Jail services

Paweł Jakub Dawidek
<pjd@FreeBSD.org>

What's this?

- dynamic extensions for the in-kernel prison structure
- allows to load some service that operates on jails and attach some data to each jail structure

Why?

- I'd like to use it to allow ZFS file systems management within a jail
- not much code needed on ZFS side - I reuse zones code
- ZFS fits just great in jails framework:
  - zpool – manages raw disks from non-jailed environment
  - zfs – logical file system – no access to raw disks

Example of use

```
main# zpool create tank mirror da0 da1
main# zfs create tank/jail
...
main# jail /tank/jail 127.0.0.1 /bin/tcsh
main# zfs jail -i <jid> tank/jail
jail# zfs create tank/jail/home
jail# zfs snapshot tank/jail/home
```

No access to raw disks

- very important feature – our file systems are not ready for corrupted metadata - it will panic entire system

KPI

```
struct prison_service *prison_register(const char *name,
    prison_service_create_t create,
    prison_service_destroy_t destroy);
void prison_deregister(struct prison_service *psrv);

void prison_service_data_attach(struct prison_service *psrv,
    struct prison *pr, void *data);
void *prison_service_data_get(struct prison_service *psrv,
    struct prison *pr);
void *prison_service_data_detach(struct prison_service *psrv,
    struct prison *pr);
```

How it works?

- calls 'create' method for every existing jail on 'register' and on new jails creation
- calls 'destroy' method for all jails going away and for all existing jails on 'deregister'
- allows to attach some service-specific data to each jail and then operate on it when needed

Why not ZFS internal?

- because can be used for other things as well, like per-jail sysvipc name spaces, per-jail network interfaces (just an example, hope to see vimage soon)