CHROOT & WEB SERVER KURULUMU

- Bu döküman da "Chroot + Apache + Php + Mysql Kurulum"u anlatılacaktır. Kurulan paketlerin sürümleri/versiyonları belirtilmemiştir. Döküman da belirtilen kurulum şemasını aynen uygulayabilir veya paket (ekleme/çıkarma) yapabilirsiniz, dilerseniz derleme seçeneklerinde değişiklikte yapabilirsiniz. Bu sizin inisiyatifinize kalmıştır. Şahısların temel UNIX bilgisi olduğunu varsayıyorum. Bazı detaylar açıklanacak bazıları da kişiye bırakılacaktır.

- Kurulum şeması
  1. Mysql kurulumu, basit birkaç ayar
  2. Apache kurulumu, gerekli dosyaların düzenlenmesi
  3. Chroot ortam yaratılması ve Mysql'i chroot ortama taşımak
  4. Apache'yi chroot ortama taşımak

MYSQL KURULUMU

- Öncelikle belirtmek istiyorum, dökümanın tamamını okumadan kesinlikle kuruluma geçmeyin. "**mysql**" chroot edilecekse 2 tür kurulum yöntemi kullanılacaktır. İsterseniz yavaş yavaş başlayalım, "**mysql**" kullanıcısını ve grubu oluşturalım, yükleme yapacağımız dizini belirleyelim, daha sonra kaynak kodumuzu derleyip install edelim.

```
#  pw groupadd mysql -g 88
#  pw useradd mysql -u 88 -g mysql -d  /dev/null -s /sbin/nologin \
   -c "MysqlServer"

#  umask 022
#  mkdir –p /usr/local/mysql/{data,tmp}
#  chown –R root:sys /usr/local/mysql
#  chown –R mysql:mysql /usr/local/mysql/{data,tmp}

#  cd /tmp
#  tar –zxvf mysql-xxx.tar.gz
#  mv  mysql-xxx  mysql
#  cd mysql
```

- *Burada 2 tür kurulumu birbirinden ayırt edecez. Şayet I. tür kurulumu uygulamak isterseniz "configure" parametresine bağlı olarak "--with-mysqld-ldflags" değişkenini kullanmalıyız. Eğer II. tür kurulumu uygulamak istersek bu değişkene ihtiyacımız olmayacak.*

NOT : *II. tür kurulum seçilirse; "mysqld" dinamik kütüphane kullanacak. Mysql'i chroot ediyorken bu dosyaları chroot dizinine* ***install*** *etmemiz lazım. Bu konuya ileride değinecez.*

```
#  ./configure --prefix=/usr/local/mysql/  --localstatedir=/usr/local/mysql/data/  \
     --with-mysqld-user=mysql  --without-debug  --without-readline  \
     --without-bench  --with-libwrap  --with-low-memory  --enable-thread-safe-client  \
     --with-unix-socket-path=/usr/local/mysql/tmp/mysql.sock  \
     --with-mysqld-ldflags=-all-static --enable-assembler  \
#  make
#  make install
```

- Kurulumu tamamlamak için standart db'yi yükleyelim.

```
#  /tmp/mysql/scripts/mysql_install_db
#  cp /tmp/mysql/support/my-medium.cnf  /usr/local/mysql/data/my.cnf
#  chmod 644 /usr/local/mysql/data/my.cnf
#  chown root:wheel /usr/local/mysql/data/my.cnf
```

NOT : Olası hatalara karşı "data" ve "tmp" dizinlerinin erişim haklarını kontrol edip, gerekirse **chown** ile dosya sahiplerini yeniden oluşturup tekrar "mysql_install_db" çalıştırılmalıdır.

- Sunucumuzu çalıştıralım ve test edelim.

```
#  /usr/local/mysql/libexec/mysqld  --user=mysql  &
```

- Mysql çalışmaya başladığı ilk anda yapmamız gereken; Mysql'in **root** kullanıcı şifresini oluşturmak olmalıdır.

```
#   /usr/local/mysql/bin/mysqladmin –u root password 'xxxxx'
#   /usr/local/mysql/bin/mysql –u root –p
Enter password:   (Burada yukarıda kullandığımız şifremizi isteyecek bizden)

mysql>   (Bu prompt ile sunucumuza ilk bağlantımızı sağlamış olduk.)
mysql>   exit
```

- Mysql'in 3306 nolu portta dinlemeye geçip geçmediğini kontrol edelim.

```
#  sockstat  -4 | grep 3306

mysql    mysqld   45711    5 tcp4   *:3306               *:*
```

- Gördüğünüz gibi **mysql** network'umuzda dinlemeye geçmiş durumda, ama şunu belirtmek istiyorum EĞER sunucuya uzaktan erişim gerekmiyorsa ve/veya işlem yapmayacak iseniz port dinlemeye kesinlikle açmayın. Mysql'in network'u dinlemesini istemiyorsanız, **my.cnf** dosyanızda **[mysqld]** tagının altına ;

```
#  vi /usr/local/mysql/data/my.cnf
[mysqld]
....
skip-networking
...
```

satırını ekleyin. Mysql'i yeniden başlatın. Uzaktan erişim kısıtlanacaktır!!

```
#  kill `cat /usr/local/mysql/data/xxx,pid`
#  /usr/local/mysql/libexec/mysqld --user=mysql &
```

- Sockstat komutunu tekrar kullanarak kontrol edebilirsiniz.

```
#  sockstat –4 | grep 3306
```

APACHE & SSL & PHP KURULUMU

- Apache kurulumu çok çeşitli olacaktır. İsteğe göre özel derleme seçenekleri kullanılabilir. İşimizi görecek kadar modulu aktif edecez. Döküman da; **Apache** sunucusunu ssl destekli kuracağımız için **Openssl** ve **mod_ssl** desteğini de verecez. Sisteminize göre uygun Apache & Openssl & mod_ssl & mm kaynak dosyalarını indirin. Seçtiğiniz paketler birbirine uyumluluk göstermelidir !!! Paketleri resmi sitelerinden seçmenizi ve indirmenizi tavsiye ederim. FreeBSD üzerinde bu işlemleri yapıyorsanız, port ağacından paketlerin ihtiyacı olan diğer paketleri listeleyebilir ve sürümlerini görebilirsiniz.
(Kararlı sürümleri indirmeye özen gösterin, varsa yamalarını geçin..)

- **apache** grubu ve kullanıcısını oluşturalım, dizinimizi belirleyelim.

```
#  pw groupadd apache -g 89
#  pw useradd apache -u 89  -g apache -d /dev/null -s /sbin/nologin  -c "ApacheServer"

#  umask 022
#  mkdir /usr/local/apache
```

- İndirdiğimiz kaynak dosyaları birbir derleyip, sisteme yükleyelim.

```
#  cd /tmp
#  tar –xzvf openssl-x.x.x.tar.gz
#  mv openssl-x.x.x  openssl
#  cd openssl
#  ./config  &&  make  &&  make test  && make install


#  cd /tmp
#  tar –zxvf  mm-x.x.x.tar.gz
#  mv mm-x.x.x  mm


#  tar –zxvf  apache-x.x.x.tar.gz
#  mv apache-x.x.x  apache


#  tar –zxvf  mod_ssl-x.x.x-x.x.x.tar.gz
#  mv mod_ssl-x.x.x-x.x.x  mod_ssl


#  cd mm
#  ./configure --prefix=/usr/local  && make && make install


#  cd ../mod_ssl
#  ./configure --with-apache=../apache  --with-ssl=../openssl \
   --with-mm=/usr/local --prefix=/usr/local/apache
```

- Apache'yi derlemeden önce kaynak kodda yer alan **src/include/httpd.h** dosyasıyla biraz oynayalım. 3 değeri de ***Unknown*** yapalım.

```
#  cd ../apache
#  vi src/include/httpd.h
......
......
#define SERVER_BASEVENDOR    "Apache Group"      ->  "Unknown"
#define SERVER_BASEPRODUCT   "Apache"            ->  "Unknown"
#define SERVER_BASEREVISION  "1.3.37"            ->  "Unknown"


#  ./configure --prefix=/usr/local/apache/ --enable-module=ssl --enable-module=auth_db \
   --enable-module=rewrite --enable-module=most  --enable-module=define  \
   --enable-module=so --disable-module=auth_dbm --disable-module=cgi  \
   --disable-module=imap --disable-module=proxy --disable-module=auth_anon \
   --disable-module=userdir --disable-module=asis --disable-module=usertrack

#  make
#  make certificate      (Yonergeleri izleyerek sertifikanızı hazırlayın)
#  make install
```

- Yukarıda disable çektiğim bazı modûller sizin işinize yarayabilir, kurulum bu aşamada sizin inisiyatifinize kalmıştır. Varolan modûl desteğini görmek için aşağıda ki komut çalıştırılmalıdır.

```
#  cd /usr/local/apache/bin
#  ./httpd –l


#  cat /usr/local/apache/conf/httpd.conf          (Ornek httpd.conf dosyasi)


# ================================================
# Basic Settings
# ================================================

Listen  sunucu_ip:80
User apache
Group apache
ServerAdmin webmaster@example.com
UseCanonicalName Off
ServerSignature Off
HostnameLookups Off
ServerTokens Prod
ServerRoot "/usr/local/apache"
DocumentRoot "/usr/local/apache/example.com"
PidFile /usr/local/apache/logs/httpd.pid
ScoreBoardFile /usr/local/apache/logs/httpd.scoreboard
```

```
# ==============================================
# Basic Performance settings
# ==============================================
HostnameLookups Off
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
<IfModule prefork.c>
   MinSpareServers 5
   MaxSpareServers 10
   StartServers 5
   MaxClients 150
   MaxRequestsPerChild 0
</IfModule>
# ==============================================
# SSL Settings
# ==============================================
<IfDefine SSL>
Listen 80
Listen 443
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl    .crl
</IfDefine>
<IfModule mod_ssl.c>
SSLPassPhraseDialog  builtin
SSLSessionCache         dbm:/usr/local/apache/logs/ssl_scache
SSLSessionCacheTimeout  300
SSLMutex  file:/usr/local/apache/logs/ssl_mutex
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
SSLLog      /usr/local/apache/logs/ssl_engine_log
SSLLogLevel info
</IfModule>
# ==============================================
# Access control
# ==============================================
<Directory />
   Options None
   AllowOverride None
   Order deny,allow
   Deny from all
</Directory>
<Directory "/usr/local/apache/example.com">
   Order allow,deny
   Allow from all
</Directory>
```

```
# ==================================================
# MIME encoding
# ==================================================
<IfModule mod_mime.c>
    TypesConfig /usr/local/apache/conf/mime.types
</IfModule>
DefaultType text/plain
<IfModule mod_mime.c>
    AddEncoding x-compress .Z
    AddEncoding x-gzip .gz .tgz
    AddType application/x-compress .Z
    AddType application/x-gzip .gz .tgz
    AddType application/x-tar .tgz
</IfModule>
# ==================================================
# Logs
# ==================================================
LogLevel warn
ErrorLog  /usr/local/apache/logs/error_log
CustomLog  /usr/local/apache/logs/access_log combined
# ==================================================
# Virtual hosts
# ==================================================
NameVirtualHost  sunucu_ip:443

<IfDefine SSL>
<VirtualHost  sunucu_ip:443>
DocumentRoot "/usr/local/apache/example.com"
ServerName  https://www.example.com
ServerAlias   https://www2.example.com
ErrorLog /usr/local/apache/logs/error_log
TransferLog /usr/local/apache/logs/access_log
SSLEngine on

---------------------------------  DIKKAT  --- Tek satirdir   ----------------------------------------
SSLCipherSuite
ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
---------------------------------  DIKKAT  --- Tek satirdir   ----------------------------------------

SSLCertificateFile    /usr/local/apache/conf/ssl.crt/server.crt
SSLCertificateKeyFile   /usr/local/apache/conf/ssl.key/server.key

CustomLog /usr/local/apache/logs/ssl_request_log \
          "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
</VirtualHost>
</IfDefine>
# ==================================================
```

NOT : example.com domain'i yerine siz kendi domain adresinizi kullanın.

- Basit bir web sayfası hazırlayalım.

```
#  umask 022
#  mkdir -p /usr/local/apache/example.com
#  vi  /usr/local/apache/example.com/index.html

<html><head>
<title> Example </title>
</head>
<body> Apache SSL! </body>
</html>
```

- Apache için ayarladığımız httpd.conf dosyasını test edelim.

```
#  /usr/local/apache/bin/apachectl configtest
   Syntax OK

#  /usr/local/apache/bin/apachectl  start        (Apache'yi normal başlatır.)
#  /usr/local/apache/bin/apachectl  startssl     (Apache'yi SSL desteği ile başlatır.)

Server https://www.example.com:443 (RSA)
Enter pass phrase:                              (Sertifika oluştururken girdiğiniz şifre!!!)
```

- Sunucumuzla aynı ağda bulunan herhangi bir client'in tarayıcısı yardımıyla Apache'den web sayfasını çağıralım. Eğer ağınızda sağlam çalışan bir DNS çözümleyici yoksa sunucu ip'sini kullanın.

http://www.example.com -> http://sunucu_ip
https://www.example.com -> https://sunucu_ip

- https://www.example.com çağrılıyorken sertifika onayı geliyorsa tebrik ederim !!! Sertifika ile alakalı herhangi bir yönerge ekrana gelmediyse tarayıcınızın sertifika bilgilerini tuttuğu bölümden size ait sertifikaya bakınız. Hala sertifikanıza rastlayamadıysanız adımları tekrar kontrol edin.

- Sıra geldi PHP ayarlarına.. İlk önce php kaynak kodumuzu derliyoruz. Apache'de verdiğimiz dso desteğini burada PHP için kullanacaz.

```
#  cd /tmp/
#  tar –zxvf php-4.x.x.tar.gz
#  mv php-4.x.x  php
#  cd php

#  ./configure  --disable-versioning  --enable-memory-limit  --with-layout=GNU --with-xml  \
   --enable-tracks  --with-apxs=/usr/local/apache/bin/apxs  --with-mysql=/usr/local/mysql

#  make
#  make install
```

NOT : Php'yi derledikten sonra "make install" komutunu verdiğimizde bu şekilde bir hata ile karşılaşabilirsiniz.

apxs:Error: Activation failed for custom /usr/local/apache/conf/httpd.conf file.
apxs:Error: At least one `LoadModule' directive already has to exist.

- Bu hatayı düzeltmek için aşağıda ki php modul satırlarını "httpd.conf" dosyasına ekleyin, daha sonra tekrar "make install" yapın. Kurulum tamamlandıktan sonra "httpd.conf" dosyasında sizin eklediğinizden fazla ***"LoadModule php4_module"*** satırı var ise kaldırın. Bir tane olması yeterlidir. Fazladan olursa apache başlatılıyorken; hata döndürecektir.

```
LoadModule  php4_module                 libexec/libphp4.so
AddType     application/x-httpd-php     .php
AddType     application/x-httpd-php     .php3
AddType     application/x-httpd-php     .php4
AddType     application/x-httpd-php-source  .phps
```

- Ayriyeten uzantılar bakımından ".php" uzantısının da yorumlamasını isteyelim.

```
<IfModule mod_dir.c>
   DirectoryIndex  index.html  index.php  index.php3  index.php4
</IfModule>
```

- Kurulum sorunsuz bitti ise; ilk yapılacak şey **php.ini** dosyasını ilgili yere kopyalamak ve üzerinde bir kaç değişiklik yapmak olmalıdır. Daha fazla seçenek için PHP'nin resmi sitesinden yardım alabilirsiniz. Şimdilik bu bize yetecektir.

```
#  cp  /tmp/php/php.ini-recommended  /usr/local/etc/php.ini
#  vi  /usr/local/etc/php.ini

max_execution_time = 60;
file_uploads = Off
safe_mode = On
safe_mode_gid = On
```

- Sunucumuzu restart etmeden önce, bir tane de index.php dosyası yaratalım.

```
#  vi /usr/local/apache/example.com/index.php

<HTML>
<SCRIPT LANGUAGE="PHP">
        print ("Apache & SSL & PHP !");
</SCRIPT>
</HTML>
```

- Dosyamızı kaydedip çıkalım. Aynı network'deki bir makinenin tarayıcısından http://apache_sunucu_ip/index.php çağırmasını istediğimiz de bize php kodlarını yorumlayıp html çıktısı olarak vermelidir !

- Eğer buraya kadar sorunsuz geldi iseniz ve php'yi derlerken "--with-mysql" parametresini de kullandı iseniz şu anda "**Apache & SSL & Php & Mysql**" sunucunuz var demektir. Kurduğumuz sistemi test edelim biraz... Mysql sunucuda bir tane tablo açalim.

```
#  /usr/local/mysql/libexec/mysqld --user=mysql  &

#  /usr/local/mysql/bin/mysql –u root –p
Enter password:         (Root kullanıcısının şifresi)

mysql>  use test;
mysql>  create table uyeler (adi VARCHAR(30), soyadi VARCHAR(30), uye_no INT ) ;
mysql>  insert into uyeler (adi, soyadi, uye_no) VALUES ('Mehmet' , 'CELIK' , '1' );
mysql>  select  * from uyeler;

+------------+-----------+-----------+
| adi        | soyadi    | uye_no    |
+------------+-----------+-----------+
| Mehmet     | CELIK     |  1        |
+------------+-----------+-----------+
```

- Mysql'e ilk kaydımızı girmiş bulunuyoruz. Bu kaydı sorgulamak için bir tane de "**mysql.php**" yaratalım.

```
#  vi  /usr/local/apache/example.com/mysql.php

<HTML>
<TITLE> PHP – Mysql </TITLE>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-9">
<meta http-equiv="Content-type" content="text/html; charset=windows-1254">
</HEAD>
<BODY>
<?php

$veri_yolu = mysql_connect("localhost", "root", "xxxxxx");
if ( ! $veri_yolu ) die ("MySQL sunucu bağlantı kurulamıyor!");

mysql_select_db("test" , $veri_yolu)
        or die ("Veritabanına ulaşılamıyor!" . mysql_error() );

$sonuc = mysql_query("SELECT * FROM uyeler",$veri_yolu);
        printf("Adı         : %s<br>\n", mysql_result($sonuc,0,"adi"));
        printf("Soyadı      : %s<br>\n", mysql_result($sonuc,0,"soyadi"));
        printf("Uye Nosu: %s<br>\n", mysql_result($sonuc,0,"uye_no"));
?>
</BODY>
</HTML>
```

- Hazırladığımız "**mysql.php**" sayfasının html çıktısını apache'den isteyelim. Her iki protokol tarafından da desteklenmesi gerekir. Sorun varsa uyguladığınız adımları tekrar edin !

http://www.example.com/mysql.php -> http://sunucu_ip/mysql.php
https://www.example.com/mysql.php -> https://sunucu_ip/mysql.php

- Döküman da buraya kadar temel düzeyde bir Web-Database sunucu hazırlamaya çalıştık. Artık dökümanın ana konusu olan "***Chroot"*** olayına biraz değinelim.

CHROOT KOK DIZINI YARATMA

- **Chroot Nedir**? Açılımı "Change root" anlamı ise root dizin değiştirmedir.

Gerçek root kök dizine bağlı kalınarak yapay oluşturulmuş bir root dizindir. Oluşturulan bu dizin altında çalıştırılacak servis için herşey olacak. Hatta burayı öyle bir dizayn edecezki! servis kendisini gerçek kök dizinde çalışıyor gibi görecek. Burada dikkat edilecek en önemli nokta sistemin dosya hiyerarşisini korumak olacaktır. Kendisine ait aygıtları, dosyaları, kütüphane dosyaları ve password dosyaları oluşturacaz. Sizin için önem arzeden ve servisle alakası olmayan diğer servisler, dosyalar, v.s. bu ortamda kesinlikle olmamalıdır. Yani kısaca sisteme bağli çalişacak; sistem içinde sistem oluşturacaz.

- Bu ortam; öyle bir hapishane olacak ki servisin kırılması, ele geçirilmesi veya durdurulması v.s. gibi saldırılar sadece o ortamı ve ortama bağlı servisleri etkileyecek.

- *Biraz daha açık olmam gerekirse, apache'nin chroot kök dizinini oluşturuyorken kullanacağım erişim izinlerini gerçek kök dizinde uygulamak inanın çok çok zordur, extra yapılacaklar bölümün de bu erişim izinlerinin faydasını anlayacaz.*

MYSQL & CHROOT

- En basta 2 tür kurulumdan bahsetmiştik. I. ve II. tür kurulumlara göre devam edecez!

I. TÜR KURULUM İÇİN CHROOT KÖK DİZİNİ

```
#  umask 022
#  mkdir –p /chroot/mysql/{dev,etc,usr,var}
#  mkdir –p /chroot/mysql/usr/local/mysql/libexec/
#  mkdir –p /chroot/mysql/usr/local/mysql/share/mysql/english
#  mkdir –p /chroot/mysql/usr/local/mysql/{data,tmp}
#  mkdir –p /chroot/mysql/var/tmp
```

II. TÜR KURULUM İÇİN CHROOT KÖK DİZİNİ

```
#  umask 022
#  mkdir –p /chroot/mysql/{dev,etc,usr,var}
#  mkdir –p /chroot/mysql/usr/local/mysql/libexec
#  mkdir –p /chroot/mysql/usr/local/mysql/share/mysql/english
#  mkdir –p /chroot/mysql/usr/local/mysql/{data,tmp}
#  mkdir –p /chroot/mysql/var/{tmp,run}
#  mkdir –p /chroot/mysql/usr/{lib,libexec}
```

- Erişim izinlerini ayarlayalım. (Her iki tür kurulum içinde geçerlidir.)

```
#  chown –R root:sys /chroot/mysql/*
#  chown –R mysql:mysql  /chroot/mysql/usr/local/mysql/{data.tmp}
```

- Mysql için aygıtları oluşturalım. (Her iki tür kurulum içinde geçerlidir.)

```
# cd /chroot/mysql/dev/
# mknod null c 2 2 root:wheel
# chmod 666 ./null
```

- Password dosyalarımızı oluşturalım. Sadece mysql grubu ve kullanıcısı kalacak şekilde dosyaları edit'leyin. (Her iki tür kurulum içinde geçerlidir.)

```
# cd /chroot/mysql/etc
# cp /etc/{hosts,host.conf,group,resolv.conf,master.passwd} ./

# vi group
  mysql:*:88:

# vi master.passwd
  mysql:*:88:88::0:0:MysqlServer:/dev/null:/sbin/nologin

# pwd_mkdb –d /chroot/mysql/etc ./master.passwd
# rm –rf ./master.passwd
# ls | grep db

pwd.db
spwd.db
```

(Bu ikisinin bulunmasi yeterli.)

- Şimdi **mysql** için gerekli dosyaları birbir alıyoruz.

I. TÜR KURULUM İÇİN CHROOT KÖK DİZİNİ

```
# cp /usr/local/mysql/libexec/mysqld /chroot/mysql/usr/local/mysql/libexec/
# cp /usr/local/mysql/data/my.cnf /chroot/mysql/usr/local/mysql/data/
# cp –R /usr/local/mysql/data/* /chroot/mysql/usr/local/mysql/data/
# chown –R mysql:mysql /chroot/mysql/usr/local/mysql/data/*
# chown –R root:mysql /chroot/mysql/usr/local/mysql/data/my.cnf
# cp /usr/local/mysql/share/mysql/english/* \
  /chroot/mysql/usr/local/mysql/share/mysql/english/
```

II. TÜR KURULUM İÇİN CHROOT KÖK DİZİNİ

```
# cp /usr/local/mysql/libexec/mysqld /chroot/mysql/usr/local/mysql/libexec/
# cp /usr/local/mysql/share/mysql/english/* \
  /chroot/mysql/usr/local/mysql/share/mysql/english/
# cp –R /usr/local/mysql/data/* /chroot/mysql/usr/local/mysql/data/
# chown –R mysql:mysql /chroot/mysql/usr/local/mysql/data/*
# chown –R root:sys /chroot/mysql/usr/local/mysql/data/my.cnf
```

- 2. tür kurulumu seçti iseniz burayı okuyorsunuzdur. **mysqld** dinamik kütüphane kullanacak. Peki kullanacağı dosyaları nasıl belirleyecez. Burada karşımıza 2 yöntem çıkıyor. İlk olarak **ldd**, ikinci olarak **strings**. Bu araçları kullanarak kütüphane dosyaları hakkında biraz bilgi toplayacaz.

NOT : Çok istisnai durumlarda kütüphane dosyalarına bağlı kütüphane dosyaları olabiliyor. Mesela herşeyi tam yaptığınızı düşünürken, bir de bakıyorsunuz ki kütüphane dosyasının çalışması için yanında bir veya birkaç tane daha kütüphane dosyası olması gerekebiliyor. Bu yüzden kütüphane dosyalarını belirlerken dikkatli olalım lütfen !!!

```
#  ldd /usr/local/mysql/libexec/mysqld
        libwrap.so.3 => /usr/lib/libwrap.so.3 (0x2817b000)
        libz.so.2 => /usr/lib/libz.so.2 (0x28183000)
        libcrypt.so.2 => /usr/lib/libcrypt.so.2 (0x28190000)
        libstdc++.so.3 => /usr/lib/libstdc++.so.3 (0x281a9000)
        libm.so.2 => /usr/lib/libm.so.2 (0x281ee000)
        libc_r.so.4 => /usr/lib/libc_r.so.4 (0x28209000)

#  strings /usr/local/mysql/libexec/mysqld  | grep lib
        libwrap.so.3
        libz.so.2
        libcrypt.so.2
        libstdc++.so.3
        libm.so.2
        libc_r.so.4
```

- Yukarıda düştüğümüz notun uygulamasını yapalım ve sonuca bakalım.

```
#  ldd /usr/lib/libwrap.so.3
```

- Sonuç negatif. Sadece kendisi yeterlidir. Tabii ki burada biz bir tanesini denedik. Siz siz olun tedbiri elden bırakmayın. Hepsini kontrol edin ! En azından kendi başınıza başka bir servis için chroot ortam hazırlıyorken kütüphane dosyalarını nasıl belirleyeceğinizi tecrübe etmiş ve öğrenmiş olursunuz.

```
#  install –m 555 -C /usr/libexec/ld-elf.so.1  /chroot/mysql/usr/libexec
#  install –m 444 -C /var/run/ld.so.hints  /chroot/mysql/var/run/

#  install –m 555 -C /usr/lib/libwrap.so.3  /chroot/mysql/usr/lib
#  install –m 555 -C /usr/lib/libz.so.2    /chroot/mysql/usr/lib
#  install –m 555 -C /usr/lib/libcrypt.so.2   /chroot/mysql/usr/lib
#  install –m 555 -C /usr/lib/libstdc++.so.3    /chroot/mysql/usr/lib
#  install –m 555 -C /usr/lib/libm.so.2    /chroot/mysql/usr/lib
#  install –m 555 -C /usr/lib/libc.so.4    /chroot/mysql/usr/lib
#  install –m 555 -C /usr/lib/libc_r.so.4   /chroot/mysql/usr/lib
```

- Yavaş yavaş ilerlerken nihayet sonuca geldik. Mysql'i yeni ortamında başlatalım. Normal olarak **"--user=mysql"** ile kullanıcıyı belirliyorduk. Bu işi chroot'a bırakıyoruz.

```
#  chroot  -u mysql  /chroot/mysql  /usr/local/mysql/libexec/mysqld  &
```

- Pekala; Mysql'in Chroot kök dizinine taşınıp taşınmadığını nasıl anlayabiliriz ?

Bunu mysql'in araçlarını kullanmaya çalışarak test edebiliriz. Mysql'e bağlanmayı deneyelim.

```
#  /usr/local/mysql/bin/mysql –u root –p
```

- Normal olarak parolayı girdiğiniz halde erişiminiz engellendi ise işlem tamamdır.

Mysql chroot kök dizinde çalışmaya başlamıştır. Peki; "mysql, mysqladmin, mysqldump" gibi araçları nasıl kullanacaz ? Biz bu araçları yeni yerine install da etmedik. Bu araçları nasıl kullanabiliriz ? Bunun için **my.cnf** dosyamızda; **"mysql.lock"** dosyasının yeni yerini belirtmeliyiz.

```
#  vi  /usr/local/mysql/data/my.cnf
...
...
socket   =    /chroot/mysql/usr/local/mysql/tmp/mysql.lock
...
```

NOT : Bu işlemle ***mysql.lock*** dosyasının kalıcı olarak yerini tayin ettik, mesela mysql'i değişik ortamlarda kullanacaksanız; yani bir chroot ortamda, bir klasik ortamda çalıştıracaksak eğer **mysql.lock** dosyasının yeri ile oynamamayı öneririm. (Orjinal haliyle bırakalım lütfen). Çünkü insan dalgın olabiliyor, boşu boşuna uğraşmanızı istemem. Bunun yerine ***"Mysql'i Chroot ve normal ortam"*** da başlatmak için basit komut alias'ları veya script'ler yazabiliriz.

```
#  chroot –u mysql /chroot/mysql   /usr/local/mysql/libexec/mysqd   \
   --socket=/istediğiniz_dizin/mysql.lock
```

- Bu komut'u içerecek bir script ile mysql'i farklı ortamlarla başlatabiliriz. Sizin hayal gücünüze kalmıştır. Mysql için yapacaklarımız şimdilik bu kadar.

APACHE & SSL & PHP & CHROOT

- Apache'nin çalışması için gerekli dizinleri oluşturalım.

```
#  umask 022
#  mkdir -p /chroot/apache/{dev,etc,usr,var}
#  mkdir -p /chroot/apache/usr/{lib,libexec,local}
#  mkdir -p /chroot/apache/usr/local/{etc,apache}
#  mkdir -p /chroot/apache/usr/local/apache/{bin,lib,libexec,conf,logs,example.com}
#  mkdir -p /chroot/apache/usr/local/apache/conf/{ssl.crt,ssl.key}
#  mkdir -p /chroot/apache/var/run
```

NOT : Burada mysql ile alakalı olan fakat mysql'den bağımsız birkaç dizin daha yaratıyoruz Yapay kök dizinde, gerçek kök dizinin hiyerarşisini koruyoruz !!! Bu konuya değinmiştik.

```
#  mkdir -p /chroot/apache/usr/local/mysql/tmp
#  mkdir -p /chroot/apache/usr/local/mysql/lib/mysql
```

- Tekrar söylüyorum dökümanın tamamını okumadan lütfen işlemlere başlamayın !!!

Şüphe duyduğunuz noktalarda araştırabilir ve seçiminize bağlı olarak değiştirebilirsiniz.

- Erişim izinlerini ayarlayalım. Öncelikle apache'nin kök dizinine ve tüm alt dizinlere 755 desteğini verelim, daha sonra özel izin gerektiren durumları tektek kendimiz belirleyecez.

```
#  chown -R  root:sys  /chroot/apache/*
#  chown -R  mysql:mysql /chroot/apache/usr/local/mysql/tmp

#  cd /chroot/
#  chmod  -R  755 apache/*

#  chmod 711 apache

#  cd apache
#  chmod 400 {dev,etc,var}
#  chmod 711 usr
#  chmod 711 usr/local
#  chmod 711 usr/local/apache

#  cd /chroot/apache/usr/local/apache/
#  chmod 711 conf
```

- Kendi kendinize sorabilirsiniz, bu kadar katı kısıtlamaya ne gerek var diye, sorunun cevabını ise yine kendi kendinize dökümanin sonunda verebileceksiniz. (Extra yapılacaklar bölümünde...)

- Apache için aygıtları oluşturalım. Mysql'den farklı olarak SSL şifresini girmek için kullanacağımız bir tane de konsol yaratacaz. SSL'yi random desteği ile kullanacaksanız, **random** & **urandom** aygıtları da oluşturulmalıdır. Biz en genel aygıtları kullanacaz.

```
#  cd /chroot/apache/dev

#  mknod null c 2 2  root:wheel
#  mknod tty  c 1 0  root:wheel

#  chmod 666  ./{null,tty}
```

- etc altında ihtiyacımız olan dosyaları oluşturalım.

```
#  cd  /chroot/apache/etc
#  cp /etc/{hosts,host.conf,resolv.conf,group,master.passwd}  ./
```

- Apache, passwd dosyaları içerisinde illa ki nogroup ve nobody'yi de görmek istiyor. Bunun nedeni ise apache default olarak nobody ve nogroup hakları ile başlıyor.

```
#  vi group
nogroup:*:65533:
apache:*:89:

#  vi master.passwd
nobody:*:65534:65534::0:0:Unprivileged user:/nonexistent:/sbin/nologin
apache:*:89:89::0:0:ApacheServer::/sbin/nologin
```

```
#  pwd_mkdb -d /chroot/apache/etc ./master.passwd
#  rm –rf master.passwd
#  ls | grep db

pwd.db
spwd.db       (Bu iki passwd dosyası bizim için yeterlidir.)
```

- Apache'ye lazım olan kütüphane dosyalarını inceleyelim.

```
#  ldd   /usr/local/apache/bin/httpd

        libcrypt.so.2 => /usr/lib/libcrypt.so.2 (0x281a8000)
        libc.so.4 => /usr/lib/libc.so.4 (0x281c1000)

#  install -m 555 -C /usr/libexec/ld-elf.so.1  /chroot/apache/usr/libexec
#  install -m 444 -C /var/run/ld.so.hints  /chroot/apache/var/run/

#  install -m 444 -C /usr/lib/libcrypt.so.2  /chroot/apache/usr/lib/
#  install -m 444 -C /usr/lib/libc.so.4  /chroot/apache/usr/lib/
```

- Bunlar yeterli gibi görünüyor değil mi ? Peki ya php nerde ? Kütüphane dosyalarının kullandığı kütüphane dosyalarından bahsetmiştik. Bunu da tecrübe edelim.

```
#  ldd /usr/local/apache/libexec/libphp4.so

        libcrypt.so.2 => /usr/lib/libcrypt.so.2 (0x28271000)
        libmysqlclient.so.10 => /usr/local/mysql/lib/mysql/libmysqlclient.so.10 (0x2828a000)
        libm.so.2 => /usr/lib/libm.so.2 (0x282a6000)
        libz.so.2 => /usr/lib/libz.so.2 (0x282c1000)
        libc.so.4 => /usr/lib/libc.so.4 (0x2806a000)
```

- Bu dosyaları da install etmemiz lazım. (Bazılarını install etmiştik)

```
#  install -m 755 -C /usr/local/apache/libexec/libphp4.so  \
   /chroot/apache/usr/local/apache/libexec/

#  install -m 444 -C /usr/lib/libm.so.2  /chroot/apache/usr/lib/
#  install -m 444 -C /usr/lib/libz.so.2  /chroot/apache/usr/lib/
```

- Php mysql ile bağlantı kuruyorken bu dosyayı da isteyecek. Bizde hiyerarşiyi koruyoruz ve /chroot/apache kök dizini altında yarattığımız **usr/local/mysql/lib/mysql** klasörüne install ediyoruz.

```
#  install -m 755 -C /usr/local/mysql/lib/mysql/libmysqlclient.so.10  \
   /chroot/apache/usr/local/mysql/lib/mysql/
```

- Ayrıca mysql başladıktan sonra apache'nin mysql.lock dosyasını okuyabilmesi için hard link yapıyoruz.

```
#  ln /chroot/mysql/usr/local/mysql/tmp/mysql.lock  \
   /chroot/apache/usr/local/mysql/tmp/
```

- Eğer bu dizinleri ve mysql.lock dosyasını oluşturmazsak ! Apache /chroot/apache kök dizini altında çalıştığından habersiz bir şekilde **usr/local/mysql/tmp/mysql.lock** dosyasını okumak isteyecektir. Php'de mysql'e bağlanacağı zaman mysqlclient kütüphanesini kullanmak isteyecektir. Bunlar uygun şekilde olmazsa **mysql_connect** hatası verecektir. Başlarda söylediğimiz gibi hiyerarşi çok önemlidir, korumamız bize her zaman avantaj sağlar.

- Bunaldığınızdan eminim, Son konfigûrasyon dosyalarımızı alıyoruz.

```
# cp /usr/local/apache/bin/httpd   /chroot/apache/usr/local/apache/bin
# cp /usr/local/apache/conf/httpd.conf  /chroot/apache/usr/local/apache/conf
# cp /usr/local/apache/conf/mime.types   /chroot/apache/usr/local/apache/conf
# cp –R /usr/local/apache/example.com/* \
   /chroot/apache/usr/local/apache/example.com/
# cp /usr/local/apache/logs/*   /chroot/apache/usr/local/apache/logs
# cp /usr/local/apache/conf/ssl.crt/server.crt  /chroot/apache/usr/local/apache/conf/ssl.crt
# cp /usr/local/apache/conf/ssl.crt/server.key   /chroot/apache/usr/local/apache/conf/ssl.key
# cp /usr/local/lib/php.ini    /chroot/apache/usr/local/etc/

# chmod 400 /chroot/apache/usr/local/etc/php.ini
# chmod 400 /chroot/apache/usr/local/apache/conf/ssl.crt/server.crt
# chmod 400 /chroot/apache/usr/local/apache/conf/ssl.key/server.key
```

- Apache'yi başlatalım ve test edelim..

```
#  chroot  /chroot/apache  /usr/local/apache/bin/httpd  &&  \
   ln  /chroot/mysql/usr/local/mysql/tmp/mysql.lock \
   /chroot/apache/usr/local/mysql/tmp/                    (Apache normal başlatılır)

#  chroot /chroot/apache /usr/local/apache/bin/httpd -DSSL  && \|
   ln  /chroot/mysql/usr/local/mysql/tmp/mysql.lock \
   /chroot/apache/usr/local/mysql/tmp/                    (Apache ssl destekli başlatılır)
```

http://www.example.com
https://www.example.com
http://www.example.com/index.php
https://www.example.com/index.php
http://www.example.com/mysql.php
https://www.example.com/mysql.php

- SSL destekli 1024 bit'le şifrelenmiş çalışan bir Apache & PHP & Mysql sunucumuz var artık. Hayırlı olsun..

EXTRA YAPILACAKLAR

- Paranoyak olalım ve default olarak gelen bir kaç db'yi temizleyelim, root kullanıcısının adını değiştirelim..

```
#  /usr/local/mysql/bin/mysql –u root –p

mysql>  drop database test;                    (Test database'sini sildik)
mysql>  use mysql;                             (mysql database'sini kullanmaya gectik)
mysql>  delete from db;                        (mysql database'sinden db tablosunu sildik)
mysql>  delete from user where not (host="localhost" and user="root");
          (user tablosundan ismi root ve hostu localhost olan kullanıcı dışında herşeyi sildik)
mysql>  update user set user="sizin_root_isminiz" where user="root";
          (root kullanıcısının ismini db'den değiştirdik,  artık root hesabi yok)
mysql>  flush privileges;                      (ayarlarin geçerliliğini sağladik)
```

- Evet paranoyakliliğimiz işe yaradı mı acaba ?

```
#  /usr/local/mysql/bin/mysql –u root –p       (Hata almanız normal, bide şöyle deneyelim)
#  /usr/local/mysql/bin/mysql –u  sizin_root_isminiz  –p
Enter password:
```

- Son birşey, **c99shell** adında ki php betiğini indirin ve bu betiği apache'de sitenizin olduğu dizine **shell.php** adında kaydedin, Varsayalım ki bu betiği, şeytani cracker sisteminize bir şekilde upload etmeyi başardı. Bakalım tarayıcıdan çağırınca sistem üzerinde ne yapabilecek. Kısaca kendi sunucunuzu kendiniz hack edin demek istiyorum. Tam bir cracker mantığı ile yapamayabiliriz belki ama en azından neler yapılabilir onlara dikkat edelim..

- Sunucumuzda eksiklerimiz ne ? Yoksa bazı dosya izinlerini yanlış tanımlamış olabilir miyiz ? Acaba chroot işe yarıyor mu !!! Bu soruların cevabını merak edin.. Apache'ye izinleri veriyorken çok katı davranmıştık. Kullandığımız bu izinlerle gerçek kök dizinde sistem sürekli birşeylerin hatasını verecektir. Ben tecrübe ettim. Chroot'u araştırmayada
bu şekilde başladım..

########################## **mysql.sh** ##########################

```
#!/bin/sh

CHROOTD=/usr/sbin/chroot
CHROOTFLAGS="-u mysql"
CHROOT_MYSQL=/chroot/mysql
CHROOT_APACHE=/chroot/apache
MYSQLD=/usr/local/mysql/libexec/mysqld
PIDFILE=/usr/local/mysql/data/makina_hostname.pid

case "$1" in

start)

${CHROOTD} ${CHROOTFLAGS} ${CHROOT_MYSQL} ${MYSQLD} \
>/dev/null 2>&1  &
echo "Mysql  baslatildi."
;;

stop)

kill `cat ${CHROOT_MYSQL}/${PIDFILE}` >/dev/null 2>&1
echo "Mysql durduruldu."
;;
*)

        echo "Kullanimi: `basename $0` {start|stop}" >&2
exit 64
;;
esac
exit 0
```

############################## **apache.sh** ##############################

```
#!/bin/bash

HTTPD=/usr/local/apache/bin/httpd
CHROOTD=/usr/sbin/chroot
CHROOT_MYSQL=/chroot/mysql
CHROOT_APACHE=/chroot/apache
CHROOT_FLAGS="-DSSL"
SOCKET=/usr/local/mysql/tmp/mysql.lock
PIDFILE=/usr/local/apache/logs/httpd.pid
CHROOT_PID=${CHROOT_APACHE}/${PIDFILE}


case "$1" in

start)
${CHROOTD}  ${CHROOT_APACHE}  ${HTTPD}  ${CHROOT_FLAGS} && \
ln ${CHROOT_MYSQL}/${SOCKET}  ${CHROOT_APACHE}/${SOCKET}
;;

stop)
if [ -f ${CHROOT_PID} ];
then
PID=`cat ${CHROOT_APACHE}/${PIDFILE}`
kill $PID
rm ${CHROOT_APACHE}/${SOCKET} >/dev/null 2>&1
echo "Apache durduruldu."
else
rm ${CHROOT_APACHE}/${SOCKET} >/dev/null 2>&1
echo " Calisan bir Apache sunucu bulunamadi."
fi
;;

*)

echo " Kullanimi : `basename $0` {start|stop}" >&2
        exit 64
        ;;
esac
exit 0
```

Yazarın Notu

Bu dokumanda elimden geldiğince Apache ve apache bileşenleri (php - ssl - mysql) üzerinde güvenliği nasıl sağlayabiliriz, nelere dikkat etmeliyiz buna değinmeye çalıştım..

Saygılarımla...
Mehmet ÇELİK
bsd_daemon at msn dot com

YARDIMCI KAYNAKLAR

1. www.faqs.org
2. docs.linux.com
3. penguin.triumf.ca
4. www.bsd.org.pe
5. www.opennet.ru
6. www.securityfocus.com
7. www.enderunix.org -> Openbsd üzerinde Apache
8. www.olympos.org -> Apache'yi güvenli hale getirmenin 20 yolu
   -> Mysql Güvenliği