徹底攻略

試験番号 LPI 303 Security Exam

Linux Professional Institute
LPI出題範囲準拠教材
ロゴ使用承認番号 LPIJ161029

LPIC

Level 3 303［Version 2.0］対応

教科書＋問題集

常泉茂雄／菖蒲淳司［著］
株式会社ソキウス・ジャパン［編］

1冊で合格力が身に付く
受験者必携の書

［詳しい解説］＋［演習問題］＋［模擬試験問題2回分］

Version 2.0 対応版

「最重要ポイント」がひと目でわかる。 試験対策

シリーズ100万部突破!!

インプレス

この書籍はLPI認定教材ではありません。
LPIロゴの使用に関しては、出版元がLPIのロゴを使用する許可を得ている教材であることを示すものであり、LPIが教材の内容につき何ら保証をするものではありません。
また該当の教材で学習することにより合格を保証するものではありません。

本書は、LPI技術者認定試験（LPIC）レベル3の受験対策用の教材です。著者、株式会社インプレスは、本書の使用によるLPI技術者認定試験（LPIC）レベル3への合格を一切保証しません。

本書の内容については正確な記述につとめましたが、著者、株式会社インプレスは本書の内容に基づくいかなる試験の結果にも一切責任を負いません。

Linuxは、Linus Torvaldsの米国およびその他の国おける登録商標または商標です。
その他、本文中の製品名およびサービス名は、一般に開発メーカーおよびサービス提供元の商標または登録商標です。なお、本文中には™、®、©は明記していません。

インプレスの書籍ホームページ

書籍の新刊や正誤表など最新情報を随時更新しております。

http://book.impress.co.jp/

まえがき

ご存知のとおり、現在、Linuxはさまざまな用途に利用されています。わかりやすいところでは、Webサーバやデータベースサーバ、ファイル共有サーバなどのサーバ用途としてはもちろんのこと、デスクトップ環境としても利用されています。また、現在では多くの人が利用しているクラウドサービスも、表に見えてはいませんがサービス提供を支える重要なインフラ環境として多くのLinuxOSが稼働しています。

しかし、これらのシステムやサービスを安全に活用するには、セキュリティ面を十分に考慮し、それを支えている製品や技術をよく把握したうえで脆弱性を可能な限り減らし、問題の発生時に被害を最小限に留めることが重要です。

実際の企業システムでは、セキュリティ対策としてベンダーが提供する有償製品を利用したソリューションが大勢を占めているという状況ですが、Linuxのオープンソースだけでもここまでできるということを、本書を通して実感していただき、303試験に無事合格していただければ著者としてこれ以上の幸せはありません。本書は次のようなコンセプトで執筆しております。

「試験範囲を効率よく学習できる構成」
303試験は試験範囲が非常に多岐に渡り、その一つ一つを詳細に解説すると何冊も本が書けるほど広く深い内容となっています。そのため、LPIレベル1、レベル2および基礎知識は十分に身に付いているという前提で、試験範囲を効率よく学習できるよう構成しています。

「CentOSをベースに動作確認」
もっともメジャーなディストリビューションであるCentOS7をベースに執筆しているため、本書に記載した例は容易に動作確認ができます。一部技術については、特定のディスクトリビューションでしか動作しないようなものがあるため、CentOSでは動作しないもののみUbuntuで動作確認しています。

ページ数の都合ですべてを詳細に解説できないことが心苦しいのですが、試験合格を目指す受験者の方々が効率よく勉強でき、かつ確実に実力が付くことを第一に配慮しています。資格の取得を志す皆様が、本書を十分に活用して合格という栄冠を勝ち取られること、そしてその合格を手に今後皆様が活躍されることを心より願っております。

最後に、編集を担当してくださった坂田様、執筆を支えてくれた家族、その他本書の執筆に関わるすべての方々に心よりお礼を申し上げます。

2016年11月吉日　菖蒲 淳司／常泉 茂雄

LPI技術者認定試験について

LPIC（エルピック）はLPI（Linux Professional Institute：Linux技術者認定機関）が実施している世界共通のLinux技術者認定制度です。特定のLinuxディストリビューションに依存しない汎用性の高い認定資格制度であることが知られています。2016年10月時点では日本国内での累計受験者数は28万人を超え、10万1,000人以上が認定を取得している人気の資格です。

LPICのレベルには、「LPIC-1（LPICレベル1）」「LPIC-2（LPICレベル2）」「LPIC-3（LPICレベル3）」の3つがあります。2016年11月現在のLPIC資格体系は以下のとおりです。

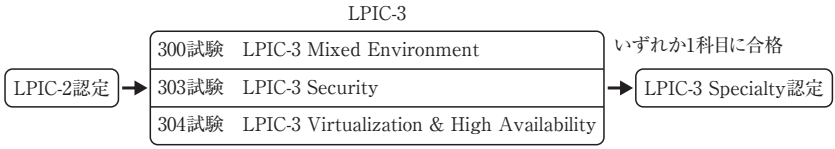
【LPIC資格体系】

資格	試験（バージョン）	認定されるスキル
LPIC-1	101試験（4.0）	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
	102試験（4.0）	
LPIC-2	201試験（4.0）	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
	202試験（4.0）	
LPIC-3	300試験（1.0）	Linux、Windows、Unixが混在するシステムの設計、構築、運用・保守ができるエキスパートエンジニアであること
	303試験（2.0）	セキュリティレベルの高いコンピュータシステムの設計、構築、運用・保守ができるエキスパートエンジニアであること
	304試験（2.0）	クラウドコンピューティングシステム（クラウド）の設計、構築、運用・保守ができるエキスパートエンジニアであること

LPIC-3 と 303 試験について

現在、LPICレベル3に対応する試験は、「LPIC-3 Specialty Mixed Environment（300試験）」「LPIC-3 Specialty Security（303試験）」「LPIC-3 Specialty Virtualization & High Availability（304試験）」の3つが実施されています。3試験のうちいずれかに合格し、かつ、有意なLPICレベル2の認定を持っていると、「LPICレベル3 Specialty」に認定されます。

【LPIC-3資格体系】



本書では、LPIC-3の試験科目「LPI 303 Security Exam（303試験）」を扱います。同試験に合格することによって、セキュリティレベルの高いコンピュータシステムの設計、構築、運用・保守ができるエキスパートエンジニアであることを証明できます。

●303試験（LPI 303 Security Exam）

・試験時間：90分
・問題数：約60問
・受験料：30,000円（税抜き）

●出題範囲について

303試験では、LinuxやOSS環境でのITシステムのセキュリティに関する知識とスキルが問われる問題が出題されます。出題範囲は、「暗号化」「ホストセキュリティ」「アクセス制御」「ネットワークセキュリティ」の4つのカテゴリに分かれています。

出題範囲の各項目には重要度が記されています。この重要度はそれぞれの目標の相対的な重要性を重み付けしたものであり、重みの範囲は1〜10で設定されています。重要度が高い項目ほど、試験において多くの問題が出題される傾向にあります。

主題325：暗号化
　325.1　X.509証明書と公開鍵の基礎（重要度：5）
　325.2　暗号化、署名および認証のX.509証明書（重要度：4）
　325.3　暗号化ファイルシステム（重要度：3）
　325.4　DNSと暗号化（重要度：5）
主題326：ホストセキュリティ
　326.1　ホストの堅牢化（重要度：3）
　326.2　ホストの侵入検知（重要度：4）
　326.3　ユーザの管理と認証（重要度：5）
　326.4　FreeIPAのインストレーションとSambaの統合（重要度：4）
主題327：アクセス制御
　327.1　任意アクセス制御（重要度：3）
　327.2　強制アクセス制御（重要度：4）
　327.3　ネットワークファイルシステム（重要度：3）
主題328：ネットワークセキュリティ
　328.1　ネットワークの堅牢化（重要度：4）
　328.2　ネットワークの侵入検知（重要度：4）
　328.3　パケットフィルタ（重要度：5）
　328.4　仮想プライベートネットワーク（VPN）（重要度：4）

【303試験の出題範囲詳細】
http://www.lpi.or.jp/lpic3/range/range303.shtml

●試験方法

試験会場に用意されたコンピュータを使って行われます。試験問題のほとんどはマウスによる選択方式ですが、キーボードから入力して解答する問題も一部出題されます。

●合否結果と資格認定

試験終了後、合否と得点が印刷されたスコアレポートが渡されます。

●認定要件

LPIC-3の試験科目のうちいずれかに合格し、かつ、有意なLPICレベル2の認定を持っていること。

受験申し込み方法

●LPI-IDの取得

LPI認定試験の受験を申し込むには、事務局が指定するWebページからLPI-IDを取得する必要があります。

・登録用Webページ……https://bloom.lpi.or.jp/caf/Xamman/register

●受験申し込み先

LPI認定試験は、ピアソンVUEのWebサイトまたはコールセンターを利用して申し込みができます。いずれの場合も希望する受験会場および受験日時を選択することができます。ただし、予約状況によっては希望どおりの会場や日時を選択できない場合もありますので、必ず予約状況を確認してください。

●ピアソンVUE

URL：http://www.pearsonvue.co.jp/
TEL：0120-355-173 または 0120-355-583

初めてピアソンVUEに申し込む場合は、受験者情報を登録する必要があります。その後、以下のいずれかの方法で申し込むことができます。

- **Webからの申し込み**
  http://www.pearsonvue.co.jp/からログインして予約
- **電話での申し込み**
  上記電話番号に直接申し込み（受付時間：平日午前9時～午後6時）
- **テストセンターに直接申し込み**
  テストセンター一覧から受験希望のテストセンターを確認したうえで直接連絡

LPI-Japanの連絡先

〒106-0041　東京都港区麻布台1-11-9 CR神谷町ビル 7F
TEL：03-3568-4482
FAX：03-3568-4483
E-Mail：info@lpi.or.jp
URL：http://www.lpi.or.jp/

最新の試験情報は、必ず上記のWebサイトで確認してください。

LPI再受験ポリシーについて

2010年11月18日より、LPI認定試験に関する再受験ポリシー（リテークポリシー）が下記のとおり実施されています。最新の再受験ポリシーについては、必ずLPI-JapanのWebサイトで確認してください。なお、再受験ポリシーに違反している場合、ピアソンVUEでは受験予約をすることができません。

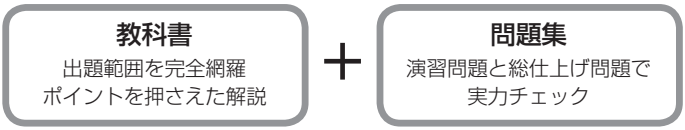
- LPIC同一科目を再受験する際、2回目の受験については、受験日の翌日から起算して７日目以降（土日含む）より可能
- 3回目以降の受験については、最後の受験日の翌日から起算して30日目以降より可能
- 一度合格した科目については、受験日から2年間は再受験できない
- 受験後2年間を経た場合、再受験ポリシーはいったんクリアになり、その後の同一試験の最初の再受験は一度目の受験としてカウントされる

本書の特長と活用方法

本書は「LPI 303 Security Exam（303試験）」の合格を目指す方を対象とした受験対策教材です。第1章から第4章は、出題範囲に沿った解説・演習問題で構成されています。第5章と第6章は、模擬問題の位置付けとなる「総仕上げ問題」です。各章で学習したのちに、実戦形式の総仕上げ問題で受験対策の仕上げをしましょう。

【教科書と問題集を兼ね備えた1冊】

解説は出題範囲を完全網羅し、出題傾向に沿ってわかりやすくポイントを押さえています。また、各セクションの演習問題と模擬問題2回分で合計264問の問題を掲載していますので、この1冊で万全の受験対策が可能です。

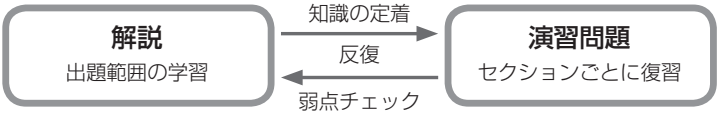


【試験対策がわかりやすい】

本書は、実際の試験を徹底分析した結果を反映した内容になっています。本文中の「試験対策」欄には試験の重要項目が記載されていますので、効率的に学習を進めることができます。

【演習問題で知識を定着】

第1章から第4章までは、出題範囲のカテゴリ別の章立てで、出題傾向に沿って解説しています。各セクションには復習を目的とした演習問題を設け、「解説を読み終えてから記憶が薄れないうちに演習問題を解く」を繰り返して知識を確実に定着させます。演習問題の解答には解説ページへの参照が記載されていますので、不正解の問題は本文に戻って復習することができます。



【模擬問題2セット分を掲載】

第5章と第6章の「総仕上げ問題」には、本試験と同レベルの問題を各60問掲載しています。本試験と同じ90分で解いてみましょう。正解した問題はチェックボックスにチェックしていき、全問正解できれば合格目前です。不正解の問題は、本文に戻って見直しをしましょう。

●解説

例

出力結果を伴う例では、実行するコマンドを太字にしています。

重要語句、重要事項

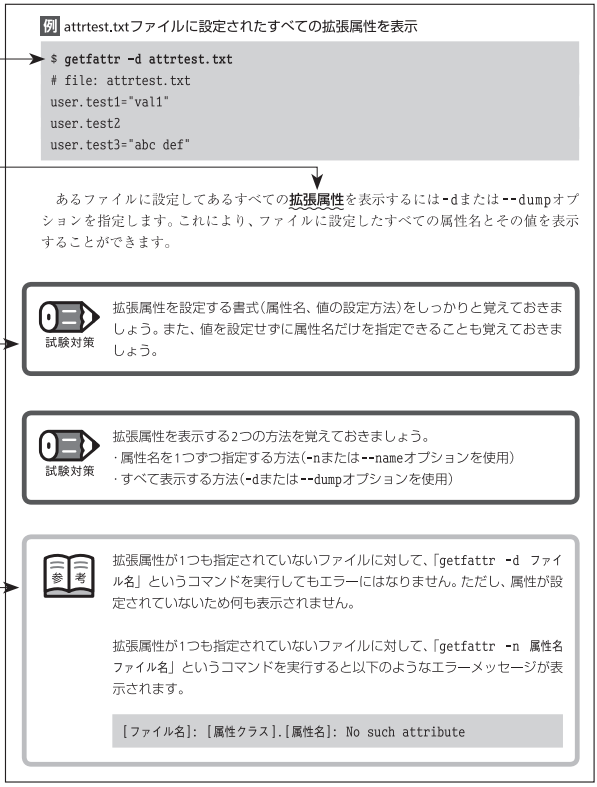
本文中の重要用語や重要語句は太字で示しています。

試験対策

試験のために理解しておかなければならないことや、覚えておかなければいけない重要事項を示しています。

参考

試験対策とは直接関係はありませんが、知っておくと有益な情報を示しています。



例 attrtest.txtファイルに設定されたすべての拡張属性を表示

```
$ getfattr -d attrtest.txt
# file: attrtest.txt
user.test1="val1"
user.test2
user.test3="abc def"
```

あるファイルに設定してあるすべての**拡張属性**を表示するには-dまたは--dumpオプションを指定します。これにより、ファイルに設定したすべての属性名とその値を表示することができます。

試験対策

拡張属性を設定する書式(属性名、値の設定方法)をしっかりと覚えておきましょう。また、値を設定せずに属性名だけを指定できることも覚えておきましょう。

試験対策

拡張属性を表示する2つの方法を覚えておきましょう。
・属性名を1つずつ指定する方法(-nまたは--nameオプションを使用)
・すべて表示する方法(-dまたは--dumpオプションを使用)

参考

拡張属性が1つも指定されていないファイルに対して、「getfattr -d ファイル名」というコマンドを実行してもエラーにはなりません。ただし、属性が設定されていないため何も表示されません。

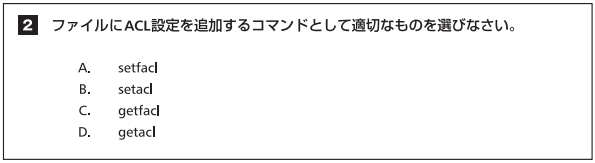
拡張属性が1つも指定されていないファイルに対して、「getfattr -n 属性名 ファイル名」というコマンドを実行すると以下のようなエラーメッセージが表示されます。

```
[ファイル名]: [属性クラス].[属性名]: No such attribute
```

●演習問題・解答

問題

問題は選択式もしくは記述式です。



2 ファイルにACL設定を追加するコマンドとして適切なものを選びなさい。

A. setfacl
B. setacl
C. getfacl
D. getacl

解答

正解の選択肢は太字で示しています。
関連する内容の記載されたページを示しています。

2 A

ファイルに対してACLを設定するコマンドはsetfaclです(➡P.157)。したがって、選択肢Aが正解です。

●総仕上げ問題

チェックボックス

確実に理解している問題のチェックボックスを塗りつぶしながら問題を解き進めると、2回目からは不確かな問題だけを効率よく解くことができます。すべてのチェックボックスが塗りつぶされれば合格は目前です。

選択式の問題

正解を1つだけ選ぶものと複数選ぶものがあります。

☐ **43.** **setcifsaclコマンドの説明として適切なものを3つ選びなさい。**

A. セキュリティ記述子を表示する
B. セキュリティ記述子のACLから、アクセス制御エントリを削除する
C. セキュリティ記述子のACLにアクセス制御エントリを修正する
D. セキュリティ記述子のACLにアクセス制御エントリを追加する

➡ P308

解答ページ

問題の右下に、解答ページが表示されています。ランダムに解くときも、解答ページを探すのに手間取ることがありません。

記述式の問題

問題文を読んで、コマンド名、オプション、ファイル名などを記述するものがあります。

☐ **39.** **シェルを新しいロールで実行するSELinuxのコマンドを記述しなさい。**

_____

➡ P307

問題ページ

解答の右端に、問題ページが表示されています。確認したい問題を簡単に探すことができます。

解答

正解の選択肢は太字で示されています。

43. **B、C、D** ➡ P294

setcifsaclコマンドには、以下のような機能があります。

- セキュリティ記述子のACLから、アクセス制御エントリを削除する（-Dオプション）
- セキュリティ記述子のACLにアクセス制御エントリを修正する（-Mオプション）
- セキュリティ記述子のACLにアクセス制御エントリを追加する（-aオプション）

したがって、選択肢**B**、**C**、**D**が正解です。セキュリティ記述子を表示するのはgetcifsaclコマンドの機能です（選択肢A）。

参照➔ P208

本文のページ

関連する本文のページが表示されています。正解できなかった問題は、本文に戻って復習しましょう。

目次

第1章

暗号化

1-1 X.509証明書と公開鍵の基礎

この節ではX.509証明書と公開鍵の基本的な仕組みと用語について学習します。また、OpenSSLを使用した認証局の作成、SSL証明書の発行方法についても学習します。

X.509証明書と公開鍵の基礎

X.509証明書とは、公開鍵基盤の標準規格であるX.509によって定義された公開鍵証明書であり、Webサービスやメールなどの通信を安全に行ううえで広く使用されています。X.509証明書を理解するために必要な用語について説明していきます。

●公開鍵基盤とは

公開鍵基盤（Public-Key Infrastructure：PKI）とは、公開鍵暗号方式やデジタル署名などの技術を利用して、メッセージの盗聴、改ざん、なりすましといった攻撃を防ぎ、安全な通信を実現する仕組みです。ネットワーク上でクレジットカード番号や氏名、住所などの個人情報のやり取りを行う際に、悪意ある第三者からの盗聴やメッセージの改ざんなどを防ぎ、安全な通信を提供します。

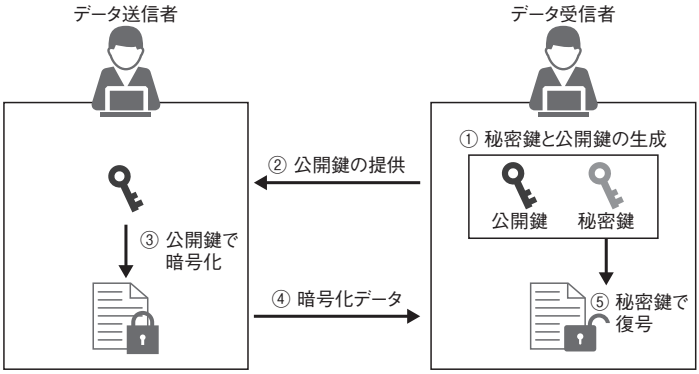
●公開鍵暗号方式

公開鍵暗号方式は、暗号化と復号のために異なる鍵（公開鍵、秘密鍵）を使う暗号方式です。公開鍵暗号方式の代表的な方式として、RSA、DSAなどがあります。公開鍵暗号方式を用いて二者の間でメッセージ（データ）の送受信を行う場合、メッセージの送受信は受信者から提供された公開鍵を用いてメッセージの暗号化を行い、受信者に送信します。受信者は秘密鍵を用いてメッセージの復号を行います。

公開鍵と秘密鍵はペアとなっており、公開鍵で暗号化されたメッセージはペアとなる秘密鍵でのみ復号することができます。秘密鍵からペアとなる公開鍵を作成することができますが、公開鍵からは秘密鍵を作成することはできません。この仕組みにより送信者が送ったメッセージは、公開鍵を作成した受信者のみが確認することができるため、公開鍵が不特定のユーザーに公開されていても第三者にメッセージを見られずに通信を行うことができます。そのため、受信者は自身が作成した秘密鍵を外部に漏えいすることがないように、厳重に管理する必要があります。

送信者・受信者の間で、公開鍵と秘密鍵を利用してメッセージを暗号化して通信を行う場合は次の図のようになります。

【公開鍵暗号方式によるメッセージの暗号化通信】



① 受信者が公開鍵およびペアとなる秘密鍵を生成
② 送信者に公開鍵を提供
③ 送信者は公開鍵を利用して暗号化を実施
④ 暗号化したメッセージを受信者に送信
⑤ 受信者は自身が作成した秘密鍵を用いて受け取ったメッセージを復号

●共通鍵暗号方式

共通鍵暗号方式は、暗号化と復号のために同じ鍵（共通鍵）を使う暗号方式です。代表的な方式として、Triple-DES、AESなどがあります。

同じ鍵を利用するため、公開鍵暗号方式と比較すると処理が高速だという特徴がありますが、ネットワーク上の通信で共通鍵が漏えいした場合、通信している暗号文が解読されてしまいます。受信者と送信者の間で行う共通鍵の受け渡しを第三者に見られないようにする必要がありますが、その方法として公開鍵暗号方式を用いて共通鍵（もしくは共通鍵の元となる情報）を暗号化する方法があります。

●公開鍵証明書

公開鍵証明書は、送信者と受信者が互いに信頼する認証局にて発行される証明書であり、公開鍵を作成した受信者の身元の審査を認証局が行い、公開鍵に署名を行うことで公開鍵の正当性を保証するものです。

公開鍵を用いてメッセージを送信する場合、その公開鍵が本当にメッセージの送信先の相手が作成したものであるか、また相手の身元が正しいかどうかを明確にする必要があります。これは、悪意のある第三者の侵入により公開鍵が入れ替わっている可能性や、相手の身元に問題がある可能性があるためです。そのため、証明書を発行する認証局は、公開鍵証明書を利用して通信を行う二者がそれぞれ信頼できる機関である必要があります。

●X.509

X.509は、ITU-T（国際電気通信連合 電気通信標準化部門）が定めたPKIの標準規格です。ITU（International Telecommunication Union）やISO（International Organization for Standardization）で標準化された規格であり、主に以下に関するフォーマットを定めています。

・公開鍵証明書
・認証局（Certification Authority：CA）
・証明書失効リスト（CRL）

X.509公開鍵方式をベースにしているセキュリティプロトコルの一例としては、TLS/SSL、S/MIME、IPSec、スマートカードなどがあります。

●認証局

認証局（Certification Authority：CA）は、公開鍵暗号方式を利用した通信において、申請者が作成した公開鍵の正当性を証明するための機関です。認証局には、主に以下のような役割があります。

1. 認証局運用規定（Certification Practice Statement：CPS）に基づいて申請元を審査し、公開鍵証明書を発行する
   発行した公開鍵証明書は「リポリトジ」と呼ばれるデータベースで管理されます。審査は登録局（RA）、証明書の発行は発行局（IA）というように、審査や発行に関して機能が分かれています。

2. 公開鍵証明書の失効、およびCRLを発行する
   申請者の要求や証明書に信頼性がなくなった場合に証明書を失効します。証明書が失効されたことを証明書の利用者が確認できるように、CRLを公開します。

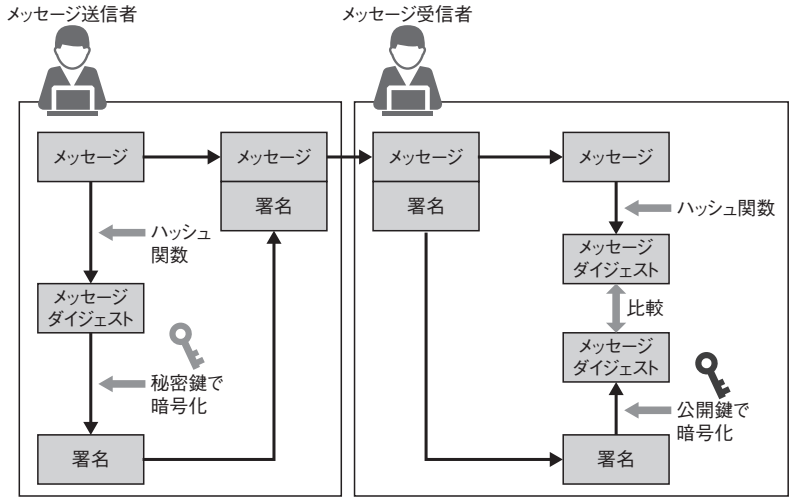
認証局には、パブリック認証局とプライベート認証局の2種類の形態があります。**パブリック認証局**は、不特定多数の範囲で利用される認証局であり、パブリック認証局が発行したルート証明書は、一般的に利用されるWebブラウザやメールソフトにあらかじめ組み込まれています。パブリック認証局が発行する公開鍵証明書は不特定の人々が利用するため、社会的に認知され信用された存在である必要があります。これに対して、**プライベート認証局**は社内システムなど特定の範囲で利用される認証局であり、プライベート認証局が発行したルート証明書は利用者に配布する必要があります。

●デジタル署名

デジタル署名は、相手にメッセージ（データ）を送信する場合に、メッセージが送信者本人のものであることや、改ざんされていないことを証明するために利用されます。メッセージ（もしくはメッセージの一部）から求めた「メッセージダイジェスト」と呼ばれるハッシュ値に対して秘密鍵を使って暗号化したもので、公開鍵証明書の発行には、デジタル署名の仕組みが利用されます。

デジタル署名が付与されたメッセージを受け取った受信者は、秘密鍵のペアとなる公開鍵で暗号化されたハッシュ値の復号を行い、メッセージから求めたハッシュ値と比較して一致することを確認することで、改ざんされていないかを検証します。デジタル署名のアルゴリズムとして、公開鍵暗号方式にRSAやDSA、ハッシュ関数にSHA-1やSHA-256の組み合わせが利用されます。

【デジタル署名の仕組み】



●X.509証明書のファイル形式

X.509証明書、CRL、秘密鍵などは「ASN.1（Abstract Syntax Notation One）」と呼ばれるデータの表記法で記されており、主に次の2つの方式で符号化（エンコーディング）されたものが利用されています。

- **DER（Distinguished Encoding Rules）**

  DERという方式でバイナリに符号化された形式です。

  `openssl x509`コマンドで`-outform`オプションを指定することによりDER形式で出力することができます。

・**PEM（Privacy Enhanced Mail）**
DER形式のファイルをBase64符号化方式を利用し、テキストに符号化した形式です。opensslコマンドでファイル形式を指定しなかった場合に、デフォルトで利用される形式です。

●PKCSとは

PKCS（Public-Key Cryptography Standards）とは、RSAセキュリティ社によって策定された公開鍵暗号の標準規格であり、PKCS#1～PKCS#15の形式があります。OpenSSLのサブコマンドで扱える形式には次のものがあります。

【PKCSの形式】

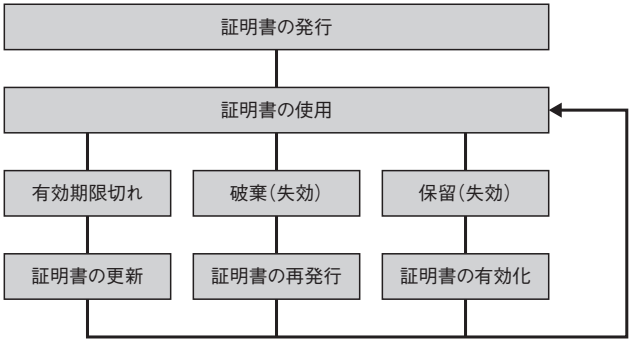
形式	説明
PKCS#7	X.509の公開鍵証明書を符号化する形式
PKCS#10	X.509の証明書署名要求（CSR）を符号化する形式
PKCS#12	X.509の秘密鍵と公開鍵証明書を管理するための形式

X.509証明書のライフサイクル

公開鍵証明書には、発行、使用、更新、破棄といった**ライフサイクル**が存在します。公開鍵基盤では証明書のライフサイクルを管理する仕組みがあり、認証局がその手続きを行います。

証明書は有効期限付きで作成されるため、有効期限を超えると効力を失います。そのため、継続して使用する場合は更新を行う必要があります。また、秘密鍵の漏えいや申請者側が認証局の規定に違反するなどの理由により信頼性が保たれなくなった場合は、有効期限内であっても証明書を失効しなければなりません。証明書の失効は認証局が行います。

【証明書のライフサイクル】



【証明書のライフサイクル】

ライフサイクル	説明
証明書の発行	申請者（証明の対象者）が証明書発行要求を認証局に申請し、認証局が審査および証明書の発行を行う
証明書の利用	有効期限内かつ無効化されていない状態で、Webサービスなどで利用される
証明書の有効期限切れ	証明書に記載された有効期限が超過した状態。継続して利用するためには更新手続きが必要となる
証明書の破棄（失効）	秘密鍵の漏えいや証明書の記載内容の変更などの理由により失効手続きを行う。失効した証明書は証明書失効リスト（CRL）に登録される
証明書の保留（失効）	秘密鍵の漏えいの疑いなどにより、一時的に証明書を失効させる場合。保留した証明書はCRLに登録される

X.509証明書のフィールドおよびX.509v3証明書の拡張

X.509証明書は、X.509バージョン1で定義されたバージョン、シリアル番号、署名アルゴリズムや発行先などが記載された領域と、X.509バージョン3で拡張された領域が存在します。これらの領域は「署名前証明書」と呼ばれ、署名アルゴリズムによってデジタル署名が行われることによって証明書となります。X.509証明書のフィールドおよびX.509v3証明書の拡張は、RFC5280で定義されています。

【証明書のフィールド】

証明書のフィールド		説明
署名前証明書（tbsCertificate）	バージョン（version）	X.509証明書のバージョン
	シリアル番号（serialNumber）	証明書のシリアル番号
	署名（signature）*	証明書の署名アルゴリズム
	発行者（issuer）	証明書を発行した認証局の名称
	有効期間（validity）	証明書の有効期限開始（Not Before）と終了（Not After）
	サブジェクト（subject）	公開鍵の所有者
	サブジェクト公開鍵情報（subjectPublicKeyInfo）	公開鍵署名アルゴリズムおよび公開鍵
	証明書拡張（extensions）	X.509バージョン3拡張領域（次表参照）
署名アルゴリズム（signatureAlgorithm）*		証明書の署名アルゴリズム
署名値（signatureValue）		署名前証明書を署名アルゴリズムで署名した値

* 同一のアルゴリズム識別子を含む

署名前証明書（tbsCertificate）を署名アルゴリズム（signatureAlgorithm）にて署名したものが署名値（signatureValue）となります。署名アルゴリズムがsha256WithRSA Encryptionの場合、署名前証明書をSHA256によってハッシュ化した結果をRSAで暗号化したものが署名値となります。

●X.509v3証明書の拡張領域

X.509で定められた拡張領域には、主に以下のフィールドがあります。

【証明書の主な拡張フィールド】

証明書の拡張フィールド	説明
サブジェクト鍵識別子（SubjectKeyIdentifier）	所有者の公開鍵を識別するための識別子
認証局鍵識別子（authorityKeyIdentifier）	認証局の秘密鍵識別子を識別するための識別子
基本制約（basicConstraints）	証明書の所有者が認証局であるかどうか、下位に続く証明書パスの最大の深さを識別する
鍵の用途（keyUsage）	鍵の用途
所有者の別名（subjectAltName）	公開鍵の所有者の別名
拡張した鍵の用途（extKeyUsage）	鍵の用途を限定
CRL配布ポイント（crlDistributionPoints）	CRLの配布ポイント
証明書ポリシー（certificatePolicies）	証明書の目的や発行する際の規定などの情報
機関情報アクセス（authorityInfoAccess）	認証局に関する追加情報へのアクセス

拡張フィールドにはcriticalという情報があり、TRUE（重要）もしくはFALSE（重要でない）の値を持ちます。TRUEの場合は、その拡張フィールドの内容を解釈し処理する必要があります。処理できない場合は棄却されます。FALSEの場合に解釈ができないと、拒否することができます。

●OpenSSLの設定

opensslコマンドで証明書を作成するための設定が**/etc/pki/tls/openssl.cnf**ファイルに記載されています。

以下は、openssl.cnfファイルにおける証明書の拡張フィールドの設定例です。認証局の証明書を作成するためには、openssl.cnfファイルのbasicConstraintsに「CA:true」を指定します。

例 basicConstraintsの指定

```
basicConstraints = critical,CA:true,pathlen:0
                   ①                ②
```

① 「critical」を指定すると、設定値の内容を解釈できない場合は証明書が棄却されます。

② 「pathlen:(正数)」は下位に作成可能な認証局の階層の数を表し、認証局の証明書を作成する場合に指定します。「pathlen:0」を指定した場合、サーバ証明書の署名は可能ですが、下位に中間証明書を作成することができなくなります。

複数のホストに対してサーバ証明書を作成するには、subjectAltNameに以下のようにホスト名（FQDN）を指定します。

例 subjectAltNameの指定

```
subjectAltName=DNS:testserver1.example.com,DNS:testserver2.example.com
```

例 証明書のフィールドの表示

```
# openssl x509 -in cacert.pem -text -noout
```

試験対策

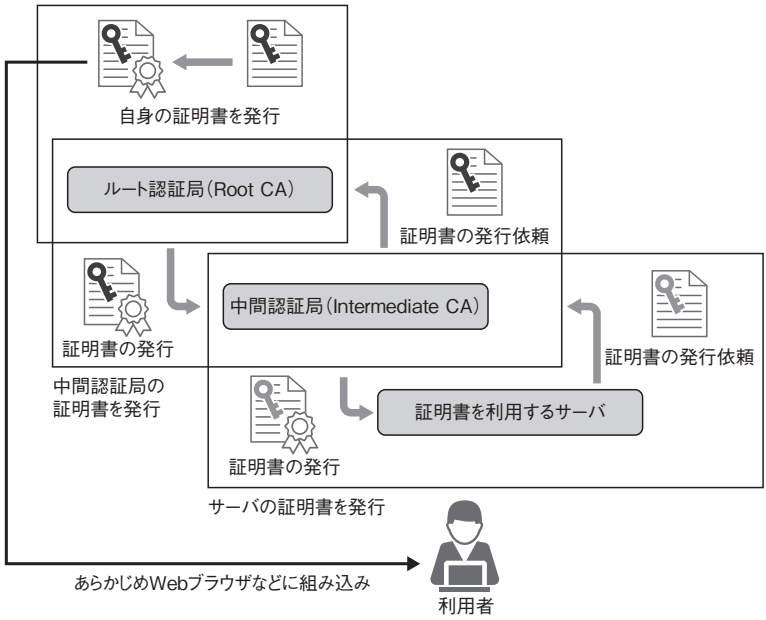
openssl.cnfファイルの拡張フィールドの設定について覚えておきましょう。

トラストチェーン（証明書チェーン）とは

証明書を発行する認証局は、最上位にあるルート認証局（Root CA）と公開鍵証明書の所有者との間に1つ以上の中間認証局（Intermediate CA）が存在する階層構造となっています。**トラストチェーン**とは、対象となる証明書からルート証明書までの一連の証明書が連鎖している状態を指しています。

次ページの図は、SSLが使われているWebサービスを利用する場合の例を示しています。Webブラウザには、ルート証明書があらかじめ組み込まれています。また、Webサーバは自身のSSLサーバ証明書とその証明書を発行した中間認証局の中間CA証明書を格納しています。ブラウザはWebサーバから中間認証局、さらにルート認証局までの証明書をたどり、順に検証していきます。

【トラストチェーンの仕組み】



OpenSSLによる公開鍵と秘密鍵の生成と管理

OpenSSLを利用して、秘密鍵と公開鍵の作成や、それらを利用した暗号化と復号を行います。

●OpenSSLの利用

OpenSSLは、オープンソースで開発されたSSL（Secure Sockets Layer）の実装で、セキュアな通信が必要なときに使用されます。OpenSSLを実行するコマンドは**openssl**です。OpenSSLの主な機能には次のようなものがあります。

・秘密鍵、公開鍵の作成およびパラメータの管理
・公開鍵の暗号操作
・X.509証明書と証明書署名要求（CSR）および証明書失効リスト（CRL）の作成
・メッセージダイジェストの計算
・RSA、DSAなどの暗号方式による暗号化と復号
・SSL/TLSクライアントとサーバのテスト
・S/MINE署名と暗号化メールの取り扱い
・タイムスタンプの要求と生成および検証

opensslコマンドの書式は次のとおりです。

書式　[　]内は省略可能

```
openssl［サブコマンド］［オプション］
```

【主なサブコマンド】

サブコマンド	説明
ca	認証局の証明書を管理
crl	CRLを管理
dgst	メッセージダイジェストを計算
genrsa	RSA秘密鍵を生成
md5	MD5ダイジェスト
pkcs12	PKCS#12形式のファイルの作成と管理
req	X.509証明書署名要求(CSR)を管理
rsa	RSAの鍵を管理
rsautl	RSAアルゴリズムを使用したメッセージの署名、検証、暗号化および復号
s_client	SSL/TLSプロトコルを使用して指定サーバに接続
s_server	SSL/TLSプロトコルを使用してメッセージを受信するサーバとして動作
version	OpenSSLのバージョン情報を表示
x509	X.509証明書データを管理

●公開鍵と秘密鍵の生成

opensslでは、秘密鍵と公開鍵の作成や、それらを利用した暗号化と復号を行います。ここでは、opensslのサブコマンドであるgenrsa、rsa、rsautlの使い方を学びます。

openssl genrsaサブコマンドは、**RSA秘密鍵**を生成するコマンドです。

書式　[　]内は省略可能。<　>内は必須

```
openssl genrsa［オプション］
```

【主なオプション】

オプション		説明
-out <ファイル名>		生成したRSA形式の秘密鍵を出力するファイル名を指定。指定がない場合は標準出力に表示
暗号化方法		暗号化方法を指定。-des、-des3、-ideaのいずれかを選択
	-des	暗号化にDESを使用
	-des3	暗号化にTripleDESを使用
	-idea	暗号化にIDEAを使用
ビット数		生成するRSA形式の秘密鍵のビット数。指定がない場合は512ビット

例 鍵長2,048ビットのTripleDESで暗号化したRSA形式の秘密鍵を作成する

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

例 暗号化せずに鍵長2,048ビットのRSA形式の秘密鍵を作成する

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

openssl rsaサブコマンドは、RSAの鍵を管理するコマンドです。

書式　[　]内は省略可能。<　>内は必須

```
openssl rsa [オプション]
```

【主なオプション】

オプション		説明
-in <ファイル名>		生成したRSA形式の秘密鍵のファイル名を指定。指定がない場合は標準入力から読み込む
-noout		エンコードされた鍵を表示しない
-out <ファイル名>		RSA形式の秘密鍵を出力するファイル名を指定。指定がない場合は標準出力に表示
-pubin <ファイル名>		生成したRSA形式の公開鍵のファイル名を指定。指定がない場合は標準入力から読み込む
-pubout		秘密鍵とペアとなる公開鍵を標準出力
-text		テキスト形式で表示
暗号化方法		暗号化方法を指定。-des、-des3、-ideaのいずれかを選択する
	-des	暗号化にDESを使用
	-des3	暗号化にTripleDESを使用
	-idea	暗号化にIDEAを使用

例 暗号化された秘密鍵ファイルを、暗号化を解除したファイルに変換する

```
# openssl rsa -in privkey.key -out privkey-nopass.key
```

例 秘密鍵から公開鍵を生成する

```
# openssl rsa -in privkey.key -pubout > pubkey.key
```

例 RSA形式の秘密鍵の内容を確認する

```
# openssl rsa -in privkey.key -text
```

例 RSA形式の公開鍵の内容を確認する

```
# openssl rsa -pubin -in pubkey.key -text
```

openssl rsautlサブコマンドは、RSAアルゴリズムを使用したメッセージの署名、検証、暗号化および復号を行うコマンドです。

書式　[　]内は省略可能。<　>内は必須

```
openssl rsautl [オプション]
```

【主なオプション】

オプション	説明
-decrypt	ファイルの復号を行う
-encrypt	ファイルの暗号化を行う
-in <ファイル名>	暗号化もしくは復号対象のファイル名を指定
-inkey <ファイル名>	秘密鍵とペアとなる公開鍵を標準出力
-out <ファイル名>	暗号化もしくは復号したファイルの内容を出力するファイル名を指定。指定がない場合は標準出力に表示
-pubin	公開鍵ファイルを入力する場合に指定

例 公開鍵を用いてテキストの暗号化を行う

```
# openssl rsautl -encrypt -pubin -inkey pubkey.key -in sample.txt
  -out sample.txt.enc
```

例 秘密鍵を用いてテキストの復号を行う

```
# openssl rsautl -decrypt -inkey privkey.key -in sample.txt.enc
  -out sample.txt
```

●証明書署名要求（CSR）の生成と証明書の発行

opensslのサブコマンドであるreq、x509、caの使い方を学びます。これらのサブコマンドでは、主に証明書署名要求（CSR）の生成と証明書の発行や管理を行います。

openssl reqサブコマンドは、CSRを生成するコマンドです。

書式　[　]内は省略可能。<　>内は必須

```
openssl req [オプション]
```

【主なオプション】

オプション	説明
-days <日数>	-x509オプションを指定した場合の有効期限を指定
-in <ファイル名>	入力するCSRのファイル名を指定。指定がない場合、標準入力で入力
-key <ファイル名>	入力する秘密鍵のファイル名
-keyout <ファイル名>	出力する秘密鍵のファイル名
-new	新規CSRのファイルを作成
-noout	エンコードされたCSRを表示しない
-out <ファイル名>	出力するCSRのファイル名を指定。指定がない場合、標準出力に表示
-text	テキスト形式で表示
-subj	DN（識別名）を指定
-x509	自己署名証明書を作成。証明書作成のテストもしくはルート証明書の作成に利用される

例 1年間（365日間）有効なX.509形式のCSRを新規作成する

```
# openssl req -new -days 365 -key privkey.pem -out newreq.pem
```

openssl x509サブコマンドは、X.509証明書のメッセージを管理するコマンドです。

書式　[　]内は省略可能。<　>内は必須

```
openssl x509 [オプション]
```

【主なオプション】

オプション		説明
-in 〈ファイル名〉		入力する証明書のファイル名。指定がない場合は標準入力で入力
-inform 〈形式〉		入力ファイルの書式を指定。DER、PEMのいずれかを選択
	DER	DER形式
	PEM	PEM形式
-noout		エンコードされた証明書を表示しない
-out 〈ファイル名〉		出力する証明書のファイル名。指定がない場合は標準出力に表示
-outform 〈形式〉		出力ファイルの書式を指定。DER、PEMのいずれかを選択
	DER	DER形式
	PEM	PEM形式
-text		テキスト形式で表示

例 PEM形式の証明書をDER形式に変換する

```
# openssl x509 -inform PEM -outform DER -in newcert.pem -out newcert.der
```

openssl caサブコマンドは、認証局の証明書を管理するコマンドです。

書式　[　]内は省略可能。<　>内は必須

```
openssl ca［オプション］
```

【主なオプション】

オプション	説明
-batch	対話を行わずにバッチモードで署名
-config 〈ファイル名〉	設定ファイル(openssl.cnfなど)を指定
-create_serial	serialファイルを作成
-crldays 〈日数〉	CRLの有効期限
-days 〈日数〉	証明書の有効期限
-extensions 〈セクション名〉	設定ファイル(openssl.cnf)の拡張用セクションを指定
-gencrl	CRLを作成
-in 〈ファイル名〉	CSRを指定
-infiles 〈ファイル名〉	CSRを指定。最後に記載した場合は複数のCSRを指定可能
-keyfile 〈ファイル名〉	CSRに署名するための秘密鍵を指定
-out 〈ファイル名〉	出力する証明書のファイル名。指定がない場合は標準出力に表示
-policy 〈セクション名〉	ポリシーの指定
-revoke	証明書を失効
-selfsign	-keyfileで指定した秘密鍵で署名

例 自己認証による証明書の発行

```
# openssl ca -out newcert.pem -keyfile privkey.pem -infiles newcsr.pem
```

例 証明書の失効

```
# openssl ca -revoke newcert.pem
```

openssl pkcs12サブコマンドは、PKCS#12形式のファイルの作成と管理を行うコマンドです。

書式　[　]内は省略可能。<　>内は必須

```
openssl pkcs12 [オプション]
```

【主なオプション】

オプション	説明
-export	PKCS#12形式のファイルを作成
-in 〈ファイル名〉	入力する証明書のファイル名。指定がない場合は標準入力で入力
-inkey 〈ファイル名〉	入力する秘密鍵のファイル名
-name 〈名称〉	証明書と秘密鍵を表した名称
-out 〈ファイル名〉	出力するPKCS#12形式のファイル名。指定がない場合は標準出力に表示

例 X.509証明書と秘密鍵を格納したPKCS#12形式のファイルを作成

```
# openssl pkcs12 -export -in newcert.pem -inkey newkey.pem
  -out demo-clinet.pfx -name "demo-clinet.example.com"
```

セキュアな認証局の作成と運用

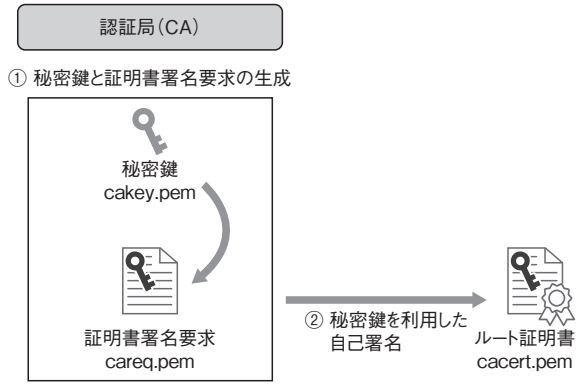
認証局(CA)の役割には電子証明書の発行と失効、および証明書失効リスト(CRL)の発行があります。これらを行うための秘密鍵と電子証明書の信頼性を証明するためのルート証明書を認証局自身が発行します。

ここでは、OpenSSLを利用して認証局が利用する秘密鍵とルート証明書の作成について説明します。通常、サーバで利用する電子証明書を発行する場合は、サーバ側で証明書署名要求(CSR)を作成し、認証局に証明書を発行してもらうための申請を行いますが、最上位の認証局(ルートCA)を作成する場合は、認証局自身で発行したCSRに署名(自己署名)して証明書を作成します。

●ルート証明書の作成

OpenSSLを利用して秘密鍵とCSRの生成を行い、秘密鍵を利用して自身のCSRに対して署名を行います。

【ルート証明書の作成】



試験対策

ルート証明書は認証局の公開鍵を含むCSRに対して、認証局の秘密鍵を用いて自己署名することにより作成されます。

【例：自己署名証明書（ルート証明書）の作成】

以下の(1)～(3)は、ルート証明書を作成する一連の例です。まず、設定ファイル**/etc/pki/tls/openssl.cnf**を編集します。

例 (1)[v3_ca]セクション

```
basicConstraints = critical,CA:true  ←CA:trueであることを確認
keyUsage = cRLSign, KeyCertSign
```

上記のほかに、[CA_default]セクション、[req]セクションによって、各要素の要否などのポリシーを定めています。

例 (2)秘密鍵と自己署名用CSRの作成

```
# cd /etc/pki/CA ←鍵が格納されたディレクトリに移動
# openssl req -new -keyout private/cakey.pem -out careq.pem
Generating a 2048 bit RSA private key
..........+++
.....................+++
writing new private key to 'private/cakey.pem'
Enter PEM pass phrase: ←認証局の秘密鍵のパスフレーズを入力
Verifying - Enter PEM pass phrase: ←認証局の秘密鍵のパスフレーズを入力（繰り返し）
<途中省略>
Country Name (2 letter code) [XX]:JP ←①DN属性値を入力
State or Province Name (full name) []:Tokyo ←②
Locality Name (eg, city) [Default City]:Bunkyo ←③
Organization Name (eg, company) [Default Company Ltd]:Lpic Company Ltd ←④
Organizational Unit Name (eg, section) []:Lpic section ←⑤
Common Name (eg, your name or your server's hostname) []:lpic303 ←⑥
Email Address []:lpic303@example.jp ←⑦

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: ←⑧空欄のままEnter
An optional company name []: ←⑨空欄のままEnter
```

DN※1の要素(属性型)	説明
① Country Name(C)	国名(2文字)
② State or Province Name(ST)	都道府県名
③ Locality Name(L)	市区町村名
④ Organization Name(O)	組織名(会社名など)
⑤ Organizational Unit Name(OU)	組織単位名(部署名など)
⑥ Common Name(CN)	CA名、サイト名(ドメイン名)、サーバ名
⑦ Email Address	メールアドレス
⑧ A challenge password	証明書破棄時のパスワード(空欄可)
⑨ An optional company name	オプションの組織名(空欄可)

※1　**【DN(識別名)】**Distinguished Name。国名や組織名など、証明書の所有者を識別するための情報

例 (3)作成した秘密鍵とCSRから証明書を作成

```
# openssl ca -create_serial -out cacert.pem -days 3650 -batch
  -keyfile private/cakey.pem -selfsign -extensions v3_ca -infiles careq.pem
Using configuration from /etc/pki/tls/openssl.cnf
Enter pass phrase for ./private/cakey.pem:  ←認証局の秘密鍵のパスフレーズを入力
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 17781258249405467012 (0xf6c3b82b2f4c0984)
        Validity
            Not Before: Jun 17 11:20:53 2016 GMT  ←有効期限が指定した期間となっている
            Not After : Jun 15 11:20:53 2026 GMT
        Subject:
            countryName               = JP
            stateOrProvinceName       = Tokyo
            organizationName          = Lpic Company Ltd
            organizationalUnitName    = 303 section
            commonName                = lpic303
            emailAddress              = lpic303@example.jp
                                                  ←CSR作成時のDN情報が表示される
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                79:0F:21:D2:83:0A:27:A4:66:51:90:A3:03:2C:4E:3B:FE:7E:F3:66
            X509v3 Authority Key Identifier:
                keyid:79:0F:21:D2:83:0A:27:A4:66:51:90:A3:03:2C:4E:3B:FE:7E:
                      F3:66

            X509v3 Basic Constraints: critical
                CA:TRUE  ←認証局の証明書の場合はTRUEとなる
            X509v3 Key Usage:
                Certificate Sign, CRL Sign
            Netscape Cert Type:
                SSL CA, S/MIME CA
Certificate is to be certified until Jun 15 11:20:53 2026 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
```

参考

OpenSSLパッケージには「CA」というスクリプトファイルが用意されています。opensslコマンドを直接実行する代わりに、目的に応じたオプションを付与して実行することで秘密鍵やCSRの生成、署名などを行うことができます。

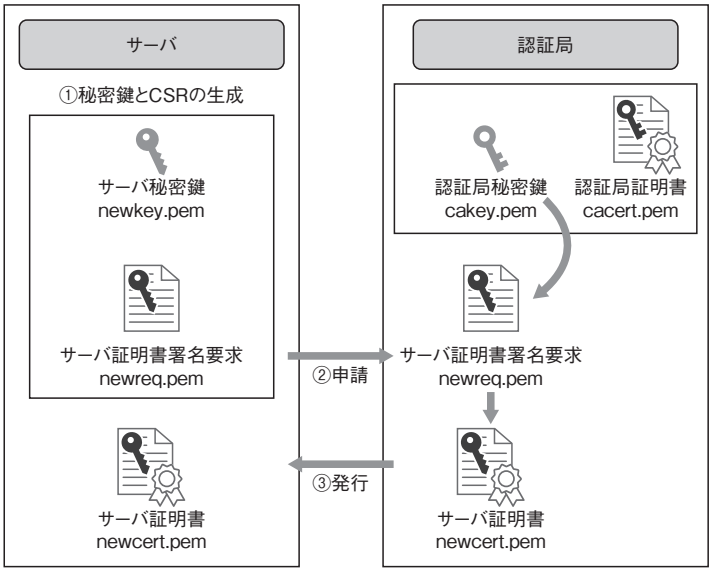
サーバ証明書とクライアント証明書の要求、署名、管理

ここでは、OpenSSLを使用したサーバ証明書とクライアント証明書の作成について説明します。

●サーバ証明書の作成

サーバ証明書とは、クライアントがサーバの信頼性を確認し、安全な通信を行うために利用される証明書であり、サーバ側で作成した証明書署名要求（CSR）を基に、認証局が署名することで発行されます。サーバ証明書の発行手続き例は以下のとおりです。

【サーバ証明書の発行例】



①対象となるサーバにて秘密鍵と証明書署名要求（CSR）を作成
②CSRを認証局に送付し、証明書の発行を申請
③認証局は審査を行い、認証局の秘密鍵でCSRに署名し、サーバ証明書を発行

【例：サーバ証明書の作成】

以下の(1)～(3)は、サーバ証明書を作成する一連の例です。まず、設定ファイル**/etc/pki/tls/openssl.cnf**(証明書を利用するサーバ側)を編集します。

例 (1)[usr_cert]セクション

```
basicConstraints = CA:FALSE  ←FALSEとなっていることを確認
```

例 (2)サーバ用の秘密鍵とCSRの作成(証明書を利用するサーバ側)

```
# openssl req -new -keyout newkey.pem -out newreq.pem -days 365
```

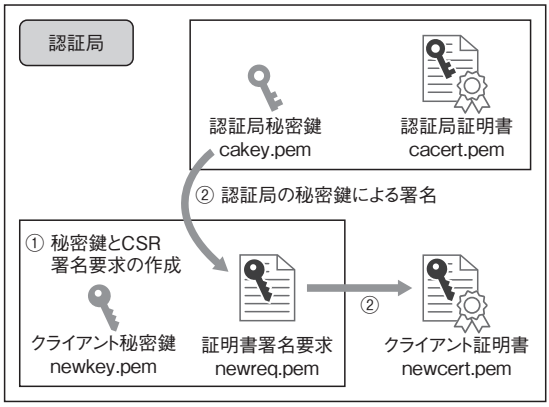
例 (3)CSRに署名を行い、サーバ証明書を作成

```
# openssl ca -policy policy_anything -out newcert.pem -infiles newreq.pem
```


●クライアント証明書の作成

クライアント証明書とは、サーバがクライアントの信頼性を確認するための証明書であり、特定のクライアントからのアクセスのみ許可するために利用されます。クライアント証明書の発行手続き例は以下のとおりです。

【クライアント証明書の発行例】



① 認証局にて秘密鍵と証明書署名要求（CSR）を作成
② 認証局の秘密鍵にてCSRに署名を行い、クライアント証明書を発行