Sadece 2 İstekle MySQL Blind SQL Injection

Canberk BOLAT
canberk.bolat[-AT-]gmail[-DOT-]com

0x01 – Giriş

Bu yazı blind sql injection saldırılarında tek bir karakter için minimum istek ile doğru sonuca varabilmek adına yeni bir tekniğin araştırılması sırasında kazandığım deneyimler üzerine kaleme alındı.

"Yeni Teknik" olarak belirtmemin sebebi aslında varolan bir blind sql injection tekniğinin farklı bir şekilde kullanılması, sonucun daha az istekle elde edilmesi ve henüz public olarak böyle bir fikrin ortaya atılmamış olmasıdır.

0x02 - Fikrin Doğuşu

Fikir blind sql injection yaparken bulmaya çalıştığımız karakterlerin ascii karşılıklarını kullanarak bazı "işaretler" yakalamaya çalışmanın sonucu doğdu. Şöyle bir giriş yapayım, klasik blind sql injection'da aşağıdaki gibi bir sorguyu çalıştırıyor ve sonuca binary search ile maksimum 7 istekte ulaşıyoruz. Örnekteki database'in ilk karakteri "c"(ASCII: 99) olsun.

1 AND 1=IF((**ASCII(SUBSTR((SELECT DATABASE()),1,1))**>97),1,0)

Bilindiği üzere altı çizili ifade bize ilk karakterin ASCII karşılığını veriyor ve daha sonra büyük mü küçük mü karşılaştırıyoruz vs... Ayrıca biliyoruz ki doğru sonuç orada öylece duruyor ve biz sadece bunu karşılaştırmakla yetiniyoruz. Neden bu rakamsal ifadeyi birşeyler için kullanmayalım?? (İlk başta banada çok çılgınca gelmişti fakat bu mümkün)

0x03 - İlk Denemeler

Burada 99 rakamını kullanabileceğimiz en uygun fonksiyon sanırım SLEEP(). SLEEP() belirtilen süre boyunca MySQL'u "uyutuyor". Aşağıdaki örneği incelersek aslında baştan beri neyden bahsettiğimi, fikrin ne olduğunu çok iyi anlayabilirsiniz.

1 AND SLEEP(**ASCII(SUBSTR((SELECT DATABASE()),1,1)**)

Bu örnek şunu söylüyor, MySQL database adının ilk karakterinin ascii karşılığı yani 99 saniye boyunca uyuyor. Fakat bu süre çok fazla :(Mesela bu süreden 90 çıkartalım yani MySQL "c" harfi için 9 saniye uyusun.

1 AND SLEEP(**(ASCII(SUBSTR((SELECT DATABASE()),1,1)-90)**)

O da ne? Gerçekten istediğimiz oldu ve MySQL 9 saniye uyudu!! Şimdi basit bir alet yazalım belirttiğimiz isteği yapsın ve bu isteğin süresine 90 ekleyerek bize hangi harfin olduğunu söylesin.(Kimse yorulmasın diye o aleti ben sizin yerinize yazdım ve test ettim J)

```
hcr@world:~$ php bsqli-v1.php
[*] Result: can
[*] Total HTTP Request: 3
[*] Total Time: 36 seconds
```

Fakat bu yöntemin bazı zorlukları var, mesela "z" harfi için 122-90 yani 32 saniye beklemek zorundasınız. Değer mi? Tek bir istek ile sonuca ulaşabiliyorsak eğer değer! Ama bununla yetinmek zorunda değiliz, tek bir karakter için istek sayısını ikiye çıkartalım ama kullandığımız yöntem biraz daha farklı olsun.

0x04 - Modüler Aritmetik

Bu yöntemde ise MySQL'un modüler aritmetik fonksiyonu olan MOD()'tan destek alarak denememizi yapıcaz. Şöyle bir sonuca ulaştım paylaşayım, 97 ile 122 aralığındaki sayıları 13'e böldüğümüzde kalan değer bu aralıkta sadece iki kez var.

MOD(ASCII('a'),13) == 6
MOD(ASCII('n'),13) == 6
---> MOD(ASCII('a'),13) == MOD(ASCII('n'),13)

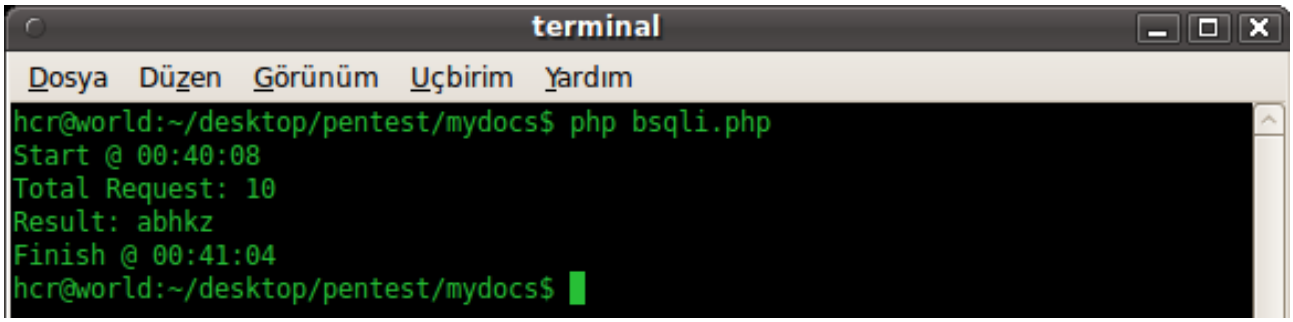
Bu durum aslında false-positive veriyor, çünkü tek istek yaptıysanız ve sleep süresi 6 saniye ise o karakter "a"da olabilir, "n"de olabilir. 97-122 aralığındaki sayıların 13'e bölümlerinden kalan değerleri bir array'de topladım ve şöyle bir sonuç çıktı.

```
array(97 => "6", 98 => "7", 99 => "8", 100 => "9", 101 => "10", 102 => "11",
      103 => "12", 104 => "0", 105 => "1", 106 => "2", 107 => "3", 108 => "4",
      109 => "5", 110 => "6", 111 => "7", 112 => "8", 113 => "9", 114 => "10",
      115 => "11", 116 => "12", 117 => "0", 118 => "1", 119 => "2", 120 => "3",
      121 => "4", 122 => "5");
```

Gördüğünüz gibi sadece 2 sayı aynı kalan değerine sahip. Bu durumda 1/2 oranında bir olasılık ile karşı karşıyayız. İşimizi olasılığa bırakmayalım ve maksimum istek sayısı 2 olsun ve sonuç tatmin edici olsun.

```
1 AND IF((ASCII(SUBSTR((SELECT DATABASE()),1,1))=97),sleep(5),0)
```

Burada bir denetim yaptık ve eğer ilk karakterin ASCII karşılığı 97 ise MySQL 5 saniye uyusun dedik anlıyacağınız klasik zaman tabanlı blind sql injection ile denetimimizi yaptık. Sonuç 97 değilse zaten otomatik olarak olasılık ortadan kalkmış oluyor sonuç "a" değilde diğer ihtimal "n" oluyor ki onuda bir daha karşılaştırmaya gerek kalmıyor. 5 karakterden oluşan bir database adını PoC ile test ettim ve 56 saniye gibi bir sürede sadece 10 istek ile sonuca ulaştım.



Tabi PoC'da bir şey eksik o da verinin uzunluğu onuda yine tek bir istek ile bulmak mümkün ve sonuç olarak toplamda 11 istek ile 5 karakteri tahmin edebiliyoruz.

1 AND 1=SLEEP((SELECT LENGTH(DATABASE())+5))

Yukarıdan sorgunun çalıştırılması 10 saniye sürsün. Biz 5 çıkartırsak kalır bize 5 ve buda karakter sayısını bize verir. :)

0x05 – İstek Sayısını Azaltmak

Modüler aritmetik ile herşey çok güzeldi fakat şöyle bir sorun çıkıyordu, bu yöntem sadece 97-122 aralığındaki karakterler için çalışıyordu ve 2 istek yapıyordu. Halbuki üzerinde biraz daha çalışma yapıldığında 2 istek ile bütün karakterleri bulmak mümkün. Örneğin 13 yerine 94 sayısını ele alalım. Aşağıdaki gibi bir sonuç elde ediyoruz..

97 => 3
98 => 4
99 => 5
...
121 => 27
122 => 28

Bu sonuç şu anlama geliyor biz 97-122 arasındaki bir karakteri tek bir istek ile maksimum 28 saniyede bulabiliriz. Ama bizim için önemli olan 48-122 aralığındaki (genel olarak verilerin bu aralıktaki karakterleri içermesinden dolayı bu aralık seçilmiştir.) karakterler.

0X06 – 2 İstek ve Tüm Karakterler

İlk başta bir istek ile karakter sayısını elde etmemiz şart yani elde var 1 istek. 48-122 aralığında 75 karakter var ve 75 karakter için uygun bir bölen elde etmek zor olacağından biz bu aralığı 48-85 ve 86-122 şeklinde iki parçaya ayıralım. Örneğin ilk karakteri arıyoruz, ilk yapacağımız iş bu karakter 85'ten büyük mü küçük mü onu tespit etmek. Aşağıda örnek bir sorgu var,

1 AND 1=IF((ASCII(SUBSTR((SELECT DATABASE()),1,1))>85),SLEEP(3),0)

Karakter 85'ten büyükse 3 saniye uyusun ve biz “sinyal”imizi alalım. Gerçektende karakter 85'ten büyük ve aradığımız aralık 86-122 aralığı. İlk isteği yaptık ve geldi sıra ikinci isteğe. 86-122 aralığında bölünecek sayı olarak 83'ü belirledim ben ve şöyle bir sonuç elde ettim.

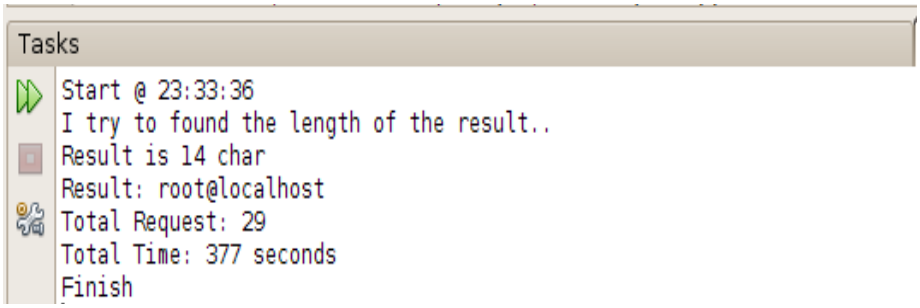
86 => 3
87 => 4
88 => 5
...
121 => 38
122 => 39

Bu iyiye işaret çünkü ilk olarak hangi aralıkta olduğunu tespit etmek için aldığım “sinyal” 3 saniyeydi ve şimdide görüyorum ki maksimum 39 saniye daha bekleyerek toplamda 42 saniyede 86-122 aralığındaki bir karakteri bulabiliyorum. 48-85 aralığındaki karakterler için biraz daha şanslıyım çünkü karakter uzunluğu 85'ten büyük değilse ekstradan 3 saniye beklemeye gerek kalmayacak.

Burada şöyle bir şey yaptım MOD((ASCII_VALUE+38),83). Örneğin 48 + 38 = 86 olacağından mod 83 işleminden 3 sonucu geldi kısacası 86-122 aralığına benzetmeye çalıştım.

48 => 3
49 => 4
50 => 5
...
84 => 39
85 => 40

Test:



14 karakterden oluşan veritabanı kullanıcı adına toplamda 29 istek ile 377 saniyede yani 6 dakika 17 saniyede ulaştım. Tabi ben PoC'u php ile yazdım, multi thread destekli bir dil ile (örn: Java, python, ruby vs..) yazılacak bir araç ile maksimum 42 saniyede tüm veriyi elde edebilirsiniz.