Testing for Format String Injection (Biçim Dizesi Enjeksiyonu Testi)

Summary (Özet)

Bir format dize, çalışma saatinde yorumlanan veya dönüştürülen dönüşüm belirteçleri de içeren hüküm ve son derecelenmiş bir karakter dizisidir. Sunucu tarafı kodu bir kullanıcının girdisini bir format dize ile uyumlu hale getirirse, bir saldırgan çalışma süresi hatasına, bilgi açıklamasına veya tampon taşmasına neden olmak için ek dönüşüm belirteçleri ekleyebilir.

Format dizeleri güvenlik açıkları için en kötü durum, argümanları kontrol etmeyen ve aynı zamanda bir tane içeren dillerde ortaya çıkar. `%n` Hafızaya yazan bir belirteç. Bu işlevler, bir format dizisini değiştiren bir saldırgan tarafından sömürüldüyse, bilgi açıklamasına ve kod yürütülmesine neden olabilir:

- C ve C++ baskı ve benzeri yöntemler fprintf, sprintf, snprintf
- Perl printf ve sprintf

Bu format dize fonksiyonları hafızaya yazamaz, ancak saldırganlar, geliştiricilerin göndermeyi düşünmediği çıkış değerlerine format dizelerini değiştirerek bilgi açıklamasına neden olabilir:

- Python 2.6 ve 2.7 str.format ve Python 3 unicode str.format, bellekteki diğer değişkenlere işaret edebilen dizeleri enjekte ederek değiştirilebilir

Aşağıdaki format dize fonksiyonları, saldırgan dönüşüm belirteçleri eklerse çalışma süresi hatalarına neden olabilir:

- Java String.format ve PrintStream.format
- PHP printf

Bir format dize kırılganlığına neden olan kod deseni, sağlıksız kullanıcı girdisi içeren bir dize biçimi işlevine yapılan bir çağrıdır. Aşağıdaki örnek bir hatanın nasıl olduğunu gösterir `printf` Bir programı savunmasız hale getirebilir:

C'de örnek:

```
char *userName = /* input from user controlled field */;

printf("DEBUG Current user: ");
// Vulnerable debugging code
printf(userName);
```

Java'da örnek:

```
final String userName = /* input from user controlled field */;

System.out.printf("DEBUG Current user: ");
// Vulnerable code:
System.out.printf(userName);
```

Bu özel örnekte, saldırgan onların setini alırsa `userName` Bir veya daha fazla dönüşüm belirtecisine sahip olmak için, istenmeyen davranışlar olacaktır. C örneği, bellek içeriğinin yazdırılması durumunda `userName` Kontrollü `%p%p%p%p%p`, ve eğer bir şey varsa hafıza içeriğini bozabilir `%n` İpin içinde. Java örneğinde, bir `username` Bir girdiye ihtiyaç duyan herhangi bir belirticiyi içeren (içerik dahil `%x` ya da `%s`) programın çökmesine neden olur `IllegalFormatException`. . Örnekler hala başka sorunlara maruz kalsa da, kırılganlık baskı argümanlarıyla giderilebilir `printf("DEBUG Current user: %s", userName)`. .

Test Objectives (Test Hedefleri)

- Biçimlendirici dize dönüşüm belirteçlerini kullanıcı kontrollü alanlara enjekte etmenin uygulamadan istenmeyen davranışlara neden olup olmadığını değerlendirir.

How To Test (Nasıl Test Edilir)

Testler, kodun analizini ve test altındaki uygulamaya kullanıcı girişi olarak dönüşüm belirteçlerinin enjekte edilmesini içerir.

(Statik Analiz)

Statik analiz araçları, kodda veya ikili ürünlerde format dize güvenlik açıklarını bulabilir. Araç örnekleri şunlardır:

- C ve C++: Kusurlu
- Java: Find SecurityBugs kuralı FORMAT_STRING_MANIPULATION
- PHP: String formatter Analyzer phpsa

(Manuel Kod Muayenesi)

Statik analiz, karmaşık kod tarafından oluşturulan format dizeleri de dahil olmak üzere daha ince durumları kaçırabilir. Bir kod tabanında manuel olarak güvenlik açıklarını aramak için, bir test cihazı, bir format dizesini kabul eden ve güvenilmemiş girişin format dizesini değiştiremeyeceğinden emin olmak için takip eden kod tabanındaki tüm aramaları arayabilir.

(Dönüşüm Belirtici Enjeksiyonu)

Testçiler, herhangi bir dize girişinde dönüşüm belirteçleri göndererek birim test veya tam sistem test seviyesinde kontrol edebilirler. Programı, test halindeki sistemin tüm dilleri için tüm dönüşüm belirteçlerini kullanarak doldurun. Kullanılması gereken olası girişler için OWASP Format dize saldırı sayfasına bakın. Test başarısız olursa, program çöker veya beklenmedik bir çıkış gösterecektir. Test geçerse, bir dönüşüm belirteci gönderme girişimi engellenmelidir veya dize, başka herhangi bir geçerli girişte olduğu gibi hiçbir sorun olmadan sistemden geçmelidir.

Aşağıdaki alt bölümlerdeki örnekler bu biçimden bir URL'ye sahiptir:

`https://vulnerable_host/userinfo?username=x`

- Kullanıcı kontrollü değerdir `x` (The için) `username` Parametresi)

(Manuel Enjeksiyon)

Test cihazları, bir web tarayıcısı veya diğer web API hata ayıklama araçlarını kullanarak manuel bir test gerçekleştirebilir. Web uygulamasına veya siteye, sorgunun dönüşüm belirteçleri olduğu gibi göz atın. Bir URL içinde gönderilirse çoğu dönüşüm belirteçlerinin kodlamaya ihtiyaç duyduğunu unutmayın, çünkü özel karakterler de dahil olmak üzere `%` ve `{`. . Test bir dizi specifier tanıtabilir `%s%s%s%n` Aşağıdaki URL ile göz atarak:

`https://vulnerable_host/userinfo?username=%25s%25s%25s%25n`

Web sitesi savunmasızsa, tarayıcı veya araç bir zaman aşımı veya HTTP iade kodu 500'ü içerebilecek bir hata almalıdır.

Java kodu hatayı geri verir

`java.util.MissingFormatArgumentException: Format specifier '%s'`

C uygulamasına bağlı olarak, süreç tamamen çökebilir `Segmentation Fault` . .

Alet Assisted Fuzzing

Wfuzz dahil olmak üzere bulaşık makineleri enjeksiyon testlerini otomatikleştirebilir.
wfuzz için, satır başına bir girişle bir metin dosyasıyla (bu örnekte fuzz.txt) ile başlayın:

fuzz.txt:

```
alice
%s%s%s%n
%p%p%p%p%p
{event.__init__.__globals__[CONFIG][SECRET_KEY]}
```

The (İngilizce) `fuzz.txt` Dosya aşağıdakileri içerir:

- Geçerli bir giriş `alice` Uygulamanın doğrulanması normal bir girdiyi işleyebilir
- C benzeri dönüşüm belirteçleri ile iki dize
- Küresel değişkenleri okumaya çalışmak için bir Python dönüşüm belirteç

Bulanık giriş dosyasını test altındaki web uygulamasına göndermek için aşağıdaki komutu kullanın:

`wfuzz -c -z file,fuzz.txt,urlencode https://vulnerable_host/userinfo?username=FUZZ`

Yukarıdaki çağrıda, `urlencode` argüman, ipler için uygun kaçmayı sağlar ve `FUZZ` (Büyük harflerle) araçta girdileri nereden tanıtacağını söyler.

Bir örnek çıktı aşağıdaki gibidir

```
ID        Response   Lines    Word     Chars      Payload
=====================================================
```

```
000000002:   500       0 L      5 W      142 Ch      "%25s%25s%25s%25n"
000000003:   500       0 L      5 W      137 Ch      "%25p%25p%25p%25p%25p"
000000004:   200       0 L      1 W      48 Ch       "%7Bevent.__init__.__g
lobals__%5BCONFIG%5D%5BSECRET_KEY%5D%7D"
000000001:   200       0 L      1 W      5 Ch        "alice"
```

Yukarıdaki sonuç, uygulamanın C benzeri dönüşüm belirteçlerinin enjeksiyonuna olan zayıflığını doğrular `%s` ve `%p`. .