



TCP connectionの保存と復元

筑波大学ソフトウェア研究室
三戸健一 (@mittyorz)

2012年9月22日

Kernel/VM探検隊@つくば



自己紹介

- 筑波大学情報科学類4年生
- ソフトウェア研究室
- 好きな言語はPerl
 - 最近全然触れていない
- @go_vmさんのVMRPCを使って面白いことしよう
- 普段は環境構築とかメンテナンスとかばかりでhacker的なことはあまりしていません

TCPソケットの状態を保存しよう

- 理想:
 - TCPコネクションを維持したままプロセスの停止、状態の保存、別の環境で再開、が簡単にできるとうれしい
- 現実:
 - 送受信キューとかシーケンス番号などの情報を読み取ってdumpしないといけない
 - close()するとFINが送信されてしまう

TCPスタックをちゃんと理解して自力でやるのは大変そう...。なんとかインテキできんのか



従来の解決法 仮想計算機を使う

- 難しいことを考えないでVMに載せてしまう
 - ライブマイグレーション！
- 利点
 - ノウハウが十二分に存在している
 - コードに手を入れる必要は無い
- 欠点
 - 仮想計算機を稼働させるのはそれなりに大変
 - OSごと環境を移動させるので、コストが高い



TCP repair mode

- 先生「3.5 kernelなら簡単らしいよ」
- 私「」

- Linux Kernel 3.5で新しくマージされた機能
- TCP_REPAIRというソケットオプションが新設された
- Repairモードに入ると、副作用無しに様々な操作が行えるようになる
- Software Design 2012年9月号
 - 「続・Linux 3.5の新機能」



Repair modeで出来ること

- connect()を用いて、実際には接続していてもESTABLISHEDにする
- ポートの衝突を無視してbind()でLISTEN
- FIN_WAIT, TIME_WAITを経由せずにclose()
 - FINパケットは送信されない
- TCPシーケンス番号、送受信キューなどソケットの再開に必要な情報の取得



ソケットの保存

1. iptables で新規パケットをブロック
 - `iptables -D INPUT -s 192.168.10.222/32 -d 192.168.10.102/32 -p tcp -m tcp --sport 58085 --dport 12345 -j DROP`
2. repair modeに移行
 - `setsockopt(sockfd, SOL_TCP, TCP_REPAIR, ...);`
3. 接続情報の取得/パケットキューの読取
4. `close()`



ソケットの復元

1. sock = socket(...);
2. repair modeに移行
3. シーケンス番号の復元
4. bind()
5. connect()
6. TCPオプションの復元
7. キューの復元
8. repair modeを終了
9. 通信のブロックを解除



実装例

- 間に合いませんでした

.....だけだと哀しいので



crtools

- an utility to checkpoint/restore a process tree
 - http://criu.org/CR_tools
 - “CRIU is sub-project of OpenVZ”
- TCPセッションだけでなく、プロセス自体を freeze/dump to file/restore出来る
- 有名なOSS projectでcheckpoint/restore出来るか試している模様
 - http://criu.org/What_software_is_supported



crtoolsのbuild

- Ver 0.2が二日前(09/20)に出てますが、試したのは0.1
- Ubuntu 12.10 beta x86_64
- `$ git clone git://git.criu.org/crtools.git`
- `$ make`
 - 色々warningとかerrorとか出るが、2,3カ所適当に直すとbuild出来た



保存、復元の対象

- echo-server-nofork-fdopen.c
 - サーバ: Ubuntu 12.10 (192.168.10.102)
 - クライアント: Debian 6.0 (192.168.10.222)
- 単純なechoサーバ
 - selectもforkもしないので、一つのconnect

- mitty@quantal:~/tcp-repair\$ gcc echo-server-nofork-fdopen.c
- mitty@quantal:~/tcp-repair\$./a.out 12345
- mitty@quantal:~\$ sudo lsof -i | grep 12345
a.out 12327 mitty 3u IPv4 18417 0t0 TCP *:12345 (LISTEN)

connect to server

- telnetコマンドでechoサーバへ接続

- debian\$ telnet 192.168.10.102 12345

- mitty@quantal:~\$ sudo lsof -i | grep 12345

```
a.out 12327 mitty 3u IPv4 18417 0t0 TCP *:12345 (LISTEN)
a.out 12327 mitty 4u IPv4 18418 0t0 TCP 192.168.10.102:12345->192.168.10.222:58085 (ESTABLISHED)
a.out 12327 mitty 5u IPv4 18418 0t0 TCP 192.168.10.102:12345->192.168.10.222:58085 (ESTABLISHED)
```

- ファイルディスクリプタ

- 3でaccept
- 4 で clientへwrite
- 5 = dup(4) を用いてclientからread

dump tcp connection to files

- mitty@quantal:~\$ sudo ./crttools/crttools **dump** --tcp-established -t 12327
Error (libnetlink.c:44): ERROR 2 reported by netlink
- mitty@quantal:~\$ ls -l *.img
-rw-r--r-- 1 root root 466 Sep 6 16:11 core-12327.img
-rw-r--r-- 1 root root 58 Sep 6 16:11 creds-12327.img
-rw-r--r-- 1 root root 4 Sep 6 16:11 eventfd.img
-rw-r--r-- 1 root root 4 Sep 6 16:11 eventpoll.img
- エラーは出ているが、dump出来た
- mitty@quantal:~\$ sudo iptables-save | grep 12345
-A INPUT -s 192.168.10.222/32 -d 192.168.10.102/32 -p tcp -m tcp --sport 58085 --dport 12345 -j DROP

restore from files

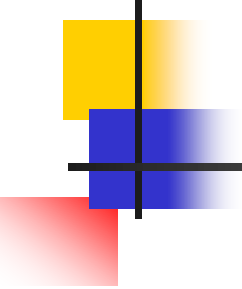
- mitty@quantal:~\$ sudo ./crtools/crtools **restore** --tcp-established -t 12327
- mitty@quantal:~\$ sudo lsof -i | grep 12345

```
a.out  12327 mitty  3u  IPv4  21686      0t0  TCP *:12345 (LISTEN)
a.out  12327 mitty  4u  IPv4  21688      0t0  TCP 192.168.10.102:12345-
>192.168.10.222:58085 (ESTABLISHED)
a.out  12327 mitty  5u  IPv4  21688      0t0  TCP 192.168.10.102:12345-
>192.168.10.222:58085 (ESTABLISHED)
```
- 復元された
- iptablesの設定はdeleteしてくれない
 - mitty@quantal:~\$ sudo iptables -D INPUT -s 192.168.10.222/32 -d 192.168.10.102/32 -p tcp -m tcp --sport 58085 --dport 12345 -j DROP



既知の問題

- なぜか0, 1, 2のファイルディスクリプタが復元されない
 - クライアントが再接続すると、0や1のfdが再利用されて、サーバのstdoutがクライアントに接続されるという楽しいことになる



ご静聴ありがとうございました