

## ENDOSNIPE COMMUTATOR 環境構築手順書

### - ENDOSNIPE COMMUTATOR SETUP MANUAL

Version 1.3

最終更新日：2013/09/10

## 1. 概要 - Abstract

本ドキュメントはSystemTapを用いてシステム監視を行うための環境構築手順について述べる。監視対象のシステムは以下の通りである。

- This document describes how to set up your local work environment to monitor systems using SystemTap. The monitoring target systems are as follows:
  - MySQL
  - PostgreSQL
  - Apache
  - PHP
  - Python
  - Ruby

## 2. 事前準備 – Preparation

本章では環境構築に必要なファイルを準備する方法について述べる。

### 2.1. 方針

ダウンロード作業とインストール作業は別々に行えるように、ファイル群は完全に分割してある。これは、ネットワーク環境が無い場所での作業を考慮したためである。

### 2.2. ダウンロードスクリプト

ダウンロードスクリプトはネットワークに接続可能な環境において利用する。  
ファイル構成は以下の通りである。

```
PackageDownloader
├── list # 必要なファイルのダウンロード先一覧
│   ├── apache.txt
│   ├── centos.txt
│   ├── dtrace.txt
│   ├── mysql.txt
│   ├── php.txt
│   ├── postgres.txt
│   ├── python.txt
│   ├── ruby.txt
│   └── systemtap.txt
├── patch # 必要なパッチファイル
│   └── php.patch
├── readme.txt
└── script # ファイルのダウンロードを実行するスクリプト群
    ├── apache.sh
    ├── mysql.sh
    ├── php.sh
    ├── python.sh
    ├── ruby.sh
    └── systemtap.sh
```

#### 2.2.1. ダウンロード実行方法

ダウンロードを実行するには `script` ディレクトリ内のスクリプトを実行する。スクリプトのファイル名はダウンロード対象のサービスを表している。例えば、**SystemTap** 本体と **SystemTap** をインストールするのに必要なファイル群をダウンロードするスクリプトは `systemtap.sh` である。

以下のコマンドを実行してファイルのダウンロードを行う。

```
$ cd PackageDownloader/script
$ sh ./script/systemtap.sh
```

### 2.2.2. ダウンロード実行結果

ダウンロードスクリプトを実行すると以下のファイル構成図のように `package` ディレクトリと、インストールするサービス名のディレクトリが作成され、その中に必要なパッケージがダウンロードされている。なお、2.2 で示したファイル構成と重複するディレクトリは省略している。

```
PackageDownloader
├── package
│   ├── dtrace
│   │   ├── libdwarf-0.20110612-1br.el6.x86_64.rpm
│   │   └── libdwarf-devel-0.20110612-1br.el6.x86_64.rpm
│   └── systemtap
│       ├── kernel-debuginfo-2.6.32-358.el6.x86_64.rpm
│       └── kernel-debuginfo-common-x86_64-2.6.32-358.el6.x86_64.rpm
```

## 2.3. インストールスクリプト

インストールはネットワーク環境の有無を問わず利用可能である。`PackageInstaller` は `/home/user_name/work/` に配置し、ダウンロードスクリプトを利用して用意したファイルを移動させる。

### 2.3.1. ダウンロードファイルの配置

ファイル構成が以下になるように `package` ディレクトリを移動させる。  
ダウンロードスクリプトによって作成された `package` ディレクトリとその中身をそのまま移動させれば良い。

```
PackageInstaller
├── package
│   ├── dtrace
│   │   ├── libdwarf-0.20110612-1br.el6.x86_64.rpm
│   │   └── libdwarf-devel-0.20110612-1br.el6.x86_64.rpm
│   └── systemtap
│       ├── kernel-debuginfo-2.6.32-358.el6.x86_64.rpm
│       └── kernel-debuginfo-common-x86_64-2.6.32-358.el6.x86_64.rpm
└── script
```

```
|─ mysql.sh  
|─ openssl.sh  
|─ rails.sh  
└─ systemtap.sh
```

### 2.3.2. インストール実行方法

---

インストールの実行方法については各サービスのインストール手順の章において述べる。

### 3. 構築する環境 – System architecture

構築する環境のシステム構成は Figure 3.1 の通りである。

- System architecture to build is shown in Figure 3.1

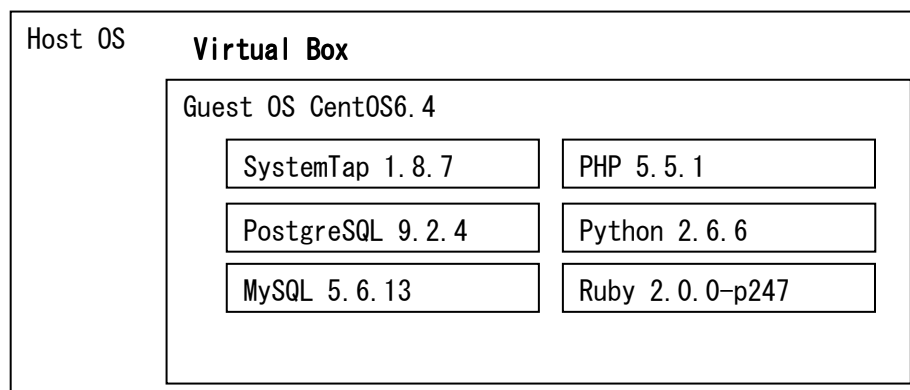


Figure 3.1 システム構成 -System architecture

## 4. OS 環境構築 – Build Operating System environment

本章では CentOS6.4 をインストールし、TeraTerm などの SSH クライアントから接続できるようにするための設定手順を説明する。

- This chapter describes how to install CentOS6.4 and configure connecting via SSH client such as TeraTerm.

### 4.1. CentOS6.4 のインストール - Installing CentOS

CentOS6.4 のインストールには以下のファイルが必要である。

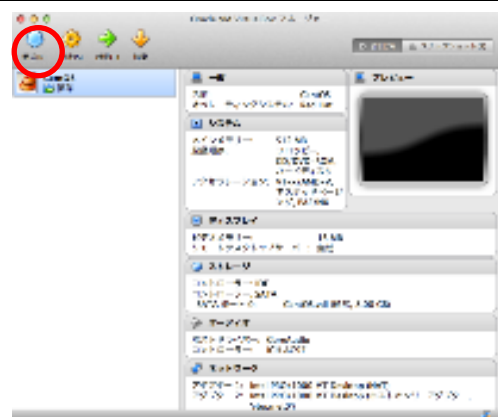
- CentOS-6.4-x86\_64-bin-DVD1.iso
- CentOS-6.4-x86\_64-bin-DVD2.iso

これらのファイルを C:\Temp\Tools\CentOS6.4 配下に配置する。

ディレクトリが存在しない場合は新規作成する。

1. 『新規』をクリックして VM を作成する


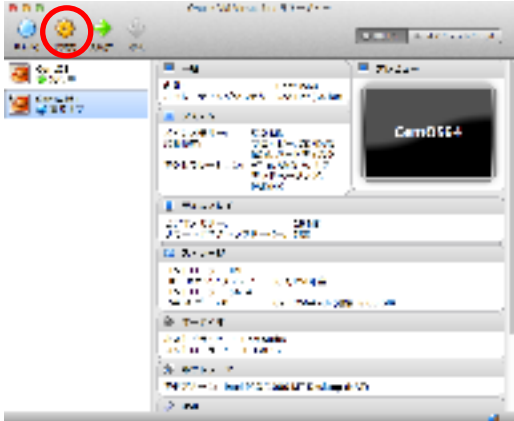
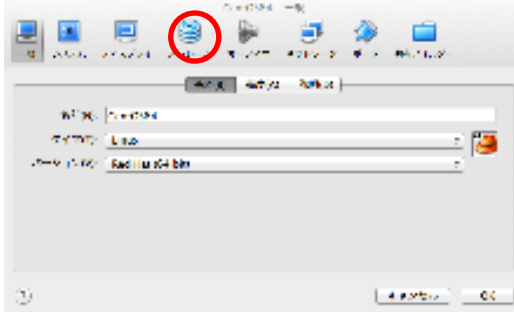
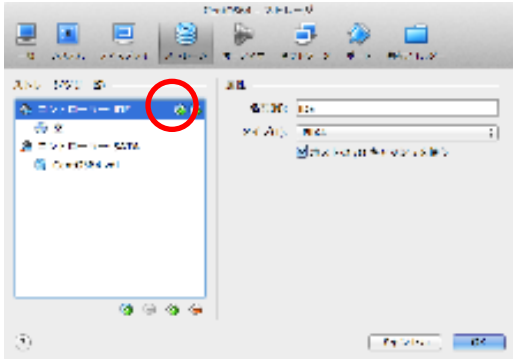
- Create new VM by clicking the “New”.

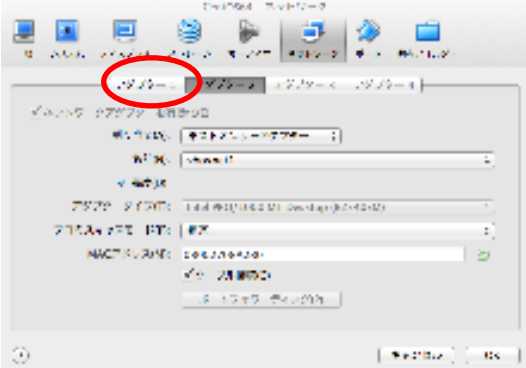

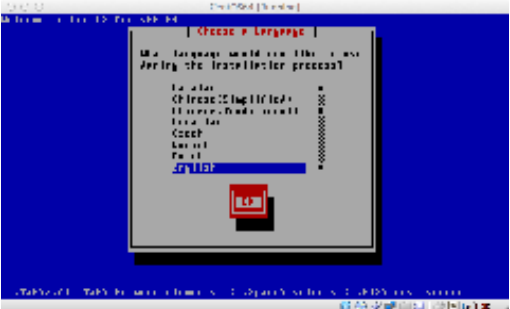


2. 名前に『CentOS64』と入力

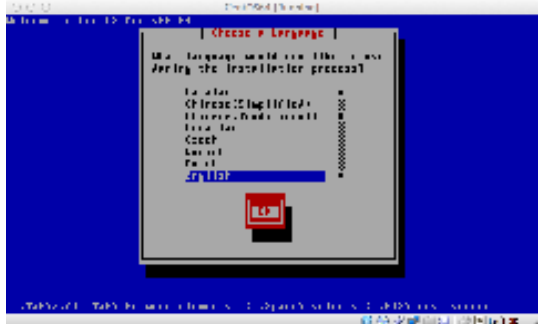
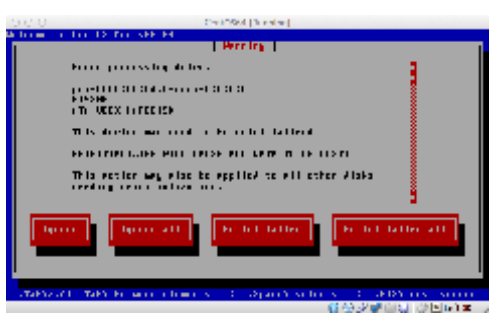
- 名前(N): CentOS64  
タイプは『Linux』を選択。
- タイプ(T): Linux  
バージョンは『Red Hat(64bit)』を選択する。
- バージョン(V): Red Hat(64bit)



<p>3. 仮想ハードドライブまでは全てデフォルト設定のままにする。ストレージの容量を 32GB に設定する。</p>	
<p>4. その後のダイアログは全てデフォルトのまま進める。(ここで設定した値は後から変更することができる。)</p> <ul style="list-style-type: none"> <li>- Keep all setting to default.</li> </ul>	
<p>5. 新しく作成した VM を選択した状態で設定をクリック</p> <ul style="list-style-type: none"> <li>- Click “Setting” button.</li> </ul>	
<p>6. ストレージタブをクリック</p> <ul style="list-style-type: none"> <li>- Click “Storage” tab.</li> </ul>	
<p>7. コントローラー:IDE 横のアイコンをクリックし、 CentOS-6.4-x86_64-bin-DVD1.iso を選択する。</p> <ul style="list-style-type: none"> <li>- Click the icon next to “コントローラー: IDE”.</li> <li>- Select CentOS-6.4-x86_64-bin-DVD1.iso.</li> </ul>	

<ul style="list-style-type: none"> <li>※インストール・ディスクを使用する場合は”空”と表示されているディスクのアイコンをクリックし、ディスクの挿入されているドライブを設定すれば、そのディスクからインストールすることが可能。</li> </ul>	
<p>8. ネットワークタブを選択</p> <ul style="list-style-type: none"> <li>- Select “ネットワーク” tab. アダプター1 を選択</li> <li>- Select “アダプター1”. 割り当て：ホストオンリーアダプターを選択</li> <li>- Attached to: Host-only Adapter. 表示されている MAC アドレスは次の手順で 使用するため控えておく。</li> <li>- Write down Mac address to use later.</li> </ul>	
<p>9. VM を起動する</p> <ul style="list-style-type: none"> <li>- Boot VM.</li> </ul>	
<p>10. Install or upgrade an existing system を選択する。 Select Install or upgrade an existing system.</p>	
<p>11. Disk Found は skip を選択する</p> <ul style="list-style-type: none"> <li>- Select skip in Disk Found.</li> </ul>	
<p>12. Choose a Language では English を選 択する</p> <ul style="list-style-type: none"> <li>- Select English in Choose a Language.</li> </ul>	



<p>13. Keyboard Type を選択する。日本語キーボードの場合は jp106</p> <ul style="list-style-type: none"> <li>- Select Keyboard Type.</li> </ul>	
<p>14. 右図のような警告が表示された場合、Re-initialize all を選択</p> <ul style="list-style-type: none"> <li>- If appear dialog like figure, select Re-initialize all.</li> </ul>	
<p>15. Time Zone Selection では作業している地域のタイムゾーンを入力する。</p> <ul style="list-style-type: none"> <li>- Select Time zone.</li> </ul>	
<p>16. Root password を入力</p> <ul style="list-style-type: none"> <li>- Input root password.</li> </ul>	
<p>17. Partitioning Type では Use entire drive を選択</p> <ul style="list-style-type: none"> <li>- Select Use entire drive in Partitioning Type.</li> <li>- Which drive(s) do you want to use for this instration?</li> </ul>	
<p>18. Writing storage configuration to disk では Write changes to disk を選択</p> <ul style="list-style-type: none"> <li>- Select “Write changes to disk” in Writing storage configuration to disk.</li> </ul>	
<p>19. インストールが開始される(完了まで10分程度時間がかかる) Start installation. (It takes about 10min.)</p>	
<p>20. Reboot したらインストールが完了する</p> <ul style="list-style-type: none"> <li>- After rebooting, installation is completed.</li> </ul>	

## 4.2. ホスト OS からゲスト OS に SSH 接続するための設定

### – How to configure enabling SSH connections to CentOS.

1. VM を起動する - Boot VM.	
2. 右記のコマンドを実行 - Run the command.	# vi  /etc/sysconfig/network-scripts/ifcfg-eth0
3. 右記のように記述する - Edit the configuration file as follows. HWADDR には前節の手順 8 で控えた MAC アドレスを記述する。 - Input HWADDR as Mac Address the same as Step 8 in previous section.	DEVICE=eth0 HWADDR=xx:xx:xx:xx:xx:xx TYPE=Ethernet UUID=xxxxxx... ONBOOT=yes NM_CONTROLLED=yes IPADDR=192.168.56.10 # このアドレス宛に SSH アクセス可能になる (This address can be accessed by SSH) NETMASK=255.255.255.0 NETWORK=192.168.56.0 ※bootproto=dhcp は削除  USERCTL=no IPV6INIT=no PEERDNS=yes
4. ネットワーク設定を有効にするために右記のコマンドを実行する。 Run the command to restart network.	# /etc/init.d/network restart

## 4.3. 一般ユーザを追加する – Create a regular user account

root ユーザのままで全ての作業を行うことは安全面の都合上適切でない。そのため、作業用の一般ユーザを作成し、そのユーザに管理者権限を与える。

- Doing all work as the SuperUser can be dangerous. You could type a command incorrectly and destroy the system. Therefore, you create a regular user account and give the user administrative privileges.

### (1) ユーザの作成とパスワードの変更 - Create a user and set the password

以下のコマンドを実行する。user\_name は適宜置き換えること。

- Execute the following commands. (replace “user\_name” with your name.)

```
# groupadd user_name
# useradd -g user_name user_name
# passwd user_name
Changing password for user user_name.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

## (2) visudo コマンドを実行

以下のコマンドを実行する。

- Execute the following commands.

```
# visudo
```

コマンドを実行すると sudoer の設定ファイルが開く。vi と同じエディタ操作で編集することができる。

- When you run the visudo command, configuration file opens. You can edit it in the operation same as vi.

## (3) 新規ユーザに sudoer 権限を与える - Give the user sudo access

以下のようにファイルを編集する。

- Finally, edit the configuration file as follows.

```
## Allow root to run any commands anywhere
root    ALL=(ALL)    ALL
user_name    ALL=(ALL)    NOPASSWD: ALL    //この行を追加 - add this line
                                                //空白にはタブを利用 - Use Tab as space
```

以上でユーザの作成作業と管理者権限の付与は完了である。

- Creating a user account and giving administrator privileges are complete.  
ここからは root ユーザではなく新しく作成したユーザを利用して作業をする。
- From now on, work with the new account.

## 4.4. OS ディスクイメージのマウント - Mount OS disk image

ディスクイメージに同梱されているパッケージを利用するためにディスクイメージを VM にマウントする。

- Mount disk image on the VM to use rpm files.

### (1) 仮想マシンの電源を切る - Shutdown virtual machine

以下のコマンドを実行する。

- Execute following command.

```
$ sudo shutdown -h now
```

### (2) ディスクイメージのセット - Set disk image

4.1 と同様の手順で CentOS-6.4-x86\_64-bin-DVD1.iso をセットする。

- Set CentOS-6.4-x86\_64-bin-DVD1.iso the same as 4.1.

属性が以下のようにになっていることを確認する。

CD/DVD ドライブ(D): IDE プライマリスレーブ

### (3) 仮想マシンの起動 - Boot virtual machine

※インストール・ディスクを使用した場合は、VerchalBox の設定画面(システム)で HARDDISK を起動順序の一番上にして起動する。

### (4) ディスクイメージのマウント - Mount disk image

以下のコマンドを実行する。

Execute following commands.

```
$ sudo mkdir -p /mnt/media  
$ sudo mount /dev/sr0 /mnt/media
```

/dev/sr0 が無ければ/dev/sr1 を試すこと

### (5) マウント結果の確認 - Operation check

以下のコマンドを入力してパッケージの一覧が表示されることを確認する。

- Make sure display package list with following commands.

```
$ ls /mnt/media/Packages/  
389-ds-base-1.2.11.15-11.el6.x86_64.rpm  
389-ds-base-libs-1.2.11.15-11.el6.i686.rpm  
389-ds-base-libs-1.2.11.15-11.el6.x86_64.rpm  
/** 以下略 **/
```

## 5. SystemTap の環境構築 - Build SystemTap Environment

本章では SystemTap の環境を構築し、カーネルの監視ができるようになるまでの手順を述べる。

### 5.1. 前提条件

本章は以下の章の手順が完了していることを前提とする。

- 4. OS 環境構築 – Build Operating System environment

### 5.2. 必要なファイルの用意 - Prepare required files

#### (1) ファイルのダウンロード - Download files.

以下のコマンドを実行する。

- Execute following commands.

```
$ cd PackageDownloader/script
$ sh ./script/systemtap.sh
```

以下のディレクトリにファイルが保存される。

- PackageDownloader/package/dtrace
- PackageDownloader/package/systemtap

#### (2) ファイルの移動 - Replace files

以下のディレクトリ構成になるようにファイルを移動させる。

- Move directories such as the following structure.

```
PackageInstaller
├─ package
│   └─ dtrace
│   └─ systemtap
```

### 5.3. カーネルバージョンの確認 - Make sure kernel version

以下のコマンドを実行し、出力値が等しいことを確認する。

```
$ uname -r  
2.6.32-358.el6.x86_64
```

カーネルのバージョンに応じて適切なパッケージを選択する必要がある。そのため、出力値が上述した値と異なる場合は、作業環境に合ったパッケージを用意する必要がある。

#### 5.4. インストールの実行 - Execute installation

以下のコマンドを実行する。

- Execute following commands.

```
$ cd /home/user_name/work/PackageInstaller/script/  
$ sh systemtap.sh
```

#### 5.5. 動作確認 - Operation check

以下のコマンドを入力して以下の出力例と同様に『Pass 5: run completed』まで表示されればインストールは成功である。

- Finally, check the output when you run the following commands. Installation is successful if terminal appears as “Pass 5: run completed”.

```
$ sudo stap -v -e 'probe vfs.read {printf("read performed\n"); exit()}'  
Pass 1: parsed user script and 86 library script(s) using  
195564virt/24204res/3036shr/21512data kb, in 120usr/10sys/137real ms.  
Pass 2: analyzed script: 1 probe(s), 1 function(s), 3 embed(s), 0 global(s) using  
425420virt/123724res/8244shr/113384data kb, in 1250usr/410sys/2799real ms.  
Pass 3: translated to C into  
"/tmp/staprAAFYi/stap_46a1073d20dc9291206be1bd8ef51bc3_1471_src.c" using  
415604virt/119608res/6604shr/113384data kb, in 10usr/90sys/114real ms.  
Pass 4: compiled C into "stap_46a1073d20dc9291206be1bd8ef51bc3_1471.ko" in  
5130usr/1270sys/10965real ms.  
Pass 5: starting run.  
read performed  
Pass 5: run completed in 10usr/30sys/390real ms.
```

## 6. PostgreSQL の環境構築 – Build PostgreSQL environment

本章では PostgreSQL の環境を構築し、PostgreSQL の情報を SystemTap を用いて取得する手順について述べる。

### 6.1. 前提条件

本章は以下の章の手順が完了していることを前提とする。

- 4. OS 環境構築 – Build Operating System environment
- 5. SystemTap の環境構築- Build SystemTap Environment

### 6.2. 必要なファイルの用意 – Prepare required files

#### (1) ファイルのダウンロード - Download files.

以下のコマンドを実行する。

- Execute following commands.

```
$ cd PackageDownloader/script  
$ sh ./script/postgres.sh
```

以下のディレクトリにファイルが保存される。

- /PackageDownloader/package/postgres

#### (2) ファイルの移動 - Replace files

以下のディレクトリ構成になるようにファイルを移動させる。前章でダウンロードした dtrace も必要であることに注意すること。

- Move directories such as the following structure.

```
PackageInstaller  
├─ package  
│   └─ postgres
```

### 6.3. postgres ユーザの設定 – Configuration of postgres user

以下のコマンドを実行する。

- Execute following commands.

```
$ sudo groupadd postgres  
$ sudo useradd -g postgres postgres  
$ sudo passwd postgres
```

### 6.4. インストールの実行 - Execute installation

以下のコマンドを実行する。

- Execute following commands.

```
$ cd /home/user_name/work/PackageInstaller/script/  
$ sh postgres.sh
```

### 6.5. インストールディレクトリのオーナー変更 – Change owner of installed file

以下のコマンドを実行する。

- Execute following commands.

```
$ sudo chown postgres:postgres /usr/local/pgsql-9.2.4/
```

### 6.6. パスの編集 – Configuration of path

#### (1) postgres ユーザへ切り替える – Change to postgres user

以下のコマンドを実行する。

- Execute following commands.

```
$ su postgres  
Password:          # input password of postgres user
```

#### (2) .bash\_profile を編集 – Edit .bash\_profile

以下のコマンドで.bash\_profile の編集を開始する。

- Edit .bash\_profile with the following command.

```
$ vi /home/postgres/.bash_profile
```



以下の記述を『export PATH』の手前に追記する。

- Append the following description to /home/postgres/.bash\_profile before “export PATH”.

```
#PostgreSQL
POSTGRES_HOME=/usr/local/pgsql-9.2.4
export POSTGRES_HOME
PATH=$PATH:$HOME/bin:$POSTGRES_HOME/bin
export PGDATA=$POSTGRES_HOME/data
export PGLIB=$POSTGRES_HOME/lib
export LD_LIBRARY_PATH=$POSTGRES_HOME/lib
```

以下のコマンドで.bash\_profile の編集を有効にする。

- Enable .bash\_profile with the following command.

```
$ source /home/postgres/.bash_profile
```

## 6.7. データベースの初期化 – Init DB

以下のコマンドを postgres ユーザで実行する。

- Execute following command by postgres user.

```
$ initdb
```

## 6.8. データベースの起動 – Start up DB

以下のコマンドを postgres ユーザで実行する。

- Execute following command by postgres user.

```
$ pg_ctl start
```

## 6.9. 動作確認 – Operation check

以下のコマンドを実行して Probe の一覧が出力されることを確認する。この手順は postgres ユーザではなく作業ユーザで行う。

- Make sure that the list of Probes is output with the following commands by not postgres user but work user.

-

```
$ sudo stap -l 'process("/usr/local/pgsql-9.2.4/bin/postgres").mark("*')'  
process("/usr/local/pgsql-9.2.4/bin/postgres").mark("buffer__checkpoint__done")  
process("/usr/local/pgsql-9.2.4/bin/postgres").mark("buffer__checkpoint__start")  
process("/usr/local/pgsql-9.2.4/bin/postgres").mark("buffer__checkpoint__sync__start"  
)  
/** 中略 **/  
process("/usr/local/pgsql-9.2.4/bin/postgres").mark("wal__buffer__write__dirty__start"  
)  
process("/usr/local/pgsql-9.2.4/bin/postgres").mark("xlog__insert")  
process("/usr/local/pgsql-9.2.4/bin/postgres").mark("xlog__switch")
```

## 7. MySQL の環境構築 – Build MySQL environment

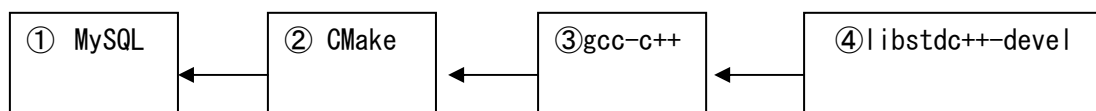
MySQL のインストールに必要なモジュールは以下の通りである。

- Modules required to install MySQL is as follows.

- ① MySQL
- ② CMake
- ③ gcc-c++
- ④ libstdc++-devel

各モジュールのバージョンは 2013 年 7 月 31 日時点での最新版である。なお、モジュール間の依存関係は図 7-1 モジュール間の依存関係の通りである。

- Each module to use the latest version in the July 31, 2013. Dependencies between modules are shown in Figure 7-1.



※モジュール間の矢印は依存を表す

- Arrow describes dependencies between modules.

図 7-1 モジュール間の依存関係  
dependencies between modules

MySQL のインストールに必要なファイルを用意する。

- Prepare the files necessary to install the MySQL.

### 7.1. gcc-c++ のインストール – Install gcc-c++

以下のコマンドを実行してパッケージをインストールする。

- Install with the following commands.

```
$ sudo rpm -ivh /mnt/media/Packages/libstdc++-devel-4.4.7-3.el6.x86_64.rpm  
$ sudo rpm -ivh /mnt/media/Packages/gcc-c++-4.4.7-3.el6.x86_64.rpm
```

## 7.2. CMake のインストール – Install CMake

インストール手順は以下の通り。

- Install with the following commands.

```
$ tar zxvf cmake-2.8.11.2.tar.gz
$ cd cmake-2.8.11.2
$ ./bootstrap --prefix=/opt/cmake-2.8.11
    #インストール先を指定 Specify install target path
$ make
$ sudo make install
```

/home/user\_name/.bash\_profile に以下の行を追加

- Append the following description to /home/user\_name/.bash\_profile.

```
#CMake
PATH=$PATH:/opt/cmake-2.8.11/bin/
```

.bash\_profile の設定を有効にするために、シェルを再起動するか、以下のコマンドを実行する。

- Reload .bash\_profile or reboot your shell for reflecting configuration.

```
$ source /home/user_name/.bash_profile
```

CMake が利用可能になったか、以下のコマンドで確認する。

正しく CMake のインストールが行われていたら以下のように出力される。

- If installation is successful, output will be as follows.

```
$ which cmake
/opt/cmake-2.8.11/bin/cmake
```

## 7.3. MySQL のインストール – Install MySQL

必要となるファイルを以下の URL からダウンロードする。ダウンロードには Oracle のユーザ登録が必要である。

- Get the files from the following URL. You need to create Oracle account.

<http://dev.mysql.com/downloads/mysql/>

ダウンロードページを開いたら、Select Platform のプルダウンメニューで Source code を

選択する(図 7-2)。そして、Generic Linux (Architecture Independent), Compressed TAR Archive をダウンロードする(図 7-3)。

- Select “Source Code” at drop-down menu(図 7-2). Download Compressed TAR Archive(図 7-3)

Select Platform:



図 7-2 プルダウンメニュー



図 7-3 ダウンロード対象

必要なファイルの用意が完了したら、以下の手順でインストールを行う。

- Install with the following commands.

ビルドを行う際に `--enable-dtrace` と `--with-debug` のオプションを忘れずに追加すること。このオプションをつけずにビルドすると SystemTap を用いた監視ができない。

- Note that don't forget `--enable-dtrace` and `--with-debug` options.

なお、`make` には非常に時間がかかる。マシン性能にも依るが約 2 時間かかった。

- “Make” command takes long time to complete. (It took about 2 hours.)

```
# sudo rpm -ivh /mnt/media/Packages/libaio-0.3.107-10.el6.x86_64.rpm
# sudo rpm -ivh /mnt/media/Packages/libaio-devel-0.3.107-10.el6.x86_64.rpm
# sudo rpm -ivh /mnt/media/Packages/bison-2.4.1-5.el6.x86_64.rpm
$ tar zxvf mysql-5.6.13.tar.gz
$ cd mysql-5.6.13
$ ./BUILD/autorun.sh
$ ./configure --with-debug --enable-dtrace --prefix=/usr/local/mysql-5.6.13
$ make
$ sudo make install
```

## 7.4. 動作確認 1: probe 一覧の表示

以下のコマンドを用いて probe の一覧が表示されることを確認する。

- Make sure that the list of Probes is output with the following commands.

```
$ sudo stap -L 'process("/usr/local/mysql-5.6.13/bin/mysqld").mark("*")'
```

ここまでの手順が正しく行われている場合、以下のように出力される。

- Make sure that the list of Probes is output with the following commands.

```
process("/usr/local/mysql-5.6.13/bin/mysqld").mark("command__done") $arg1:long
process("/usr/local/mysql-5.6.13/bin/mysqld").mark("command__start")    $arg1:long
$arg2:long $arg3:long $arg4:long
process("/usr/local/mysql-5.6.13/bin/mysqld").mark("connection__done")  $arg1:long
$arg2:long
/** 中略 **/
process("/usr/local/mysql-5.6.13/bin/mysqld").mark("update__row__done") $arg1:long
process("/usr/local/mysql-5.6.13/bin/mysqld").mark("update__row__start") $arg1:long
$arg2:long
```

## 7.5. MySQL の起動 – Start up MySQL

mysql グループとユーザの作成と、パスワードの変更を以下のコマンドで実行する。

- Create “mysql” user and group and set password with the following commands.

```
$ sudo groupadd mysql
$ sudo useradd -g mysql mysql
$ sudo passwd mysql
```

データを格納するディレクトリのオーナーを変更する。

- Change owner of data directory.

```
$ sudo chown mysql:mysql /usr/local/mysql-5.6.13/data/
```

以下のコマンドでデータベースを初期化する。

- Initialize DB with the following commands.(by mysql user)

```
$ su mysql
$ cd /usr/local/mysql-5.6.13/
$ /usr/local/mysql-5.6.13/scripts/mysql_install_db
```

my.cnf の以下の行を編集する。

※cp ./support\_files/my\_default.cnf /etc/my.cnf で my.cnf を事前にコピー

(既存ファイルのバックアップを取り、mysql ユーザーは/etc 以下の書き込めないので root で作業する。)

- Append the following description to my.cnf.

```
# These are commonly set, remove the # and set as required.  
#インストール先ディレクトリ (Install target directory)  
basedir = /usr/local/mysql-5.6.13/  
#データを格納するディレクトリ (Directory to store data)  
datadir = /usr/local/mysql-5.6.13/data/
```

以下のコマンドを実行すると MySQL のデーモンが起動する。

- Start MySQL daemon with the following commands.(by mysql user)

```
$ su mysql  
$ cd /usr/local/mysql-5.6.13/  
$ /usr/local/mysql-5.6.13/bin/mysqld
```

## 8. PHP の環境構築

必要となるファイルを以下の URL からダウンロードする。

- Get the files from the following URL:
- <http://www.php.net/get/php-5.5.1.tar.gz/from/jp1.php.net/mirror>
- <http://ftp.gnu.org/gnu/autoconf/autoconf-latest.tar.gz>
- パッチファイル(添付) - Patch file

### 8.1. autoconf のインストール - Install autoconf

インストール手順は以下の通り。

- Install with the following commands.

```
$ tar zxvf autoconf-2.69.tar.gz
$ cd autoconf-2.69
$ ./configure
$ make
$ sudo make install
```

### 8.2. Apache のインストール

事前に lua 5.1.5 のインストールをする。(apache2.4.6 の mod\_lua に対応するバージョン)  
make でプラットフォームの一覧が表示される。

Linux を指定して make を実行。

```
$ tar zxvf lua-5.1.5.tar.gz
$ cd lua-5.1.5
$ make linux
$ sudo make install
```

Apache のインストール手順は以下の通り。

- Install with the following commands.

※apr, apr-util, pcre は rpm でインストールした場合.../lib64 の下に入ってしまい、認識されないため、tar ファイルを入手してビルドインストールする。



```
$ tar zxvf apr-1.4.8.tar.gz
$ cd apr-1.4.8
$ ./configure
$ make
$ sudo make install

$ tar zxvf apr-util-1.5.2.tar.gz
$ cd apr-util-1.5.2
$ ./configure
$ make
$ sudo make install

$ tar zxvf pcre-8.33.tar.gz
$ cd pcre-8.33
$ ./configure
$ make
$ sudo make install

$ cd /home/user_name/work/installer/package/apache/
$ tar zxvf httpd-2.4.6.tar.gz
$ cd httpd-2.4.6
$ ./configure ¥
--with-apr=/usr/local/apr/ ¥
--with-mpm=prefork ¥
--prefix=/usr/local/apache-2.4.6/ ¥
--with-apr-util=/usr/local/apr ¥
--enable-modules=all ¥
--enable-mods-shared=all ¥
--enable-mpm5-shared='prefork worker event'¥
--enable-lua -enable-sed
$ make
$ sudo make install
```

lua.conf を/usr/local/apache-2.4.6/conf/extra に置く

lua.conf

```
LoadModule lua_module modules/mod_lua.so
AddHandler lua-script .lua
```

/usr/local/apache-2.4.6/conf/httpd.conf に追記する

```
Include conf/extra/lua.conf
```

lua.conf の設定に従い、/var/www 配下にテスト用のファイルを配置する。

```
$ pwd
/var/www/lua
$ ls
handle_quick.lua  html
$ ls html
axs_test.html  index.html
```

※テスト用ファイルは GitHub の doc ディレクトリに var\_www\_lua として配置されています。

handle\_quick.lua が URL へのアクセスを検出する lua のスクリプトであり、html 配下にあるのはアクセス試験用の Web ページである。

この試験を実施する場合は、httpd.conf で DocumentRoot を変更し、httpd を再起動する。

```
#DocumentRoot "/usr/local/apache-2.4.6/htdocs"
#<Directory "/usr/local/apache-2.4.6/htdocs">
DocumentRoot "/var/www/lua/html"
<Directory "/var/www/lua/html">
```

なお、lua の出力は/tmp/access.log に対して行われるので、Demo 用ツールを介して ENdoSnipeCommutator を経由し、DataCollector に渡すことで、DashBord に URL 毎のアクセス状況を示すグラフとなって表示される。

```
$ tail -f /tmp/access.log | ¥
./Demo/luademotool | ./ENdoSnipeCommutator/ens_commutator ¥
./ENdoSnipeCommutator/conf/commutator.properties
```

※luademotool は GitHub の doc ディレクトリに配置されています。別途ビルドしてください。

Apache の使用ポートのアクセス制御

/etc/sysconfig/iptables に以下の 1 行を追加する。

- Append the following description to /etc/sysconfig/iptables.

```
-A INPUT -p tcp -m tcp --dport 80 --tcp-flags FIN,SYN,RST,ACK SYN -j ACCEPT
```

以下のコマンドを実行して iptables を再起動し、Apache を起動する。

- Restart iptables and start Apache up with the following commands.

```
$ sudo /etc/init.d/iptables restart  
$ sudo /usr/local/apache-2.4.6/bin/apachectl start
```

/home/user\_name/.bash\_profile に以下の記述を追記する。

- Append the following description to /home/user\_name/.bash\_profile.

```
#Apache  
PATH=$PATH:/usr/local/apache-2.4.6/bin/
```

Web ページを格納するディレクトリのオーナーを変更する。

- Change owner of web page directory.

また、同梱する htdocs ファイルの中身を移動しておく。

- Move htdocs files to /usr/local/apache-2.4.6/htdocs

```
$ sudo chown -R user_name:user_name /usr/local/apache-2.4.6/htdocs/
```

ここまでの手順が正しく行われている場合、ブラウザで <http://192.168.56.10> にアクセスすると『It works!』と表示される。

- Make sure that the “It works!” is output with connecting <http://192.168.56.10> by Browser.

### 8.3. PHP のインストール - install PHP

インストール手順は以下の通り。

- Install with the following commands.

libxml2 を rpm よりインストールした場合、.../lib64 の下に入って認識されないなので tar ファイルを入手してインストールする。

```
$ sudo rpm -ivh /mnt/media/Packages/patch-2.6-6.el6.x86_64.rpm
$ tar zxvf libxml2-2.6.30.tar.gz
$ cd libxml2-1.6
$ ./configure
$ make
$ sudo make install

$ tar zxvf php-5.5.1.tar.gz
$ cd php-5.5.1
$ patch < patch.txt
$ ./buildconf --force
$ ./configure --enable-dtrace ¥
--prefix=/usr/local/php-5.5.1/ ¥
--with-apxs2=/usr/local/apache-2.4.6/bin/apxs ¥
--with-mysql=/usr/local/mysql-5.6.13/ ¥
--with-pgsql=/usr/local/pgsql-9.2.4/ ¥
$ make
$ sudo make install
```

※patch.txt はダウンロードした php.patch よりコピーする。

.bash\_profile に以下の行を追加

- Append the following description to .bash\_profile.

```
#PHP
PATH=$PATH:/usr/local/php-5.5.1/bin
```

.bash\_profile の設定を有効にするために、シェルを再起動するか、以下のコマンドを実行する。

- Reload .bash\_profile or reboot your shell for reflecting configuration.

```
$ source .bash_profile
```

PHP が利用可能になったか、以下のコマンドで確認する。

正しく PHP のインストールが行われていたら以下のように出力される。

- If installation is successful, output will be as follows.

```
$ which php
/usr/local/php-5.5.1/bin/php
```

#### 8.4. 動作確認 1: probe 一覧の表示 - Operation check 1: Display probe list

以下のコマンドを用いて probe の一覧が表示されることを確認する。

- Make sure that the list of Probes is output with the following commands.

```
$ sudo stap -L 'process("/usr/local/php-5.5.1/bin/php").mark("*")'
```

ここまでの手順が正しく行われている場合、以下のように出力される。

- Make sure that the list of Probes is output with the following commands.

```
process("/usr/local/php-5.5.1/bin/php").mark("compile__file__entry")    $arg1:long
$arg2:long
process("/usr/local/php-5.5.1/bin/php").mark("compile__file__return")    $arg1:long
$arg2:long
process("/usr/local/php-5.5.1/bin/php").mark("error")    $arg1:long    $arg2:long
$arg3:long
process("/usr/local/php-5.5.1/bin/php").mark("exception__caught") $arg1:long
process("/usr/local/php-5.5.1/bin/php").mark("exception__thrown") $arg1:long
process("/usr/local/php-5.5.1/bin/php").mark("execute__entry")    $arg1:long
$arg2:long
process("/usr/local/php-5.5.1/bin/php").mark("execute__return")    $arg1:long
$arg2:long
```

#### 8.5. Web アプリケーション監視環境の構築 - Build monitoring web application environment

/usr/local/apache-2.4.6/conf/httpd.conf に以下の記述を追加する。


- Append the following description to /usr/local/apache-2.4.6/conf/httpd.conf.

```
<FilesMatch ¥.php$>
    SetHandler application/x-httpd-php
</FilesMatch>
```

ここまでの手順が正しく行われている場合、ブラウザで `http://192.168.56.10/info.php` にアクセスすると図 8-1 のように表示される。

- Make sure that figure 8-1 is output with connecting `http://192.168.56.10` by Browser.

Webpage Screenshot

PHP Version 5.5.1	
	
System	Linux localhost.localdomain 2.6.32-358.6.2.el6.x86_64 #1 SMP Thu May 16 20:59:36 UTC 2013 x86_64
Build Date	Aug 3 2013 12:52:43
Configure Command	"/configure" "--enable-dtrace" "--prefix=/usr/local/php-5.5.1/" "--with-apxs2=/usr/local/apache-2.4.6/bin/apxs" "--with-mysql=/usr/local/mysql-5.6.12/" "--with-pgsql=/usr/local/pgsql/"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	/usr/local/php-5.5.1/lib
Loaded Configuration File	(none)
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20121113
PHP Extension	20121212
Zend Extension	220121212
Zend Extension Build	API20121212.TS
PHP Extension Build	API20121212.TS

http://192.168.56.10/hello.php

図 8-1 phpinfo のブラウザ画面出力

Browser output of phpinfo

## 8.6. Web アプリケーションの監視 – Monitoring Web application

以下のコマンドを入力すると、PHP の Web アプリケーションを監視するサンプルスクリプトが動作する。

- Start monitoring script with the following command.

```
$ sudo stap /home/user_name/work/sample/stap/php_mod.stp
```

ブラウザで <http://192.168.56.10/hello.php> にアクセスすると、呼び出された関数が標準出力に出力されることを確認する。出力の一例は以下の通り。

- Make sure that logs are output with connecting <http://192.168.56.10/hello.php> by Browser.

```
Function entry: File /usr/local/apache-2.4.6/htdocs/hello.php, Class , Function name
hello
```

## 9. Python の環境構築

### 9.1. Python のインストール - Install Python

インストール手順は以下の通り。

- Install with the following commands.

```
$ sudo rpm -ivh /mnt/media/Packages/python-libs-2.6.6-36.el6.x86_64.rpm
$ sudo rpm -ivh /mnt/media/Packages/python-2.6.6-36.el6.x86_64.rpm
$ sudo rpm -ivh /mnt/media/Packages/python-devel-2.6.6-36.el6.x86_64.rpm
```

※debuginfo は無くても差し支えない。

※python は rpm で.../lib64 配下に入っても差し支えない。

Python が利用可能になったか、以下のコマンドで確認する。

正しく Python のインストールが行われていたら以下のように出力される。

- If installation is successful, output will be as follows.

```
$ which python
/usr/bin/python
```

以下のコマンドを用いて probe の一覧が表示されることを確認する。

- Make sure that the list of Probes is output with the following commands.

```
$ sudo stap -L "python.*.*"
```

ここまでの手順が正しく行われている場合、以下のように出力される。

- Make sure that the list of Probes is output with the following commands.

```
python.function.entry  filename:string  funcname:string  lineno:long  $arg1:long
$arg2:long $arg3:long
python.function.return  filename:string  funcname:string  lineno:long  $arg1:long
$arg2:long $arg3:long
```

## 9.2. Web アプリケーション監視環境の構築 - Build monitoring web application environment

mod\_wsgi のインストール手順は以下の通り。

- Install with the following commands.

```
$ cd /home/user_name/work/PackageInstaller/package/python/  
$ tar zxvf mod_wsgi-3.4.tar.gz  
$ cd mod_wsgi-3.4  
$ ./configure ¥  
--with-apxs=/usr/local/apache-2.4.6/bin/apxs ¥  
--with-python=/usr/bin/python  
$ make  
$ sudo make install
```

/usr/local/apache-2.4.6/conf/httpd.conf に以下の記述を追加する。

- Append the following description to /usr/local/apache-2.4.6/conf/httpd.conf.

```
LoadModule wsgi_module modules/mod_wsgi.so  
WSGIScriptAlias /python /usr/local/apache-2.4.6/htdocs/sample.wsgi
```

Apache を再起動する。

- Restart apache with the following commands.

```
$ sudo apachectl stop  
$ sudo apachectl start
```

## 9.3. Web アプリケーションの監視 - Monitoring Web application

以下のコマンドを入力すると、Python の Web アプリケーションを監視するサンプルスクリプトが動作する。

- Start monitoring script with the following command.

```
$ sudo stap /home/user_name/work/sample/stap/python_mod.stp
```

ブラウザで <http://192.168.56.10/python> にアクセスすると、呼び出された関数が標準出力に出力されることを確認する。出力の一例は以下の通り。

- Make sure that logs are output with connecting <http://192.168.56.10/python> by



Browser.

```
Function entry: File /usr/local/apache-2.4.6/htdocs/sample.wsgi, Function name  
application
```

```
Function entry: File /usr/local/apache-2.4.6/htdocs/sample.wsgi, Function name hello
```

## 10. Ruby の環境構築

### 10.1. Ruby のインストール - Install Ruby

インストール手順は以下の通り。

- Install with the following commands.

```
$ tar zxvf ruby-2.0.0-p247.tar.gz
$ cd ruby-2.0.0-p247
$ ./configure --prefix=/usr/local/ruby-2.0.0-p247/
$ make
$ sudo make install
```

`.bash_profile` に以下の行を追加

- Append the following description to `.bash_profile`.

```
#Ruby
PATH=$PATH:/usr/local/ruby-2.0.0-p247/bin
```

`.bash_profile` の設定を有効にするために、シェルを再起動するか、以下のコマンドを実行する。

- Reload `.bash_profile` or reboot your shell for reflecting configuration.

```
$ source .bash_profile
```

Ruby が利用可能になったか、以下のコマンドで確認する。

正しく Ruby のインストールが行われていたら以下のように出力される。

- If installation is successful, output will be as follows.

```
$ ruby -v
ruby 2.0.0p247 (2013-06-27 revision 41674) [x86_64-linux]
```

### 10.2. 動作確認 1: probe 一覧の表示

以下のコマンドを用いて `probe` の一覧が表示されることを確認する。

- Make sure that the list of Probes is output with the following commands.

```
$ sudo stap -L 'process("/usr/local/ruby-2.0.0-p247/bin/ruby").mark("*")'
```

ここまでの手順が正しく行われている場合、以下のように出力される。

- Make sure that the list of Probes is output with the following commands.

```
process("/usr/local/ruby-2.0.0-p247/bin/ruby").mark("array__create")    $arg1:long
$arg2:long $arg3:long
process("/usr/local/ruby-2.0.0-p247/bin/ruby").mark("cmethod__entry")  $arg1:long
$arg2:long $arg3:long $arg4:long
process("/usr/local/ruby-2.0.0-p247/bin/ruby").mark("cmethod__return") $arg1:long
$arg2:long $arg3:long $arg4:long
/** 中略 **/
process("/usr/local/ruby-2.0.0-p247/bin/ruby").mark("require__entry")  $arg1:long
$arg2:long $arg3:long
process("/usr/local/ruby-2.0.0-p247/bin/ruby").mark("require__return") $arg1:long
$arg2:long $arg3:long
process("/usr/local/ruby-2.0.0-p247/bin/ruby").mark("string__create")  $arg1:long
$arg2:long $arg3:long
```

### 10.3. Web アプリケーション監視環境の構築- Build monitoring web application environment

Ruby on Rails を用いた Web アプリケーション環境を構築する。

- Build Ruby on Rails web application environment.

-

最初に、Rails のインストールに用いる gem コマンドを有効にするために openssl をインストールする。実行するコマンドは以下の通り。

- Install openssl with the following commands.

```
$ cd /home/user_name/work/PackageInstaller/script/
$ sh ./openssl.sh # 自動インストールスクリプトを利用 - use auto install script
$ cd /home/user_name/work/package/ruby/ruby-2.0.0-p247/ext/openssl
$ ruby extconf.rb
$ make
$ sudo make install
```

root の .bash\_profile に ruby の PATH を追加する。

```
#Ruby
PATH=$PATH:/usr/local/ruby-2.0.0-p247/bin
```

次に、Rails とそれに必要なパッケージをインストールする。

- Install required packages with the following commands.

```
$ cd /home/user_name/work/script/  
$ sh ./rails.sh # 自動インストールスクリプトを利用 - use auto install script
```

次に、Rails のサンプルアプリケーションを以下のコマンドで作成する。

- Make sample application using Rails with the following commands.

```
$ mkdir /home/user_name/work/webapp  
$ mkdir /home/user_name/work/webapp/rails  
$ cd /home/user_name/work/webapp/rails/  
$ rails new helloworld
```

外部ネットワークに接続できない環境においては以下のエラーメッセージが最後に出力されるが、このまま手順を続けて良い。

- Don't worry this error message.

```
Fetching source index from https://rubygems.org/  
Could not fetch specs from https://rubygems.org/
```

./helloworld/Gemfile に以下の記述を追加する。

- Append the following description to ./helloworld/Gemfile.

```
gem 'execjs'  
gem 'therubyracer'
```

最後に、3000 番ポートの通信を許可する設定に変更する。

これは、Ruby on Rails の Web アプリケーションが 3000 番ポートを用いて通信を行うためである。

ネットワーク設定ファイルを \$ sudo vi /etc/sysconfig/iptables で編集する。

以下の行を追加する。

- Append the following description to /etc/sysconfig/iptables.

```
-I INPUT -p tcp -m tcp --dport 3000 --syn -j ACCEPT
```

以下のコマンドを実行して iptables を再起動し、Rails サーバを起動する。

- Restart iptables and start Rails server up with the following commands.

```
$ sudo /etc/init.d/iptables restart
$ cd helloworld/
$ rails server
```

以上の手順までで、Web サーバが起動した。

ブラウザで <http://192.168.56.10:3000/> にアクセスすると図 10-1 のように表示される。

- Make sure that Figure10-1 is output with connecting <http://192.168.56.10:3000/> by Browser.

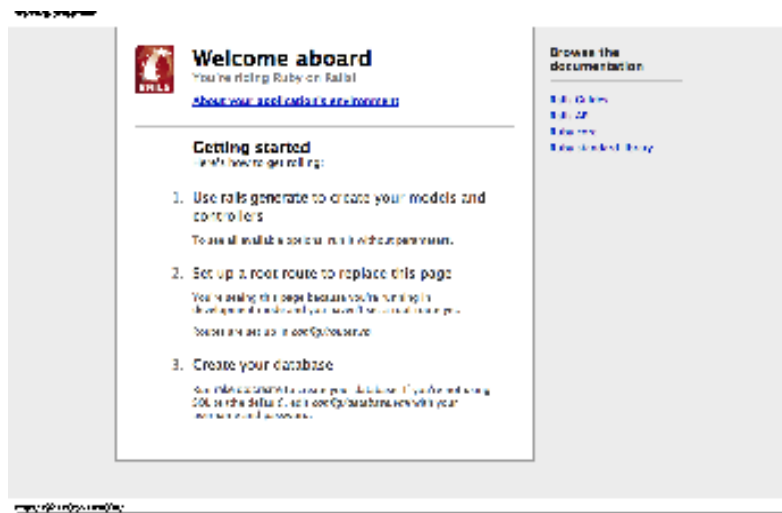


図 10-1 ブラウザ表示

以下のコマンドを入力して簡単な web アプリケーションのサンプルを作成する。

- Make simple web application with the following commands.

```
$ cd /home/user_name/work/webapp/rails/helloworld
$ rails g scaffold product name price:integer
$ rake db:migrate
```

ブラウザで <http://192.168.56.10:3000/products> にアクセスすると Listing products と表示される。“New Product”のリンクをクリックしてデータを入力すると、表示されるデータを追加することができる。

- Make sure that the “Listing products” is output with connecting <http://192.168.56.10> by Browser. You can add date by “New Product”.

## 10.4. Web アプリケーションの監視 - Monitoring Web application

以下のコマンドを入力すると、Ruby on Rails の Web アプリケーションを監視するサンプルスクリプトが動作する。

- Start monitoring script with the following command.

```
$ sudo stap /home/user_name/work/sample/stap/rails.stp
```

先ほど構築したサンプルアプリケーションを操作すると、呼び出された関数が標準出力に出力されることを確認する。出力の一例は以下の通り。

- Make sure that logs are output with connecting `http://192.168.56.10:3000/product` by Browser.

```
Method entry: File
/home/user_name/work/webapp/rails/helloworld/app/controllers/products_controller.r
b, Class ProductsController, Method name set_product
Method entry: File
/home/user_name/work/webapp/rails/helloworld/app/controllers/products_controller.r
b, Class ProductsController, Method name edit
```

## 11. 参考情報 - Reference

### 11.1. SystemTap

- [http://sourceware.org/systemtap/SystemTap\\_Beginners\\_Guide/](http://sourceware.org/systemtap/SystemTap_Beginners_Guide/)

### 11.2. PostgreSQL

- <http://www.postgresql.org/docs/9.2/static/dynamic-trace.html>

### 11.3. MySQL

- <http://www.slideshare.net/posullivan/monitoring-mysql-with-dtracesystemtap>
- <http://dev.mysql.com/doc/refman/5.5/en/dba-dtrace-mysqld-ref.html>