



東京大学  
THE UNIVERSITY OF TOKYO



東京大学情報基盤センター  
INFORMATION TECHNOLOGY CENTER, THE UNIVERSITY OF TOKYO

# オンライン講習会の手引き 「第2回 GPUミニキャンプ ～GPU化にチャレンジする会～」

東京大学情報基盤センター  
2020年8月

質問は下川辺まで: [shimokawabe\(at\)cc.u-tokyo.ac.jp](mailto:shimokawabe(at)cc.u-tokyo.ac.jp)

# はじめに

- 東京大学情報基盤センター(以下, 本センター)では2020年8月3日-8月4日に第137回お試しアカウント付き並列プログラミング講習会「第2回 GPUミニキャンプ～GPU化にチャレンジする会～」を開催します。
  - 本講習会では, Reedbushシステムを利用した実習を実施します。
  - 本講習会は, Zoom、Slackを用いたオンライン講習会として実施します。
- 本資料は, オンライン講習会受講のための事前準備について記載します。
  - 本講習会においては, Reedbushシステムへログインできるようになっていることを前提とします。

# 「第2回 GPUミニキャンプ ～GPU化にチャレンジする会～」概略

- <https://www.cc.u-tokyo.ac.jp/events/lectures/137/>
- **開催日:**  
2020年 8月3日(月) 10:00 - 17:00  
2020年 8月4日(火) 10:00 - 17:00
- **形態:** Zoomによるオンライン講習会
- **講習会プログラム:**
  - 1日目
    - 10:00 - 10:20 Reedbush-Hの使い方講座
    - 10:20 - 10:50 自己紹介と目標設定など(1枚スライドによる1分自己紹介)
    - 10:50 - 13:00 実践(適宜自由に休憩)
    - 13:00 - 13:30 お役立ち情報
    - 13:30 - 16:50 実践(適宜自由に休憩)
    - 16:50 - 17:00 事務連絡・終了
  - 2日目
    - 10:00 - 10:20 昨日の結果と本日の目標設定
    - 10:20 - 16:00 実践(適宜自由に休憩)
    - 16:00 - 16:50 実施内容の紹介
    - 16:50 - 17:00 アンケート記入・終了

# お願い等

- ハンズオンのためのPC, Zoom及びスパコンへ接続するためのネットワーク環境は各受講者でご準備ください。
- PCは Windows/Microsoft Update, Apple Security Updateなどで最新のセキュリティアップデートを行ってください。
- 必ずウイルス対策ソフトウェアをインストールし, ウィルス検索を実行して問題がないことを事前に確認してから受講してください。
  - セキュリティ対策未実施の場合はオンライン講習会受講を認めません。
- OSは、Windows、Macどちらでも構いませんが、SSHを用いてセンターのスーパーコンピューターへ接続ができることが必要です(後述)。
- 演習の実施に当たり, 受講生にセンターのスーパーコンピューターを1月間利用できる無料アカウント(お試しアカウント)を発行します。

1. 自分のパソコンへのSSH環境の準備
2. Reedbushスパコンへのログイン
3. Reedbush利用の準備
4. コミュニケーションツールの準備

# 1. 自分のパソコンへのSSH環境の準備

## 1.1 SSHを使うためのターミナルの準備

- スパコンへの接続にはSSH(Secure Shell)を使います
- SSHを使える環境を準備します
  - Windows の方は以下などをインストール
    - WSL (Windows Subsystem for Linux, windows10以降)
    - Cygwin (次ページよりインストール方法の説明あり。60-90分かかるので、お早めに。)
    - PuTTY
  - Mac の方はターミナルにデフォルトで入っているはず

## 1.2 SSHで使う公開鍵の作成

- sshにはパスワード認証方式と公開鍵認証方式がありますが、公開鍵認証方式を使います

上記が済んでいる方は 2. Reedbushスパコンへのログインへ

# Cygwin: Windows上のUNIXライクな環境

<https://www.cygwin.com/>

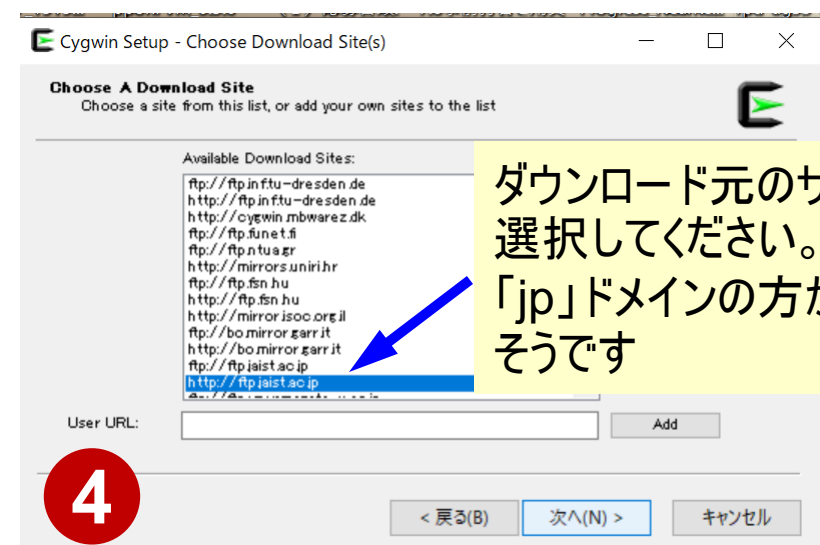
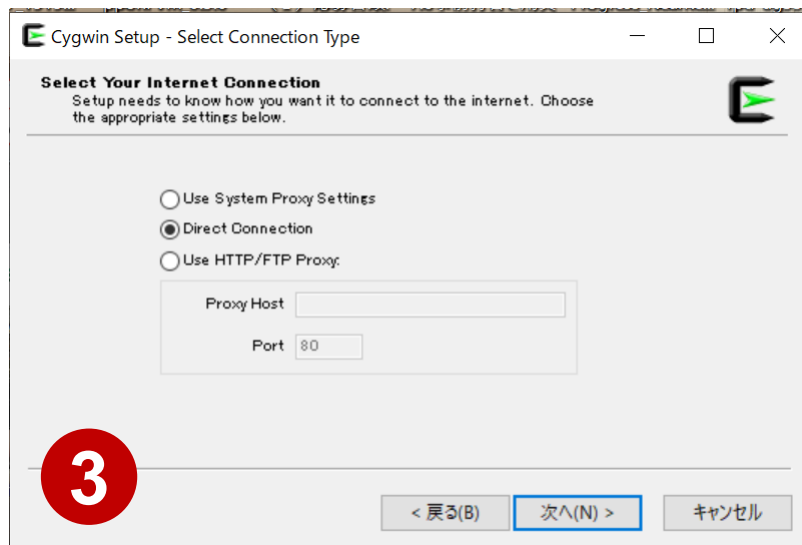
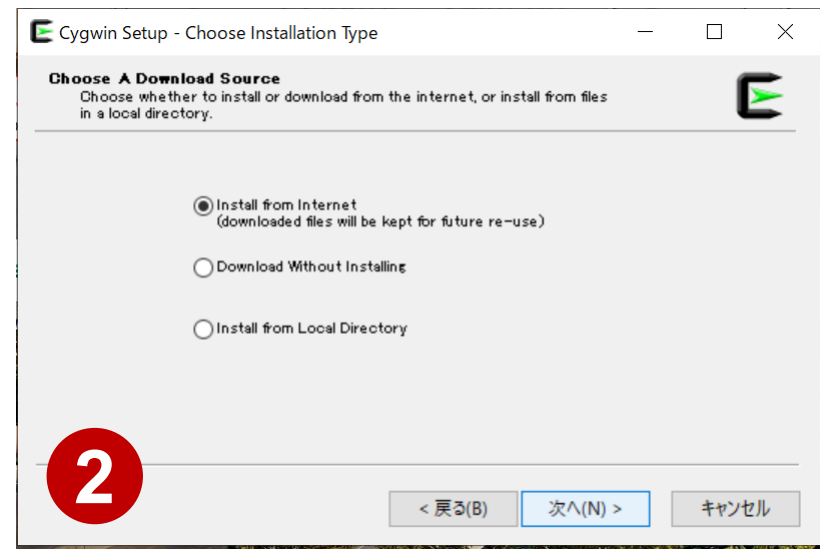
- 利用者ガイド
  - <https://cygwin.com/cygwin-ug-net.html>
- まずやるべきこと
  - インストーラ(setup-x86\_64/32.exe)を予め入手
  - インターネットに接続し, インストーラをダブルクリック
  - 以下指示に従ってインストールを進めてください

以下しばらくはCygwinの話



# 指示に従ってください

<https://www.cygwin.com/>





# まずはデフォルト機能のインストールから

<https://www.cygwin.com/>

- 基本的な機能はデフォルトのインストールでOKですが、本講習会で必要なものが抜けている可能性があります。
  - 従ってマニュアルでインストールする必要があります
  - インストーラをダブルクリックすれば後で追加も可能です
- **本講習会では下記が必須です（デフォルトのインストールでは抜けている可能性あり）**
  - **gcc-core (for C/C++ users)**
  - **gcc-fortran (for Fortran users)**
  - **openssh (for all users)**
  - **openssl (for all users)**
  - **make (for all users)**
  - **emacs, vim etc.**
- **インストールされているかどうかは確認が可能です**

# “gcc-core”の有無に関するチェック

Type “gcc-core”

Cygwin Setup - Select Packages

Select packages to install

View Full Search gcc-core Clear

Keep  Best  Sync  Test

Package	Current	New	Src?	Categories	Size	Description
cygwin32-gcc-core		Skip	<input type="checkbox"/>	Devel	16,464k	GCC for Cygwin 32bit toolchain (C, OpenMP)
djgpp-gcc-core		Skip	<input type="checkbox"/>	Devel	7,926k	GCC for DJGPP toolchain (C)
gcc-core	9.3.0-1	Keep	<input type="checkbox"/>	Devel	20,500k	GNU Compiler Collection (C, OpenMP)
mingw64-i686-gcc-core		Skip	<input type="checkbox"/>	Devel	16,851k	GCC for Win32 (i686-w64-mingw32) toolchain (C, OpenMP)
mingw64-x86_64-gcc-core		Skip	<input type="checkbox"/>	Devel	17,464k	GCC for Win64 toolchain (C, OpenMP)

Hide obsolete packages

< 戻る(B) 次へ(N) > キャンセル

20:19 2020/04/14

# “gcc-core”の有無に関するチェック

Type “gcc-core”

Cygwin Setup - Select Packages

Select packages to install

View Full Search gcc-core Clear

Keep  Best  Sync  Test

Package	Current	New	Src?	Categories	Size	Description
cygwin32-gcc-core		Skip	<input type="checkbox"/>	Devel	16,464k	GCC for Cygwin 32bit toolchain (C, OpenMP)
gcc-core	9.3.0-1	Keep	<input type="checkbox"/>	Devel	7,926k	GCC for DJGPP toolchain (C)
gcc-core		Skip	<input type="checkbox"/>	Devel	20,500k	GNU Compiler Collection (C, OpenMP)
gcc-core		Skip	<input type="checkbox"/>	Devel	16,851k	GCC for Win32 (686-w64-mingw32) toolchain (C, OpenMP)
mingw64-x86_64-gcc-core		Skip	<input type="checkbox"/>	Devel	17,464k	GCC for Win64 toolchain (C, OpenMP)

“Keep”と出てきたら  
“gcc-core”はインストール済み

Hide obsolete packages

<戻る(B) 次へ(N) > キャンセル

20:19  
2020/04/14

# “g++”のインストール例 (1/4)

Type “g++”

Cygwin Setup - Select Packages

Select Packages  
Select packages to install

View Full Search  Clear

Keep  Best  Sync  Test

Package	Current	New	Src?	Categories	Size	Description
cygwin32-gcc-g++		Skip	<input type="checkbox"/>	Devel	10,456k	GCC for Cygwin 32bit toolchain (C++)
djgpp-gcc-g++		Skip	<input type="checkbox"/>	Devel	8,279k	GCC for DJGPP toolchain (C++)
gcc-g++		Skip	<input type="checkbox"/>	Devel	16,257k	GNU Compiler Collection (C++)
mingw64-i686-gcc-g++		Skip	<input type="checkbox"/>	Devel	14,358k	GCC for Win32 (i686-w64-mingw32) toolchain (C++)
mingw64-x86_64-gcc-g++		Skip	<input type="checkbox"/>	Devel	14,781k	GCC for Win64 toolchain (C++)

“Skip” が現れたら「未インストール」を意味する

以下「g++」を例にマニュアルインストールの実施方法を紹介する

Hide obsolete packages

< 戻る(B) 次へ(N) > キャンセル

# “g++”のインストール例 (2/4)

Cygwin Setup - Select Packages

Select Packages  
Select packages to install

View Full Search g++ Clear  Keep  Best  Sync  Test

Package	Current	New	Src?	Categories	Size	Description
cygwin32-gcc-g++		Skip	<input type="checkbox"/>	Devel	10,456k	GCC for Cygwin 32bit toolchain (C++)
diagn-gcc-g++		Skip	<input type="checkbox"/>	Devel	8,279k	GCC for DJGPP toolchain (C++)
gcc-g++		Skip	<input type="checkbox"/>	Devel	16,257k	GNU Compiler Collection (C++)
mingw64-i686-gcc-g++		Skip	<input type="checkbox"/>	Devel	14,358k	GCC for Win32 (i686-w64-mingw32) toolchain (C++)
mingw64-x86_64-gcc-g++		Skip	<input type="checkbox"/>	Devel	14,781k	GCC for Win64 toolchain (C++)

プルダウンメニュー

Hide obsolete packages

< 戻る(B) 次へ(N) > キャンセル

20:18  
2020/04/14

# “g++”のインストール例 (3/4)

Cygwin Setup - Select Packages

Select Packages  
Select packages to install

View Full Search g++ Clear  Keep  Best  Sync  Test

Package	Current	New	Src?	Categories	Size	Description
cygwin32-gcc-g++		Skip	<input type="checkbox"/>	Devel	10,456k	GCC for Cygwin 32bit toolchain (C++)
diipp-gcc-g++		Skip	<input type="checkbox"/>	Devel	8,279k	GCC for DJGPP toolchain (C++)
gcc-g++		Skip	<input checked="" type="checkbox"/>	Devel	16,257k	GNU Compiler Collection (C++)
mingw64-i686-gcc-g++		Skip	<input type="checkbox"/>	Devel	14,358k	GCC for Win32 (i686-w64-mingw32) toolchain (C++)
mingw64-x86_64-gcc-g++		Skip	<input type="checkbox"/>	Devel	14,781k	GCC for Win64 toolchain (C++)

Uninstall  
 Skip  
 7.4.0-1  
 8.3.0-1 (Test)  
 9.2.0-1 (Test)  
 9.2.0-2  
 9.2.0-3  
 9.3.0-1

適切なバージョンを選択  
(通常は最新版)

Hide obsolete packages

< 戻る(B) 次へ(N) > キャンセル

20:18  
2020/04/14

# “g++”のインストール例 (4/4)

Cygwin Setup - Select Packages

Select Packages  
Select packages to install

View Full Search g++ Clear  Keep  Best  Sync  Test

Package	Current	New	Src?	Categories	Size	Description
cygwin32-gcc-g++		Skip	<input type="checkbox"/>	Devel	10,456k	GCC for Cygwin 32bit toolchain (C++)
diipp-gcc-g++		Skip	<input type="checkbox"/>	Devel	8,279k	GCC for DJGPP toolchain (C++)
gcc-g++		9.3.0-1	<input type="checkbox"/>	Devel	16,257k	GNU Compiler Collection (C++)
mingw64-i686-gcc-g++		Skip	<input type="checkbox"/>	Devel	14,358k	GCC for Win32 (i686-w64-mingw32) toolchain (C++)
mingw64-x86_64-gcc-g++		Skip	<input type="checkbox"/>	Devel	14,781k	GCC for Win64 toolchain (C++)

「Skip」のかわりにバージョン番号が出てきたら“g++”のインストール準備完了  
(インストールは完了していない)

ここをクリック

Hide obsolete packages

< 戻る(B) 次へ(N) > キャンセル

# “gcc” : インストールの確認

```
$ gcc -v
```

```
組み込み spec を使用しています。
```

```
COLLECT_GCC=gcc
```

```
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-pc-cygwin/9.3.0/lto-wrapper.exe
```

```
ターゲット: x86_64-pc-cygwin
```

```
configure 設定: /cygdrive/i/szsz/tmp/gcc/gcc-9.3.0-1.x86_64/src/gcc-9.3.0/configure --
```

```
srcdir=/cygdrive/i/szsz/tmp/gcc/gcc-9.3.0-1.x86_64/src/gcc-9.3.0 --prefix=/usr --exec-prefix=/usr --
```

```
localstatedir=/var --sysconfdir=/etc --docdir=/usr/share/doc/gcc --htmldir=/usr/share/doc/gcc/html -C
```

```
--build=x86_64-pc-cygwin --host=x86_64-pc-cygwin --target=x86_64-pc-cygwin --without-libiconv-prefix -
```

```
--without-libintl-prefix --libexecdir=/usr/lib --enable-shared --enable-shared-libgcc --enable-static -
```

```
--enable-version-specific-runtime-libs --enable-bootstrap --enable-__cxa_atexit --with-dwarf2 --with-
```

```
tune=generic --enable-languages=c,c++,fortran,lto,objc,obj-c++ --enable-graphite --enable-
```

```
threads=posix --enable-libatomic --enable-libgomp --enable-libquadmath --enable-libquadmath-support --
```

```
disable-libssp --enable-libada --disable-symvers --with-gnu-ld --with-gnu-as --with-cloog-
```

```
include=/usr/include/cloog-isl --without-libiconv-prefix --without-libintl-prefix --with-system-zlib -
```

```
--enable-linker-build-id --with-default-libstdcxx-abi=gcc4-compatible --enable-libstdcxx-filesystem-ts
```

```
スレッドモデル: posix
```

```
gcc バージョン 9.3.0 (GCC)
```



# “gfortran” : インストールの確認

```
$ gfortran -v
```

組み込み spec を使用しています。

COLLECT\_GCC=gfortran

ターゲット: x86\_64-pc-cygwin

configure 設定: /cygdrive/i/szsz/tmp/gcc/gcc-9.3.0-1.x86\_64/src/gcc-9.3.0/configure --srcdir=/cygdrive/i/szsz/tmp/gcc/gcc-9.3.0-1.x86\_64/src/gcc-9.3.0 --prefix=/usr --exec-prefix=/usr --localstatedir=/var --sysconfdir=/etc --docdir=/usr/share/doc/gcc --htmldir=/usr/share/doc/gcc/html -C --build=x86\_64-pc-cygwin --host=x86\_64-pc-cygwin --target=x86\_64-pc-cygwin --without-libiconv-prefix --without-libintl-prefix --libexecdir=/usr/lib --enable-shared --enable-shared-libgcc --enable-static --enable-version-specific-runtime-libs --enable-bootstrap --enable\_\_cxa\_atexit --with-dwarf2 --with-tune=generic --enable-languages=c,c++,fortran,lto,objc,obj-c++ --enable-graphite --enable-threads=posix --enable-libatomic --enable-libgomp --enable-libquadmath --enable-libquadmath-support --disable-libssp --enable-libada --disable-symvers --with-gnu-ld --with-gnu-as --with-cloog-include=/usr/include/cloog-isl --without-libiconv-prefix --without-libintl-prefix --with-system-zlib --enable-linker-build-id --with-default-libstdcxx-abi=gcc4-compatible --enable-libstdcxx-filesystem-ts

スレッドモデル: posix

gcc バージョン 9.3.0 (GCC)

# “ssh-keygen (OpenSSH)” : インストールの確認

```
$ ssh-keygen --h
```

```
ssh-keygen: unknown option -- -  
usage: ssh-keygen [-q] [-b bits] [-C comment] [-f output_keyfile] [-m format]  
                [-t dsa | ecdsa | ecdsa-sk | ed25519 | ed25519-sk | rsa]  
                [-N new_passphrase] [-O option] [-w provider]  
ssh-keygen -p [-f keyfile] [-m format] [-N new_passphrase]  
            [-P old_passphrase]  
ssh-keygen -i [-f input_keyfile] [-m key_format]  
ssh-keygen -e [-f input_keyfile] [-m key_format]  
ssh-keygen -y [-f input_keyfile]  
  
(...)  
  
ssh-keygen -L [-f input_keyfile]  
ssh-keygen -A [-f prefix_path]  
ssh-keygen -k -f krl_file [-u] [-s ca_public] [-z version_number]  
            file ...  
ssh-keygen -Q -f krl_file file ...  
ssh-keygen -Y find-principals -s signature_file -f allowed_signers_file  
ssh-keygen -Y check-novalidate -n namespace -s signature_file  
ssh-keygen -Y sign -f key_file -n namespace file ...  
ssh-keygen -Y verify -f allowed_signers_file -I signer_identity  
            -n namespace -s signature_file [-r revocation_file]
```

# “ssh (OpenSSH)” : インストールの確認

```
$ ssh
```

```
usage: ssh [-46AaCfGgKkMnqsTtVvXxYy] [-B bind_interface]
          [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
          [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
          [-i identity_file] [-J [user@]host[:port]] [-L address]
          [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
          [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
          [-w local_tun[:remote_tun]] destination [command]
```

# “make, emacs, vi etc” : インストールの確認

```
$ make -version
```

```
GNU Make 4.3
```

```
このプログラムは x86_64-pc-cygwin 用にビルドされました
```

```
Copyright (C) 1988-2020 Free Software Foundation, Inc.
```

```
ライセンス GPLv3+: GNU GPL バージョン 3 以降 <http://gnu.org/licenses/gpl.html>
```

```
これはフリーソフトウェアです: 自由に変更および配布できます.
```

```
法律の許す限り、 無保証 です.
```

```
$ emacs -version
```

```
GNU Emacs 26.3
```

```
Copyright (C) 2019 Free Software Foundation, Inc.
```

```
GNU Emacs comes with ABSOLUTELY NO WARRANTY.
```

```
You may redistribute copies of GNU Emacs
```

```
under the terms of the GNU General Public License.
```

```
For more information about these matters, see the file named COPYING.
```

```
$ vi -version
```

```
VIM - Vi IMproved 8.2 (2019 Dec 12, compiled Mar 30 2020 21:54:08)
```

```
Garbage after option argument: "--version"
```

```
More info with: "vim -h"
```

1. 自分のパソコンへのSSH環境の準備
  - 1.1 SSHを使うためのターミナルの準備
  - 1.2 SSHで使う公開鍵の作成
2. Reedbushスパコンへのログイン
3. Reedbush利用の準備
4. コミュニケーションツールの準備

# 1.2 SSHで使う公開鍵の作成

- SSH公開鍵認証 (SSH Public Key Authentication, SSH=Secure Shell)に基づく
  - パスワード認証よりも安全, と言われている
- 手順
  - **Windows: Cygwinなどを立ち上げる, Mac・Unix: Terminal起動**
  - 公開鍵が作成済みかどうかを確認
    - 過去に作ったものがあれば、それを使って構わない
  - ssh-keygen コマンドを使い、PC上で鍵 (秘密鍵, 公開鍵) を生成する
    - 秘密鍵, 公開鍵のペアができる。公開鍵をスパコンなどに置き、自分のPCの秘密鍵と照合する。**秘密鍵は移動してはならない。他人に見せてもいけない。**
    - **パスフレーズ (Passphrase)**: 鍵認証のためのパスワードを設定する

# PC上の鍵（秘密鍵，公開鍵）を確認

```
$ cd .ssh
```

```
$ ls
```

```
id_rsa          ⇒秘密鍵 (Private Key)  
id_rsa.pub      ⇒公開鍵 (Public Key)
```

```
$ cat id_rsa.pub
```

```
ssh-rsa  
AAAAB3NzaC1yc2EAAAADAQABAAQDA6InmOYYaCrWjQDukjiNEfdW8veUwJyZtEI3oDu0A28  
eey6p0wbtI7JB09xnI1707HG4yYv0M81+/nIAHy5tAfJly0dsPzjTgdTBLdgi3cSf5pWEY6U96  
yaEr0Ei8Wge1HkXrhcwUjGDVTzvT0Refe6zLdRziL/KNmmesSQfR5IsZ/ihsjMgFxGaKsHHq/I  
ErCtHIIIf9V/Ds2yj6vkAaWH6asBn+ZsRiRFvwHPhkYAnp/j3LY6b8Qfqg0p4WZRenh/HgySWT  
YIGi8x67VzMaUIm9qIK0QFMCaK2riviX1fmbwyWJ/vrWDqiek6YXoxLDu+GPeQ4CPvxJcZnqF9g  
f3 nakajima@KNs-NEW-VAIO
```

公開鍵未作成の場合には  
id\_rsa, id\_rsa.pub が  
出てこない

# PC上で鍵（秘密鍵，公開鍵）を生成

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

Generating public/private rsa key pair.

Enter file in which to save the key (/home/user/.ssh/id\_rsa):

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /home/user/.ssh/id\_rsa.

Your public key has been saved in /home/user/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:vt880+PTcscHk0yabvxGjeRsMWLAWds+ENsDcReNwKo nakajima@KNs-NEW-VAIO

The key's randomart image is:

```
+---[RSA 2048]---+
|                 |
|  . 0=00.0+      |
|  + 0...         |
|  .+0+          |
|  +0B.          |
| So *0*         |
| .E B.o         |
|  . = . 0        |
|  . =0B 0 +     |
|  .+0+*0 ..     |
|                 |
+---[SHA256]---+
```

## 操作手順

- `ssh-keygen -t rsa <Return>`
- `<Return>`
- `お好きなPassphrase <Return>`
- `同じPassphrase <Return>`



1. 自分のパソコンへのSSH環境の準備
2. Reedbushスパコンへのログイン
3. Reedbush利用の準備
4. コミュニケーションツールの準備

## 2. Reedbushスパコンへのログイン

- 手順

- ①スパコンポータルサイトにログインする
  - センターから供給された利用者ID (t00XYZ)と「初期パスワード」を使用
  - ポータルサイトにログイン後、パスワード(Password)変更を求められる、字数、使用文字等に色々規則があるので注意すること
- ②スパコンポータルサイトに「公開鍵」を登録する
- ③PCからsshによってスパコンにログインする

# ①スパコンポータルサイトにログイン(1/3)

## 情報基盤センターから送付されたファイル

Oakforest-PACS\_workshop\_t00100.pdf - Adobe Acrobat Reader DC

ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)

ホーム ツール 2019年源泉徴収.pdf... Oakbridge-CX\_work... Oakforest-PACS\_wor... × ログイン

共有

(教育利用)

Oakforest-PACS 利用登録のお知らせ  
Notification of Your Account of Supercomputer System (Oakforest-PACS)

様

下記のとおり登録しましたのでお知らせします。

2020年4月8日  
東京大学情報基盤センター長  
Information Technology Center, The University of Tokyo

公印  
省略

プロジェクト名称	お試しアカウント付き並列プログラミング講習会	プロジェクトコード Project code	gt00
利用期間	2020年4月 ~ 2020年6月		
利用者番号 User ID	t00XYZ	初期パスワード※ Password	01234567
研究	情報基盤センターから送付された利用者ID (t00XYZ)	情報基盤センターから送付された初期パスワード(8桁)	

14:58 2020/04/12

# ①スパコンポータルサイトにログイン(2/3)

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

Login to Reedbush System

You must enter a username and password to login.

Username

Password

Login Clear

情報基盤センターから  
送付された利用者ID  
(t00XYZ)

情報基盤センターから  
送付された初期パス  
ワード

# ①初期パスワードの変更(3/3)

Reedbush Portal

reedbush-www.cc.u-tokyo.ac.jp

Login: g24000

Change Language  
prepost予約状況  
システムからのお知らせ  
ツール  
ドキュメント閲覧  
パスワードの変更  
公開鍵アップロード

ログアウト

## パスワードの変更

### Password change

※パスワードの条件  
8文字以上であること  
数字、大文字、小文字、特殊文字(#、\$、&、%、+、!、-など) をそれぞれ1文字以上含むこと  
Linux辞書にある単語ではないこと  
前回、前々回と同じパスワードではないこと

変更するユーザー g24000

現在のパスワード

あたらしいパスワード

あたらしいパスワードをもう1度

変更する

情報基盤センターから送付された初期パスワード

変更後のパスワードを入力(2回)

### パスワード規約

- 8文字以上，現在と3文字以上異なる
- 2世代前までと異なる
- 英字(小文字，大文字)，数字，特殊文字各1字以上
- Linux辞書に登録されている語は不可
- 全角文字不可

## ②公開鍵登録(id\_rsa.pub) (1/2)

Reedbush Portal

reedbush-www.cc.u-tokyo.ac.jp

Login: g24000

Change Language  
prepost予約状況  
システムからのお知らせ  
ツール  
ドキュメント閲覧  
パスワードの変更  
公開鍵アップロード  
ログアウト

### 公開鍵追加

※注意事項  
改行文字や全角文字が含まれていないこと、ヘッダ(ssh-rsa、ssh-dss) が付与されていることを確認してください。  
RSA公開鍵は2048bit、DSA公開鍵は1024bit以上で作成してください。  
鍵の形式が「ssh-rsa 文字列 鍵の名前（通常はユーザ名@ホスト名）」の3つのフィールドであることを確認してください。

Key digits

作成

← Key List に戻る

1. 「公開鍵アップロード」を選択
2. 「id\_rsa.pub」を貼り付ける
3. 「作成」をクリック

## ②公開鍵登録(id\_rsa.pub) (2/2)

```
$ cd .ssh
```

```
$ ls
```

```
id_rsa  
id_rsa.pub
```

```
$ cat id_rsa.pub
```

```
ssh-rsa
```

```
AAAAB3NzaC1yc2EAAAADAQABAAQDA6Inm0YYaCrWjQDukjiNEfdW8veUwJyZtEI3oDu0A28  
eey6p0wbtI7JB09xnI1707HG4yYv0M81+/nIAHy5tAfJly0dsPzjTgdTBLdgi3cSf5pWEY6U96  
yaEr0Ei8Wge1HkXrhcwUjGDVTzvT0Ref6zLdRziL/KNmmesSQfR5IsZ/ihsjMgFxGaKsHHq/I  
ErCtHIIIf9V/Ds2yj6vkAaWH6asBn+ZsRiRFvwHPhkYAnp/j3LY6b8Qfqg0p4WZRenh/HgySWT  
YIGi8x67VzMaUIm9qIK0QFMCaK2rivX1fmbwyWJ/vrWDqiek6YXoxLDu+GPeQ4CPvxJcZnqF9g  
f3 nakajima@KNs-NEW-VAIO
```

### 公開鍵のコピー手順

- ターミナル(cygwinなど)を開く
- **cd .ssh <Return>**
- **cat id\_rsa.pub <Return>**
- “ssh-rsa”にカーソルを合わせ
- 最後の「ユーザ名@ホスト名」までを選択して「Copy」によって記憶

## ③PCからログイン(1/2)

```
$ ssh t00XYZ@reedbush.cc.u-tokyo.ac.jp  
Enter passphrase for key '/home/user/.ssh/id_rsa:
```

Your Passphrase

Return

1. `ssh t00XYZ@reedbush.cc.u-tokyo.ac.jp` <Return>
2. **鍵生成時に打ち込んだPassphrase** <Return>



## ③PCからログイン(2/2)

```
$ ssh t00XYZ@reedbush.cc.u-tokyo.ac.jp
Warning: No xauth data; using fake authentication data for X11 forwarding.
Last login: Sat Apr 7 14:31:04 2018 from 113.41.142.145
*****
Reedbush      system will be SHUT DOWN   on Tue Jul 31, 2020, at 09:00
Reedbush      system will be STARTED       on Tue Jul 31, 2020, at 17:00
Operation for Reedbush-U has been terminated. (June 30, 2020)

For more information about this service, see

  https://www.cc.u-tokyo.ac.jp/supercomputer/schedule.php
  https://www.cc.u-tokyo.ac.jp/guide/hpc/rbh/
*****

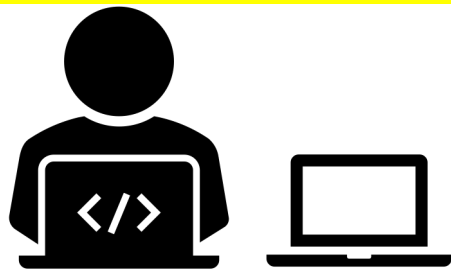
[t00XYZ@reedbush-u4 ~]$
```

ログインに成功したら、  
今後のメンテナンス  
のスケジュールなどが  
表示される

# SSH公開鍵認証の手順(1/4)

## ①PC上での秘密鍵・公開鍵作成

ここでは、OBCX、OFPスパコンを例に挙げて説明していますが、Reedbushでも同様です。



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

**id\_rsa**

秘密鍵/Private Key

+ Passphrase

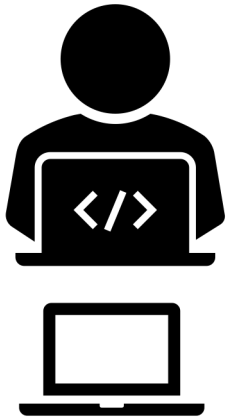
**id\_rsa.pub**

公開鍵/Public Key



# SSH公開鍵認証の手順(2/4)

## ②スパコンポータルサイトへのログイン



tXYZZZ  
+ Password

Portal Site  
OBCX



id\_rsa  
秘密鍵/Private Key

+ Passphrase

id\_rsa.pub  
公開鍵/Public Key



tABCCC  
+ Password

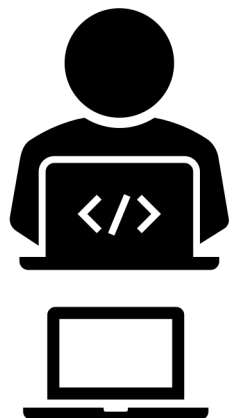
Portal Site  
OFP



# SSH公開鍵認証の手順(3/4)

## ③公開鍵(id\_rsa.pub)の登録

同じ公開鍵を複数のスパコンに登録可能



**id\_rsa**  
秘密鍵/Private Key

+ Passphrase

**id\_rsa.pub**  
公開鍵/Public Key

Portal Site  
OBCX



**id\_rsa.pub**  
公開鍵/Public Key

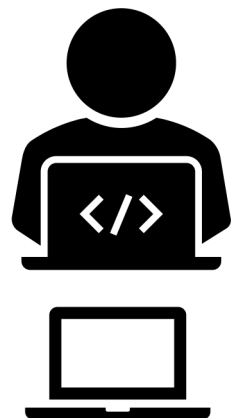
Portal Site  
OFP



# SSH公開鍵認証の手順(3/4)

## ③公開鍵(id\_rsa.pub)の登録

同じ公開鍵を複数のスパコンに登録可能



**id\_rsa**  
秘密鍵/Private Key

+ Passphrase



Portal Site  
OBCX



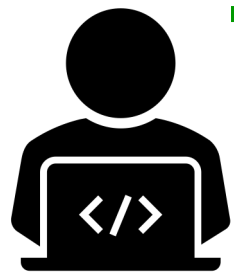
Portal Site  
OFP



# SSH公開鍵認証の手順(4/4)

## ④PCからスパコンへのログイン

秘密鍵(id\_rsa) + Passphrase



```
$> ssh tXYZZZ@obcx.cc.u-tokyo.ac.jp
```

**id\_rsa**  
秘密鍵/Private Key



+ Passphrase



```
$> ssh tABCCC@ofp.jcahpc.jp
```

**id\_rsa**  
秘密鍵/Private Key



+ Passphrase



# SSH Public Key Authentication

## SSH公開鍵認証

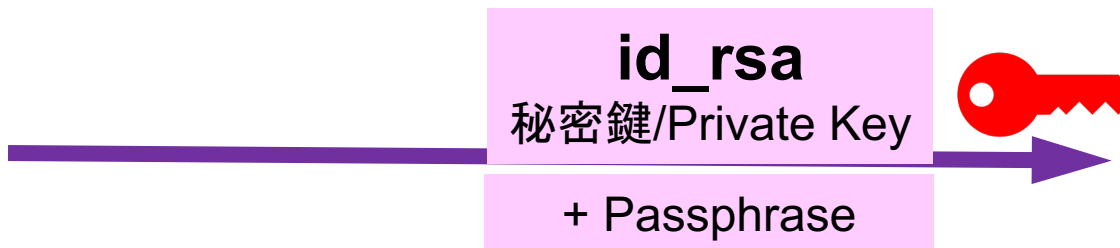
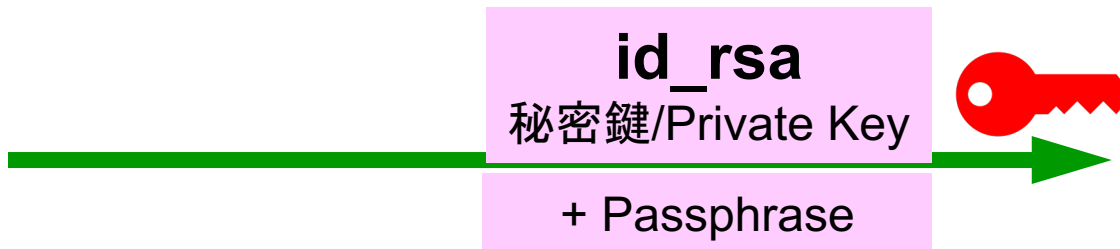
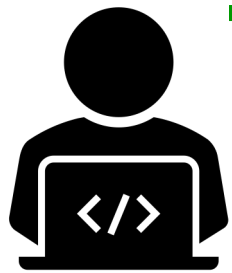
SSH= Secure Shell

- **id\_rsa**
  - Private Key (秘密鍵) : PC上
  - 文字通り「秘密」にしておくこと
    - 他の人に送ってはいけない
    - 基本的には作成した場所からコピーしたり移動することもしないこと
- **id\_rsa.pub**
  - Public Key (公開鍵) : スパコン上
  - コピー可能, 他の人にe-mailで送ることも可能
- **もし複数のPCからスパコンにログインする場合は, 各PCごとに「公開鍵・秘密鍵」のペアをssh-keygenによって作成**
  - 各スパコンに複数の公開鍵を登録することは可能
  - スパコン上の公開鍵のうちの一つがPC上の「秘密鍵 + Passphrase」とマッチすると確認されるとログインできる

# SSH公開鍵認証の手順(4/4)

## ④PCからスパコンへのログイン

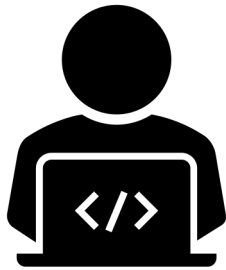
秘密鍵(id\_rsa) + Passphrase





# 複数のPCからスパコンへログインする場合には 各PCで「公開鍵・秘密鍵」のペア作成

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



**id\_rsa**  
秘密鍵/Private Key

+ Passphrase

**id\_rsa.pub**  
公開鍵/Public Key

**id\_rsa**  
秘密鍵/Private Key

+ Passphrase

**id\_rsa.pub**  
公開鍵/Public Key

Portal Site  
OBCX



Portal Site  
OFP



# スパコンには複数の公開鍵を登録できる

Oakbridge-CX 利用支援ポータル

https://obcx-www.cc.u-tokyo.ac.jp/cgi-bin/hpcportal\_u.ja/index.cgi

Oakbridge-CX 利用支援ポータル

ログアウト

- お知らせ
- SSH公開鍵登録**
- メール転送設定
- パスワード変更
- トークン表示
- ディスク使用量表示
- プリポスト予約
- ドキュメント閲覧
- OSS

## SSH公開鍵登録

公開鍵を登録しました。

登録されている公開鍵	ssh-rsa AAAAB3NzaC.....JcZnqF9gf3	表示	削除
	ssh-rsa AAAAB3NzaC.....pWGVie6w==	表示	削除

登録方式

直接入力

ファイルアップロード

Copyright 1999 FUJITSU LIMITED

ページ内検索

すべて強調表示(A) 大文字/小文字を区別(C) 発音区別符号を区別(I) 単語単位(W)

22:36  
2020/04/15

# スパコンには複数の公開鍵を登録できる

```
$ cd .ssh  
$ ls authorized_keys
```

```
authorized_keys
```

```
$ cat authorized_keys
```

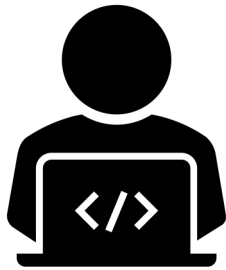
```
ssh-rsa  
HGCA3NzaC1yc2EAAAABIwAAAQEA1r0Hr8M1JIJB02n9S0GQm0xzGCwh3PpcJo7Z8oDr6HCAXhbK  
zHA0ibRMJFCwDJCRGNJIYiHEYHWzouuXGNa9teso7aXYkq2Pxb076C60ZCPoLqf/jQRqnUSnjHJ4  
UgmDdIQWaAks+q/2Ex0wjBB6GZmaHGijTxim0FGiM1DI780HkHC8pFzjvP2kT9yRvykv0VvIG10V  
Yi+5CawYfuR0iRBjfUS47RS0ICzjNP20pY057DUCf0v+/8B1+l1wiIbjKQHjuNp5XucIFfFdGaxf  
JchD/sB5sRxtYfz80xzwGmN8pVecpUjd//xAqdYYHmLAKUE2oH8MnBIRybpWGVie6w64  
ssh-rsa  
AAAAB3NzaC1yc2EAAAADAQABAAQDA6Inm0YYaCrWjQDukjiNEfdW8veUwJyZtEI3oDu0A28ee  
y6p0wbtI7JB09xnI1707HG4yYv0M81+/nIAHy5tAfJly0dsPzjTgdTBLdgi3cSf5pWEY6U96yaEr  
0Ei8Wge1HkXrhcwUjGDVTzvT0Refe6zLdRziL/KNmmesSQfR5lsZ/ihsjMgFxGaKsHHq/IErCtHI  
IIf9V/Ds2yj6vkAaWH6asBn+ZsRiRFvwHPhkYAnp/j3LY6b8Qfqq0p4WZRenh/HgySWTYIGi8x67  
VzMaUIm9qIK0QFMCaK2rivX1fmbwyWJ/vrWDqiek6YXoxLDu+GPeQ4CPvxJcZnqF9gf3
```

```
$ cp authorized_keys tmp  
$ cat tmp new_public.key > authorized_keys
```

**.ssh/authorized\_keys**には登録された公開鍵が格納されている。  
このファイルの後ろに新たな公開鍵(**new\_public.key**)を付け加えることができる。

# 各スパコンに複数の鍵を登録する

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



**id\_rsa**  
秘密鍵/Private Key

+ Passphrase

**id\_rsa.pub**  
公開鍵/Public Key



**id\_rsa**  
秘密鍵/Private Key

+ Passphrase

**id\_rsa.pub**  
公開鍵/Public Key

Portal Site  
OBCX



**id\_rsa.pub**  
公開鍵/Public Key



**id\_rsa.pub**  
公開鍵/Public Key

Portal Site  
OFFP



**id\_rsa.pub**  
公開鍵/Public Key



**id\_rsa.pub**  
公開鍵/Public Key

1. 自分のパソコンへのSSH環境の準備
2. Reedbushスパコンへのログイン
3. Reedbush利用の準備
4. コミュニケーションツールの準備

# Reedbush 利用の手引き

わからないことがあれば、まず下記を参照してください。

<https://reedbush-www.cc.u-tokyo.ac.jp> でポータルサイトへ入り、  
左のメニューから「**ドキュメント閲覧**」へ進みます。

ドキュメントの中の

- Reedbush Quick Start Guide: まず目を通します。
- Reedbush システム利用手引書(概要・Reedbush-U 編) : ジョブ投入なのでわからなければ読みます。
- Reedbush システム利用手引書(Reedbush-H 編): GPU関係のアプリやジョブでわからなければ読みます。

# ポータルサイトでのマニュアル等閲覧(1/2)

Reedbush Portal

reedbush-www.cc.u-tokyo.ac.jp

Login: g24000

- Change Language
- prepost予約状況
- システムからのお知らせ
- ツール
- ドキュメント閲覧
- パスワードの変更
- 公開鍵アップロード
- ログアウト

## ドキュメント閲覧について

ReedbushマニュアルのWeb閲覧サービスを利用するにあたっては、以下の禁止事項を遵守していただきます。

- 核兵器または生物化学兵器およびこれらを運搬するためのミサイルなどの大量破壊兵器の開発・設計・製造・保管および使用などの目的に利用しない。
- スーパーコンピュータの利用が認められた利用者本人のみが使用し、他者に利用させない。
- 本マニュアルの情報(印刷、コピーしたものを含む)を、利用者以外に開示または提供しない。
- 当センターが上記条項の違反、その他不正利用を検知した場合、当センターは利用者のWeb閲覧サービスの利用を停止することができる。また、利用者はこれに対して一切異議を唱えない。

上記の規定を遵守しますか?

## Conditions and restrictions for using manuals

You are required to accept following conditions and restrictions to use Reedbush Web Portal manual.

- You are not allowed to use manuals for the development, production and/or use of the weapons of mass destruction (WMD), namely nuclear weapon, chemical weapon, biological weapon and/or WMD delivery systems such as missiles.
- Only who are accepted to use supercomputer are allowed to use Reedbush Web Portal manual.
- You are not allowed to disclose or provide Reedbush manuals(including printed copy) for third party.
- Information Technology Center is entitled to cancel user's access to Reedbush Web Portal manual if any fraud or violation of these conditions and restrictions is detected. Such cancelation is final and undisputable.

Do you accept these conditions and restrictions?

# ポータルサイトでのマニュアル等閲覧(2/2)

Login: g24000

Change Language

prepost予約状況

システムからのお知らせ

ツール

ドキュメント閲覧

パスワードの変更

公開鍵アップロード

ログアウト

トップ Staff Only

## Reedbush System Documents

**お知らせ**

- [Horovod利用方法](#)が追加されました。(2020/04/24)
- [Reedbush システムモジュール変更のお知らせ](#)(2020/04/03)

**Frequently Asked Questions (よくある質問の回答)**

- [FAQ](#) (随時更新)

**Reedbush system**

日本語(Japanese)	更新日	English	update
<a href="#">Reedbush Quick Start Guide</a> (PDF:2MB)	2018/09/21	<a href="#">Reedbush Quick Start Guide</a> (PDF1.3MB)	2018/09/21
<a href="#">Reedbush システム利用手引書(概要・Reedbush-U 編)</a> (PDF:2.1MB)	2018/09/21	<a href="#">Reedbush Supercomputer System Instruction Manual</a> (PDF:4.2MB)	2019/04/01
<a href="#">Reedbush システム利用手引書(Reedbush-H 編)</a> (PDF:0.6MB)	2018/11/30	<a href="#">Reedbush Supercomputer System Instruction Manual(Reedbush-H)</a> (PDF:1.1MB)	2019/04/01
<a href="#">Reedbush システム利用手引書(Reedbush-L 編)</a> (PDF: 816KB)	2018/04/11	<a href="#">Reedbush Supercomputer System Instruction Manual(Reedbush-L)</a> (PDF:934KB)	2019/04/01
<a href="#">Reedbushチューニングガイド</a> (PDF:2.2MB)	2017/03/01		
<a href="#">Reedbushチューニングガイド性能評価ツール編</a> (PDF:11MB)	2016/10/24		
<a href="#">Reedbushチューニングガイド性能評価ツール編(GPU)</a> (PDF:3.6MB)	2017/03/01		
<a href="#">Reedbushグループ管理者機能の手引き</a> (PDF:1.1MB)	2016/11/25		

**MPI Library**

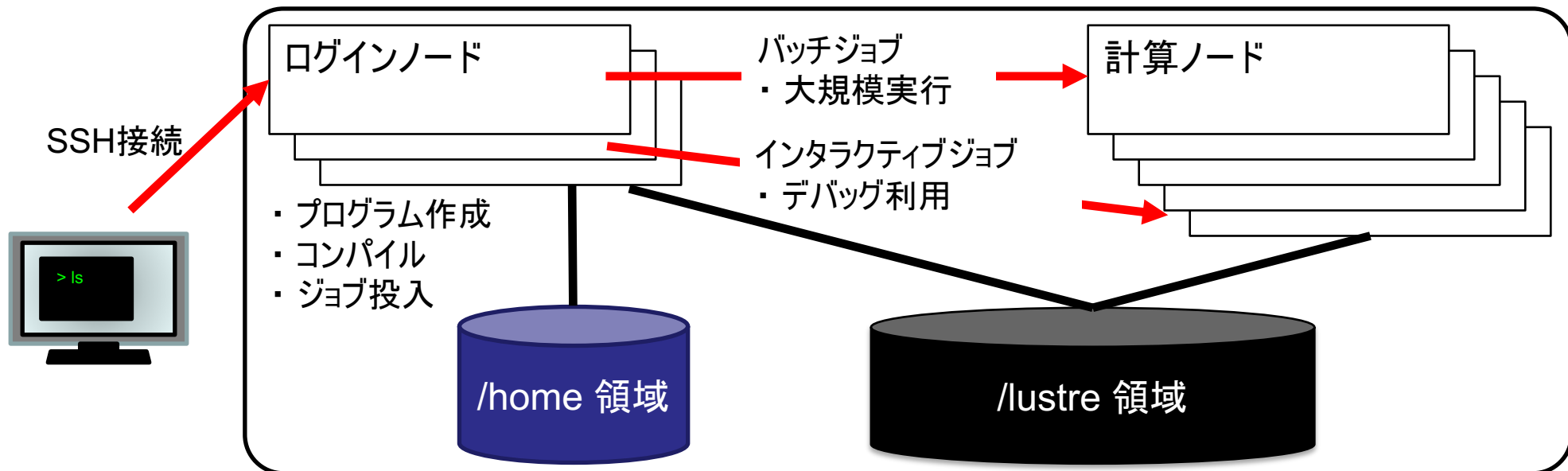
日本語(Japanese)	更新日	English	update	update notes
---------------	-----	---------	--------	--------------



# ログインノードと計算ノード

Reedbush は主にログインノードと計算ノードから構成されています。

- ログインノード: プログラム作成、コンパイル、ジョブ投入
- 計算ノード: ユーザプログラムを実行
  - 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。スパコン環境で一般的。実行処理をスクリプトで指示。長時間、大規模実行。
  - インタラクティブジョブ実行: PCでの実行のように、コマンド入力に対話的に実行する方法。デバッグ用、短時間、数ノード実行。
- /home はログインノードからのみ参照できる。/lustre はログインノードと計算ノード両方から参照できる(詳細は後述)。



# PCから Reedbush へログイン

```
$ ssh t00XYZ@reedbush.cc.u-tokyo.ac.jp  
Enter passphrase for key '/home/user/.ssh/id_rsa:
```

Your Passphrase

Return

1. `ssh t00XYZ@reedbush.cc.u-tokyo.ac.jp <Return>`
2. **鍵生成時に打ち込んだPassphrase** <Return>

- X を飛ばす場合

```
$ ssh -Y t00XYZ@reedbush.cc.u-tokyo.ac.jp
```

X を飛ばす場合には、-Y をつけてください。

# ディレクトリについて(home と lustre)

- ログイン時のディレクトリ(/home/gt00/txxxxx)には**ログインに必要なファイルのみ**を置く。
- プログラム作成や実行などに必要なファイルやライブラリは **/lustre** 以下のディレクトリ(/lustre/gt00/txxxxx)に置く。
- **/home** は計算ノードからは参照できない。
- /lustre は、ログインノードと計算ノード両方から参照できる。
- cdw コマンドで Lustreファイルシステムへ移動できる。

```
$ pwd
/home/t00XYZ
$ cdw      (= cd /lustre/gt00/t00XYZ ワークディレクトリへのショートカットコマンド)
$ pwd
/lustre/gt00/t00XYZ
$ cd
$ pwd
/home/t00XYZ
```

# コンパイルおよび実行のための環境準備

- コンパイルおよび実行のための環境を準備するために `module` コマンドを使用する。これによって様々な環境を簡単に切り替えて使用できる。

**\$ module load <module\_name>**

モジュール名 **<module\_name>** のモジュールをロードして環境を準備。  
。環境変数PATHなどが設定される。

**\$ module avail**

使用可能なモジュール一覧を表示する。

**\$ module list**

使用中のモジュールを表示する。

# モジュールの切り替え

- PGIコンパイラ(OpenACCやCUDA Fortran)を使う場合  
\$ module load pgi/19.10
- CUDA開発環境を使う場合  
\$ module load cuda10
- Intelコンパイラを使う場合  
\$ module load intel
- MPIを使う場合  
\$ module load openmpi/4.0.2/intel  
\$ module load mvapich2/2.3.3/pgi  
など。適切な組み合わせやオプションなどは、「ドキュメント閲覧」の
  - Reedbush システムモジュール変更のお知らせ
  - MPI対応表2020年4月
  - MVAPICH2 GDRを参照してください。
- Deep Learningフレームワーク(例:PyTorch)  
\$ module load pytorch/1.4.0  
など。その他のフレームワークの使い方は、「ドキュメント閲覧」の
  - Deep Learningを参照してください。
- ジョブ実行時にもコンパイル時と同じモジュールを load します。

# Reedbushからファイルをコピー

#Reedbush にログインしていない自分の端末のターミナルで以下を実行

```
$ scp t00XYZ@reedbush.cc.u-tokyo.ac.jp:/lustre/gt00/t00XYZ/ファイル名 ./
```

/lustre/gt00/t00XYZ下にあるファイルを、PC上のCurrent Directory下にコピーする。ファイルでなくフォルダをコピーする場合は、`scp -r` とする。

# Reedbushへファイルをコピー

#Reedbush にログインしていない自分の端末のターミナルで以下を実行

```
$ scp ./ファイル名 t00XYZ@reedbush.cc.u-tokyo.ac.jp:/lustre/gt00/t00XYZ/
```

PC上のCurrent Directory下にあるファイルを /lustre/gt00/t00XYZ 下にコピーする。ファイルでなくフォルダをコピーする場合は、`scp -r` とする。

# 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
- FileZilla
  - <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)

# ライブラリのインストールなど

- 個人利用するライブラリなどは、計算ノードから参照できるように **/lustre** 以下のディレクトリ (**/lustre/gt00/txxxxx**) にインストールしてください。
- コンパイルに必要なシステムにインストール済みのライブラリは事前に module load してください。プログラム実行時にもそれらを module load する必要があります。
- ライブラリはログインノードでコンパイルして構いません。ただし、ログインノードは他ユーザと共有して利用されているため、コンパイルやファイル表示・編集などを除いてプログラムの実行は禁止されていますので、ご注意ください。
- Python利用者で必要なモジュールを PyPIでインストールする (pip コマンドを使う) 場合は、事前に必要なモジュールを module load して、いくつかオプションを与えて pip install する必要があります。詳細は下記などを参照してください。
  - <https://www.cc.u-tokyo.ac.jp/public/VOL21/No4/11.201907python.pdf>



# 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のため、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

1. 自分のパソコンへのSSH環境の準備
2. Reedbushスパコンへのログイン
3. Reedbush利用の準備
4. コミュニケーションツールの準備

# コミュニケーションツールの準備

- Zoom
  - 事前の登録をお願いします。別途メールにてご登録方法についてご連絡致します。
- Slack
  - ミニキャンプ中は、メンターがSlackによる質問対応を行います。
  - 申し込み時にご登録されたメールアドレスをSlackへ登録致します。登録するメールアドレスを変更または追加する場合は、事務局へご連絡ください。