LinuC

寺子屋ＩＴライセンス
TERAKOYA IT LICENCE

LinuC試験から学ぶセキュリティ入門

2019/10/10

株式会社クロスパワー
寺子屋ＩＴライセンス
森　遼太

LPI-JAPAN

講師プロフィール

■ 株式会社クロスパワー

- □ 企業のシステム導入をサポートするシステムインテグレーション
- □ AWSのサービスに注力したAWSクラウドサービス
- □ エンジニア育成を目的とした寺子屋ITライセンス

の3つのビジネスを軸としています。
アプリ開発やクラウド導入サポートだけではなく、オープンソースの開発やサービス同士を組み合わせ、より利用しやすいサービスにして多彩なサービスをご提供しています。

■ 講師

- □ 他業種から株式会社クロスパワーに入社後、サーバの運用構築業務を経験し、現在は未経験から技術者へ転向した経験をいかして、自社サービスの１つである、「寺子屋ＩＴライセンス」にて人材育成、教育を担当しております。

本日のセミナー内容

1. LinuC試験について

i. LinuC試験の概要
ii. レベル1の出題範囲について

2.セキュリティについて

i. セキュリティ管理業務の実施
ii. ホストのセキュリティ設定
iii. 暗号化によるデータの保護

LINUC試験について

LinuC試験の概要（3つのレベル）

寺子屋ITライセンス
TERAKOYA IT LICENCE

LinuCレベル3
Mixed Environment
（300試験）

Linux、Windows、Unixが混在するシステム

LinuCレベル3
Security
（303試験）

セキュリティレベルの高いコンピュータシステム

LinuCレベル3
Virtualization & High Availability
（304試験）

仮想化システムや高可用性システム

LinuCレベル2（201試験／202試験）
Linuxサーバやネットワークを含むシステムの構築・運用・保守

Linuxのシステムデザイン、ネットワーク構築において、企画、導入、維持、トラブルシューティング、キャパシティプランニングができるエンジニアであることを証明できます。

LinuCレベル1（101試験／102試験）
Linuxシステムの構築・運用・管理

実務で必要なLinuxの基本操作とシステム管理が行えるエンジニアであることを証明できます。

LinuC

LinuC試験の概要（受験について）

■CBT（コンピュータベーストテスト）試験

- マウスを使った選択問題が殆ど（キーボード入力問題も多少ある。）
- 問題は時間内であれば見直し可能

■試験時間90分60問

- 試験後のアンケートに5分の時間を要するため、試験問題を解く時間は実質「85分」
- 試験終了と同時に合否が分かる

■有意性の期限

- 認定の有効期限はないが、「有意性の期限」が5年が設定されている。
  認定の有意性を維持するためには「認定日から5年以内」に再認定が必要。

レベル1の出題範囲について

レベル1の出題範囲について

寺子屋ITライセンス
TERAKOYA IT LICENCE

101試験　出題範囲	
主題101：	システムアーキテクチャ
主題102：	Linuxのインストールとパッケージ管理
主題103：	GNUとUnixのコマンド
主題104：	デバイス、Linuxファイルシステム、ファイルシステム階層標準

102試験　出題範囲	
主題105：	シェル、スクリプト、およびデータ管理
主題106：	ユーザインターフェイスとデスクトップ
主題107：	管理業務
主題108：	重要なシステムサービス
主題109：	ネットワークの基礎
主題110：	セキュリティ

LinuC

主題110：セキュリティ

110.1	セキュリティ管理業務の実施	• システムを監査して、SUID/SGIDビットが設定されているファイルを探す • ユーザのパスワードおよびパスワードエージング情報を設定または変更する • nmapおよびnetstatを使用して、システムの開いているポートを見つける • ユーザのログイン、プロセス、メモリー使用量を制限する • **基本的なsudoの設定および利用方法**
110.2	ホストのセキュリティ設定	• シャドウパスワードおよびその機能について知っている • 使用していないネットワークサービスをオフにする • **TCPラッパーの役割について理解している**
110.3	暗号化によるデータの保護	• **基本的なOpenSSH 2クライアントの設定および利用方法** • **OpenSSH 2サーバのホストキーの役割について理解している** • 基本的なGnuPGの設定および利用方法 • SSHポートトンネル（X11トンネルを含む）について理解している

110.1　セキュリティ管理業務の実施
（基本的なSUDOの設定および利用方法）

ユーザについて

■管理者ユーザroot

Linuxには、rootという名前の管理者アカウントが存在する。
あらゆる管理作業をする権限を持っており、スーパーユーザとも呼ばれる。
rootユーザ以外のユーザは一般ユーザという。

■プロンプト…ログインすると表示される下記の行

```
[root@vm01 ~]#
```

root	ログインしているユーザ名
vm01	ログインしている端末のホスト名
~	現在のカレントディレクトリ
#	スーパーユーザ（一般ユーザは「$」表記）

rm –rf /　コマンド

「rm -rf /」とは

- rm…削除コマンド
- -rf…ファイル/ディレクトリ関係なくすべて消す
- /…ルートディレクトリ配下全て

⇒全てのファイルを消し去るコマンド

一般ユーザの場合は権限の問題により消せるものが少なく影響度はまだ小さいが、
rootユーザで実行するとありとあらゆるものを吹き飛ばすことが出来る。

root権限が必要な時だけrootユーザーとして作業する。

suコマンド、sudoコマンド

ユーザ環境の切り替え

su　[オプション]　[ユーザ名]

オプション	説明
－	環境変数を全て初期化する。 カレントディレクトリも切り替えたユーザーのホームディレクトリになる。

別のユーザ権限でコマンドを実行する

sudo　[オプション]　[コマンド]

オプション	説明
-u　ユーザ名	指定したユーザ権限でコマンドを実行。 -uオプションが指定されなかった場合、rootユーザ権限でコマンドを実行する。

sudoの設定方法

/etc/sudoersファイル　…　sudoが使用できるユーザ、コマンドを定義

-r--r-----. 1 root root 3907 11月 5 2016 /etc/sudoers

visudoコマンドによって設定を書き換える
　　viエディタが起動し、編集できる
　　保存終了時にエラーチェックをしてくれる

入力の例）
[root@c7test1 ~]# visudo

user01　ALL=(ALL)　　/sbin/reboot　　←　ファイル内に追記

sudoの実行例

[user01@c7test1 ~]$ reboot　←　**sudoなしでrebootを実行**
（略）
Failed to execute operation: Access denied
Must be root.　←　**rebootコマンドは一般ユーザでは実行できない**

[user01@c7test1 ~]$ **sudo reboot**
[sudo] password for user01:　←　**user01のパスワードを入力**
システムの再起動が実行される

[user01@c7test1 ~]$ **sudo useradd testuser**　←　**ユーザ作成するコマンドを実行**
[sudo] password for user01:　←　**user01のパスワードを入力**
ユーザー user01 は‘/sbin/useradd testuser’ を root として c7test1 上で実行することは許可されていません。すみません。　←　**sudoresに記載していないコマンドは実行できない。**

LinuC

実行許可されたコマンドの確認


```
[user01@c7test1 ~]$ sudo -l
[sudo] password for user01:
このホスト上でユーザー user01 に一致したデフォルト項目:
    !visiblepw, always_set_home, env_reset, env_keep="COLORS DISPLAY HOSTNAME
    HISTSIZE KDEDIR LS_COLORS", env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG
    LC_ADDRESS LC_CTYPE", env_keep+="LC_COLLATE LC_IDENTIFICATION
    LC_MEASUREMENT LC_MESSAGES", env_keep+="LC_MONETARY LC_NAME
LC_NUMERIC
    LC_PAPER LC_TELEPHONE", env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS
    _XKB_CHARSET XAUTHORITY", secure_path=/sbin¥:/bin¥:/usr/sbin¥:/usr/bin

ユーザー user01
    は次のコマンドをこのホスト上で実行できます:
    (ALL) /sbin/reboot
```

suコマンドの制限

本来は、適切な設定を行い動作確認ができたら、管理者パスワードだけで管理者特権を持つことが出来るsuコマンドは無効化することが推奨される。（※レベル2範囲）

/etc/pam.d/suファイル

```
#%PAM-1.0
auth            sufficient      pam_rootok.so
# Uncomment the following line to implicitly trust users in the "wheel" group.
auth            required        pam_deny.so     ←     ①追記
#auth           sufficient      pam_wheel.so trust use_uid
# Uncomment the following line to require a user to be in the "wheel" group.
auth            required        pam_wheel.so use_uid   ←     ②コメントイン
```

①で「suを無効にする」ことができ、②で[suの使用者を限定する]ことが出来る。

110.2　ホストのセキュリティ設定
（TCPラッパーの役割について理解している）

ホスト側のセキュリティ

■**ホスト自身でセキュリティを高める方法としていくつかの手法がある。**
（一例）

- **iptables**
  パケットフィルタリング
  入力（Input）、出力（Output）、転送（Forward）のルールセットに基づいて
  パケットの破棄or通過を決定する（LinuCレベル2）
- **firewalld**
  ファイヤウォールの仕組み。
  Redhat7ではiptablesに替わって採用されている。

ネットワークサービスのアクセス制御として、
「/etc/hosts.allow」と「/etc/hosts.deny」の２つのファイルを使った制限の方法がある。

/etc/hosts.allow（deny）ファイル

■/etc/hosts.allowファイル

このファイルに指定したホスト（IPアドレス）からのアクセスを**許可**する。

■/etc/hosts.denyファイル

このファイルに指定したホスト（IPアドレス）からのアクセスを**拒否**する。

■書式（共通）

サービス名：ホスト名（又はIPアドレス）

■使用出来るワイルドカードの例

ALL	全てのサービスまたはホスト
A EXCEPT B	B以外のA
LOCAL	localhostのように「.」を含まないホスト

/etc/hosts.allow（deny）ファイルの記載例

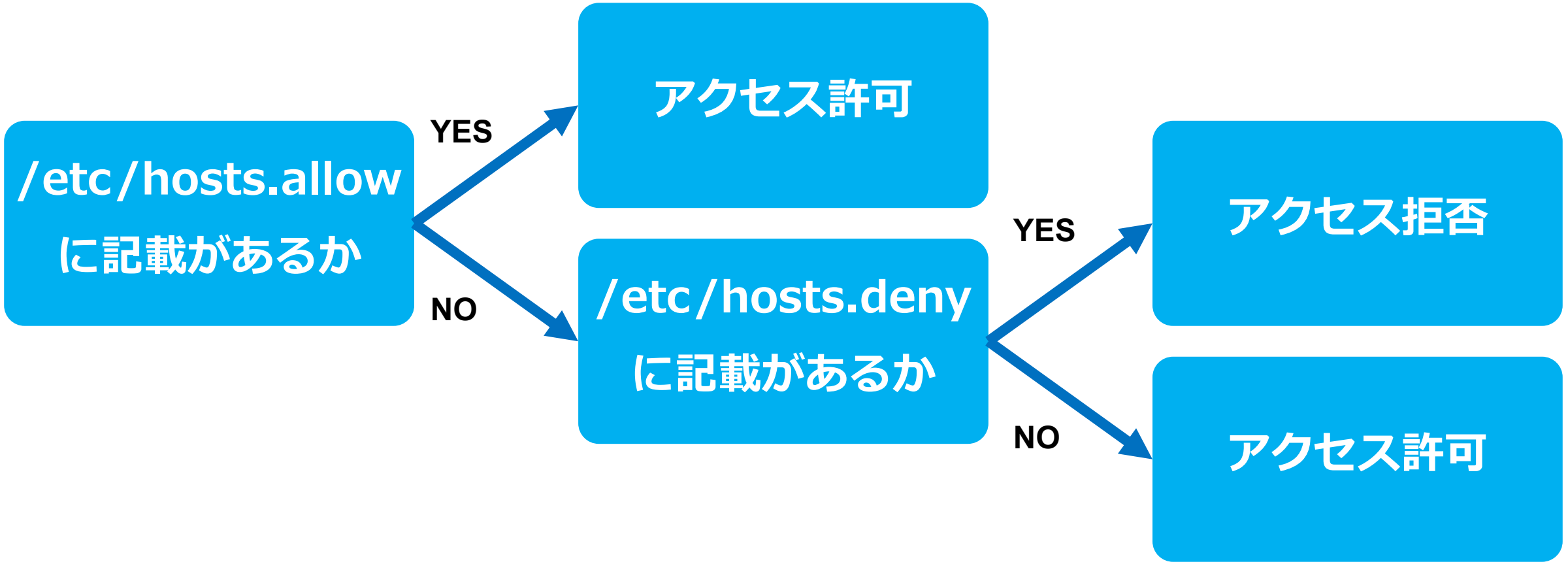
[root@c7test1 ~]# **vi /etc/hosts.deny**
（略）
ALL : ALL

**全てのホストからの
すべての接続を「拒否」する**

[root@c7test1 ~]# **vi /etc/hosts.allow**
（略）
sshd : 10.0.0.12（　←　許可するIPアドレス）

**記載したIPアドレスからの
SSH接続を「許可」する**

アクセス制御の流れ



スーパーサーバとTCP Wrapper

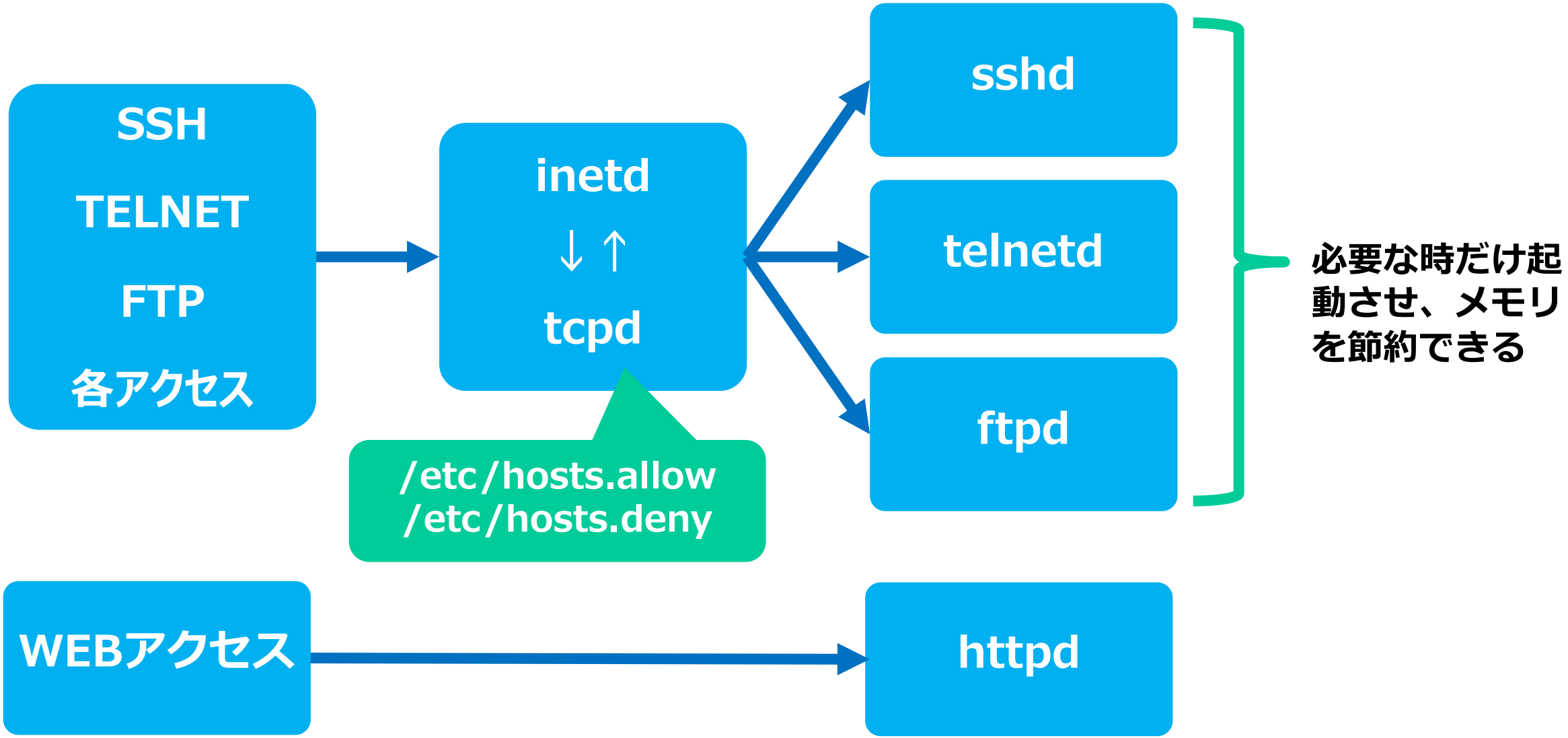
■「/etc/hosts.allow」と「/etc/hosts.deny」でIP制限できるのは
→sshdがTCPWrapperのライブラリを使用している

■TCP Wrapper
→スーパーサーバinetdで利用されるアクセス制御のプログラム

■inetd
複数のサービスを監視し、要望があればそのデーモンを起動するためのスーバーサーバ。もともとinted自体はアクセス制御機能を持っていないので、tcpwrapperと連携して動作する。

■inetdはサービス要求を受けるとそれをtcpd（tcpwrapperの本体）に渡してtcpdが許可/拒否を決定する。

inetdの動き



xinetd

■xinetd
inetdを発展させ、さらに安全性を高め、より詳細な設定をできるようにした

■inetdとの違い

- xinetdの設定ファイルの記述のみで、アクセス制御をかけることができる。（inetd＋TCP Wrapperの機能）
- TCPだけでなく、UDPのアクセス制御もできる。
- xinetdではサービス専用のアカウントを作成することができ、そのアカウントの権限でデーモンを起動することができる。（inetd経由でデーモンを起動するユーザはrootである必要があった。）

110.3　暗号化によるデータの保護

（基本的なOPENSSH 2クライアントの設定および利用方法）
（OPENSSH 2サーバのホストキーの役割について理解している）

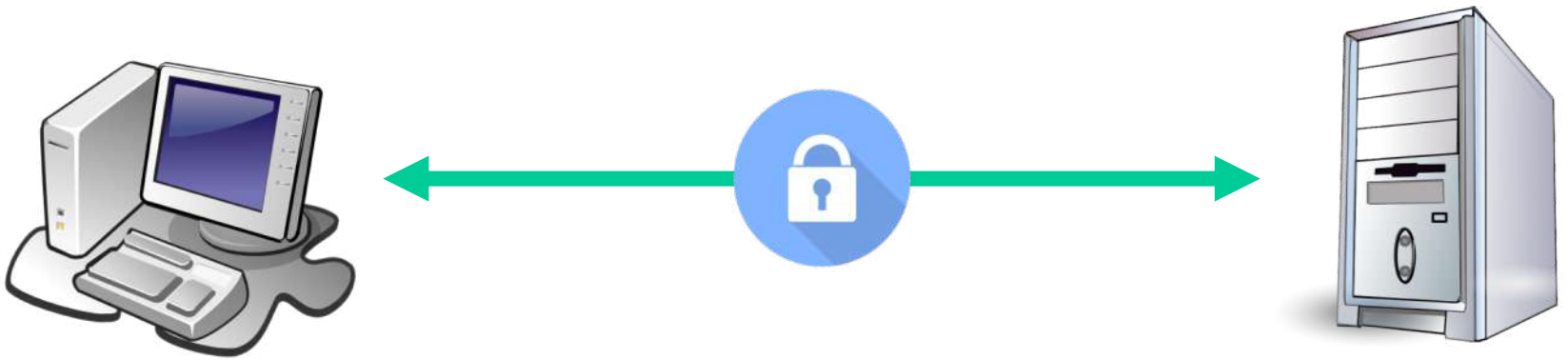
telnetとssh

telnet

- ネットワークに接続された機器を遠隔操作するために使用するプロトコル。
- パスワード情報を含め全てのデータが暗号化されずに送信される。

SSH

- Secure Shellの略で、ネットワークに接続された機器を遠隔操作するために使用する。
- パスワード情報を含めて全てのデータが暗号化されて送信される。



共通鍵暗号と公開鍵暗号の違い

■共通鍵暗号とは

- 暗号用と復号用に同じ鍵を用いる方式
- 暗号化されたデータを復号化する前に鍵を渡す必要がある。
- 相手方へ鍵を届けるために、安全な受け渡しを実施しなければならない。
- 処理が軽い

■公開鍵暗号とは

- 暗号用の鍵と復号用の鍵が異なる方式
- 公開鍵で暗号化したものはペアとなっている秘密鍵でしか複合できない。
- 接続相手別の鍵を生成する必要がない。
- 共通鍵と比べて処理が遅い

共通鍵暗号と公開鍵暗号のイメージ

■共通鍵暗号

鍵の配布

鍵を使用して暗号化

■公開鍵暗号

公開鍵の配布

公開鍵を使用して
暗号化

SSH接続の流れ

■ホスト認証

- 接続先のホストがなりすましによる偽物ではないか確認をおこなう。

■ユーザ認証

- 接続元のクライアントが信頼できる接続元かどうかを確認する。

ホスト認証

■初めて接続するホストの場合

[root@c7test1 ~]# **ssh [接続するホスト（以下の例では「10.0.0.12」）]**

The authenticity of host '10.0.0.12 (10.0.0.12)' can't be established.
ECDSA key fingerprint is 5a:d7:c7:20:9a:89:58:53:c4:c3:0a:1f:83:71:a8:77.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.0.12' (ECDSA) to the list of known hosts.

■公開鍵が書き込まれるファイル

~/.ssh/known_hosts ← ユーザ単位
/etc/ssh/ssh_known_hosts ← すべてのユーザで共通

■以前接続したことのあるホストの場合

/known_hostsに書き込まれている情報と照らし合わせて接続の可否を決定する

登録される公開鍵の確認

■**登録される公開鍵の内容**

[root@c7test1 .ssh]# **cat known_hosts**
10.0.0.12 ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAA
ABBBLGEgOTs5Ar8yQqCSkoWnz68yS2N6t2oVPVn++yYQnKFgn
uq3fafOZ+iWblSRtc+Ejy4cwvNuvZZtzokgbY5gj8=

■**接続先が持つ公開鍵を確認**

[root@vm01 ~]# **cat /etc/ssh/ssh_host_ecdsa_key.pub**
ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAA
ABBBLGEgOTs5Ar8yQqCSkoWnz68yS2N6t2oVPVn++yYQnKFgn
uq3fafOZ+iWblSRtc+Ejy4cwvNuvZZtzokgbY5gj8=

同じ情報が書き込まれていることが確認出来る

クライアント認証

■クライアント認証
クライアント（接続元）が信頼できる接続元かどうかを確認する
一番簡単な認証の方法は「パスワード認証」

- ユーザ／パスワードを入力して認証させる

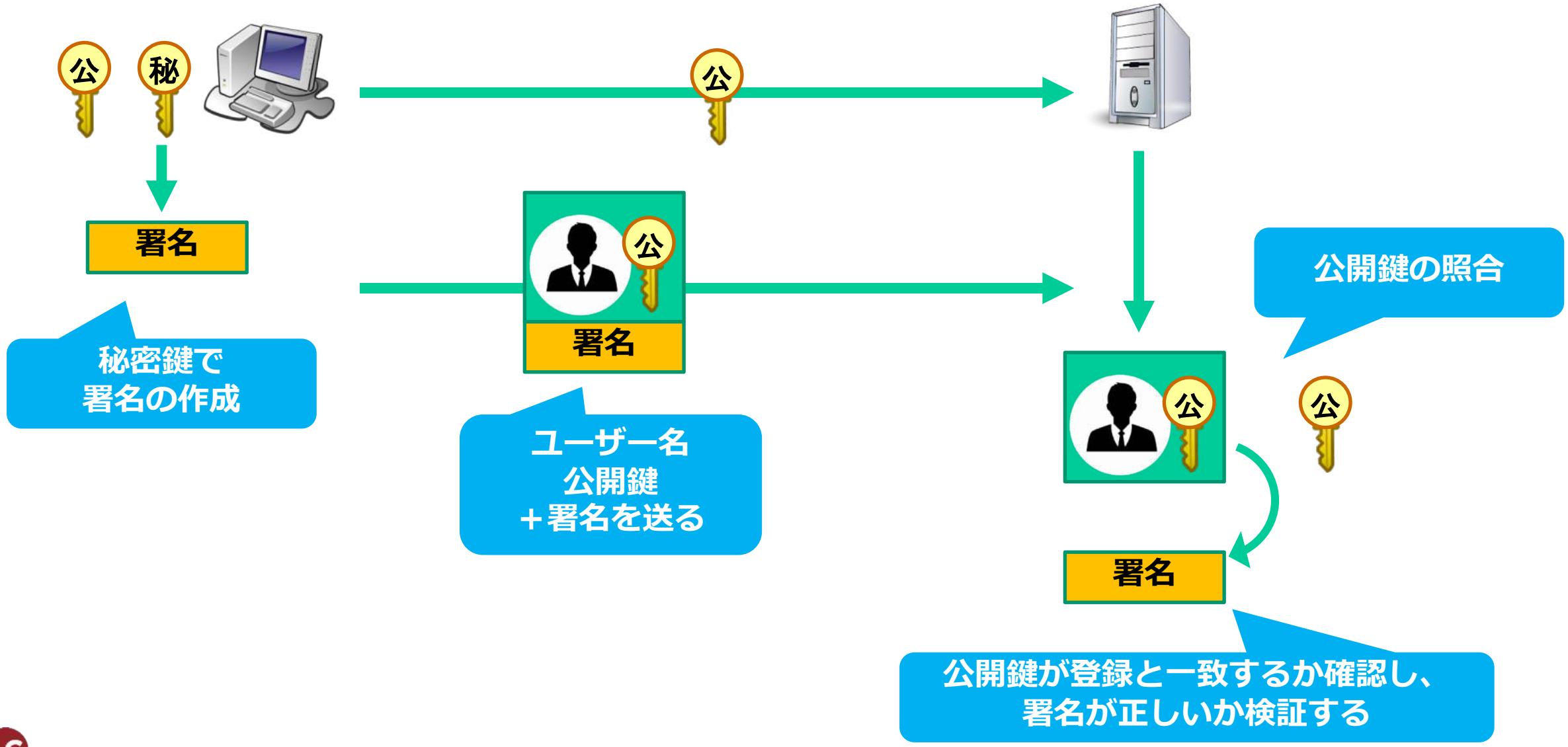
それ以外に「公開鍵認証」がある

■公開鍵認証設定の簡単な流れ

① クライアント側で公開鍵と秘密鍵のペアを作成する。
② クライアントからサーバに公開鍵を転送する。
③ サーバ側で貰った公開鍵をauthorized.keysに登録する。

④ クライアントからサーバへsshコマンドを使って接続する。

接続時の動き



①公開鍵と秘密鍵のペアを作成

■キーペアを作る手順

```
[root@c7test1 ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):  ← パスフレーズの入力
Enter same passphrase again:  ← パスフレーズの再入力
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
57:51:27:86:b0:12:71:af:89:06:37:79:ff:ac:5f:9b root@c7test1
The key's randomart image is:
+--[ RSA 2048]----+
|        o.o..o+ .|
|         + o.o o |
|      . = o o    |
|       o = =     |
|        S + .    |
|       . .   o   |
|               o.|
|              . .|
|             ...E|
+-----------------+
```

秘密鍵を使用する際の認証用文字列

■作成された鍵の確認

```
[root@c7test1 ~]# ls .ssh/
id_rsa  id_rsa.pub
```

②サーバ側に公開鍵を転送する

寺子屋ITライセンス
TERAKOYA IT LICENCE

■**scpを使って公開鍵を転送**
- **ssh接続を使用したリモート端末へのファイルのコピー**
- **secure copy　の略**
- **書式　scp [コピー元] [コピー先]**

■[root@c7test1 ~]# **scp .ssh/id_rsa.pub 10.0.0.12:~**
root@10.0.0.12's password:　　← **パスワードの入力**
id_rsa.pub　　100%　394　　0.4KB/s　　00:00

■**転送されて来たかチェック**
[root@vm01 ~]# **ls**
anaconda-ks.cfg　**id_rsa.pub**

LinuC

③サーバ側で公開鍵を登録

■サーバ側では、クライアントから貰った公開鍵をauthorized.keysに登録する。
- 登録先「~/.ssh/authorized_keys」

■.sshディレクトリの作成
[root@vm01 ~]# **mkdir -m 700 .ssh**

■リダイレクトで公開鍵を登録
[root@vm01 ~]# **cat id_rsa.pub >> .ssh/authorized_keys**

■他のユーザから参照されないようパーミッションを変更
[root@vm01 ~]# **chmod 600 .ssh/authorized_keys**

④sshコマンドを使って接続

寺子屋ITライセンス
TERAKOYA IT LICENCE

■**クライアントからsshコマンドを使って接続する。**

[root@c7test1 ~]# **ssh 10.0.0.12**
Enter passphrase for key '/root/.ssh/id_rsa':xxxxxxx
秘密鍵作成時に指定したパスフレーズを入力　↑↑

■**規定値ではパスワード認証での接続もOKになっている。**

[root@c7test1 ~]# **ssh 10.0.0.12**
Enter passphrase for key '/root/.ssh/id_rsa':**（空エンター）**
root@10.0.0.12's password:　**←　パスワードの入力**
Last login: Sun Sep 15 17:18:25 2019 from 192.168.41.17
[root@vm01 ~]#　←　パスワードで接続出来た

■**公開鍵による認証しか受け付けないようにすることも可能。**

LinuC

パスワード認証の無効化

■**設定ファイルの書換え**

[root@vm01 ~]# **vi /etc/ssh/sshd_config**
PasswordAuthentication yes　← **規定では「yes」になっている**
PasswordAuthentication no　← **「no」に書き換える**

■[root@vm01 ~]# **systemctl restart sshd**　← **sshdをリスタート**

■**パスワード認証できるか確認**

[root@c7test1 ~]# **ssh 10.0.0.12**
Enter passphrase for key '/root/.ssh/id_rsa':　**（空エンター）**
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
→ **パスワード認証に移行せず、接続は拒否される**
[root@c7test1 ~]#

まとめ

主題110：セキュリティ

110.1	セキュリティ管理業務の実施	• システムを監査して、SUID/SGIDビットが設定されているファイルを探す • ユーザのパスワードおよびパスワードエージング情報を設定または変更する • nmapおよびnetstatを使用して、システムの開いているポートを見つける • ユーザのログイン、プロセス、メモリー使用量を制限する • **基本的なsudoの設定および利用方法**
110.2	ホストのセキュリティ設定	• シャドウパスワードおよびその機能について知っている • 使用していないネットワークサービスをオフにする • **TCPラッパーの役割について理解している**
110.3	暗号化によるデータの保護	• **基本的なOpenSSH 2クライアントの設定および利用方法** • **OpenSSH 2サーバのホストキーの役割について理解している** • 基本的なGnuPGの設定および利用方法 • SSHポートトンネル（X11トンネルを含む）について理解している

本日ご紹介したのは一部分です。

Linuxの学習方法

セキュリティだけでなくLinuxの知識を習得するには、もっと広く勉強が必要。

■出来るだけLinuxに触る時間を作る

- コマンド（オプション含）に慣れる
- 色々なディストリビューションを試す
- 実際に構築、運用してみる

■色々な学習方法を取り入れる。

- LPI-Japan認定教材
- 無料セミナー
- アカデミック認定校の講座を受ける
- ブログを書くなど

■アカデミック認定校
寺子屋ITライセンス

LinuC

寺子屋ITライセンス
TERAKOYA IT LICENCE

LinuC試験から学ぶセキュリティ入門

ご清聴ありがとうございました

LPI-JAPAN