復習 FreeBSD Jail

佐藤　広生 <hrs@FreeBSD.org>

東京工業大学/ FreeBSD Project

2020/9/25

FreeBSD Jailとは

- **ユーザプロセスから見える資源を分離するための機能**
- **実現できること：**
  - **init(8)レベルで分離する**
    → 複数の独立した環境のように見える
    例：2台の物理マシンを、1台の物理マシン上に作った 2 個の jail で置き換える
  - **個別のプロセスレベルで分離する**
    → 特定の資源が見えないように孤立化できる
    例：httpd プロセスを、アクセスが必要なファイル群のみが見える状態で動かす

FreeBSD Jailとは

- **ユーザプロセスから見える資源を分離するための機能**

- **実行環境 = ユーザプロセスの集合体**
- カーネルは要求に応じて資源を分配するだけ
  - 分配に必要な情報はカーネルの外にあることがほとんど
  - 例：/etc/passwd はどこにある？誰がチェックしてる？

- **カーネルは共通**

  → 抽象化されたハードウェア（/dev の内容、NIC 等）は、jail を使っても使わなくても 1 組しかない

実行環境の分割

- **昔からある方法: chroot jail**
  - UNIX = 資源をファイルで管理している
  - 見えるファイルを分ければいいのでは！

実行環境の分割

- **昔からある方法: chroot jail**
  - UNIX = 資源をファイルで管理している
  - 見えるファイルを分ければいいのでは！
  - chroot システムコールを発行すると、
    - そのプロセスから見えるルートディレクトリが変わる
    - ファイルシステム上でアクセスできる範囲が限定される
- **この種の環境の分割　＝「名前空間の分離」**
  - 実際の資源はまとまって管理されているけれど、
    それにアクセスする方法（名前）が分離されている

fs 名前空間の分離

- **やってみよう**
  - /a/jail に今のシステムのコピーをつくって、そこでプロセスを動かす

```
# mkdir -p /a/jail /a/jail/dev /a/jail/tmp
# cp -Rp /etc /lib /bin /sbin /usr /var /a/jail
# cd /
# mount -t devfs devfs /a/jail/dev
# chroot /a/jail /bin/sh
```

これだけでおしまい！

- make installworld でも作れる（/etc とかの設定は用意しないとダメ）

```
# mkdir -p /a/jail
# cd /usr/src && make DESTDIR=/a/jail installworld
```

fs 名前空間の分離

- **chroot jail でできること**
  - カーネルが共通で見えるルートディレクトリが異なる（だけ）
  - 複数のマシンのルートディレクトリをまるまる
    あるディレクトリにコピーして、chroot で /bin/sh を
    起動すると、ほぼ独立した環境になる
  - chroot したディレクトリの上は見えない

- セキュリティ対策として使われることがしばしば
  （ただし抜け道はいくつか存在する）

fs 名前空間の分離

- **できないこと**
  - ファイルとして管理されていない資源は分離されてないので、垣根を超えることができてしまう
  - ネットワーク設定、プロセスに対する kill など

FreeBSD Jail による分離

- **もっと分離したい！**
  - FreeBSD は chroot を拡張して、
    他の資源も分離できるものをつくった ＝ “FreeBSD Jail”
  - **中身は chroot + α**
  - カーネル内の名前空間が分離されている
    - sysctl MIB
    - proc list
    - PID 空間（PID + Jail ID で管理）
    - 注意：UID はカーネルが管理するものではないので非分離
  - **資源は同じカーネルが一括管理しているが、**
    Jail 環境はその外の資源が見えない

FreeBSD Jail による分離

- **やってみよう**
  - ファイルシステムの分離をしないで /bin/sh を jail で実行してみる

```
# jail -c name=j1 host.hostname=j1.example.com path=/ command=/bin/sh
```

これだけでおしまい！

- jail コマンドを使うと、分離したプロセスが作れる
- ファイルシステムを分離したければ chroot と同じ原理でできる
- コマンドラインは複数の指定方法があるが、今は A=B 形式の設定を並べるか、/etc/jail.conf に設定を並べる方法が推奨

FreeBSD Jail による分離

- **やってみよう**
  - ファイルシステムの分離をしないで /bin/sh を jail で実行してみる

```
# jail -c name=j1 host.hostname=j1.example.com path=/ command=/bin/sh
# ps auwwx
USER    PID  %CPU %MEM     VSZ     RSS  TT  STAT STARTED     TIME COMMAND
0     47173   0.0  0.2   14636    2168   3  SJ    6:08AM 0:00.01 /bin/sh
0     47178   0.0  0.1   14328    1524   3  R+J   6:11AM 0:00.00 ps auwwx
#
```

- jail -c コマンドを使うと、分離したプロセスが作成される
- ファイルシステムを分離したければ chroot と同じ原理で可能
- コマンドラインは複数の指定方法があるが、今は A=B 形式の設定を並べるか、/etc/jail.conf に設定を並べる方法が推奨

FreeBSD Jail による分離

- **やってみよう**
  - ちなみに chroot の時は... （たくさん出てくる）

```
# mkdir -p /a/jail /a/jail/dev /a/jail/tmp
# cp -Rp /etc /lib /bin /sbin /usr /var /a/jail
# cd /
# mount -t devfs devfs /a/jail/dev
# chroot /a/jail/bin/sh
# ps auwwx
USER    PID  %CPU %MEM    VSZ    RSS  TT  STAT STARTED       TIME COMMAND
0        11 199.9  0.0      0     32  ??  RL    7:59AM 2611:43.88 [idle]
0         0   0.0  0.0      0    160  ??  DLs   7:59AM    0:00.06 [kernel]
   :
   :
   :
   :
   :
   :
```

FreeBSD Jail による分離

- **やってみよう**
  - jail(8) で起動したプロセスは、jailed process になる

```
# jail -c name=j1 host.hostname=j1.example.com path=/ command=/bin/sh
# ps auwwx
USER   PID  %CPU %MEM    VSZ    RSS  TT  STAT STARTED    TIME COMMAND
0    47173   0.0  0.2  14636   2168   3  SJ    6:08AM 0:00.01 /bin/sh
0    47178   0.0  0.1  14328   1524   3  R+J   6:11AM 0:00.00 ps auwwx
```

- jail には番号が付いている(JID)。jail -c は、新しいjail をつくる
- すべての jailed process が終了すると、jail は消える
- 消えないようにするには、jail -c persist を指定する
- jexec 1 /bin/sh とすると、jid=1 の環境で実行
- jls とすると、現在の jail 一覧を取得

FreeBSD Jail

- **FreeBSD Jail の環境**
  - root 以外の UID からのファイルに対する操作は特に変わらず
    - ただし、root 権限が限定的
      - カーネルはUID==0を特別扱いしている
      - ファイルシステムに対しては通常通り（ACL を使う）
      - sysctl や jail 環境外部に影響を与えるシステムコールは拒否（raw socket, mount, sysvipc など）
  - 名前空間が分かれているので、Jail の外のプロセスは見えない
  - ネットワークは、アドレスだけ名前空間を分離している

FreeBSD Jail のネットワーク

- **ネットワークの名前空間分離（とてもわかりにくい...）**
  1. jail コマンドで設定したアドレスが、NICに見える。複数設定できる
  2. jail の中では、127.0.0.1/32 が最初に設定したアドレスのaliasになる（IPv6は ::1/128）
- 設定したアドレスは、ホスト環境でももちろん見える
- **何が問題？**
  - Jail にあるプロセスが 127.0.0.1 で listen すると、外部ネットワークからもアクセスできてしまう
  - 同じサブネットのアドレスを持つ隣の jail も見える

FreeBSD Jail のネットワーク

- **よく使う対策**
  - lo1 を作って、jail には 127.0.1.1 のように 127/8 のアドレスを付ける

```
# ifconfig lo1 create
# jail -c ipv4.addr="lo1|127.0.1.1/32" ipv4.addr+="igb0|192.168.0.1/24"
```

  - lo1 は他の jail にも見えてしまうので、同じアドレスでの通信（例えば 127.0.1.1/32 <-> 127.0.1.1/32）のみを許可するよう、パケットフィルタを設定する
  - パケットフィルタは分離されていないので、ホスト環境へ。

起動時にJailを作りたい

- **起動時に自動的にjail環境を**
  - /etc/rc.conf と /etc/jail.conf に書く

```
jail_enable="YES"
jail_list="j1"
```

```
# /etc/rc.d/jail start
```

- thin jail: 特定のプロセスだけを jail に入れて、それを実行する
- fat jail: chroot で行ったように、システムのディレクトリ構造をコピーして、/etc/rc を実行する

ネットワークの分離

- **vimage という機能がデフォルトで使えるようになった**
  - ネットワークスタックの分離
- **やってみよう**
  - ファイルシステムの分離をしないで /bin/sh を jail で実行してみる

```
# jail -c name=j1 host.hostname=j1.example.com path=/ vnet command=/bin/sh
# ifconfig
```

ネットワークの分離

- スタックレベルで分離されているので、NIC は共有されない
  - NIC の移動：ifconfig igb0 vnet 1 のように指定する
- その他の設定は、ホスト環境と同じ
  - プロセスレベルでの jail では操作が難しいので、/etc/rc.conf を使うことを推奨

- epair(4) 仮想 ethernet NIC
  - ifconfig epair0 create -> epair0a と epair0b ができる
  - カーネル内部で接続されているので、jail間・ホスト-jail間で通信したい時に便利

実例

おしまい

- 質問はありますか？