第02回おしながき

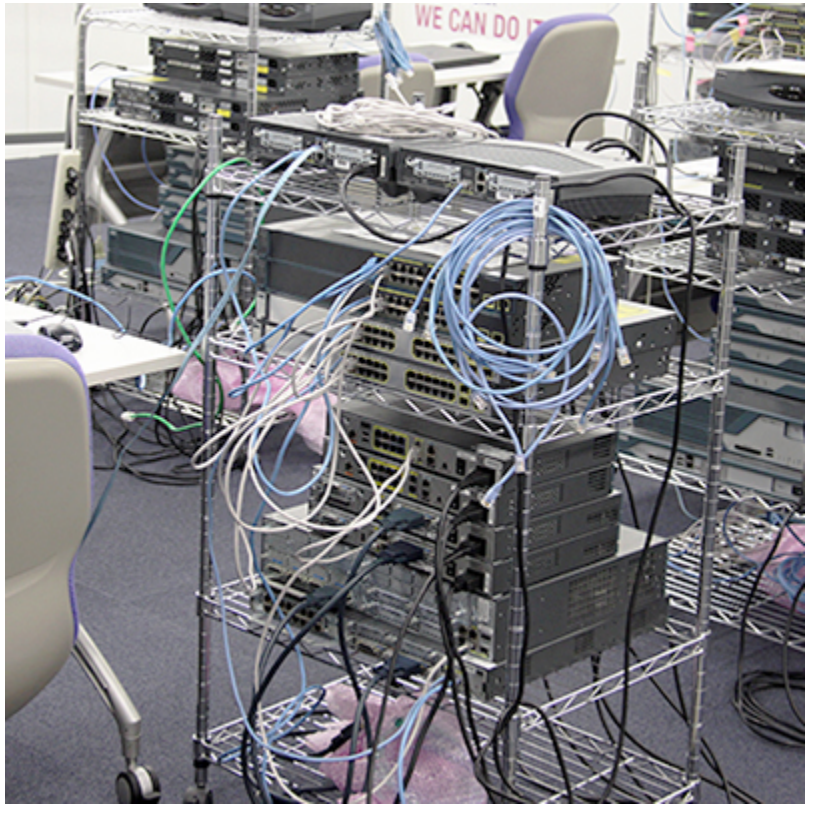
1. おしらせ
    - 登録漏れがあるなら申し出てください
    - インターンシップの御紹介
2. EL (確認テスト)
3. 今回のハイライト
4. 演習 (春学期の復習)
    - EC2のインストール
    - 頭出しと復習を少々（ps、パス）

(脚注1) 授業後半は「春学期の復習ではない人」向けの予備の時間(バッファ)として取ってあると思ってください。 情報システム工学科3年生は早く終わるはずですが、それが正常なので、早めに授業を終えてOKです。 (脚注2) 今回、支援システムColitasは使いたいだけ使うという方針です。 コマンドの使い方は、システムの代わりに、技術同人誌を頼ってください:-)

ELの解説

インターンシップの御紹介

- 実機（Ciscoが12台）でネットワークを組むという世にも珍しい研修。たぶん日本でここだけ -> ポータルを参照。人数によっては築地説あり
- NRIやCTCなどの有名SI企業は、この研修をさせています。
  - 大手SI(システムインテグレータ)
  - NTTデータ
  - NRI(野村総研),CTC(伊藤忠),...,IIJ
  - （富士通,NEC,日立;SIもやってるけど...）
- **「実機が分からないのにクラウドが分かるわけないだろ」**というウチ（H205）の主張は正しい



(脚注1) プロセス (脚注2) マルチプロセスは幻影です。 どういう理屈でそうなっているのか？は、大学院の授業へ移動したので省略

(脚注3) 少し不正確ですが、 ここでは、マルチプロセスの説明をしたいだけなので精密さは無視していきます。

第02回のハイライト

- 第02-05回は「権限」関連の話題を見ていきます。OSの状態を調べるためにコマンドを使います
- 用語：TSS=タイムシェアリングシステム、選択肢がある場合はUnixの用語を使っています
- TSSは**マルチユーザ**
  - コンピュータを同時に複数人で利用できます
  - 当然ユーザを区別する必要があります
  - Unixの一番重要なユーザの区別は「一般ユーザ」か「管理者ユーザ」か？です
    - 演習の**EC2(debian)では、一般ユーザがadmin、管理者がroot**
- TSSは**マルチプロセス**
  - プログラムが実行されている状態をプロセスと呼んでいます
  - 同時に複数のプロセスが実行されています
  - WWWサーバは同時に複数のアクセスに対応しています
    - 1つのWWWアクセスごとに、1つのWWWサーバプロセスがあり、そのプロセスがリクエストに対応しています (脚注3)

(脚注1) プロセス (脚注2) マルチプロセスは幻影です。どういう理屈でそうなっているのか？は、大学院の授業へ移動したので省略

(脚注3) 少し不正確ですが、ここでは、マルチプロセスの説明をしたいだけなので精密さは無視していきます。

例題: プロセスを調べる

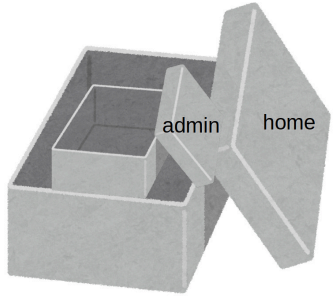
```
$ ps
$ ps u
$ ps ux

$ ps auxww
```

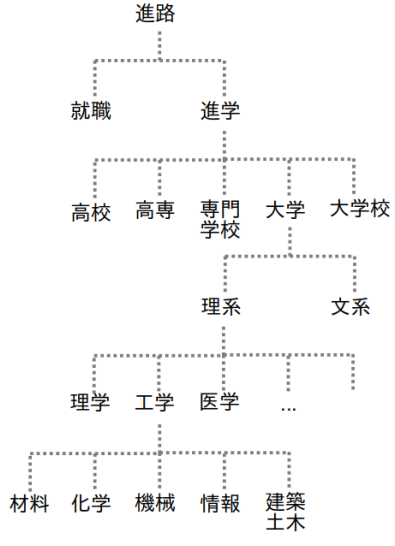
- AWS Academy(vocareum)のターミナルで同じようにやってみよう
- psコマンドは第6回で、もう少し詳しくやります。今日は体験だけです。 なおpsコマンドの詳しい解説は同人誌を参照してください:-)

(脚注) URLは、ポータルの添付資料欄にもあります

復習： /home/admin

- （来週）EC2にログインすると**/home/admin**に出ます
  - 第7回で詳しくやります
  - 例：home箱のなかにadmin箱があって、admin箱の中にwww.pyファイルを入れる…
- 大きい処理単位から考える
  - 5W1H
  - 【復習】 コンピュータネットワーク設計編





(脚注1) 西洋哲学史としては、プラトンのディアイレシスが初出です。 いちばんガチでやっているオペラが「ソピステース」ですが、初出は、その２つ前の作品ということになっています。 (脚注2) ビジネス用語？ではMECE（ミーシー）が似たような用語

演習

- 課題
  - EC2の構築
  - EIPの取得 ... 来週(明日？)からは各自のサーバにドメイン名でアクセス可
    - EIPとEC2の関連付けを行う
    - 取得後EIPの申請をしてください (portal 添付資料欄の URL)
- テキスト
  - 情報開発基礎演習 第1回

第03回おしながき

1. おしらせ(or アンケート)
    - 履修変更が今日までらしいですよ？
    - ルータの実機インターンシップ続報 **千歳スペシャルプランを作ってもらいました！**
    - 12/24 オンライン、移動； 12/25-26 築地で実機のネットワーク演習（超レア体験）_
      もう先着順でいいんじゃないか？ **（いちおう10/24〆切です）**
      （年末なので飛行機とホテルの手配が大変だから早く決めたい）
    - ポータルの添付資料欄を参照
    - ポータルのパスワードは、Pa5S
2. EL (確認テスト)
3. 今回のハイライト
4. 演習 “ユーザまわり”
    - 課題はポータルのレポート機能で提出
    - 今回、課題の動作確認はありません

ハイライトの前に補足

```
$ sudo python3 www.py
```

- 春学期に散々叩いたコレ、いったい何をやっているのか？
- 秋学期の授業では「コレを、きちんと説明できること」を目指しています

第03回のハイライト

- タイムシェアリングシステム(TSS)はマルチユーザ
- ユーザの区別
  - ユーザ登録時に各ユーザに数字(uid = user id)を割り当てています、名前の文字列は自由
  - 歴史的に管理者ユーザは、uid = 0, ユーザ名root
- ユーザに、できること、できないこと
  - 管理者ユーザrootは何でもできます
  - 一般ユーザadminも日常こまることはないはずです。 OSの管理やデバイスの設定などの危険なことは出来ません(させません)
- ユーザrootは極力つかわない、必要な場面で一時的に使う(最小権限の原則); suやsudoコマンド

演習

- おおむね一緒に操作をしていきます
- vocareum と aws console 両方の画面を出しておいてください
- 課題: ワークシートを出してください
  - 答え合わせは来週します

5W1HはUnixでも大事

- システム構築でも、 **つねに、 いま自分が誰(役)で、どこで、何をしているのか？を把握する** ことは重要です

- よくつかうUnixコマンド

  - id ... いま、誰の役？ (id は、そのまま id = identifictaionの頭文字?)
  - pwd ... いま、どこにいるのか？を確認する (print working directory)

(脚注) 例題は一緒に手を動かして行きます

[演習の準備] AWS EC2にログインしてください？

- 秋学期では、本来のサーバ操作にも慣れていくことにします。
  - （今学期は、初心者向けの支援システムColitasは使わない想定がデフォルトですが）
  - 秋学期末の自由課題(システム構築)は、本来のやり方でも、Colitas支援のもと実行でも、 好きな方を使ってください

- 本来(普通の作業では)、「AWS EC2にログインしてください」とは「sshコマンドでAWS EC2にリモートログインしてください」という意味です

(脚注) 演習の前に春学期の種明かしと正しいオペレーションについて話します

例題: vocareum から EC2 に ssh ログインする

1. いったん vocareum の画面を出してください
2. Elastic IPの「学籍番号.cloud.fml.org」への紐付けは終わっています
   - 以下、適宜「IPアドレス -> 学籍番号.cloud.fml.org」と読み替えてください
   - ダメな人はAWS EC2のPublic IPをクリップボードにコピーして「IPアドレス」を使ってください
     （DNSサーバ障害時も同様です）
3. ターミナルで次のコマンドを叩いてください
   - IPアドレスは各自となります
   - IPアドレス以外は全員おなじ文字列です

```
$ ssh -i .ssh/labsuser.pem admin@b2902900.cloud.fml.org
  もしくは
$ ssh -i .ssh/labsuser.pem admin@10.20.30.40
```

- 10.20.30.40の部分は上でコピーしたPublic IPです。各自となります
- $の部分は、もっと長い変な文字が書いてありますが、上の例では省略しています

[解説] sshコマンド

```
$ ssh -i .ssh/labsuser.pem admin@10.20.30.40
```

- sshコマンドの引数 = ログインしたい目的のサーバ。 フォーマットは「ユーザ名@サーバのIPアドレス」
  - だれ = ユーザ名 = admin (AWS EC2 debianの場合デフォルトが admin なので本科目ではadmin)
  - どこ = サーバのIPアドレス = EC2のPublic IP
- sshコマンドのオプション
  - 公開鍵暗号の認証情報を `-i ファイル名` で指定します。 ここで `-i` はidentityの頭文字です。 たいていオプション名は機能や役割を意味する英語の頭文字から選んでいます。 `-h` がhelpを意味するのも同様です。 代表的なオプションは、ほぼ、このパターンなので、よく使う英語は覚えておくべきです
  - **パスワード認証は危ないので、サーバでは利用できません** (ふつう、サーバ構築時にパスワード認証など出来ないように作成します)。 よってシステム構築・利用の際、公開鍵暗号の理解は必須です
  - **ファイル名の .ssh/labsuser.pem はAWS Academy特有のファイル名**です。
    - ここは案件ごとに違うはずです。各案件の指示書に従ってください

[解説] sshコマンド (初回だけの例外)

- 初回だけは、このペッセージが出るので yes を入力してください。 これは、なにをやっているのか？を調べるのは自由課題(任意)としましょう

```
eee_W_3695760@runweb140968:~$ ssh -i .ssh/labsuser.pem admin@54.205.249.209The authenticity of host '54.205.249.2
Are you sure you want to continue connecting (yes/no)?
```

- yes を入力後

```
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.205.249.209' (ECDSA) to the list of known hosts.
Linux ip-172-31-37-84 6.1.0-26-cloud-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.112-1 (2024-09-30) x86_64

The programs included with the Debian GNU/Linux system are free software;
  ... 省略 ...
Last login: Fri Oct 11 05:15:24 2024 from 44.243.172.246
Script started, output log file is '/usr/local/src/himo/typescript_476'.
```

例題: 目的のEC2にログインできたのか？

- 自分が今**インターネット上の「どこ」にいるのか？**を確認するには？
- sshログインした先が正しい目的地か？を調べる方法その１はサーバの名前を確認することです
  - hostname ... サーバの名前(ホスト名)を教えてくれるコマンド
- ただ、AWS EC2の場合、ホスト名では自信が持てません。 今日のところは、ターミナルの左端のプロンプトが、 へんてこな`eee_W_3695760@runweb140968:~$`のような文字列から `admin(Public IPアドレス)`に変更されていればOKとしましょう

```
[参考例] EC2でhostnameコマンドを実行したところ

admin(54.205.249.209):~ $ hostname
ip-172-31-37-84
```

(脚注1) このホスト名はEC2のプライベートIPアドレスを表示しています。 ただ、このIPアドレスはランダムに割り振られているし、 変わっていくことも多いので、この数字を覚えるのは無意味です

(脚注2) ホスト名にはwww.fml.orgといった識別しやすいドメイン名をつけるのが普通です

(脚注3) その他の確認方法は第5回にでもやりましょう

例題: いま自分は誰(役)なのか？

- いま、みなさんはAWS EC2にログインしていますね？
- コマンド
  - id ... ユーザ情報を表示するコマンド

```
// デフォルトでは、たくさん表示されますが
$ id

// uidのみを表示したいなら -u オプションをつけます
$ id -u

// では、これを実行すると、どうなるでしょう？
$ sudo id -u
```

例題は一緒に手を動かして行きます

解説: いま自分は誰(役)なのか？

- id コマンドを実行しているユーザの uid が表示されています
- sudo を前(左側)につけて id コマンドを実行すると、
  uid = 0 つまりユーザ root として実行していることが分かります

```
$ id -u
1000

$ sudo id -u
0
```

例題: では、これをやってみましょう

- 実行する前に、どういう結果になるでしょうか？を5秒間かんがえてからやってみくてださい

```
$ vi /etc/passwd

$ vi /etc/shadow
```

(脚注1) ちなみに、これがユーザの設定情報が書かれている設定ファイルです。

(脚注2) このファイルを直接編集してはいけません！必ずuseradd/usermod/userdelなどの管理コマンドを使ってください

例題: では、こうすると？

```
$ sudo vi /etc/shadow
```

(脚注) このファイルを直接編集してはいけません！ 何もせず、すぐに終了してください

課題

- 必須: なんども春学期に実行した`sudo python3 /home/admin/www.py`について、
  - 何のコマンドを、だれが、引数とオプションを何にして実行してるのか？を、きちんと説明してください
- 任意(自由課題、すこしは加点していく)
  - ssh コマンドで初回接続時にだけ発生する問い合わせは何をやっているのか？を説明してください (該当スライドページを参照)

(脚注) よくわからなくても、とりあえず考えて見てください。 来週、答え合わせをします。 基本、毎週の課題は、参加することに意義があります (課題提出 = 出席あつかいです、ポータルの出席は成績を付ける上では重要ではありません)

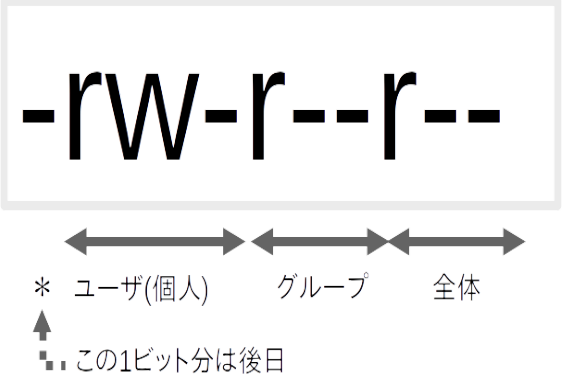
第04回おしながき

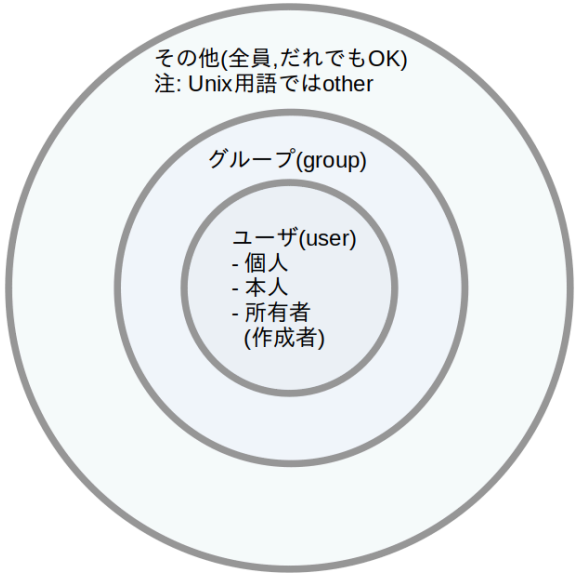
1. おしらせ(or アンケート)
    - 特になし?
    - ポータルのパスワード ... jUkH
    - (理由のある)欠席とかあった人は言ってくださいね
        - 課題を見てもらう -> メディコンさん
    - 付録として、大学院のスライドと動画再生リストを「諸注意」に追加しました
        - たしか四年生は履修登録できるはずです
2. EL (確認テスト)
3. 今回のハイライト
4. 前回(第03回)の課題の答え合わせ
5. 演習 “プロセスとファイル”
    - vocareum と aws console 両方の画面を出しておいてください
    - 例題は、一緒に操作をしていきます
    - 課題はポータルのレポート機能で提出
    - 今回、課題の動作確認はありません

第04回のハイライト

- ユーザAが開始したプロセスX(「ユーザAのプロセスX」)はユーザAと同じ権限を持ちます
- ファイルには属性があり、権限は9通り(だれ(3通り)」X「読み/書き/実行(3通り))あります

- `ls -l`(lsコマンド)は、上の9通りの属性を「 rw-r--r-- 」のように左端に表示しています

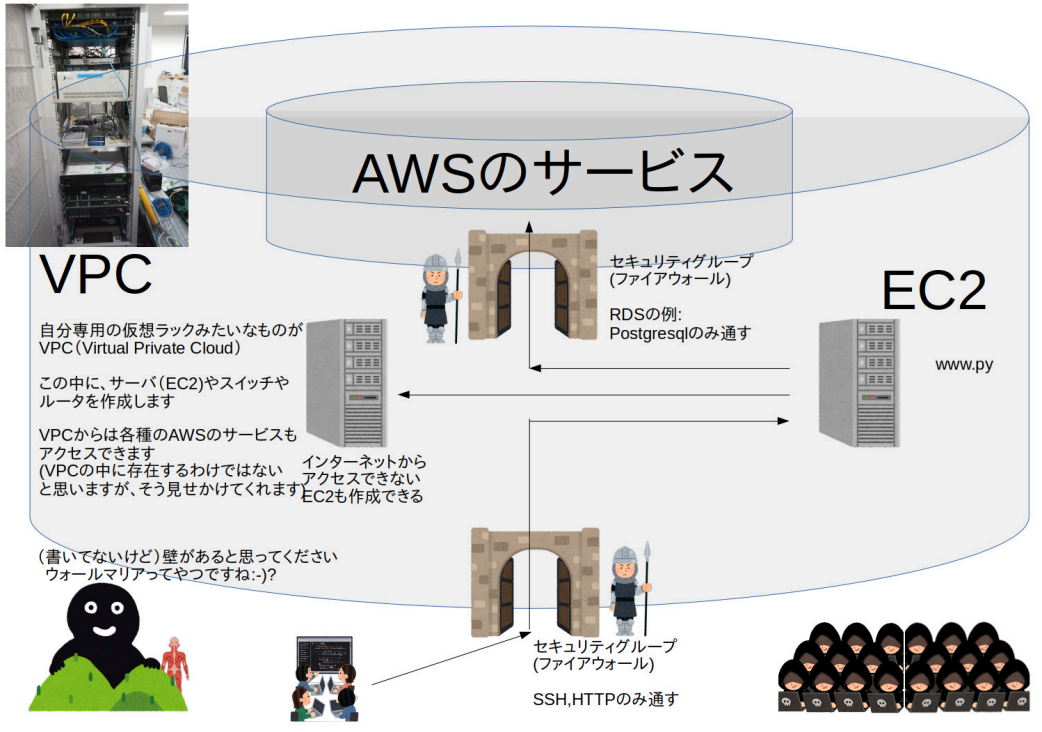




(脚注) 予習スライドに書いたとおり、「Unixセキュリティモデルが、開発の現場には、この程度の区別があれば十分という設計思想」だからです (これがベストであるという主張ではありません)。「エリート開発チームばかりの職場なら、この程度の区別で十分」と言い換えられそうです。 一般の開発現場は、このゆるい区別では運用できないでしょう

【第3回の補足】AWS VPC

- フィルタの話が前回のELにありました
- どこ(WHERE)でなに(WHAT)をどうやって(HOW)が整理されているなら、この説明が納得できると思います

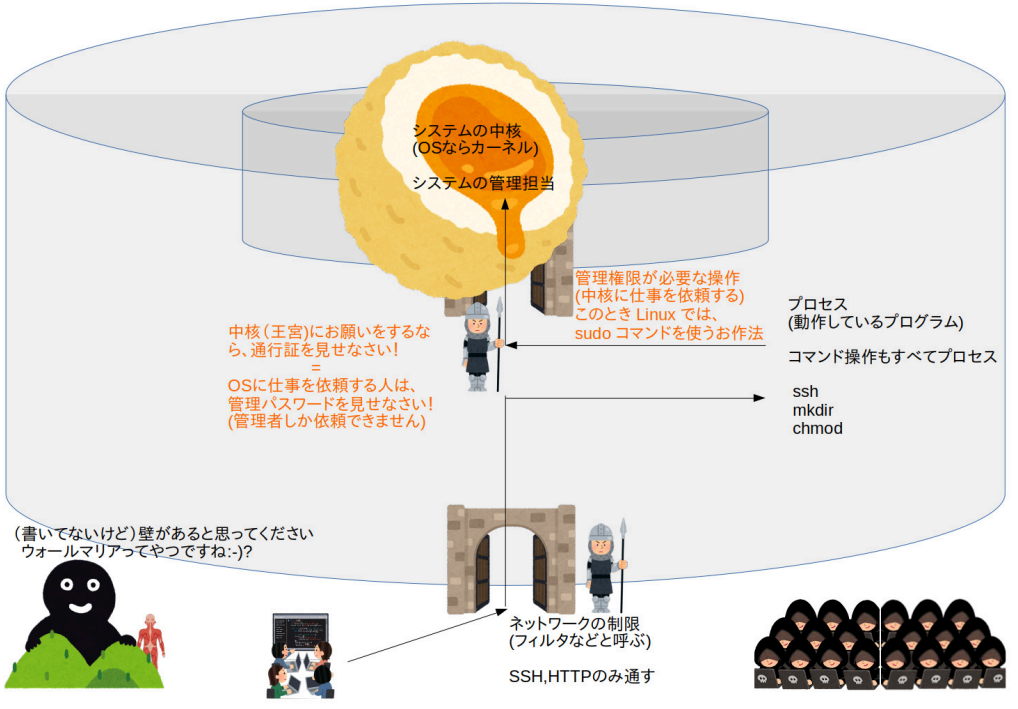


(脚注1) 進撃の巨人モデル :-)

(脚注2) ゼロトラストセキュリティとは 「次の瞬間となりの人が巨人になる想定でネットワークを設計・運用しろ」 という意味です:-)

【第3回の補足】Unixセキュリティモデル

- 前ページの一つのPCの中を拡大した様子
- 抽象化すれば、セキュリティモデルとしては同じ階層構造の絵になることを理解する
- 注：図では「出来ること」が書いてありますが、実際に設定するか？は別問題です
  - **EC2では、このフィルタは掛けていません**
  - この前段階にあるセキュリティグループで守ります



(脚注1) Unixの設計思想とクラウドの設計思想が異なることを理解する必要があります。 AWSの設計思想は、「Unix側では何もせず（いわゆるザルのセキュリティ）、AWSの仕組みでセキュリティを担保しなさい」というものです

第03回のレポートの答え合わせ(必須)

> 必須: sudo python3 /home/admin/www.pyについて、何のコマンドを、だれが、引数とオプションを何にして実行してるのか？を、きちんと説明してください

1. adminユーザがsudoコマンドをpython3 /home/admin/www.pyを引数に実行する。sudoのオプションは無し。

2. sudoコマンドは(rootユーザの権限下で)、python3コマンドを/home/admin/www.pyを引数として実行する。python3のオプションも無し。

3. python3(インタープリタ)はwww.pyファイルの中身をpythonのソースコードとして解釈し、実行結果を表示する

第03回のレポートの答え合わせ(自由)

> 任意(自由課題): ssh コマンドで初回接続時にだけ発生する問い合わせは何をやっているのか？を説明してください (該当スライドページを参照)

公開鍵暗号を応用して「誰(この場合はサーバのこと)」を確認/検証する方法がある。

sshの初回接続時には、この検証に使う情報（接続先サーバの情報）を保存する必要があり、「この情報を保存してよいか？を yes/no で尋ねている様子。 これをしておけば、接続2回め以降の検証は自動的に行われる。

検証しない場合、同じサーバ名もしくは同じIPアドレスで、こっそりサーバの中身がすりかわっていることに気づかないかもしれない。 これを検出する仕組み。 よって、サーバをリニューアルすると、ふたたび yes/no を聞かれることになるが、これは正しい動作。

なお、逆に言えば、初回接続時は「ユーザが自信を持って操作している」という大前提がある。ここが間違っていたら、ずっとだまされっぱなしとなることに注意

演習

- vocareum と aws console 両方の画面を出しておいてください
- 例題は、一緒に操作をしていきます
- 課題: ワークシートを出してください
  - 答え合わせは来週します
- [再掲] 本科目の目標(の2/3位?)は「`sudo python3 /home/admin/www.py`を詳細に説明できる」こと

5W1HはUnixでも大事

- システム構築でも、 **つねに、 いま自分が誰(役)で、どこで、何をしているのか？を把握する** ことは重要です
- 今回は、5W1Hから少しずれていますが、 うまくいかないとき、 **誰**=ユーザ(自分)にとって、 **ファイルの権限が何に当たるのか？** を確認する必要があります
- ファイルの権限まわりで、よく使うUnixコマンド
  - ls ... ファイルの一覧を表示するコマンド(lはエルです、limaのl)
  - chmod ... 属性(モード)を変えるという意味(そのままの)change modeの頭文字

(脚注) Unixでは「lsはLiSt directory contentsの略」と称していますが、 Multicsに由来する説(lsはlist segmentの頭文字)があり、たぶんそちらが正解です (注: Unixのファイルに相当するものをMulticsではセグメントと呼んでいた)。 Unix開発チームは、Unix開発以前MulticsというOS開発に関係していました。 Unixの用語や概念はMulticsや(さらにその先祖)CTSSに由来するものがいろいろあります

AWS EC2にログインしてください

1. いったん vocareum の画面を出してください
2. AWS EC2のPublic IPをクリップボードにコピーしてください
3. ターミナルで次のコマンドを叩いてください
    - IPアドレスは各自ことなります
    - IPアドレス以外は全員おなじ文字列です

```
$ ssh -i .ssh/labsuser.pem admin@10.20.30.40
```

- 10.20.30.40の部分は上でコピーしたPublic IPです。各自ことなります
- `$`の部分は、もっと長い変な文字が書いてありますが、上の例では省略しています
- sshコマンドの引数 = ログインしたい目的のサーバ。 フォーマットは「ユーザ名@サーバのIPアドレス」
    - だれ = ユーザ名 = admin (AWS EC2 debianの場合デフォルトが admin なので本科目ではadmin)
    - どこ = サーバのIPアドレス = EC2のPublic IP
    - 公開鍵暗号の認証情報を`-i ファイル名`で指定します。

【復習】例題: いま自分は誰(役)なのか？

- いま、みなさんはAWS EC2にログインしていますね？
- コマンド
  - id … ユーザ情報を表示するコマンド

```
// デフォルトでは、たくさん表示されますが
$ id

// uidのみを表示したいなら -u オプションをつけます
$ id -u
1000

// では、これを実行すると、どうなるか？
$ sudo id -u
0
```

例題: ダウンロードするとファイル属性はどうなるか？

- 実行する前に、どういう結果になるでしょうか？を5秒間かんがえてからやってみましょう

```
// ダウンロードする
$ curl -O https://api.fml.org/dist/cc-04.py

// ファイルの属性を見ると？
$ ls -l /home/admin/cc-04.py

// これも、ためしてください
$ /home/admin/cc-04.py
```

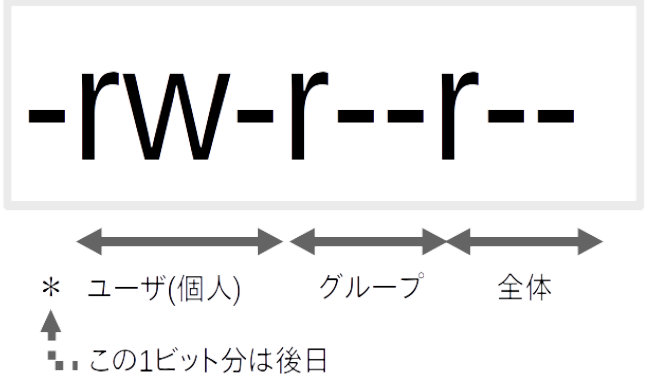
(脚注)

Q: なぜ属性が、このように設定されるのか？

A: (1) ソースコード内でファイルを作成するコード(curlのプログラマによる指定) (2) ファイルを作成する関数が、シェルの設定(umask)を見て、 (1)と(2)をあわせた結果が、これ。 もっともファイルを作成するために呼び出す関数が、 こてこていろいろする関数なのか、生っぽいか？によって挙動は微妙に異なる。 上の説明は、いちばん生っぽい場合(OSの機能を呼び出すopenシステムコール)の説明

解説: ls -l /home/admin/cc-04.py

```
$ ls -l /home/admin/cc-04.py
-rw-r--r-- 1 admin admin 587 Oct 25 09:25 /home/admin/cc-04.py
```



(脚注) つまりユーザ(本人)だけは読み書き可能、自分以外の人も読むことだけ出来ます

例題: 実行権限を付けてみる

- 実行権限を付けてみましょう

```
// 実行権限を付けて
$ chmod a+x /home/admin/cc-04.py

// 何か作業をしたら、そのあと、つねに結果を確認するクセをつけること！ 【超重要】
$ ls -l     /home/admin/cc-04.py

// 実行してみると？
$ /home/admin/cc-04.py
```

(脚注) 大事なところから2度書いておきますね。

【大事】 何か作業をしたら、そのあと、つねに結果を確認するクセをつけること！ 【超重要】

例題: では、こうすると？

- 前回(第03回)と今回(第04回)のハイブリッドな作業内容ね

```
// 前ページの実行結果
$ /home/admin/cc-04.py
hello world
uid (user  id) = 1000
gid (group id) = 1000

// もちろん、こうやっても同じ結果
$ python3 /home/admin/cc-04.py

// こうすると、どうなる？実行する前に5秒考えてから実行してください
$ sudo /home/admin/cc-04.py
```

例題: chmodできるユーザは誰？

```
// 前回みえなかったファイルですね？
$ vi /etc/shadow

// こうすると、どうなるかな？
$ chmod a+w /etc/shadow
```

(脚注1) ユーザrootは最強なので、全ユーザの権限を変更できます

(脚注2) 普通のサーバで試さないように！

(脚注3) まちがえてsudo chmod と打つなよ！

課題

- 必須:
  - なぜダウンロードすると rw-r--r-- が定番なのか？ (つまり実行権限(x)が付けないようにしているのか？) を考察しなさい
- 任意(自由課題、すこしは加点していく)
  - `--x--x--x` という設定をすると、どういう動作になるのか？ そして、この設定は、どういう場面で必要になるだろうか？ (実際、うちの運用サーバ上の、とある場所では、この設定をしています)
  - 実行権限(x)を付ければコマンドになるわけではない (たとえばzipはコマンドに成らない)。 www.pyやcc-04.pyが実行権限(x)を付けるとコマンドになるのは、 どういう理屈なのだろうか？

(脚注) これも、とりあえず考えてみよう。 参加することに意義があるぞ。 あと、丸暗記するのが仕事じゃないぞ、 丸暗記して成果を検索する仕事なら Google 先生とか OpenAI とかのほうが圧倒的に強いから、 人間より OpenAI とか Gemini とか Bedrock (AWS)を下請けにしたほうが安くて速くて圧倒的にいいわね

第05回おしながき

1. おしらせ
    - ポータルのパスワード ... RHYV
    - 実機インターンシップ千歳特別対応は今日が〆切(第3回の添付資料欄を参照)
2. EL (確認テスト)
3. 今回のハイライト
4. 前回(第04回,必須)の課題の答え合わせ
5. 演習
    - vocareum と aws console 両方の画面を出しておいてください
    - 例題は、一緒に操作をしていきます
    - 課題はポータルのレポート機能で提出
    - 今回、課題の動作確認はありません

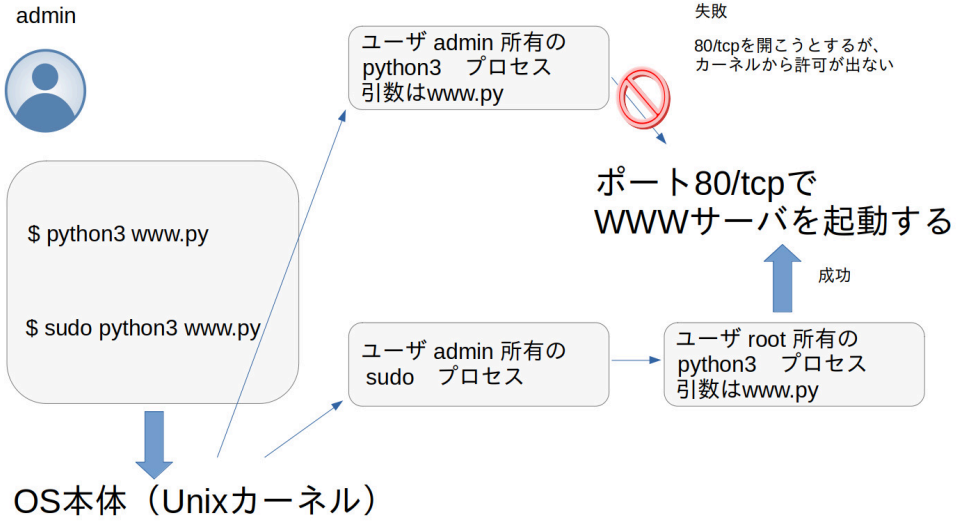
前半のハイライトは「sudo python3 /home/admin/www.pyを詳細に説明できる」ことです

第05回のハイライト

- ユーザの権限は、ユーザが起動したプロセスが利用する機能(例:ネットワーク)にも影響を与えます
  - 【復習】ユーザAが起動したプロセスXはユーザAと同じ権限を持つ
- 一般ユーザが普通に利用する(例:ブラウザを使うなどの)場合、制限はありません
- 管理者ユーザ(root)だけが主要サービスのサーバ(ポート番号が1024未満)の起動ができます
- 【復習】UnixとAWSの機能の区別をきちんとしてください
  - security groupはAWSの機能で、今回のテーマとは(つまりUnixの話とは技術的に)別物

第05回のハイライト(図解)

- ユーザの権限は、ユーザが起動したプロセスが利用する機能(例:ネットワーク)にも影響を与えます
    - 【復習】ユーザAが起動したプロセスXはユーザAと同じ権限を持つ
- 管理者ユーザ(root)だけが主要サービスのサーバ(ポート番号が1024未満)の起動ができます



第04回のレポートの答え合わせ(必須)

> Q: なぜダウンロードすると rw-r--r-- が定番なのか？

- A: 「実行できる」のは良くない
  - アタックの定番のひとつは、なんとかして攻撃目標でUnixコマンドを実行することなので、 実行できるファイルは出来るだけ作成しない
  - これは、(ダウンロードする際に、 権限が a-x となるようにプログラムを書く人+OSの設定をする人が気を使っている)
  - ついでに言えば、本人以外が編集できるのもよくない

演習のハイライト

- ネットワーク情報の基本的な調査方法を体験してみる
- ユーザの権限とネットワーク(サーバ)を実感する
- security groupの意義を実感する

演習

- vocareum と aws console 両方の画面を出しておいてください
- 例題は、一緒に操作をしていきます
- 課題: ワークシートを出してください
  - 答え合わせは来週します

【再掲】AWS EC2にログインしてください

1. いったん vocareum の画面を出してください
2. AWS EC2のPublic IPをクリップボードにコピーしてください
3. ターミナルで次のコマンドを叩いてください
    - IPアドレスは各自ことなります
    - IPアドレス以外は全員おなじ文字列です

```
$ ssh -i .ssh/labsuser.pem admin@10.20.30.40
```

- 10.20.30.40の部分は上でコピーしたPublic IPです。各自ことなります
- `$`の部分は、もっと長い変な文字が書いてありますが、上の例では省略しています
- sshコマンドの引数 = ログインしたい目的のサーバ。 フォーマットは「ユーザ名@サーバのIPアドレス」
    - だれ = ユーザ名 = admin (AWS EC2 debianの場合デフォルトが admin なので本科目ではadmin)
    - どこ = サーバのIPアドレス = EC2のPublic IP
    - 公開鍵暗号の認証情報を`-i ファイル名`で指定します。

例題: ネットワークの調査(1): EC2のIPアドレスを確認する

- EC2にログインしてipコマンドを実行します
- ip コマンド、フォーマットは `ip コマンド オプション`
  - コマンドのところが色々あります。もっとも基本となる「IPアドレスの調査」であれば address を使います。 曖昧さがないなら、コマンド部分は省略形が使えます。aで始まるコマンドは他にないのでaでもよいです

```
$ ip addr
    〜省略〜
```

(脚注) 一般のサーバでは、そのサーバのグローバルIPアドレスが表示されるので、どこのサーバか？確認できます。 一方、EC2の場合、Unix側からPublic IPは確認できません。 プライベートアドレスが表示されるだけです。 たしかにAWS Consoleと同じだね！と確認できるだけだし、プライベートアドレスが変わることもあるので、特に重要な意味はありません。

例題: ネットワークの調査(2): EC2で開いているポート番号を確認

- ss ... (ss = socket status の略だったはず)
  - フォーマットは `ss [オプション]`
  - `-l` ... listenしているポート番号を表示。つまり「サーバが利用しているポートを教えろ」という指示
  - `-4` ... IPv4アドレスのみ表示
  - `-n` ... ポート番号を数字で表示(指定しないとプロトコル名が表示される) (脚注2)

```
$ ss -l4n
    〜省略〜
```

(脚注1) **システム構築の各段階で、こういった動作確認をしながら進めていきます。 そういうクセがあるかないかが、プロとアマチュアの違いだ！と言ってよいでしょう**

(脚注2) ユーザにやさしくなるように、プロトコル名とかドメイン名に変換した結果を出そうとしますが、 この変換システムが障害の場合は問い合わせ途中で止まってしまうし、 見間違えるのも困るので、 **プロは、生のIPやポート番号で読み書きできるべき**

【補足】例題: ネットワークの調査(3): EC2のフィルタを確認

```
$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

- iptablesコマンドは、カーネルのフィルタシステムを操作する管理コマンドです(-LはLimaのL)
- ヘッダが表示されるだけで、意味のある行は表示されていません
  - フィルタが無い = すべての通信を許可するという意味に（Debianでは）なります

(脚注1) **期末試験には出ません** (脚注2) 自分でサーバを借りて運用する場合は、きちんとUnix/Linuxでフィルタを設定するようにしてください (脚注3) フィルタを設定しない時の設計思想は各サーバ/ネットワーク機器ごとにマチマチです。「すべてを許可」「すべてを禁止」どちらもありえます。 必ず確認すること！（自分の知識/常識を疑うように）

【追記】www.pyでcgiのエラーが出る場合

- www.pyを起動しようとして次のエラー（cgiというモジュールはありません）が出る場合は
- python3-legacy-cgiパッケージをインストールしてください

```
[エラーメッセージ]

～省略～
ModuleNotFoundError: No module named 'cgi'

[python3-legacy-cgiパッケージをインストールする]

$ sudo apt update
$ sudo apt install python3-legacy-cgi
```

課題(1): www.pyの権限を確認する

1. curlでwww.pyをダウンロードし、
2. `python3 /home/admin/www.py`を実行すると、どうなるのか？試しなさい
3. どうすればwww.pyが動くのか？考えて試してみなさい

```
[実行例]

$ curl -O https://api.fml.org/dist/www.py
$ python3 /home/admin/www.py
```

- オプションの`-O`は大文字のオー(OscarのO)です

(脚注) これは、ほぼ春学期と同じ作業なので、みまもってみましょう

課題(2): セキュリティグループ無しの場合どうなるのか？SSH編

- 新たに、もうひとつ**「すべてのセキュリティグループにチェックを入れない」**EC2を作成してください。 以下、EC2(2号機)と呼称します
- このEC2(2号機)にSSHしてみてください
- (次の2つの情報が分かるような)スクリーンショットを貼ったレポートを出してください
  - EC2(2号機)のPublic IP
  - sshして失敗している様子

(脚注) これは、ほぼ春学期と同じ作業なので、みまもってみましょう

課題(3): セキュリティグループ無しの場合どうなるのか？HTTP編

1. EC2(2号機)の security group の管理画面を開き、22/tcpと80/tcpを許可
2. EC2(2号機)のPublic IP情報を記録(スクリーンショットを取り、レポートに貼る)
3. EC2(2号機)にSSHでログインし、www.pyをダウンロードする
4. WWWサーバ(つまりwww.py)を起動し、サーバにアクセスしている記録をとる
    - ブラウザの上の方だけスクリーンショットを取り、レポートに貼えば十分です
    - Public IPとwww.pyの出力がわかるようにスクリーンショットを取ってください
5. EC2(2号機)の security group の管理画面を開き、80/tcpを不許可にする
6. WWWサーバにアクセスできなくなった様子を記録 (スクリーンショットを取り、レポートに貼る)
7. EC2(2号機)の security group の管理画面を開き、80/tcpを許可
8. WWWサーバにアクセスできるようになったことを確認 (スクリーンショットを取り、レポートに貼る)
9. 最後に、**今回あらたに作成したEC2(2号機)は削除してください** (課金を避けるため不要なEC2は削除)
10. EC2の管理画面でEC2が一つ(第2回に作成したもの)だけになっていることを確認
    (スクリーンショットを取り、レポートに貼る)

(脚注) これは、手順だけ書いておきますので、レッツ、チャレンジ

課題の提出

- いつもと同じで、ポータルのレポートボックスに提出してください
- 作成したEC2は削除してください (無駄に課金されます、削除しましょう)

第06回おしながき

1. おしらせ
   - ポータルのパスワード ... gfEG
   - 11/21 ?前半の総まとめで口頭試問をしようかな？です
   - 11月末からはクラウドらしいシステム構築を始める（前半は、半分が春学期の復習）
   - インターンシップの御紹介
     - 12/10 (online): たった3時間のプログラムでIT業界と自分自身を理解する
     - https://avinton.com/blog/2025/10/oneline-winter-internship/
   - パイプの話ほかコンピュータ開発史をかすめる動画が、こちら https://youtu.be/zouFcwWaW6M?list=PLS2cEmI21XYLf3k7IjKTDql-rYXHhe_1P
2. EL (確認テスト)
3. 今回のハイライト
4. 前回の課題の答え合わせ
5. 演習
   - vocareum と aws console 両方の画面を出しておいてください
   - 例題は、一緒に操作をしていきます
   - 課題はポータルのレポート機能で提出
   - 今回、課題の動作確認はありません

第06回のハイライト

- デバイスドライバ、デバイスごとに必要なプログラム
- 仮想記憶(virutal memory: これも略称はVM)
  - 複数のプロセス
  - 物理メモリよりも巨大な仮想メモリ(負け)
- ファイルシステムとプロセスの階層構造
  - 情報の先出し
  - マトリョーシカ本棚:-)
  - 来週もうすこし詳しく階層型ファイルシステムの演習をやります

(脚注1) その他の話題せいぞろいなので、すこし統一感が無いですけどね (脚注2) 負けPCが多すぎるってか

第05回の課題の答え合わせ？

- 正解は、はっきりしてないです
  - とくにAWSだと、手が出せない
- 「ssh とかブラウザが timeout している様子」をコピペしか出来ないはず
- （オンプレなら）、このくらいの確認はします
  1. フィルタを設定したルータのログの出力レベルを上げて、通信を拒否する様子を確認
  2. ルータでパケットをダンプしながら、その弾いている様子を見せる （これが出来るか？は機種によります）
  3. サーバ（EC2）側でもダンプしてパケットが来ていないことを示す

演習のハイライト

- コンピュータの色々な要素(部品)の現状を調査する方法
- プロセス群ストレージの階層構造を見てみる
  (Unixでは何でもファイルだし、可能な限り全て階層構造)

演習

- vocareum と aws console 両方の画面を出しておいてください
- 例題は、一緒に操作をしていきます
- 課題: ワークシートを出してください
  - 答え合わせは来週します

【再掲】AWS EC2にログインしてください

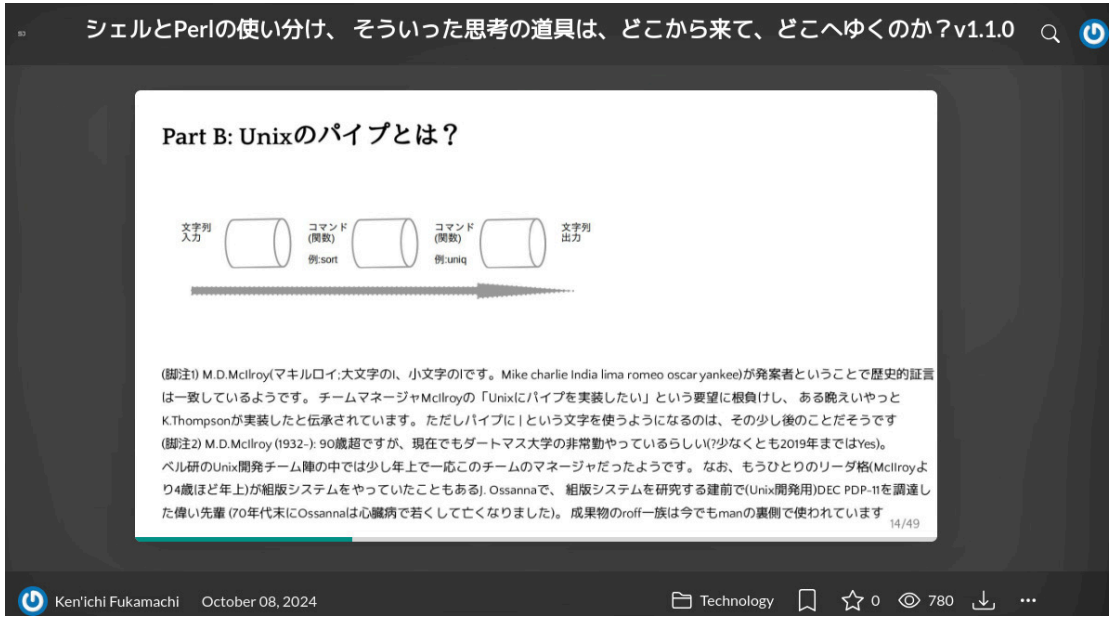
1. いったん vocareum の画面を出してください
2. AWS EC2のPublic IPをクリップボードにコピーしてください
3. ターミナルで次のコマンドを叩いてください
    - IPアドレスは各自ことなります
    - IPアドレス以外は全員おなじ文字列です

```
$ ssh -i .ssh/labsuser.pem admin@10.20.30.40
```

- 10.20.30.40の部分は上でコピーしたPublic IPです。各自ことなります
- `$`の部分は、もっと長い変な文字が書いてありますが、上の例では省略しています
- sshコマンドの引数 = ログインしたい目的のサーバ。 フォーマットは「ユーザ名@サーバのIPアドレス」
    - だれ = ユーザ名 = admin (AWS EC2 debianの場合デフォルトが admin なので本科目ではadmin)
    - どこ = サーバのIPアドレス = EC2のPublic IP
    - 公開鍵暗号の認証情報を`-i ファイル名`で指定します。

パート1: EC2の構成要素を探る

- 方法その１
  - ‘lsなんとか’コマンドシリーズによる調査
- その他の方法は省略



(脚注) このあとのコマンドには出力が長いものがいくつかあります。 そういう場合、**コマンド | head**とすると最初の10行だけで表示が打ちきられます。 パイプ(| 部分のこと)とは何？の説明はスライド右側の YAPC::Hakodate 2024招待講演の**スライド(P.14-)**を参照

パイプの話はパートBで、その後のスライド本編「コンピュータ開発史」の序章部分です。

講演の動画が [Youtube](#) **で公開**されています。パイプの話は８分〜あたり

CPU

```
admin(54.205.249.209):~ $ lscpu
Architecture:                       x86_64
CPU op-mode(s):                     32-bit, 64-bit
Address sizes:                      46 bits physical, 48 bits virtual
Byte Order:                         Little Endian
CPU(s):                             1
On-line CPU(s) list:                0
Vendor ID:                          GenuineIntel
Model name:                         Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
CPU family:                         6
Model:                              79
 ... 省略(すごく長い) ...
```

(脚注1) lscpu | head すると最初の10行だけで表示が打ちきられます。 | については前ページの脚注を参照のこと (脚注2) head は最初の10行だけを表示して終了するコマンド。「head -数字」とすれば先頭の「数字」行を表示。 ちなみに反対の動作をするコマンドがtail

メモリ

```
admin(54.205.249.209):~ $ lsmem
RANGE                                 SIZE  STATE REMOVABLE BLOCK
0x0000000000000000-0x00000003fffffff   1G online       yes   0-7

Memory block size:       128M
Total online memory:       1G
Total offline memory:      0B
```

- 細かい話はともかく、メモリ量が1GBであることは分かる

ストレージ

```
admin(54.205.249.209):~ $ lsblk
NAME     MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
xvda     202:0    0    8G  0 disk
 ├─xvda1  202:1    0  7.9G  0 part /
 ├─xvda14 202:14   0    3M  0 part
 └─xvda15 202:15   0  124M  0 part /boot/efi
```

- これも細かい話はともかくストレージの総量が8GBであることが分かる

(脚注) 下の3行はストレージ(仮想ディスク)のレイアウトの話になります。 レイアウトについてはGPTを勉強してください。

ちなみにDebianのOSイメージが書きこまれているのはxvda1という名称の部分だけです

パート2: Unixシステムの状態確認 (システムのサマリ)

- 最重要のものだけを２、３
  - これも色々ありますけどね

df コマンド ... Unixのストレージ利用状況の詳細

```
admin(54.205.249.209):~ $ df
Filesystem     1K-blocks    Used Available Use% Mounted on
udev              485380       0    485380   0% /dev
tmpfs              99336     492     98844   1% /run
/dev/xvda1       8025124 1905800   5690116  26% /
tmpfs             496660       0    496660   0% /dev/shm
tmpfs               5120       0      5120   0% /run/lock
/dev/xvda15       126678   11840    114838  10% /boot/efi
tmpfs              99332       0     99332   0% /run/user/1000
```

- xvda1つまりDebian GNU/Linuxがインストールされている部分の詳細確認
  - **つねに残り容量が十分大きいか?を確認するべきです。** 上から4行目の`/dev/xvda1 8025124 1905800 5690116 26% /`が最重要で、 ぱっと見わかりやすいのは五列めの(すでに使用済が)26%というところ。 具体的に残り何KB利用可?が4列目ここでは5,690,116KB (5.69Gb)。 一番右の`/`はファイルシステムの`/`つまり一番上で、HDD全体を意味します

(脚注) df -m とすればMB単位の表示に切り替わり、より見やすくなります

top コマンド ... OSのステータスサマリ

```
top - 01:29:59 up 1 min,  1 user,  load average: 0.09, 0.04, 0.01
Tasks:  86 total,   1 running,  85 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.3 sy,  0.0 ni, 89.0 id,  0.6 wa,  0.0 hi,  0.0 si, 10.1 st
MiB Mem :    970.0 total,    748.8 free,    229.1 used,    123.9 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.    740.9 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
     36 root       0 -20       0      0      0 I   0.3   0.0   0:00.01 kworker/0:1H-kblockd
      1 root      20   0  102128  12068   9184 S   0.0   1.2   0:00.79 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kthreadd
   ... 以下略 ...
```

- **数秒おきに自動的に更新**されます。つねに確認するのは次の2ヶ所か
  - `load average: 0.09, 0.04, 0.01` ... どれくらいOSが忙しいのか？の目安
  - `MiB Mem : 970.0 total, 748.8 free` ... トータル1GB、748MBがフリー(空き)
- いま動いているプロセスが上の方に表示されます。例:**絶えず同じコマンドが表示される=重たい犯人か?**

(脚注) q (quebecc の q, quit の q)を押せば終了します。 エディタ以外は、とりあえず q とか Ctrl-C を押してみるのが終了の定番

ps コマンド ... プロセスのステータス

```
admin(54.205.249.209):~ $ ps
    PID TTY          TIME CMD
    486 pts/1    00:00:00 bash
   1010 pts/1    00:00:00 ps
```

- ps だけでは、自分のプロセスだけを表示して終わり
- 管理者が年中つかうのは「全プロセスの詳細な情報を表示せよ」という`ps -auxww`

(脚注1) psのオプションは、コマンドのフォーマット規則の例外です。 ps -auxwwも(- なしの)ps auxwwも同じく受け付けてくれて、同じ動作をします。 psには引数がなくてオプションしかないので - なしでもオプションだと分かるから大丈夫な例ですかね。 ps は、ひんぱんに使うコマンドなので - の1文字分すら面倒という気持ちは、よくわかります。 Unixは全般的にそういうモノです

(脚注2) もちろんps auxwwは大量の情報を表示するので見るのが大変です。 この情報を取捨選択して必要な部分だけを取り出すために、 grepやsedやperlを駆使して、ぱぱっと出来るのがプロ(お金もらえる)あるね

(脚注3) auxww のオプションの意味を調べるのは各自の宿題としましょう

パート3: Unixシステムの階層構造を調べる

pstree コマンド ... プロセスの階層(親子関係)を一覧表示する

```
admin(54.205.249.209):~ $ pstree
systemd─┬─2*[agetty]
        ├─cttyd.sh─────ttyd
        ├─dbus-daemon
        ├─polkitd──────2*[{polkitd}]
        ├─sshd─────sshd─────sshd─────bash─────script─────bash─────pstree
        ├─sudo─────bash─┬─bash
        │               └─inotifywait
        ├─systemd──────(sd-pam)
        ├─systemd-journal
        ├─systemd-logind
        ├─systemd-network
        ├─systemd-resolve
        ├─systemd-timesyn──────{systemd-timesyn}
        ├─systemd-udevd
        └─unattended-upgr
```

(脚注) まぁ面白いだけで、管理上つかうことは、まず無いですけどね:-)

[作業] tree コマンドをインストールする

```
admin(54.205.249.209):~ $ sudo apt update
 ...省略...
admin(54.205.249.209):~ $ sudo apt install tree -y
Reading package lists... Done
 ...省略...
The following NEW packages will be installed:
  tree
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 52.5 kB of archives.
After this operation, 116 kB of additional disk space will be used.
Get:1 file:/etc/apt/mirrors/debian.list Mirrorlist [38 B]
 ... 省略 ...
Unpacking tree (2.1.0-1) ...
Setting up tree (2.1.0-1) ...
Processing triggers for man-db (2.11.2-2) ...
```

tree コマンド ... ファイルとディレクトリを樹木状に表示する

```
admin(54.205.249.209):~ $ tree
.
├── htdocs
│   └── index.html
└── www.py

2 directories, 2 files

admin(54.205.249.209):~ $ tree /home
/home
└── admin
    ├── htdocs
    │   └── index.html
    └── www.py

3 directories, 2 files
```

(脚注) WindowsのフォルダのLeft menuをUnix ターミナル上で真似る感じ?

パート4: ファイルシステムの操作(序)

少しだけ

pwd コマンド ... 今どこにいるのか？

```
admin(54.205.249.209):~ $ pwd
/home/admin
```

- つねに**自分が、いま、どこで作業しているのか？**を確認することは非常に重要です
  - 5W1Hの話を思い出してください

(脚注) pwd は Print Working Directory の頭文字で、すなおな略称です

mkdir コマンド ... ディレクトリを作成する

```
admin(54.205.249.209):~ $ mkdir z
admin(54.205.249.209):~ $ tree /home
/home
└── admin
    ├── htdocs
    │   └── index.html
    ├── www.py
    └── z

4 directories, 2 files
```

- z ディレクトリを作成し、その結果をtreeで確認してみましょう

(脚注) mkdir は make directory の頭文字なので、やっぱり素直な略称。 make-directory の方が良い?長いから嫌です(w)

最終課題の希望調査（の頭出し）

- 今年は、定食＆カフェテリア方式にしようかと思うんよ。まだ案ですけど、
  - 定食から選ぶ、定食メニュー(次ページ参照)をこちらで用意します。一つを選択
  - カフェテリア方式
    - ようするに自由課題
    - (部品を好きに組み立てる、持ちこみもあり)

- ~~Google Formで希望調査をしています~~
  - 自由課題は妄想でもいいので、こういうのやろうかな〜作ってみたいな〜でOKです
  - 来週、希望調査をしましょうかね

定食メニュー(案)

- A定食: 最低限のショッピングカート(Sになることは無い,基本B?)
  - EC2が1台死んでもショッピングカートの中身が消えない
  - 必要な部品は全て授業で扱うので、組み合わせられるか？だけ
  - 答えを全て教えているので、ほめません(あたりまえ)
- B定食: A定食 + ELB (やや難?)
  - ドキュメントはあります、それ読んでガンバレ
- C定食: A定食 + RDS (やや難?)
  - きちんとカートの情報を(キャッシュだけでなく)SQLサーバにも保存してください
  - ドキュメントはAWS ACFにあります、それ読んでガンバレ
- D定食: EC2 x2 + EFS (やや難?)
  - キャッシュではなくファイルシステムを利用する。 通常、カートの保存目的で、この形態を使うことはないですが、 自分でドキュメントを読んで作れるなら大変よい
  - ドキュメントはAWS ACFにあります、それ読んでガンバレ

(脚注1) まだ案なので、プラン内容は変わる可能性があります。 成績は、その他の口頭試問、期末試験、普段の取り組みなどの合計なので、 それ次第

(脚注2) ACF = AWS Cloud Foundation、AWSの一番やさしい資格試験。 でも授業よりはるかに難しいの:-)

カフェテリア方式(自由課題)

- ようするに特にテーマはありません、予算の範囲内で好きに作ってください
- XXXのテクニックを使うにはどうすればいいの？などはコンサルは受け付けます
  いろいろな部品が使えて楽しいよね。たとえば
  - (もうすこしちゃんとした商品写真とかもあるサイトにするとして)
  - S3 (を商品写真のサーバとして使うという話の方な)
  - Cloudfront (おなじくCDNで商品写真)
  - Cloudformationで自動化(Infrasturacture as Code)
  - docker上にディプロイ (AWSのサービスは使わない、EC2のうえに独自システムを一式構築)
    - dockerは、去年のテキストを参照？まぁ検索すればdockerの使い方は山ほど見つかるよね
  - k8s上にディプロイ (AWSのサービスは使わない、EC2のうえに独自システムを一式構築)

第07回おしながき

- いつもと順番が少し異なります

1. おしらせ
    - ポータルのパスワード ... fS2s
    - 第9回は前半の締めくくりとして口頭試問をする予定です
        - いままでやったことをよく復習しておいてください
        - sudo python www.py を詳しく説明できるか？が前半のテーマでしたね？
        - 10月は権限まわりをテーマにしていました
        - 11月のテーマも復習してね
    - 実習パートの動画の再生リストは、がんばってつくっておきます
2. EL (確認テスト)
3. 演習
4. 今回のハイライト

階層性の復習

- 春学期の設計編にもどります
  - https://lectures.fml.org/slides/skill-network/design/exercises/
- 階層的に考えるのは（西洋）科学の基本原則です
  - つねにそう考えるくせをつけてください

少し探検(AWS)

```
$ pwd
$ ls
$ ls -l

$ cd /
$ pwd
$ ls
$ ls -l
```

第08回おしながき

1. おしらせ
    - ポータルのパスワード ... gCSj
    - 授業の録画の再生リスト、ポータルに貼りました。復習にどうぞ
    - 次回は前半の締めくくりとして口頭試問をする予定です(再掲)
        - いままでやったことをよく復習しておいてください
        - sudo python www.py を詳しく説明できるか？が前半のテーマでしたね？
        - 10月は権限まわりをテーマにしていました
        - 11月のテーマも復習してね
2. EL (確認テスト)
3. 今回のハイライト
4. AWS S3の使い方のデモ
5. 演習

第08回のハイライト

- 二大ストレージ
  - ブロック ... 512バイトで読み書き
  - オブジェクト... ファイルをHTTPSで読み書き(アップロード/ダウンロード)
- AWS S3
  - HTTPSで読み書きするオブジェクトストレージ
  - 無限大の容量、eleven 9 の耐久性、(IAMによる)セキュリティ
  - バックアップ先、災害対策

[実習] S3を体験する

手順

- S3 つくる、これはだれでもできるだろう
- S3のサイトを作る
  - 1時間限定でサイトを作るオプションがあるので、それでOK
  - ガチのサイトを作るのは、けっこう面倒なので、そっちはいいや
- S3に適当なindex.htmlをアップロードする
  - とりあえず学籍番号だけは書いてください（必須要件）
  - 見栄えを頑張るとかは、おまかせです
- S3のサイトを作れたことをTAさんに確認してもらう
  - URLを申請するフォームを作るので、それで申請する（？）
  - 確認されたら、画面に確認が終わった学籍番号が反映される（？）

課題

- 【課題】
- 冗長構成なショッピングサイトを作る際にS3を使う案もある。
  - （もちろん最終課題の設計を念頭においています）
  - どういう使い方をするべきか？を考えてください。
  - また、他のAWSサービスに適切なものがあるか？も調べてください。 もしそういったサービスが存在するなら、それらのサービス群を、どのように使い分けるのがベストプラクティスだろうか？
- portal のレポートボックスで提出してください /small>

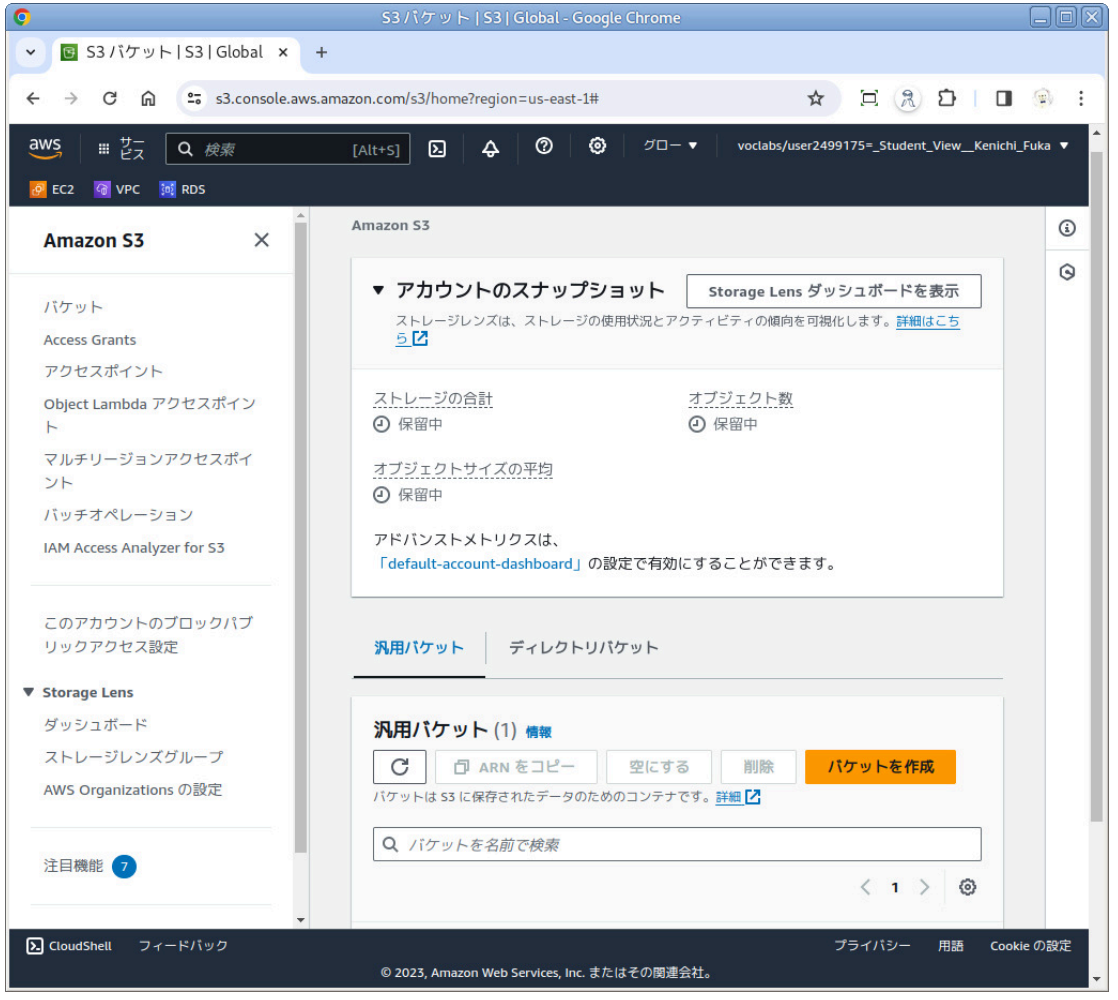
【付録】AWS S3バケットの作成

AWS S3とは?

- Google Driveみたいなもの
  - 一般には、Google DriveやAWS S3のことをオブジェクトストレージと呼んでいます
  - 第9回の講義で少し話します
- 用語が違うので、ここは覚えること
  - バケット(Bucket) ... Google Driveの一番上の階層のフォルダみたいなもの
  - オブジェクト(Object) ... Google Driveの各ファイルのこと

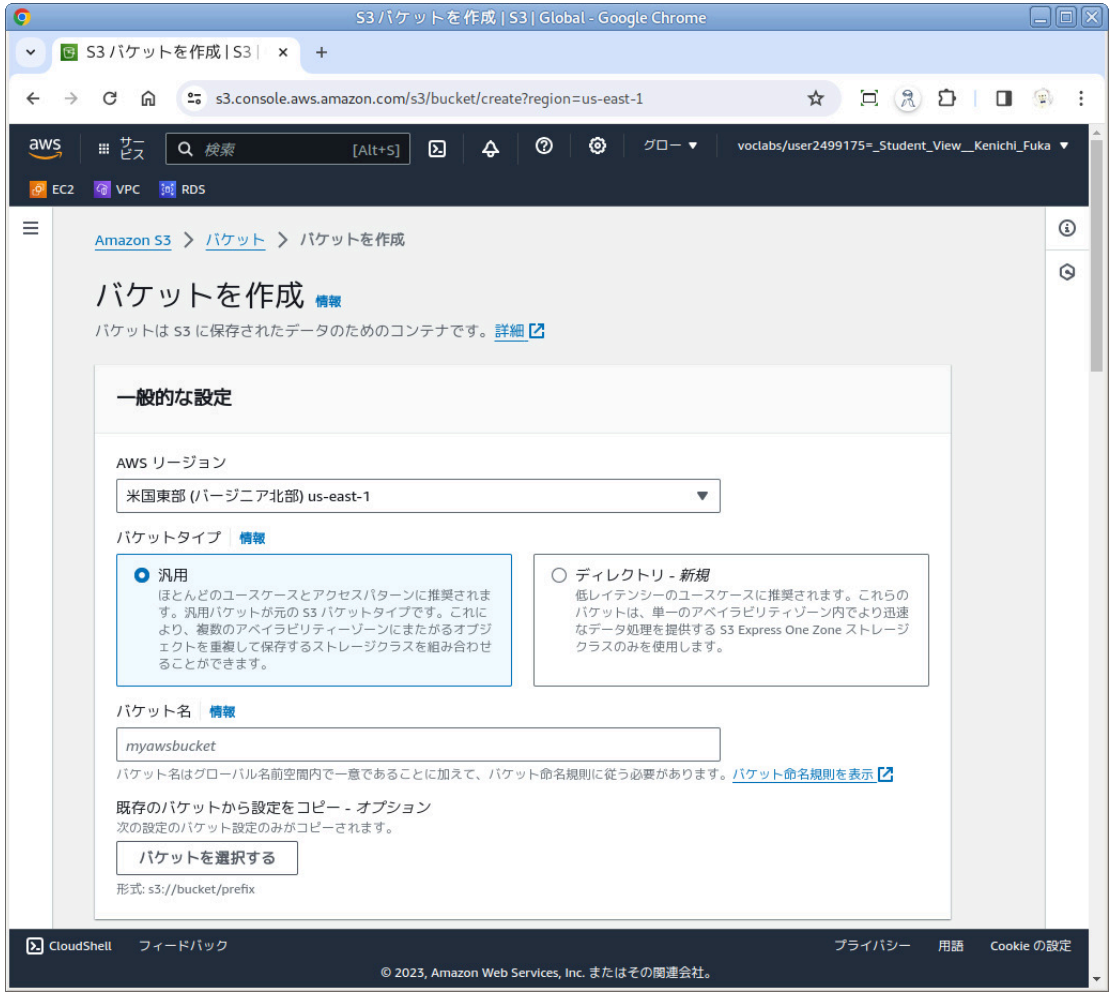
S3バケットの作成(1)

- AWS consoleでS3を探してクリック
- **画面右下のオレンジの「バケットを作成」**をクリック(右図)



S3バケットの作成(2)

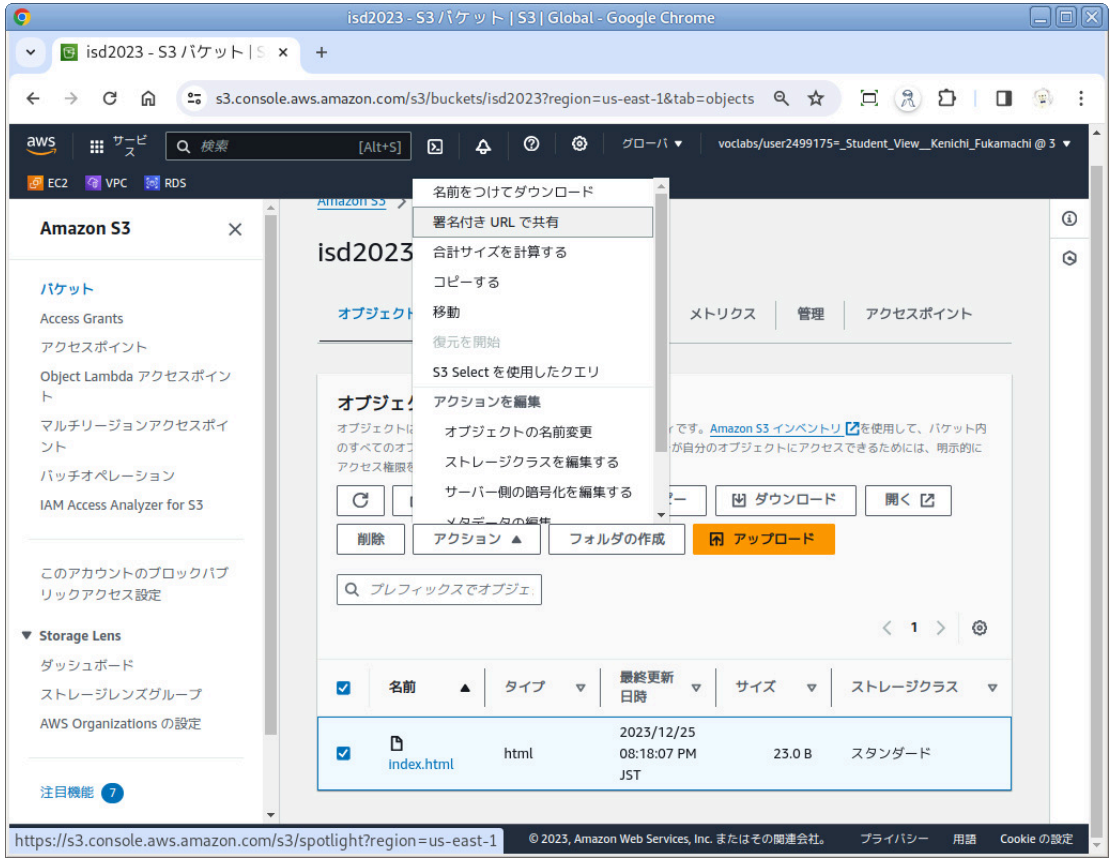
- 「バケットを作成」画面(右図)で
- **バケット名**をつけます
  - バケット名は**学籍番号**にします
  - それ以外の設定はデフォルトのまま
- (縦長の画面なので、だいぶ下にスクロールして) 画面右下のオレンジの「バケットを作成」をクリック(右図)
- 最初の画面(前頁の図)に戻るので、 下の方に作成したバケット名があることを確認してください



AWS S3のファイルをウエブ公開(時間制限付)する

AWS S3のファイルをウエブ公開(時間制限付)する(1)

1. S3の管理画面に入り
2. バケットを選んでクリックすると
3. 「オブジェクト」という画面になります。オブジェクトはファイルです
4. 公開したいオブジェクトを選択し、
5. 「アクション」で「書名付きURLで共有」をクリックします (右図)



AWS S3のファイルをウエブ公開(時間制限付)する(2)

- 時間を設定して、
  - 「分」か「時間」を選び
  - 「分数」欄に数字を設定します
- 右下のオレンジ色の「署名付きURLを作成」をクリックしてください
  - 生成されたURLは、自動的にクリップボードにコピーされます
- 動作確認
  - ブラウザでクリップボードのURLへアクセスしてみてください