Linux on Android

Based on this, which is based from this.

I want to modify it so that it doesn't rely on debian... made possible mainly by Aboriginal Linux.

Download Requirements

- Firstly, get busybox binary from here - use *busybox-armv5l*
- Get root filesystem from here
- **(optional)** Get ssh (dropbear) binary from here

Prepare Linux Image

- Create 500MB (as needed) image file

```
dd if=/dev/zero of=linux.img bs=1024 count=500K
```

- Format for ext2 filesystem

```
mke2fs -b 1024 -m 0 -F linux.img
```

- Loop-mount the image

```
mkdir -p loopfs ; mount -o loop linux.img loopfs
```

- Extract root filesystem (as root)

```
tar -xvf root-filesystem-armv5l.tar.bz2 ; cp -a root-filesystem-
armv5l/* loopfs/
```

- Optionally, copy ssh (dropbear) utility

```
cp dropbearmulti-armv5l loopfs/bin/{dropbear,ssh}
```

- Unmount and you get yourself a system image in linux.img!

```
umount loopfs ; rmdir loopfs
```

Linux Start-Up Script

linux.sh

```
#!/system/busybox/ash

THISPATH=${THISPATH:="$(cd $(dirname $0);pwd)"}

export BUSYPATH=${BUSYPATH:="/system/busybox"}
export SYSIMAGE=${SYSIMAGE:="$THISPATH/linux.img"}
export SYSMOUNT=${SYSMOUNT:="$THISPATH/linux"}
export SYSSHELL=${SYSSHELL:="/bin/ash"}
```

```
export PATH=$BUSYPATH:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
export TERM=linux
export HOME=/root

DO_VERBOSE=${DO_VERBOSE:="yes"}
DO_MOUNTSYS_RW="no"
DO_MOUNT_LINUX="yes"
DO_NET_SETUP="yes"
DO_CHROOT_LINUX="yes"
DO_NET_CLEANUP="yes"
DO_UMOUNT_LINUX="yes"
DO_MOUNTSYS_RO="no"
DO_REMOVE_SYSMOUNT="no"

# task in functions

must_be_root()
{
    [ "$(id -u)" -ne 0 ] && echo "Must be root! Aborting!" && exit 1
}

mountsys_rw()
{
    [ "$DO_VERBOSE" == "yes" ] && echo "Setup /system RW access..."
    mount -o remount,rw /system
}

mountsys_ro()
{
    [ "$DO_VERBOSE" == "yes" ] && echo "Reset /system RO access..."
    mount -o remount,ro /system
}

mount_linux()
{
    [ "$DO_VERBOSE" == "yes" ] && echo "Mounting Linux Image..."
    [ ! -d $SYSMOUNT ] && DO_REMOVE_SYSMOUNT="yes" && mkdir $SYSMOUNT
    mount -t ext2 -o loop,noatime,nodiratime $SYSIMAGE $SYSMOUNT
    #mount --bind /mnt/sdcard $SYSMOUNT/mnt/sdcard
    #mount --bind /mnt/sdcard2 $SYSMOUNT/mnt/sdcard2
}

umount_linux()
{
    [ "$DO_VERBOSE" == "yes" ] && echo "Un-mounting Linux Image..."
    #umount $SYSMOUNT/mnt/sdcard
    #umount $SYSMOUNT/mnt/sdcard2
    umount $SYSMOUNT
    [ "$DO_REMOVE_SYSMOUNT" == "yes" ] && rmdir $SYSMOUNT
}
```

```
net_setup()
{
    [ "$DO_VERBOSE" == "yes" ] && echo "Setting Up Networking..."
    sysctl -w net.ipv4.ip_forward=1 >/dev/null
    # basic settings - using google public dns ipv4 address
    echo "nameserver 8.8.8.8" > $SYSMOUNT/etc/resolv.conf
    echo "nameserver 8.8.4.4" >> $SYSMOUNT/etc/resolv.conf
    echo "127.0.0.1 localhost" > $SYSMOUNT/etc/hosts
}

net_cleanup()
{
    [ "$DO_VERBOSE" == "yes" ] && echo "Cleaning Up Networking..."
    sysctl -w net.ipv4.ip_forward=0 >/dev/null
}

chroot_linux()
{
    mount --bind /proc $SYSMOUNT/proc
    mount --bind /sys $SYSMOUNT/sys
    echo "Entering Linux CHROOT..."
    echo
    echo "If this is your first time using this image,"
    echo "run 'mdev -s' to create /dev nodes!"
    echo
    chroot $SYSMOUNT $SYSSHELL
    echo
    echo "...exiting Linux CHROOT!"
    umount $SYSMOUNT/proc
    umount $SYSMOUNT/sys
}

# check parameter

while [ "$1" != "" ]; do
    case $1 in
        --skip-cleanup)
            DO_NET_CLEANUP="no"
            DO_UMOUNT_LINUX="no"
            DO_MOUNTSYS_RO="no"
            ;;
        --startup-only)
            DO_CHROOT_LINUX="no"
            DO_NET_CLEANUP="no"
            DO_UMOUNT_LINUX="no"
            DO_MOUNTSYS_RO="no"
            ;;
        --cleanup-only)
            DO_MOUNTSYS_RW="no"
            DO_MOUNT_LINUX="no"
            DO_NET_SETUP="no"
```

```
            DO_CHROOT_LINUX="no"
            ;;
        --quiet) DO_VERBOSE="no" ;;
        *) echo "Unknown parameter '$1'!" ; exit 1 ;;
    esac
    shift
done

# do your thing!

must_be_root
[ "$DO_MOUNTSYS_RW" == "yes" ] && mountsys_rw
[ "$DO_MOUNT_LINUX" == "yes" ] && mount_linux
[ "$DO_NET_SETUP" == "yes" ] && net_setup
[ "$DO_CHROOT_LINUX" == "yes" ] && chroot_linux
[ "$DO_NET_CLEANUP" == "yes" ] && net_cleanup
[ "$DO_UMOUNT_LINUX" == "yes" ] && umount_linux
[ "$DO_MOUNTSYS_RO" == "yes" ] && mountsys_ro

exit 0
```

Pre-Image Work

- Get Aboriginal Linux system-image for armv5l from here
- Get lfs-bootstrap image for armv5l from there as well
- Make sure you have QEMU installed in your system
- Extract the system image - run dev_environment.sh (requires QEMU)
- Exit and you should find hdb.img
- Loop mount the image file and copy lfs-bootstrap image and git source tarball into it
- Re-run dev_environment.sh
- ...
- To compile git, use *./configure –prefix=/usr/local –with-python= CFLAGS="${CFLAGS} -static" NO_GETTEXT=1*
- *to be continued*
- **(optional)** Get git source code (also, get an armv5l system image and lfs-bootstrap)