Inisialisasi Servis Pada Sistem Tertanam (*Embedded System)* Berbasis Prosesor Arm Dan Kernel Linux Dengan Proses *Init.d*

Zainal Husin[1]
[1]Jurusan Teknik Elektro, Fakultas Teknik
Universitas Sriwijaya
Inderalaya, Indonesia
Penulis korespondensi: zaenalhusin@gmail.com

***Abstrak*—Sejak diperkenalkan oleh Linus Torvalds, bahasa pemograman Linux dengan sifatnya yang terbuka, telah dikembangkan oleh begitu banyak pihak secara bersama-sama. Pihak-pihak ini adalah para akademisi, industri maupun perorangan dengan bakat dan keahlian yang luar biasa, dan dengan berbagai berkepentingan. Hasilnya adalah sesuatu yang sangat besar, baik pada segi kapasitas maupun kwalitas. Dengan kapasitas yang begitu besar, tidaklah semua terpakai pada suatu rancangan produk. Sehingga menjadi salah satu karakteristik Linux, fleksibilitas. Yaitu Sistem Operasi dapat diubah sesuai kebutuhan. Kernel Linux yang banyak dipakai sekarang pada beberapa embedded systems berbasis prosesor ARM, yaitu sebagai penggerak robot, alat-alat komunikasi dan sebagainya, dengan sendirinya menuruni sifat fleksibilitas ini. Maka terkadang pengguna masih memerlukan menambahkan atau mengubah isi kernel Linux tersebut, misalnya dengan salah satu tujuan: sudah terjadinya inisialisasi service tertentu setelah 'booting'.**

Kata kunci*— booting, embedded systems, linux, inisialisasi servis, prosesor ARM*

Abstract—Since its introduction by Linus Torvalds, Linux programming language with its open nature, has been developed by so many parties together. These parties are academics, industry and individuals with extraordinary talents and expertise, and with various interests. The result is something very big, both in terms of capacity and quality. With such a large capacity, not all of it is used in a product design. So that becomes one of the characteristics of Linux, flexibility. Namely the Operating System can be changed as needed. The Linux kernel which is widely used today in several ARM processor-based embedded systems, namely as a robot driver, communication tools and so on, naturally decreases this flexibility. So sometimes users still need to add or change the contents of the Linux kernel, for example with one goal: there has been a certain service initialization after 'booting'.

Keywords— booting, embedded systems, linux, initialization services, processor ARM

I. PENDAHULUAN

Dengan pesatnya pemakaian sistem tertanam yang berbasis *kernel Linux*, baik yang dapat kita jumpai sehari-hari seperti *smartphone*, dalam dunia pendidikan seperti pemograman gerak robot serta alat-alat perkantoran maupun rumah tangga yang telah memanfaatkan teknologi sistem tertanam berbasis *kernel Linux*. Kemampuan pemograman dengan *Linux* yang *powerful*, dibarengi dengan kemampuan fabrikasi memproduksi prosesor-prosesor 32 bit dengan harga yang tidak terlalu berbeda dengan prosesor generasi terdahulu, 8 dan 16 bit telah membuat pemakaian sistem tertanam berbasis *kernel Linux* begitu meluas. *Start-up* suatu komputer berbasis *kernel Linux* yang dikenal juga dengan '*booting*' berawal dengan memuat *kernel* ke memori dan meng-eksekusinya. Kemudian beberapa variasi tugas-tugas inisialisasi dimulai dan kemudian sistem diserahkan ke pemakai untuk dipergunakan. Tugas-tugas inisialisasi ini disimpan berupa proses dalam *folder /system/etc/ini.d*, sedangkan proses-proses ini disusun dalam suatu script yang dinamakan *init scripts* dan dijalankan oleh init secara berurutan. Penulisan dan cara-cara pelaksanaan *init script* ini yang dibahas dalam penelitian ini. *Init Scripts* sebenarnya juga hanyalah *Shell Scripts* biasa, hanya dieksekusi pada saat '*booting*' karena diperlukan untuk pelaksaanaan proses selanjutnya. Proses yang telah dimulai ini dapat saja tetap berlanjut walaupun proses '*booting*' telah selesai. Init adalah proses pertama yang berjalan setelah '*booting*' dan menjadi dasar semua proses pengguna dan beberapa proses sistem. Beberapa level dimana init ini berjalan, yaitu Level 0 pada saat sistem mati, Level 1 sebagai representasi pemakai tunggal, Level 2 sampai 5 adalah dukungan jaringan dan Level 6 adalah level untuk 'reboot'. Seorang Adminitrator dapat saja meng-intervensi dengan menjalankan *init script* ini secara manual.

Ada beberapa permasalahan yang terjadi yakni : yang pertama adalah *Kernel Linux* bersifat fleksibel, dapat dipergunakan dalam besaran menurut kebutuhan yang diperlukan saja. Oleh sebab itu setiap peralatan yang berjalan diatas kernel Linux biasa tidak ada yang sama. Selalu terjadi kelebihan atau kekurangan antara suatu peralatan dengan peralatan lainnya, yang kedua yaitu Untuk kebutuhan pengguna yang berbeda-beda pula, maka diperlukan suatu perubahan pada *kernel* itu sendiri. Cara pertama yaitu dengan membongkar dan kemudian menyusun kembali *kernel* tersebut setelah dilakukan perubahan yang diperlukan. Tetapi cara ini dianggap sangat

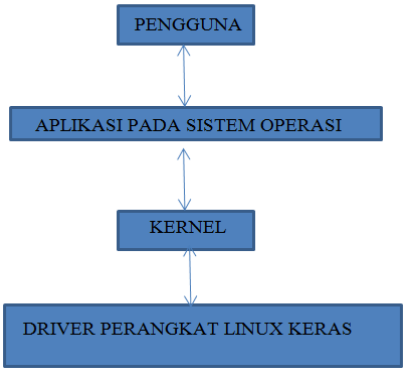
sulit dan dapat berakibat peralatan itu malahan tidak dapat *boot* sama sekali (*bricked*), sedangkan yang ketiga adalah Suatu susunan perintah pada suatu script yang ditempatkan pada folder init.d yang akan dieksekusi pada saat boot, diharapkan telah dapat menggantikan proses pembongkaran dan penyusunan kernel tersebut.

II. TINJAUAN PUSTAKA

A. *Kernel Linux*

First, Pada awal perkembangan Mikroprosesor, bahasa Assembly dipakai sebagai bahasa pemograman oleh para pengembang aplikasi. Sebagai contohnya, Assembly Z80 yang dipergunakan pada prosesor buatan Zylog. Bahasa Assembly ternyata menawarkan portabilitas yang rendah terhadap berbagai arsitektur prosesor buatan pabrik lainnya, seperti antara buatan Intel dan Motorola misalnya. Linus Torvalds menawarkan suatu perangkat lunak yang mengatur proses dan driver perangkat keras yang memiliki portabilitas yang tinggi, yang dinamakan Kernel Linux. Kernel ini pada awalnya dibuat untuk arsitektur x86 dari Intel, dan kemudian berhasil dibuat menjadi portable untuk arsitektur lainnya seperti Motorola, ARM dan sebagainya.

Kernel ini sendiri bukanlah suatu Sistem Operasi yang lengkap, sehingga tanpa dilengkapi dengan perantara yang berupa aplikasi yang berjalan pada Sistem Operasi, maka kernel ini belum dapat dipergunakan oleh pengguna. Sebagai digambarkan pada Gambar 1, Kernel berada diantara Aplikasi dan Driver perangkat keras.

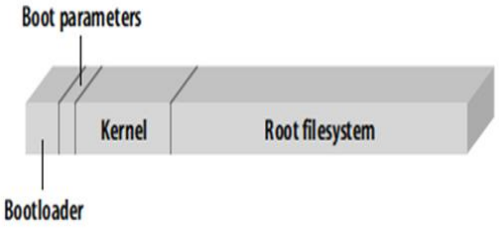


Gambar 1. Kernel masih memerlukan aplikasi yang berjalan pada Sistem Operasi sebagai perantara ke pengguna

Dari Gambar 1 tersebut, ditunjukkan bahwa Driver perangkat keras yang diatur kernel antara lain berupa memori, LCD. USB, keyboard dan sebagainya. Aplikasi sendiri dapat berjalan pada platform Android, Windows CE, dan Shell Script, distribusi Linux seperti Debian, Fedora, RedHat dan Ubuntu. Sistem Operasi Linux terpecah menjadi beberapa distribusi karena sifat dari Open Source yang membuat para pengembang bergerak keberbagai arah. Suatu distribusi, Ubuntu misalnya dapat berjalan pada arsitektur x86, misalnya dengan kernel versi 2.60. Dalam penelitian ini, Sistem Operasi yang dipilih adalah Android. Google, sebagai pemilik Android tidak membuat kernel tersendiri untuk Sistem Operasi Android. Alasannya kernel Linux bersifat terbuka (‘open source’) sehingga nantinya tidak terganjal oleh gugatan hak cipta. Google sangatlah berhati-hati berantisipasi dengan gugatan hak cipta, seperti tercermin dari Android, yang tidaklah secara mentah-mentah memakai Java sebagai bahasanya, walaupun pada saat Android dikembangkan, Java masih bersifat open source. Hal ini terbukti dengan dibelinya Java oleh Oracle, yang kemudian menggugat Android atas kemiripannya dengan Java. Alasan lainnya adalah karena sifat ‘powerful’ nya tadi, dianggap akan dapat mengimbangkan perkembangan perangkat keras, yang pada masa kini telah tidak asing lagi tersedianya prosesor multi cores sampai dengan 4 ataupun 8.

B. *Sistem Tertanam (Embedded Systems)*

Secara sederhana, suatu sistem embedded dapat didefinisikan sebagai suatu alat yang mempunyai komputer pada alat itu sendiri. Suatu embedded Linux dapat digambarkan sebagai suatu susunan: Bootloader, Kernel dan Filesystems. Bootloader adalah bagian pertama yang berfungsi setelah POWER ON, berfungsi pada inisialisasi perangkat keras dan menyiapkan kode-kode start up kernel. Ada beberapa jenis bootloader, sebagai contoh GRUB untuk arsitektur x86 dan UBOAT untuk arsitektur ARM. Bootloader ditempatkan pada harddisk atau NANDFLASH seperti terlihat pada gambar 2. Nandflash adalah memori padat (solid state) yang harus dibaca secara berurutan (bukan acak), jadi tidak dapat dipergunakan sebagai RAM (Random Access Memory) dan disebut juga ROM (Read Only Memory). Karena Sistem Operasi ditempatkan pada ROM, maka terjadi suatu kesalahkaprahan yaitu Sistem Operasi itu dinamakan juga ROM. Sebagai bandingannya, pada Sistem Operasi Microsoft Windows yang berjalan pada laptop atau PC yang berfungsi sebagai Nandflash adalah Harddisk dan juga Solid State Drive (SSD) yang sudah banyak dipergunakan sekarang.



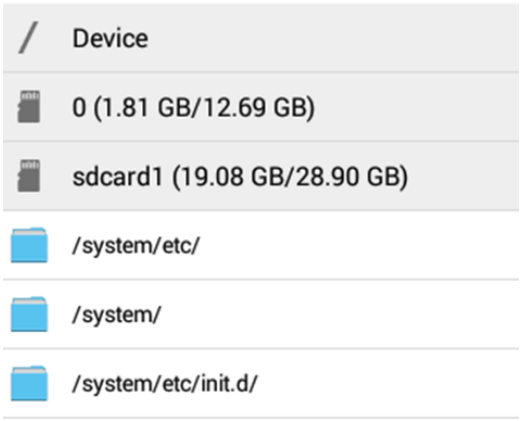
Gambar 2. Susunan pada NANDFLASH

Pada Gambar 2 tersebut menggambarkan suatu blok NANDFLASH, Bootloader ditempatkan dengan cara 'burned' pada posisi yang paling awal, disusul dengan parameter Bootloader, kernel dan FileSystem. Sehingga pada proses POWER ON atau RESET, maka proses sistem akan berawal dari Bootloader ini. Kernel kemudian mengambil alih proses selanjutnya dengan memuat (mounting) RootFileSystem. Kernel pada suatu sistem disesuaikan dengan perangkat-perangkat keras yang terpasang pada sistem itu sendiri, sehingga kernel dapat dimodifikasi sesuai kebutuhan (elastis). Suatu' Over Clocking' yang dikenal untuk modifikasi kecepatan CPU dari kecepatan bakunya dengan perubahan pada kernel. Suatu smartphone atau table PC dengan Sistem Operasi Android yang berada dilapisan atas kernel Linux sangat memenuhi syarat sebagai suatu Linux embedded system. Sistem Operasi Android menjalankan aplikasi –aplikasi Android dengan ekstensi .apk yang biasanya ditulis dengan bahasa Java dan dikompilasi ke bytecode. File –file dengan ekstensi .class pada Byte code ini biasanya berjalan diatas Java Virtual Machine. Oleh Google, dengan alasan hak cipta tadi, dikonversikan ke ekstensi .dex (Dalvik Executable) yang kompatible dengan Dalvik. Perkembangan terakhir, Google masih kuatir dengan .dex ini dan telah mulai mempergunakan platform DART yang telah dikembangkan secara rahasia beberapa tahun belakangan ini.

C. Folder Init

Posisi folder init.d berada dalam /system/etc yang dapat dimanfaatkan oleh pengguna untuk menyimpan file –file yang berupa script. Script –script ini akan dieksekusi sewaktu embedded system sewaktu boot. Apabila script –script tersebut merupakan perintah – perintah yang mengadakan perubahan –perubahan pada system, maka seoleh-olah telah terjadi perubahan pada kernel. Karena terjadi sebelum level aplikasi berjalan, sehingga dapat dikatakan juga script-script pada init.d menjadikan kernel stok (stock kernel) menjadi suatu kernel khusus (custom kernel) atau malahan menjadikan suatu kernel khusus menjadi lebih khusus.

Ada atau tidaknya folder init.d ini, tergantung pada ROM (Sistem Operasi yang di-instal dalam suatu Nandflash, sehingga dinamakan Read Only Memory). Dipakai istilah ROM mendapatkan support init.d. Apabila dengan suatu program file manager langsung terlihat folder init.d pada /system/etc maka secara otomatis dikatakan ROM mendapatkan support init.d. Akan tetapi apabila tidak terlihat folder tersebut, maka tetap ada kemungkinan ROM tersebut mendapat init.d support hanya saja folder tersebut belum tercipta dan belum di-enabled. Terdapat beberapa cara untuk mengetahui ada atau tidaknya init.d support terhadap suatu ROM. Gambar 3 berikut ini menunjukkan folder init.d



Gambar 3. Keberadaan folder init.d

Beberapa jenis perintah yang dituliskan dalam script – script init.d antara lain:

a. Pemuatan (loading) dari suatu driver yang diperlukan oleh suatu aplikasi, driver tersebut dapat saja yang berasal dari library linux yang tidak dimuatkan, karena sifat fleksibiltas dan dapat pula driver yang khusus

dibuat oleh pembuat aplikasi. Apabila tidak ditumpangkan ke folder init.d, untuk memuatkan driver dapat dilakukan dengan melalui suatu terminal emulator seperti pada Gambar 4

Gambar 4. Ikon Terminal Emulator

Contoh perintah-perintah di diesksekusi pada terminal emulator untuk memuatkan suatu driver kendali.koyang tersimpan pada folder /system/lib/modules supaya aplikasi yang memerlukan driver tersebut dapat memanfaatkannya yaitu :

```
$ su
# /system/xbin/insmod /system/lib/modules/kendali.ko
```

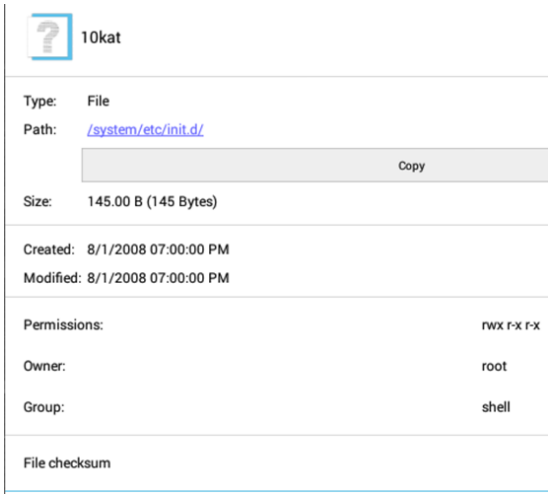
Perintah su untuk mendapatkan hak sebagai superuser, hak sebagai administrator sebagai padanan pada Sistem Operasi Windows. Sebagai pemegang hak superuser, hak yang seharusnya hanya dimiliki oleh system, maka pengguna dapat dapat mengubah isi folder –folder yang berada pada lapisan root. Hak sebagai superuser dapat diperoleh oleh pengguna dengan melalui proses yang disebut rooting. Peralatan berbasis pada sistem Operasi Android termasuk yang mudah dilakukan proses rooting ini, karena sifat Android yang ‘terbuka’ sehingga bootloader tidaklah dikunci (locked) terlalu ketat. Biasanya peralatan berbasis Android sudah berada pada posisi terbuka (unlocked) sewaktu mencapai tangan pengguna akhir, kalaupun berada pada posisi terkunci (locked) maka proses unlocking mendapat bantuan dari pihak pembuatnya. Sebagai perbandingan, peralatan-peralatan berbasis pada Sistem Operasi lainnya seperti Microsoft Windows dan IOS dari Apple akan sangat sulit untuk proses unclocking bootloader, karena memang dipersulit oleh pembuatnya. Hal yang berlawanan ini merupakan manifestasi dari perbedaan kutub yang terbuka terhadap kutub yang tertutup. Gambar 5 berikut ini menunjukkan posisi root setelah perintah su

Gambar 5. Posisi root setelah su

Perintah insmod yang terdapat dalam folder /system/xbin berfungsi untuk menyisipkan module (insert module) kendali.ko yang tersimpan pada folder /system/lib/modules, ekstensi .ko menyatakan file tersebut merupakan suatu driver. Apabila tidak terdapat error sewaktu di run pada terminal emulator, maka dapat dibuatkan suatu script dengan nama 33kendali misalnya. Dengan isi script:

```
#!/system/bin/sh
/system/xbin/insmod /system/lib/modules/kendali.ko
```

Sebagai pengganti su, pada script diberikan hak membaca (r), menulis (w) dan mengeksekusi (x) atau rwx. Karena r juga dsetarakan 4, w = 3 dan x =1 maka ‘previlege’ rwx =7, apabila hanya mendapatkan privilege 4 berarti hanya dapat membaca tanpa hak menulis dan mengeksekusi. Pada Gambar 6 menunjukkan priveledge script.



Gambar 6. Priviledge 755 script

b. Perintah –perintah merupakan suatu proses, sebagai contoh prosessymlink dari suatu folder ke folder lainnya. Biasanya folder asal berada pada memori internal yang kapasitasnya lebih kecil daripada memori eksternal yang merupakan memori tujuan.
Script ini apabila dijalankan pada suatu terminal emulator:
su
mkdir /storage/extsd/cabang
cp /storage/internal/data1 /storage/extsd/cabang/data1
rm /storage/internal/data1
ln –s /storage/extsd/cabang/data1 /storage/internal/data1

dimana pada script diatas, perintah mkdir menciptakan suatu direktori baru bernama cabang yang merupakan anak cabang directori /storage/extsd, extsd adalah eksteranal sdcard. Perintah copy (cp) menggandakan data1 dari /storage/internal ke cabang extsd tadi dengan nama yang sama, data 1. Rm adalah perintah mengosongkan isi data1dari internal memori. Perintah ln-s symlink menjadikan data1 yang terletak pada internal sebagai ‘shortcut’ ke yang terletak pada eksternal sdcard. Apabila perintah-perintah diatas disusun sebagai suatu script dan di eksekusi pada boot, maka symlinking telah berjalan sewaktu Sistem Operasi telah selesai booting.

III. METODE PENELITIAN

Penelitian ini dilakukan di Ruang kelas dan Laboratorium Kontrol Digital Jurusan Teknik Elektro Fakultas Teknik, Universitas Sriwijaya dengan langkah yang tersusun sebagai berikut:

A. Studi Literatur dan Referensi

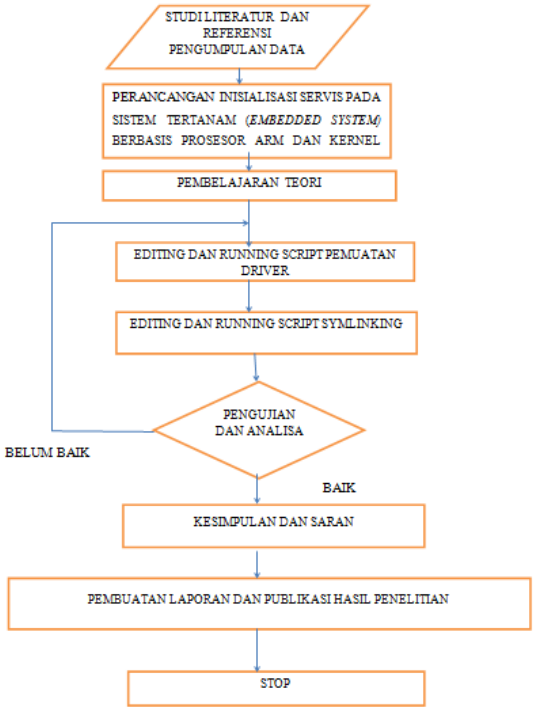
Pengumpulan bahan-bahan berupa referensi tentang teori dan manual/datasheet dari perangkat keras yang akan dipergunakan dari sumber-sumber seperti internet, dan pemesanan buku yang akan dipergunakan sebagai penunjang penelitian ini.

B. Perancangan Penelitian

Pada langkah ini akan dilakukan setelah mengumpulkan referensi mengenai kernel Linux dan init.d.

C. Persiapan sarana penelitian

Persiapan sarana penelitian mengenai kernel Linux dan init.d yang meliputi latihan pemograman dalam bahasa Linux, instalasi aplikasi untuk membuka folder-folder pada peralatan untuk melihat system file, persiapan rooting supaya dapat masuk ke system file root, memilih percobaan- percobaan yang akan dilakukan. Adapun metode penelitian ini digambarkan pada Gambar 7 berikut ini.



Gambar 7. Metode Penelitian

IV. PENGUJIAN DAN ANALISA

Pada penelitian ini dilakukan pengujian yang meliputi sebagai berikut:.

a. Pemuatan driver melalui peng-eksekusian script pada init.d, dalam hal ini dipilih driver untuk voodoo, yang disalinkan ke folder /system/lib/modules. Aplikasi yang akan dijalankan dengan driver tersebut dapat berjalan langsung setelah proses boot selesai).

b. Menambahkan script yang menjalankan proses symlink pada folder init.d, Sehingga setelah proses boot selesai, folder data pada internal memori telah ter-symlink ke folder korespondensinya pada eksternal memori.

Pada Tabel 1 berikut ini menunjukkan perbandingan antara hasil yang dicapai :

TABLE I. TABEL PERBANDINGAN SCRIPT DAN HASIL YANG DICAPAI

	Tanpa script pada init.d	Dengan script pada init.d
Aplikasi voodoo yang memerlukan driver voodoo.ko	Harus memuat driver dengan bantuan Terminal Emulator	Langsung setelah boot
Symlink folder dari memori internal ke memori eksternal	Harus melalui Terminal Emulator	Langsung setelah boot secara otomatis

Berdasarkan Tabel 1 tersebut Script yang ditulis khusus untuk suatu keperluan tertentu dan diletakkan pada folder init.d, telah dapat menggantikan proses perubahan pada kernel itu sendiri secara baik.

V. KESIMPULAN

Perubahan langsung ke kernel (tweaking) mungkin dapat digantikan dengan cara menempatkan script- script yang ditulis sesuai kebutuhan ke folder init.d. ‘Tweaking’ ini dapat diperluas dengan mengubah parameter parameter pada CPU dan memori sehingga dapat dicapai suatu optimalisasi kinerja peralatan, misalnya ‘overclocking’ (mempercepat clock diatas standard) hanya dengan mengubah isi script-script init.d tersebut. Suatu proses yang jauh lebih mudah dibandingkan dengan membongkar dan menyusun kembali kernel tersebut. Sebagai saran, pemenuhan sarana pembelajaran tersebut dapat memberikan dampak yang sangat positip.

REFERENSI

[1] Evi Nemeth, Garth Snyder, Unix and Linux System Administration Handbook 4th edition, 2011, Perentice Hall

[2] Mark G. Sobell, A Practical Guide to Linux Commands, Editor, and Shell Programming, Perentice Hall PTR.

[3] Karim Yaghmour, Jon Masters, Building Embedded Linux Systems 2ed, O’Reilly, http://news.cnet.com/8301-13505_3-9823038-16.html, Torvalds calls flexibility the 'biggest strength' of Linux