

PHP

```
if ( $_SERVER[‘HTTP_REFERER’] != ‘http://www.example.com/index.php’ ) {
    throw …
}
```

Yönlendirici Kontrolü

Yönlendirici HTTP paketinin başlık alanındaki opsiyonel kısımlardan olup bir önceki konumu içinde barındırır.Saldırgan bu bilgiyi rahatlıkla değiştirilebilir;yönlendirici bilgisi dikkatli bir şekilde ele alınmalıdır; zira saldırganlar genellikle doğru yönlendirici bilgileriyle uygulamanızdaki kontrolleri aşmaya çalışacaktır.

Genel olarak uygulamalarda yönlendirici kodu kullanmamak daha iyidir.

Saldırıya açık olup olmadıgınızı anlama (yolları)

Zayıflık bir kaç farklı yerden gelebilir:

- Kodunuz yönlendiriciyi kontrol ediyor mu; ediyorsa etmesi kesinlikle gerekli mi?
- Yönlendirici kodu basit ve her türlü atağa karşı korunumlu mu?
- URL’leri oluştururken onlardan faydalanıyor musunuz?Tüm geçerli URL’leri test
- etmenizi imkansızlaştırlarır.

Örneğin login.jsp’ye sadece http://www.example.com/index.jsp ‘den talep gidebilir; bu durumda yönlendirici bilgisi kontrol edilmelidir.

Kendinizi koruma (yolları)

Çoğunlukla yönlendirici bilgisini kullanmak arzu edilir bir şey değildir; çünkü saldırganlar tarafından kolaylıkla modifiye edilebilir;hiçbir güven ilişkisi yönlendirici bilgisi üzerinden yapılmamalıdır.

Web kayıt analizcileri gibi yönlendirici bilgisini gösteren araçlar yazılırken XSS ve diğer HTML Enjeksiyonu ataklarına karşı dikkatli olunmalıdır.

Eğer uygulamanız yönlendirici bilgisini kesin olarak kullanması gerekiyorsa; sıkı bir savunma mekanizmasıyla incelenmeli ve doğru değilse bilgi kullanılmamalıdır.Her kod hatalar barındırır ve kodu azaltmak için yönlendirici alanından kısıtlamaya gidilmeye çalışılmalıdır.

Eğer login.jsp sadece http://www.example.com/index.jsp tarafından uyarılabilecekse; şu şekilde bir kontrol yapılmaıdır :

Java

```
HttpServletRequest request = getRequest();
if ( !
request.getHeader("REFERER").equals("http://www.example.com/index.jsp")
) {
      throw …
}
```

.NET (C#)

```
if ( Request.ServerVariables("HTTP_REFERER") !=
'http://www.example.com/default.aspx' ) {
      throw …
}
```

PHP

```
if ( $_SERVER['HTTP_REFERER'] != 'http://www.example.com/index.php' ) {
      throw …
}
```

Fakat basitçe herhangi bir oturum değişkeni ile karşılaştırılınca yönlendirici bilgisi daha zayıf bir yetkilendirme kontrolüdür.

Tarayıcılar parolaları hatırlar

Modern tarayıcılar kullanıcılarına; oturum bilgilerini güvensiz bir şekilde bilgisayar üzerinde saklamayı teklif ederler.

Saldırıya açık olup olmadıgınızı anlama (yolları)

- Tüm durum bilginizi tarayıcınızdan temizleyin.Genellikle bunu yapmak için en güvenilir yol; temiz bir test hesabı açıp test bilgisayarında onu kullanmaktır.Testler arasında kullanıcı hesaplarını silip tekrar oluşturarak testlerinize devam edebilirsiniz.
- Bir tarayıcı kullanarak uygulama üzerinde oturum açın.
- Eğer tarayıcınız hesap bilgilerini saklamayı teklif ederse; uygulamanız risk altında demektir.
- Risk uygulamanın finansal ya da hassas bilgi içermesine bağlı olarak değişir.

Kendinizi koruma (yolları)

Modern tarayıcılar kullanıcılarına; oturum bilgilerini güvensiz bir şekilde bilgisayar üzerinde saklamayı teklif ederler.

HTTP içinde;kullanıcı adları,parolalar, parola tekrar doğrulamaları,kredi kartı numaraları ve CCV alanları gibi hassas bilgileri aşağıdaki gibi yollayın :

```
<form … AUTOCOMPLETE="off"> - for all form fields
<input … AUTOCOMPLETE="off"> - for just one field
```

Bu tarayıcılara belirtilen alanları “parola hatırlama” özelliği çerçevesinde saklamaması gerekliliğini hatırlatır.Ama unutmayın ki bu hatırlatma tavsiye düzeyindedir ve her tarayıcı bu etiketi desteklemeyebilir.

Varsayılan Hesaplar

Varsayılan hesaplardaki genel bir zayıflıktır –iyi/çok bilinen kullanıcı adları ve parolalarla hesapların olması.Özellikle kötü örnekler :

- SQL 2000 SP 3’e kadar Microsoft SQL Sunucusunun kullanıcı adı “sa” idi.
- Oracle – çok sayıda parolalarıyla birlikte bilinen hesap (sonraki sürümlerinde

- düzeltildi)

Saldırıya açık olup olmadıgınızı anlama (yolları)

- Sözü edilen alt yapıdaki varsayılan hesapların var olup olmadığını tespit edin(örneğin ; Administrator,root,sa,ora,dbsnmp,etc)
- Kodun herhangi bir varsayılan,özel,hata ayıklama amacıyla ya da arka kapı bilgisi barındırıp barınmadığını belirleyin.
- Kurucunun herhangi bir varsayılan,özel,hata ayıklama amacıyla ya da arka kapı bilgisi barındırıp barınmadığını belirleyin.
- Her hesabın; özellikle yönetimsel hesapların tamamen kullanıcılara özgü olduğundan emin olun.

Herhangi bir örnek ya da kullanıcı adıyla beraber resim dökümantasyonda var olmamalıdır

Kendinizi koruma (yolları)

- Yeni uygulamalarda kesinlikle varsayılan hesap olmamalıdır.
- Dökümantasyonda hiçbir varsayılan hesap bırakılmaması belirtilmelidir.
- Kodun herhangi bir varsayılan,özel,hata ayıklama amacıyla ya da arka kapı bilgisi bulundurmasına izin vermeyin.
- Program yükleyiciyi oluştururken; yükleyicinin kesinlikle varsayılan,özel bir kimlik bilgisi oluştırmamasından emin olun.
- Tüm hesapların; özellikle idari hesapların kullanıcı/kurucu tarafından oluşturulduğundan emin olun.
- Herhangi bir örnek ya da kullanıcı adıyla beraber resim dökümantasyonda var olmamalıdır.

Kullanıcı adı seçimi

Eğer tahmin edilebilir bir kullanıcı adı seçerseniz;saldırganlar size karşı servis dışı bırakma saldırısı yapabilirler.Örneğin; özellikle bankalar monoton artan müşteri numaraları ya da kredi kartı kullanıyorlarsa büyük risk altındalar.

Saldırıya açık olup olmadıgınızı anlama (yolları)

- Kötü kullanıcı adları :
- İsim.soyisim

- E-Posta adresleri(kullanıcılar yeterince rastgele olursa bu bir problem olmayabilir; ya da siz mail sağlayıcısıysanız)
- Herhangi bir monoton artan numara
- Yarı-gizli bilgiler;örneğin vatandaşlık numarası,vs.

Gerçekte vatandaşlık bilgisini kullanıcı adı olarak kullanmak yasadışıdır; zira onları uygun bir gerekçe göstermeksizin toplayamazsınız

Kendinizi Koruma (Yolları)

Mümkünse; kullanıcılara kendi kullanıcı adlarıyla hesap oluşturmalarını ve kullanıcı adlarının tekil olmasına özen gösterin.

Kullanıcı adları HTML,SQL VE LDAP saldırılarına karşı güvenli olmalı; sadece A..Z,a..z ve 0-9 karakter setlerini kullanmayı önerin.Boşluklara izin verecekseniz; @ sembolu ya da tırnak işaretleri gibi özel karakterlerden korunmalısınız.(detaylar için Veri Doğrulama bölümüne bakınız).

Ad.soyad, e-posta adresi,kredi kartı numarası,müşteri numarası gibi yarı-açık bilgilerin kullanımından kaçınınız.

Şifre Değiştirmek

Kullanıcının kimlik bilgisinin bir parçasını hatırlaması gerektiği yerlerde; bazen o bilgiyi değiştirir.Örneğin parolalar kazara 3.şahıslar tarafından öğrenilirse kullanıcı parolayı değiştirmek isteyecektir.

Saldırıya açık olup olmadığınızı anlama (yolları) :

Test etmek için :

- Parolanızı değiştirin .
- Tekrar parolanızı değiştirin- değiştirilen parolanın tekrar değiştirilmesi için geçmesi gereken zamana göre işlemi yapın.(örneğin 1 gün)

Kendinizi koruma(yolları)

- Uygulamanızın parola değiştirme fonksiyonu olduğuna emin olun.
- Form eski şifreyi; yeni şifreyi ve yeni şifre doğrulamasını istemelidir.
- AUTOCOMPLETE=OFF etiketini tarayıcıların parolaları saklamalarını engellemek üzere kullanın .

- Eğer kullanıcı eski parola çokça sefer yanlış girerse, hesabı kitleyip oturumu kapatın.

Yüksek riskli uygulamalar için; parolaların sıklıkla değiştirilmesini engellemelisiniz; eski parolaların da kriptografik özetlerini 24 özete kadar saklamalısınız.

Kısa Parolalar

Parolalar kaba kuvvetle; daha önceden hesaplanmış özet parolalar ile ya da basit sözlük tabanlı ataklarla kırılabilir.Maalesef parolalar tüm risk profillerindeki ilk oturum açma metodudur.Kısa parolaların kırılma oranı yüksektir.

Saldırıya açık olup olmadığınızı anlama (yolları) :

- Uygulamanızın herhangi bir yerinde parolasız kullanıcı kabul edip etmediğini belirleyin; ve kesin olarak buna izin vermeyin.
- Uygulamanızın kullanıcıların tehlikeli-kısa(4 karakterden kısa) parolalar kullanmasına izin verip vermediğini belirleyin.Güçlü kimlik doğrulama gerektiren uygulamalar bu şekilde parola kullanılmasına engel olur;orta düzeyli uygulamalar parolanın zayıf olduğunu belirten bir tehlike mesajı yayınlar; fakat zayıf uygulamalar ise herhangi bir tepkide bulunmazlar.
- Uygulama maksimum parola uzunluğuna eriştiğiniz ikazını yapana kadar parolanın uzunluğunu arttıracak biçimde değiştiriniz.Güzel bir uygulama rastgele seçilmiş parola uzunluklarına izin verir.

Her denemede daha kısa bir parolanın çalışıp çalışmadığına dikkat ediniz.(sıklıkla sadece 8 veya 16 karakter)

Kendinizi Koruma (Yolları)

- Uygulamanızın boşluk parolalara izin vermediğine kesin olarak emin olun.
- Bir minimum parola uzunluğu yaptırımı uygulayın.Önemli uygulamalarda;kullanıcının kısa parolalar seçmesini engelleyin.Çok kritik olmayan uygulamalarda ise altı karakterden kısa parolaların zayıf olduğuna dair bir uyarı mesajı gösterin.
- Kullanıcılara “My milk shake brings all the boys to the yard” ya da “Let me not to the marriage of true minds Admit impediments” gibi uzun parolalar kullanmalarını tavsiye edin.

- Tek yönlü kriptografik özet fonksiyonları kullanan uygulamanızın uzun parolaları kabul ettiğine emin olun.

Zayıf Parola Kontrolleri

ISO 17799 ve bir çok güvenlik politikaları kullanıcıların mantıklı parolalar seçip kullanmasını ve onları belli bir sıklıkla değiştirmesini tavsiye eder.Çoğu web uygulaması bu güvenlik politikalarına pek uymaz.Eğer uygulamanız ISO 17799 ya da benzer standartlara ihtiyaç duyuyorsa; basit kimlik doğrulama kontrolleri ile gerçekleştirilmelidir.()()

Saldırıya açık olup olmadığınızı anlama (yolları) :

Uygulamanızın :

- Boşluk parolaları kabul ediyorsa
- Sözlükte bulunan kelimeleri parola olarak kabul ediyorsa; sözlük local olmalı ve sadece ingilizce olmamalı.
- Geçmişte kullanılmış parolalara izin veriyorsa.Güçlü kimlik doğrulama gerektiren uygulamalar eski parolaların kriptografik özetini ;parolaların tekrar kullanımını engellemek amaçlı olarak saklamalı.

Kendinizi Koruma (Yolları)

- İngilizceden başka dillere izin vermeli(Belçika ve İsviçre gibi birden çok dil kullanılan bölgeler için birden fazla dil seçeneği olmalı)
- Uygulamanız aşağıdaki kontrolleri yapmalı(ama opsiyonel olarak zorlamalı)
- Minimum parola uzunluğu (ama asla maximum parola uzunluğu kısıtlanmamalı)
- Parola Değiştirme Sıklığı
- Password minimum password age (to prevent users cycling through the password history)
- Parola karmaşıklık gereklilikleri
- Parola geçmişi
- Parola kitleme süreleri ve politikaları(hiç kitleme yapmamak,X dakika için kitlemek,süresiz kitlemek)

Çok riskli uygulamalar için;kullanıcıların girdiği parolaların çok zayıf olup olmadığını kontrol için basit bir sözlük

Not: Karmaşık sıklıkla parola değişimleri güvenlik açısından beklenmedik etkiler gösterebilir.10 karakterden fazla uzunlukta ve her 30 günde bir değişen parolalar kullanmak daha

iyi olacaktır.30 günde bir olması tüm iş yerinin üzerine parolaların yazıldığı PostIt™ notlarıyla dolmasını sağlar.

Tersinir Parola Şifreleme

Parolalar gizlidir.Hiçbir şart altında onları şifresiz hallerine döndürmeye lüzüm yoktur. Yardım masası ekipleri yeni parolalar belirleyebilirler,fakat eskileri okuyamamalıdır. Bundan dolayı parolaları tersine çevrilebilir bir şifreleme ile tutmak için hiçbir sebep yoktur.

Sık görülen mekanizma MD5 ya da SHA-1 gibi kriptografik özet fonksiyonlari kullanmaktır.Ama bazı kriptografik özet fonksiyon türlerinin zayıf oldukları gösterildi; eğer kriptografik özet fonksiyonu kolleksiyonunuz çok fazla değilse; daha güçlü algoritmalara yönelmek şarttır..

Saldırıya açık olup olmadığınızı anlama (yolları) :

Form tabanlı bir kimlik doğrulama mekanizması kullanıyorsanız; kullandığınız algoritmayı test edin.Algoritmanız AES-128'i özet modunda ya da SHA-1'i 256 bit modda kullanmalıdır.

- MD5 ve SHA1 (160 bit outputluk) algoritmaların zayıf olduğu gösterildi; daha fazla kullanılmamalı.
- Hiçbir algoritma kullanmamak güvensizlik yaratır; ve mutlaka bir algoritma kullanılmalıdır.
- DES, 3DES, Blowfish ya da AES chiper modda çalışan algoritmalar parolaların geri dönüştürülebilmelerine olanak verebilmektedir.

Kendinizi Koruma (Yolları)

Parola şifrelemenin arkasındaki kriptografiyi anlamıyorsanız; yüksek olasılıkla bir şeyler yanlış gidiyor demektir.Lütfen güvenli parola implementasyonları tekrar kullanmayı deneyin.

- AES-128'i özet modunda; ya da SHA – 1'i 256 modda kullanın.
- Statik olmayan bir karıştırma algoritması kullanın.
- Parola özetini ya da parolayı kullanıcıya hiçbir şekilde geri döndürmeyin.

Otomatik Parola Yenilemeleri

Otomatik parola yenileme mekanizmaları; kimlik doğrulamaya harcanan yüksek tutardaki yardım masası ücretlerini düşürmeye çalışan organizasyonların genel olarak kullandığı bir

mekanizmadır.Risk yönetimi perspektifinden bakarsak, parola yenileme fonksiyonelitesi çoğu durumda kabul edilebilir bir özelliktir.Ama aynı zamanda; parola yenileme özelliği ikincil ve daha zayıf bir parola kontrolüne eşittir.İlgili öne çıkan bir çalışmaya göre(referanslara bakın),beş cevaplı bir parola yenileme sistemi iki karakterli bir parola kadar güvenli ve tersinir şifreleme gerektirdigi ya da arka planda çalışan sistemde şifresiz olarak parolaların tutulmasını gerektirdiği için en iyi güvenlik uygulamaları ve çoğu güvenlik politikaları ile uyuşmazlar.

Genel olarak; parolayı yeniden belirlemek için sorulan sorulan genel olarak halka açık kayıtlarda olan bilgilerdir.(annenin kızlık soyadı;arabanın rengi,vs) Çoğu durumda; parola yeniden belirleme toplanması problem yaratacak bilgiler sormaktadır.Çoğu gizlilik yönetimlerinde direk olarak uygulamanıza gereken bilgiyi toplamanız ve niye bu bilgiyi topladığınızı açıklamanız gerekmektedir.

Genellikle;kimlik doğrulama mekanızmanız tarafından kullanılan bilgiler pratik olarak kötüyse,parola yeniden belirleme mekanızmalarını kullanmayın.

Saldırıya açık olup olmadığınızı anlama (yolları) :

Parola yeniden belirleme mekanizmaları çok çeşitlilik gösterirler; ama sıklıkla suistimal edilirler.

- Parola yeniden belirleme mekanizmaları ipuçlarını kullanırlar; doğum tarihi,kimlik numarası, annenin adı gibi ipuçlarını inceleyin. Olmazsa bu bilgiler harici kaynaklar ya da sosyal mühendislik metodlarıyla ele geçirebilirler.
- Parola yeniden belirlemeleri hesabın kilidini açmak için e-posta adresini kullanıyorsa; sonuç üreten e-posta parola içermemeli, ama bir seferlik doğrulama tokeni kısa bir süre için geçerli olmalıdır.Eğer tokeniniz uzun bir periyod için iyiyse; tokenin tahmin edilebilir ya da kolay şekilde oluşturulabilir olup olmadığını kontrol ediniz.
- Eğer e-posta'da tıklanabilir bir link varsa; linkin olta saldırısı için kullanılabilir.

Kendinizi Koruma (Yolları)

- Çok değerli işlemler yapan sistemler parola yeniden belirleme kullanmamalıdır.
- Daha ucuz ve güvenli sistemler düşünün.
- Eğer sorular ve cevaplar kullanıcıyı yardım amsasına tanıtmak için kullanılıyorsa; basitçe "Yardım masasını nasıl çağırırım" sayfasında bir rastgele sayı üretin; ve kullanıcı arayınca sayıyı doğrulayın.
- Otomatik parola yeniden belirleme sistemlerini gerçekleştirirken dikkatli olun.En kolay yolu; "kullanıcıya e-posta yolla" seçeneği olup sadece bir gizli bilgi içermektedir;

kullanıcının e-posta adresi.Ama kullanıcının e-posta hesabı ele geçirilmişse bu yol da riskli olacaktır.

- Kullanıcıya birisinin parola yeniden belirleme fonksiyonunu harekete geçirdiğini açıklayan bir mesaj gönderin.Eğer kendisi yeniden belirleme isteğini yapmadıysa; rapor etmesini isteyin.Eğer kendisi isteği yaptıysa; kısa kriptografik bir zaman limitiyle bir token yollayın.Bunu phishing saldırılarına karşı bağ(link) olarak yapmayın.Bu değer token için bekleyen uygulamaya girilmelidir.Tokenin süresinin dolup dolmadığını ve bu hesap için geçerli olup olmadığını kontrol edin.Burada kullanıcıya parolasını değiştirip değiştirmeyeceğini sorun.Eğer başarıyla işlem gerçekleşirse kullanıcıya ve yöneticiye e-posta atılmasını sağlayın; her şeyi kayıt edin.

Eğer ip ucu tabanlı bir alternative seçecekseniz;özel bilgi tavsiyeleri kullanın.”Favori renginiz nadir”,”En güzel anınız” ,vs.Anneninizin adı,vatandaşlık numarası ya da onun gibi şeyler kullanmayın.Kullanıcı kayıt sırasında beş ipucu girmeli ve parolayı yeniden girerken üç tanesi ile temsil edilmeli.

Bütünlük ve gizliliği sağlamak için parola yeniden belirleme işlemleri sırasında SSL kullanın.

Kaba Kuvvet

Bilinen yetkili bir hesap ismi ya da tahmin edilen hesabın şifresine karşı kaba-kuvvet ya da sözlük atakları yapmak genel bir atak çeşididir.

Uygulamalar kararlı kaba kuvvet ve sözlük saldırılarına,Brutus’ten ya da başka betiklerden, karşı tutarlı olmalıdır.Kararlı kaba kuvvet saldırıları kolay kolay kesilememekte; sadece geciktirilebilmekte-bekletilebilmektedir.

Saldırıya açık olup olmadığınızı anlama (yolları) :

Uygulamanızı test etmek için :

- Bir kaba kuvvet uygulaması kullanın; Brutus gibi ya da özel olarak yazılmış bir perl betiği olabilir.Bu atağı ancak araçlarla deneyebilirsiniz.
- Sadece İngilizce değil; birden çok sözlük kullanın.
- “Genel parolalar” sözlüğünü kullanın. Hangi sıklıkla “root”, “password”, “” gibi parolaların kullanıldığını görünce şaşıracaksınız.
- Hata mesajı kimlik doğrulama sürecindeki sorunla ilgili bilgi veriyor mu?
- Başarısız kimlik doğrulama girişimleri kayıtları bir kaba kuvvet saldırısına mı işaret ediyor?Uygulamanız IP’nizi blokluyor ya da oturumunuzu kapatıyor mu?

- Kaba kuvve saldırısına n-1 kere denedikten sonra oturumu kapatıp devam edebiliyor musunuz?Örneğin 5 denemede oturum kapanıyorsa; 4 tanesini kullanıp yeni bir oturum açabiliyor musunuz?

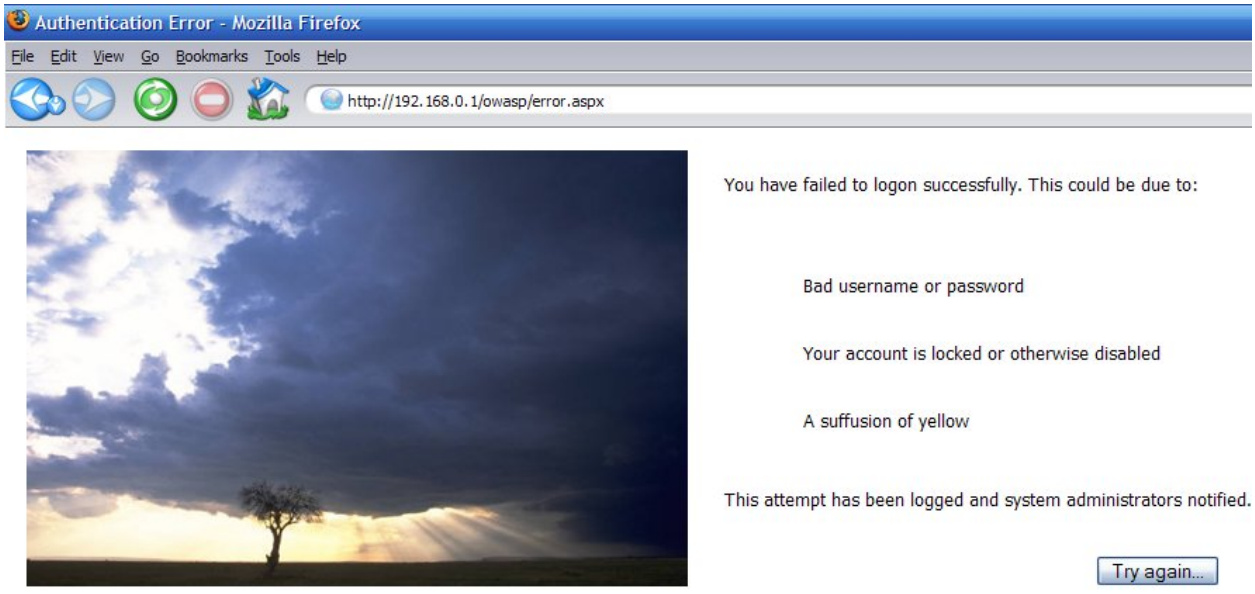
Eğer uygulamanız bir IP ya da IP grubundan beş denemeye izin veriyorsa; ya da saniyede 10'dan fazla deneme yapma imkanı sağlıyorsa bir kaba kuvvet atağına mahal vermektedir.

Kendinizi Koruma (Yolları)

Uygulamayı kontrol edin :

- Kullanıcının hatalı ya da doğru kimlik bilgisi denemeleri arasındaki gecikmeye dikkat edin.Üç saniyelik bir aralık otomatik kaba-kuvvet saldırılarını imkansız hale getirmektedir.İlerleyen(3 saniye sonra 15 sonra 30 sonra ve kapat) aralıklar kaba-kuvvet akalarını tamamen etkisiz hale getirecektir.

- Tüm kimlik doğrulama hataları için ortak bir hata sayfası hazırlayıp; hangi kimlik doğrulama bilgisinde sorun olduğu ile ilgili bir bilgi vermeden kimlik bilgilerinde sorun olduğunu ifade eden ortak sayfalar kullanın.



- Başarısız kimlik doğrulama denemelerini kayıt edin (aslında güzel bir web uygulaması tüm kimlik doğrulama denemelerini kayıt eder)

- daha güçlü kimlik doğrulama kontrolleri gerektiren uygulamalar için; kötü IP adreslerini bloklanabilir (örneğin üçten fazla hesaba aynı IP'den erişen ya da birden fazla hesabı aynı anda kapatmaya çalışan IP'leri)

- Oturumu çok fazla başarısız denemeden sonra siler.

Örnek bir senaryoda; log incelemeleri aynı sayfaya kısa bir zaman dilimi içinde aynı IP'den erişim olduğunu ortaya çıkarabilir.Simple Event Correlator gibi yazılımlara anımlanan kurallar doğrultusunda belirli olaylara rastlanıldığında alarm ya da uyarı verdirilebilir.Bu ayrıca SNORT'a eklenecek bir kural ile kullanıcıya giden HTTP Kimlik Doğrulama Başarısız mesajlarının uyarı üretmesi sağlanabilir.

Beni Hatırla

Halka açık/paylaşılan bilgisayarlarda "beni hatırla" özelliği,kişisel hesabına kimlik doğrulama olmaksızın geri dönmek,tehlikeli olabilmektedir. Örneğin internet kafelerde, önceki kullanıcıların çoktan oturum açtıkları hesaplarıyla onlar gibi mesaj yollayabilir ya da sipariş verebilirsiniz(örneğin eBay ile).

Saldırıya açık olup olmadığınızı anlama (yolları) :

- Uygulamanın "beni hatırla" özelliği var mı?
- Eğer varsa; ne kadar süre içinde hatırlayabiliyor?Sürekli hatırlıyorsa;çerez bilgileri ne kadar sürede ölüyor?
- Çerez bilgileri tahmin edilebilir mi?Eğer öyleyse bu bilgi tüm kimlik doğrulama mekanizmasını etkisiz kılmakta kullanılabilir mi?

Kendinizi Koruma (Yolları)

- Eğer uygulamanız yüksek değerdeki işlemlerle ilgileniyorsa; bu özelliğe sahip olmamalı.
- Eğer risk en az seviyede ise; kullanıcılarınız bu özelliğin tehlikeli olabileceğine dair uyarmalısınız.
- Tahmin edilebilir daha "önce kimlik doğrulamamış" bir token kullanmamalısınız.

İşe Yaramaz Zaman Aşımları

Özel bilgileri taşıracak ya da kimlik bilgileri çalınmasında kullanılacak uygulamaların belli bir zaman diliminden sonra erişimi kapatılmalıdır

Saldırıya açık olup olmadığınızı anlama (yolları) :

- Uygulamaya oturum açın.
- Uygulama "canlı kal" ya da "beni otomatik olarak tanı" gibi bir özelliğe sahip mi?Eğer öyleyse uygulamanın testte başarısız olma ihtimali yüksek.
- 20 dakika bekleyin.
- Uygulamayı tekrar kullanmayı deneyin.
- Eğer uygulama kullanıma açıksa; uygulama risk altında demektir.

Kendinizi Koruma (Yolları)

- Kullanıma uygun bir zaman periyodu belirleyin.
- Oturum yöneticinizin zaman aşımı süresini belirlenen zaman süresinden sonra oturumu kapatacak üzere ayarlayın

Oturumu Kapat

Her uygulamanın bir oturumu kapatma metodu vardır.Bu özellikle kimlik bilgileri çalınabilecek uygulamalar için hayati derece önemlidir.

Saldırıya açık olup olmadığınızı anlama (yolları) :

- Uygulama bir "oturumu kapat" butonu ya da linki içeriyor mu?
- Uygulanın her görüntüsü/sayfası bir oturumu kapat linki ya da butonu içeriyor mu?
- Oturumu kapattıktan sonra; aynı oturumu tekrar kullanabiliyor musunuz?(örneğin iki ya da üç tıklama öncesinde adresi tekrar tıklayın ya da tekrar kullanmayı deneyin)
- (Yüksek riskli uygulamalar için)Oturum kapatıldıktan sonra uygulama tarayıcı ön belleği ve geçmişini temizlemeniz için sizi uyarıyor mu?

Kendinizi Koruma (Yolları)

- Uygulamalarınıza oturum kapatma özelliğini implemente edin.
- Sadece index sayfasında değil; her sayfada bir oturumu kapat linki ya da butonu bulundurun
- Uygulamanız oturumu kapattığı ve tarayıcı tarafındaki tüm çerez bilgilerini temizlediğinden emin olun.
- (Yüksek riskli uygulamalar için)Uygulamanızda; kullanıcılarınızın paylaşılan bir bilgisayar kullanıyorlarsa tarayıcı geçmiş ve önbelleğini temizlemesi gerektiğini belirten bir yazı bulundurun.

Hesabın Kapatılması

Servisinizden hesap açtırıp, kullanan fakat daha sonra hizmetinizi kullanmaya devam etmeyecek olan kullanıcılar olacak; ya da çoğu kullanıcı başka hiçbir işlem için dönmeyeceklerdir.

Saldırıya açık olup olmadığınızı anlama (yolları) :

- Uygulama bir hesap kapatım mekanizmasına sahip mi?
- Bu kullanıcı ile ilgili tüm kayıtları siliyor mu?(modüler kayıtlar vergi ve hesaplama kontrolleri için işlem geçmişini tutmalıdır)
- Eğer kayıtlar önemsizse;gerekli olmayan tüm kayıtları siliyor mu?

Kendinizi Koruma (Yolları)

- Kullanıcıların kendi hesaplarını kapatmak için yetkileri olmalı.Bu işlem onay gerektirmelidir, ama aksi durumda kullanıcılara kendi hesaplarını silmesi konusunda zorlanacaksınız.
- Uzun bir sure oturum açılmamış hesaplar kitlenmeli; tercihen silinmeli.
- Eğer kayıtları tutarsanız, niye tuttuğunuzu gizlilik rejimlerine ve gizlilik bildirgenizde kullanıcılarınıza açıklamalısınız.

Hesapları kısmen silerken(örneğin işlem ya da hesaplarla ilgili kayıtları tutmalısınız) kişisel olarak tanımlanabilen bilgiler hiçbir şekilde bir web uygulaması arayüzü ile ulaşılamamalıdır.Örneğin onları harici bir veri tabanında ya da CSV formatında saklayabilirsiniz.

Kendi Kendine Kayıt

Kendi kendine kayıt uygulama şemaları, gizli kullanıcıların korunan kaynaklara erişmelerini sağlayabileceğini öğrenene kadar mükemmeldir.Kendi kendine kayıt özelliğini gerçekleştiren her uygulama suistimal edilmemek için atması gereken güvenlik adımlarını uygulamalıdır.

Saldırıya açık olup olmadığınızı anlama (yolları) :

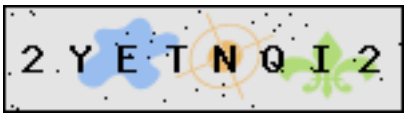
- Kendi kendine kayıt özelliği insan onayı olmaksızın tüm özelliklere tam erişim sağlıyor mu?
- Eğer limitler mevcutsa ; limitler onları tanımanızı sağlıyor mu?Çoğu uygulamalar özel URL bilgisini görmenize izin vermez; ama daha yetkili bir hesaptan bu URL koplayanıp yapıştırılabilir mi?
- Hesap yetkilerinin arttırılması için uygulama zorlanabilir mi ya da sosyal mühendislik yapılabilir mi?

Kendinizi Koruma (Yolları)

- İşinizin riskine göre dikkatli şekilde kendi kendine kayıt özelliğini gerçekleştirin. Örneğin yeni hesaplara işlem ya da parasal limitler koyabilirsiniz.
- Eğer limitler gereksiz yük getiriyorsa;işlemler onay sürecinden geçmelidir.Sadece güvenlikle değil; aynı zamanda gizlilikle yapılmalıdır.
- Hesabın özelliklerinin sadeliğinin maksimum olduğu konusunda emin olun.
- Hesaplar modifiye edildilirse;makul bir izinin ya da aktivitenin bilançosunun kayıt edildiğinden emin olun.

TOBIAT(CAPTCHA)

TOBIAT (“tamamen otomatikleştirilmiş bilgisayar – insane ayırma testi”) sistemleri söylenildiğine göre web tasarımcılarını insan olmayanların(bilgisayarların) web sitelerine kayıt olmalırını engelliyor.



CAPTCHA sistemlerini kullanmanın genel sebebi spamcileri spam ve pornografik linklerle forum ya da blogları kirletmelerini engellemektir.Arama motorlarının indeksleyebildiği tüm uygulamalar bu risk altındadır.

Saldırıya açık olup olmadığınızı anlama (yolları) :

CAPTCHA sistemlerini kırmanın temel yolu resmi alıp; insanları kullanarak onları kırmaktır.Bu genellikle yetişkinler için olan web sitelerde görülür.Bedava resimlere bakmak isteyen herhangi birisi ufak bir uğraş ile resimdeki harfleri yazar. Bu CAPTCHA mekanizmasını tamamiyle bozguna uğratır.

Sanal ya da duyulabilir CAPTCHA sistemleri doğaları gereği kör ya da sağır insanlar için elverişsizdirler ve sonuç olarak zeki optik karakter okuma programları ile bu aşamayı geçmeyi deneyebilirler.Genel olarak renk körü olan insanları kitlemektedir (erkek populasyonunun %10’u gibi yüksek bir rakam)

Not : Yasal olarak erişilmesi gereken hiçbir web sitesi CAPTCHA sistemlerini kullanmamalıdır

Kendinizi Koruma (Yolları)

CAPTCHA etiketlerini kullanmayın.Eğer herkes tarafından ulaşılması gereken bir uygulama hazırlıyorsanız CAPTCHA kullanımı yasadışıdır (genel olarak hükümet, sağlık, bankacılık uygulamaları)

Kullanmak zorundaysanız :

- Her zaman web sitenize kayıt olabilmek için başka metodlar da sağlayın.
- “no follow” etiketini kullanarak otomatik kayıtları engelleyin. Arama motorları bu etiket bulunan link ve sayfaları yok sayacaktır,link spamming kullanımını çokça düşürecektir.
- Yeni kayıt olmuş banka hesaplarının yetkilerini pozitif bir doğrulama yapılana kadar sınırlayın.Bu; kayıtlı bir kredi kartına tekil bir kimlik numarası ;ya da belli özelliklerin kullanıma açılması için belli bir süre beklenilmesi gibi limitler olabilir.

Daha Fazla Bilgi İçin

- "*Body Check*", c't Magazine. Very amusing article from 2002
  http://www.heise.de/ct/english/02/11/114/
- Klein, A., *NTLM Authentication and HTTP proxies don't mix*, posting to webappsec
  http://packetstormsecurity.nl/papers/general/NTLMhttp.txt
- How much does it take before your signature is verified? Apparently three plasma screens:
  http://www.zug.com/pranks/credit_card/
- Schneier, B., *The failure of two factor authentication*, blog / essay
  http://www.schneier.com/blog/archives/2005/03/the_failure_of.html
- Group blog led by Kim Cameron, *The Laws of Identity*
  http://www.identityblog.com/stories/2004/12/09/thelaws.html
- van der Stock, A., "*On the entropy of password reset systems*", unpublished research paper. If you'd like participate in the survey portion of this research, please contact vanderaj@owasp.org

Dökümanın Aslı : OWASP GUIDE 2.0.1

Çeviri **:** Mesut Timur – mesut@h-labs.org