

OpenSSHの ちよつと ディープな話

2014/04/18 JANOG 33.5 @togakushi

自己紹介

- 名前：かわもと
- Twitter ID : @togakushi
- 検索：ssh力をつけよう



SSHとはなんぞや？

- プロトコルの総称
 - The Secure Shell
 - RFCで定義されている（全部で17）
- 安全にリモートホストへ接続する手段のひとつ
- OpenSSHはSSHのフリーな実装

OpenSSHのバージョン

- 2014/04/18現在 v6.6
 - クライアント
 - 4.3 簡易VPN
 - 5.1 視覚的なホスト鍵の表示
 - 5.3 netcat mode
 - 5.7 ECDSAの追加
 - 6.5 ED25519の追加、特定条件下で設定の適応
 - サーバ
 - 4.3 簡易VPN
 - 4.4 強制コマンド実行、条件分岐
 - 4.8 ChrootDirectory
 - 6.2 複数要素認証、認証鍵コマンド

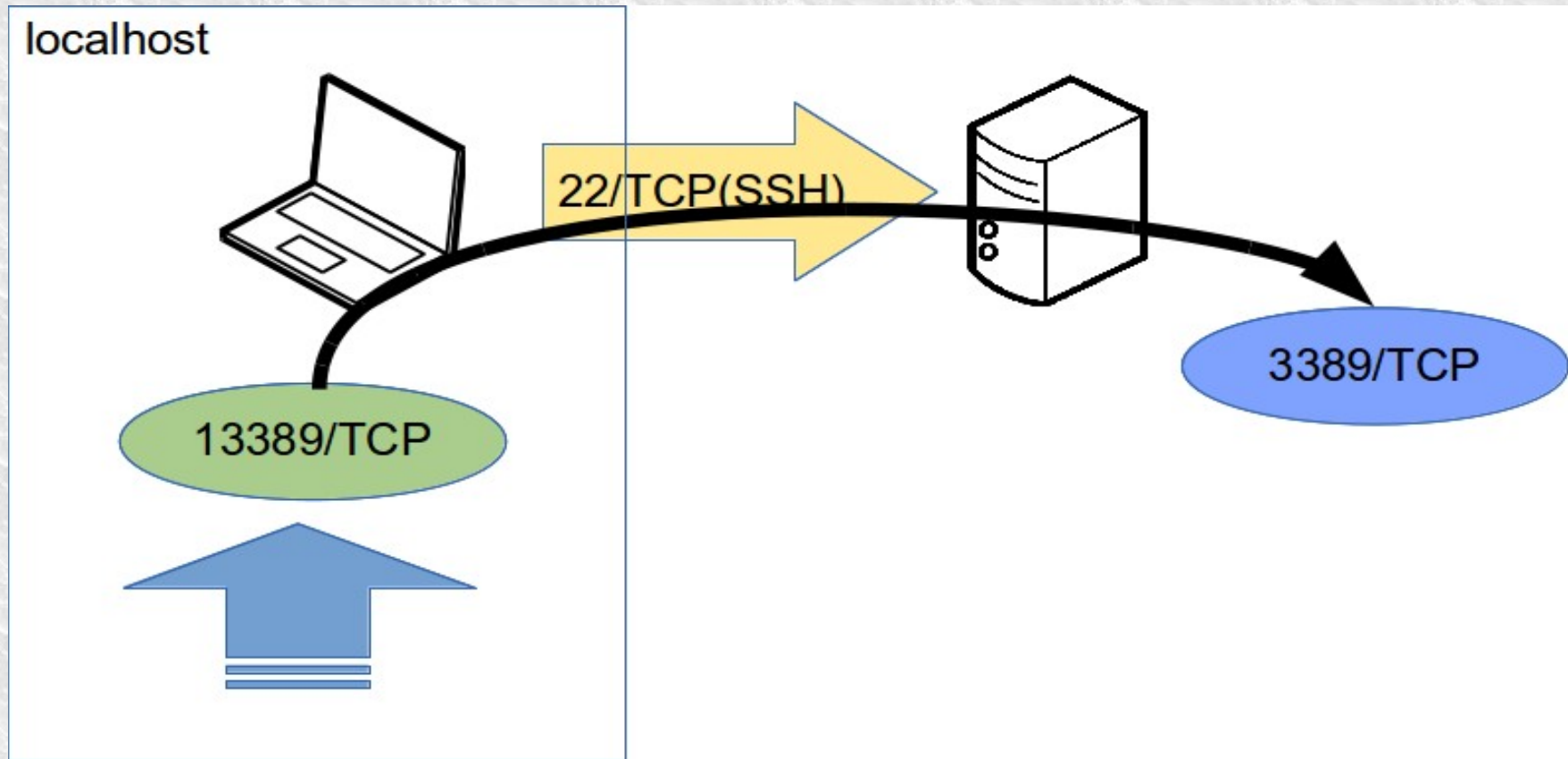
OpenSSHのオプション

- sshのコマンドラインオプション：44個
 - 1246ACEDFIKMLONQPSRTWVYXacbegfikmlonqpstwvy
x
- scpのコマンドラインオプション：21個
 - 12346BCFPScdfilopqrtv
- ssh-keygenのオプション：47個
 - ACBDGFIHKJMLONQPSRTWVXZacbegfihkjmlonqpsrutv
yxz
- -oで指定できるオプション：79個

ログインしてペチペチ
コマンド打っただけが
SSHじゃないよ！！

TCP PortForward

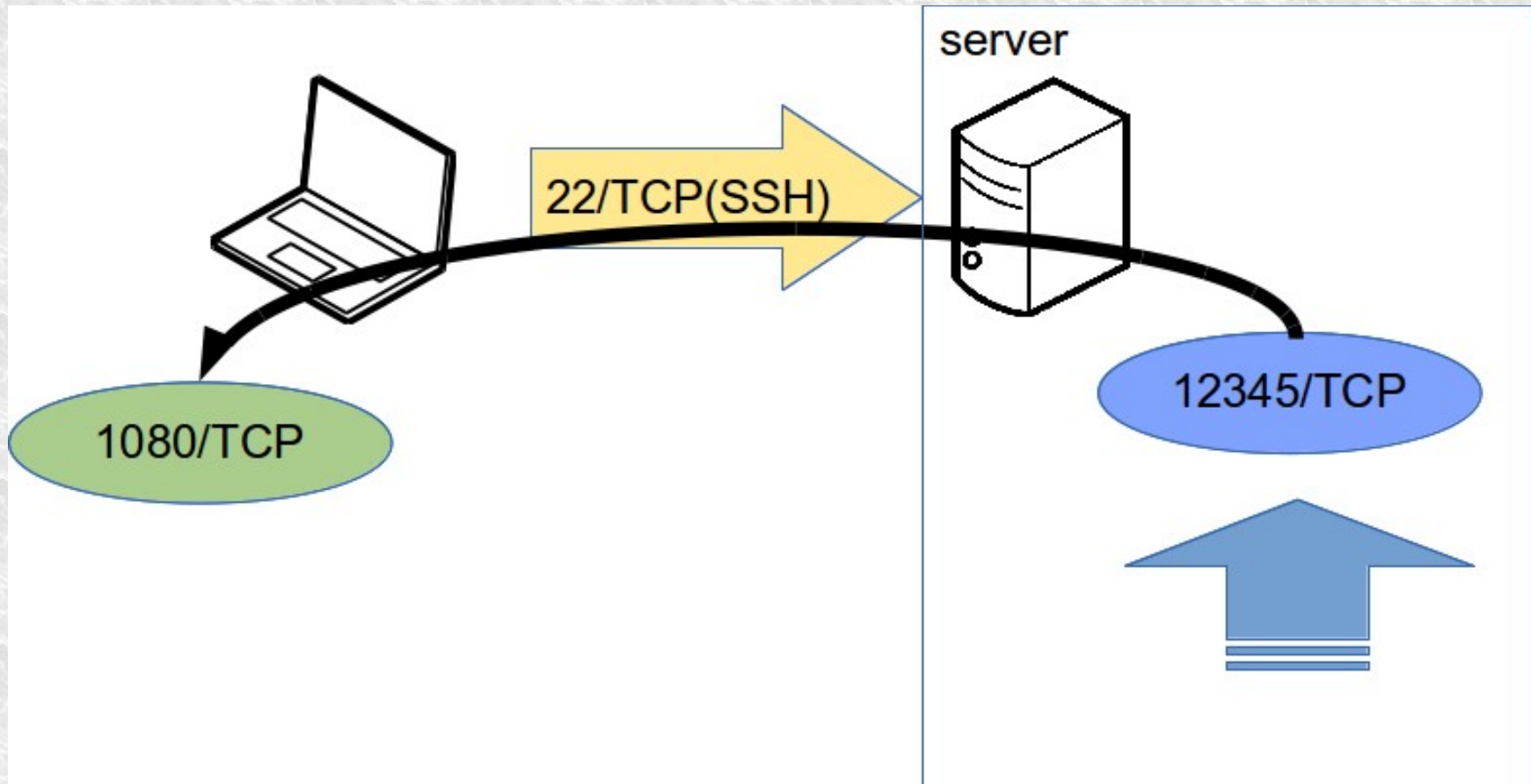
- ローカル



```
% ssh -L[localhost:]13389:転送先:3389 server
```

TCP PortForward

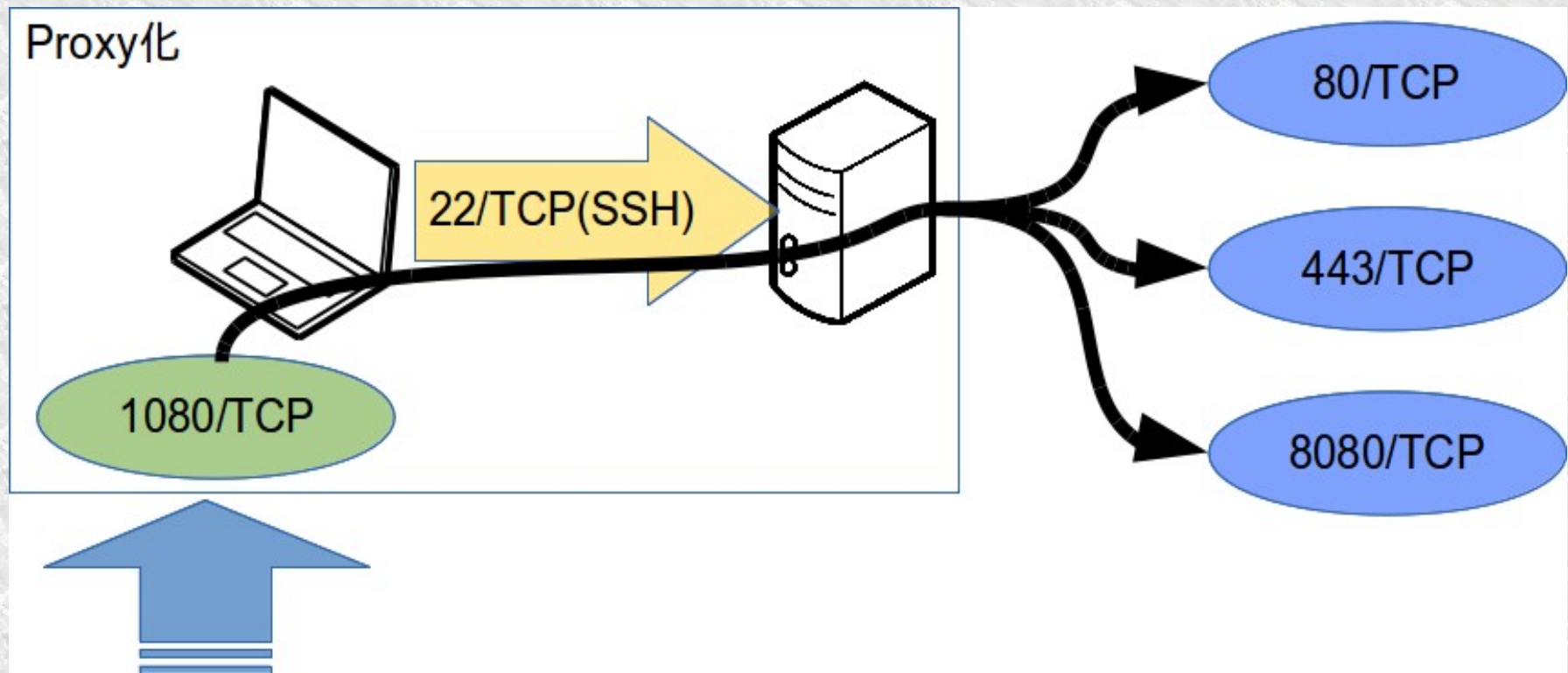
- リモート



```
% ssh -R[localhost:]12345:転送先:1080 server
```


TCP PortForward

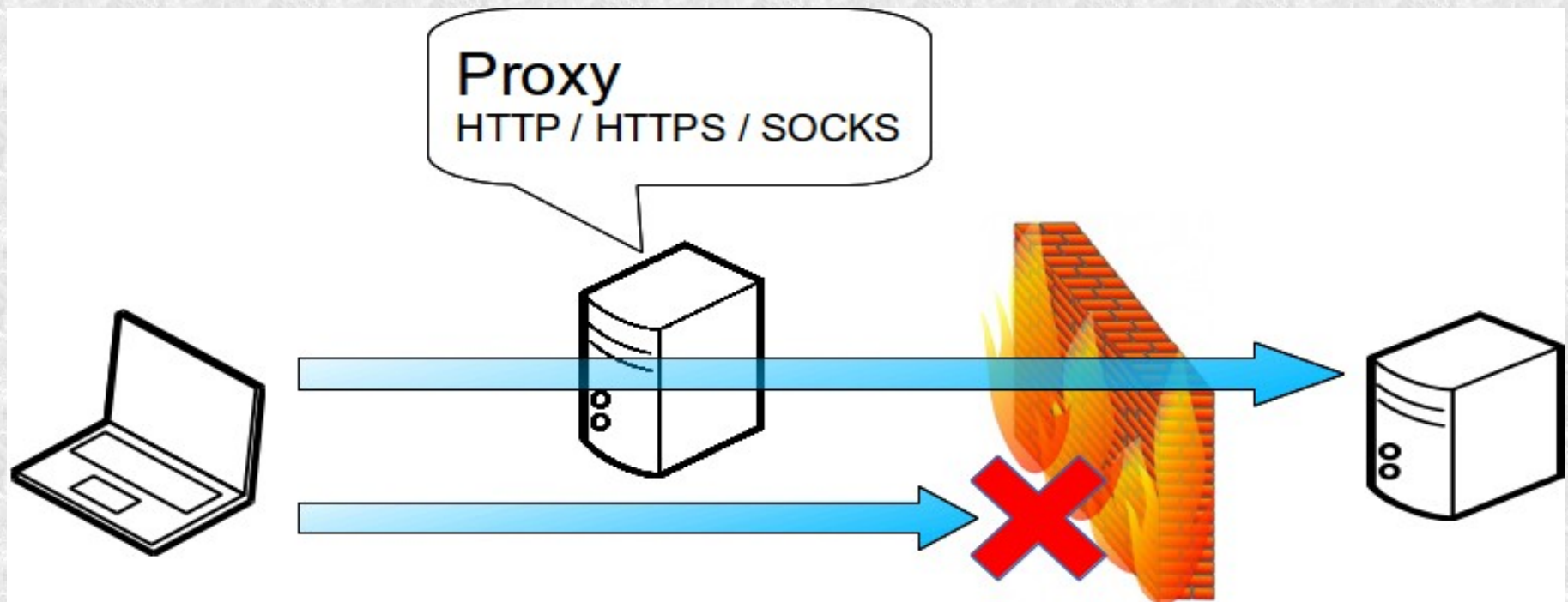
- ダイナミック



```
% ssh -D[localhost:1080] server
```

ProxyCommand

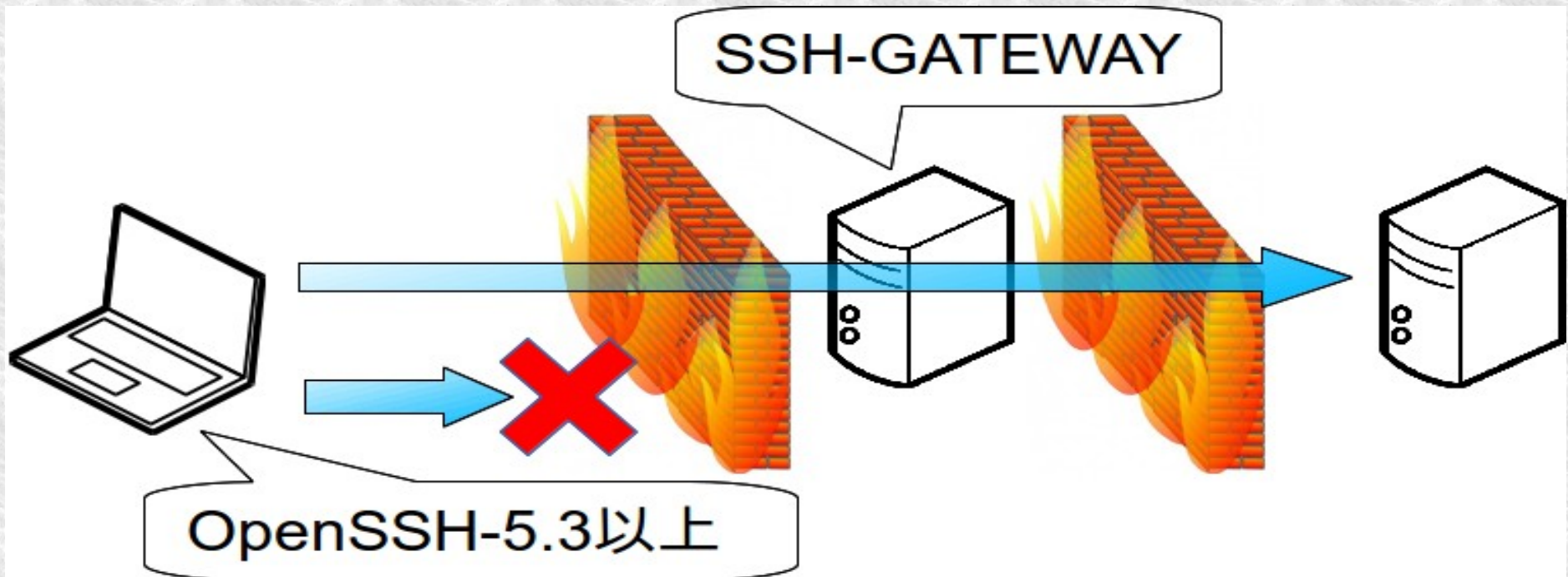
- Proxyを経由して接続



```
% ssh -oProxycommand='nc -X connect proxy:port' server
```

netcat mode

- SSH-GATEWAYをProxyの代わりに
 - GATEWAYのsshdのバージョンは不問



```
% ssh -oProxycommand='ssh -W %h:%p sshgw' server
```

ControlMaster

- セッション情報を共有
 - 未使用時

```
[username@centos6 ~]$ w
```

```
19:34:08 up 11:36, 2 users, load average: 0.00, 0.03, 0.05
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
username pts/0    192.168.122.1    19:31    2:26   0.00s  0.00s  -bash
username pts/1    192.168.122.1    19:31    0.00s  0.02s  0.00s  w
```

```
[username@centos6 ~]$ ss -tn
```

State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port
ESTAB	0	0	192.168.122.128:22	192.168.122.1:55759
ESTAB	0	0	192.168.122.128:22	192.168.122.1:55760

ControlMaster

- セッション情報を共有
 - 使用時

```
[username@centos6 ~]$ w
```

```
19:40:07 up 11:42, 3 users, load average: 0.00, 0.00, 0.01
USER      TTY      FROM          LOGIN@      IDLE        JCPU        PCPU        WHAT
username pts/0    192.168.122.1 19:39      0.00s      0.01s      0.00s      w
username pts/1    192.168.122.1 19:39     45.00s      0.00s      0.00s     -bash
username pts/2    192.168.122.1 19:39     42.00s      0.00s      0.00s     -bash
```

```
[username@centos6 ~]$ ss -tn
```

State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port
ESTAB	0	0	192.168.122.128:22	192.168.122.1:55844

ControlMasterのデメリット

- 最初のセッション(マスターセッション)を切断できない
 - マスターセッションが切れると共有しているセッションがすべて切れる
 - 転送機能(TCP Port/X/SSH Agent)はすべてマスターセッションでコントロール
- 共有セッションがない状態でもマスターセッションは残り続ける
 - 5.6でControlPersistが追加
 - 共有セッションがない場合にタイムアウトさせる

ControlMasterの使い方

- マスターセッションの開始(デフォルト無効)
 - `% ssh -oControlMaster=yes -oControlPath=/path/to/%h-%p-%r server`
- セッションを共有する場合
 - `% ssh -oControlMaster=no -oControlPath=/path/to/%h-%p-%r server`

ControlPathが設定されていない場合は無効(通常の接続になる)

毎回指定するのはメンドクサイので

- ~/.ssh/config

```
Host *  
    ControlMaster    auto  
    ControlPath      ~/.ssh/ControlMaster-%r-%h.%p  
    ControlPersist   30
```


escape sequences

- 通信中のセッションをコントロール
 - エンター直後(改行のみの入力)の「~」

```
[username@centos6 ~]$ ~?
```

```
Supported escape sequences:
```

- ~. - terminate session
- ~B - send a BREAK to the remote system
- ~R - Request rekey (SSH protocol 2 only)
- ~# - list forwarded connections
- ~? - this message
- ~~ - send the escape character by typing it twice

```
(Note that escapes are only recognized immediately after newline.)
```

escape sequences

- SSHプロンプト
 - 「~」の後に「C」

```
[username@centos6 ~]$ ~C
```

```
ssh> help
```

```
Commands:
```

```
-L[bind_address:]port:host:hostport
```

```
Request local forward
```

```
-R[bind_address:]port:host:hostport
```

```
Request remote forward
```

```
-D[bind_address:]port
```

```
Request dynamic forward
```

```
-KL[bind_address:]port
```

```
Cancel local forward
```

```
-KR[bind_address:]port
```

```
Cancel remote forward
```

```
-KD[bind_address:]port
```

```
Cancel dynamic forward
```

```
!args
```

```
Execute local command
```

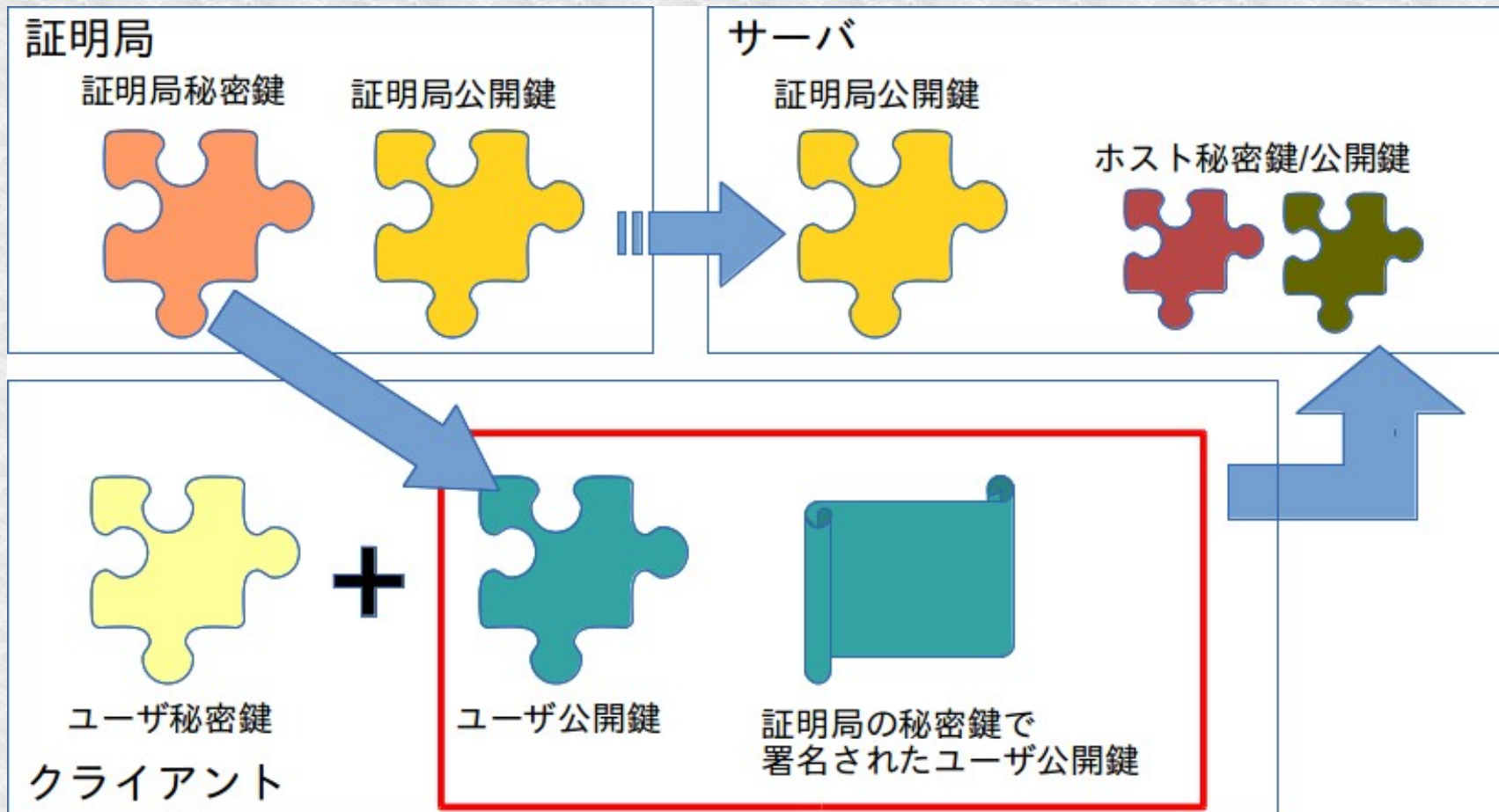
公開鍵のオプション

- 公開鍵の先頭部分にオプションが記述できる
 - サーバ側で強制的にオプションの有効化/無効化が可能
 - クライアントのコマンドラインオプションより優先される
- 公開鍵のフォーマット
 - `ssh-rsa AAAA(... 省略 ...)AAA== comment`
- オプションの追加
 - `no-port-forwarding,no-X11-forwarding,command="/usr/local/sbin/backup.sh -a" ssh-rsa AAAA...`
 - スペースを含む場合はクォートで囲む
 - 区切り(,)の前後のスペースは含めない

証明書による認証(5.4以降)

- 証明局の秘密鍵で署名したユーザ公開鍵を簡易的な証明書としてユーザ秘密鍵とセットで使う
 - 署名した秘密鍵のペア(公開鍵)を持つサーバに対して、証明書とユーザ秘密鍵を持つクライアントを信用する
 - サーバにユーザの公開鍵を保存する必要がなくなる
 - 証明書による制限が可能
 - 有効期限
 - 失効
 - オプション

証明書による認証(5.4以降)



二要素認証(6.2以降)

- 認証メソッドを2種類以上強制

```
PubkeyAuthentication    yes
PasswordAuthentication  yes
AuthenticationMethods    publickey,password
```

- 公開鍵認証に成功した後にパスワード認証を求める
 - クライアント側では認証処理がループしているように見える

まとめ

- OpenSSHはバージョンが上がっていくにつれて激しく機能追加してくる
- どんどん便利に、安全になっていくので積極的に使うべし

続きはWebで

- その他検証項目まとめ
 - OpenSSH 6.2 で追加された機能を試す
 - <http://togakushi.bitbucket.org/build/html/openssh-6.2.html>
 - OpenSSH 6.5 で追加された機能を試す
 - <http://togakushi.bitbucket.org/build/html/openssh-6.5.html>
- 過去の資料
 - SSH力をつけよう
 - <http://www.slideshare.net/tohakushi/ssh-13118950>
 - SSH力をつかおう
 - <http://www.slideshare.net/tohakushi/ssh-15554045>
 - sshdのお話
 - <http://www.slideshare.net/tohakushi/sshd>

*** exit ***