お試しアカウント付き 並列プログラミング講習会 「Reedbush利用の手引き」

東京大学情報基盤センター

ノートパソコンの設定： 公開鍵の生成、登録

※ネットワーク環境に接続してから行ってください

鍵の作成

1. ターミナルを起動する
2. 以下を入力する

```
$ ssh-keygen  -t  rsa
```

3. 鍵ファイルの保存先を聞かれるので、リターンを押す
4. 鍵を使うためのパスワードを聞かれるので、 センターのパスワードではない、自分の好きな パスワードを入れる（パスフレーズとよぶ）
5. もう一度、上記のパスフレーズを入れる
6. 鍵が生成される

鍵の利用（1/2）

1. 生成した鍵は、以下に入っている

```
.ssh/
```

2. 以下を入力する

```
$ cd  .ssh/
```

3. 以下を入力すると、ファイルが見える

```
$ ls
id_rsa  id_rsa.pub  known_hosts
```

4. ここで、以下のファイルを区別する

```
id_rsa      : 秘密鍵
id_rsa.pub  : 公開鍵
```

この公開鍵の収納ディレクトリを覚えておく（後で使います）

Reedbushへの公開鍵の登録

- Webブラウザで登録用ページにアクセスする

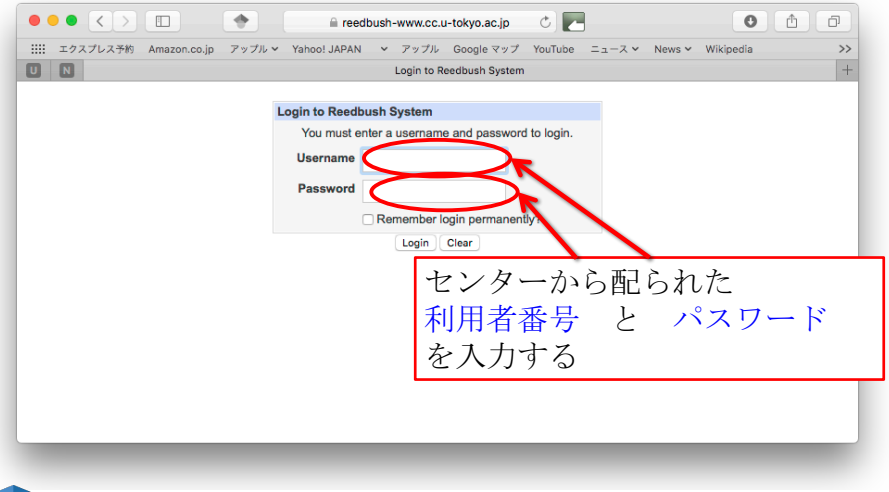
  https://reedbush-www.cc.u-tokyo.ac.jp/

- ユーザ名とパスワードを聞かれるので、
  センターから発行されたユーザ名とパスワードを入れる。
- 注意：記載されたパスワードそのままではNG！

センター発行のパスワードの意味

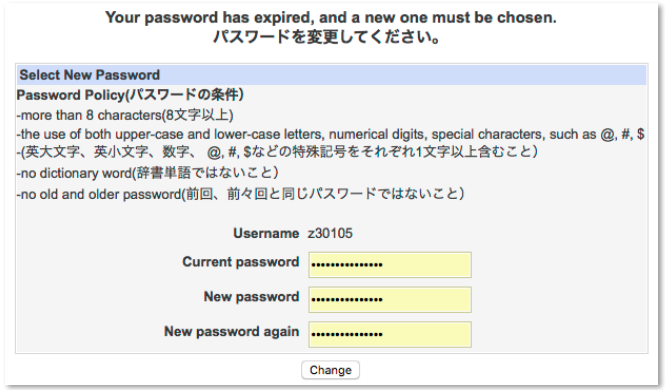
- （配付資料には未掲載）

ポータル画面（ログイン前）



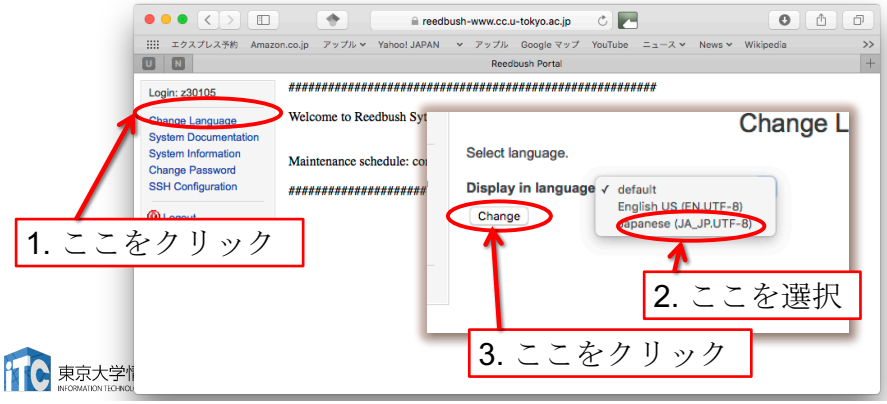
パスワード変更

- 最初のログイン時にパスワード変更を求められるので、
  新しいパスワードを入力してください。



言語の変更

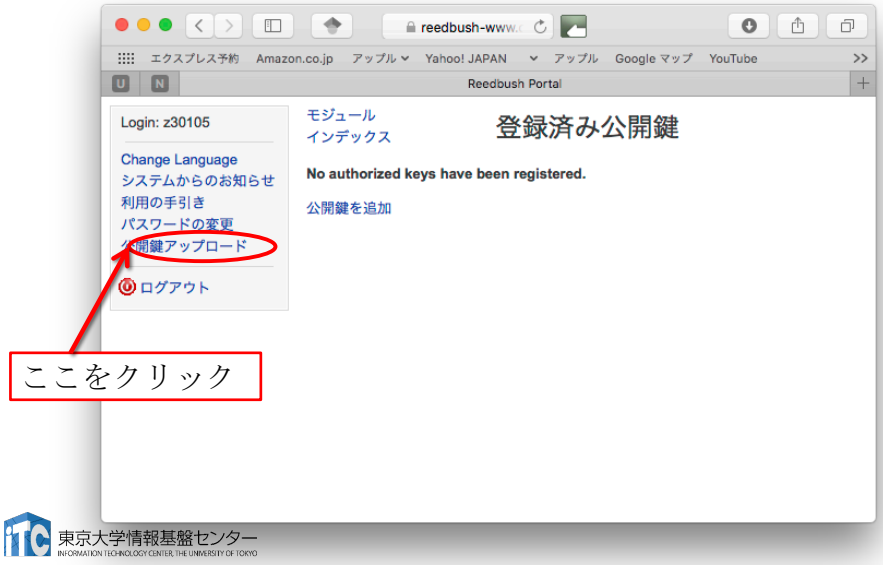
- “Change Language” で日本語に変更できます。
  - “Japanese (JA_JP.UTF-8)” を選んで “Change”を押す
  - 終わったら、ブラウザで再読み込み



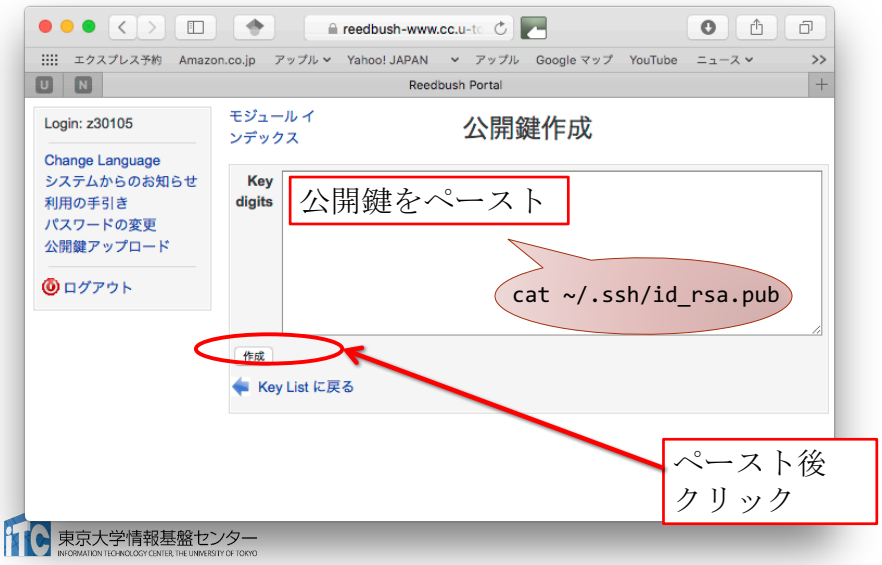
鍵の登録

1. 左側メニューの「公開鍵アップロード」をクリックする
2. 「公開鍵を追加」をクリックし、画面に公開鍵をコピーアンドペーストする
3. 「作成」ボタンを押す

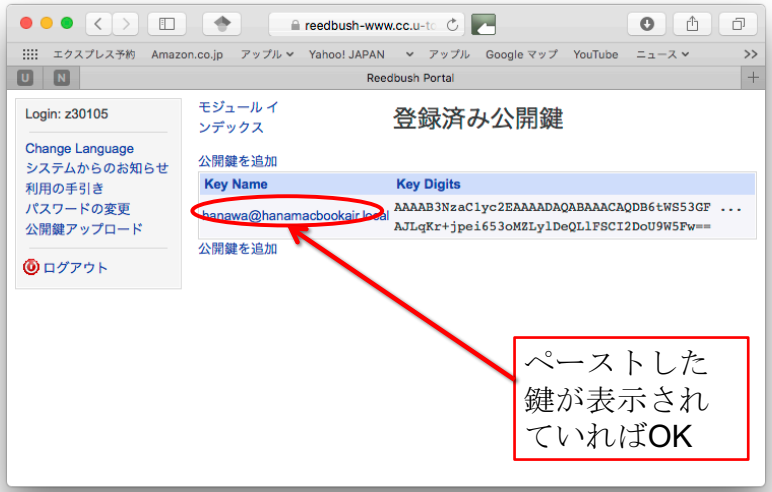
ポータル画面（言語変更後）



ポータル画面（公開鍵登録）



ポータル画面（公開鍵登録成功）



スパコンへのログイン

Reedbushへログイン

- ターミナルから、以下を入力する
  `$ ssh -Y reedbush.cc.u-tokyo.ac.jp -l txxxxx`
  「-l」はハイフンと小文字のＬ、「-Y」はハイフンと大文字のY
  「txxxxx」は利用者番号
- 接続するかと尋ねられるので、　yes　を入力する
- 鍵の設定時に入れた
  自分が決めたパスワード（パスフレーズ）
  を入力する
- 成功すると、ログインができる

ReedbushのデータをPCに取り込む

- ターミナルでscpコマンドを使う
- `$ scp txxxxx@reedbush.cc.u-tokyo.ac.jp:~/a.f90 ./`
  「txxxxx」は利用者番号
  - Reedbush上のホームディレクトリにある a.f90 をPCのカレントディレクトリに取ってくる
  - ディレクトリごと取ってくる場合は -r を指定する
- `$ scp -r txxxxx@reedbush.cc.u-tokyo.ac.jp:~/SAMP ./`
  - Reedbush上のホームディレクトリにあるSAMPフォルダを、その中身ごと、PCのカレントディレクトリに取ってくる

PCのファイルをReedbushに置く

- 同様にターミナルでscpコマンドを使う

- `$ scp ./a.f90 txxxxx@reedbush.cc.u-tokyo.ac.jp:`
  「txxxxx」は利用者番号
  - PCのカレントディレクトリにある a.f90 を、Reedbush上のホームディレクトリに置く
  - ディレクトリごと置くには、 -r を指定する
- `$ scp -r ./SAMP txxxxx@reedbush.cc.u-tokyo.ac.jp:`
  - PCのカレントディレクトリにあるSAMPフォルダを、その中身ごと、Reedbush上のホームディレクトリに置く

EmacsのTramp機能によるファイル操作（必要な人のみ）

- emacs が自分のパソコンに入っている人は、Tramp機能による遠隔ファイルの操作も可能
- Reedbushの秘密鍵をSSHに登録する
- emacs を起動
- ファイル検索モードにする
  ^x ^f （ ^ はcontrol）
- “Find file: ”の現在のパス名部分を消し、以下を入力する（txxxxxは自分のログインIDにする）
  `Find file:/ssh:txxxxx@reedbush.cc.u-tokyo.ac.jp:`
- パスフレーズを入れると、ローカルファイルのようにReedbush上のファイルが編集できる

GUIによるファイル操作（主にWindowsユーザ向け）

- FileZillaやWinSCPを使えば手元のパソコンとReedbush間のファイル転送をGUI操作で行うことができる
- FilzeZilla
  - `https://filezilla-project.org`
  - “Download Filezilla Client”からダウンロード
  - サイトマネージャにてプロトコルをSFTPに設定、ログオンの種類を鍵ファイルにする（Putty形式の公開鍵ファイルが必要、puttygenによって変換すると良い）
- WinSCP
  - `https://winscp.net/eng/download.php`
  - プロトコルをSFTPまたはSCPに設定する
  - ホスト設定画面の設定からSSH-認証を選び、秘密鍵を指定する（OpenSSH形式・Putty形式の両方に対応）

Reedbushにおける注意

- /home ファイルシステムは容量が小さく、ログインに必要なファイルだけを置くための場所です。
  - /home に置いたファイルは計算ノードから参照できません。ジョブの実行もできません。
- 転送が終わったら、/lustre ファイルシステムに移動(mv)してください。
- または、直接 /lustre ファイルシステムを指定して転送してください。

- ホームディレクトリ: /home/gt00/txxxxx
  - cd コマンドで移動できます。
- Lustreディレクトリ: /lustre/gt00/txxxxx
  - cdw コマンドで移動できます。

UNIX備忘録 (1/3)

- emacsの起動： emacs 編集ファイル名
  - ^x ^s（^はcontrol）：テキストの保存
  - ^x ^c ：終了
    （^z で終了しないことするとスパコンの負荷が上がるため絶対にしないこと）
  - ^g ：作業の取消（訳がわからなくなったときにも）
  - ^k ：カーソルより行末まで消す、消した行は一時的に記憶される
  - ^y ：^kで消した行を、現在のカーソルの場所にコピーする
  - ^s 文字列：文字列の箇所まで移動する（検索機能）
  - ^M x goto-line：指定した行まで移動する

UNIX備忘録 (2/3)

- rm ファイル名： ファイル名のファイルを消す
  - rm *~： test.c~ などの、~がついたバックアップファイルを消す。
    ※使う時は慎重に。*~ の間に空白が入ってしまうと、全て消えます。
- ls： 現在いるフォルダの中身を見る
- cd フォルダ名：フォルダに移動する
  - cd ..：一つ上のフォルダに移動する
  - cd ~：ホームディレクトリに移動する
- cat ファイル名： ファイルの中身を表示する
- make：実行ファイルを作る
  （Makefileに適切な記述が必要）
  - make clean：実行ファイルを消す
    （clean がMakefileで定義されている必要がある）

UNIX備忘録 (3/3)

- less ファイル名： ファイル名の中身を見る（スクロール操作が可能なため、1画面では収まらない場合に便利）
  - スペースキー：1画面スクロール
  - / ：文字列の箇所まで移動
  - q ：終了

スパコン上でのプログラムの実行

「ジョブ」の実行形態と実行方法について

Reedbush-Hスーパーコンピュータシステムでのジョブ実行形態

- 以下の２通りがあります
- インタラクティブジョブ実行
  - ＰＣでの実行のように、コマンドを入力して実行する方法
  - スパコン環境では、あまり一般的でない
  - デバック用、大規模実行はできない
  - Reedbush-Hでは、以下に限定
    - １ノード（36コア, 2GPU）：15分まで
    - 2ノード（72コア, 4GPU）：5分まで

※講習会アカウントではバッチジョブ実行のみ、最大2ノード10分まで

- バッチジョブ実行
  - バッチジョブシステムに処理を依頼して実行する方法
  - 実行させたい処理をファイル（ジョブスクリプト）で指示する
  - スパコン環境で一般的
  - 大規模実行用
  - Reedbush-Uでは、最大128ノード（4,608コア）(24時間)

インタラクティブ実行のやり方

参考情報（講習会アカウントでは使えません）

- コマンドラインで以下を入力
  - 1ノード実行用
  - `$ qsub -I -q u-interactive -l select=1 -l walltime=01:00 -W group_list=gt00`

※コマンドは改行せず1行で入力すること

  - 4ノード実行用
  - `$ qsub -I -q u-interactive -l select=4 -l walltime=01:00 -W group_list=gt00`

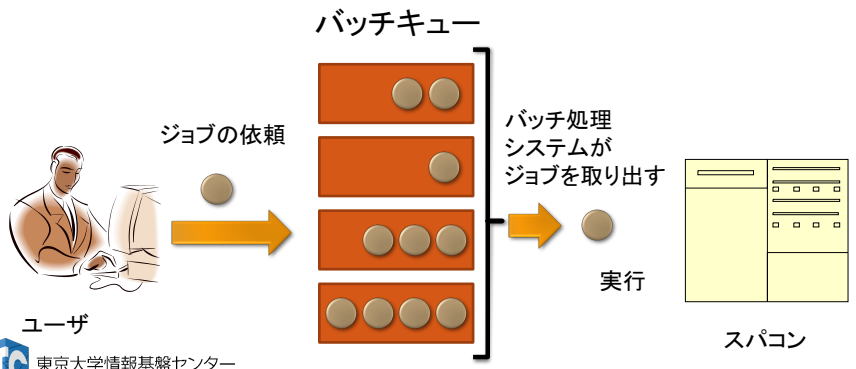
※インタラクティブ用のノードがすべて使われている場合、資源が空くまで、ログインできません。

コンパイラの種類とインタラクティブ実行およびバッチ実行

- Reedbushでは、コンパイラはバッチ実行、インタラクティブ実行で共通に使えます。
- 例） Intelコンパイラ
  - Cコンパイラ: icc, mpiicc (Intel MPIを使う場合)
  - Fortran90コンパイラ: ifort, mpiifort (Intel MPIを使う場合)

バッチ処理とは

- スパコン環境では、通常は、インタラクティブ実行（コマンドラインで実行すること）はできません。
- ジョブはバッチ処理で実行します。



バッチキューの設定のしかた

- Reedbushでのバッチ処理は、Altair社のバッチシステム PBS Professionalで管理されています。
- 以下、主要コマンドを説明します。
  - ジョブの投入：qsub <ジョブスクリプトファイル名>
  - 自分が投入したジョブの状況確認：rbstat
  - 投入ジョブの削除：qdel <ジョブID>
  - バッチキューの状態を見る：rbstat --rsc
  - バッチキューの詳細構成を見る：rbstat --rsc -x
  - 投げられているジョブ数を見る：rbstat -b
  - 過去の投入履歴を見る：rbstat -H
  - 同時に投入できる数／実行できる数を見る：rbstat --limit

ジョブスクリプトの例

※実行させたい処理によって各項目の内容は異なります

```
#!/bin/bash
#PBS -q u-lecture
#PBS -Wgroup_list=gt00
#PBS -l select=1
#PBS -l walltime=00:01:00
cd $PBS_O_WORKDIR
. /etc/profile.d/modules.sh

./a.out
```

リソースグループ名 :u-lecture

利用グループ名 :gt00

利用ノード数

複数スレッドや複数ノードを使う場合には対応する指定が必要

実行時間制限 :1分

プログラムを実行

カレントディレクトリ設定、環境変数設定（必ず入れておく）

本お試し講習会でのキュー・グループ名

- 本演習中のキュー名：
  - h-tutorial
  - 最大10分まで
  - 最大ノード数は2ノード(4GPU) まで
- 本演習時間以外（24時間）のキュー名：
  - h-lecture
  - 利用条件は演習中のキュー名と同様
- グループ名：gt00

rbstat --rsc の実行画面例

```
$ rbstat --rsc
QUEUE                  STATUS           NODE
u-debug                [ENABLE ,START]    54
u-short                [ENABLE ,START]    16
u-regular              [ENABLE ,START]
  |---- u-small        [ENABLE ,START]   288
  |---- u-medium       [ENABLE ,START]   288
  |---- u-large        [ENABLE ,START]   288
  |---- u-x-large      [ENABLE ,START]   288
u-interactive          [ENABLE ,START]
  |---- u-interactive_1 [ENABLE ,START]    54
  |---- u-interactive_4 [ENABLE ,START]    54
u-lecture              [ENABLE ,START]    54
u-lecture8             [DISABLE,START]    54
u-tutorial             [ENABLE ,START]    54
```

使える キュー名 (リソース グループ)

現在 使えるか

ノードの 利用可能数

rbstat --rsc -x の実行画面例

```
$ rbstat --rsc -x
QUEUE                STATUS          MIN_NODE MAX_NODE MAX_ELAPSE REMAIN_ELAPSE MEM(GB)/NODE  PROJECT
u-debug              [ENABLE ,START]        1       24   00:30:00      00:30:00        244GB  pz0105,gcXX
u-short              [ENABLE ,START]        1        8   02:00:00      02:00:00        244GB  pz0105,gcXX
u-regular            [ENABLE ,START]
  |---- u-small       [ENABLE ,START]        4       16   12:00:00      12:00:00        244GB  gcXX,pz0105
  |---- u-medium      [ENABLE ,START]       17       32   12:00:00      12:00:00        244GB  gcXX
  |---- u-large       [ENABLE ,START]       33       64   12:00:00      12:00:00        244GB  gcXX
  |---- u-x-large     [ENABLE ,START]       65      128   06:00:00      06:00:00        244GB  gcXX
u-interactive        [ENABLE ,START]
  |---- u-interactive_1 [ENABLE ,START]      1        1   00:15:00      00:15:00        244GB  pz0105,gcXX
  |---- u-interactive_4 [ENABLE ,START]      2        4   00:05:00      00:05:00        244GB  pz0105,gcXX
u-lecture            [ENABLE ,START]        1        8   00:10:00      00:10:00        244GB  gt00,gtYY
u-lecture8           [DISABLE,START]        1        8   00:10:00      00:10:00        244GB  gtYY
u-tutorial           [ENABLE ,START]        1        8   00:10:00      00:10:00        244GB  gt00
```

使える
キュー名
(リソース
グループ)

現在
使えるか

ノードの
実行情報

課金情報(財布)
実習では1つのみ

rbstat --rsc -b の実行画面例

```
$ rbstat --rsc -b
QUEUE                STATUS          TOTAL RUNNING QUEUED HOLD BEGUN WAIT EXIT TRANSIT NODE
u-debug              [ENABLE ,START]     1       1      0    0     0    0    0       0   54
u-short              [ENABLE ,START]     9       3      5    1     0    0    0       0   16
u-regular            [ENABLE ,START]
  |---- u-small       [ENABLE ,START]    38      10      6   22     0    0    0       0  288
  |---- u-medium      [ENABLE ,START]     2       2      0    0     0    0    0       0  288
  |---- u-large       [ENABLE ,START]     4       2      0    2     0    0    0       0  288
  |---- u-x-large     [ENABLE ,START]     1       0      1    0     0    0    0       0  288
u-interactive        [ENABLE ,START]
  |---- u-interactive_1 [ENABLE ,START]     0       0      0    0     0    0    0       0   54
  |---- u-interactive_4 [ENABLE ,START]     0       0      0    0     0    0    0       0   54
u-lecture            [ENABLE ,START]     0       0      0    0     0    0    0       0   54
u-lecture8           [DISABLE,START]     0       0      0    0     0    0    0       0   54
u-tutorial           [ENABLE ,START]     0       0      0    0     0    0    0       0   54
```

使える
キュー名
(リソース
グループ)

現在
使え
るか

ジョブ
の総数

実行して
いるジョブ
の数

待たされて
いるジョブ
の数

ノードの
利用可能
数