Appendix D. Encoded Injection (Ek D. Kodlanmış Enjeksiyon)

Background (Arka plan)

Karakter kodlaması, karakterlerin, sayıların ve diğer sembolleri standart bir formata haritalama işlemidir. Tipik olarak, bu, gönderen ve alıcı arasında iletmeye hazır bir mesaj oluşturmak için yapılır. Basit bir ifadeyle, karakterlerin (İngilizce, Çince, Yunanca veya bilinen başka herhangi bir dil gibi farklı dillere ait) baytlara dönüştürülmesidir. Yaygın olarak kullanılan bir karakter kodlama şemasının bir örneği, başlangıçta 7-bit kodları kullanan Amerikan Standart Kodunu (ASCII). Kodlama şemalarının daha yeni örnekleri Unicode olacaktır `UTF-8` ve `UTF-16` Bilgisayar endüstrisi standartları.

Uygulama güvenliği alanında ve mevcut kodlama şemalarının bolluğu nedeniyle, karakter kodlaması popüler bir kötüye kullanıma sahiptir. Kötü amaçlı enjeksiyon iplerini onları gizleyecek şekilde kodlamak için kullanılıyor. Bu, giriş doğrulama filtrelerinin baypas edilmesine yol açabilir veya tarayıcıların kodlanmış metinleri oluşturmanın belirli yollarından yararlanabilir.

Input Encoding – Filter Evasion (Giriş Kodlama – Filtre Kaçış)

Web uygulamaları genellikle kullanıcı tarafından gönderilebilecek girişi sınırlamak için farklı türde giriş filtreleme mekanizmaları kullanır. Bu giriş filtreleri yeterince iyi uygulanmazsa, bu filtrelerden bir veya iki karakterin kayması mümkündür. Örneğin, bir `/` Temsil edilebilir `2F` (hex) ASCII'de, aynı karakter (`/`) kodlandığı gibi kodlanmıştır `C0AF` Unicode'da (2 bayt dizi). Bu nedenle, giriş filtreleme kontrolünün kullanılan kodlama şemasının farkında olması önemlidir. Filtrenin sadece tespit edildiği tespit edilirse `UTF-8` Kodlanmış enjeksiyonlar, bu filtreyi atlamak için farklı bir kodlama şeması kullanılabilir.

Output Encoding – Server & Browser Consensus (Çıkış Kodlama – Sunucu ve Tarayıcı Konsensüsü)

Web tarayıcıları, tutarlı bir şekilde bir web sayfasını görüntülemek için kullanılan kodlama şemasının farkında olmalıdır. İdeal olarak, bu bilgiler HTTP başlığındaki tarayıcıya sağlanmalıdır (`Content-Type` Alan, aşağıda gösterildiği gibi: `Content-Type: text/html; charset=UTF-8`

veya HTML META etiketi ile (`META HTTP-EQUIV`), aşağıda gösterildiği gibi:

```
<META http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
```

Bu karakter kodlama beyanları aracılığıyla, tarayıcının bytes karakterleri karakterlere dönüştürürken hangi karakter setinin kullanılacağını anladığıdır. HTTP başlığında belirtilen içerik tipinin META etiket beyannamesi üzerinden önceliği olduğunu unutmayın.

CERT bunu burada şöyle anlatıyor:

Birçok web sayfası karakter kodlamasını bırakır (`charset` parametresi HTTP) tanımlanmamış. HTML ve HTTP'nin daha önceki sürümlerinde, karakter kodlamasının varsayılan olması gerekiyordu. `ISO-8859-1` Eğer tanımlanmadıysa. Aslında, birçok tarayıcı farklı bir varsayılana sahipti, bu yüzden varsayılan varlığa güvenmek mümkün değildi.

`ISO-8859-1`. . HTML sürümü 4 bunu meşrulaştırır - karakter kodlama belirtilmemişse, herhangi bir karakter kodlama kullanılabilir.

Web sunucusu hangi karakter kodlamasının kullanıldığını belirtmezse, hangi karakterlerin özel olduğunu söyleyemez. Belirsiz karakter kodlaması olan web sayfaları çoğu zaman çalışır, çünkü çoğu karakter seti aynı karakterleri 128'in altındaki değerlere atamaktadır. 128'in üzerindeki değerlerden hangisi özeldir? Bazı 16-bit `character-encoding` Şemalar, özel karakterler için ek çoklu bayt temsillere sahiptir. `<` . . Bazı tarayıcılar bu alternatif kodlamayı tanır ve üzerinde hareket
eder. Bu "doğru" davranıştır, ancak kötü amaçlı komut dosyaları kullanarak saldırıları önlemek için çok daha zor hale getirir. Sunucu, hangi bayt dizilerinin özel karakterleri temsil ettiğini bilmiyor.

Bu nedenle, sunucudan gelen karakterin bilgi almaması durumunda, tarayıcı ya kodlama şemasını tahmin etmeye çalışır ya da varsayılan bir şemaya geri döner. Bazı durumlarda, kullanıcı tarayıcıdaki varsayılan kodlamayı açıkça farklı bir şemaya ayarlar. Web sayfası (seçmen) ve
tarayıcı tarafından kullanılan kodlama şemasındaki bu tür uyumsuzluklar, tarayıcının sayfayı istenmeyen veya beklenmedik bir şekilde yorumlamasına neden olabilir.

Encoded Injections (Kodlanmış Enjeksiyonlar)

Aşağıda verilen tüm senaryolar, giriş filtrelerini atlamak için gizlenmenin sağlanabileceği çeşitli yolların bir alt kümesi oluşturur. Ayrıca, kodlanmış enjeksiyonların başarısı, kullanımdaki tarayıcıya
bağlıdır. Örneğin,
`US-ASCII` Endoş enjeksiyonlar daha önce sadece IE tarayıcısında başarılıydı, ancak Firefox'ta değil. Bu nedenle, kodlanmış enjeksiyonların, büyük ölçüde, tarayıcıya bağımlı olduğu belirtilebilir.

Basic Encoding (Temel Kodlama)

Tek bir alıntı karakterinin enjeksiyonuna karşı koruyan temel bir giriş doğrulama filtresi düşünün. Bu durumda aşağıdaki enjeksiyon bu filtreyi kolayca atlaracaktır:

`<script>alert(String.fromCharCode(88,83,83))</script>` `String.fromCharCode` JavaScript
fonksiyonu belirli Unicode değerlerini alır ve ilgili dizeyi döndürür. Bu, kodlanmış enjeksiyonların en temel biçimlerinden biridir. Bu filtreyi atlamak için kullanılabilecek başka bir vektör şunlardır:

`<IMG src="" onerror=javascript:alert(&quot;XSS&quot;)>`

Veya ilgili HTML karakter kodlarını kullanarak:

`<IMG src="" onerror="javascript:alert(&#34;XSS&#34;)">`

Yukarıdakiler enjeksiyon dizesini oluşturmak için HTML Kuruluşlarını kullanır. HTML Kuruluşları kodlaması, HTML'de özel bir anlamı olan karakterleri görüntülemek için kullanılır. Örneğin, `>` Bir
HTML etiketi için kapanış parantez olarak çalışır. Bu karakterin web sayfasında gerçekten gösterilmesi için HTML karakter varlıkları sayfa kaynağına eklenmelidir. Yukarıda belirtilen enjeksiyonlar kodlamanın bir yoludur. Bir ipin yukarıdaki filtreyi atlamak için kodlanabileceği
(sıkıntılı) başka yollar vardır.

Hex Encoding (Hex Kodlama)

Hex, Hexadecimal'in kısaltması, 16 sayılık bir sistemdir, yani 16 farklı değere sahiptir. `0` tot için `9` ve `A` tot için `F` Çeşitli karakterleri temsil etmek. Hex kodlaması, bazen giriş doğrulama
filtrelerini atlamak için kullanılan başka bir bulanıklık şeklidir. Örneğin, dizenin yan kodlanmış versiyonu `<IMG SRC=javascript:alert('XSS')>` Öyle `<IMG SRC=%6A%61%76%61%73%63%72%69%70%74%3A%61%6C%65%72%74%28%27%58%53%53%27%29>`

Yukarıda dizenin bir varyasyonu aşağıda verilmiştir. '%' filtrelenirse kullanılabilir:

```
<IMG
SRC=&#x6A&#x61&#x76&#x61&#x73&#x63&#x72&#x69&#x70&#x74&#x3A&#x61&#x6C&#x65&#x72&#x74&#x28&#x27&#x58&#x53&#x53&#x27&#x29>
```

Base64 ve Octal gibi, bulanıklık için kullanılabilecek başka kodlama planları da vardır. Her ne kadar her kodlama şeması her seferinde çalışmasa da, akıllı manipülasyonlarla birleştirilmiş biraz deneme ve hata, zayıf inşa edilmiş bir giriş doğrulama filtresindeki boşluğu kesinlikle ortaya çıkaracaktır.

UTF-7 Encoding (UTF-7 Kodlama)

UTF-7 kodlaması

```
<SCRIPT>
  alert('XSS');
</SCRIPT>
```

Aşağıdaki gibidir

```
+ADw-SCRIPT+AD4-alert('XSS');+ADw-/SCRIPT+AD4-
```

Yukarıdaki komut dosyasının çalışması için tarayıcının web sayfasını kodlanmış olarak yorumlaması gerekir `UTF-7` . .

Multi-byte Encoding (Multi-bayt Kodlama)

Değişken-geniş kodlama, karakterleri kodlamak için farklı uzunluklarda kodları kullanan başka bir karakter kodlama şeması türüdür. Multi-Byte Kodlama, bir karakteri temsil etmek için değişen sayıda bayt kullanan bir değişken geniş kodlama türüdür. Multi-bayt kodlama, öncelikle büyük bir karaktere ait karakterleri kodlamak için kullanılır. Çince, Japon ve Koreliler.

Multibyte kodlaması geçmişte standart giriş doğrulama işlevlerini atlamak ve sitenin çapraz komut dosyası ve SQL enjeksiyon saldırılarını gerçekleştirmek için kullanılmıştır.

Referances (Referanslar)

- Kodlama (Semioti)
- HTML Kuruluşları
- Giriş doğrulama saldırıları nasıl önlenir
- Unicode ve Karakter Setleri