PERBANDINGAN KOMPILATOR SILANG UNTUK PAKET TARBALL GUNA DITANAM PADA PDA DENGAN SISTEM OPERASI LINUX FAMILIAR DI QEMU

Abdi Wahab[1], Abdusy Syarif[2]
[1,2] *Fakultas Ilmu Komputer, Program Studi Teknik Informatika - Universitas Mercu Buana*
E-mail: nangdul56@gmail.com, abdusyarif@mercubuana.ac.id

ABSTRAKS

Pada penelitian ini, kami melakukan perbandingan antara beberaoa aplikasi (tools) kompilator silang, seperti monmotha, arm-linux-gcc, dan gcc cross-compiler. Adapun paket yang diujicoba untuk dikompilasi silang adalah paket berekstensi tar.gz dan ditanam pada Personal Digital Assistance (PDA) atau Pocket-PC yang menggunakan siistem operasi Linux Familiar. Dari ketiga kompilator silang tersebut, hanya gcc cross-compiler yang melakukan kompilasi silang, baik kompilasi di komputer asal (host) atau di komputer target (tujuan). Sedangkan untuk monmotha dan arm-linux-gcc hanya mampu mengkompilasi di komputer host saja, sedangkan eksekusi di komputer target gagal. Hal ini disebabkan karena arsitektur mesin yang dipakai Linux Familiar yang diinstal di emulator berbeda dengan arsitektur mesin tujuan kompilator tersebut.

Kata Kunci: Kompilasi silang, paket tarball

1. PENDAHULUAN

Sistem operasi kecil, seperti Linux Familiar yang diinstall di Pocket PC memiliki keterbatasan media penyimpanan. Ini menyebabkan pengurangan paket yang ada didalamnya. Salah satu paket yang dikurangi adalah gcc, yang merupakan kompilator dasar di linux. Hal ini menyebabkan penambahan paket baru menjadi susah, karena kebanyakan paket yang baru disediakan dalam bentuk .tar.gz (tarball), dan harus dikompilasi dengan gcc.

Salah satu cara untuk penambahan paket baru berekstensi .tar.gz ke Linux Familiar adalah dengan cara kompilasi silang. Kompilasi silang yang dilakukan pada penelitian ini hanya sebatas untuk merubah paket .tar.gz menjadi .so, .o dan file biner jika ada. Juga mencoba untuk mengkompilasi silang source code dalam bahasa C/C++.

Kompilasi silang dilakukan dengan kompilator silang, seperti monmotha, arm-linux-gcc, dan gcc cross compiler. Dari ketiga kompilator silang tersebut, hanya gcc cross compiler yang mampu melakukan kompilasi silang, baik kompilasi di komputer host ataupun saat eksekusi di komputer target. Sedangkan untuk monmotha dan arm-linux-gcc hanya mampu mengkompilasi di komputer host saja, sedangkan eksekusi di komputer target gagal. Hal ini disebabkan karena arsitektur mesin yang dipakai Linux Familiar yang diinstal di emulator berbeda dengan arsitektur mesin tujuan kompilator tersebut.

Kompilator silang ini diharapkan mampu memutakhirkan Linux Familiar. Dan dari hasil penelitian, akan lebih baik jika dapat dikembangkan kompilator silang dengan banyak arsitektur mesin.

Pocket PC memiliki keterbatasan dalam media penyimpanan, sehingga Linux Familiar yang diinstal di Linux Familiar mengalami pengurangan paket. Salah satu paket yang dikurangi di dalamnya adalah gcc. Gcc digunakan untuk mengkompilasi paket baru berekstensi .tar.gz. Sehingga penambahan paket berekstensi .tar.gz ke linux Familiar tidak dapat dilakukan. Salah satu cara yang dapat dilakukan untuk menambahkan paket .tar.gz tersebut adalah dengan cara kompilasi silang paket tersebut.

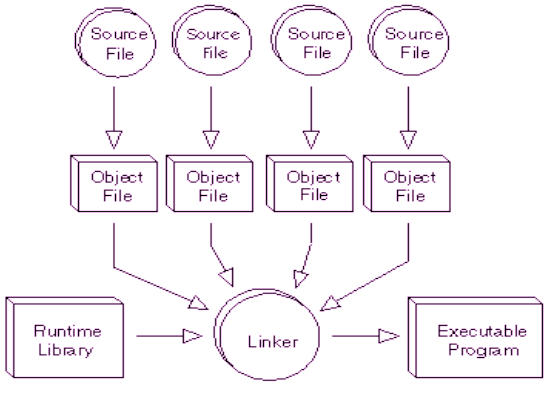
Kompilasi silang memungkin paket .tar.gz yang diinginkan berjalan pada Linux familiar. Dengan menggunakan kompilator silang yang sesuai dengan arsitektur dari komputer target.

2. KOMPILASI SILANG

2.1 Konsep dasar kompilasi

Kompilasi adalah kemampuan kompilator untuk merubah *source code* dari bahasa pemrograman tingkat tinggi menjadi bahasa pemrograman tingkat rendah. Sehingga tujuan dari kompilasi sendiri adalah untuk mendapatkan program yang dapat dieksekusi.

Fungsi dari kompilasi adalah melakukan translasi dari bahasa pemrograman tingkat tinggi (*source code*) ke bentuk yang sama dengan bahasa program mesin, dan memberikan pesan sebagai diagnostik bila terdapat kesalahan pada sumber program di bahasa tingkat tinggi yang dibuat oleh programmer.



Gambar 1. Struktur Kompilasi

Kinerja kompilator saat melakukan kompilasi pertama-tama adalah membuat *Object File* yang kemudian oleh *Linker* dihubungkan dengan *Runtime Library* sehingga menjadi *Executable file***.**

2.2 Kompilasi silang

Kompilasi silang adalah kemampuan kompilator untuk menciptkan kode yang dapat dieksekusi untuk platform lain selain untuk platform dimana kompilator dijalankan. Kebanyakan digunakan untuk melakukan kompilasi pada sistem tertanam atau platform beragam. Terdapat beberapa faktor kompilasi silang digunakan pada kebanyakan sistem tertanam, diantaranya adalah:

- Kecepatan – Platform target biasanya lebih lambat dibandingkan paltform *host*. Kebanyakan perangkat keras tertempel khusus dibuat dengan *cost* yang rendah dan juga konsumsi power yang rendah pula, bukan untuk performa yang tinggi.
- Kemampuan. Mengkompilasi sangat intensif dalam sumber daya. Platform target tidak mempunyai bergiga-giga memori pada ruang disk sebagaimana yang dimiliki oleh PC biasa.
- Ketersediaan. Untuk memutakhirkan Linux dengan fitur-fitur baru sangatlah susah, bahkan untuk platform yang sudah lama stabil seperti ARM dan MIPS. Salah satu cara adalah kembali lagi ke kompilasi silang.

Pada kompilasi silang biasanya terdapat dua istilah, yaitu paltform *host* dan platform target. Di paltform *host* terjadi kompilasi, sedangkan di platform target hasil kompilasi yang berupa file yang dapat dieksekusi/dijalankan.

Gambar 2 Kompilasi Silang

2.2.1 Beberapa alat kompilator silang

Banyak alat (*tool*) untuk melakukan kompilasi silang untuk Linux Familiar yang biasanya berjalan pada mesin arm. Diantaranya adalah:

- Monmotha. Merupakan kompilator untuk mesin arm yang banyak digunakan oleh Pocket PC.
- Arm-linux-gcc. Tidak jauh berbeda dengan monmotha, hanya saja memiliki versi dari setiap keluarannya.

Gcc *cross compiler*. Tool kompilasi silang yang disediakan oleh gcc. Dan memiliki beberapa keterkaitan library untuk membangun kompilasi silang, diantaranya adalah binutils, libc6, libgcc1, dan beberapa paket terkait.

2.3 Qemu

Qemu adalah sebuah emulator, lebih tepatnya emulator processor. Qemu menggunakan translasi dinamik sehingga membuat Qemu dapat berjalan dengan cepat. Dua mode yang dimiliki Qemu adalah emulasi seluruh sistem, dan emulasi mode *user*.

Beberapa target perangkat keras (processor) yang didukung oleh Qemu adalah X86, PowerPC, Sparc, ARM, dan MIPS.

Qemu juga dapat berjalan di banyak platform. Seperti di Linux, Windows (Qemu-win), Mac OS (Q), bahkan disediakan juga untuk OpenSolaris.

2.3.1 Poky-qemu

Poky-qemu adalah salah satu program yang disediakan oleh poky-script. Poky-script sendiri adalah paket yang berisikan kumpulan-kumpulan skrip yang berguna dalam membantu menjalankan image yang disediakan oleh Poky. Poky sendiri adalah sebuah tool yang dikembangkan untuk untuk membantu dalam pengembangan sistem tertanam, agar lebih mudah untuk diaplikasikan dandikembangkan.

Skrip poky-qemu ini digunakan untuk menjalankan image Familiar Linux dengan image berekstensi ext2 yang disediakan oleh Poky. Dan juga dengan kernel image yang harus disertakan untuk dijalankan bersamaan dengan image Familiar Linux.

3. ANALISA DAN PENGUJIAN

3.1 Analisa masalah

Dalam melakukan kompilasi silang ke dalam Linux Familiar yang kebanyakan berjalan di mesin arm diperlukan kompilator silang yang sesuai dengan mesin target. Penelitian ini menggunakan 3 buah kompilator silang yang memiliki target untuk mesin arm, yaitu monmotha, arm-linux-gcc, dan gcc cross compiler.

Setiap dari kompilator tersebut memiliki kelebihan dan kekurangan masing-masing. Sebagai contoh kelebihan dari monmotha dan arm-inux-gcc, keduanya memiliki kernel header sendiri, sehingga pengguna tidak perlu mencari kernel header yang ingin digunakan. Untuk kekurangannya, keduanya memiliki cara instalasi yang lumayan sulit dan membingungkan.

Kernel header di dalam kompilasi silang sangat diperlukan. Karena ketergantungan yang tinggi dalam linux membuat sebuah *source code* yang ingin dikompilasi memerlukan beberapa *file header* yang biasanya terdapat di dalam kernel *header*.

Satu hal yang perlu diperhatikan dalam melakukan kompilasi silang adalah mengarahkan kompilator menggunakan kompilator silang. Untuk paket tar.gz biasanya mengarahkan skrip untuk kompilasi di dalam *file* Makefile ke kompilator silang. Skrip yang

 ISSN: 1907-5022

biasanyadi arahkan ke kompilator silang adalah skrip CC, LD, dan CPP. Contohnya: CC = gcc diarahkan menjadi CC = arm-linux-gcc

3.2 Implementasi kompilasi silang

Untuk dapat melakukan kompilasi silang, Terlebih dahulu perlu melakukan instalasi terhadap kompilator silang di komputer *host*.

Adapun langkah-langkah untuk melakukan instalasi kompilator silang di komputer *host* adalah sebagai berikut:

1. Unduh kompilator silang.
2. *Unpack* kompilator silang.
3. *Retrieve* kernel *source code* yang sesuai dengan kernel yang digunakan pada linux Familiar saat ini.
4. Buat link file dari direktori "asm" dan "linux" pada kompilator silang diarahkan ke arm kernel *header source*.
5. Konfigurasi PATH untuk kompilator silang dan direktori /usr/src/linux mengacu ke arm kernel *header source*.

Gambar 3. Linux Familiar di Qemu

Untuk gcc *cross compiler* menggunakan repository debian dapat dengan dengan langkah berikut:

1. Menambahkan daftar repository ke sources.list yang terdapat di dalam direktori /etc/apt.
2. Perbarui *repository*.
3. Setelah *repository* diperbarui, install gcc *cross compiler* untuk mesin arm beserta paket-paket yang mengikutinya (*dependencies)*.
4. *Retrieve* kernel *header source* sesuai dengan versi kernel *header* Linux Familiar yang dipakai saat ini.
5. Buat link file dari direktori "asm" dan "linux" pada direktori include di kompilator silang diarahkan ke arm kernel *header*.

Selanjutnya untuk menjalankan linux Familiar di Qemu bisa menggunakan image linux Familiar yang disediakan oleh poky. Untuk mendapatkan image linux Familiar dan juga kernel imagenya dapat dilihat di http://pokylinux.org/release/pinky-3.1/ .

Linux Familiar yang dijalankan di Qemu memerlukan sebuah skrip poky-qemu. Sebagaimana dijelaskan pada point 2.3.1. Bila poky qemu sudah terinstal hanya perlu menggunakan perintah berikut
$ poky-qemu kernel-image image-familiar-linux.

4. PENGUJIAN DAN ANALISA HASIL

4.1 Kompilasi Silang untuk .c dan .cpp

File .c dan .cpp yang berdiri sendiri, tidak terkait dengan *file* lain, jika akan dikompilasi silang sama dengan cara untuk mengkompilasi *file* .c dan .cpp menggunakan kompilator biasa, umumnya sintaks kompilasi yang dipakai di gcc, hanya saja kompilator yang digunakan diganti dengan kompilator silang yang telah diinstal.

Secara umum sintaks untuk melakukan kompilasi silang untuk sebuah *file* .c dan .cpp adalah sebagai berikut:
$ arm-linux-gcc -o nama-file-keluaran source-file.c

4.2 Kompilasi Silang untuk Paket Linux (.tar.gz)

Sebuah paket tarball (.tar.gz) biasanya terdiri dari banyak *file* .c dan .cpp. Semuanya saling terkait antara satu dengan yang lainnya. Untuk memudahkan melakukan kompilasi diperlukan sebuah makefile.

Makefile juga sangat diperlukan untuk kompilasi silang, karena dalam 3 tahap instalasi paket di linux, yaitu *configure*, *make*, dan *make install*, di tahap make-lah terjadi kompilasi. Sehingga diperlukan sedikit perubahan pada Makefile. Diantara beberapa skrip makefile yang harus diarahkan ke kompilator silang adalah CC, LD, dan CPP. Contohnya:

CC = gcc dirubah menjadi CC = arm-linux-gcc

Tapi ada beberapa paket .tar.gz yang makefilenya tidak dapat dirubah secara manual, melainkan dirubah melalui tahap sebelumnya yaitu *configure*. Skrip yang umumnya dipakai untuk mengarahkan kompilasi dilakukan menggunakan kompilator silang adalah:

```
$ ./configure --host=arm-linux-gcc
```

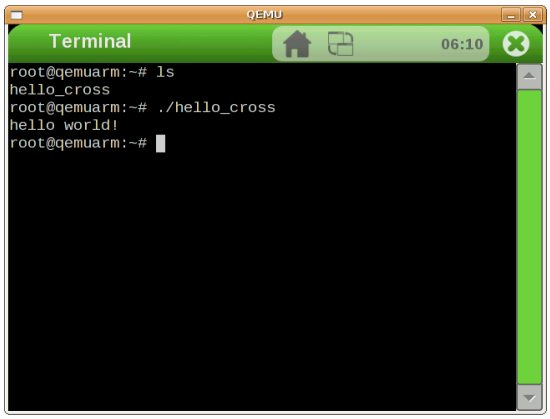
Akan lebih baik jika membaca README dan Install *file* terlebih dahulu.

 ISSN: 1907-5022

4.3 Skenario Pengujian

Skenario untuk *file* dan paket yang dikompilasi silang sebagai berikut:

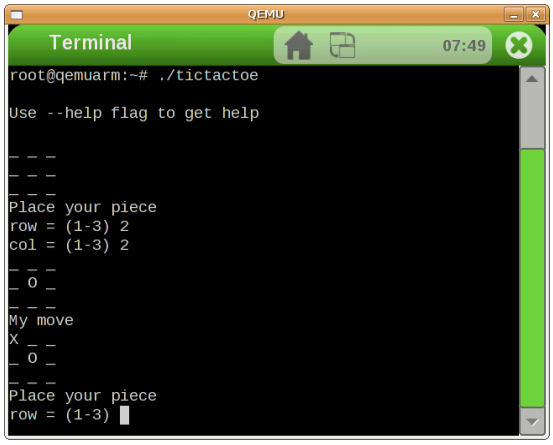
- Kompilasi silang untuk file .c dan .cpp dengan gcc cross compiler
- Untuk kompilasi silang *file* .c/.cpp dengan menggunakan gcc *cross compiler* dapat dilakukan pada komputer *host*. Dan juga dapat dieksekusi pada komputer target.



Gambar 4 File .c/.cpp Hasil Kompilasi Silang

- Kompilasi silang untuk paket .tar.gz dengan gcc cross compiler

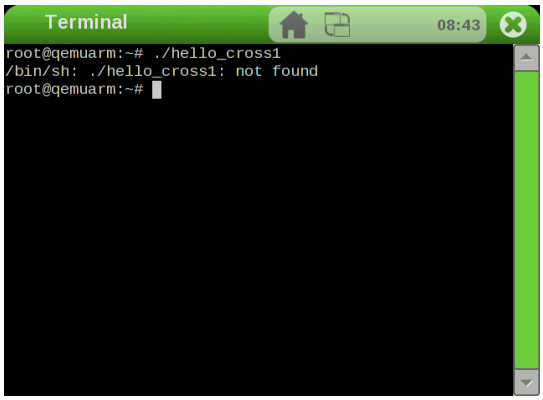
Paket tar.gz (tictactoe, sudoku, dan libpng) yang dikompilasi silang dengan menggunakan gcc cross compiler berhasil dilakukan pada komputer *host*. Dan juga berhasil dieksekusi pada komputer target.



Gambar 5 Paket tar.gz Hasil Kompilasi Silang

- Kompilasi silang untuk file .c/.cpp dengan monmotha dan arm-linux-gcc

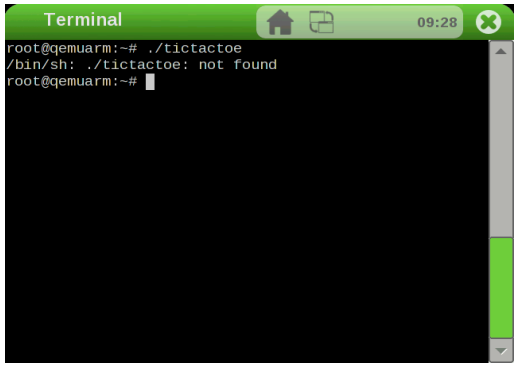
Monmotha dan arm-linux-gcc berhasil melakukan kompilasi silang untuk *file* C/C++ pada komputer host. Akan tetapi *file* biner yang dihasilkan tidak dapat dieksekusi di komputer target.



Gambar 6 Hasil Kompilasi Silang dari C/C++ yang Gagal

- Kompilasi silang untuk paket tar.gz dengan monmotha dan arm-linux-gcc

Untuk paket tar.gz yang dikompilasi silang, monmotha dan arm-linux-gcc hanya mampu melakukan sebatas merubah paket menjadi file biner saja. Tetapi file biner tidak dapat dieksekusi di komputer target.



Gambar 7 Hasil Kompilasi Silang Paket tar.gz yang Gagal

4.4 Analisa Perbandingan Hasil Pengujian

Berdasarkan hasil pengujian diatas, diperoleh beberapa hasil, diantaranya:

1. Kompilator silang yang berhasil untuk melakukan kompilasi silang (untuk paket tar.gz, *file* .c dan .cpp) secara sempurna adalah gcc *cross compiler*.
2. Monmotha dan arm-linux-gcc hanya bisa untuk merubah *file* source menjadi *file* biner saja pada komputer *host*, tapi *file* biner (executable) tersebut tidak dapat dijalankan di komputer target. Karena kedua kompilator silang ini tidak mendukung port baru yaitu EABI (*Embedded Application Binary Interface*).
3. Jika kompilasi silang berhasil maka *file* biner akan dapat dijalankan di komputer target dengan baik.

4. Kompatibilitas gcc juga mempengruhi. Dalam hal ini gcc yang digunakan versi 4.2 yang merupakan *default* dari Ubuntu 8.04. Sehingga gcc *cross compiler* yang bisa terinstal adalah gcc cross compiler versi 4.2 juga.

Kernel *header* yang dipakai adalah 2.6.19. Karena poky pinky 3.1 (Linux Familiar) yang terinstal di emulator Qemu menggunakan kernel 2.6.23. Walaupun berbeda nomor revisi minornya (yaitu 19 dan 23) tapi masih dapat digunakan karena revisi majornya masih sama (6).

5. KESIMPULAN

Dari hasil pembahasan dan implementasi yang diuraikan pada bab-bab sebelumnya, maka dapat diambil kesimpulan sebagai berikut:

[1] Perbedaan monmotha *cross compiler*, arm-linux-gcc-3.4.1 dan gcc *cross compiler* ada yang berhasil ada pula yang tidak. Hal ini dikarenakan perbedaan arsitektur mesin yang dipakai oleh komputer target. Sehingga mengharuskan kompilator silang harus sesuai dengan arsitektur mesin target.

[2] Versi gcc *cross compiler* harus sama dengan versi gcc yang terinstal pada komputer host.

[3] Untuk mengkompilasi silang sebuah *file* dengan ekstensi .c atau .cpp hampir sama dengan cara mengkompilasi biasa dengan gcc. Hanya saja kompilatornya menggunakan kompilator silang.

[4] Untuk mengkompilasi silang sebuah paket tar.gz, perlu untuk membaca *file* README dan Install terlebih dahulu, karena setiap paket berbeda dalam *file* configure dan makefilenya.

Penelitian kompilasi silang (cross compile) ini masih banyak terdapat kekurangan, sehingga memerlukan beberapa penelitian dikemudian hari, diantaranya adalah:

1. Kompilasi silang yang memiliki perbedaan diantara mesin target yang dituju, sehingga membuat pengguna harus mencari terlebih dahulu kompilator silang yang cocok untuk mesin target yang dituju.
2. Akan lebih baik jika dapat diciptakan *cross compiler* untuk banyak mesin, sehingga tidak perlu susah untuk instalasi banyak kompilator silang pada komputer host jika ingin melakukan kmpilasi silang untuk banyak target mesin.
3. Untuk penambahan paket baru yang berekstensi tar.gz, dapat menggunakan cara yang hampir sama dengan penelitian ini. Dan alangkah lebuh baik jika membaca *file* README dan Install terlebih dahulu**.**

PUSTAKA

Bovet, D.P., & Cesati, Marco. *Understanding the Linux Kernel* CA: O'Reilly, 2000.

Bellard, Fabrice. *Qemu Emulatro User Documentation*. [online] diakses tanggal 9 Januari 2009 terdapat di URL http://bellard.org/qemu/qemu-doc.html

Griffith, Arthur. *GCC: The Complete Reference* Osborne: McGraww-Hill, 2002.

Katriena, Flory. *Linux Untuk Pemula* Jakarta: PT Elex Media Komputindo, 1999.

Purdie, Richard et al. *Poky Handbook.* [online] diakses tanggal 8 Januari 2009 terdapat di URL http://www.pokylinux.org/doc/poky-handbook.html.

Sokolvsky, Paul. *Report on running Familiar Linux Under Qemu-arm*. [online] diakses tanggal 20 Januari 2009 terdapat di URL http://www.handhelds.org/pipermail/familiar/312/31225.html.

- - - - - - - -. *Compiler*. [online] diakses tanggal 3 Januari 2009 terdapat di URL http://en.wikipedia.org/wiki/Compiler.

- - - - - - - -. *Compiler*. [online] diakses tanggal 3 Januari 2009 terdapat di URL http://www.webopedia.com/TERM/C/compiler.html.

- - - - - - -. *Introduction to cross-compiling for Linux*. [online] diakses tanggal 5 Januari 2009 terdapat di URL http://landlay.net/writing/docs/cross-compiling.html.

- - - - - - -. *Tar (fileformat)*. [online] diakses tanggal 4 Januari 2009 terdapat di URL http://en.wikipedia.org/wiki/Tarball.

- - - - - - - -. *GNU Compiler Collection*. [online] diakses tanggal 6 Januari 2009 terdapat di URL http://en.wikipedia.org/wiki/GNU_Compiler_Collection.

- - - - - - -. *Familiar Distribution*. [online] diakses tanggal 4 Januari 2009 terdapat di URL http://www.handhelds.org/moin/moin.cgi/FamiliarDistribution.

- - - - - - -. *Familiar Linux*. [online] diakses tanggal 4 Januari 2009 terdapat di URL http://en.wikipedia.org/wiki/Familiar_Linux.