Java Uygulama Güvenliği

Bora Güngören
Portakal Teknoloji
bora@portakalteknoloji.com

Telif Hakları – CopyLeft

- Bu belge GNU Belgeleme Lisansı ile korunmaktadır.
- Kullanıcılar kaynak belirttikleri ve GNU Belgeleme Lisansı'nı korudukları sürece
  - Belgeyi dağıtabilir
  - Belgeden alıntı yapabilir
  - Belgeyi değiştirebilir
- Lisans hakkında daha çok bilgi için belgenin yazarı yada Linux Kullanıcıları Derneği ile bağlantıya geçebilirsiniz.

Nelerden bahsedeceğiz?

- Güvenlik ve uygulama güvenliği kavramları
- Java'nın ilk çıkışı ve güvenlik
- Java'da virüs olur mu?
- Bayt kodu güvenliğinin önemi
- Nesne yaratılması ve kopyalanması
- Alt sınıflar
- Java'da Uygulama Güvenliği
- Java Güvenlik API'leri
  - JCE (Bugünkü örnek API'miz)
  - JSSE (Detaya girilmeyecek)
  - JAAS (Detaya girilmeyecek)

Güvenlik ve Uygulama Güvenliği Kavramları

- Bir bilgisayar sisteminin güvenliği çok boyutlu bir kavramdır
  - Virüsler, solucanlar ve truva atları
  - Kişisel bilgileri toplamaya ve işlemeye yönelik saldırılar
  - Yazılım bileşenlerinin, kaynaklarının ve kullanı-cıların kimliklerinin doğrulanması (authentication)
  - Veri aktarımı güvenliği (ağ üzerinde şifreleme)
  - Saklanan verinin güvenliği (dosyada yada veri tabanında şifreli saklama)

Güvenlik ve Uygulama Güvenliği Kavramları

- Bir bilgisayar sisteminin güvenliği çok boyutlu bir kavramdır
  - Şifreleme mekanizmasının güvenliği (kolay bulunan şifreler, kısa şifreler, kolay kırılan algo-ritmalar, rastgele sayı kaynaklarının niteliği)
  - Eylemlerin kaydı ve denetlenmesi (transaction logging and auditing)
  - Fiziksel güvenlik (Görevimiz Tehlike, Hudson Hawk, vs.)

Güvenlik ve Uygulama Güvenliği Kavramları

- Başarılı güvenlik sisteminin özellikleri
  - İyi tanımlanan kurallar ve prosedürleri olması
  - Kurallar ve prosedürlerin doğrulanması, test edilmesi ve güncellenmesi
  - Bağımsız güvenlik danışmanları tarafından incelenmesi ve teftişten geçmesi
  - Başarısız olduğunda yani saldırı önlenemediğinde ne yapılacağının planlanması
  - Sertifikasyon

Güvenlik ve Uygulama Güvenliği Kavramları

- Güvenli uygulamanın özellikleri
  - Kendisinin güvenilir olması (kilitlenmemesi, çökmemesi, aşırı kaynak tüketmemesi, vs.)
  - Virüs, solucan ve truva atlarının çalışmasına ve yayılmasına yardımcı olmaması
  - Kullanıcının bilgilerine özellikle erişme ihtiyacı duymaması (bilmediği verinin çalınmasına neden olamaz)
  - Çeşitli aşamalarda kimlik doğrulama yapması
  - Çeşitli aşamalarda şifreleme kullanması
  - Çeşitli kodlama hatalarından uzak olması

Java'nın İlk Çıkışı ve Güvenlik

- Java 1996'da duyuruldu ve 1997'den bu yana etkin biçimde kullanılmaktadır.
- 1996'daki güvenlik kavramları bugünkü kavramlardan farklıydı.
  - Internet yeni ortaya çıkmaktaydı.
  - Virüslerin davranışları farklıydı. (Boot virüsleri ile Melisa'yı karşılaştırın)
  - Internet'in virüs ve solucanların yayılması için bir ortam olmasından endişe ediliyordu. (Ne yazık ki oldu.)

Java'nın İlk Çıkışı ve Güvenlik

- Java ilk çıktığı zaman iki türde uygulama için pazarlandı:
  - Çok küçük masa üstü uygulamaları
  - Applet
- Her iki uygulama da Java Sanal Makinesi üzerinde bir “kum havuzu” içinde çalışacaktır.
- “Kum havuzu”nun kendisinin güvenlik sağ-laması yeterli olur mu?

Java'da Virüs Olur mu?

- Java'yı tasarlayanlar bir Java uygulamasının prensipte bir virüs barındırmasını engelle-meye çalıştılar. Bayt kodları ile bunu başar-dılar.
  - Java uygulamaları bayt kodu (Byte Code) adını verdiğimiz bir çeşit assembly diline dönüştürülmüş olarak saklanır.
  - Bayt kodları Java'nın varsaydığı kuramsal bir işlemciyi hedef alır. Bu 32 bitlik Sparc benzeri bir RISC işlemcisidir.

Java'da Virüs Olur mu?

- Bir Java uygulamasının .class uzantılı dosyası bayt kodlarından oluşur.
- Sanal Makine bu bayt kodlarını işletmeden önce denetler.
  - Bayt kodları elle değiştirilmiş ve bozulmuş uygulamalar tespit edilir ve çalıştırılmaz.
  - Klasik virüslerin yayılması böylece engellenmiş olur.
- Java Sanal Makinesi'nin bu özelliği yıllar içinde son derece gelişmiştir ve artık güvenilir bir seviyededir.

Java'da Virüs Olur mu?

- Çağdaş bir Java Sanal Makinesi'nde çalışan Java uygulamalarında klasik anlamda virüs-lerin barınamayacağını söyleyebiliriz.

Bayt Kodu Güvenliğinin Önemi

- Ancak bayt kodlarına müdahale ederek daha farklı sorunlar yaratmak da mümkündür.
  - Java tip sistemi açısından hassas bir dildir. Tip sistemine yapılan müdahaleler bir uygulamanın çökmesine neden olabilir.
  - Bir sınıfın yerine geçen bir sınıf yada bir yöntemin yerine geçen bir yöntem yaratarak sis-teme sokabilirsiniz.

Bayt Kodu Güvenliğinin Önemi

- Bayt kodlarına müdahale ederek yapılabilecekler
  - System.arraycopy() yönteminin yerine geçen bir yöntem ile Sanal Makine'de çalışan tüm uygulamalarda kopyalanan her diziye erişebilirsiniz.
  - String sınıfı yöntemlerinin yerine geçen yöntemler ile işlenen tüm metinlere erişebilirsiniz.
  - Akış (stream) merkanizmasında kullanılan sınıfların yerine geçip yada araya yerleşip tüm dosya erişimlerini dinleyebilirsiniz.

Bayt Kodu Güvenliğinin Önemi

- Bayt kodu güvenliği bu gibi nedenlerle son derece önemli olmuştur.
- Buradaki tüm güvenlik önlemleri Sanal Makine tarafından sağlanmaktadır.
- Java Sanal Makinesi'nin güncel bir sürümünün kullanılması son derece önemlidir.
  - Örneğin MS Windows 98 ile gelen standart sanal makineyi kullanmak uygun değildir.

Nesne Yaratılması ve Kopyalan-ması

- Java'da nesneleri yaratmanın tek yolu new işlecini kullanmaktır.
  - Ancak daha önceden yaratılmış ve saklanmış bir nesneyi veri tabanı kayıtlarından yada serileş-tirme ile dosyadan almak mümkündür.
  - Klonlama mekanizmaları yeni bir nesne verme-yebilir.
  - Yada RMI ile uzaktaki bir makinedeki nesnelere erişebiliriz.
- Yaratmak dışındaki bir şekilde elimize geçen nesnelere referans tutan tek kişi biz olma-yabiliriz.

Nesne Yaratılması ve Kopyalan-ması

- Bir uygulamanın kullandığı nesneye referans sahibi bir diğer uygulama o nesnenin içeriğini alabilir yada değiştirebilir.
  - Örneğin işletim sistemi tarafında çalıştırılacak bir komut varsa, bu komutu saklayan String nesnesinin sonuna “; rm -rf /” gibi bir metin eklenebilir.
  - Çok kanallı mimariler bunun yapılmasına olanak verecektir.
  - Bu tip nesnelere erişimin eş zamanlılık denetimi ile yapılması için synchronized sözcüğünün kullanılması gerekir.

Alt Sınıflar

- Bir sistemde bulunan herhangi bir sınıfın bir alt sınıfı yazıldığında bu alt sınıfın yapacağı işler konusunda bir varsayımda bulunamayız.
  - Alt sınıf nesnesini orijinal sınıf nesnesi yerine sokan saldırgan bir çok yere erişebilir.
  - Bu da bir çeşit bayt kodu saldırısı sayılabilir.
- Kritik işlemler yapan sınıfların “son sınıf” (final class) olarak tanımlanması gerekir.
  - Bu sayede saldırgan bir alt sınıf yazamaz.

Java'da Uygulama Güvenliği

- Java'da sanal makine düzeyindeki güvenlik mekanizmaları ancak ilkel uygulamalar için yeterlidir.
  - Bütün diğer güvenlik ölçütleri açısından Java'nın C++ yada başka bir dile karşı özel bir üstünlüğü yoktur.
  - Ancak Java'nın güvenlik için son derece yararlı 3 adet API'si vardır.
  - Bu API'lerin yerinde kullanımı ile son derece güvenli Java uygulamaları yazılabilir.
  - Ayrıca Tomcat, vb. uygulamaların da kendi güvenlik becerileri olacaktır.

Java Güvenlik API'leri

- Java Cryptography Extension (JCE)
  - Şifreleme
- Java Secure Sockets Extension (JSSE)
  - SSL kullanımı
- Java Authentication and Authorization Service (JAAS)
  - Kimlik doğrulama ve kimliğe dayalı yetkilendirme

Java Güvenlik API'leri

- Bu üç API her ne kadar tek başlarına kullanılacak gibi gözükse de doğru kullanım birlikte kullanılmalarıdır.
  - Salt şifreleme ile güvende değilsiniz.
  - Sadece SSL kullandığınızda yeterince güvende değilsiniz.
  - Sadece kullanıcıların kimliklerini doğruladığınızda yeterince güvende değilsiniz.

Java Cryptography Extension - JCE

- JCE dört temel işleve yöneliktir
  - Şifreleme (en yaygın olarak kullanılan işlevi)
  - Güvenli anahtar değişimi
  - Güvenli mesajlaşma
  - Anahtar yönetimi
- JCE Bazı J2SDK'lar ile gelmez.
  - http://java.sun.com/products/jce/ adresinden indirebilirsiniz.
  - 1.4 ve 1.5 ile standart olarak gelir.
  - Şu andaki sürümler ABD'nin ilgili yasalarına uyumludur. JCE'nin ABD dışına çıkartılmasında sorun yoktur.

Java Cryptography Extension - JCE

- JCE kurulumu standart bir JAR kurulumundan farklıdır.
  - “Bundled” kurulum yapmak için sistem yöneticisi olmanız gerekmez ancak bu sefer CLASSPATH ve çok sayıda güvenlik ayarlarını elinizle yapmanız gerekir.
  - “Installed” kurulum için ise sistem yöneticisi olmanız gerekir. $JREHOME dizini içine bazı dosyalar konulması gerekecektir.

Java Cryptography Extension - JCE

- Installed Kurulum:
  - JCE ile gelen JAR dosyalarını $JREHOME/lib/ext dizinine yerleştirin.
  - $JREHOME/lib/security/java.security dosyası içinde aşağıdaki biçimde satırlar olacaktır
    - security.provider.<n>=<sınıf adı>
  - Bunlardan en az 1 tane gereklidir
    - security.provider.1=sun.security.provider.Sun
  - Biz bunların arkasına sayıyı artırarak aşağıdaki gibi bir satır ekleyeceğiz:
    - security.provider.4=com.sun.crypto.provider.SunJCE

Java Cryptography Extension - JCE

- Installed Kurulum:
  - Rastgele sayı kaynağının ayarlanması önemlidir.
  - UNIX'lerde /dev/random yararlı bir kaynaktır.
  - Ancak eğer bir kriptografi kartınız varsa onun rastgele sayı üreten bir aygıt olarak tanıtılması mümkündür.
  - java.security dosyasında aşağıdaki ayar bunu belirtir
    - securerandom.source=file:/dev/random

Java Cryptography Extension - JCE

- Nasıl kod yazacağız?
  - javax.crypto paketi
- En yoğun kullanacağımız sınıflar ve arabirimler
  - KeyGenerator
  - SecretKey (arabirim)
  - SecretKeySpec
  - Cipher

Java Cryptography Extension - JCE

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;

class JCEDemo {

public static void main(String args[]){
try {
```

Java Cryptography Extension - JCE

```
KeyGenerator yaratici =
                    new KeyGenerator("DES");
SecretKey anahtar = kgen.generatekey();
byte[ ] bayt = secretKey.getEncoded();
SecretKey spec =
                   new SecretKeySpec(bayt, "DES");
Cipher sifre = Cipher.getInstance("DES");
// ...
```

Java Cryptography Extension - JCE

```
sifre.init(Cipher.ENCRYPT_MODE,spec);
String sifresiz = "Bora Güngören";
byte[ ] sifrelibayt = sifre.doFinal(
                                    sifresi.getbytes());
//...
sifre.init(Cipher.ENCRYPT_MODE,spec);
byte[ ] cozuldu = sifre.doFinal(sifrelibayt);
//...
}
catch (Exception e) { /* */}
}}
```

Java Cryptography Extension - JCE

- Daha neler var?
  - Anahtarların transferi için uygulamalar
  - Şifreli akışlar
  - Şifreleme için gerekli sınıfları yaratan Abstract Factory sınıfı
  - Nesne şifreleme
  - Anahtar yönetimi
- Referans belgelemesi
  - http://java.sun.com/j2se/1.4.2/docs/guide/security/jce/JCERefGuide.html

Java Secure Sockets Extension - JSSE

- Uygulamalarımızın HTTP ve FTP üzerinden yaptığı işlemlerde SSL ve TLS desteği sağlar.
- SSL Sunucusu ve SSL İstemcisi olarak iki parçalı çalışır.

Java Authentication and Authorization Service - JAAS

- Her kullanıcının sisteme girişi (login) doğrulanır.
- Her girişin bir hedefi (Subject) olacaktır.
- Her varlığın erişim hakları olur.
- Hedefimiz bir yada birden fazla kimlik ve bu kimliklere erişim hakkı tanınmış varlıklardan oluşur.

Java Authentication and Authorization Service - JAAS

- JAAS ayar dosyalarınızda bir LoginModule uyarlaması (javax.security paketi içinde) tanımlanır
- LoginModule ile kullanıcı arasındaki iletişimin nasıl olacağı planlanır ve bu plan uygulanır.
- Bir LoginContext nesnesi yaratılır. Bu nesne gerçek sisteme giriş (login) işlerini üstlenir.
- LoginContext nesnesi, hedefi içerir.

Önerilen Kaynaklar

- http://java.sun.com/
- “Hacking Exposed J2EE & Java”, Buege, Layman, Taylor, Osborne McGrawHill, 2002
- “Java Security”, Oaks, O'Reilly, 2001
- “Java Network Programming”, Harold, O'Reilly, 2000.