

btrisk

BİLGİ GÜVENLİĞİ VE BT YÖNETİŞİM HİZMETLERİ

BLIND SQL INJECTION SALDIRILARI

Emre Karadeniz

OSCP



İçindekiler

I. Blind SQL Injection (Content Based)	2
II. Blind SQL Injection (Content-Based) Örneđi	3
III. Blind SQL Injection (Time-Based).....	16

I. Blind SQL Injection (Content Based)

SQL Injection açıklığı UNION tekniđi ile farklı tablolardan veri sızdırmamıza imkan vermiyorsa, ancak injection'ı yaptığımız noktada AND operatörü sonrasında oluşturacağımız TRUE ve FALSE koşullarına karşı farklı yanıtlar üretiyorsa, o halde bir şansımız daha var demektir.

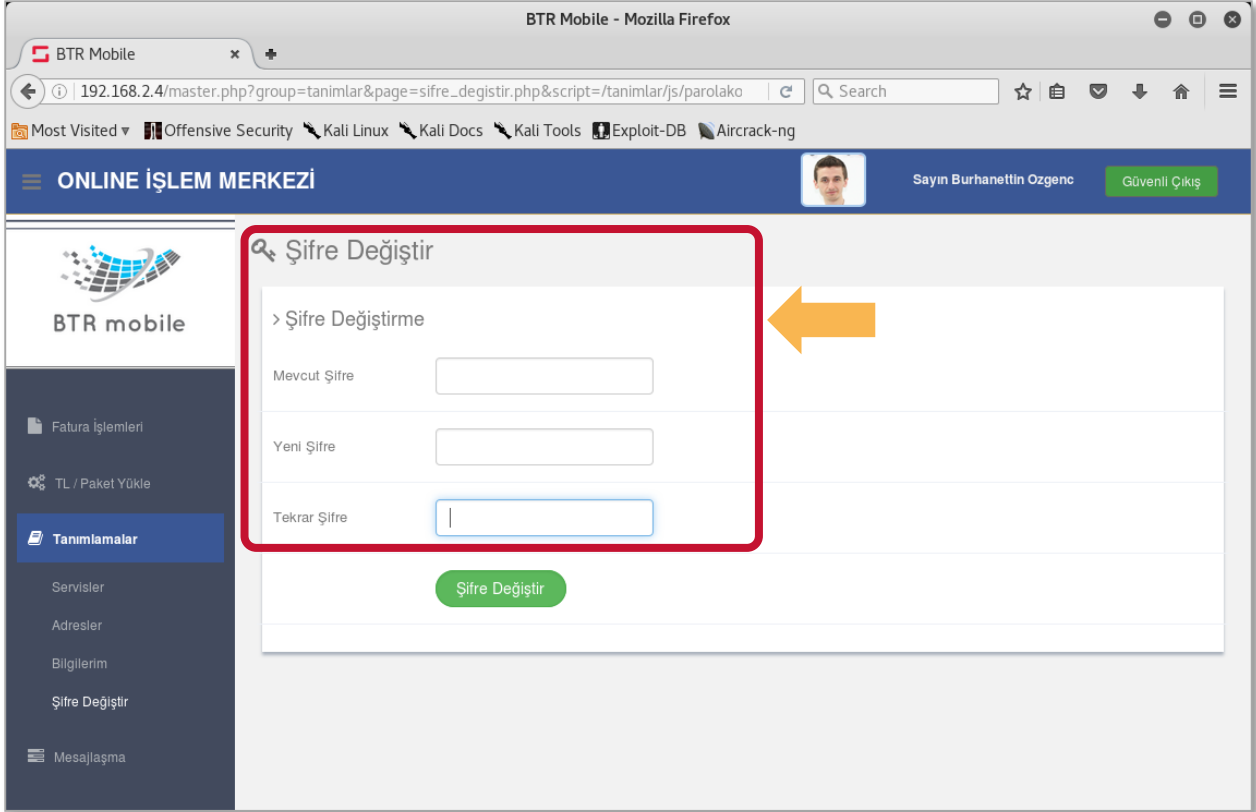
UNION tabanlı veya doğrudan yanıt doğuran yöntemlere nazaran çok daha fazla sayıda istek göndermemiz gerekecektir.

Blind SQL Injection yönteminde genellikle veriler harf harf belirlenir ve her bir harfin değerini bulmak için binary search yöntemi kullanılır. Örneđin 0-100 arasında bulunması gereken 83 değerini bulmak için önce değerin 50'den büyük olup olmadığını sorgularsınız, eđer büyükse bu sefer 75'ten büyük olup olmadığını sorgularsınız, bundan da büyükse 88'den büyük olup olmadığını sorgularsınız. Bu sayıdan büyük olmadığı için bu defa 75 ve 88 sayılarının ortasında bulunan 81'den büyük olup olmadığını sorgularsınız ve bu sayı bulununcaya kadar devam eder.

II. Blind SQL Injection (Content-Based) Örneđi

Content Base Blind SQL Injection örneđi için uygulamamızın Şifre Deđiştirme fonksiyonundan yararlanmaya çalışacağız.

Form alanlarından görünür olanları Mevcut Şifre ve Yeni Şifre alanları

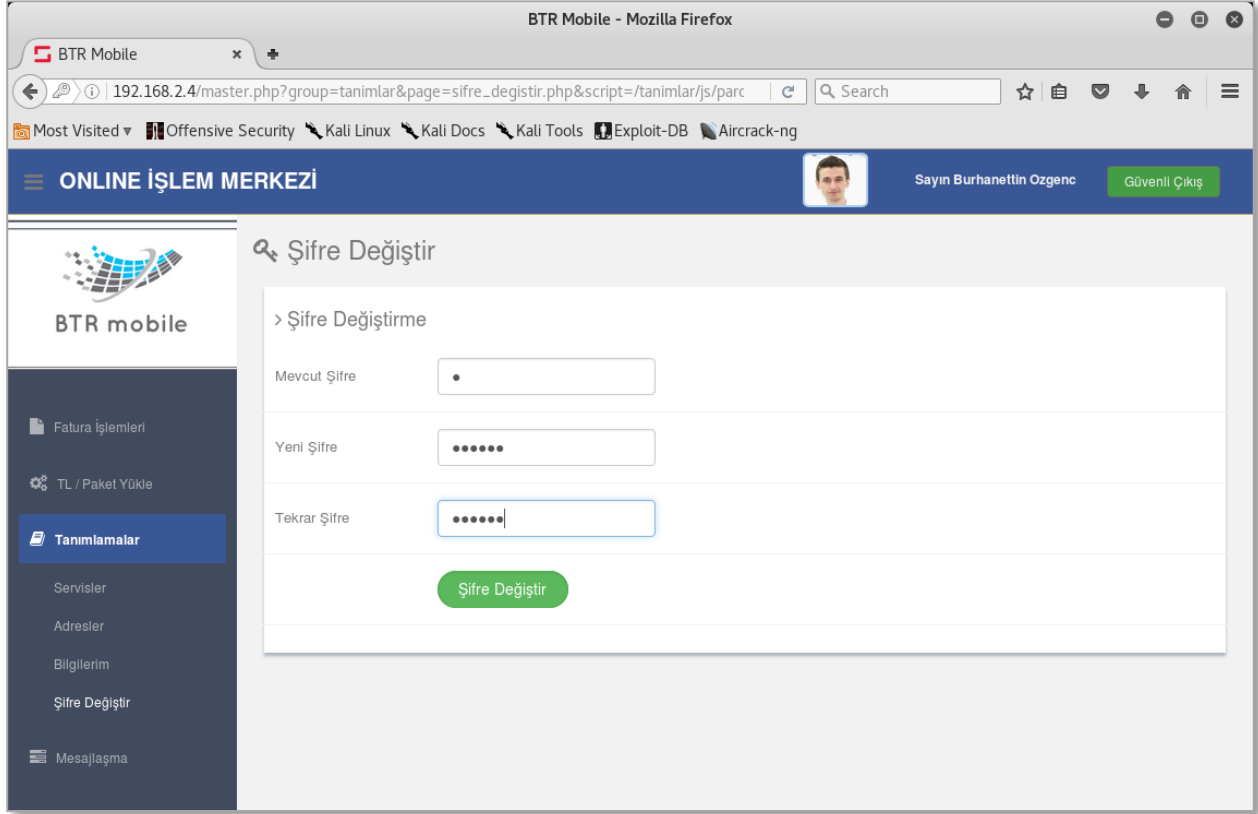


The screenshot shows a web browser window titled "BTR Mobile - Mozilla Firefox". The address bar displays the URL "192.168.2.4/master.php?group=tanimlar&page=sifre_degistir.php&script=/tanimlar/js/parolako". The page header includes "ONLINE İŞLEM MERKEZİ" and a user profile for "Sayın Burhanettin Ozgenc" with a "Güvenli Çıkış" button. The main content area is titled "Şifre Deđiştir" and contains a form with the following fields:

- Mevcut Şifre
- Yeni Şifre
- Tekrar Şifre

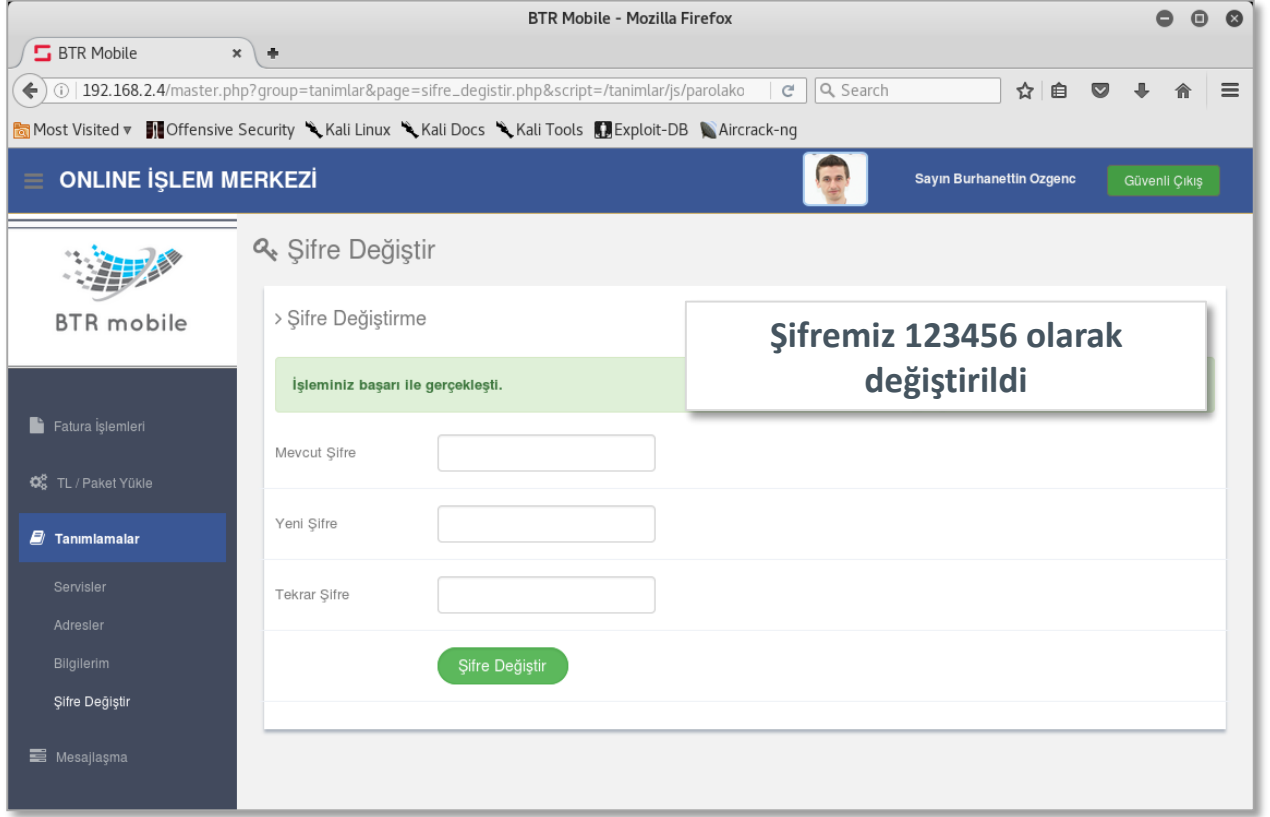
A red box highlights the "Mevcut Şifre" and "Yeni Şifre" fields, and a yellow arrow points to the "Yeni Şifre" field. A green "Şifre Deđiştir" button is located below the form.

Bundan sonraki adımlarda şifre deđişiklik işleminde mevcut şifre hatalı durumuna düşmemek için (uygulamanın yeni şifrenin eski şifre ile aynı olup olmadığını kontrol etmemesinden de faydalanarak) şifremizi 123456 olarak deđiştiriyoruz.



The screenshot shows a web browser window titled "BTR Mobile - Mozilla Firefox". The address bar displays the URL "192.168.2.4/master.php?group=tanimlar&page=sifre_degistir.php&script=/tanimlar/js/parc". The browser's most visited sites include "Offensive Security", "Kali Linux", "Kali Docs", "Kali Tools", "Exploit-DB", and "Aircrack-ng". The page header features the "ONLINE İŞLEM MERKEZİ" logo, a user profile picture, the name "Sayın Burhanettin Ozgenc", and a "Güvenli Çıkış" button. The main content area is titled "Şifre Deđiştir" and contains a form with three input fields: "Mevcut Şifre" (Current Password), "Yeni Şifre" (New Password), and "Tekrar Şifre" (Repeat Password). A green "Şifre Deđiştir" button is located below the form. The left sidebar menu includes options like "Fatura İşlemleri", "TL / Paket Yükle", "Tanımlamalar", "Servisler", "Adresler", "Bilgilerim", "Şifre Deđiştir", and "Mesajlaşma".

Şifremiz 123456 olarak deđiştirildi

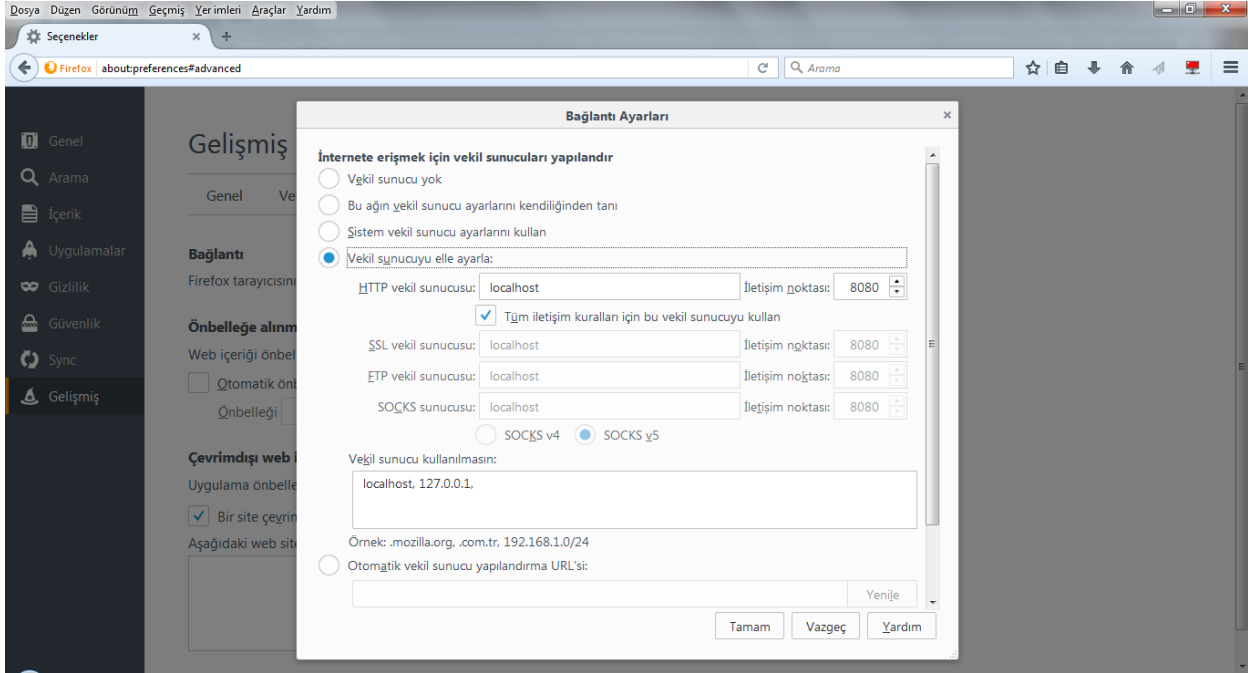


The screenshot displays the BTR Mobile web application interface. The browser title is "BTR Mobile - Mozilla Firefox". The address bar shows the URL: "192.168.2.4/master.php?group=tanimlar&page=sifre_degistir.php&script=/tanimlar/js/parolako". The page header includes "ONLINE İŞLEM MERKEZİ" and a user profile for "Sayın Burhanettin Ozgenc" with a "Güvenli Çıkış" button. The main content area is titled "Şifre Deđiştir" and shows a confirmation message: "İşleminiz başarı ile gerçekleşti." and "Şifremiz 123456 olarak deđiştirildi". Below the message are input fields for "Mevcut Şifre", "Yeni Şifre", and "Tekrar Şifre", along with a "Şifre Deđiştir" button. A sidebar on the left contains navigation options: "Fatura İşlemleri", "TL / Paket Yükle", "Tanımlamalar", "Servisler", "Adresler", "Bilgilerim", "Şifre Deđiştir", and "Mesajlaşma".

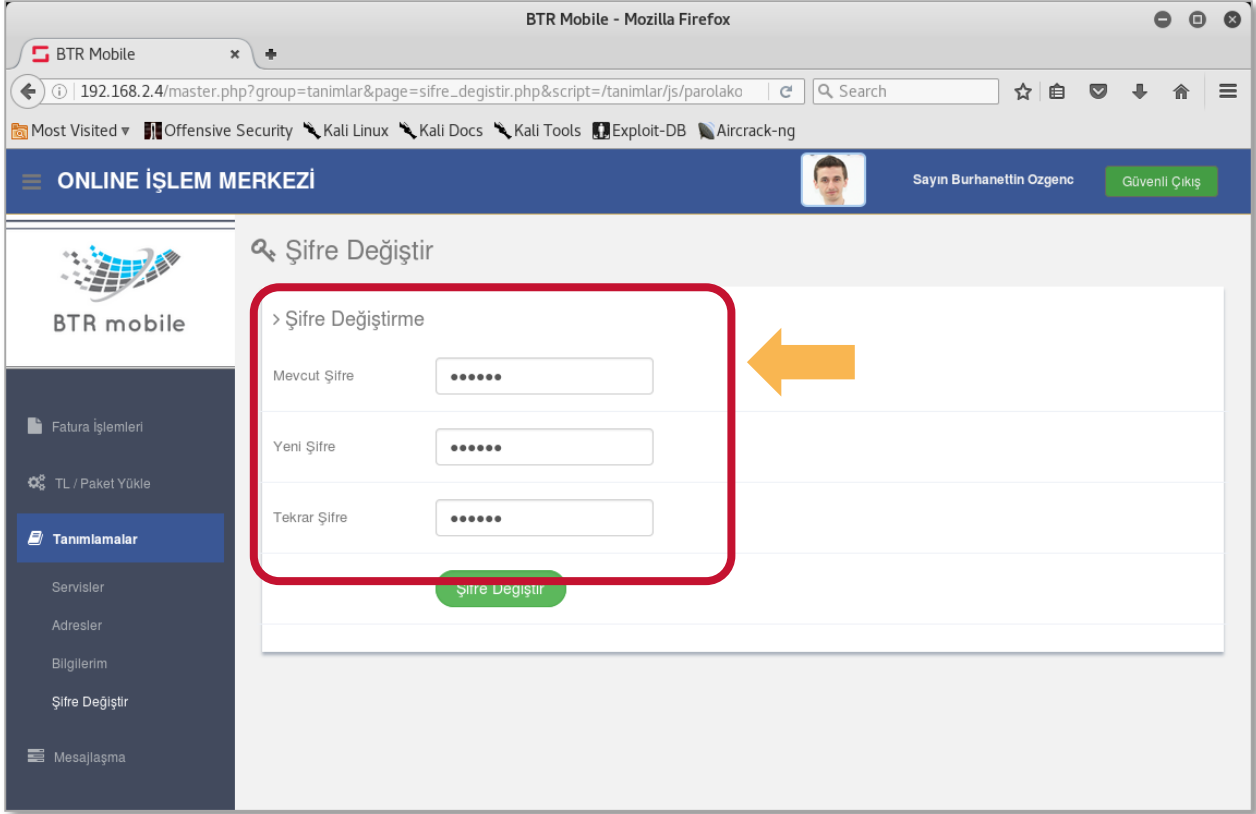
Ancak formu gönderdiğimizde form alanlarının tamamı görünenlerden mi ibaret, inceleyelim.

Bu işlem için bir atak proxy olan Burp Suite aracından faydalanacağız. Bu araç ile trafiđi keserek giden ve gelen isteklerdeki form alanlarını inceleme fırsatımız olacak.

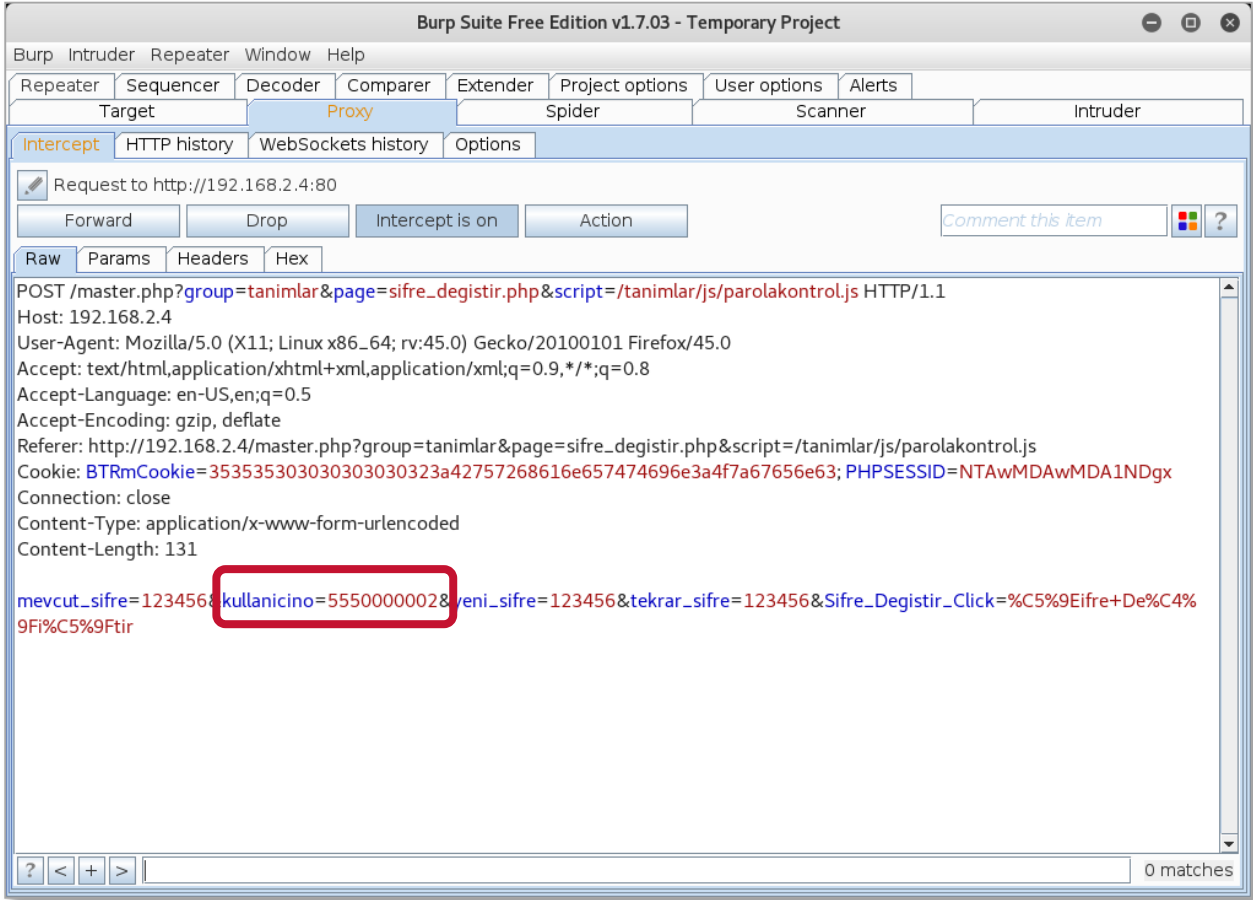
Tarayıcıımızdan proxy ayarımızı yaptıktan sonra istekler Burp üzerinden geçerek sunucuya iletilecektir. Bu işlem için internet tarayıcıımızın proxy ayarı aşağıdaki şekilde yapılmalıdır.



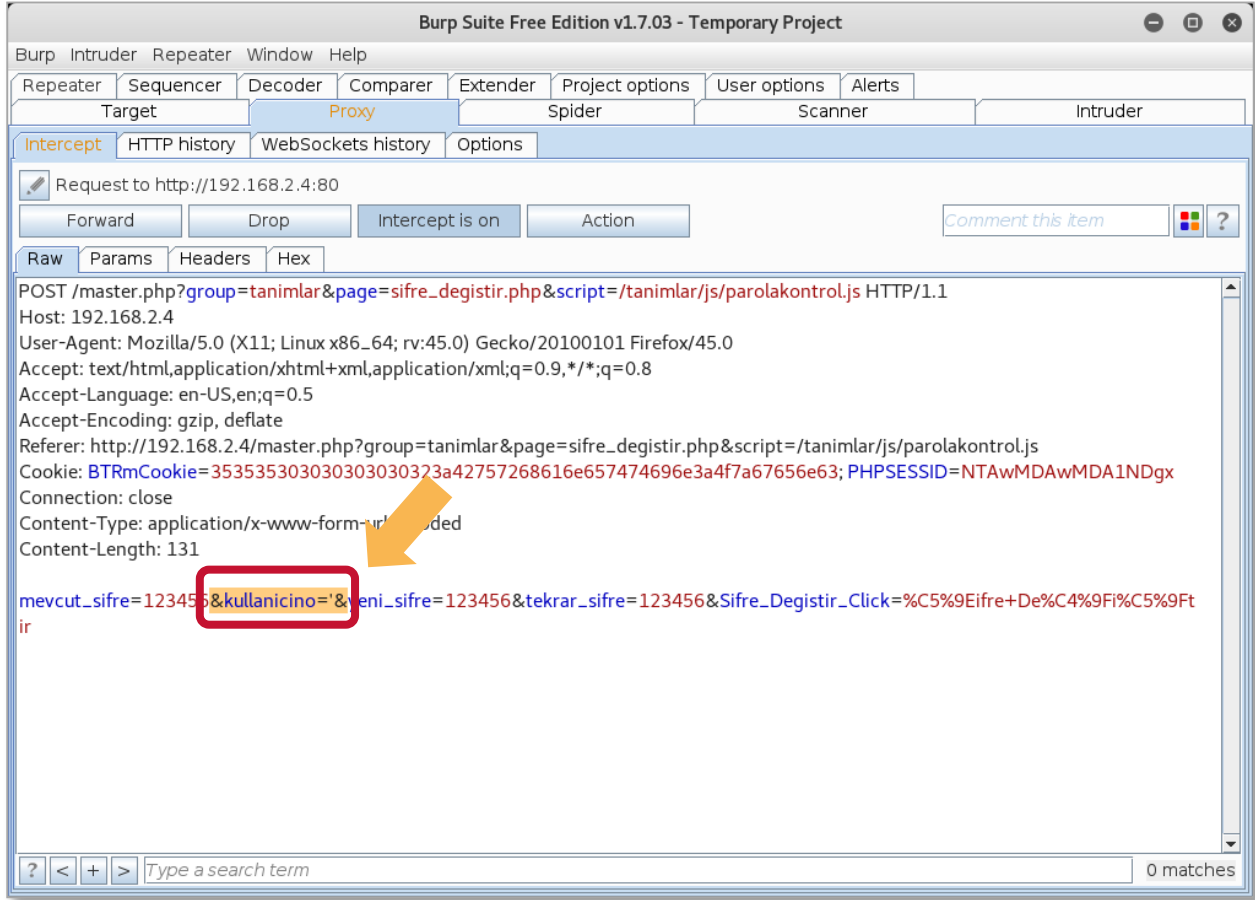
Bu adımdan sonra şifre deđiştirme ekranında tekrar işlem yapalım ve Burp üzerinde bu isteđi inceleyelim.



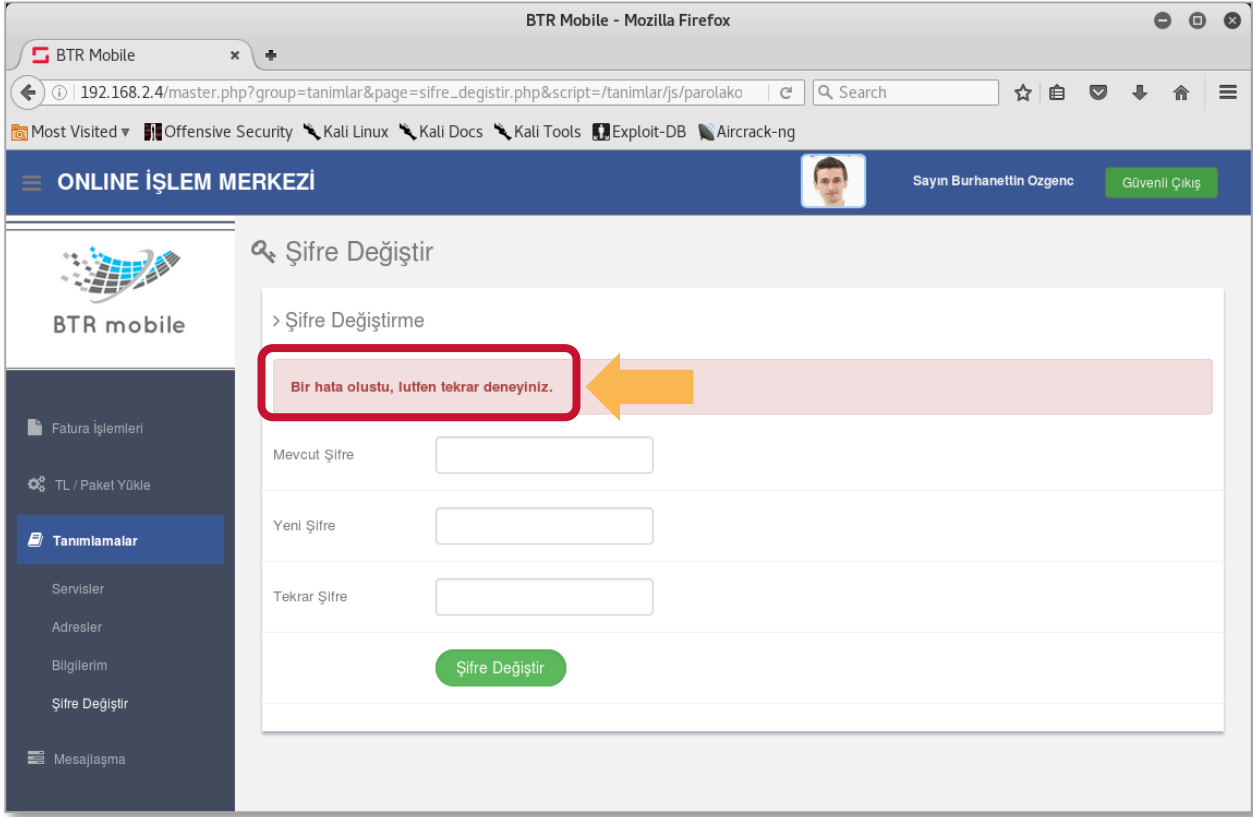
Bu aşamada, giden parametreler arasında bir de "kullanicino" parametresinin bulunduđunu görüyoruz.



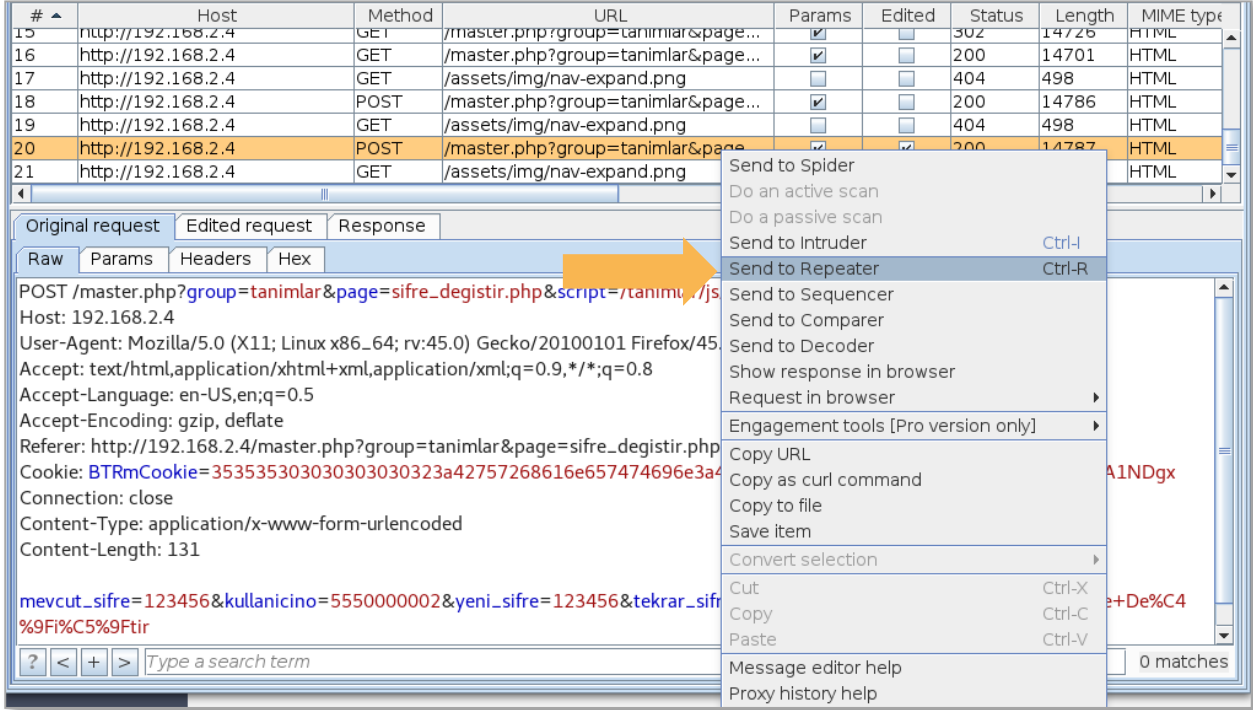
Bu parametreyi tırnak işareti değeri ile ilettiğimizde ne olacağını gözlemleyelim. Bu sayede sisteme hata aldırıp aldırılmadığı gözlemlenebilir.



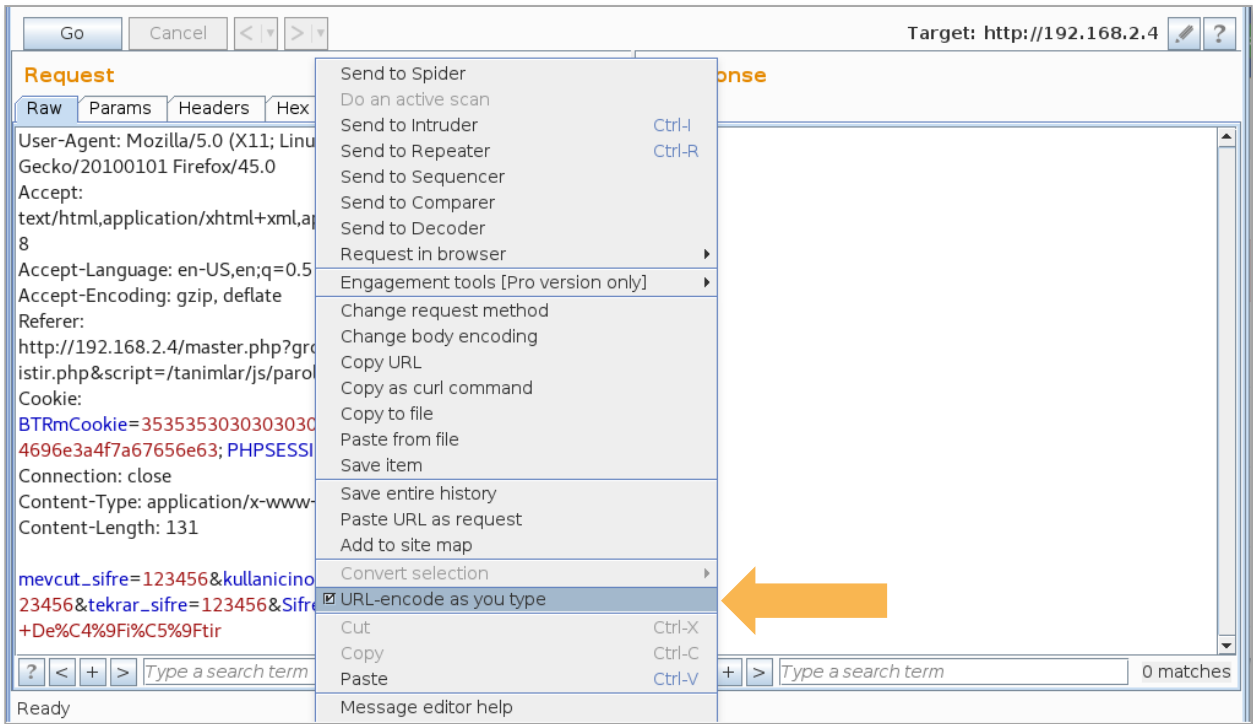
Hatalı bir durum oluştu, ancak uygulama hatayı ele aldı. SQL Injection olma ihtimali yüksek, ancak hata mesajı olmadığı için farklı bir neden de olabilir. Elde ettiğimiz bu ipucuna biraz daha odaklanalım.



Bu isteğin üzerinde daha rahat çalışabilmek için isteği Burp'ün "Repeater" modülüne atalım. Bu modül ile aynı istek üzerinde değişiklik yapıp sunucuya göndererek yanıtları inceleyebiliriz. İstek üzerinde sağ tıklayıp açılan pencerede "Send To Repeater" tıklamamız yeterli olacaktır.

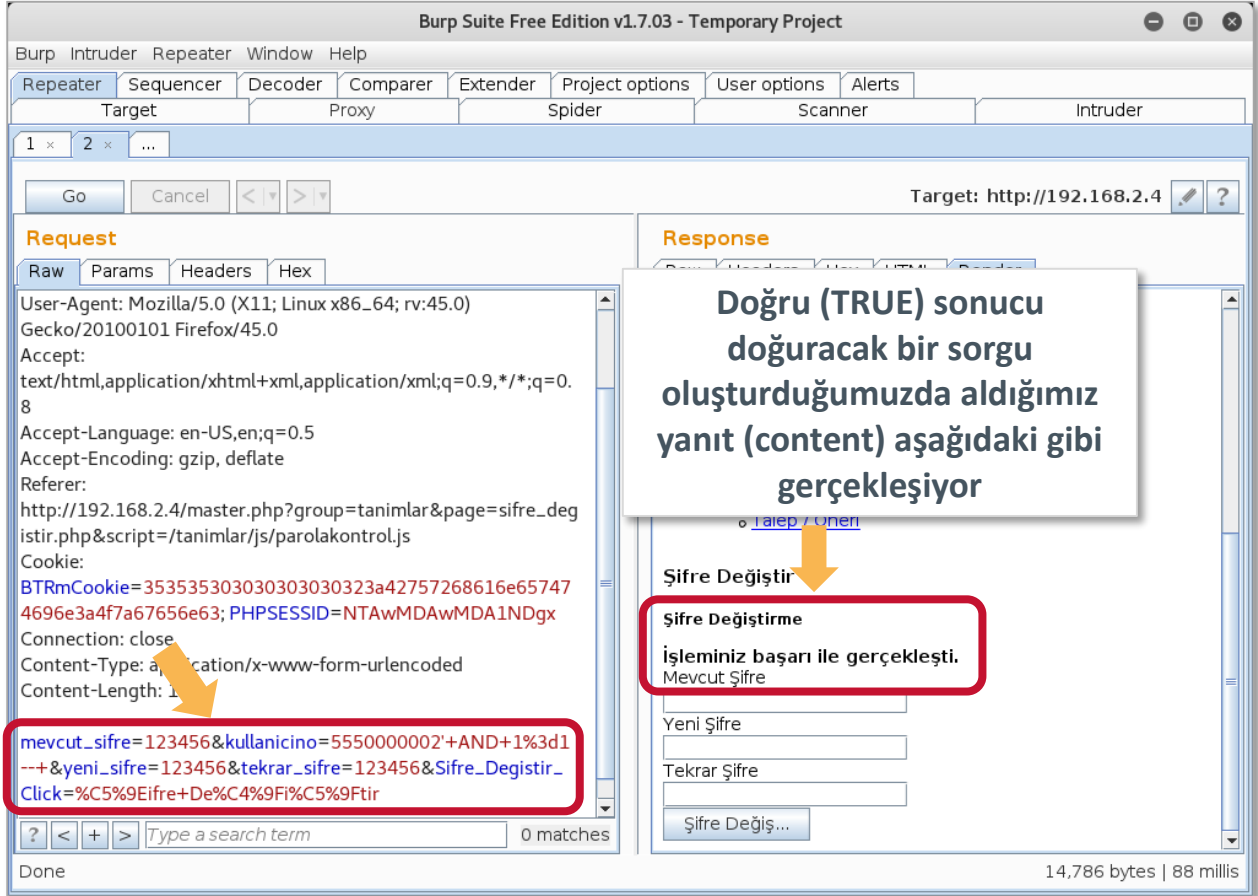


İstek parametreleri üzerinde değişiklik yaparken URL kodlama kurallarına uygunsuzluğu önlemek amacıyla Burp'ün aşağıdaki yardımcı fonksiyonunu aktif hale getirelim. Böylece yazdığımız özel karakterler Burp tarafında URL kodlama yöntemiyle yazılacaklar.



"555000002'+AND+1%3d1--+" injection payloadu ile "kullanicino" parametresinin string tipinde olduğu varsayımında bulunarak ve geçerli bir kullanıcı kodu kullanarak doğru (TRUE) sonuç doğuracak bir sorgu oluşturmayı hedefliyoruz.

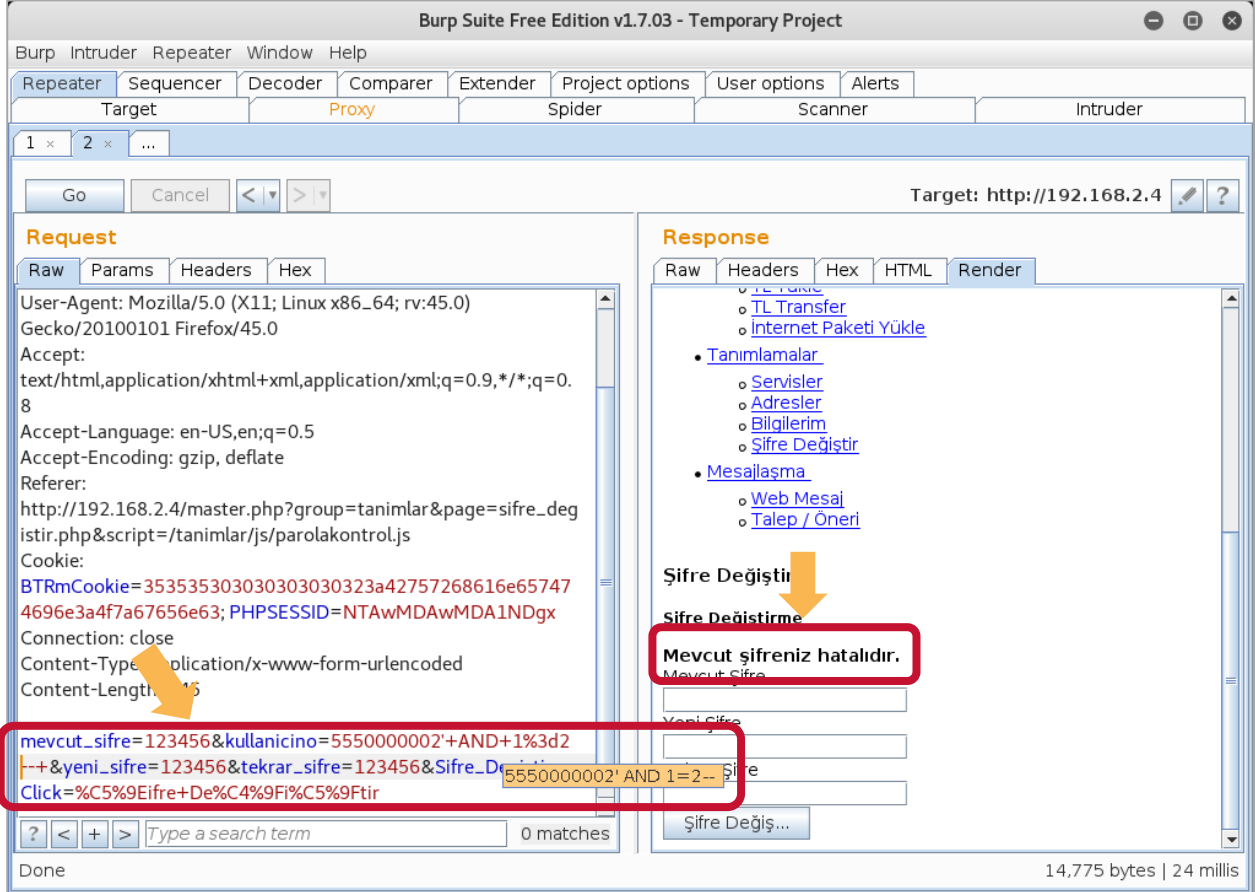
Payload eklediğimiz isteği sunucuya gönderdiğimizde aşağıdaki şekilde (TRUE) sonucunu elde edebiliyoruz.



The screenshot displays the Burp Suite interface with a request and response. The request is a GET request to the target URL with the following parameters: `mevcut_sifre=123456&kullanicino=555000002'+AND+1%3d1--+'¥i_sifre=123456&tekrar_sifre=123456&Sifre_Degistir_Click=%C5%9Eifre+De%C4%9Fi%C5%9Ftir`. The response is a 200 OK status with a content type of text/html and a success message: "İşleminiz başarı ile gerçekleşti." (Your operation was successfully completed). A red box highlights the payload in the request, and a yellow arrow points to the success message in the response. A text box above the response states: "Doğru (TRUE) sonucu doğuracak bir sorgu oluşturduğumuzda aldığımız yanıt (content) aşağıdaki gibi gerçekleşiyor" (When we create a query that will give a correct (TRUE) result, the response (content) we receive is as follows).

Koşulun yanlış (FALSE) olacağı şekilde bir düzenleme yaptığımızda ise aldığımız yanıt (content) doğru koşuldan farklı olmaktadır. İşte bu aradaki fark bize veritabanına yönelik çok sayıda sorgu yaparak veri sızdırma imkanı sağlayacaktır

“555000002'+AND+1%3d2--+” payload olarak kullanıp tekrar istek yapalım ve yanıtın doğru (TRUE) koşuldan farklı olduğuna dikkat edelim.



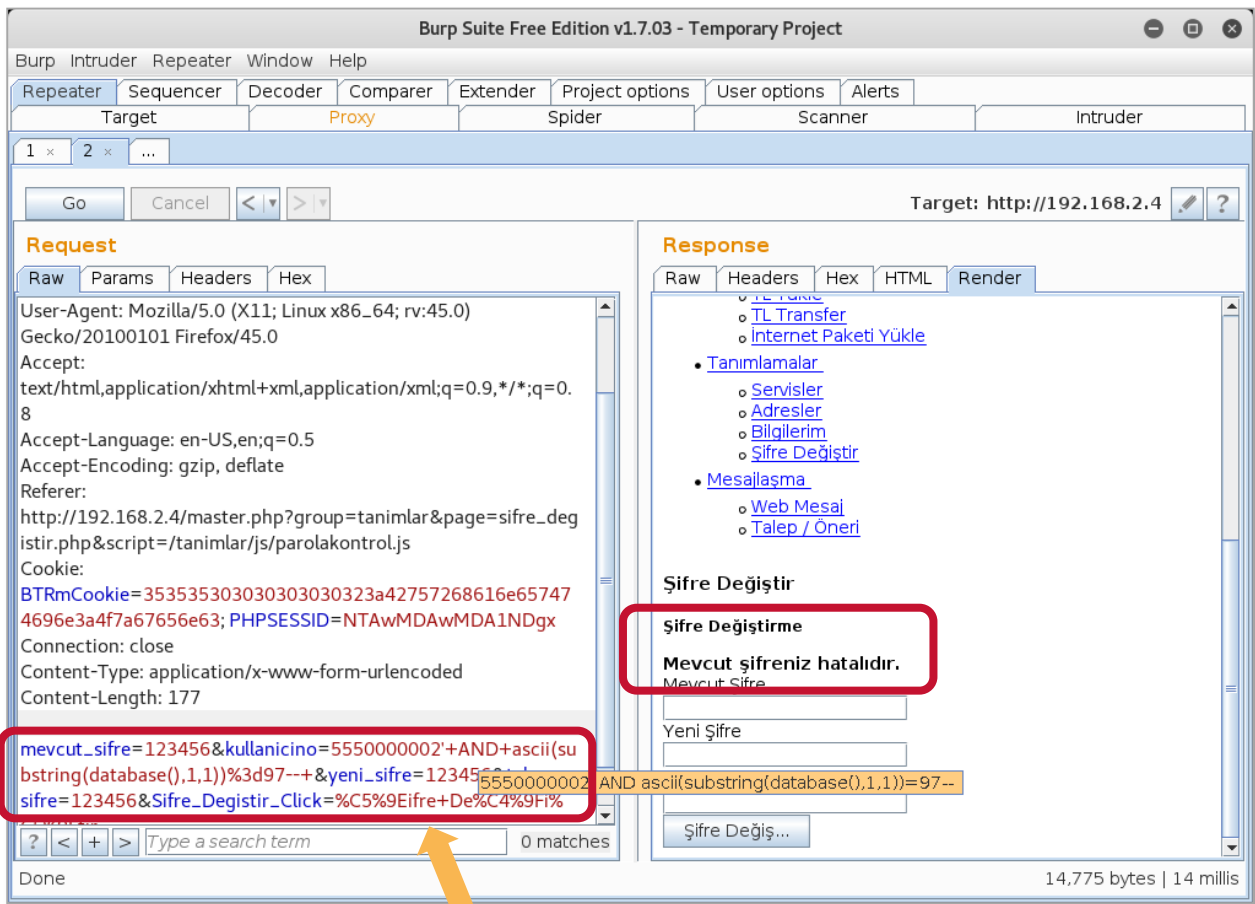
TRUE ve FALSE durumlarından emin olduktan sonra biraz daha kontrollü istekler gerçekleştirebiliriz.

İlk testimizde MySQL database() fonksiyonunun döndürdüğü değerın ilk karakterinin "a" (ASCII 97) olup olmadığını test ediyoruz. Test için aşağıdaki payloadu kullanabiliriz;

"555000002'+AND+ascii(substring(database()),1,1))%3d97--+"

Kullandığımız bu payload içerisindeki substring fonksiyonu yaptığımız sorgudaki parametreler sayesinde bize database() fonksiyonundaki ilk string ifadenin ilk değerini döndürecektir.

Dönen yanıt FALSE durumunda oluşan yanıt olduğundan ilk karakterin "a" olmadığı sonucuna varabiliriz.



mevcut_sifre=123456&kullanicino=555000002'+AND+ascii(substring(database()),1,1))%3d97--+¥i_sifre=123456&tekrar_

III. Blind SQL Injection (Time-Based)

Eđer SQL injection yapabilmemize ve TRUE ve FALSE koşullar oluşturabilmemize rağmen yanıtlarda bir farklılık gözlenmiyorsa o zaman TRUE veya FALSE koşullar oluşturduğunda yanıtın dönme hızını etkileyerek yine veri sızdırma şansımız olabilir.

Hem çok sayıda istek göndermemiz hem de zaman farkına dayanan bir yöntem olduğu için çok daha fazla süre beklememiz gerekecektir.

Time based SQL injection yönteminde sonucun hata doğurup doğurmadığı önemli değildir, ancak belli bir koşul gerçekleştiğinde sunucunun daha geç yanıt vermesine dayanarak veriler tahmin edilir.

MySQL'e özel bir Time-Based SQL injection sorgusu;

```
http://www.site.com/news.php?id=5 and 1=IF(ascii(substring((SELECT concat(username,0x3a,password) from users limit 0,1),1,1))>80, sleep(15),1)
```

Veritabanı sunucularının TIME-BASED BLIND SQL INJECTION saldırılarında kullanılabilecek fonksiyonlarına örnekler:

MSSQL

- **WAITFOR DELAY '0:0:10'** (10 sn. bekle)

MySQL

- **sleep(5000)** (sleep fonksiyonu MySQL 5.0.12 ve sonraki versiyonlarında kullanılabilir, 5.000 milisaniye bekle anlamına gelir)
- **benchmark(50000, SHA1('test'))** (50.000 defa 'test' verisinin SHA1 hash değerini hesapla, bunun dışında CPU zamanını alabilecek farklı MySQL fonksiyonları da bulunmaktadır)