مرکز تخصصی آپا
دانشگاه صنعتی اصفهان
(آگاهی‌رسانی، پشتیبانی و امداد در حوزه شبکه)

قرار دادن آپاچی در زندان

شهریور ۱۳۹۲

www.nsec.ir

۱. مقدمه

گاهی اوقات نفوذگرها از آسیب پذیری های تازه کشف شده، استفاده می کنند و وارد سیستم می شوند. اولین کاری که آنها انجام می دهند این است که خود را superuser کنند. اگر این مسئله اتفاق بیافتد، سیستم را باید از ابتدا نصب کرد و راه دیگری نیست.

در این قسمت سعی می کنیم سیستم را به گونه ای تغییر دهیم که نفوذگر فقط به جزئی از آن دسترسی داشته باشد. و این کار را بوسیله فراخوانی سیستمی موسوم به chroot (2) انجام می دهیم. این وسیله امکان می دهد که برای یک پروسه محدودیت قائل شویم و دسترسی او را به فایل سیستم محدود کنیم. این مسئله این گونه اتفاق می افتد که ما یک پوشه را انتخاب می کنیم و می گوییم که این پوشه root سیستم باشد(ریشه اصلی مثل /). هنگامی که system call اجرا می شود، پروسه نمی تواند بازگردد و در همان ریشه تعریف شده باقی می ماند که اصطلاحا آن را زندان می گویند.

کاربر root همواره می تواند از زندان بیرون بیاید. مسئله کلیدی در ساختن یک زندان بدون درز و رخنه، این است که اجازه ندهید هیچ پروسه ای در درون زندان jail ، با دسترسی root اجرا شود. همین طور نباید خارج از jail، پروسه ای با دسترسی همان کسی که در داخل زندان است اجرا شود. در برخی موارد نفوذگر می تواند از یک پروسه در داخل زندان به پروسه ای در خارج آن بپرد و از زندان خارج شود. برای همین است که تاکید می شود که از اکانت های مجزا برای اجرای آپاچی استفاده کنید.

با بوجود آمدن زندان این مزایا را بدست خواهید آورد:

محدودیت

اگرنفوذگر وارد سیستم شود، فقط به قسمت محدودی از سیستم فایل دسترسی دارد.

بدون پوسته

بسیاری از exploit ها نیاز به پوسته /bin/sh دارند تا اجرا شوند و شما می توانید این ابزار را از زندان حذف کنید.

محدودیت ابزارهای موجود

نبود پوسته (shell) و کامپایلر و دیگر ابزارها ...

نبود فایلهای باینری suid شده

اگر فایل های SetUID شده در درون زندان باشند، تمامی تلاش ما برای بستن زندان بی فایده است.

۲. نحوه های استفاده از ابزارهای مرتبط برای chroot

در هنگام برخورد با مشکلاتی که معمولا در تغییر دسترسی برنامه و زندانی کردن آنها روی می دهد. چرا که آپاچی انتظار دارد که با دسترسی کامل روی سیستم عامل اجرا شود. لازمه استفاده از chroot این است که با ابزار هایی آشنا باشید که شما را در مشکلاتی که بدین خاطر بوجود می آید کمک کند.

۲.۱. یک نمونه استفاده از chroot

ابتدا نیاز داریم برای ریشه جدید، پوشه بسازیم. دستور زیر این کار را خواهد کرد:

```
# mkdir  /chroot
```

دستور chroot یک مسیر را به عنوان پارامتر اول می گیرد و یک پروسه را به عنوان پارامتر دوم. حال دستور زیر را اجرا می کنیم و خطای زیر را می بینیم:

```
# chroot /chroot /bin/bash
chroot: /bin/bash: No such file or directory
```

مشکل دستور بالا این است که chroot ابتدا خود را در زندان می اندازد و بعد سعی می کند که bin/bash/ را اجرا کند و از آنجا که در درون زندان این مسیر وجود ندارد خطا می دهد. Bash را در زندان کپی می کنیم و دستور را دوباره اجرامی کنیم:

```
# mkdir /chroot/bin
# cp /bin/bash /chroot/bin/bash
# chroot /chroot /bin/bash
chroot: /bin/bash: No such file or directory
```

این خطا هم به علت این است که bash به یکسری library ها وابسته است که آنها موجود نیستند. پس برای حل این گونه مشکلات ابزار قسمت بعد را نیاز خواهیم داشت.

۲.۲. استفاده از ldd برای کشف وابستگی ها

ldd در همه لینوکس های موجود است و کارش نشان دادن وابستگی های موجود برای یک فایل باینری است. اگر این دستور را مانند زیر اجرا کنیم، می بینیم:

```
# ldd /bin/bash
        libtermcap.so.2 => /lib/libtermcap.so.2 (0x0088a000)
        libdl.so.2 => /lib/libdl.so.2 (0x0060b000)
        libc.so.6 => /lib/tls/libc.so.6 (0x004ac000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x00494000)
```

پس موارد بالا را کپی می کنیم:

```
# mkdir /chroot/lib
# cp /lib/libtermcap.so.2 /chroot/lib
# cp /lib/libdl.so.2 /chroot/lib
# cp /lib/tls/libc.so.6 /chroot/lib
# cp /lib/ld-linux.so.2 /chroot/lib
```

اجرای bash بالاخره موفق به انجام می رسد:

```
# chroot /chroot /bin/bash
bash-2.05b#
```

اگر چه shell داریم اما کار زیادی نمی توانیم بکنیم. چرا که بسیاری از باینری های دیگر موجود نیستند. (مثلا ls). تنها برخی از دستورات تعبیه شده در bash را می توان اجرا کرد. برای مثال:

```
bash-2.05b# pwd
/
bash-2.05b# echo /*
/bin /lib
bash-2.05b# echo /bin/*
/bin/bash
bash-2.05b# echo /lib/*
/lib/ld-linux.so.2 /lib/libc.so.6 /lib/libdl.so.2 /lib/libtermcap.so.2
```

۲.۳. استفاده از strace برای مشاهده داخل یک پروسه

ابزار strace (در سیستم های غیر لینوکس truss)می تواند درون پروسه را ببیند و فراخوانی های سیستمی آنها را مشاهده کند. این ابزار دید زیادی نسبت به برنامه ها و نحوه اجرای آنها می دهد ، بدون این که به کد منبع آنها دسترسی داشته باشد. بوسیله chroot و ldd شما می توانید کاری کنید که برنامه ها در درون زندان اجرا شوند، اما به strace برای فهمیدن این که چرا برنامه ها خطا می دهند (اگرچه پیام خطا ندارند)، احتیاج خواهید داشت. پس strace را در زندان کپی کنید و فراموش نکنید که بعدا آن را پاک کنید.

اگر می خواهید تجربه کنید، برنامه زیر را بنویسید:

```
#include <stdio.h>
#include <stdarg.h>

int main(void) {
    puts("Hello world!");
}
```

یکبار برنامه را با پشتیبانی فایل های share سیستم و یکبار بدون آنها اجرا کنید:

```
# gcc helloworld.c -o helloworld.shared
# gcc helloworld.c -o helloworld.static -static
```

استفاده از strace برای نوع static خروجی زیر را می دهد:

```
# strace ./helloworld.static
execve("./helloworld.static", ["./helloworld.static"], [/* 22 vars */]) = 0
uname({sys="Linux", node="ben", ...})   = 0
brk(0)                                  = 0x958b000
brk(0x95ac000)                          = 0x95ac000
fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
old_mmap(NULL, 4096, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xbf51a000
write(1, "Hello world!\n", 13Hello world!
)           = 13
munmap(0xbf51a000, 4096)                = 0
exit_group(13)
```

خروجی آن زشت و بد است. اما اهمیتی ندارد که هر خط چه می گوید. برنامه های درون زندان بر ای این که نمی توانند به فایلی دسترسی داشته باشند، خطا تولید می کنند. اگر چنین مسئله ای وجود داشته باشد، در نزدیکی اواخر خروجی strace، می توان این فایل ها را دید. مثلا :

```
open("/usr/share/locale/locale.alias", O_RDONLY) = -1 ENOENT
(No such file or directory)
```

می توان نسخه دینامیک آن را نیز با strace ببینید و بدانید برای یک برنامه کوچک، مقدار زیادی دسترسی به فایل های دیگر اتفاق می افتد.

۳. استفاده از chroot برای قرار دادن آپاچی در زندان Jial

اکنون نیاز مندیم که خانه جدیدی برای آپاچی درست کنیم و آن را به آنجا منتقل کنیم:

```
# mkdir -p /chroot/apache/usr/local
# mv /usr/local/apache /chroot/apache/usr/local
# ln -s /chroot/apache/usr/local/apache /usr/local/apache
# mkdir -p /chroot/apache/var
# mv /var/www /chroot/apache/var/
# ln -s /chroot/apache/var/www /var/www
```

استفاده از لینک سمبلیک از محل قبلی آپاچی به ما اجازه می دهد که آپاچی را هر گاه که بخواهیم، در زندان یا خارج از آن استفاده کنیم. مثلا برای به روز کردن آن.

مانند بسیاری از برنامه های دیگر ، آپاچی نیز نیازمند فایلهای library است. با ldd می توانیم نام آن ها را پیدا کنیم: (این خروجی مربوط به آپاچی است که همه ماژول ها را به طورت ایستا در خود دارد):

```
# ldd /chroot/apache/usr/local/apache/bin/httpd
        libm.so.6 => /lib/tls/libm.so.6 (0x005e7000)
        libcrypt.so.1 => /lib/libcrypt.so.1 (0x00623000)
        libgdbm.so.2 => /usr/lib/libgdbm.so.2 (0x00902000)
        libexpat.so.0 => /usr/lib/libexpat.so.0 (0x00930000)
        libdl.so.2 => /lib/libdl.so.2 (0x0060b000)
        libc.so.6 => /lib/tls/libc.so.6 (0x004ac000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x00494000)
```

ما یک کپی از همه در زندان درست می کنیم:

```
# mkdir /chroot/apache/lib
# cp /lib/tls/libm.so.6 /chroot/apache/lib
# cp /lib/libcrypt.so.1 /chroot/apache/lib
# cp /usr/lib/libgdbm.so.2 /chroot/apache/lib
# cp /usr/lib/libexpat.so.0 /chroot/apache/lib
# cp /lib/libdl.so.2 /chroot/apache/lib
# cp /lib/tls/libc.so.6 /chroot/apache/lib
# cp /lib/ld-linux.so.2 /chroot/apache/lib
```

۳.۱. قرار دادن resolution file های کاربرها و گروه ها در زندان

اگر چه کاربر httpd در سیستم موجود است، اما در زندان خبری از او نیست. زندان باید قابلیت های احراز اصالت خود را داشته باشد:

```
# mkdir /chroot/apache/etc
# cp /etc/nsswitch.conf /chroot/apache/etc/
# cp /lib/libnss_files.so.2 /chroot/apache/lib
```

همان شناسه های کاربری و گروه ها و همان شماره های کاربری و شماره گروه هایی که در قبل استفاده می کردید رانیز در زندان استفاده کنید. سیستم فایل، شماره های کاربری و گروهی را برای حل مسئله Ownership ذخیره می کند. این وظیفه دستور ls است که نام کاربران را از لیست آن ها گرفته و آن را روی صفحه نشان دهد. اگر یک لیست کاربری درون سیستم و دیگری در درون زندان، ((با شماره کاربری متفاوت))، باشد، Directory Listing منطقی خواهد :

```
# echo "httpd:x:500:500:Apache:/:/sbin/nologin" > /chroot/apache/etc/passwd
# echo "httpd:x:500:" > /chroot/apache/etc/group
```

در این لحظه تقریبا آپاچی آماده راه اندازی است. تنها یک سری فایل های دیگر نیاز است تا بتواند domain name resolution را انجام دهد:

```
# cp /lib/libnss_dns.so.2 /chroot/apache/lib
# cp /etc/hosts /chroot/apache/etc
# cp /etc/resolv.conf /chroot/apache/etc
```

۳.۲. انتهای آماده سازی زندان برای آپاچی

اگر چه این فایل ها لازم نیستند، اما تجربه نشان داده برخی از اسکریپت ها برای اجرا شدن نیاز به آن ها دارند. پس آن ها را کپی کنید تا از خطاهای مرموز جلوگیری کنید.

یکسری Device های خاص را بعد از استفاده از ls بوجود آورید، تا وجود پوشه dev/ را امتحان کنید تا بدانید که چه اعدادی باید استفاده شوند:

```
# mkdir /chroot/apache/dev
# mknod -m 666 /chroot/apache/dev/null c 1 3
# mknod -m 666 /chroot/apache/dev/zero c 1 5
# mknod -m 644 /chroot/apache/dev/random c 1 8
```

اضافه کردن پوشه های موقتی:

```
# mkdir /chroot/apache/tmp
# chmod +t /chroot/apache/tmp
# chmod 777 /chroot/apache/tmp
```

در نهایت، محدوده زمانی و locale را تنظیم کنید(ما می توانیم کل /usr/share/locale را کپی کنیم اما به خاطر حجمش نمی کنیم):

```
# cp /usr/share/zoneinfo/MET /chroot/apache/etc/localtime
# mkdir -p /chroot/apache/usr/lib/locale
# set | grep LANG
LANG=en_US.UTF-8
```

```
LANGVAR=en_US.UTF-8
# cp -dpR /usr/lib/locale/en_US.utf8 /chroot/apache/usr/lib/locale
```

۳.۳. آماده سازی PHP برای کار در Jail

برای کار php در زندان شما باید آن را به صورت معمولی نصب کنید و سپس فایل ها و libraryهایی که نیاز دارد را در زندان کپی کنیم:

```
# ldd /chroot/apache/usr/local/apache/libexec/libphp4.so
        libcrypt.so.1 => /lib/libcrypt.so.1 (0x006ef000)
        libresolv.so.2 => /lib/libresolv.so.2 (0x00b28000)
        libm.so.6 => /lib/tls/libm.so.6 (0x00111000)
        libdl.so.2 => /lib/libdl.so.2 (0x00472000)
        libnsl.so.1 => /lib/libnsl.so.1 (0x00f67000)
        libc.so.6 => /lib/tls/libc.so.6 (0x001df000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x00494000)
```

برخی از libraryها از قبل موجود هستند، پس نیازی به کپی کردن آن ها نیست.(در خروجی قبلی پررنگ نشان داده شده است):

```
# cp /lib/libresolv.so.2 /chroot/apache/lib
# cp /lib/libnsl.so.1 /chroot/apache/lib
```

یکی از مشکلاتی که برای php در زندان پیش می آید این است که نمی تواند ایمیل بفرستند زیرا باینریsendmail موجود نیست. برای حل آن می توانید در تنظیمات php دستکاری کنید تا از پروتکل SMTP برای فرستادن ایمیل استفاده کند. (localhot یا سرورهای smtp دیگر). مورد زیر را در فایلphp.ini قرار دهید:

```
SMTP = localhost
```

۳.۴. آماده سازی Perl برای کار در Jail

برای درست کارکردن perl در jail، باید فایل های زیر را کپی کنید:

```
# cp -dpR /usr/lib/perl5 /chroot/apache/usr/lib
# mkdir /chroot/apache/bin
# cp /usr/bin/perl /chroot/apache/bin
```

مشخص کردن فایل های گم شده:

```
# ldd /chroot/apache/bin/perl
        libperl.so => /usr/lib/perl5/5.8.1/i386-linux-thread-multi
/CORE/libperl.so (0x0067b000)
        libnsl.so.1 => /lib/libnsl.so.1 (0x00664000)
        libdl.so.2 => /lib/libdl.so.2 (0x0060b000)
        libm.so.6 => /lib/tls/libm.so.6 (0x005e7000)
        libcrypt.so.1 => /lib/libcrypt.so.1 (0x00623000)
        libutil.so.1 => /lib/libutil.so.1 (0x00868000)
        libpthread.so.0 => /lib/tls/libpthread.so.0 (0x00652000)
        libc.so.6 => /lib/tls/libc.so.6 (0x004ac000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x00494000)
```

اضافه کردن آن ها به زندان:

```
# cp /lib/libutil.so.1 /chroot/apache/lib
# cp /lib/tls/libpthread.so.0 /chroot/apache/lib
```

۳.۵. برحذر شدن از مشکلات کوچک زندان

ما نمی توانیم در درون زندان از sendmail استفاده کنیم و نصب sendmail در زندان دلایلی را که ما برایش زندان درست کردیم، به چالش می کشد. اگر شما به چنین مشکلی برخوردید می توانید از mini_sendmail استفاده کنید، (http://www.acme.com/software/mini_sendmail/) یک جانشین برای sendmail که مخصوص کار در زندان ساخته شده است. بسیاری از زبان ها می توانند بوسیله library های خود مستقیما به یک سرور smtp، ایمیل بفرستند، perl هم می تواند با استفاده از libraryموسوم به Mail::Sendmail، این کار را انجام دهد. استفاده از این بسته، تعداد بسته هایی با که باید در زندان نصب شود را کم می کند.

شما احتمالا هنگام اجرای اسکریپت ها مشکل اتصال با پایگاه داده ها را خواهید داشت. دلیلش هم سعی برنامه برای دسترسی به خارج از زندان است. این مورد هنگامی روی خواهد داد که شما از localhost به عنوان سرور پایگاه داده، برای اتصال به پایگاه داده استفاده می کنید. هنگامی که library مربوط به database client عبارت localhost را می بیند، سعی می کند تا بوسیله Unix domain socket به پایگاه داده متصل شود. این سوکت معمولا فایلی است که در آدرس var/lib/ یا tmp/ یا var/run/ قرار دارد. یکی از راه های حل این مشکل استفاده از ۱۲۷.۰.۰.۱ به عنوان host name است تا database client را مجبور کنید تا از TCP/IP برای اتصال استفاده کند. اگر چه این مورد از لحاظ کارایی، کندتر خواهد شد. یک راه بهتر این است که یک فایل از نوع socket در درون زندان داشته باشیم.

برای PostgreSQL، فایل Ipostgresql.conf را بیابید (معمولا در /var/lib/pgsql/data) و خط شامل unix_socket_directory را به صورت زیر تغییر دهید:

```
unix_socket_directory = '/chroot/apache/tmp'
```

یک لینک سمبلیک از محل قبل به محل جدید تهیه کنید:

```
# ln -s /chroot/apache/tmp/.s.PGSQL.5432 /tmp
```

mySQL معمولا فایل تنظیماتش را با نام my.cnf و در مسیر etc/ ذخیره می کند. شما می توانید یک مورد را برای کلاینت اضافه کنید و به او بگویید که در کجا به دنبال socket بگردد:

```
[mysqld]
datadir=/var/lib/mysql
socket=/chroot/apache/var/lib/mysql/mysql.sock

[client]
socket=/chroot/apache/var/lib/mysql/mysql.sock
```

یا همان طور که در postgreSQL دیدید میتوانیدلینک سمبلیکی درست کنید:

```
# mkdir -p /chroot/apache/var/lib/mysql
# chown mysql /chroot/apache/var/lib/mysql/
# ln -s /chroot/apache/var/lib/mysql/mysql.sock /var/lib/mysql
```