ADB (Android Debug Bridge): How it works?

2012.2.6 early draft

Tetsuyuki Kobayashi

Let's talk about inside of Android.

http://www.kmckk.co.jp/eng/kzma9/
http://www.kmckk.co.jp/eng/jet_index.html

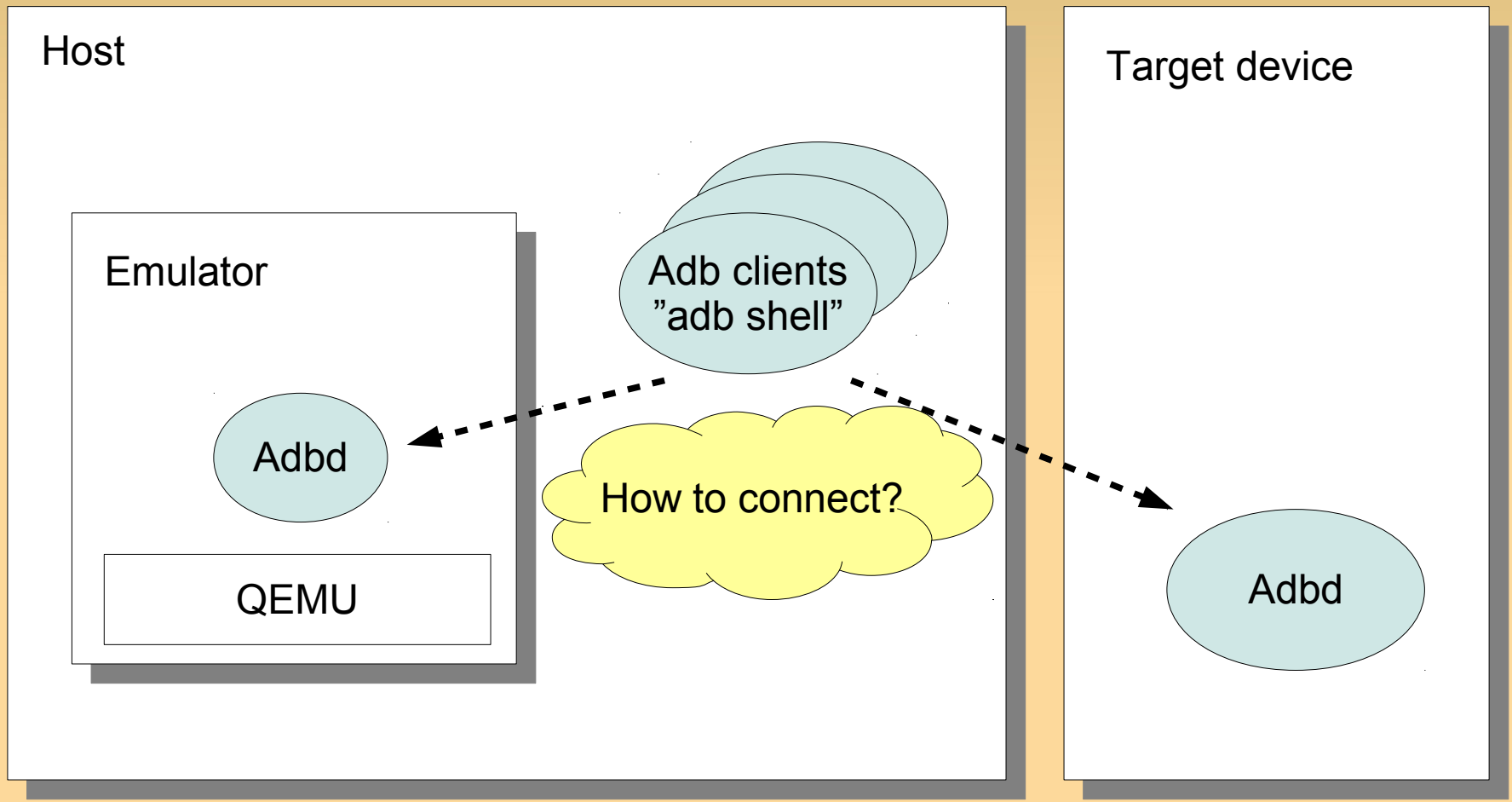
Who am I?

- 20+ years involved in embedded systems
  - 10 years in real time OS, such as iTRON
  - 10 years in embedded Java Virtual Machine
  - Now GCC, Linux, QEMU, Android, …
- Blogs
  - http://d.hatena.ne.jp/embedded/ (Personal)
  - http://blog.kmckk.com/ (Corporate)
  - http://kobablog.wordpress.com/(English)
- Twitter
  - @tetsu_koba

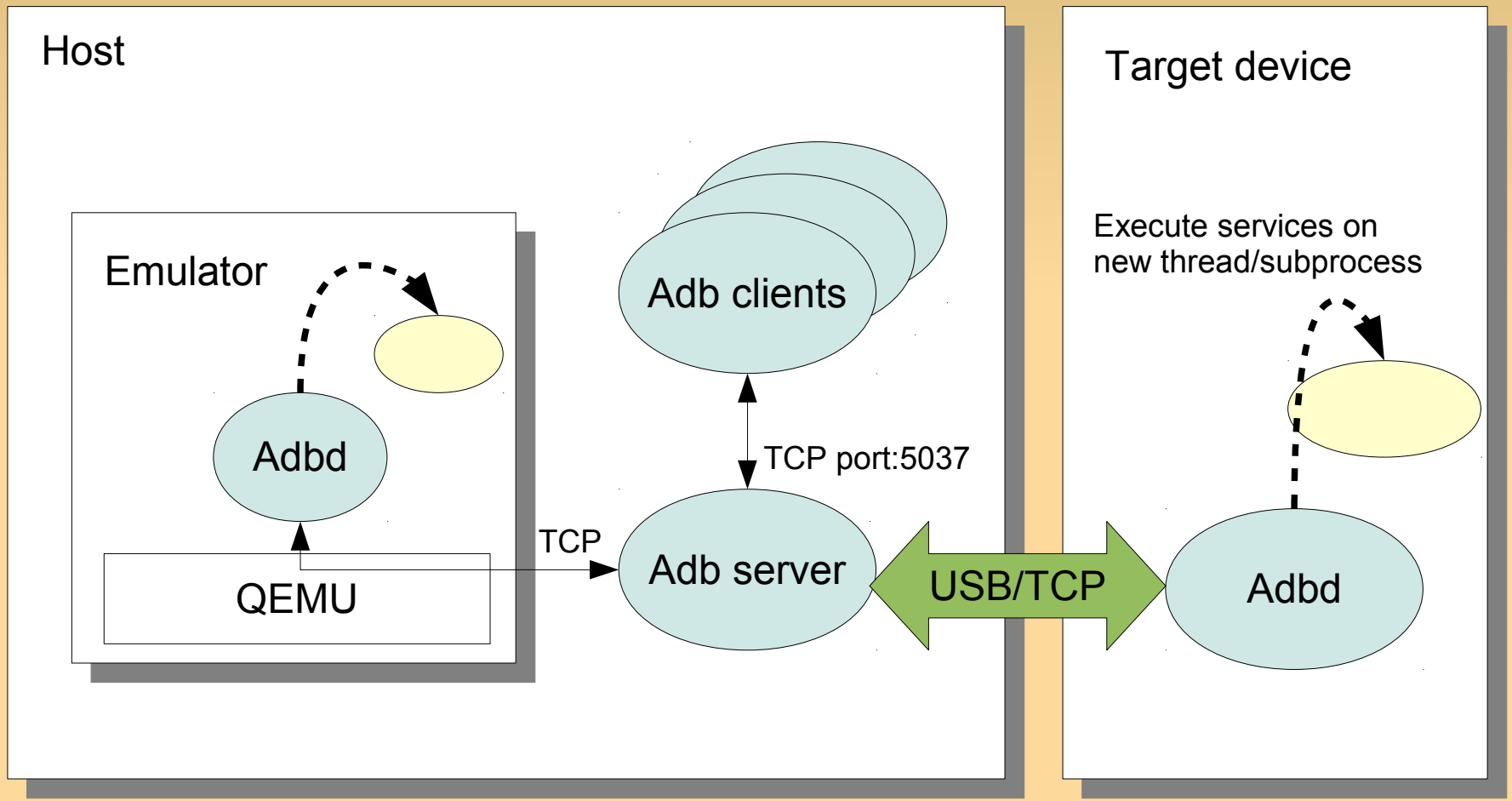
What is ADB?

- If you are an Android builder, you have used ”adb logcat”, ”adb shell”
- Even if you only use DDMS in Eclipse, adb is working under the hood.
- Using adb, you can connect emulator or actual target device.
- ”adb kill-server”? what does it mean?

How to connect?



ADB overview



2 roles of ADB

- Providing ”Transport”
  - communication path between host and target device
  - USB or TCP: but clients don't have to aware
- Providing ”Services”
  - executing something on the target devices through the transport.
    - ”adb shell” for executing command
    - ”adb push/pull” for file transfer

3 elements of ADB

- adb clients
  - executable with subcommand
  - ”adb shell”, ”adb logcat” : the end point of host side
- adb server
  - running on host on back-ground
  - act as proxy between adb clients and adbd
- adb daemon (adbd)
  - running on target device
  - started by init, if die, restarted by init again

When does adb server start?

- Explicitly, ”adb start-server”
  - It starts adb server as back ground process.
- Usually it does automatically on demand. You don't have to do ”adb start-server”.
- When you want to restart adb server, do ”adb kill-server”
- Actually, adb clients and adb server shares same executable
  - ”adb start-server” equals ”adb fork-server server &”

ADB internal

- Source code
- How to get ADB logs
- Sequence chart
- Simple ruby script to connect adb server
- Command details
- Secure mode
- Add USB Vendor ID
- Switching transport mode

Source code

- system/core/adb in Android source tree
- From this directory adb and adbd are built
  - Don't confuse.
  - common files between adb and adbd
    - adb.c, fdevent.c, transort.c, transport_local.c, tansport_usb.c, service.c, sockets.c, util.c

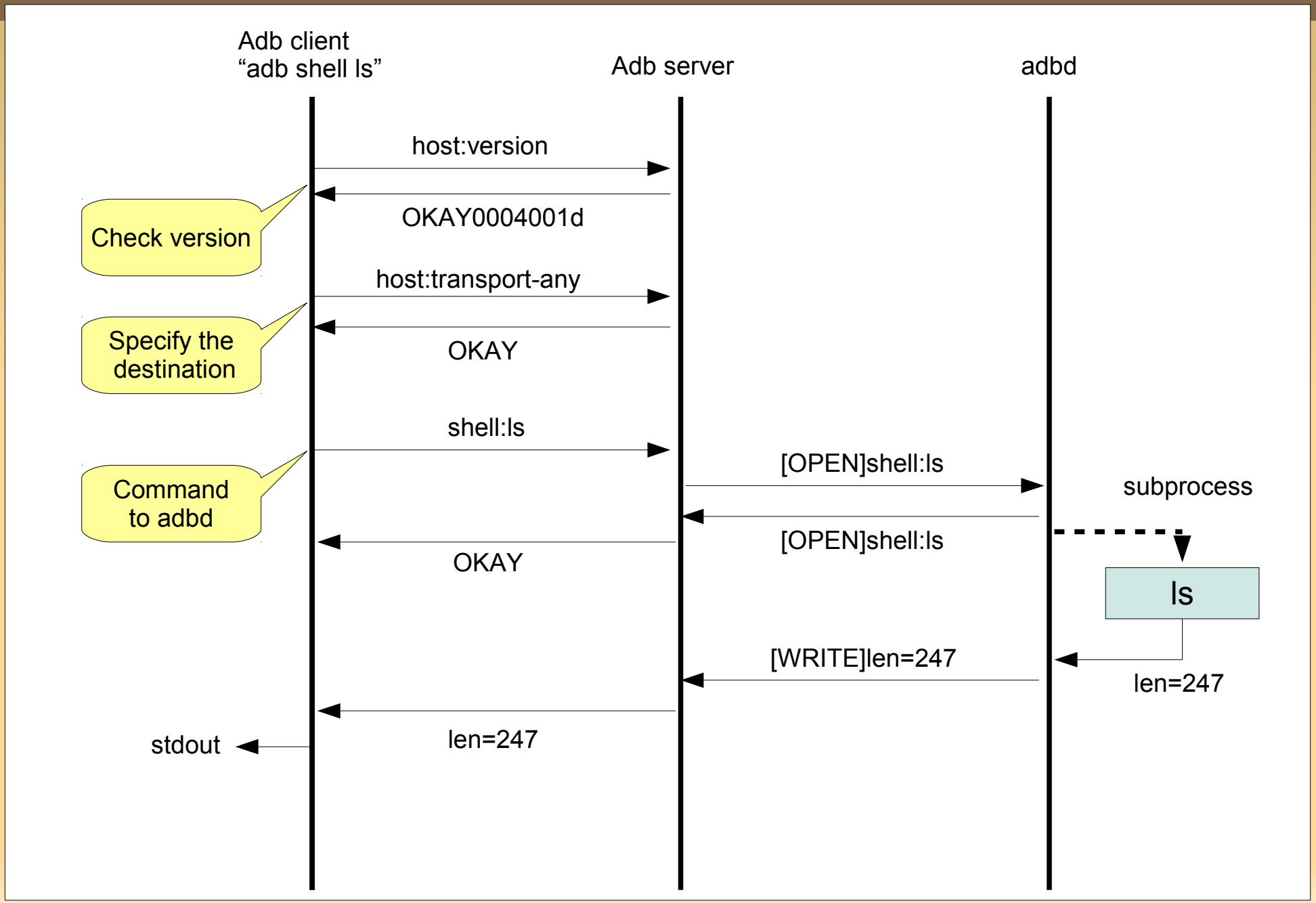
```
#if ADB_HOST
    /* code for adb*/
#else
    /* code for adbd */
#endif
```

- files only for adbd
  - backup_service.c, file_sync_service.c, jdwp_service.c, framebuffer_service.c, remount_services.c, usb_linux_clients.c, log_service.c
- files only for adb
  - console.c, adb_clients.c, file_sync_client.c, usb_vendors.c, get_my_path_{linux,darwin,windows,freebsd}.c, usb_{linux,macos,libusb,windows}.c

How to get ADB logs

- For adb clients and adb server, set environment variable ADB_TRACE
- For adbd, set system property ”persist.adb.trace_mask”
- http://blog.kmckk.com/archives/4080002.html

Sequence chart



Simple ruby script to connect to adb server

```
require 'socket'

def error_exit
  puts "Error"
  exit 1
end

def send_to_adb(s, msg)
  s.printf("%04x%s",
msg.length, msg)
end

def check_okay(s)
  (s.read(4) == "OKAY")
end

def check_version(s)
  (s.read(12) ==
"OKAY0004001d")
end
```

```
hostname = 'localhost'
port = 5037

s = TCPSocket.open(hostname, port)
send_to_adb(s, "host:version")
error_exit if ! check_version(s)
s.close

s = TCPSocket.open(hostname, port)
send_to_adb(s, "host:transport-any")
error_exit if ! check_okay(s)
send_to_adb(s, "shell:ls")
error_exit if ! check_okay(s)

while line = s.gets
  puts line.chop
end
s.close
```

change ”shell:ls” as you like.

http://blog.kmckk.com/archives/4092210.html

Command details

- adb logcat
- adb install/uninstall
- adb reboot
- screen capture from DDMS

Secure mode

- Android smart phone products have adbd. Usually it runs on secure mode. (secure = 1)
- if secure == 1, change adbd as SHELL user(= not privileged), else it keeps running as root user
- In secure mode, all services invoked by adbd ran as SHELL user. Some causes ”permission denied”.

How secure mode decided

- Running on emulator → secure = 0
- System property ”ro.secure” == 1 → secure = 1
  - if ”ro.debuggable” == 1, you can restart adb unsecure by ”adb root”
- All Android phone products are shipped in ”ro.secure” = 1, ”ro.debuggable” = 0.
- See adb.c: adb_main

Add USB Vendor ID

- When connecting USB device, adb checks USB Vendor ID
- Many USB Vendor IDs are hard coded in adb. (But not enough)
- To add USB Vendor ID, make ”$HOME/.android/adb_usb.ini” and write one ID in one line
- See usb_vendors.c

Switching transport mode

Switching USB mode to TCP mode

```
$ adb shell netcfg
lo       UP    127.0.0.1       255.0.0.0       0x00000049
eth0     UP    192.168.1.139   255.255.255.0   0x00001043
$ adb tcpip 5555
restarting in TCP mode port : 5555
$ adb devices
List of devices attached

$
```

disconnected from USB. Then restart adb server with specifying target IP address.

```
$ adb kill-server
$ ADBHOST=192.168.1.139 adb devices
* daemon not running. starting it now on port 5037 *
* daemon started successfully *
List of devices attached
emulator-5554   device

$
```

Switching transport mode (What happen inside?)

- See service.c restart_tcp_service
- property_set(”service.adb.tcp.port”, value);
  - Note: before Android 4.0, this cause ”permission denied” in secure mode and ignored silently!
- After that, exit(1);
  - init restarts adbd.
  - ”service.adb.tcp.port” is checked in adb.c adb_main

Tips

- adb emu
- adb backup/restore
- Joke commands
- Modify emulator to allow any server socket

adb emu

- You can send a single command to emulator console easily
  - send only. can not receive.
  - For emulator console,
    - http://developer.android.com/guide/developing/devices/emulator.html
- Simple example. ”adb emu window scale 0.5” after starting emulator
- http://blog.kmckk.com/archives/4091258.html

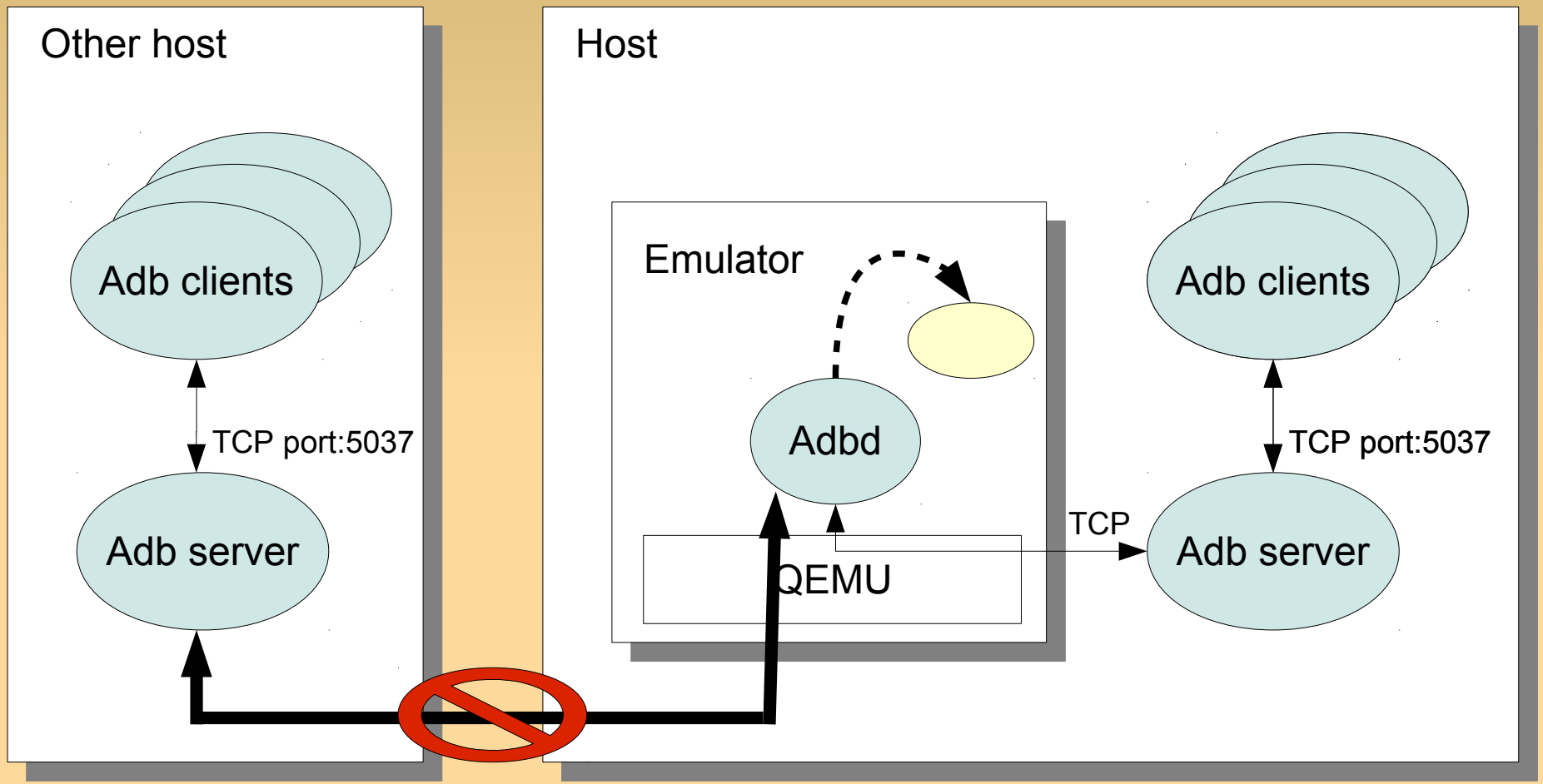
adb backup/restore

- New in Android 4.0
- You can backup/restore installed applications with their saved status.

Joke commands

- adb hell
  - same as ”adb shell” except ”hell” color :)
  - Just try.
- adb lolcat
  - same as ”adb logcat”

Modify emulator to allow any server socket



All server sockets in Android emulator accepts only from localhost.
If you feel inconvenient in this restriction, apply the patch in next page.

http://blog.kmckk.com/archives/3882865.html

Patch to allow any server socket (for experience)

```
diff --git a/slirp-android/socket.c b/slirp-android/socket.c
index 439590a..ed16d5a 100644
--- a/slirp-android/socket.c
+++ b/slirp-android/socket.c
@@ -650,7 +650,7 @@ solisten(u_int port, u_int32_t laddr, u_int lport, int
flags)
     so->so_laddr_ip   = laddr; /* Ditto */
     so->so_haddr_port = port;

-    s = socket_loopback_server( port, SOCKET_STREAM );
+    s = socket_inaddr_any_server( port, SOCKET_STREAM );
     if (s < 0)
         return NULL;

diff --git a/sockets.c b/sockets.c
index 1063339..55b5d57 100644
--- a/sockets.c
+++ b/sockets.c
@@ -1337,6 +1337,11 @@ socket_in_client( SockAddress*  to, SocketType  type )
     return socket_connect_client( s, to );
 }

+int
+socket_inaddr_any_server( int  port, SocketType  type )
+{
+    return socket_in_server( INADDR_ANY, port, type );
+}

 int
 socket_loopback_server( int  port, SocketType  type )
```

Advanced Topics

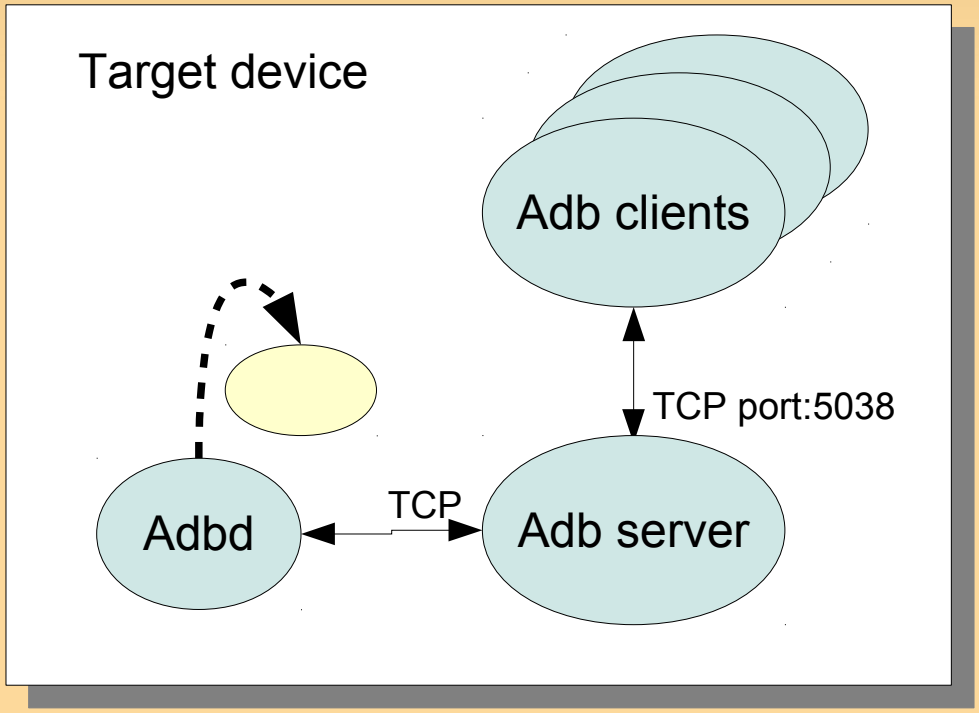
- adb in Android device
- Port adbd to other than Android

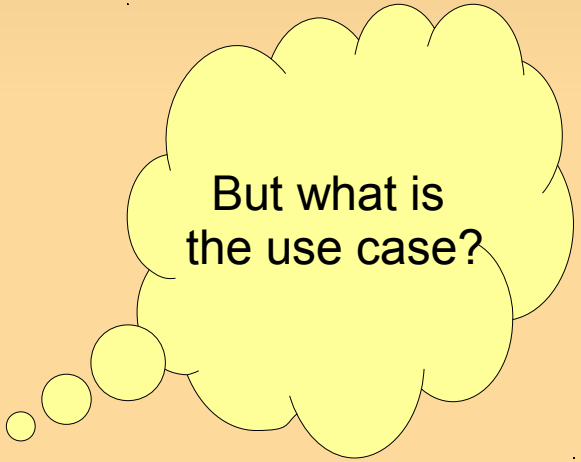
adb in Android device

- Usually, adb is running on the host side, such as Linux, MacOS and Windows
- In Android 4.0 there is /system/bin/adb in Android file system
- What for is this?

Connect its own adbd by adb

- Restart adbd in TCP mode
- Then type ”adb devices”
- It can connect its own adbd by local loop back





http://blog.kmckk.com/archives/4092970.html

Connect other Android device by adb on Android

NexusOne
(Android 2.3.6)

USB device
(micro B connector)

USB Host
(A connector)

KZM-A9-Dual board
(Android 4.0.3)

http://blog.kmckk.com/archives/4094716.html

At the serial console on KZM-A9-Dual board

```
# adb devices
* daemon not running. starting it now on port 5038 *
* daemon started successfully *
List of devices attached
HT015P803242    device

#
```

It worked!
You can do ”adb shell”, ”adb logcat”, too.

Even you can install application from the board to NexusOne by ”adb install foo.apk”.

Port adbd to other than Android

- Quick hack!
- Consider dependency for better porting

Quick hack!

- /sbin/adbd is statically linked executable.
- Just copy this file to ARM Ubuntu 11.10 just for experience (using Android patched kernel)
- Somehow, it worked without any recompilation
  - sudo chmod 666 /dev/android_adb*
  - make symbolic link /bin/sh to /system/bin/sh
  - adbd runs in secure mode
  - It worked ”adb shell”, ”adb push/pull”
- http://blog.kmckk.com/archives/4093193.html

Consider dependency for better porting

- Some service requires other android commands
  - adb install/uninstall, adb bugreport
  - framebuffer service invokes /system/bin/screencap
- Adbd uses Android property system
  - At the binary experiment, all property_get returns default values.
    - That's why adbd ran in secure mode.
- Switching adbd mode assumed that init restarts adbd process

Q & A

Thank you for listening!
Any comments to blogs are welcome.