Sya Raihan Heggi
1301184219

Panduan Instalasi dan Hal-Hal Berkenaan P4

Untuk instalasi P4 sendiri dapat dilakukan dengan beberapa metode yang dapat dilakukan, metode tersebut dijelaskan sebagai berikut ini.

- **Mendownload Official OVA File**
  Untuk instalasi ini merupakan yang paling simpel karena menggunakan vm, yang sudah disiapkan oleh P4 Community, untuk mendownloadnya dapat menggunakan tautan berikut ini p4-guide/README-install-troubleshooting.md at master · jafingerhut/p4-guide (github.com) (pada guide yang dibuat pada jafingerhut ini sudah disediakan beberapa vm yang siap digunakan), dengan menggunakan metode ini dapat disimpulkan.
    - **Pro :** VM dapat langsung digunakan dan sudah berisi dependencies dari P4 itu sendiri, proses tidak terlalu ribet dan cocok untuk pemula yang ingin mencoba.
    - **Cons :** Tampilan vm atau os yang digunakan terkadang tidak familiar dengan yang biasa digunakan.

  Selain menggunakan link yang ada di P4-Guide oleh Jafingerhut dapat digunakan pula beberapa OVA yang dapat digunakan.

    - **NGSDN OVA** : http://bit.ly/ngsdn-tutorial-ova (berisi tutorial p4 yang dikolaborasikan dengan SDN, namun untuk menggunakan ini harus memilki basic Docker karena sistem yang disiapkan semuanya sudah berupa container docker)
    - **2018 P4 Developer Day OVA :** P4 Developer Day - Open Networking Foundation

- **Membuat VM menggunakan Vagrant**
  Metode ini dapat digunakan namun sedikit rumit karena melibatkan pembuatan VM menggunakan Vagrant, untuk metodenya sendiri dapat menggunakan repository p4lang/tutorials: P4 language tutorials (github.com), pada repository tutorial tersebut sudah disiapkan script vagrant yang dapat digunakan untuk membuat sebuah VM lengkap dengan depedenciesnya, namun metode ini tidak digunakan karena menurut pengalaman pribadi penggunaan vagrant susah dan rentan error.

- **Menginstall Depedencies Secara Manual**
  Pada Metode ini adalah metode yang digunakan untuk menyiapkan *environment* P4 pada tugas akhir, metode yang digunakan adalah menginstall komponen-komponen yang diperlukan untuk melakukan simulasi P4, komponen tersebut akan berisi hal-hal berikut ini.
    - **P4C :** Adalah komponen yang berfungsi untuk melakukan kompilasi kode P4 menjadi sebuah program yang dapat dieksekusi, dalam hal ini bisa menjadi code untuk diletakan pada chip atau membuat Behavioral-model untuk simulasi jaringan menggunakan P4, lebih jelas p4lang/p4c: P4_16 reference compiler (github.com).

- **Behavioral-Model :** Dikarenakan pada tugas akhir tidak dimiliki chip atau peralatan yang bisa diletakkan program P4, maka perlu dibuat behavioral-model yang merupakan model digital dari sebuah program P4 yang nantinya dapat digunakan untuk melakukan simulasi jaringan, p4lang/behavioral-model: The reference P4 software switch (github.com), berdasarkan tautan tersebut input masuk dari behavioral model ini adalah sebuah file JSON dari hasil proses yang dilakukan oleh P4C, namun perlu diingat untuk **bmv2** sendiri memiliki kemampuan dibawah dari **OpenVSwitch** yang sudah lebih dulu hadir.
- **Protobuf :** Merupakan sebuah komponen yang membuat standarisasi dalam bentuk yang dapat dibaca oleh P4Runtime, format ini digunakan karena pada dasarnya P4 akan menggunakan basis gRPC.
- **P4Runtime/PI :** Merupakan sebuah komponen yang berfungsi untuk untuk menghubungkan entitas P4 dengan komponen p4 lainnya, dimana didalamnya nantinya akan menjalankan gRPC server untuk melakukan binding dengan service yang tersedia untuk dokumentasi dapat dilihat disini p4lang/p4runtime: Specification documents for the P4Runtime control-plane API (github.com), PDF (P4Runtime-Spec.pdf).
- **gRPC :** gRPC adalah sebuah kerangka untuk pemanggilan suatu fungsi atau media transmisi (layaknya API/RESTAPI) yang berguna untuk bisa menerapkan prinsip server dan client, yang membedakan dengan REST adalah penggunaan basis protobuf sebagai media transmisi data yang digunakan, penjelasan lebih lanjut (REST vs. gRPC: Pertempuran API (tutsplus.com)
- **PTF :** merupakan python-based dataplane test framework yang digunakan agar dapat menggunakann python untuk melakukan testing, layaknya jaringan openflow pada umumnya dengan menggunakan PTF ini memungkinkan untuk membuat script menggunakan p4 untuk melakukan simulasi secara otomatisasi.
- **Mininet :** Digunakan untuk membuat sebuah simulasi jaringan SDN, menggunakan network simulator, untuk instalasi cukup mudah ditemukan metodenya di internet.
- **CMake :** Digunakan untuk membuat otomatisasi build yang dibuat, dalam hal ini bekerja layaknya package manager, dan kenapa ini harus diinstal karena kebanyakan dari repository P4 akan membutuhkan ini untuk melakukan compile.

Selain itu yang perlu disiapkan dalam menginstall P4 sendiri adalah hardware yang digunakan untuk melakukan simulasi berdasarkan p4-guide/README-install-troubleshooting.md at master · jafingerhut/p4-guide (github.com), didapatkan spesifikasi seperti berikut ini.

a. an unmodified fresh installation of Ubuntu Linux 18.04, 20.04, or 22.04, with ...
    i. at least 2 GB of RAM (**4 GB recommended**)

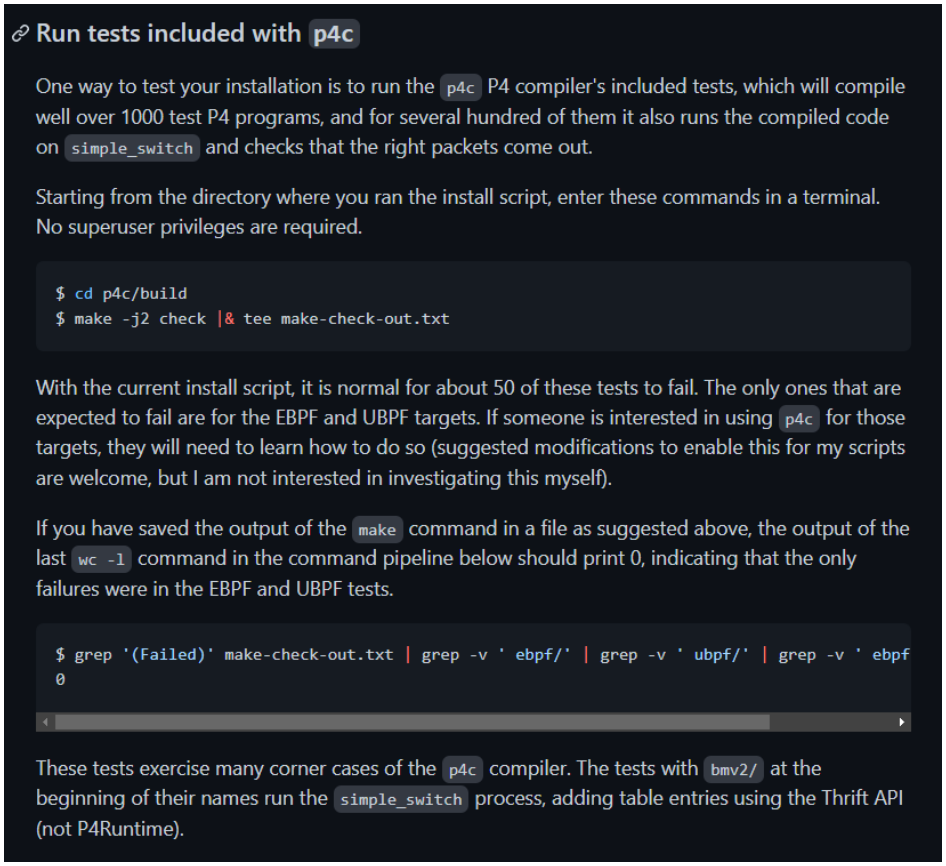
ii. **at least 13 GB** of free disk space (not 13 GB of disk space total for the VM, but 13 GB free disk space after the OS has been installed), and
iii. **a reliable Internet connection** that is up for the entire duration of running the install script -- it will download approximately 1 to

Untuk OS yang digunakan sendiri disarankan untuk menggunakan 18.04, mengapa karena pada percobaan di 20.04 kebanyakan dependencies ada yang sudah tidak dapat diinstal, namun melihat dari repository github tersebut bahwa saat ini untuk versi 20.04 dapat digunakan untuk menginstall komponen yang diperlukan.

Untuk mempermudah pengerjaan ada baiknya menyiapkan VMnya terlebih dahulu dengan spesifikasi yang sudah disebutkan sebelumnya, selain itu pastikan untuk menggunakan Python3 sebagai default dari Sistem Operasi, kenapa karena saat percobaan kemarin sempat ditemukan error ketika ada Python2 dan Python3 didalam satu sistem yang sama p4.config Module Not Found · Issue #462 · p4lang/tutorials (github.com), **Perlu diingat percobaan ini dilakukan menggunakan Virtualbox atau VMWare, dan berdasarkan laporan WSL tidak dapat menjalankan script yang dapat mempermudah instalasi**.

Untuk mempermudah instalasi maka kita dapat menggunakan script yang sudah disiapkan oleh repository jafingerhut, metode yang digunakan dapat melakukan running shell script yang sudah disediakan **Perlu Diingat Jika ada kesalahan pada instalasi maka perlu melakukan fresh installation Kembali, karena dari pemilik repository tidak memberikan fitur untuk melakukan update atau replace terhadap komponen yang diinstal sehingga rentan terjadi kesalahan**, lakukan instalasi dan kemudian jalankan pengujian yang diberikan.

Sya Raihan Heggi
1301184219



Run tests included with p4c

One way to test your installation is to run the `p4c` P4 compiler's included tests, which will compile well over 1000 test P4 programs, and for several hundred of them it also runs the compiled code on `simple_switch` and checks that the right packets come out.

Starting from the directory where you ran the install script, enter these commands in a terminal. No superuser privileges are required.

```
$ cd p4c/build
$ make -j2 check |& tee make-check-out.txt
```

With the current install script, it is normal for about 50 of these tests to fail. The only ones that are expected to fail are for the EBPF and UBPF targets. If someone is interested in using `p4c` for those targets, they will need to learn how to do so (suggested modifications to enable this for my scripts are welcome, but I am not interested in investigating this myself).

If you have saved the output of the `make` command in a file as suggested above, the output of the last `wc -l` command in the command pipeline below should print 0, indicating that the only failures were in the EBPF and UBPF tests.

```
$ grep '(Failed)' make-check-out.txt | grep -v ' ebpf/' | grep -v ' ubpf/' | grep -v ' ebpf
0
```

These tests exercise many corner cases of the `p4c` compiler. The tests with `bmv2/` at the beginning of their names run the `simple_switch` process, adding table entries using the Thrift API (not P4Runtime).

Check P4C apakah berkerja dengan benar

Tahapan Selanjutnya adalah Melakukan Testing dengan topology

Untuk mempermudah hal ini maka dapat melakukan clone dengan sistem yang sudah dibuat dan dapat langsung digunakan yaitu dapat diakses pada tautan https://github.com/p4lang/tutorials .

Pada tahapan itu dapat dilakukan pengujian dan modifikasi sesuai dengan kebutuhan, karena sistem dapat melakukan simulasi langsung dengan mininet, namun kekurangan menggunakan repository ini adalah tidak terkoneksinya dengan controller.

Bagaimana Jika ingin menghubungkan ke Controller

Saat ini untuk controller yang support dengan P4 hanya tersedia ONOS, untuk versi onos sendiri yang dapat digunakan adalah diatas 2.0, sebenarnya ada rincian namun website saat ini tidak dapat diakses 502 Bad Gateway (onosproject.org), namun untuk melakukan koneksi ini sendiri cukup rumit mengapa karena ada beberapa hal yang harus dipersiapkan, diantaranya adalah membuat pipeconf baru untuk dapat meregister komponen baru yang kita miliki, untuk lebih jelasnya ada pada thread ini https://forum.p4.org/t/connecting-onos-and-p4-dataplane-in-simulation/295/2 thread

tersebut berkenaan dengan metode yang dapt digunakan dan ditanyakan pribadi oleh saya, saran yang diberikan adalah lebih baik untuk menggunakan OVA NGSDN, dan spend waktu untuk mempelajari OVA tersebut, namun karena keterbatasan waktu maka metode ini tidak dapat digunakan.

Link yang berguna untuk belajar

Ada beberapa link yang menurut saya cukup bagus untuk melakukan pembelajaran P4

- https://github.com/nsg-ethz/p4-learning **(Sudah Lama tapi ada beberapa pembahasan mengenai P4)**
- https://github.com/p4lang/tutorials **(Repository resmi pembelajaran awal bagaimana melakukan coding menggunakan P4 Language)**
- csie.ngu.edu.tw/smallko/sdn/sdn.htm **(Repository project terkait P4 dan SDN lainnya, didalamnya ada beberapa metode dan repository alternatif yang dapat digunakan untuk menunjang pengerjaan P4)**
- opennetworkinglab/ngsdn-tutorial: Hands-on tutorial to learn the building blocks of the Next-Gen SDN architecture (github.com) **(Pembelajaran lebih lanjut mengenai SDN-P4)**
- P4~16~ Language Specification **(Dokumentasi P4)**