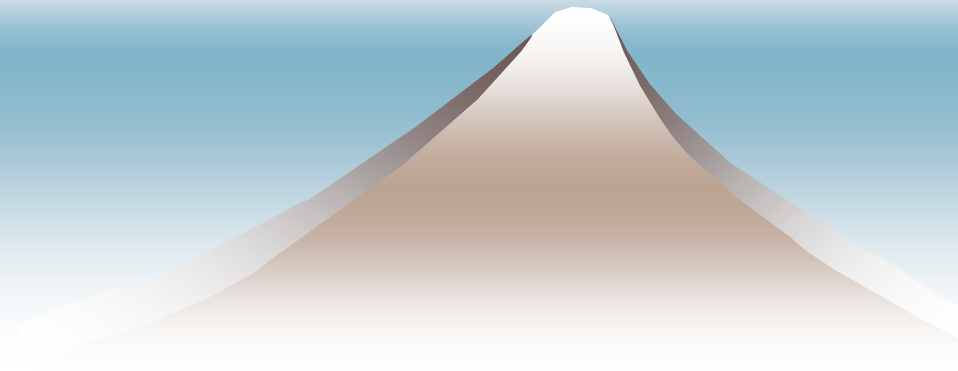


ネットワークアプリケーションと セキュリティ



代表的ネットワーク アプリケーション

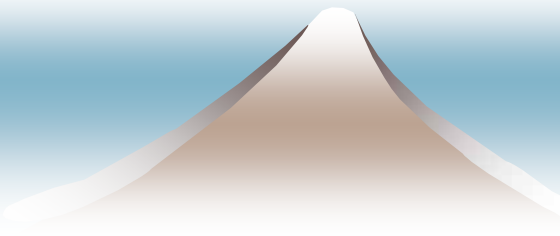
- ◆ Web
- ◆ メール
- ◆ リモートログイン
 - SSH, RSH, etc.
- ◆ ネットワークファイル共有
 - CIFS/Samba, NFS, etc.
- ◆ 遠隔端末
 - X Window, Windows Remote Desktop

まとめ知識

- ◆ WindowsとVMWare内のLinux間のファイル共有を試してみたい人は、去年の演習のページに書いてあります
 - <http://www.logos.t.u-tokyo.ac.jp/~masamiti/enshu/index.html>から「過去の情報」「昨年度の情報」をたどる
- ◆ WindowsからLinuxへのリモートログインについても同様

ネットワークAPI : ソケット

- ◆ さまざまなプロセス間通信プロトコルに共通のAPI
 - インターネット(IP, UDP, TCP)
 - いくつかのLANプロトコル(AppleTalk, etc.)
 - 1 Unixコンピュータ内 (Unix domain)
- ◆ しかし現状は,
 - ソケット ≈ インターネットのためのAPI



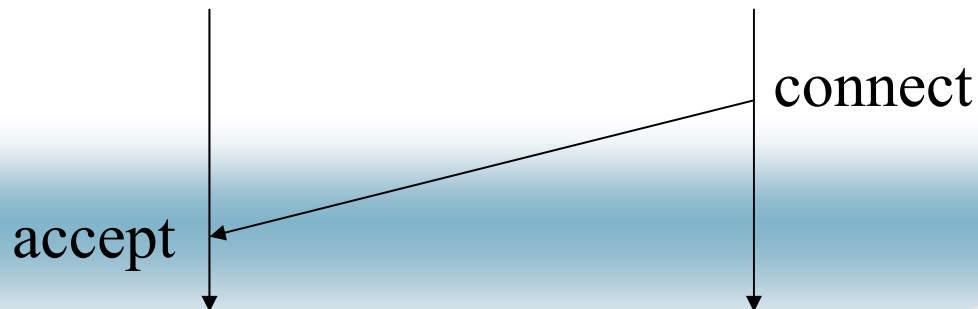
ソケットAPI

◆ サーバ

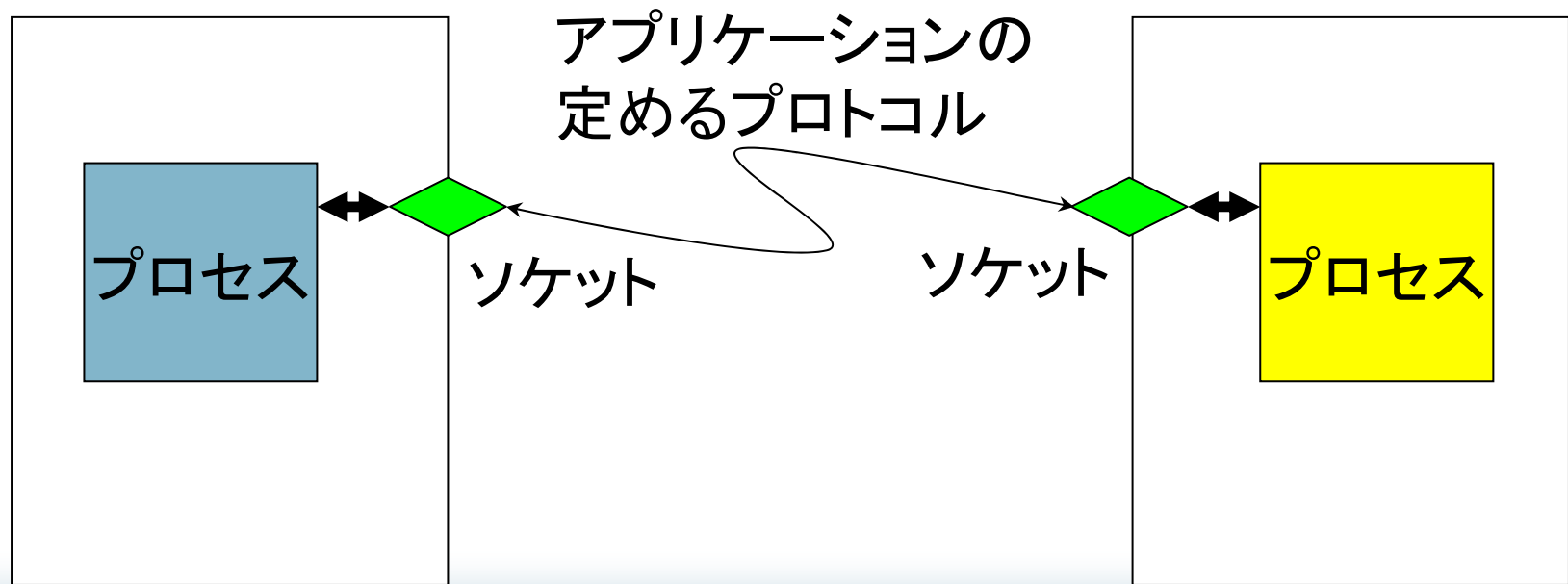
```
s = socket(...);  
bind(s, addr+port);  
listen(s, n);  
new_s = accept(s);  
send/recv(new_s, ...);
```

◆ クライアント:

```
s = socket(...);  
connect(s, addr+port, ...);  
send/recv(s, ...);
```



ネットワークアプリケーションの 典型的構成



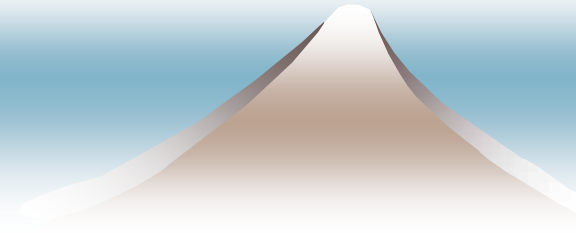
まとめ知識

◆ ps -ef

- すべてのプロセスを表示

◆ netstat -a

- 現在使われているソケットの状態を表示
 - 待機中(LISTEN)
 - 接続中(ESTABLISHED)

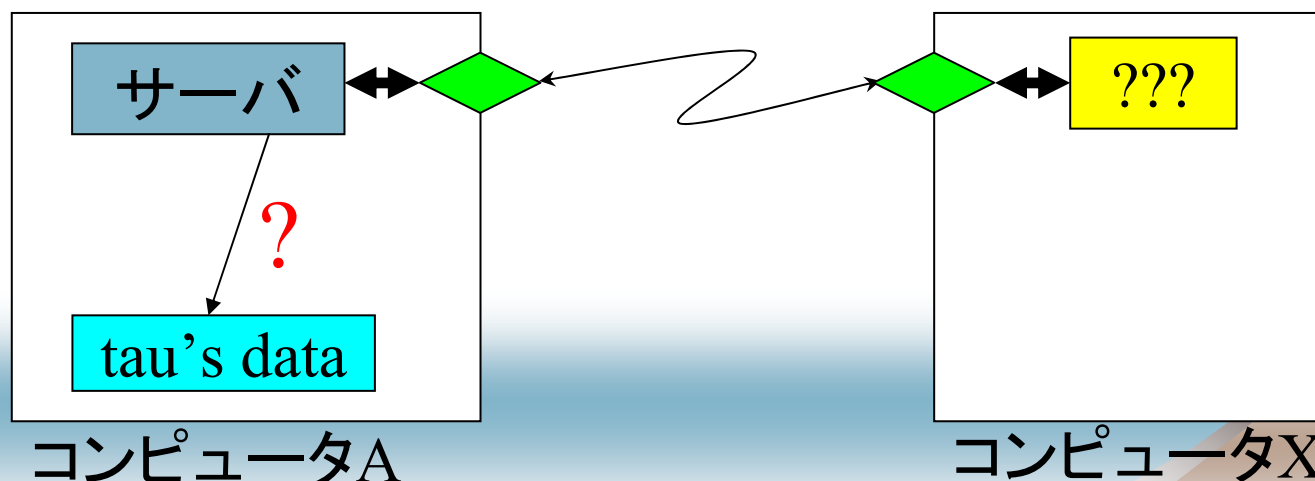


インターネットとセキュリティ (1)

- ◆ ソケットに対するアクセス制御は、OSには組み込まれていない
 - acceptしているソケットには誰でもconnectできる
 - connectした相手のIPアドレス、ポートなどは(IPパケット中に書かれているものを)知ることができる
 - しかし、相手プロセスのユーザIDなどを知る機構は組み込まれていない
 - 全世界のユーザを管理・把握することはできない

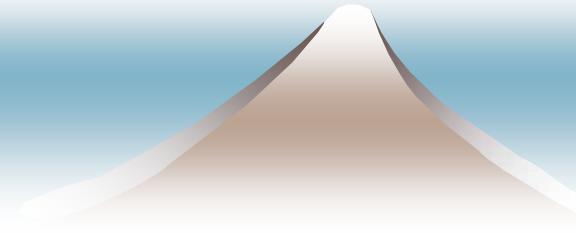
インターネットとセキュリティ (2)

- ◆ 現在のOSにはインターネット越しのユーザに対する保護・アクセス制御の概念はない
- ◆ アクセス制御はアプリケーションの役目



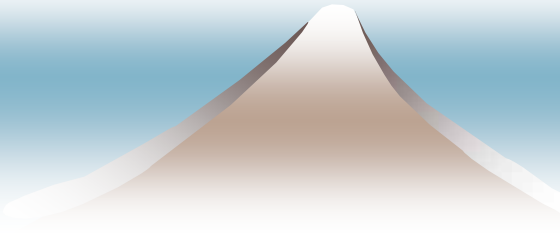
インターネットとセキュリティ (3)

- ◆ ひとたび計算機がインターネットに接続すれば, LANを流れるデータは容易に傍受可能



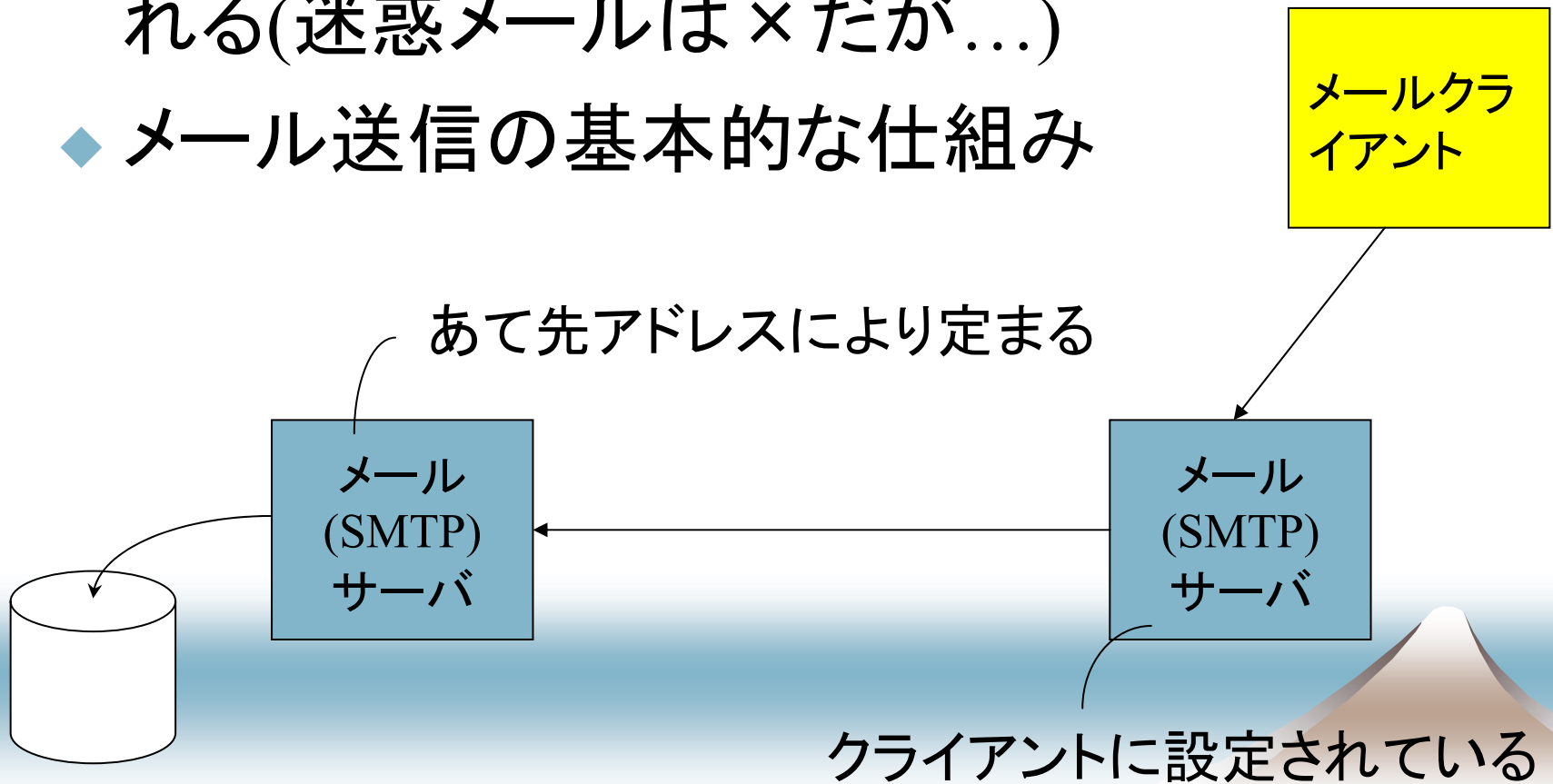
ネットワークアプリケーションの アクセス制御の実例

- ◆ アプリケーションごとに異なる, アクセス制御の方針
- ◆ それを実現するための, アプリケーションの構成



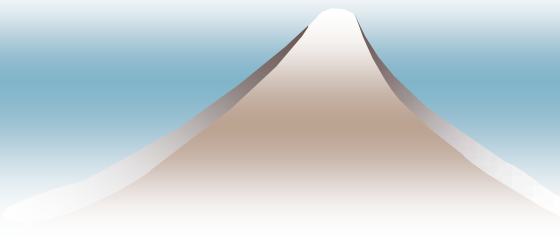
メールの送信

- ◆ 方針: 誰でも誰へでもメールを送れる(迷惑メールは×だが...)
- ◆ メール送信の基本的な仕組み



通常のSMTPサーバのアクセス 制御

- ◆ 同一LANからの送信要求は許可
- ◆ 受信(自分宛のメール)は無条件で許可
 - 送信者の身元確認(認証)は行われない



リモートログイン

◆ 基本方針

- ローカルユーザ X と同一人物(と思われる人物)からの接続を受け付ける
- その人に、ローカルユーザ X と同一の権限を与える

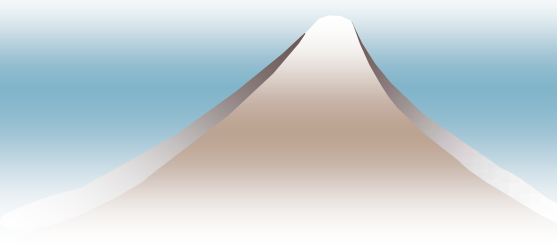
◆ メール送信と異なり、**認証が必須**

遠隔ユーザの認証

◆ パスワード認証

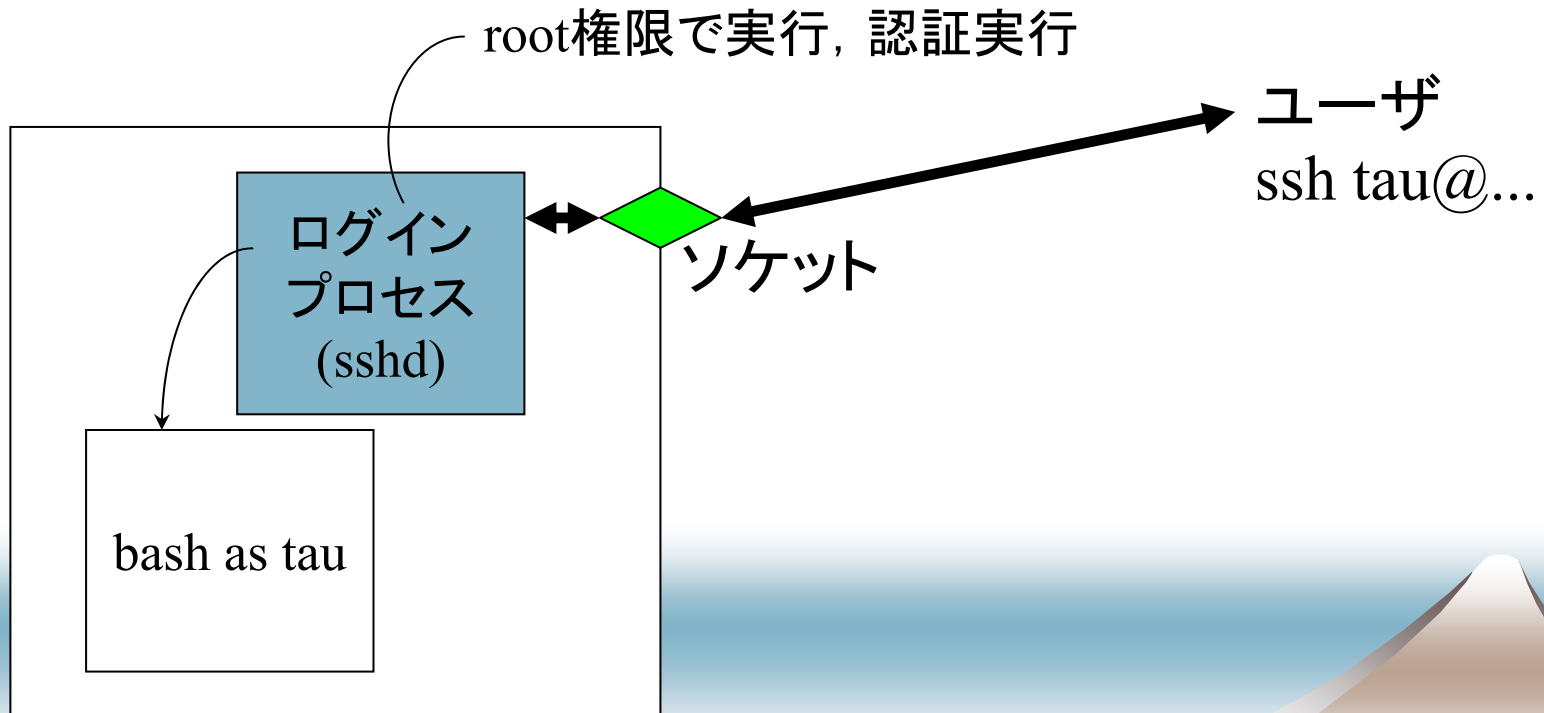
- クライアントがサーバへ、ユーザ X のパスワードを送信
- SSH, RSH

◆ 公開鍵認証

- サーバに保管してある公開鍵と、クライアントに保管してある秘密鍵が、対応する鍵の対であることを検証する
 - SSH, PGP
- 

認証後の処理

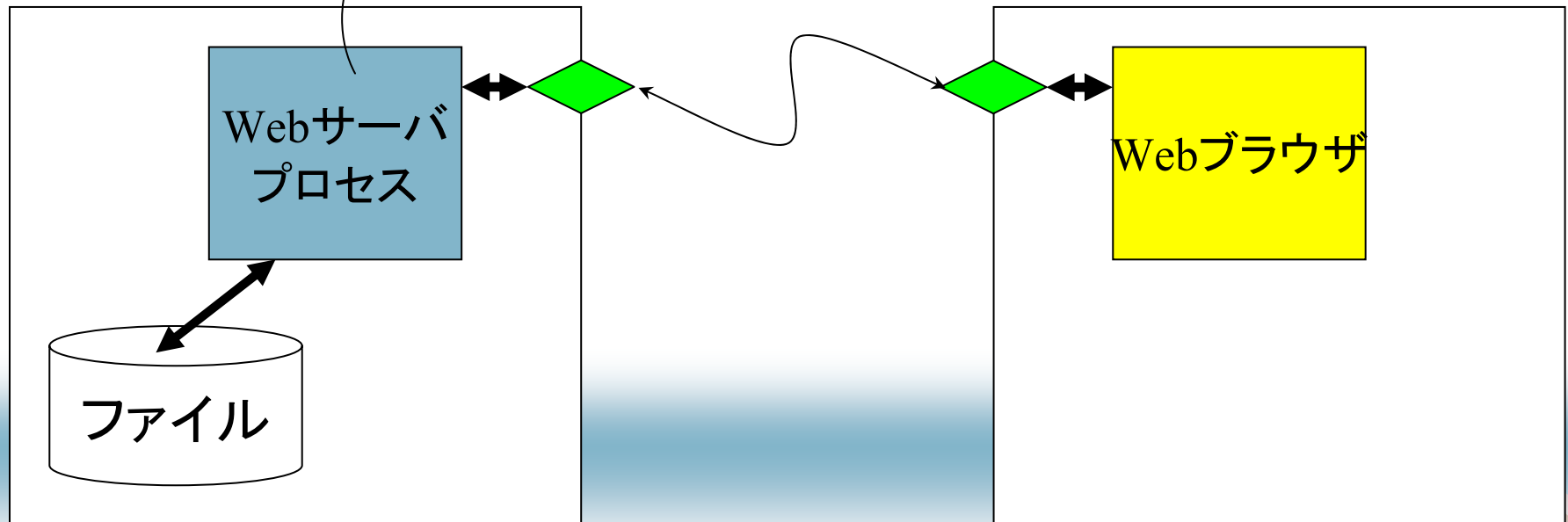
- ◆ 認証成功後, 要求されたユーザに成りすま
す(setuid)



(一昔前の) Web

◆ 認証不要

- サーバの方針: 誰からのアクセスも許可
 - クライアントの方針: ページを画面に表示するのみ
- 適当なユーザID



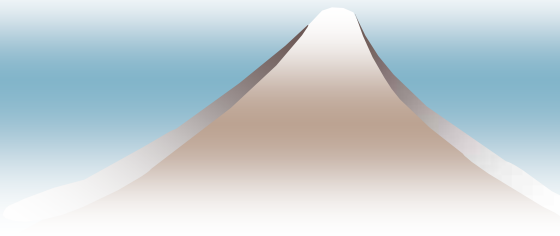
現在のWeb

◆ サーバ側

- 買い物, 銀行などの個人データへのアクセスを伴うアプリケーション

◆ クライアント側

- サーバから送られたscript (プログラム)の実行による動的な(見栄えの良い)ページの表示



Webにおけるアクセス制御

◆ クライアントの認証

- Webサーバに組み込まれた基本認証(パスワードによるページの保護)
- その他の各Webアプリケーション(CGI)ごとの認証

◆ サーバの認証

- IPアドレスによる認証
- 公開鍵(証明書)による認証

ネットワークセキュリティ(まとめ)

- ◆ 多くの部分がOSの守備範囲外
- ◆ 遠隔ユーザの認証, それに基づくアクセス制御を**正しく**実行するのは大部分が(OSではなく)サーバアプリケーションの役目
- ◆ Unixはrootに,
 - ほとんどのファイルへのアクセス権限
 - 他のユーザになりすます権限を与え, あとはアプリケーションに任せる

注: ネットワークでよく使われているアクセス制御

◆ LANへの接続の制御

- 無線LAN WEP key
- ダイアルアップサーバへの認証

◆ ルータでのアクセス制御

- 〈ソースIP, 宛先IP + ポート〉に対して「許可・不許可」を設定

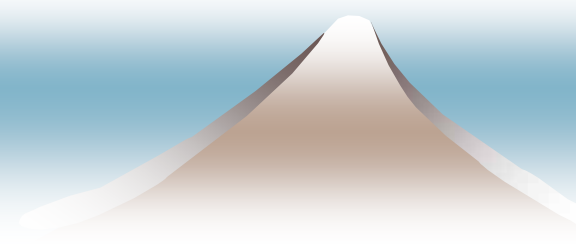
◆ ホストでのアクセス制御

- TCP wrapper
 - インターネットファイアウォール
- 



おまけ: 世の中を騒がせている 「セキュリティ騒動」のパターン(1)

- ◆ 例: XXXXのバッファオーバーフロー脆弱性
 - XXXX : ネットワークサーバ(ssh, Window file共有, sendmail, etc.)
 - root権限で実行中にバッファオーバーフローにより, 任意の命令列が実行される
 - 狙いはshellを実行する
- ◆ 考察: なぜroot権限で走る必要があるのか?



世の中を騒がせている「セキュリティ騒動」のパターン(2)

- ◆ Internet Explorerのバグで情報がネットワークに流出
 - ブラウザはローカルユーザ権限で実行
 - ブラウザは(カッコいいページを表示するため)ダウンロードされたscriptプログラムを勝手に実行する
 - ブラウザ自身がアクセス制御をしなければ、直ちにローカルユーザのデータは丸見え

世の中を騒がせている「セキュリティ騒動」のパターン(2) 続き

- ◆ そこでブラウザは「どのサイトからのscriptはどのデータにアクセスしてよいか」という独自のアクセス制御を「自前で、事細かに」実装する
- ◆ そこに間違い(バグ)があるとたちまち情報流出の危険ができる