～クラウドサービス時代を支えるOSS/Linux人材育成～

Skill Brain
スキルブレイン株式会社

Linux Professional Institute Japan LPI-JAPAN

LPIC303技術解説無料セミナー

LPI-Japanアカデミック認定校
スキルブレイン株式会社　インストラクター
三浦　一志

LPIC303試験の概要

LPIC303で求められる人材像

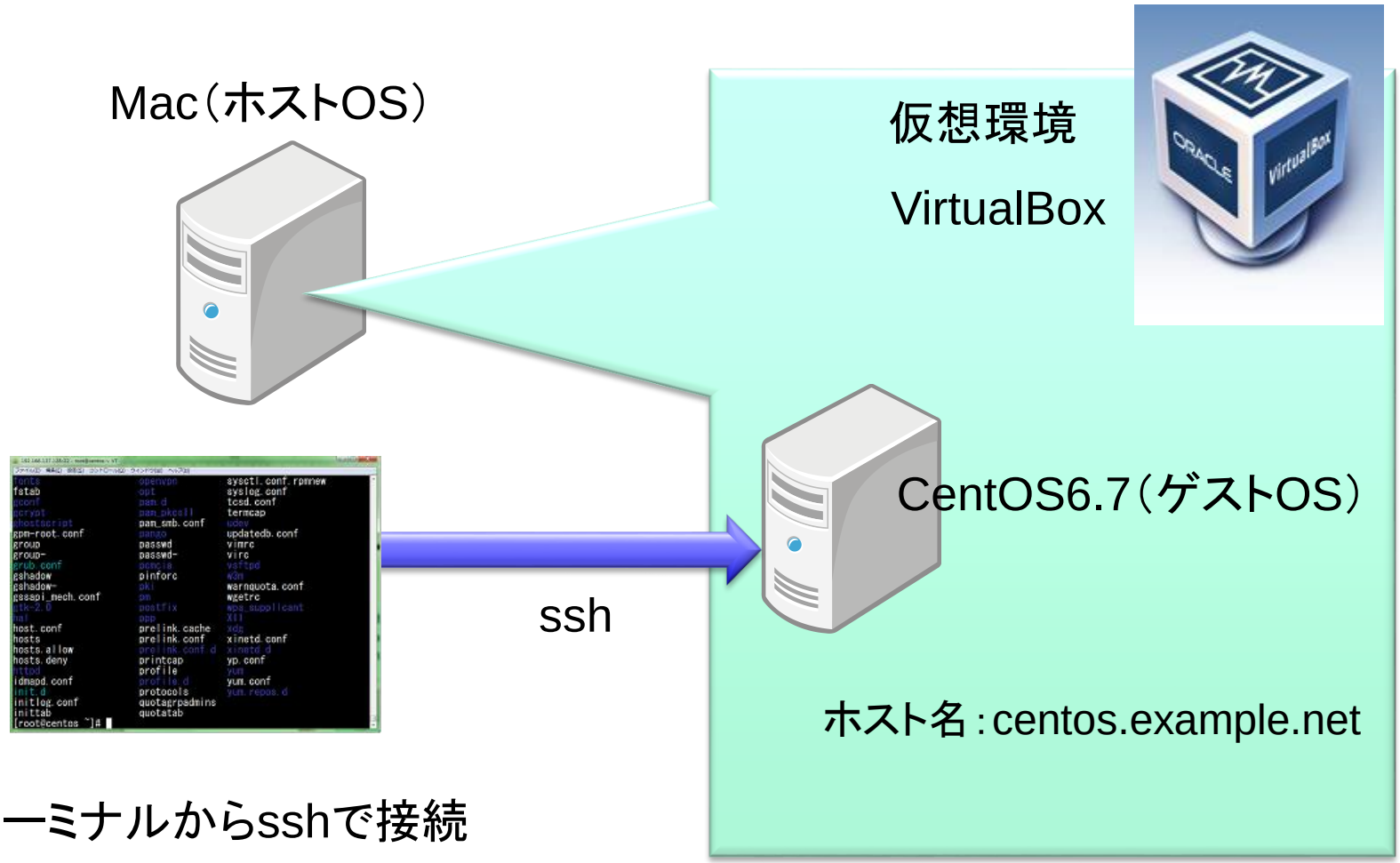
- マルチサイトの企業や負荷が非常に高いインターネットサイトなどのように、複雑な自動化の問題向けにカスタマイズしたソリューションを設計して実装することができること
- プロジェクトを開始し、予算を意識して作業することができること
- アシスタントを監督し、問題のトラブルシューティングを支援することができること
- 上位管理職のコンサルタントとなれること

主題

- 主題320：暗号化
- 主題321：アクセス制御
- 主題322：アプリケーションセキュリティ
- 主題323：操作のセキュリティ
- 主題324：ネットワークセキュリティ

使用する環境

- 各主題のポイントとなる部分を紹介
- 仮想環境を利用し、デモで確認を行う



主題320:暗号化

Skill Brain

- 320.1 OpenSSL
- 320.2 高度な GPG
- 320.3 暗号化ファイルシステム

Linux Professional Institute

SSLの概要

- SSL (Secure Sockets Layer)はセキュリティーを要求される通信を行うためのプロトコル
  - IETFではTLS（Transport Layer Security）でインターネット標準とされている
- 主な機能
  - 通信相手の認証
  - 通信内容の暗号化
  - 改竄の検出など
- OpenSSL
  - SSLプロトコル・TLSプロトコルの、オープンソースで開発・提供されるソフトウェア
  - サポート：SSL 2.0、3.0、TLS 1.0、1.1、1.2、DTLS 1.0、1.2
  - 暗号方式：DES、RC2、RC4、RC5、SEED、IDEA、AESなど
  - ハッシュ関数：MD5、MD2、SHA-1、SHA-2、MDC-2
  - 公開鍵暗号方式：RSA暗号、DSA、Diffie-Hellman鍵共有

SSLによる通信手順

クライアント

サーバ

使用するアルゴリズムの合意

サーバ認証のアルゴリズム、鍵交換のアルゴリズムなど

サーバの認証

・サーバから、サーバ証明書を送る

・クライアントでは、サーバの証明書を確認する

データ伝送で使用する鍵の確立

鍵生成関数を使用して、ランダムな文字列から、

暗号鍵とMAC鍵を生成する。

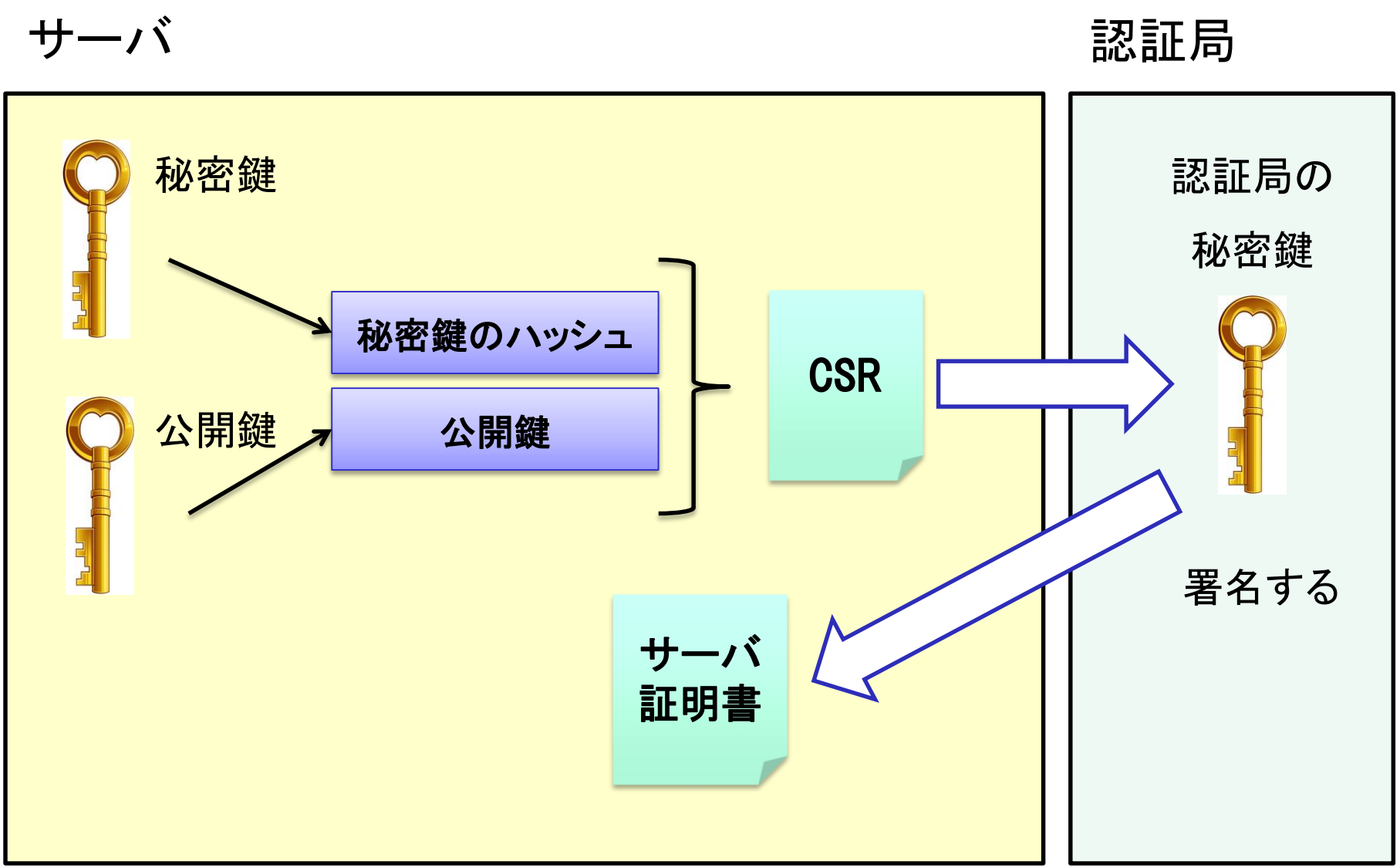
ハンドシェイクが正しくおこなわれたことの確認

データ伝送

ハンドシェーク
フェーズ

データ伝送
フェーズ

サーバ証明書発行の手順



自己署名証明書を作成する

■秘密鍵の生成(RSA形式)

```
# openssl genrsa -des3 -out privkey.key 2048
```

■署名リクエスト CSR(Certificate Signing Request)の作成

```
#openssl req -new -key privkey.key -out server.csr
```

■サーバ証明書の作成

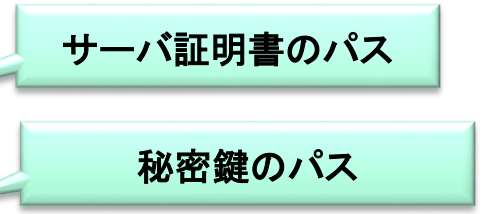
```
#openssl ca -out server.crt -infiles server.csr
```

- Apacheにサーバ証明書を組み込む

  /etc/httpd/conf.d/ssl.conf

  SSLCertificateFile /etc/httpd/conf.d/server.crt

  SSLCertificateKeyFile /etc/httpd/conf.d/privkey.key



- SSLのテスト：サーバのポート443に接続する

  # openssl s_client -connect centos.example.net:443

主題321:アクセス制御

Skill Brain

- 321.1 ホストベースのアクセス制御
- 321.2 拡張属性とACL
- 321.3 SELinux
- 321.4 その他の強制アクセス制御システム

Linux Professional Institute

SELinuxの概要①

- Linuxのカーネルに強制アクセス制御 機能を付加する
  - 強制アクセス制御　MAC（Mandatory Access Control）

- SELinuxを使用しないLinuxのアクセス権は・・・
  - ファイルやディレクトリのパーミッションに基づいて行われる
  - rootはこのパーミッションを無視してアクセスが可能
  - root権限が乗っ取られると、致命的な被害を受ける
  - ファイルによるパーミッションの設定は任意アクセス制御と呼ばれる
    任意アクセス制御（DAC: Discretionary Access Control）

- SELinuxでは以下のようなことが可能
  - HTTP、FTPといったプロセスごとにアクセス制限をかけるType Enforcement（TE）
  - rootも含む全てのユーザに関して制限をかけるロールベースアクセス制御（RBAC）

SELinuxの概要②

TE（Type Enforcement）

- 全てのプロセスに対して「ドメイン」と呼ばれるラベルを付加する。
- リソース（ファイルやディレクトリ）に対しても同じく「タイプ」と呼ばれるラベルを付与する。
- 各リソースには「アクセス・ベクタ」が割り当てられる。
- アクセス・ベクタとは「読み込み」、「書き込み」といったリソースに対して行える操作の種類
- 各ドメインとタイプに対して許可されるアクセス・ベクタを、セキュリティーポリシーとして設定可能

RBAC（Role-based access control）

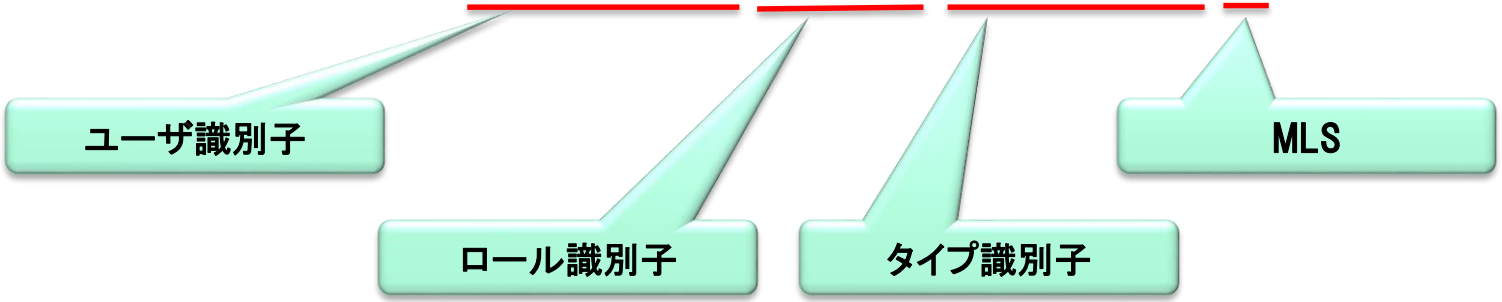
- 「ロール」と呼ばれるいくつかのドメインを束ねたものを設定し、それをユーザに付与する仕組み
- ユーザは付与されたロール内のドメインの権限でのみファイルにアクセス可能
- この機能により各ユーザ毎に細かく権限を付与、制限することが可能である。

コンテキスト(contexts)

- SELinuxを有効にすると、リソースやプロセスにコンテキストが付与される
- コンテキストには、以下の識別子がある
  - ユーザ識別子
  - ロール識別子
  - タイプ識別子
  - MLS(Multi Level Security)

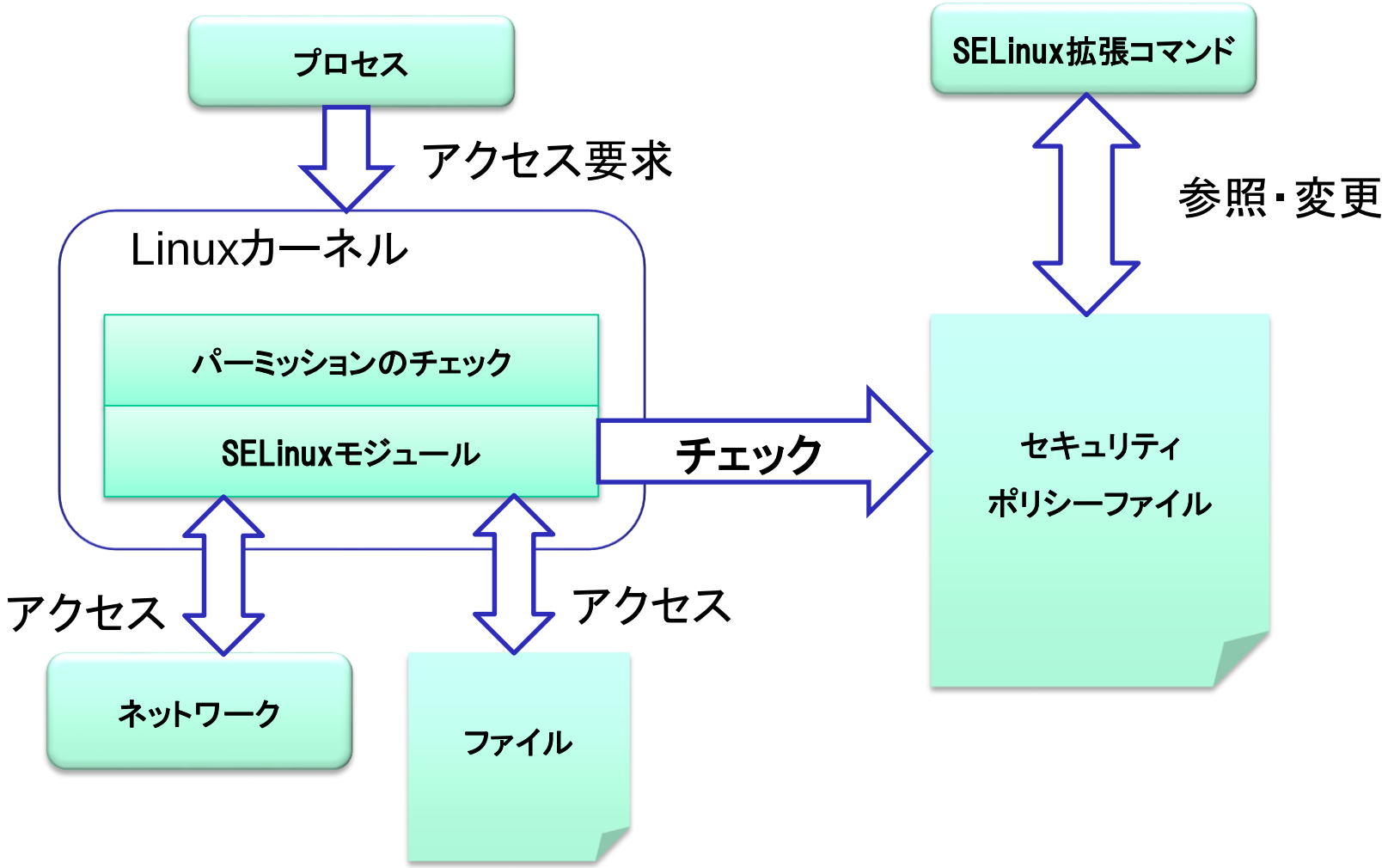
- リソース(ファイル)のコンテキスト

```
-rw-rw-r--. centuser centuser unconfined_u:object_r:user_home_t:s0 testfile
```



SELinux動作の仕組み

Skill Brain



ドメイン遷移

- ドメイン遷移とは
  - 通常は子プロセスは親プロセスと同じドメインで動作する
  - 設定により親プロセスとは違うドメインで子プロセスを実行すること
- httpdプロセスの実行

/etc/init.d/httpd start → ドメイン: initrc_t
/usr/sbin/httpd → タイプ: httpd_exec_t(エントリ・ポイント)
httpdプロセス → ドメイン: httpd_t

- ドメイン遷移の役割
  - プロセスに権限(ドメイン)を割り当てることができる
  - 不要な権限の昇格が避けられる

    SUIDによる一般ユーザからrootへの昇格など

SELinuxを有効にする

SELinuxが有効か確認する

- getenforce
- ステート

「Enforcing」　→　SELinuxが有効になっている（強制モード）

「Permissive」　→　SELinuxは有効になっている（許容モード）

「Disabled」　→　SELinuxは無効になっている

SELinuxを設定する

- コマンドで設定　setenforce
- ”setenforce　0”　→　Permissiveモードで動作する
- ”setenforce　1”　→　Enforcingモードで動作する
- 設定ファイルで設定 /etc/selinux/config

SELINUX=enforcing

SELINUX=permissive

SELINUX=disable

コンテキストの確認

■コンテキストを確認する

- ファイルのコンテキストを確認する

$ ls -lZ

- プロセスのコンテキストを確認する

$ ps axZ

- ユーザのコンテキストを確認する

$ id -Z

ポリシー

- アクセス制御を行うルールはポリシーによって決められている

- CentOSのデフォルトポリシーは「targeted」が設定されている
  - targeted → ネットワークやデーモンについて制限している
  - strict → すべてにおいて制限している

- ポリシーは自作することもできるが、ルールが非常に複雑である
- targetedポリシーをもとに、カスタマイズする方法が容易
- ポリシーの変更は/etc/selinux/configで行う

semanage

Skill Brain

- semangeは以下のことができます
  - SELinuxの有効／無効
  - リソースに対するセキュリティコンテキストの変更
  - ユーザに対するセキュリティコンテキストの割当て
  - ネットワークに対するセキュリティコンテキストの割当て
  - booleanの設定

semanageで制御できる項目はmanのマニュアルなどで調べてください。

Boolean

- SELinuxのポリシーを変更するのは複雑である
- ポリシーは変更せずに、ある特定の機能だけを有効にしたり無効にする機能がある
- 現在のboolean値を確認する

  # semanage boolean -l　もしくは　# getsebool -a

- boolean値を変更する

  # setsebool -P allow_ftpd_full_access on

  （再起動後も設定値を維持する場合は-Pオプションを使用する）

  - 設定値が変更されたか確認する

  # getsebool allow_ftpd_full_access

主題322:アプリケーションセキュリティ

Skill Brain

- 322.1 BIND/DNS
- 322.2 メールサービス
- 322.3 Apache/HTTP/HTTPS
- 322.4 FTP
- 322.5 OpenSSH
- 322.6 NFSv4
- 322.7 syslog

Linux Professional Institute

BINDのセキュリティ

BINDの脅威

- DNSキャッシュポイズニング
  偽のゾーン情報をサーバに記憶させ、別のサイトに誘導されてしまう
- 中間者攻撃
  通信しているホスト間の情報を改ざんされたりする
- Smurf攻撃
  相手のコンピュータに大量の偽装パケットを送る

BINDの対策

- 最小限の権限を与えたユーザでnamedを実行する
- chroot jailを利用して、任意のディレクトリをルートディレクトリと見せるようにする
- named.confを設定して、BINDのバージョンを表示しないようにする
- 特定のIPアドレスから不正なアクセスや攻撃を受けた場合、そのIPアドレスからのアクセスを禁止する
- ファイヤーウォールを導入して、パケットフィルタリングを行う

■323.1 ホスト構成管理

Puppetの概要

- Rubyで開発された、Unix系OSのシステム管理を自動で行うためのツール
- Puppetの特徴
  - GPLに基づいたオープンソース
  - 独自の宣言型言語によりシステムを管理
  - クライアント/サーバ型アーキテクチャ
  - 抽象化レイヤー
  - 依存関係の処理
  - LDAPサポート
  - コミュニティの活動や、開発が活発

Puppetのインストール

- パッケージからインストールする
- リポジトリの登録

  # rpm -ivh http://yum.puppetlabs.com/puppetlabs-release-el-6.noarch.rpm

- パッケージのインストール

  # yum install puppet puppet-server

  （クライアントはpuppetのみインストール）

- インストールの仕方は公式ページを参照

  http://docs.puppetlabs.com/guides/puppetlabs_package_repositories.html

Puppetの設定

■サーバとクライアントのホストで時刻を合わせておく

```
# ntpdate ntp.nict.jp
```

■サーバの起動

```
# service puppetmaster start
```

■クライアントでpuppet agentをテストする

```
# puppet agent --test --server centos.example.net
```

（初回の起動のときは、SSLの証明書がないと言われる）

■ サーバで証明書のリクエストを確認し署名する

```
#puppet cert list
#puppet cert sign debian7.example.net
```

■クライアントでもう一度実行する

```
# puppet agent --test --server centos.example.net
```

マニフェスト

- 設定を記述したファイル
- 拡張子は「.pp」
- /etc/puppet/manifestsディレクトリに保存する

- マニフェストの書式

```
リソース　{　'リソース識別名':
　　パラメータ => パラメータの値,・・・
}
```

  - リソース識別名の後にはコロン「:」を付ける
  - コメントは「#」の後に記述する
  - クラスを宣言するときは「class」の後にクラス名を付ける
  - 宣言したクラスを使用する場合は「include クラス名」を記述する

マニフェストの例①

- /etc/hostsのパーミッションを644に変更する

```
file { '/etc/hosts':
    owner => 'root',
    group => 'root',
    mode  => 644,
}
```

/etc/puppet/manifests/site.ppとして保存する

マニフェストの例②


```
class createuser {
  user { 'student1':
      ensure => present,
      comment => 'Linux test user',
      home => '/home/student1',
      managehome  => true,
      shell => '/bin/bash'
  }
}

node 'debian7.example.net' {
   include 'createuser'
}
```

ユーザのホームディレクトリも同時に作成

マニフェストの例③


```
class createfile {
  file { '/home/student1/puppettest':
      ensure => present,
      owner => 'student1',
      group => 'student1',
      mode  => 0644,
      content => "Hello, student1!"
  }
}

class createfilechild inherits createfile {
  File['/home/student1/puppettest'] { mode => 0755 }
}
```

主題324：ネットワークセキュリティ

Skill Brain

- 324.1 侵入検出
- 324.2 ネットワークセキュリティスキャン機能
- 324.3 ネットワークの監視
- 324.4 netfilter/iptables
- 324.5 OpenVPN

Snortの概要

- 侵入検知ソフトウェア
- Snortの特徴
  - IPネットワーク上でのリアルタイムの解析
  - GPLライセンス
  - パケットスニファ／パケットロガーとしても使用できる
  - 豊富なプリプロセッサが用意されている
    - portscan：ポートスキャンの検出を行う
    - frag2：IPフラグメントの再構築を行う
    - stream4：TCPストリームの再構築とステートフルな解析を行う
    - telnet_decode：Telnetの制御文字を正規化する
  - さまざまな形式でアラートを出力することができる
  - 公式サイト：https://www.snort.org/

Snortの設定

■ソースからインストールを行う

- ソースからのインストールは複雑なため、Linuxセキュリティ標準教科書を参照してください

■ソースを展開したディレクトリから設定ファイルをコピーする

```
#  cp snort-2.9.7.3/rpm/snort.sysconfig /etc/sysconfig/snort
```

■ /etc/sysconfig/snortを編集する

```
INTERFACE=eth1
USER=snort
GROUP=snort
LOGDIR=/var/log/snort
```

Listenするネットワークインターフェースを指定する

Snortの設定

- 起動用スクリプトをコピーする

```
# cp /home/centuser/work/snort-2.9.7.3/rpm/snortd /etc/init.d/
# chmod 755 /etc/init.d/snortd
```

- コミュニティ版ルールの配置

```
# mkdir -p /etc/snort/rules
# chown -R snort.snort /etc/snort
# wget https://www.snort.org/rules/community.tar.gz
# tar -xvfz community.tar.gz -C /etc/snort/rules
```

- ライブラリが利用するディレクトリの作成

```
# mkdir /usr/local/lib/snort_dynamicrules
```

- ログ領域の作成

```
# mkdir /var/log/snort ; chown -R snort.snort /var/log/snort
```

Snortの基本設定

- 設定ファイル　/etc/snort/snort.conf
- ネットワークの設定を行う

  ipvar HOME_NET 192.168.56.0/24

  ipvar EXTERNAL_NET any
- 以下の変数のパスをカレントディレクトリからのパスとする(../を./に変更する)

  var RULE_PATH ./rules

  var SO_RULE_PATH ./so_rules

  var PREPROC_RULE_PATH ./preproc_rules

  var WHITE_LIST_PATH ./rules

  var BLACK_LIST_PATH ./rules

Snortのルール設定

- 読み込むルールの設定を記述

  include $RULE_PATH/local.rules → 追加する

  include $RULE_PATH/community.rules → 追加する

  #これ以下ルールはすべてコメントにする

  #include $RULE_PATH/app-detect.rules

  #include $RULE_PATH/attack-responses.rules

  ・・・

- 独自ルールを作成する

  # vi /etc/snort/rules/local.rules

  alert icmp any any -> any any (msg: "ICMP Packet detected"; sid:999999;)

Snortの実行

■プロミスキャスモードの設定

vi /etc/sysconfig/network-scripts/ifcfg-eth1

PROMISC=yes

ファイルの最後に記述する

- （VirtulaBoxを使用している場合は、ネットワークの設定でプロミスキャスモードを許可にする）

■Snortを起動する

/etc/init.d/snortd start

■Snortが起動しない場合は・・・

- /var/log/messagesにエラーメッセージが出力されていないか確認

■動作テスト

- 他のホストからpingを実行してみる
- /var/log/snort/alertにログが出力されていればOK

- シグネチャのルールは、ルールヘッダとルールボディから成る
- 書式

＜ルールアクション＞ ＜プロトコル＞
＜IPアドレス＞ ＜ポート番号＞ ＜方向演算子＞ } ルールヘッダ
＜IPアドレス＞ ＜ポート番号＞
＜（オプション・・・）＞ } ルールボディ

ルールアクション	説明
activate	ルールに該当するパケットが存在する場合、警告を出す（dynamicアクションを呼び出す）
alert	ルールに該当するパケットを記録し、警告を出す
dynamic	activateアクションから呼び出され、該当するパケットを記録する
log	ルールに該当するパケットを記録する
pass	ルールに該当するパケットを無視する

シグネチャの例

例1

```
alert tcp any any -> any 80 (msg: "http request GET" ; content:"GET"; http_method; sid:1000000)
```

例2

```
alert tcp any any -> any 80 (msg: "http request URI" ; content:"/index.html"; http_uri; sid:1000001)
```

msg: → ログに出力するメッセージ

content: → パケットのペイロード部にマッチする文字列を指定する

httpd_method → HTTPリクエストのメソッドでマッチするもの

http_uri → HTTPリクエストのURIでマッチするもの

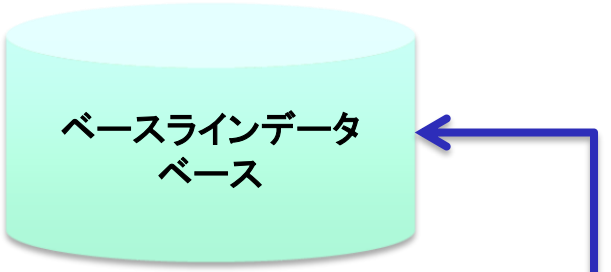
sid: → シグネチャのIDを指定する。独自ルールは1，000，000以上

Tripwireの概要

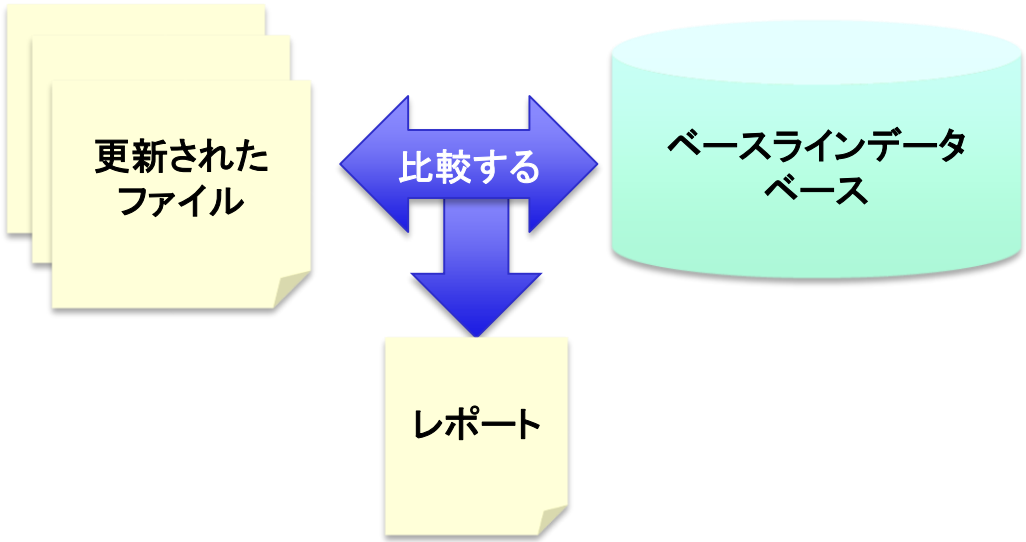
- 改善検知ソフトウェア
- Tripwireの特徴
  - GPLライセンス
  - 公式サイト：https://www.tripwire.co.jp/
  - オープンソースの入手先：http://sourceforge.net/projects/tripwire/
- Tripwireにおいてできること
  - ファイルの内容が変更されたことを検知
  - ファイルやディレクトリが追加／削除されたことを検知
  - ファイルやディレクトリの所有者／パーミッションが変更されたことを検知
- Tripwireではできないこと
  - リアルタイムでの検知
  - 変更者や変更箇所を特定すること
  - 変更前の状態に戻すこと

Tripwireの仕組み

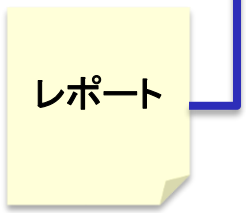
①ベースラインデータベースを作成する



②整合性のチェックをするとレポートが作成される



③レポートを基にベースラインデータベースを更新する



tripwireのインストール

- ソースファイルからインストール

```
$ wget http://jaist.dl.sourceforge.net/project/tripwire/tripwire-src/tripwire-2.4.2.2/tripwire-2.4.2.2-src.tar.bz2
$ tar jxvf tripwire-2.4.2.2-src.tar.bz2
$ cd tripwire-2.4.2.2-src
$ ./configure
$ make
# make install
```

（インストールの途中で使用許諾とサイトパスフレーズを聞かれる）

Tripwireのシステム設定

- システム設定ファイル：/usr/local/etc/twcfg.txt
  - ログレベルの変更
  - SMTPサーバの指定など

- 更新の手順
  - /usr/local/etc/twcfg.txtを編集する
  - 以下のコマンドを実行して、システム設定を反映させる

```
# twadmin -m -F -S /usr/local/etc/site.key /usr/local/etc/twcfg.txt
```

ポリシーファイル

- 検査対象となるファイル・ディレクトと検査内容を定義したファイル
- /usr/local/etc/twpol.txtを基にして作成
- 書式

```
(
  属性1，属性2，・・・
)
{
  検査対象オブジェクト名 -> プロパティマスク;
  検査対象オブジェクト名 -> プロパティマスク;
}
```

ポリシーの属性一覧

属性の一覧

属性	説明
rulename	ルールに名前をつける
emailto	整合性チェックの結果をメールに送信する。--email-report付で整合性のチェックをした場合使用される
severity	ルールに重要度(0～1,000,000)をつける。デフォルトは0。
recurse	ディレクトリを再帰的に検査するか指定する。デフォルト値はtrue。

プロパティの一覧とプロパティマスクはman twpolicyで調べてみてください

ポリシーファイルの例

Skill Brain

```
(
  rulename = "HTML FILE",emailto="root@centos.example.net",
)
{
 /var/www/html  -> +m; #ファイルの修正時刻をチェック
 /etc/httpd  -> $(ReadOnly); #ファイルが変更されるとチェックされる
}
```

ポリシーの設定　データベース初期化

■ポリシーファイルの修正

- /usr/local/etc/twpol.txtを修正する

■ポリシーファイルの作成

```
# twadmin -m P -S /usr/local/etc/site.key /usr/local/etc/twpol.txt
```

■データベースの初期化

```
# tripwire --init
```

整合性のチェック　データベースの更新

■整合性のチェック

export LANG=C

tripwire --check

■整合性チェックの結果

- 「/usr/local/lib/tripwire/report/ホスト名-日付-時刻.twr」で出力
- 整合性チェックの時にロケールをLANG=Cとしておく

■整合性の確認

twprint --print-report --report-level [0-4] --twrfile レポートファイルのパス

0:簡易表示　～　4:詳細表示

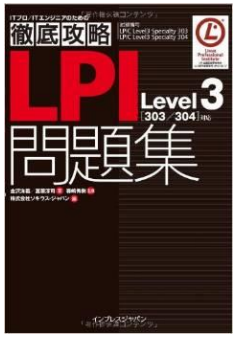
■データベースの更新

- ベースラインデータベースの更新が行われる

tripwire --update --twrfile レポートファイルのパス

参考資料

Skill Brain



徹底攻略LPI問題集Level3[303／304]対応 2012/2/23発行
金沢 泳義 (著), 菖蒲 淳司 (著), 森嶋 秀樹 (監修), ソキウス・ジャパン (編集)
出版社:翔泳社
272ページ
価格3,456円
ISBN-10: 4844331582 ISBN-13: 978-4844331582

Linuxセキュリティ標準教科書（Ver1.0.0）
詳しくは下記URLで
http://www.lpi.or.jp/linuxtext/security.shtml
発行:エルピーアイジャパン

SELinux徹底ガイド―セキュアOSによるシステム構築と運用 基本的な仕組みから高度な運用管理方法までを徹底解説
中村 雄一 (著), 水上 友宏 (著), 上野 修一 (著), & 3 その他
出版社: 日経BP社
318 ページ
価格4913円
ISBN-10: 4822221113
ISBN-13: 978-4822221119

食べる！SSL！ ―HTTPS環境構築から始めるSSL入門 [Kindle版]
小島 拓也 (著), 中嶋 亜美 (著), 吉原 恵美子 (著), 中塚 淳 (著)

119 ページ
価格320円

質疑応答についてはお気軽にお声掛けください。

ご清聴ありがとうございました。

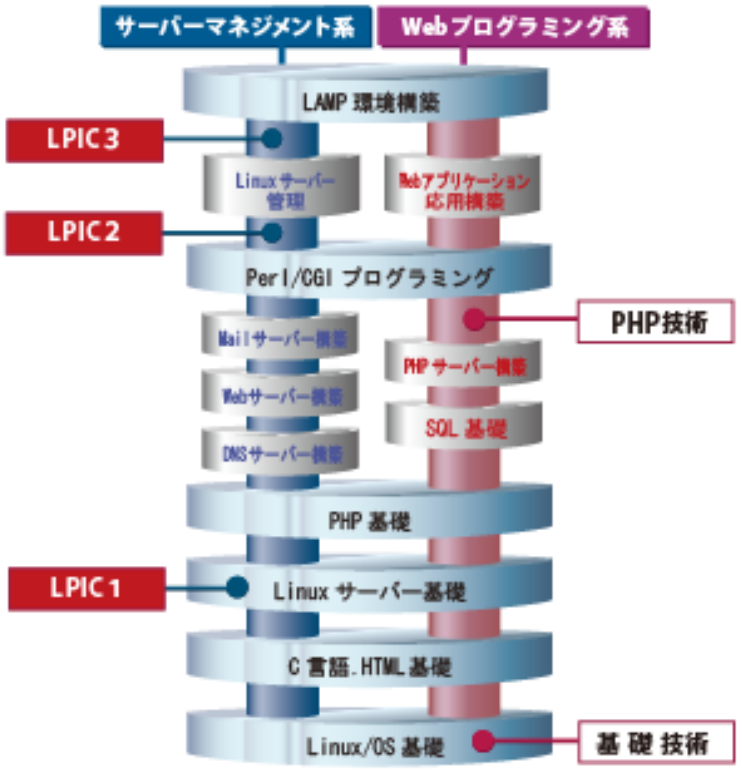
Skill Brain スキルブレイン株式会社

http://www.skillbrain.co.jp

info@skillbrain.co.jp

■OSS/Linux エンジニア研修

■Linux 基礎
■Linux サーバー構築実践
■Linux サーバー管理・運用実践
■Linux サーバーセキュリティ構築実践
■LPIC（レベル 1・2・3）試験対策
■OSS-DB（Silver/Gold）試験対策
■Oracle 認定 Java（OCJ）試験対策
■ITIL® ファンデーション（シラバス 2011）研修
■仮想化技術研修
■セキュリティ（FW・IDS・ウイルス対策）
■階層別アセスメント研修　ビジネスマナー研修
※その他、企業様ごとにセミオーダー研修を承ります。



経験・スキルともに豊富な講師陣が技術や資格取得をサポート

Skill Brain

三浦　一志

サーバ管理者として8年以上の実務経験を積み、講師としても10年以上のキャリアを持つ。
法人向けにLPIC研修・Linuxサーバ構築・セキュリティ研修やITIL研修を主として担当。
ITIL認定講師　情報セキュリティスペシャリスト

【担当講習】
・Linux/UNIX　・LPIC試験対策　・セキュリティ　・Java　・PHP　・OSS-DB　・HTML5

河原木　忠司

Linux・Windowsを使ったインフラ環境の構築・運用、セキュアなインターネットサーバーの構築など、企業・官公庁向けの技術研修を担当。
MCT（マイクロソフト認定トレーナー）　VoIP認定講師

【担当講習】
・Linux　・Windows　・VoIP　・セキュリティ　・仮想化　・LPIC試験対策　・OSS-DB

大崎　茂

OSS研修専任講師として、大手電機メーカー・通信キャリア・大手プロバイダー等、IT企業のLPIC対策研修ならびにOSSを中心とした技術研修などを専門に担当。

【担当講習】
・Linux　・C言語　・PHP　・Java　・Ajax　・LAMP関連　・LPIC試験対

木村　祐

ITILV3 Expert　ITILV2 Manager　ITILV2 OSA･RCV･SOA･PPO　EXIN認定インストラクター
ISO20000 Consultant/Manager

【担当講習】
・ITILファウンデーション　・ITILエキスパート　・ITILプラクティショナー