**\Orchestrating** a brighter world **\EC** 

国立大学法人 大阪大学 サイバーメディアセンター 御中

# 高性能計算・データ分析基盤システム SQUID 利用者説明会

2021年05月13日 日本電気株式会社 第一官公ソリューション事業部

# **Orchestrating** a brighter world

NECは、安全・安心・公平・効率という 社会価値を創造し、 誰もが人間性を十分に発揮できる 持続可能な社会の実現を目指します。

# 目 次

- 1. SQUIDの概要
- 2. SQUIDの構成
  - 1. 3つの計算環境
  - 2. 3つのストレージ
  - 3. 3つのフロントエンド
- 3. SQUIDの利用方法
- 4. プログラムの開発、実行
- 5. データアクセス

# 1. SQUIDの概要



### 1. SQUIDの概要

# SQUID

Supercomputer for Quest to Unsolved Interdisciplinary Datascience

#### SOUID

weiなジョジェクトは、全国の国・公・私立大学等の研究者の学術研究のため Jusolの利用を目的とし、計算機科学を応用した研究領域において、特に大規模 htterdicに科学技術計算の需要に対応する、最高水準の計算およびデータ分析基盤 を提供するためのスーパーコンピュータシステムを導入するものである。



### 1. SQUIDの概要





# SQUIDの特徴



Intel 最新鋭プロセッサ(IceLake)を含む総演算理論性能 16.21PFLOPS となる計算資源の拡充

高性能計算環境を支えるストレージ領域の拡充(21.2PB) 相互接続として、Infiniband HDRによるDragonfly+トポロジー

#### HPCの利便性向上に向けた、 コンテナサポート、及びモジュール環境の整備

HPDAの利用者向けの対話型環境の強化 データ転送を用意にする多様なデータアクセス手段のサポート

2段階認証をはじめとするセキュリティの強化



# 2. SQUIDの構成



### 2.1.3つの計算環境

### 汎用CPU/ベクトル/GPGPUの3種の計算環境をご利用いただけます。







	汎用CPU計算環境	ベクトル計算環境	GPGPU計算環境
用途	一般的な計算用途。 大規模並列実行用	ベクトルエンジンによる 高メモリ帯域ベクトル演算用	GPGPU加速器による 高速演算計算用
モデル名	NEC LX 103Bj-8	NEC SX-Aurora TSUBASA B401-8	NEC LX 106Rj-4G
ノード数	1,520ノード	288 VE	42ノード
演算器	Intel Xeon Platinum 8368 (2.4GHz/28Core)x2	Type 20A Vector Engine (3.07 TFlops,1.53TB/s) x8	Intel Xeon Platinum 8368 (2.4GHz/28Core)x2
加速器	-	-	NVIDIA A100 x8
メモリ容量	256GiB	48GiB(HBM2)	512GiB
相互結合網	InfiniBand HDRx1	Infiniband HDR x2	InfiniBand HDR100x4



### 2.1.3つの計算環境 - 汎用CPU計算環境

汎用CPU計算環境の CPUノードは、ブレードシステムを採用し、1,520ノードが、ラックあたり4シャーシ、シャーシあたり19ノードで構成されます。



### 2.1.3つの計算環境 - 汎用CPU計算環境

### 汎用CPU計算環境 - CPUノード スペック

項目		構成	
総ノード	数	1,520台	
サーバ 構成	プロセッサ	Intel Xeon Platinum 8368 (2.4 GHz/38core) x2	
	メモリ構成	256 GiB (16GiB DDR4-3200 RDIMM x16)	
	ハードディ スク	240GB SATA SSD x1	Dual-port 25G/10G NIC
	インタ フェース	InfiniBand HDR x1、25/10GbE x2、BMCx1	1x 200G IB
ソフト ウェア	OS	CentOS 8.2 (64bit)	
環境	コンパイラ	Intel Compiler	2x Intel® Xeon Platinum 8368
	MPI	Intel MPI	



### 2.1.3つの計算環境 - ベクトル計算環境

ベクトル計算環境は、旧システムの後継製品である NEC SX-Aurora TSUBASA を採用し、1VH(Vector Host)あたり8VE(Vector Engine)で、 288 VEの構成です。





### 2.1.3つの計算環境 - ベクトル計算環境

### ベクトル計算環境 - ベクトルノード スペック

#### ベクトルエンジン(VE)

	項目	構成		
総VE数		288 VE ( 1VH当たり8VE )		
モデル名		Type 20A		
	演算性能(倍精度)	307 GFlops / 10core		
VE構成	メモリ構成	48 GiB (HBM2)		
	メモリ帯域	1.53 TB/s		
	OS	VEOS 2.5.0		
ソノトワェ ア環境	コンパイラ	NEC SDK for Vector Engine		
	MPI	NEC MPI		



#### ▲ ベクトルホスト(VH)

	項目	構成		
VH数		36VH		
\//.i=#	プロセッサ	AMD EPYC 7402P Processor(2.8GHz/24core) x1		
	メモリ構成	128GiB (DDR4-3200 16GiB x8)		
VIIDH/L	ストレージ	960GB SATA SSD x1		
	インタフェース	InfiniBand HDR x2, 1000Base-T x1, BMC		
ソフトウェ ア環境	OS	CentOS 8.2 (64bit)		



**\Orchestrating** a brighter world **NEC** 

### 2.1.3つの計算環境 - GPGPU計算環境

| GPGPU計算環境は、NVIDIA A100 GPUを搭載したGPUノード、42ノー ドで構成されます。1ノードあたり、NVIDIA A100 GPUが8基搭載され、 NVSwitch 経由で相互接続されます。





### 2.1.3つの計算環境 - GPGPU計算環境

### GPGPU計算環境 - GPUノード スペック

項目		構成	
総ノード	ॐ数	42ノード	
サーバ 構成	プロセッサ	Intel Xeon Platinum 8368 (2.4 GHz/38core) x2	
	メモリ構成	512 GiB (32GiB DDR4-3200 ECC RDIMM x16)	
	ハードディ スク	240GB SATA SSD x1	
	インタ フェース	InfiniBand HDR100x4, 1000BASE-T x1, BMC x1	P
	GPGPU	NVIDIA A100 (SXM4) x8	
ソフト ウェア	OS	CentOS 8.2 (64bit)	
環境	コンパイラ	NVIDIA HPC SDK	_
	MPI	OpenMPI	





### 2.1.3つの計算環境 - 相互接続網

- 計算環境はInfiniband HDRの相互接続網で接続されます
- ●汎用CPU計算環境:3つのNWグループによるDragonfly+トポロジー
- ●ベクトル/GPGPU計算環境:Fat Tree トポロジー





### 3つのストレージ領域がご利用いただけます。







	SSDストレージ	HDDストレージ	アーカイブストレージ
特徴	All Flashの環境による 超高速IO領域	高速かつ大容量の データ格納領域	複製や暗号化機能を有 する柔軟なオブジェク トストレージ領域
容量	(追加購入)	home:10GiB work:5TiB + 追加購入	別途利用申請
ファイル システム	DDN ExaScaler (Lustre)	DDN ExaScaler (Lustre)	Cloudian HyperStore
総容量	1.2 PB	20 PB	840 TB(物理容量)
ディスク装置	15.36TB NVMe SSD	16TB 7,200rpm NL-SAS	960GB SSD(メタデータ) 10TB SAS
ハードウェア	DDN ES400NVX x5	DDN ES7990X x10	Cloudian HyperStore Appliance 1610 x7

※計算環境から直接利用が可能なのは、SSDならびにHDD です。





### 2.3.3つのフロントエンド

### 3つのフロントエンドがご利用いただけます。



	HPCフロントエンド	HPDAフロントエンド	セキュアフロントエンド
特徴	HPCアプリケーション 開発向け環境 可視化処理環境、バッ チジョブ投入環境	HPDA向けNoteBook環境 バッチジョブ投入環境	仮想マシンによる専用 フロントエンド環境 バッチジョブ投入環境
ノード数	4 ノード	4 ノード	仮想マシン
アプリケー ション	NICE DCV Server	Jupyter NoteBook	_
その他	NVIDIA Quadro RTX6000x1	512GiB メモリ(2倍)	別途利用申請



# 3. SQUID の 利用方法



### ログインはSSHによる2段階認証が必要です

### ログイン先

サーバ	ホスト名
HPCフロントエンド	squidhpc.hpc.cmc.osaka-u.ac.jp
HPDAフロントエンド	squidhpda.hpc.cmc.osaka-u.ac.jp

### 2段階認証アプリ ※ 2段階認証には**事前にアプリのインストールが必要**です

OS	アプリ	入手元
Android	Google Authenticator	Google Play Store
iOS	Google Authenticator	Apple App Store
Windows	WinAuth	https://winauth.github.io/winauth/down load.html
macOS	Step Two	Apple App Store
		※利用可能アプリケーションの一例です



### 初回ログイン

 2段階認証アプリの入手 お持ちのスマートフォンやPCで、入手可能なアプリをインストールく ださい。説明はGoogle Authenticator の例です。

 SSHアクセス お手元のターミナルソフトからSSHアクセスします。
 例:HPCフロントエンドの場合

\$ ssh -1 (利用者番号) squidhpc.hpc.cmc.osaka-u.ac.jp Password: (パスワード)

✓ ユーザ名とパスワードは、申請後に通知される利用者番号と 利用者管理システムのパスワードを入力します。



### 初回ログイン

#### ③ 2段階認証アプリの入手 初回ログイン時、2段階認証のセットアップ画面が表示されます。

#### Initiallize google-authenticator Warning: pasting the following URL info your browser exposes the OTP secret to Google:

https://www.google.com/chart?chs=200\*200&chld=M|0&cht=qr&otpauth://totp/user1@squidhpc.hpc.cmc.osak a-u.ac.jp%3Fsecret%3DDXXXXXXXCLI%26issuer%3Dsquidhpc.hpc.cmc.osaka-u.ac.jp



 $\otimes$  QR $\Box$  –  $\ltimes$ 

#### ✓ ウィンドウサイズによってはQRコードが崩れます。フォントサイズを 調整するか、記載のURL、シークレットキーをお使いください。



### 初回ログイン



⑤ 読込完了

完了するとGoogle Authenticatorに、 登録され、ワンタイムパスワードが 発行されます。





### 初回ログイン

#### ⑥ ログアウト ターミナルソフト側は、「Enter code from app」と聞かれていますの で、「-1」を入力し、一度ターミナルを終了します。



© NEC Corporation 2021

Your new secret key is: XXXXXXXXXX Enter code from app (-1 to skip): -1 Code confirmation skipped Your emergency scratch codes are: Completed to initialized google-authenticator After logout, and please relogin. It will be authenticated 'OTP' by Google Authenticator. Hit [Enter] key

✓ 2段階認証の初期登録方法は以上です。



### 2回目以降のログイン



- ✓ ユーザ名とパスワードは、申請後に通知される利用者番号と 利用者管理システムのパスワードを入力します。
- ✓ アプリトで確認したワンタイムパスワードをターミナルに入力します。

ログイン手順は以上です。



### ファイルシステム

ファイルシステムとして、SSDストレージおよびHDDストレージに直接 アクセス可能です。ご利用可能なディスク領域およびそのクォータ制限は 以下の通りです。

スト レージ	ファイル システム	領域名	パス	サイズ クォータ	備考
		ホーム領域	/sqfs/home/(利用者番号)	10GiB	ブラウザアクセ ス領域を含む
HDD	EXAScaler	拡張領域	/sqfs/work/ (グループ名)/(利用者番号)	5TiB(+)	追加購入可
	(Lustre)		/sqfs/s3/(UUID)	5112(1)	S3アクセス用
SSD		高速領域	/sqfs/ssd/ (グループ名)/(利用者番号)	0B(+)	追加購入可

### ファイルシステムの特徴

ホーム領域

- ●ユーザ登録時に与えられる最低限の領域です。初期容量として、10GiBが割り当 てられます。
- ●WEBブラウザから、アクセスするための領域を含みます。

拡張領域

- ●初期容量として5TiBが割り当てられており、申請によって所属グループ毎に容 量を追加購入可能です。
- ●SQUID外部からS3 APIでアクセスするための領域が、別パスで用意されます。

高速領域

- ●SSDストレージによる超高速I/Oが可能です。
- ●初期容量の割り当てはありませんが、申請によって所属グループ毎に容量を追加 購入が可能です。



#### 利用状況の確認 (usage\_view)

計算環境は実行時間と消費係数に基づくポイントを消費して利用します。 ストレージは購入額に応じたクォータ制限の範囲で利用します。 これらの利用状況を確認するには、usage\_viewコマンドを利用します。

<b>\$ usage</b> [Group su	_ <b>view</b> + Group: ummary]	G012345 +-			
	SQUID points	HDD (G i B)	SSD (GiB)		グループの利用状況
usage limit remain rate(%)	0. 0 11000. 0 11000. 0 0. 0	87. 9 46080. 0 45992. 1 0. 2	0. 0 20480. 0 20480. 0 0. 0		<ul> <li>・ポイント消費</li> <li>・HDD領域の利用量</li> <li>・SSD領域の利用量</li> </ul>
[Detail]					
	SQUID points		Home(GiB)	HDD (GiB)	SSD (GiB)



### 利用状況の確認 (usage\_view)

-						
[DETAIL]						
	SQUID POINT	S	HOI	ME(GIB)	HDD (GIB)	SSD (GIB)
USER001	0.	 0    0. 0  /	10.0	/ 10.0	0.0	0.0
USER002	0.	0 0.0 /	10.0	/ 10.0	0.0	0.0
USER003	0.	0 0.0 /	10.0	/ 10.0	0.0	0.0
Node-Hour:	S]				/	グループ禾
		USA	GE	AVAILABI	E*	
 PU	NODE	 0.	00	9090.	90	
PU	NODE	0.	00	8184.	52	
ECTOR	NODE	0.	00	8461.	53	
URE CPU	NODE	0.	00	6547.	61 🖡	†昇ノート
GPU	NODE	0.	00	0.	00	δび残りσ
I CPU	NODE	0.	00	6547.	61	
GPU	NODE	0.	00	0.	00	
ECURE CPU	NODE	0.	00	6666.	66	
GPU	NODE	0.	00	6666.	66	
VEC	TOR NODE	0.	00	6666.	66 🤳	

環境設定(Environment Module)

# コンパイラやライブラリの環境変数は、Environment Moduleで制 御します。module コマンドを使用して、開発環境を準備します



コマンド	説明
module avail	利用可能な開発環境/アプリの一覧表示
module list	読み込み済みのモジュールの一覧表示
module switch [file1] [file2]	モジュールの入れ替え(file1 → file2)
module load [file]	モジュールの読み込み
module unload [file]	モジュールの読み込み解除
module purge	ロード済みの全モジュールの解除
module show [file]	モジュールの詳細表示



#### 基本環境と推奨環境

# モジュールは標準的な開発環境をまとめた基本環境を用意しています。基本環境の中には、各計算環境の利用に適した推奨環境を用意しています。 基本環境

モジュール名	推奨環境	内容
BaseCPU/2021	汎用CPU	汎用CPUノード向けプログラム開発の推奨環境
BaseVEC/2021	ベクトル	ベクトルノード向けプログラム開発の推奨環境
BaseGPU/2021	GPGPU	GPUノード向けプログラム開発の推奨環境
BaseGCC/2021		GCCを利用する際の開発環境
BasePy/2021		Python 向けのプログラム開発環境
BaseR/2021		R言語向けのプログラム開発環境
BaseJulia/2021		Julia言語向けのプログラム開発環境
BaseJDK/2021		JAVA言語向けのプログラム開発環境
BaseApp/2021		ISV及びOSSアプリケーション向けのベース環境
	<b>モジュール名</b> BaseCPU/2021 BaseVEC/2021 BaseGPU/2021 BaseGPU/2021 BaseGCC/2021 BasePy/2021 BasePy/2021 BaseJulia/2021 BaseJDK/2021 BaseApp/2021	モジュール名推奨環境BaseCPU/2021汎用CPUBaseVEC/2021ベクトルBaseGPU/2021GPGPUBaseGCC/2021GPGPUBasePy/2021BaseR/2021BaseJulia/2021BaseJDK/2021BaseApp/2021

具体的な利用方法は、次節にてご説明します。



### 3.2. フロントエンドの利用 (HPC)

#### リモート可視化(NICE DCV)環境の利用

HPCフロントエンドでは、グラフィックアクセラレータを利用した、可視化環境 (NICE DCV)の利用が可能です。

\$ create-session --type=virtual セッション名 (仮想セッションを作成します) \$ dcv list-sessions (作成した仮想セッションを確認します)





### 3.2. フロントエンドの利用 (HPDA)

#### Jupyter Notebook 環境の利用

データ分析やインタラクティブな言語開発として、Jupyter Notebook をHPDA フロントエンド上でご利用いただけます。

<pre>\$ singularity pull jupyter.sif oras://cntm:5000/master_image/jupyter:1.0 \$ run_jupyter_container.sh</pre>
JUPYTER NOTEBOOK UKL   NTTPS:// SQUIDNPDAI. NPC. CMC. OSAKA-U. AC. JP: 10125
INFO: Converting SIF file to temporary sandbox
INFU: Instance started successfully
iupyter login token : 3c4b9f3fb424b9059d1076c559adff837f9fc9ee8ce81842

pc.cmc.osaka-u.ac.ip:10125	l.hpc.cmc.osaka-u.ac.jp:10125/login?next=%2Ftree%3F	← → C 介 🔒 squidhpda1.hpc.cmc.osi
ログイン	Ç jupyter	Ç jupyter
https://squidhpdat.hpc.cmc.osaka-uac.jpt10125 ユーブーを /(スフード ログイン キャンセル	パスワードまたはトークン ログイン Token authentication is enabled If no password has been configured, you need to open the notebook server with its login token in the URL, or passe it above. This requirement will be lifted if you shalks a password? The command: Jupyter notebook list will show you the URLs of running servers with their tokens, which you can copy and paste into your browser. For example: Qurrently running servers:	ファイル <u>東行中 クラスタ</u> アクションを実行する為のアイテムを選





# 4. プログラムの開発、実行



# コンパイラ、MPI

### 標準的な開発環境は、計算環境別にモジュールが用意されています。

計算環境	推奨 モジュール名	コンパイラ	MPI
汎用CPU	BaseCPU	Intel Parallell Studio	Intel MPI
ベクトル	BaseVEC	NEC SDK for VE	NEC MPI
GPGPU	BaseGPU	NVIDIA HPC SDK CUDA	OpenMPI

\$ module avail					
BaseCPU/2021(default) BasePy/2021	/system/app BaseVEC/2021 BaseR/2021	s/env/base BaseGPU/2021 BaseApp/2021	BaseGCC/2021		
\$ module load BaseCPU/2	021	汎用CPU向け	推奨環境を読み込み		
<b>\$ ifort</b> -o sample.out	sample.f90	Intel Compile	erによるコンパイル		
<pre>\$ mpiifort -o sample_mp</pre>	i sample_mpi.f90	Intel MPIを用	りいたコンパイル		



### コンテナの利用

Singularity 3.7.2 が実装されており、コンテナを用いたプログラム実行環境を整備することが可能です。
 フロントエンドトで、外部公開されているコンテナイメージをグロンロード、カフタフ

フロントエンド上で、外部公開されているコンテナイメージをダウンロード、カスタマ イズした上で、バッチジョブとしてコンテナを実行することが可能です。

例) 外部イメージをsandboxとしてhomeに展開する



### コンテナの利用 (続き)

Singularity 3.7.2 が実装されており、コンテナを用いたプログラム実行環境を整備することが可能です。
 フロントエンド上で、外部公開されているコンテナイメージをダウンロード、カスタマイズした上で、バッチジョブとしてコンテナを実行することが可能です。

#### 例) sandbox から、イメージファイルをビルドする

- \$ singularity build -f <sifファイル名> <sandbox名>
- \$ singularity build -f test.sif test

#### <u>例) コンテナイメージを実行する</u>

- \$ singularity exec (コンテナ名) (<u>コンテナ内の</u>プログラムパス)
- \$ singularity exec test.sif /path/to/myprog.out

※ご自身の環境で作成したコンテナイメージを持ち込んでいただくことも可能です。※バッチジョブとしての実行方法は後述します。



ジョブの実行

#### ジョブ管理システム (NQSV)

#### 計算環境の利用は、ジョブ管理システムにジョブを投入して利用します。 混雑状況をみて、システム側で実行時刻とサーバを決定します。



#### ●<u>バッチジョブ</u>:実行内容を記述したジョブスクリプトを投入します。非対話的に 実行されます。

●<br />
●
インタラクティブジョブ:デバッグ用などに、対話的な実行環境を要求します。



### バッチリクエストの実行

#### qsub コマンドにて、バッチジョブを投入します。 投入にはグループの指定が必要です。

\$ qsub 「オプション] [ジョブスクリプトファイル名]

qsubコマンドを実行すると、リクエストIDが採番され、下記のように標準出力 に表示されます。

Request **1182.sqd** submitted to queue: small

ジョブスクリプト例





ジョブの実行

#### ジョブクラスと投入キュー

ジョブはいくつかジョブクラスに分類されます。 ジョブクラスは、ジョブ管理システム上のキューに対応しており、利用者は キューにジョブを投入することで計算環境の利用が可能です。

一部のキューを除いて基本的には、計算環境はノード単位での管理となり、 ノード占有、ノード共有でポイント消費が変わります。

投入キュー名	種類	用途	備考
SQUID	バッチ	標準利用のためのジョブクラス	1ノード(VE)あたり1 ジョブ利用
SQUID-S	バッチ	他のジョブとのノード内共有を許容し、 ポイント消費を抑えたい方向け	
SQUID-H	バッチ	ポイント消費を多くし <mark>高優先度ジョブを</mark> 投入し、待ち時間を短縮したい方向け	
SQUID-R	バッチ	NW帯域が狭い経路の利用を許容して、 実行待ち時間を短縮したい方向け	汎用CPUのみ
DBG	バッチ	デバッグ用の短時間の利用向け	キュー優先度高
INTX	インタラク ティブ	デバッグ用の対話型利用向け	キュー優先度高





### 

種別	ジョブ クラス	利用可能 経過時間	利用可能 最大Core数	利用可能 メモリ	同時利用可能 ノード数	備考
	SQUID	24時間	38,912core (76c/ノード)	124TiB (248GB/ノード)	512	ノード内は占有 利用
	SQUID-R 24時間()	38,912core (76c/ノード)	124TiB (248GB/ノード)	512		
共有利田	SQUID-H	24時間	38,912core (76c/ノード)	124TiB (248GB/ノード)	512	
CI (C. IV	SQUID-S	24時間	38core	124GiB	1	
	DBG	10分	152core (76c/ノード)	496GiB (248GB/ノード)	2	デバッグ用
	INTC	10分	152core (76c/ノード)	496GiB (248GB/ノード)	2	インタラクティ ブ利用
占有 利用	mySQUID	無制限	76core × 占有ノード数	248GiB × 占有ノード数	占有数	別途利用申請





### ジョブクラス (ベクトル計算環境向け)

種別	ジョブ クラス	利用可能 経過時間	利用可能 最大Core数	利用可能 メモリ	同時利用可能 VE数	備考
S	SQUID	24時間	2,560core (10c/VE)	12TiB (48GB/VE)	256	
	SQUID-H	24時間	2,560core (10c/VE)	12TiB (48GB/VE)	256	
共有 利用	SQUID-S	24時間	40core	192GiB	4	
	DBG	10分	20core (10c/VE)	96GiB (48GB/VE)	2	デバッグ用
	INTV	10分	20core (10c/VE)	96GiB (48GB/VE)	2	インタラクティ ブ利用
占有 利用	mySQUID	無制限	10core × 占有VE数	48GiB × 占有VE数	占有数	別途利用申請





### ジョブクラス (GPGPU計算環境向け)

種別	ジョブ クラス	利用可能 経過時間	利用可能 最大Core数	利用可能 メモリ	同時利用可能 ノード数	備考
	SQUID	24時間	2,432core (76c/ノード)	15.75TiB (504GB/ノード)	32	
	SQUID-H 24時間	24時間	2,432core (76c/ノード)	15.75TiB (504GB/ノード)	32	
共有 利用	SQUID-S	24時間	38core	252GiB	1	
	DBG	10分	152core (76c/ノード)	1,008GiB (504GB/VE)	2	デバッグ用
	INTG	10分	152core (76c/ノード)	1,008GiB (504GB/VE)	2	インタラクティ ブ利用
占有 利用	mySQUID	無制限	76core × 占有ノード数	504GiB × 占有ノード数	占有数	別途利用申請

CASE1:汎用CPU計算環境でスレッド並列ジョブを実行する

#### スレッド並列 1ノード で76スレッド実行の例

#### 青字はコメントです

1 2	#!/bin/bash # goub_option	
2	#PBS -a SQUID	# 投入キューに SQUID を指定
4	#PBSgroup=G01234	# ポイント消費先のグループ
5	#PBS -I elapstim_req=00:30:00	# 実行時間を指定
6	<b>#PBS</b> -v OMP_NUM_THREADS=76	# スレッド数にCPUコア数を指定
/ 8	# Program execution	_
9		
10	module load BaseCPU/2021	# コンパイル時に module load していたものを記載
11 12	module load xxx/xxx	
13	cd \$PBS_0_WORKDIR	# qsub したディレクトリに移動
14	. /a. out	# 1proc 76thread の実行
15		
16		
/ 10		
10 10		
19		



<u>CASE2</u>:汎用CPU計算環境でMPI並列ジョブを実行する

#### MPI並列(Flat MPI) 4ノード304プロセスの実行例

#### 青字はコメントです

```
#!/bin/bash
1
2
   #----- gsub option -----
3
   #PBS -q SQUID
                                           # 投入キューに SQUID を指定
   #PBS ---group=G01234
                                           # ポイント消費先のグループ
4
5
   #PBS -1 elapstim_req=00:30:00
                                           # 実行時間を指定
                                           #実行ノード数を指定
6
   #PBS -b 4
7
   #PBS -T intmpi
                                           # MPIに Intel MPI を指定
8
9
    #----- Program execution ------
10
   module load BaseCPU/2021
                                           # コンパイル時に module load していたものを記載
11
12
   module load xxx/xxx
13
   cd $PBS_0_WORKDIR # qsub したディレクトリに移動
mpirun $ {NQSV_MPIOPTS} -np 304 ./a.out # Intel MPI の引数に NQSV_MPIOPTSを引き渡し
14
15
16
17
18
19
```



## <u>CASE3</u>:ベクトル計算環境でスレッド並列ジョブを実行する

#### スレッド並列 1VEで10スレッド実行の例

青字はコメントです

1	#!/bin/bash	
2	# qsub option	
3	#PBS –q <mark>SQUID–S</mark>	# 投入キューにSQUID-Sを指定 ※SQUIDは8VEからのため
4	#PBSgroup= <mark>G01234</mark>	# ポイント消費先のグループ
5	#PBS -l elapstim_req=00:30:00	# 実行時間を指定
6	#PBSvenode=1	# VE数を指定
7	#PBS -v OMP_NUM_THREADS=10	# VEのコア数 10 を指定
8		
9	# Program execution	
10		
11	module load <mark>BaseVEC/2021</mark>	# コンパイル時に module load していたものを記載
12	module load xxx/xxx	
13		
14	cd \$PBS_0_WORKDIR	# qsub したディレクトリに移動
15	. /a. out	# 1proc 10thread/VE x 1VEの実行
16		
17		
18		
19		



### CASE4:ベクトル計算環境でMPI並列ジョブを実行する

#### MPI並列(Flat MPI) 40VE 400プロセスの実行例

#### 青字はコメントです

#!/ DTH/ DASH	
# qsub option	
#PBS -q <mark>SQUID</mark>	# 投入キューにSQUIDを指定
#PBSgroup= <mark>G01234</mark>	# ポイント消費先のグループ
#PBS -  elapstim_req=00:30:00	# 実行時間を指定
#PBSvenode= <mark>40</mark>	# 総VE数を指定 ※ 8VE/VH x 5VH の 指定
#PBS -T necmpi	# MPIに NEC MPI を指定
# Program execution	-
module load BaseVEC/2021	# コンパイル時に module load していたものを記載
module load xxx/xxx	
cd \$PBS_0_WORKDIR	# qsub したディレクトリに移動
mpirun - <mark>np 400</mark> ./a.out	# 10proc/VE x 40VE の実行
*######## # nn cn	<pre>#//DIN/bash # qsub option #PBS -q SQUID #PBSgroup=G01234 #PBS -1 elapstim_req=00:30:00 #PBSvenode=40 #PBS -T necmpi # Program execution module load BaseVEC/2021 module load BaseVEC/2021 module load xxx/xxx ed \$PBS_0_WORKDIR mpirun -np 400 ./a.out</pre>





<u>CASE5</u>: GPGPU計算環境でスレッド並列ジョブを実行する

#### スレッド並列 1ノード で48スレッド実行の例

#### 青字はコメントです

1	#!/bin/bash	
2	# qsub option	
3	#PBS –q <mark>SQUID</mark>	# 投入キューにSQUIDを指定
4	#PBSgroup= <mark>G01234</mark>	# ポイント消費先のグループ
5	#PBS -  elapstim_req=00:30:00	# 実行時間を指定
6	#PBS -  <mark>gpunum_job=8</mark>	# GPU数の指定
7	#PBS -v OMP_NUM_THREADS=48	# スレッド数の指定
8		
9	# Program execution	
10		
11	module load <mark>BaseGPU/2021</mark>	# コンパイル時に module load していたものを記載
12	module load xxx/xxx	
13		
14	cd \$PBS_0_WORKDIR	# qsub したディレクトリに移動
15	. /a. out	
16		
17		
18		
19		



### <u>CASE6</u>: GPGPU計算環境でMPI並列ジョブを実行する

#### MPI並列 2ノードで4MPIプロセスの例

#### 青字はコメントです

1	#!/bin/bash	
2	# qsub option	
3	#PBS –q <mark>SQUID</mark>	# 投入キューにSQUIDを指定
4	#PBSgroup=G01234	# ポイント消費先のグループ
5	#PBS -  elapstim_req=00:30:00	# 実行時間を指定
6	#PBS -b 2	# 実行ノード数を指定
7	#PBS -  gpunum_job=8	# 1ノードあたりのGPU数を指定
8	#PBS -T openmpi	# MPIに OpenMPI を指定
9	#PBS -v NQSV_MPI_MODULE=BaseGPU/2021	# MPIを読み込むモジュール名を指定
10		
11	# Program execution	
12		
13	module load BaseGPU/2021	# コンパイル時に module load していたものを記載
14	module load xxx/xxx	
15		
16	cd \${PBS 0 WORKDIR}	# qsub したディレクトリに移動
17	mpirun \${NQSV_MPIOPTS} -np 4 -npernode 2	./mpi_prog
18		# OpenMPI の引数に NQSV_MPIOPTSを引き渡し
19		

<u>CASE7</u>: GPGPU計算環境でコンテナジョブを実行する

#### MPI並列 2ノード で 16MPIプロセスのコンテナジョブの例

#### 青字はコメントです

#!/bin/bash	
# qsub option	
#PBS –q <mark>SQUID</mark>	# 投入キューにSQUIDを指定
#PBSgroup= <mark>G01234</mark>	# ポイント消費先のグループ
#PBS -  elapstim_req=00:30:00	# 実行時間を指定
#PBS <mark>-b 2</mark>	# 実行ノード数を指定
#PBS -  <mark>gpunum_job=8</mark>	# 1ノードあたりのGPU数を指定
#PBS -T openmpi	# MPIに OpenMPI を指定
<b>#PBS</b> -v NQSV_MPI_MODULE=BaseGPU/2021	# MPIを読み込むモジュール名を指定
# Program execution	
module load <mark>BaseGPU/2021</mark>	# コンパイル時に module load していたものを記載
module load xxx/xxx	
cd \${PBS_0_WORKDIR}	# qsub したディレクトリに移動
<pre>mpirun \${NQSV_MPIOPTS} -np 16 -npernode</pre>	8 ¥
singularity exec —nv (コンテ	ナ名)(コンテナ内プログラム名)
	# singularity の引数に —nv を指定する
	<pre>#!/bin/bash # qsub option #PBS -q SQUID #PBS -group=G01234 #PBS -l elapstim_req=00:30:00 #PBS -b 2 #PBS -l gpunum_job=8 #PBS -T openmpi #PBS -v NQSV_MPI_MODULE=BaseGPU/2021 # Program execution module load BaseGPU/2021 module load SaseGPU/2021 module load xxx/xxx cd \${PBS_0_WORKDIR} mpirun \${NQSV_MPIOPTS} -np 16 -npernode</pre>







### WEBブラウザを利用したグラフィカルなデータ転送手段

#### ログイン先

サーバ	URL
データ転送ゲートウェイ	https://onionweb.hpc.cmc.osaka-u.ac.jp

#### ブラウザ操作



パスワードをお忘れですか?
デバイスを使ってログインします。

① ユーザ名とパスワードを入力

② SQUIDホーム領域 に

### SQUID内の保存先

しログイン

#### パス

/sqfs/home/(利用者番号)/OnionWeb/



② SQUIDホーム領域 にドラッグアンドドロップ等でデータ転送



領域

ホーム領域



### S3 APIを利用したアプリケーションからのデータ転送手段

#### エンドポイント名 サーバ エンドポイント名 S3DS https://squidgw.hpc.cmc.osaka-u.ac.jp

#### アクセスキーの発行

フロントエンド上で、s3dskey コマンドによりキーを発行します

	ł
accesskey   AKIA7X1KXXXYYYZZZ000   enabled   True   fspaths   None   fsuid   60101:10   secretkey   hagwyutos8TThj3S0v9ahPJMF8TTK2dYFcV9Q   tag   (連番):(ユーザ名):(グループ名)   uuid   ea60f00a60hufaweofapo12813nfawe9506e1	アクセスキー w7T シークレットキー 216 UUID



### S3 API でのデータアクセス

### 外部からのS3 APIでのデータ操作

s3curl を例としてご説明します。設定ファイル(~/.s3curl)



### SQUID内の保存先

領域	パス
拡張領域	/sqfs/s3/(UUID)/(バケット名)/(オブジェクト名)



### OCTOPUSからのデータアクセス

#### OCTOPUS - SQUID 間は、Infiniband 経由でデータ転送可能です



データ保存領域は下記で参照可能なため、フロントエンド上で相互にデー タ転送が可能です

ファイルシステム	ファイルパス
高速領域	/sqfs/ssd
ホーム領域	/sqfs/home
拡張領域	/sqfs/work

ファイルシステム	ファイルパス
ホーム領域	/octfs/home
拡張領域	/octfs/work



# **Orchestrating** a brighter world

