

計算機クラスターの 環境構築とMPI導入

社会人向けスパコン実践セミナー

2009年6月16日(火)

担当：小林泰三

連絡先：tkoba@cc.kyushu-u.ac.jp

お願い

- **いつでも質問！**
- **分からなくなったら直ぐに質問！！**
- **腑に落ちる迄質問！！！！**

はじめに

- **する事**

- Linux(CentOS5.3/x86_64)を**計算クラスター向き構成**でインストール
- MPI(mpich2 v1.1)を標準構成でインストール

- **方針**

- 応用や発展性を考慮
- 設定作業はターミナルで

- **進行**

- 「説明」→「実演」→「実習」のサイクルで

もくじ

• Linuxインストール編

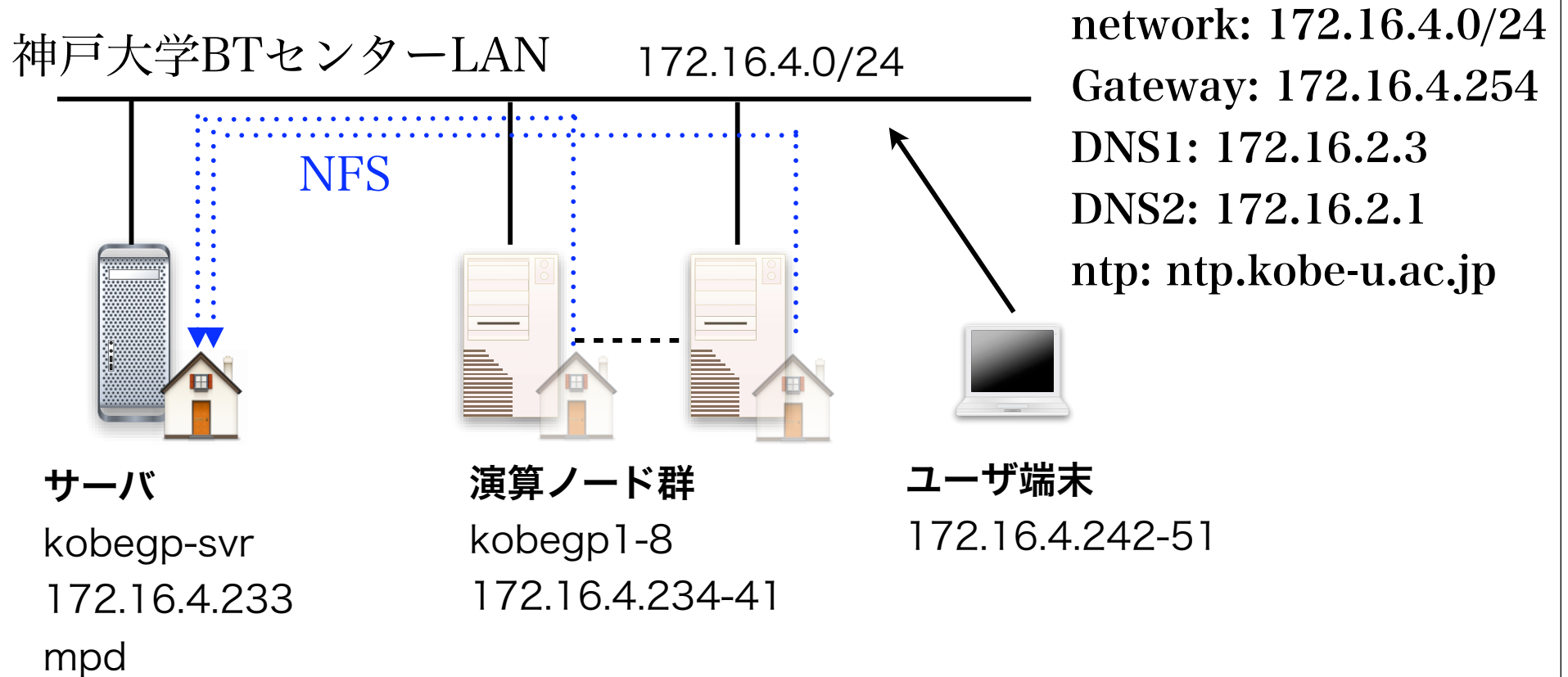
- 下準備：ケーブル類の接続や設定項目の整理
- インストール：インストーラーでの作業と設定

- 設定：計算クラスター向けの設定

• MPIインストール編

- 下準備：ソースを取得
- インストール：configure make を実行
- 設定：環境変数などの設定

全体の構成



- /home, /opt をNFSで共有
- /etc/hosts でhost管理

Linuxインストール

下準備

- **ハードの設定**
 - 電源、キーボード、マウス、モニタの結線
 - ネットワークケーブルの結線
 - BIOS設定の確認
 - boot の順番： CD/DVD ドライブが最初

<実演><実習>

CD boot で network install

1. ネットワークの設定
2. OS Repository を指定
http://ftp.riken.go.jp/Linux/centos/5.3/os/x86_64/
3. 第2ステージへ（これ以降はDVDからのインストールと同様）

DVDからインストール

1. メディアのチェック → skip
2. 言語/キーボードの選択 → 日本語
- 3. HDDのフォーマットとパーティショニング**
4. ネットワークの設定
5. タイムゾーンの設定
6. rootパスワードの設定
- 7. パッケージの選択**

この後、HDDのフォーマットと
パッケージのインストールが始まる

HDDの設定

- カスタム設定を選択

- パーティショニング

1. /boot 300MB

2. / 50GB

3. swap 20GB

4. /opt 20GB (※今回はサーバのみ)

5. /home 残り全て (※今回はサーバのみ)

Linux kernel と boot img

OS と ディストリ提供の
アプリケーションソフト

搭載メモリ量が目安

ディストリ以外のソフト

パッケージの選択

1. “Desktop - Gnome” をチェック

2. 「今すぐカスタマイズ」 をチェック

1. デスクトップ環境
GNOME デスクトップ環境
2. アプリケーション
Emacs, エディタ, グラフィカルインターネット, グラフィックス, 技術系と科学系
3. 開発
X ソフトウェア開発, 開発ツール, 開発ライブラリ
4. ベースシステム
X Window System

実演

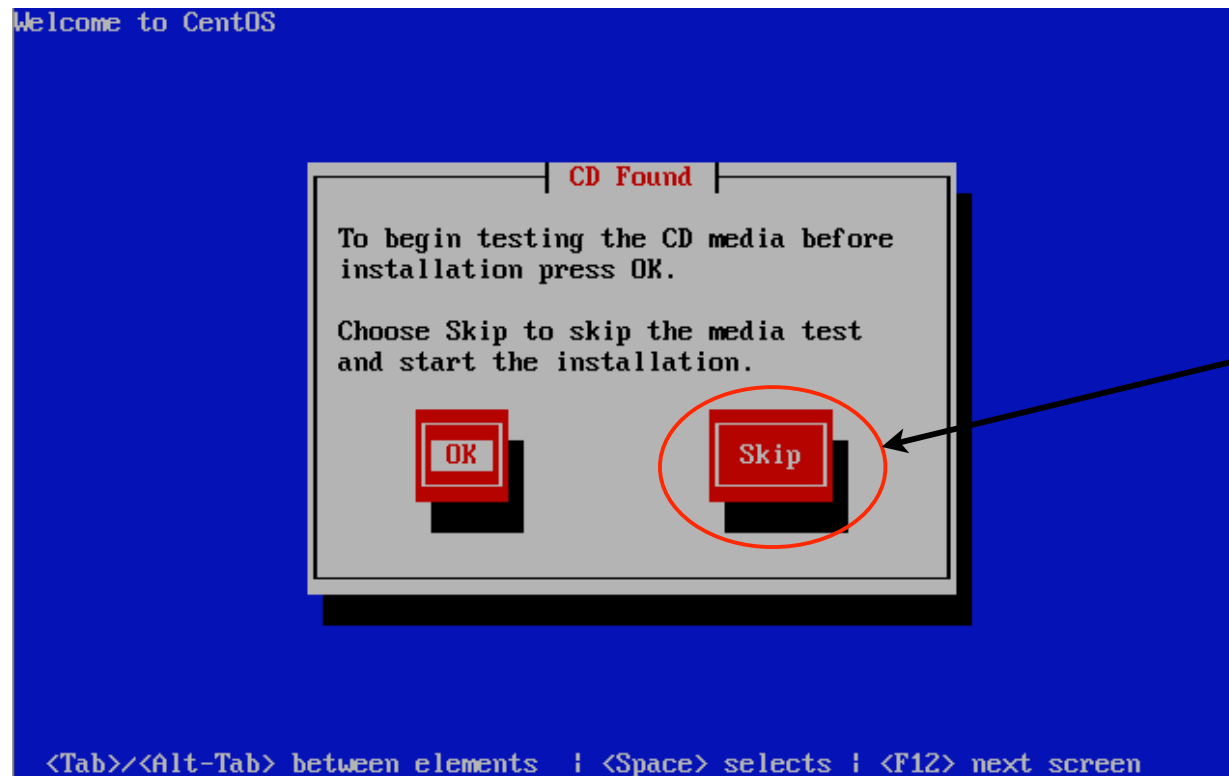
CentOS5.3/x86_64 の インストール

実演 (boot画面)



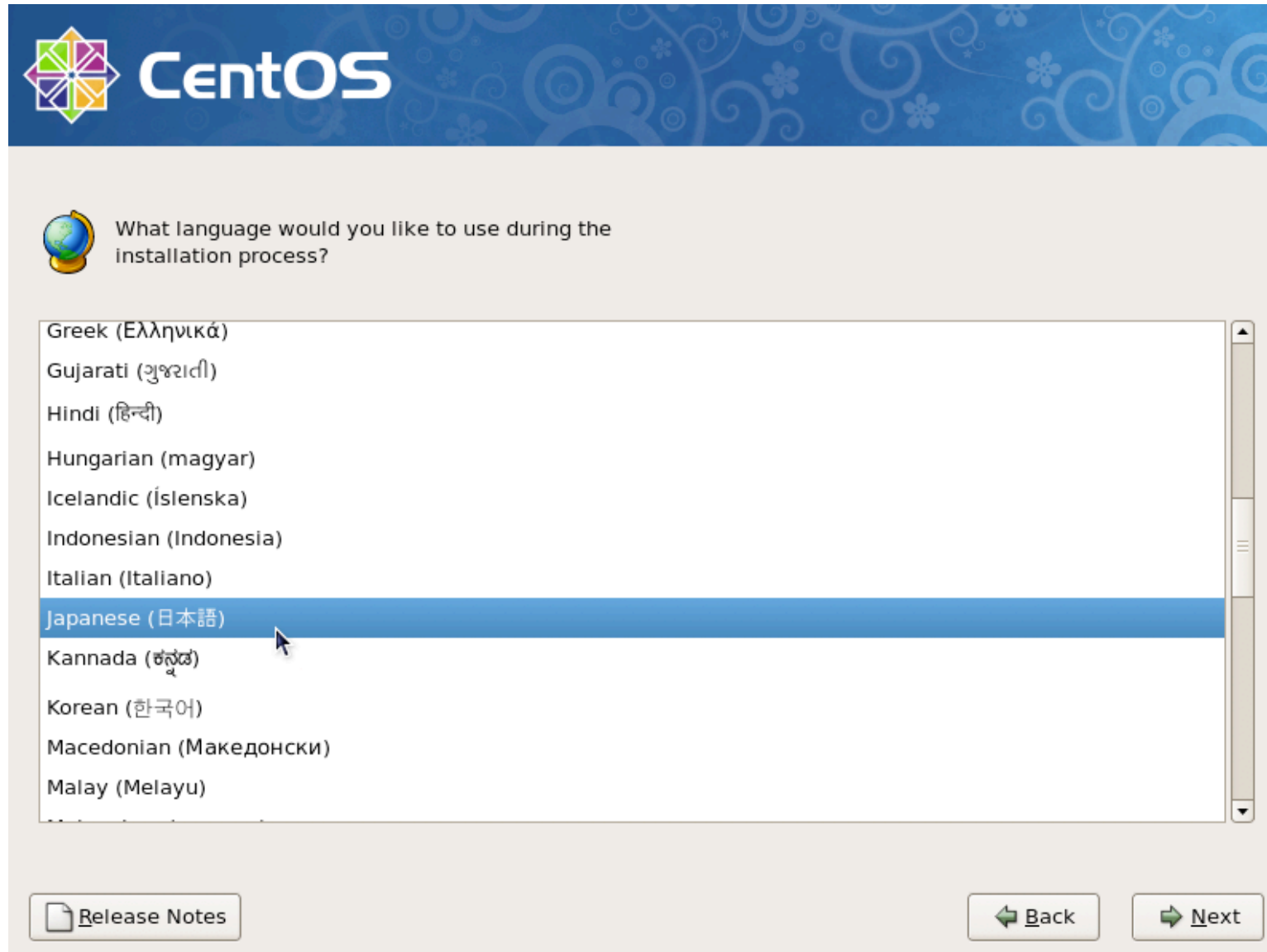
- Enter キーを押して起動

実演（メディアチェック）




- “Skip” を選択
（選択にはTab
キーを利用）

実演（言語設定）



実演（キーボード設定）



このシステム用の適切なキーボードを選択します。

- マケドニア語
- ラテンアメリカ語
- ルーマニア語
- ロシア語
- ロシア語 (Microsoft)
- ロシア語 (cp1251)
- ロシア語 (ru1)
- ロシア語 (ru2)
- ロシア語 (utf8ru)
- ロシア語 (win)
- 日本語**
- 英語 (U.S. インターナショナル)
- 英語 (アメリカ合衆国)
- 英語 (英国)

リリースノート(R) 戻る(B) 次(N)

実演 (HDD設定0)

システム用の適切なキーボードを選択します。

警告

デバイス hda (Virtual HDD [0] 32765 MB) にあるパーティションテーブルを読み込めませんでした。
新規パーティションを作成するには初期化が必要ですが、本ドライブにあるすべてのデータを失うことになります。

この操作は、以前のインストールで選択した無視するドライブの選択よりも優先されます。

すべてのデータを消去して、このドライブを初期化しますか?

いいえ(N) はい(Y)

● 「はい」を選択

ターナショナル)

実演 (HDD設定 1)

インストールには、ハードドライブのパーティション設定が必要です。デフォルトでは、ほとんどのユーザーに適しているパーティション設定のレイアウトが選択されます。デフォルト

選択したドライブ上のすべてのパーティションを削除してデフォルトのレイアウトを作成します。

選択したドライブ上の linux パーティションを削除してデフォルトのレイアウトを作成します。

選択したドライブ上の空き領域を使用して、デフォルトレイアウトを作成します。

カスタムレイアウトを作成します。

このインストールに使用するドライブを選択してください(S)。

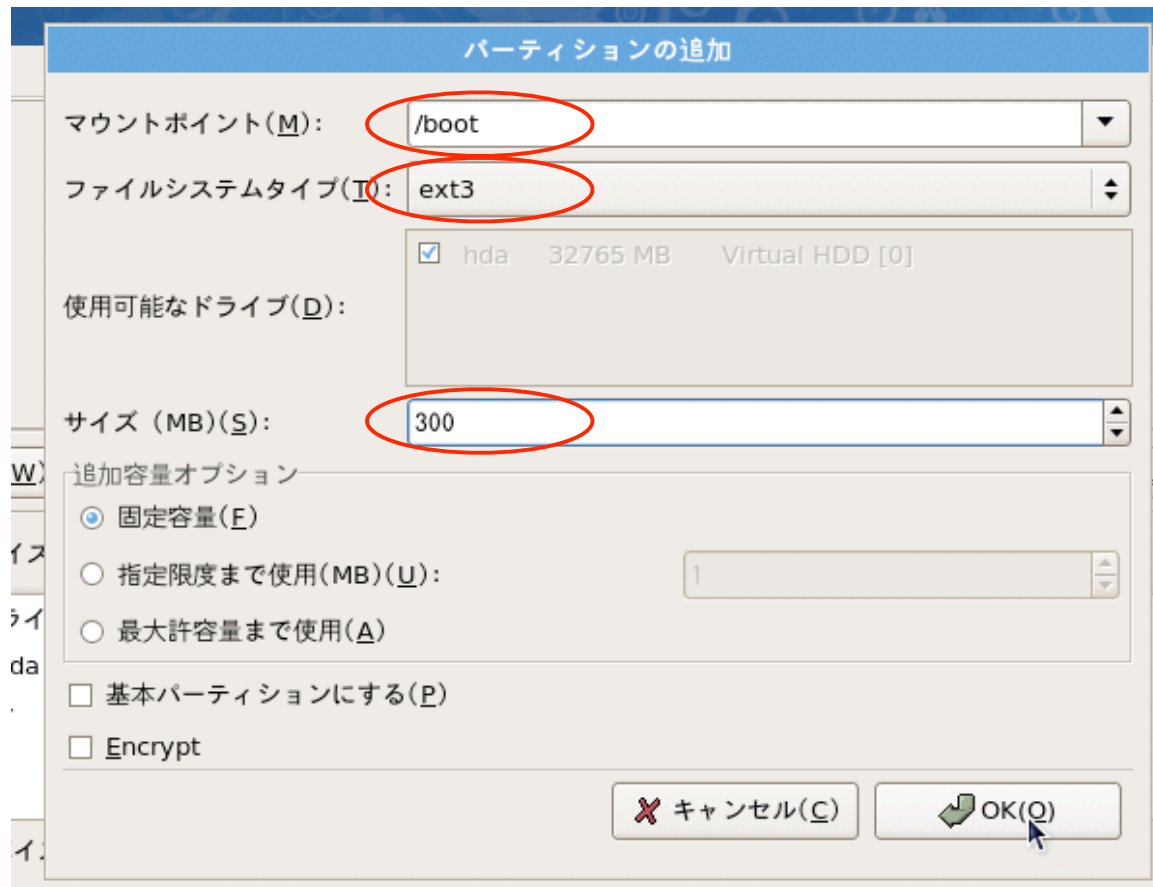
<input checked="" type="checkbox"/>	hda	32765 MB	Virtual HDD [0]
-------------------------------------	-----	----------	-----------------

+ 高度なストレージ設定 (A)

パーティションレイアウトの再確認と変更(V)

- パーティショニング
- 「カスタムレイアウトを作成します。」を選択
- 画面右下の「次」をクリック

実演 (HDD設定 2)



- パーティショニング
- 「新規」を選択して「マウントポイント」「ファイルシステムタイプ」「サイズ」を設定
- 画面右下の「OK」をクリック

全てのパーティションで繰り返す。

実演 (HDD設定 2)

ドライブ /dev/hda (32765 MB) (モデル: Virtual HDD [0])

hda2 10236 MB	hda3 10236 MB	hda5 2047	hda6 9946 MB
------------------	------------------	--------------	-----------------

新規(W) 編集(E) 削除(D) リセット(S) RAID(A) LVM(L)

デバイス	マウントポイント/ RAID/ボリューム	タイプ	フォーマットする	容量 (MB)	開始	終了
/dev/hda2	/opt	ext3	✓	10236	39	1343
/dev/hda3	/	ext3	✓	10236	1344	2648
▼ /dev/hda4		拡張領域		11993	2649	4177
/dev/hda5		swap	✓	2047	2649	2909
/dev/hda6	/home	ext3	✓	9946	2910	4177

RAID デバイス/LVM ボリュームグループメンバーを非表示にする(G)

最終的な形の例

※ 「マウントポイント」や「容量」は
適宜状況に合わせて変更する事。

実演 (HDD設定 3)

GRUB ブートローダは、/dev/hda 上にインストールされます。

ブートローダはインストールされません。

他のオペレーティングシステムがブートできるようにブートローダを設定できます。これにより、一覧からブートするオペレーティングシステムを選択できるようになります。自動的に認識されない他のオペレーティングシステムを追加するには、「追加」をクリックします。デフォルトでブートするオペレーティングシステムを変更するには、目的のオペレーティングシステムで「デフォルト」を選択します。

デフォルト	ラベル	デバイス	
<input checked="" type="checkbox"/>	CentOS	/dev/hda3	<input type="button" value="追加(A)"/> <input type="button" value="編集(E)"/> <input type="button" value="削除(D)"/>

ブートローダパスワードによってカーネルに渡されるオプションをユーザが変更してしまうのを防ぎます。より高度なシステムセキュリティを確保するために、パスワードを設定することを推奨します。

ブートローダパスワードを使用(U)

高度なブートローダオプションの設定(O)

- ブートローダのインストールを確認
- 画面右下の「OK」をクリック

実演 (ネットワーク設定0)

ネットワークデバイス

起動時にアクティブ	デバイス	IPv4/ネットマスク	IPv6/プレフィックス	
<input checked="" type="checkbox"/>	eth0	DHCP	Auto	編集(E)

ホスト名

ホスト名を設定:

DHCP経由で自動設定(A)

手動設定(M) (例、 host.domain.com)

その他の設定

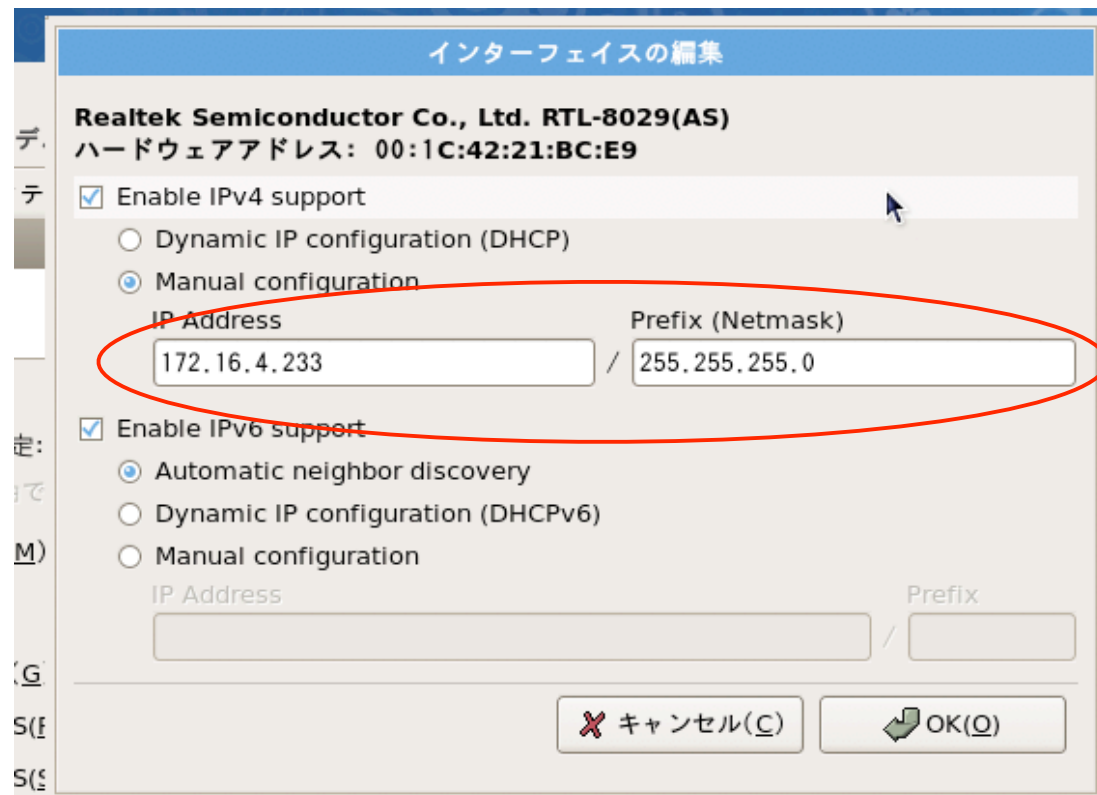
ゲートウェイ(G):

1 番目の DNS(P):

2 番目の DNS(S):

- 「編集」をクリック

実演（ネットワーク設定1）



- “IP Address” と “Netmask” に適切な値を設定

※IPは、実習では 172.16.4.234～241 の範囲になる

実演（ネットワーク設定2）

ネットワークデバイス

起動時にアクティブ	デバイス	IPv4/ネットマスク	IPv6/プレフィックス	編集(E)
<input checked="" type="checkbox"/>	eth0	172.16.4.233/24	Auto	

ホスト名

ホスト名を設定:

DHCP経由で自動設定(A)

手動設定(M) (例、 host.domain.com)

その他の設定

ゲートウェイ(G):

1 番目の DNS(P):

2 番目の DNS(S):

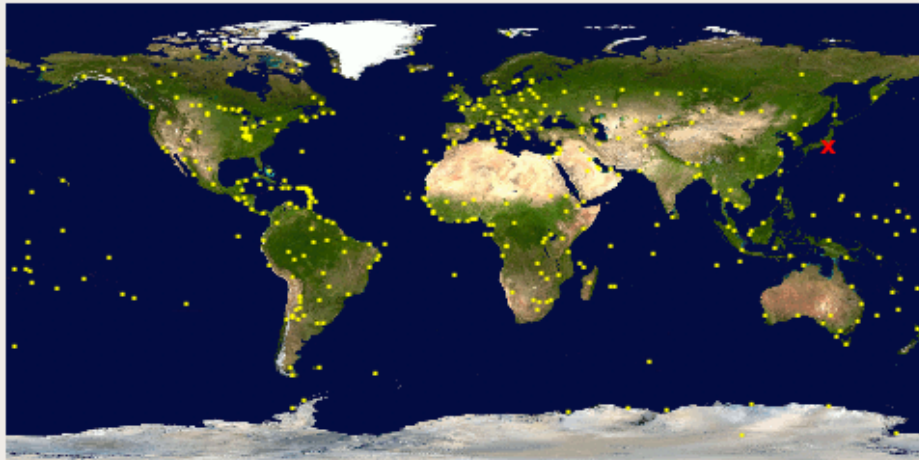
- 「ホスト名」
「ゲートウェイ」
「DNS」に適切な
値を設定

※ホスト名は“kobegp1,...,kobegp8”を利用

※ゲートウェイとDNSは共通

実演 (タイムゾーン設定)

マップ内で地域を選んでクリックしてください:

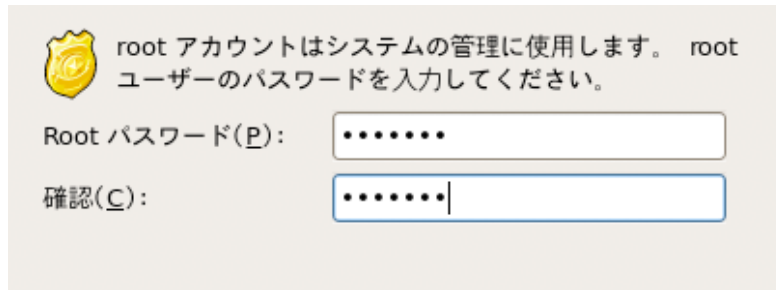


アジア/東京

システムクロックで UTC を使用 (S)

- 「アジア/東京」を選択
- UTC使用のチェックを外す

実演 (root パスワード設定)



root アカウントはシステムの管理に使用します。 root ユーザーのパスワードを入力してください。

Root パスワード(P):

確認(C):

- root のパスワードを入力

※パスワード一般の注意事項を守る事。

(「数字や記号を混ぜる」「口外しない」等)

今回は、各グループ毎に相談して決定。

実演（パッケージの選択1）

CentOS のデフォルトインストールには全般的なインターネット使用に適用できるソフトウェアのセットが含まれています。システムをサポートさせる追加タスクを指定してください。

- Desktop - Gnome
- Desktop - KDE
- Server
- Server - GUI

ソフトウェアのインストールに使用する予定の追加リポジトリを選択してください。

- Packages from CentOS Extras

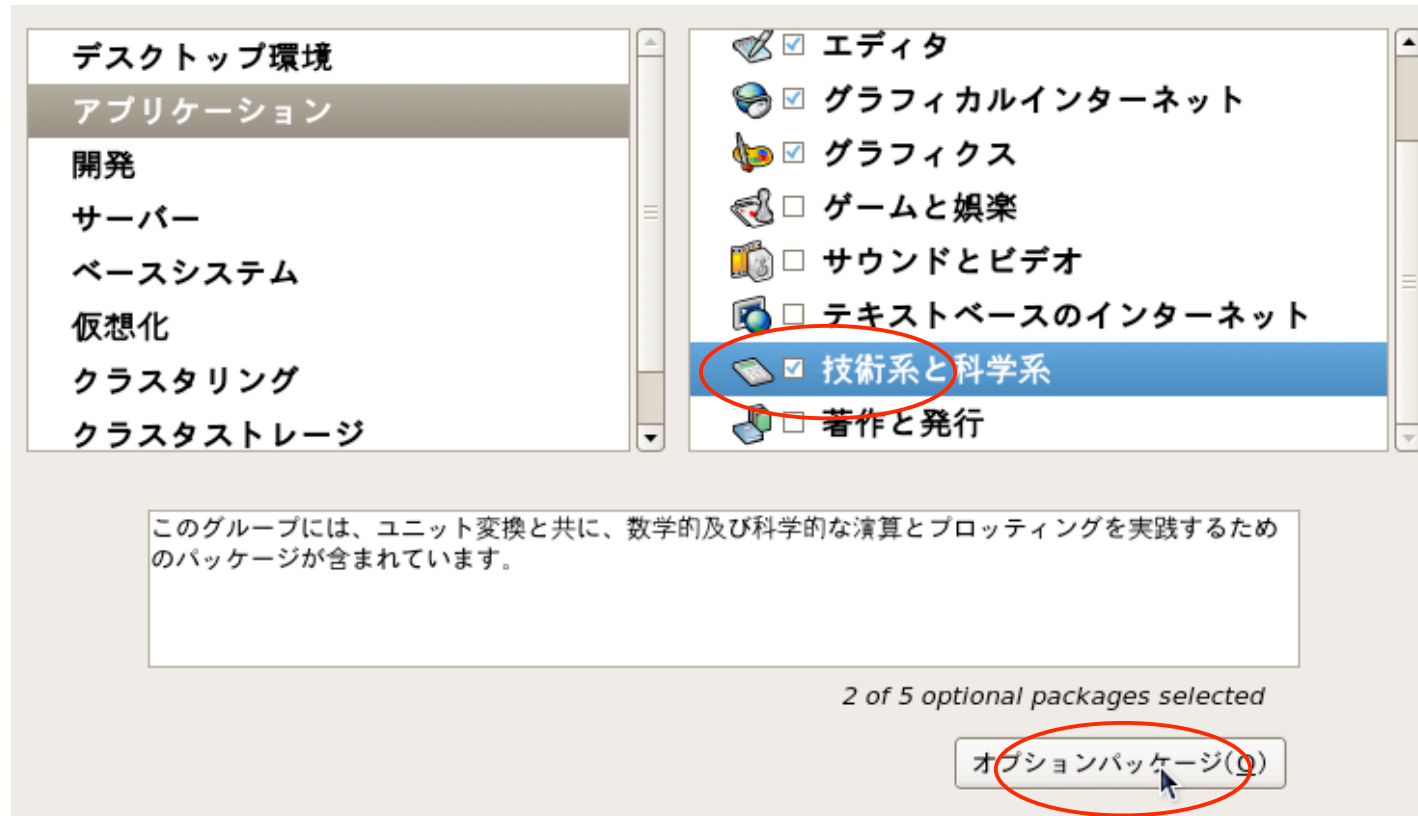
+ 追加でソフトウェアリポジトリを加える (A)

ここで、ソフトウェア選択をさらに詳細にカスタマイズすることができます。また、インストール完了後にソフトウェア管理アプリケーションから行うことも可能です。

- 後でカスタマイズする (L)
- 今すぐカスタマイズする (C)**

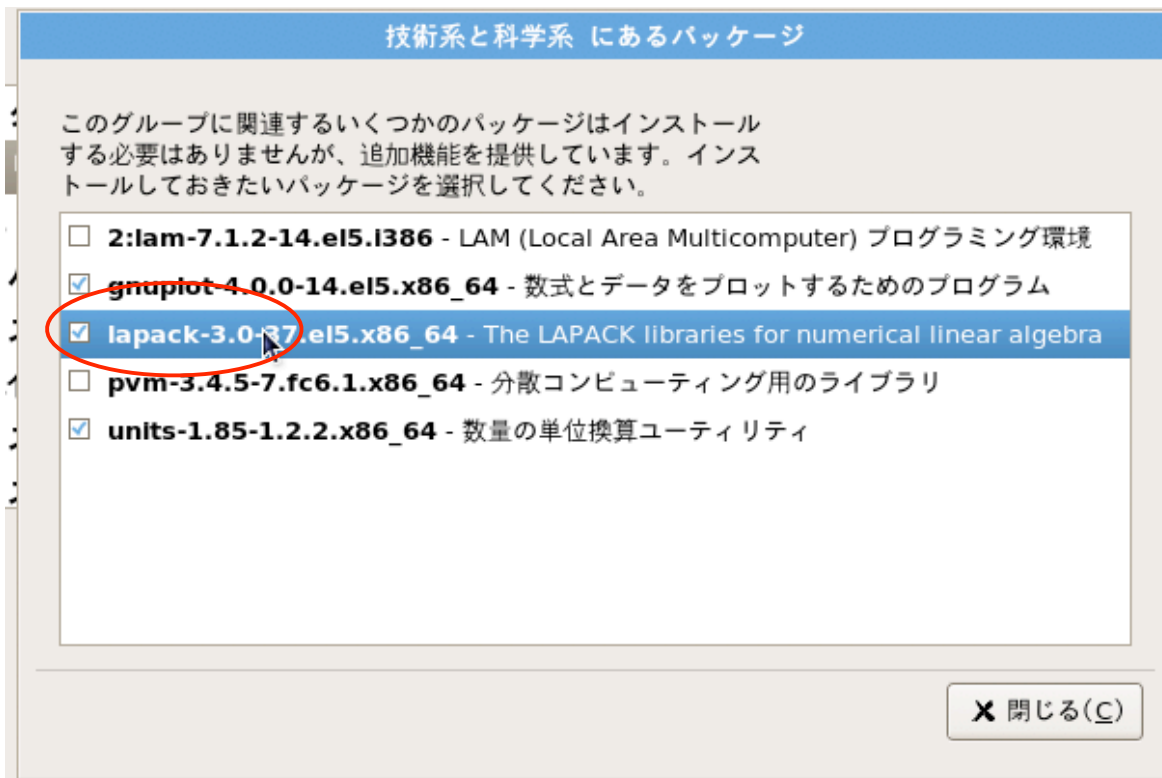
- 「今すぐカスタマイズ」をチェック

実演（パッケージの選択2）



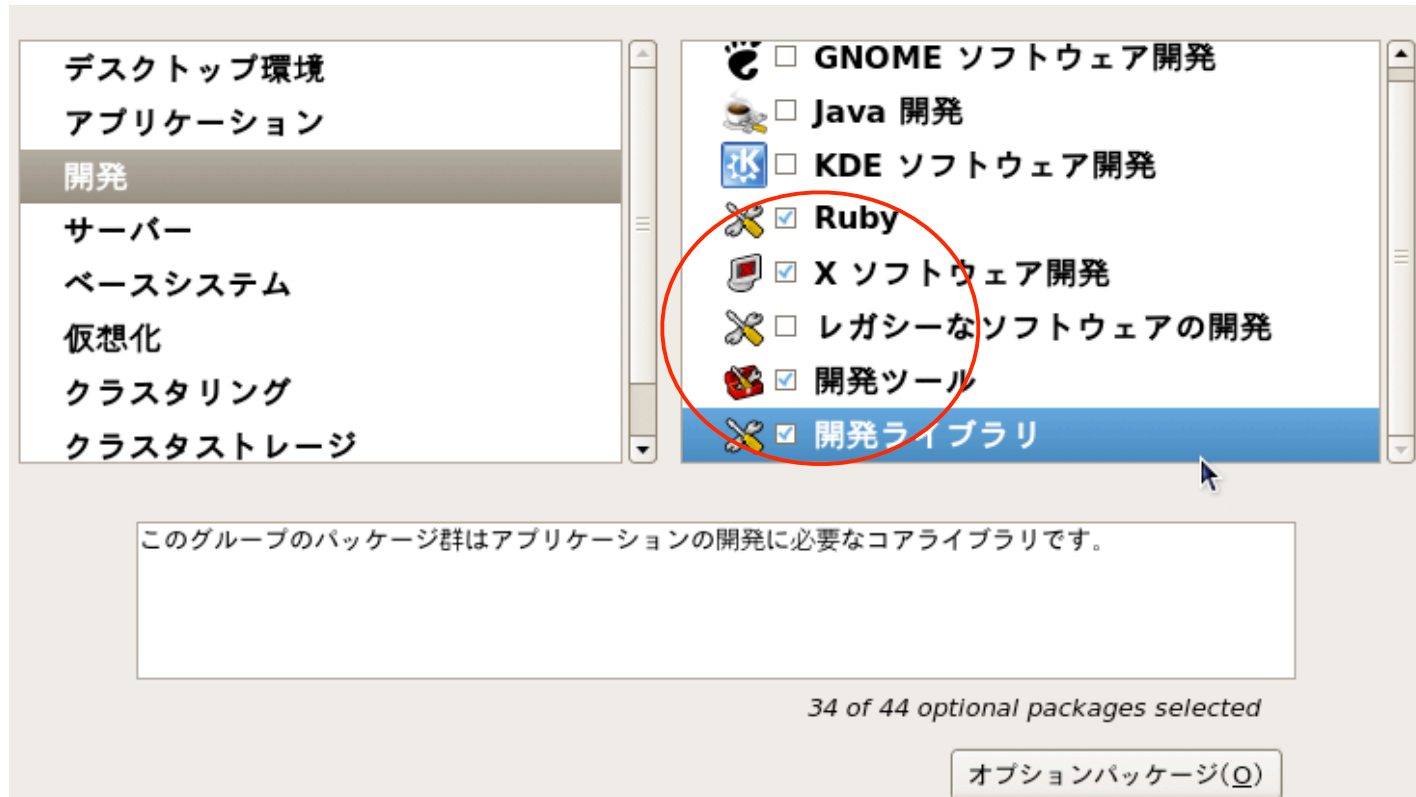
- 左列「アプリケーション」を選択して、右列の項目を取捨選択する。「技術系と科学系」ではオプションパッケージをクリックする。

実演 (パッケージの選択3)



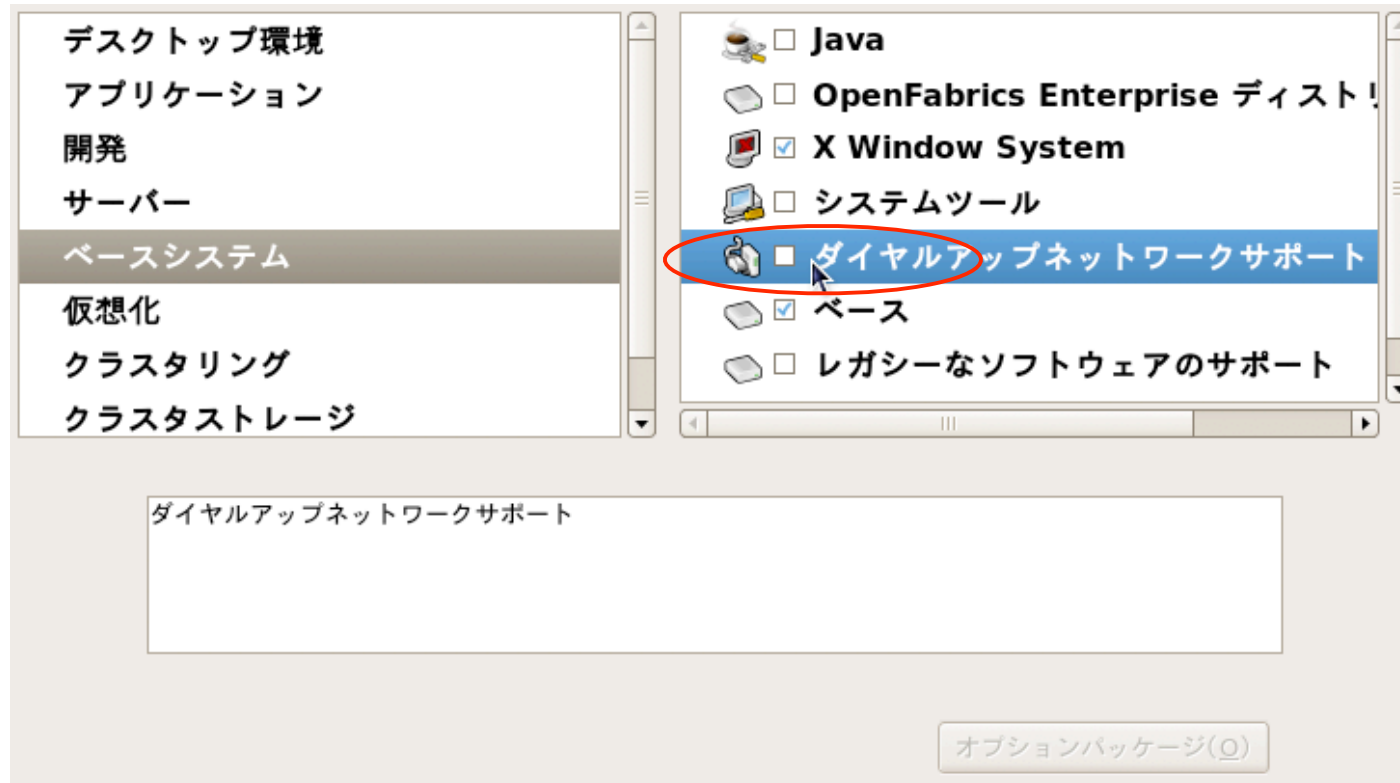
- “lapack” を
チェックする

実演（パッケージの選択4）



- 「Ruby」「Xソフトウェア開発」「開発ツール」「開発ライブラリ」をチェックする。

実演（パッケージの選択5）



- 「ダイヤルアップネットワークサポート」の
チェックを外す。

実演（インストール開始）



- 場合によっては40分ほど掛かることがある。

実習

CentOS5.3/x86_64 の インストール

インストール後の設定 1

1. ファイアウォールの設定

ssh, NFS4, http, https

2. SELinux の設定 利用しない

3. 日付と時刻の設定 ntp を利用

4. ユーザーの作成

ここは、インストールした環境で最初に起動する時に行われる設定である。

実演

インストール後の設定 1

実演（ファイアウォール設定）



- “NFS4”,
“SSH”,
“HTTPS”,
“HTTP” に
チェック

実演 (SELinux設定)



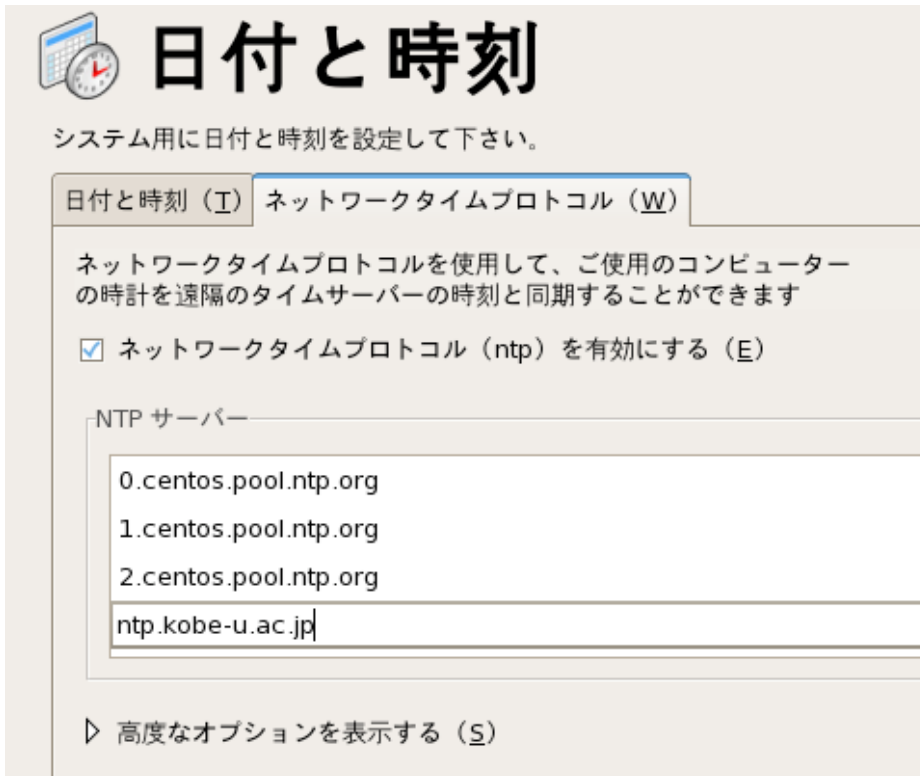
SELinux

セキュリティ拡張 Linux (SELinux) は伝統的な Linux のシステムで使用できる以上の finger 許可されたセキュリティ制御を提供します。無効の状態、拒否しているものに対する警告のみの状態、そして完全にアクティブな状態に設定することもできます。ほとんどの人はデフォルトの設定にしておくべきです。

SELinux 設定:

- 「無効」を選択

実演（日付と時刻の設定）



- 「ネットワークタイムプロトコルを有効にする」にチェック
- 「追加」ボタンをクリックして ntp server を指定

※ntp server は出来るだけ近くのものを選ぶ事

実演（ユーザーの作成）



ユーザーの作成

システムの通常(管理業務以外)の使用には「ユーザー名」を作成されることをお薦めします。システムの「ユーザー名」を作成するには、以下の要求される情報を準備してください。

ユーザー名(U):

フルネーム(E):

パスワード(P):

パスワードの確認(M):

Kerberos や NIS などのネットワーク認証を使用する必要がある場合は、「ネットワークログインを使用する」ボタンをクリックしてください。

- 作業用のユーザーを作成

※今回は後ほど kobegp-svr 上で作成したユーザーに置き換える。

実習

インストール後の設定 1

インストール後の設定 2

1. システムの更新

yum update を実行

2. 不要なデーモンをOFFにする

3. /etc/hosts の設定

4. NFSの設定

kobegp-svr の /home, /opt を kobegp? がNFSマウント

5. ユーザーの作成

kobegp-svr 上で全参加者のアカウントを作成し、/etc/
{passwd,shadow,group,gshadow} の必要部分を kobegp?
がコピーする

※ユーザー管理はNIS等の利用が一般的ですが今回は割愛しました。

実演

インストール後の設定2

※これ以降の操作は、端末上で行う。

※各ページ毎に「説明」「実演」「実習」を行う。

実演 (システムの更新)

```
[tkoba@kobegp-svr ~]$ su -  
パスワード:  
[root@kobegp-svr ~]# yum update  
.....  
[root@kobegp-svr ~]# yum clean all  
Loaded plugins: fastestmirror  
Cleaning up Everything  
Cleaning up list of fastest mirrors  
[root@kobegp-svr ~]# reboot
```

root になります。
プロンプトが \$ から
に変わります。

- yum を用いて最新の状態にします。

実演 (デーモンの整理)

```
[root@kobegp-svr rc5.d]# chkconfig auditd off
[root@kobegp-svr rc5.d]# chkconfig bluetooth off
[root@kobegp-svr rc5.d]# chkconfig cups off
[root@kobegp-svr rc5.d]# chkconfig sendmail off
[root@kobegp-svr rc5.d]# chkconfig xfs off
[root@kobegp-svr rc5.d]# chkconfig avahi-daemon off
[root@kobegp-svr rc5.d]# chkconfig nfs on
```

NFS server (kobegp-svr) のみ

- **chkconfig** を用いて計算クラスターには必要の無いデーモンをOFFにします。

※gnome のGUIツール (「アプリケーション」 → 「システム設定」 → 「サーバ設定」 → 「サービス」) や、 /usr/sbin/ntsysv などでも、同様な作業が出来ます。

実演 (/etc/hostsの設定)

```
[root@kobegp-svr ~]# cp -p /etc/hosts /etc/hosts.orig  
[root@kobegp-svr ~]# vi /etc/hosts  
[root@kobegp-svr ~]# cat /etc/hosts
```

```
# Do not remove the following line, or various  
programs  
# that require network functionality will fail.  
127.0.0.1      localhost.localdomain localhost  
:::1         localhost6.localdomain6 localhost6  
172.16.4.233   kobegp-svr.kobe-u.ac.jp kobegp-svr  
172.16.4.234   kobegp1.kobe-u.ac.jp kobegp1  
172.16.4.235   kobegp2.kobe-u.ac.jp kobegp2  
172.16.4.236   kobegp3.kobe-u.ac.jp kobegp3  

```

← バックアップ

← vi で編集

“i” で編集モードに “Esc”
でコマンドモードになる。

保存は “Esc” “:w”。

終了は “Esc” “:q”。

保存して終了は “Esc” “ZZ”

← 編集結果

- vi を用いて /etc/hosts を編集し、計算クラスターの名前解決をします。

※DNSが適切に設定されていればこの作業は必要ありません。

実演 (NFSの設定 1)

NFS server (kobegp-svr) での設定

```
[root@kobegp-svr ~]# cp /etc/sysconfig/nfs /etc/sysconfig/nfs.orig
[root@kobegp-svr ~]# vi /etc/sysconfig/nfs
[root@kobegp-svr ~]# cat /etc/sysconfig/nfs
.....
MOUNTD_PORT=2050 ← rpc.mound のポート指定
.....
STATD_PORT=2051 ← rpc.statd のポート指定
.....
LOCKD_TCPSPORT=2052 ← lockd の tcp ポート指定
.....
LOCKD_UDPPORT=2052 ← lockd の udp ポート指定
.....
[root@kobegp-svr ~]#
```

- **/etc/sysconfig/nfs にNFSで利用するポートを明示的に指定**

実演 (NFSの設定2)

NFS server (kobegp-svr) での設定

```
[root@kobegp-svr ~]# cp /etc/sysconfig/iptables /etc/sysconfig/iptables.orig
[root@kobegp-svr ~]# vi /etc/sysconfig/iptables
[root@kobegp-svr ~]# cat /etc/sysconfig/iptables
.....
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -s 172.16.4.0/24 -m tcp -p tcp -m multiport
--dports 111,2049,2050,2051,2052 -j ACCEPT ← ※前の行の続き
-A RH-Firewall-1-INPUT -m state --state NEW -s 172.16.4.0/24 -m udp -p udp -m multiport
--dports 111,2049,2050,2051,2052 -j ACCEPT ← ※前の行の続き
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 20000:21000 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 20000:21000 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
.....
[root@kobegp-svr ~]# service iptables restart
```

- /etc/sysconfig/iptables にNFS (上の2行) と MPI (下の2行) で利用するポートを指定

実演 (NFSの設定3)

NFS server (kobegp-svr) での設定

```
[root@kobegp-svr ~]# cp /etc/exports /etc/exports.orig ← バックアップ
[root@kobegp-svr ~]# vi /etc/exports
[root@kobegp-svr ~]# cat /etc/exports
/home 172.16.4.0/255.255.255.0(rw,async,no_root_squash) ← 編集結果
/opt  172.16.4.0/255.255.255.0(rw,async,no root squash)
[root@kobegp-svr ~]# chkconfig portmap on ← デーモンの自動
[root@kobegp-svr ~]# chkconfig nfslock on ← 起動設定
[root@kobegp-svr ~]# chkconfig nfs on
[root@kobegp-svr ~]# service portmap start
[root@kobegp-svr ~]# service nfslock start
[root@kobegp-svr ~]# service nfs start
```

- /etc/export にNFSの提供内容を記述
- chkconfig でシステム起動時にデーモンが起動するよう設定
- service でデーモンを起動

実演 (NFSの設定4)

NFS client (kobegp?) での設定

```
[root@kobegp1 ~]# cp /etc/fstab /etc/fstab.orig ← バックアップ
[root@kobegp1 ~]# vi /etc/fstab
[root@kobegp1 ~]# cat /etc/fstab
LABEL=/                                /                                ext3      defaults      1 1
LABEL=/boot                            /boot                          ext3      defaults      1 2
tmpfs                                    /dev/shm                       tmpfs     defaults      0 0
devpts                                   /dev/pts                       devpts    gid=5,mode=620 0 0
sysfs                                    /sys                            sysfs     defaults      0 0
proc                                     /proc                          proc      defaults      0 0
LABEL=SWAP-hda5                        swap                            swap      defaults      0 0
kobegp-svr:/home                        /home                          nfs       defaults      0 0
kobegp-svr:/opt                         /opt                           nfs       defaults      0 0
[root@kobegp1 ~]# chkconfig netfs on ← デーモンの自動起動設定
[root@kobegp1 ~]# chkconfig autofs on
[root@kobegp1 ~]# mount /home ← NFS マウント
[root@kobegp1 ~]# mount /opt
```

編集箇所

- /etc/fstab にNFSのマウントポイントを記述
- mount コマンドでマウントする

実演（ユーザー作成1）

NFS server (kobegp-svr) での設定

```
[root@kobegp-svr ~]# useradd [ユーザー名]  
[root@kobegp-svr ~]# passwd [ユーザー名]
```

- ユーザーアカウントの作成とパスワードを設定
- これを人数分繰り返す

実演（ユーザー作成2）

NFS client (kobegp?) での設定

```
[tkoba@kobegp1 ~]$ ssh kobegp-svr ← サーバにログイン
[tkoba@kobegp-svr ~]$ su -
[root@kobegp-svr ~]# cat /etc/passwd
.....
aruuser:x:500:500:DAREKA San:/home/aruuser:/bin/bash
tuginouser:x:501:501:DOREKA San:/home/tuginouser:/bin/bash
.....
```

↑ この部分を kobegp? の /etc/passwd にコピー

- サーバの /etc/passwd のUID(3カラム)が500以上の行を全てを、クライアントの /etc/passwd にコピーする。
- これを /etc/{shadow,group,gshadow} で繰り返す。

MPIのインストール

MPIのインストール

- MPIインストール編
 - 下準備：ソースを取得
 - インストール：configure make を実行
 - 設定：環境変数などの設定

※今回はサーバの/optにインストールしたものをNFSマウントして利用するので、「下準備」と「インストール」はサーバのみで行う。

下準備

```
[tkoba@kobegp-svr ~]$ mkdir Work ← 作業用ディレクトリを作成し移動
[tkoba@kobegp-svr ~]$ cd Work/
[tkoba@kobegp-svr Work]$ lftpget http://www.mcs.anl.gov/
research/projects/mpich2/downloads/tarballs/1.1/
mpich2-1.1.tar.gz
[tkoba@kobegp-svr Work]$ ls ← mpich2 を取得 (全て1行で)
mpich2-1.1.tar.gz
[tkoba@kobegp-svr Work]$ tar zxf mpich2-1.1.tar.gz
[tkoba@kobegp-svr Work]$ ls ← mpich2 の tar を展開
mpich2-1.1 mpich2-1.1.tar.gz
[tkoba@kobegp-svr Work]$
```

- 作業用ディレクトリを作成
- lftp を利用してmpich2のソースコードを取得
- tar で展開

インストール

```
[tkoba@kobegp-svr ~]$ cd Work/
[tkoba@kobegp-svr Work]$ ls
mpich2-1.1  mpich2-1.1.tar.gz
[tkoba@kobegp-svr Work]$ cd mpich2-1.1
[tkoba@kobegp-svr mpich2-1.1]$ ./configure --prefix=/opt/mpich2
.....
[tkoba@kobegp-svr mpich2-1.1]$ make
.....
[tkoba@kobegp-svr mpich2-1.1]$ su
パスワード:
[root@kobegp-svr mpich2-1.1]# make install
.....
[root@kobegp-svr mpich2-1.1]#
```

展開した mpich2 のトップ
に移動
configure
make
root で make install

- 展開した mpich2 をユーザーアカウントで configure; make
- /opt/mpich2 にインストールする為に root で make install を実行

設定 1 (ssh nopasswd)

```
[tkoba@kobegp-svr ~]$ ssh-keygen ← ssh2 rsa 鍵を作成
Generating public/private rsa key pair.
Enter file in which to save the key (/home/tkoba/.ssh/id_rsa):
Created directory '/home/tkoba/.ssh'.
Enter passphrase (empty for no passphrase): ← nopasswd にする為に
Enter same passphrase again: ← 空にする
Your identification has been saved in /home/tkoba/.ssh/id_rsa.
Your public key has been saved in /home/tkoba/.ssh/id_rsa.pub.
The key fingerprint is:
12:80:5b:09:80:ab:xx:08:xx:47:xx:29:7a:08:12:38 tkoba@kobegp-svr.kobe-u.ac.jp
[tkoba@kobegp-svr ~]$ cd .ssh
[tkoba@kobegp-svr .ssh]$ cat id_rsa.pub >> authorized_keys
[tkoba@kobegp-svr .ssh]$ chmod 644 authorized_keys ← nopasswd login する
                                                    為に公開鍵を登録
```

- ssh2 rsa 鍵を作成し、相互にパスワード無しでssh login 出来る様にする。

※今回はhomeがNFS共有されているので、鍵(id_rsa*)の移動や authorized_keys の編集も一カ所で良い。

設定 2 (環境変数の設定)

```
[tkoba@kobegp-svr ~]$ vi .bashrc
[tkoba@kobegp-svr ~]$ cat .bashrc
.....
# User specific aliases and functions
export PATH=/opt/mpich2/bin:$PATH
export MPICH_PORT_RANGE=20000:21000
[tkoba@kobegp-svr ~]$ source .bashrc
[tkoba@kobegp-svr ~]$ which mpirun
/opt/mpich2/bin/mpirun
```

編集箇所

設定の読み込み

- インストールした /opt/mipch2/bin にパスを通す
- MPI で利用するポートレンジを指定

※今回はhomeがNFS共有されているので、サーバ上の作業だけで良い。

設定3 (mpich2の設定)

編集箇所

```
[tkoba@kobegp-svr ~]$ vi .mpd.conf
[tkoba@kobegp-svr ~]$ cat .mpd.conf
secretword=hogehoge
[tkoba@kobegp-svr ~]$ chmod 600 .mpd.conf
[tkoba@kobegp-svr ~]$ vi mpd.hosts
[tkoba@kobegp-svr ~]$ cat mpd.hosts
kobegp-svr.kobe-u.ac.jp
kobegp1.kobe-u.ac.jp
kobegp2.kobe-u.ac.jp
kobegp3.kobe-u.ac.jp
kobegp4.kobe-u.ac.jp
kobegp5.kobe-u.ac.jp
kobegp6.kobe-u.ac.jp
kobegp7.kobe-u.ac.jp
kobegp8.kobe-u.ac.jp
```

← 編集箇所

- 各自の home に .mpd.conf を作成し、自分専用の暗号を記述する。
- mpdboot 用に、mpd.hosts を作成する。

※今回はhomeがNFS共有されているので、サーバ上の作業だけで良い。

設定 4 (firewall)

NFS client (kobegp?) での設定

```
[root@kobegp-svr ~]# cp /etc/sysconfig/iptables /etc/sysconfig/iptables.orig
[root@kobegp-svr ~]# vi /etc/sysconfig/iptables
[root@kobegp-svr ~]# cat /etc/sysconfig/iptables
.....
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 20000:21000 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 20000:21000 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
.....
[root@kobegp-svr ~]# service iptables restart
```

- /etc/syscongi/iptable に MPI で利用するポートレンジを記述

テスト

NFS server (kobegp-svr) で実行

```
[tkoba@kobegp-svr ~]$ mpdboot -n 9 -f mpd.hosts  
[tkoba@kobegp-svr ~]$ mpiexec -n 18 hostname
```

mpd を起動

テスト

- サーバ上で mpd を起動する。
- mpi job は、サーバ上の各自のアカウントで実行する。

テスト2

NFS server (kobegp-svr) で実行

```
[tkoba@kobegp-svr ~]$ cp /tmp/codes.tar .
[tkoba@kobegp-svr ~]$ tar xf codes.tar ← mpi test src を展開
[tkoba@kobegp-svr ~]$ cd codes/sieve1
[tkoba@kobegp-svr ~]$ mpicc -O3 sieve1.c -o sieve1 -lm ← コンパイル
.....
[tkoba@kobegp-svr ~]$ mpirun -np 18 ./sieve1 2048 ← 実行
.....
[tkoba@kobegp-svr ~]$ mpdallexit ← mpd を終了
```

- サーバ上の /tmp にあるテストコードを展開
- mpicc でコンパイルし実行
- mpd を終了



お疲れさまでした

連絡先：小林泰三

tkoba@cc.kyushu-u.ac.jp



付録

NISの設定 1

NFS server (kobegp-svr) での設定

```
[root@kobegp-svr ~]# yum install ypserv ypbind yp-tools
[root@kobegp-svr ~]# ypdomainname [nis domain name]
[root@kobegp-svr ~]# /sbin/service ypserv start
[root@kobegp-svr ~]# /usr/lib64/yp/ypinit -m
[root@kobegp-svr ~]# /sbin/service yppasswdd start
[root@kobegp-svr ~]# vi /etc/sysconfig/network
[root@kobegp-svr ~]# cat /etc/sysconfig/network
NETWORKING=yes
NETWORKING_IPV6=yes
HOSTNAME=kobegp-srv.kobe-u.ac.jp
START_YPSERV=yes
START_YPPASSWDD=yes
NISDOMAIN=[nis domain name]
[root@kobegp-svr ~]# cd /var/yp; make
```

必要なパッケージをインストール

NISドメイン名を設定

← 編集箇所

- ypserv, ypbind, yp-tools パッケージをインストール
- NISドメイン名を設定してデータベースを作成し、サービスの設定をする

NISの設定 2

NFS client (kobegp?) での設定

```
[root@kobegp-svr ~]# yum install ypbind yp-tools
[root@kobegp-svr ~]# vi /etc/sysconfig/network
[root@kobegp-svr ~]# cat /etc/sysconfig/network
NETWORKING=yes
NETWORKING_IPV6=yes
HOSTNAME=kobegp1.kobe-u.ac.jp
START_YPPASSWDD=yes
NISDOMAIN=[nis domain name]
[root@kobegp-svr ~]# vi /etc/yp.conf
[root@kobegp-svr ~]# cat /etc/yp.conf
.....
domain [nis domain name] server kobegp-svr
[root@kobegp-svr ~]# vi /etc/passwd
[root@kobegp-svr ~]# cat /etc/sysconfig/network
.....
+:::::::
[root@kobegp-svr ~]# /sbin/service ypbind start
```

← 必要なパッケージをインストール

← 編集箇所

← 編集箇所

← 編集箇所

- ypbind, yp-tools パッケージをインストール
- NISドメイン名を設定し、サービスの設定をする