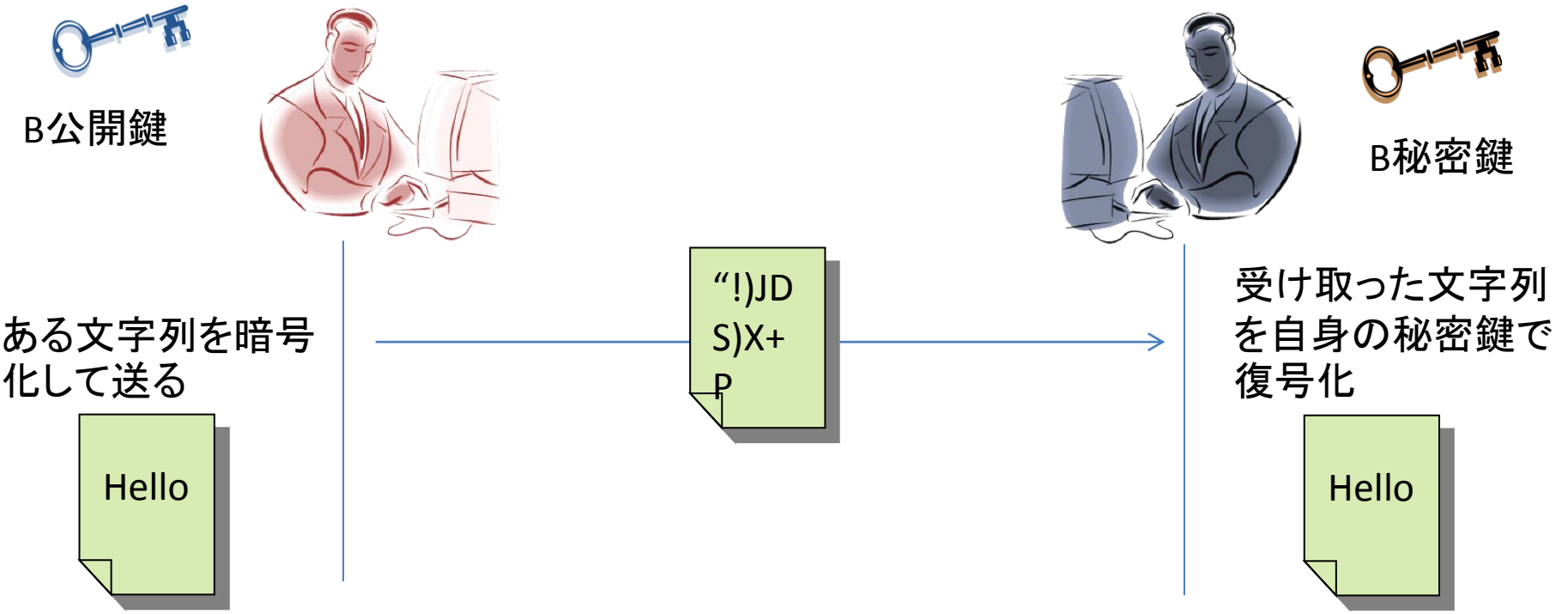
暗号技術入門

高瀬　亘

内容

- 公開鍵暗号方式
- デジタル証明書
- デジタル署名
- PKI
- SSLを用いたWebサイトの構築例

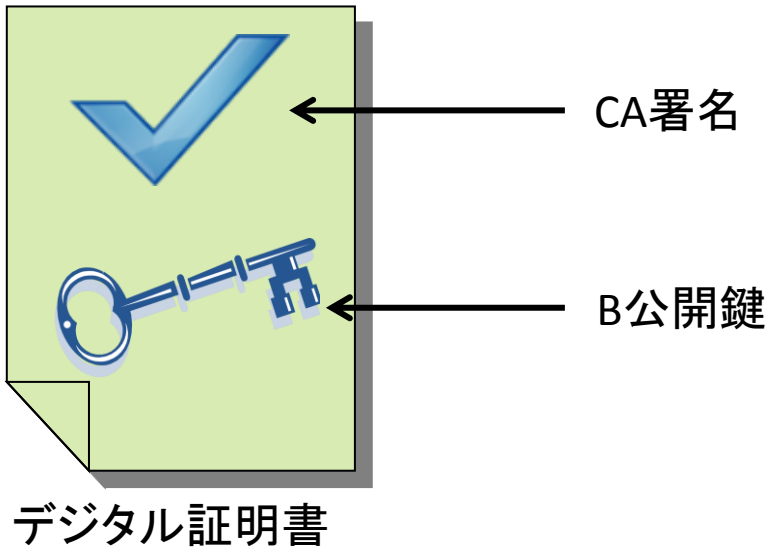
公開鍵暗号方式



- 公開鍵で暗号化し、秘密鍵で復号化する
- 公開鍵はあらかじめ配布しておく
- 秘密鍵は自身で厳重に保管
- お互いがお互いの公開鍵を持ち合うことで暗号化された通信が可能となる

公開鍵の信頼性

- 公開鍵の信頼性を検証するために、第三者(CA)が署名を行う
- 公開鍵(+αの情報)に署名が加わったものをデジタル証明書と呼ぶ



デジタル証明書

- X.509 v3証明書の中身
  - 公開鍵(例.Bの公開鍵)
  - 署名(例.CAの署名)
  - バージョン
  - シリアルナンバー
  - アルゴリズム関連情報
  - 発行者
  - 有効期限
  - 認証対象

+αの情報

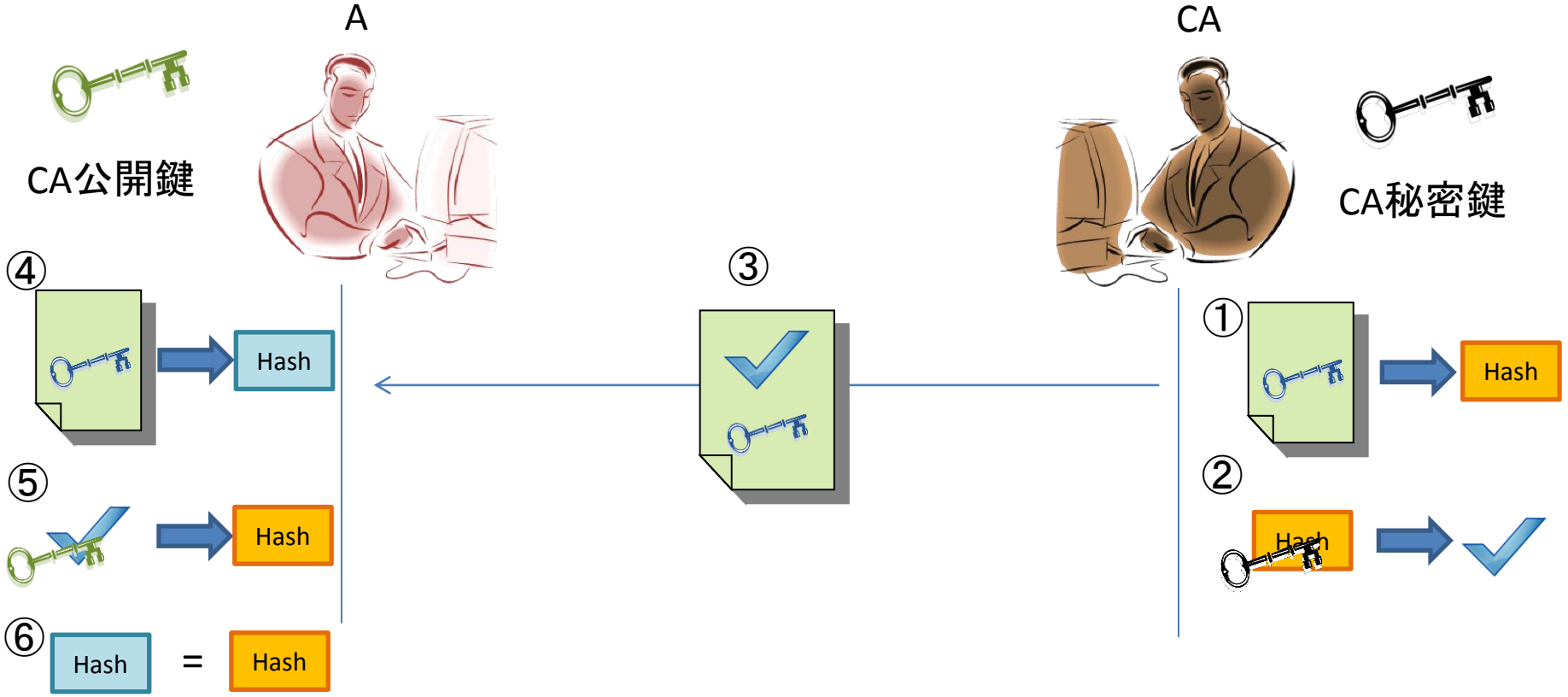
デジタル署名

- CAはCA秘密鍵で署名を行い、AはCA公開鍵で署名を検証する
- 検証が正しければAはBの公開鍵を手に入れたことになる
- CA公開鍵はあらかじめ配布しておく
- 上記の形でCAがデジタル証明書を配布する

署名へのRSAアルゴリズムの利用

- RSAアルゴリズム
  - 公開鍵で暗号化、秘密鍵で復号化に加え、秘密鍵で暗号化、公開鍵で復号化が行える
  - 改ざんを防ぐこと、送信者の検証が目的
    - デジタル署名の仕組みで盗聴は防げない

署名へのRSAアルゴリズムの利用



① CAはBの公開鍵(+α)から、ハッシュ関数(MD*やSHA-*等)を用いてハッシュ値を算出

② CAは、算出したハッシュ値をCA秘密鍵で暗号化(これが署名となる)

③ CAはBの公開鍵(+α)に署名を添えてAに送る(これがデジタル証明書となる)

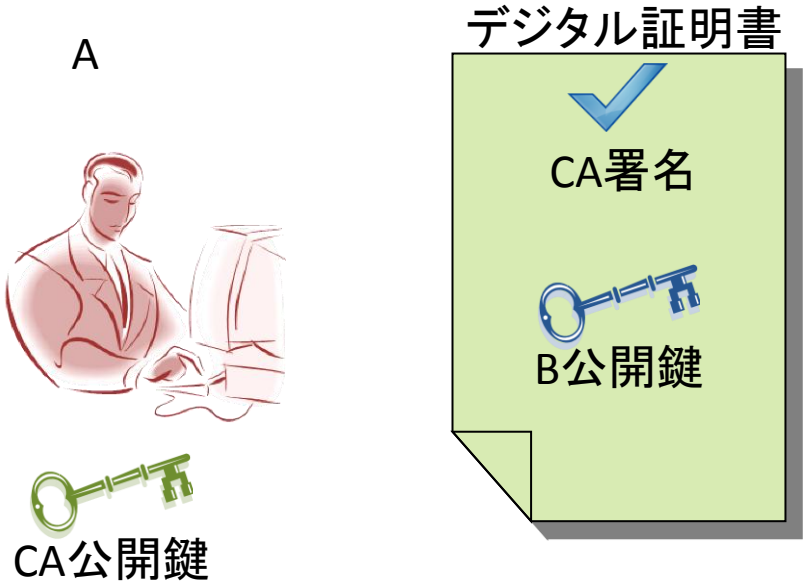
④ Aは、CAと同じハッシュ関数(証明書内に書いてある)を用いて受け取った公開鍵(+α)からハッシュ値を算出

⑤ Aは、署名を公開鍵で復号化してハッシュ値を得る

⑥ Aは、復号化したCAのハッシュ値と、A側で算出したハッシュ値が一致するか確認(検証)

証明書の信頼性

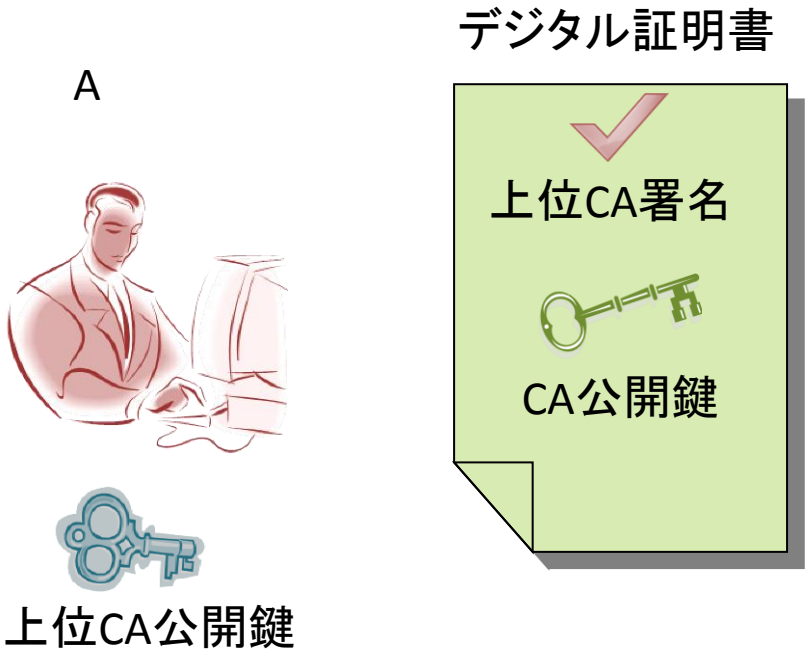
- 署名した秘密鍵に対応する公開鍵があれば、その公開鍵で署名を検証することで証明書の信頼性を確認できる
  - 証明書の改ざんを検知できる



暗号化された証明書を、CAの公開鍵で検証できれば、証明書内のBの公開鍵は信頼できるものといえる

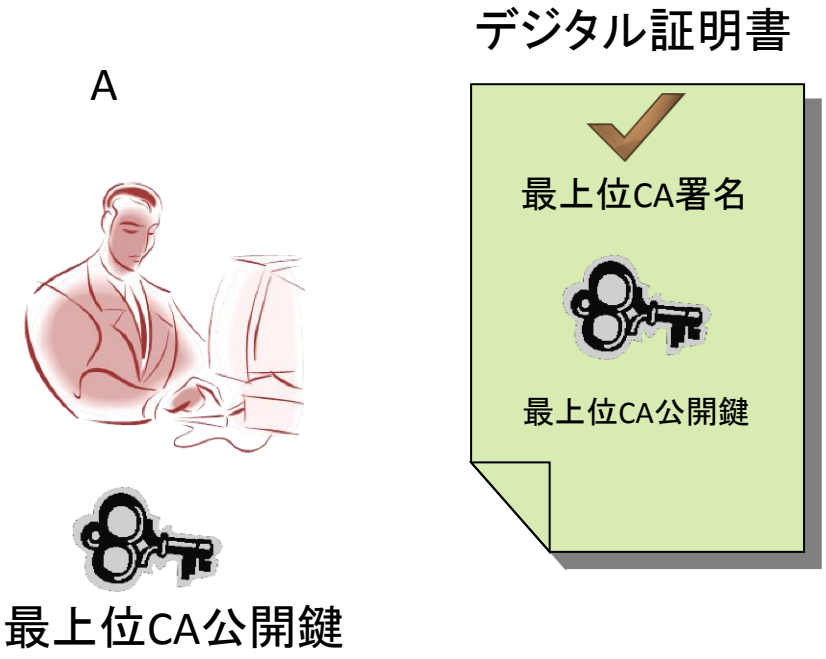
CAの公開鍵の信頼性

- CAの公開鍵は、上位CAの署名がある証明書で信頼性をえる



最上位CAの信頼性

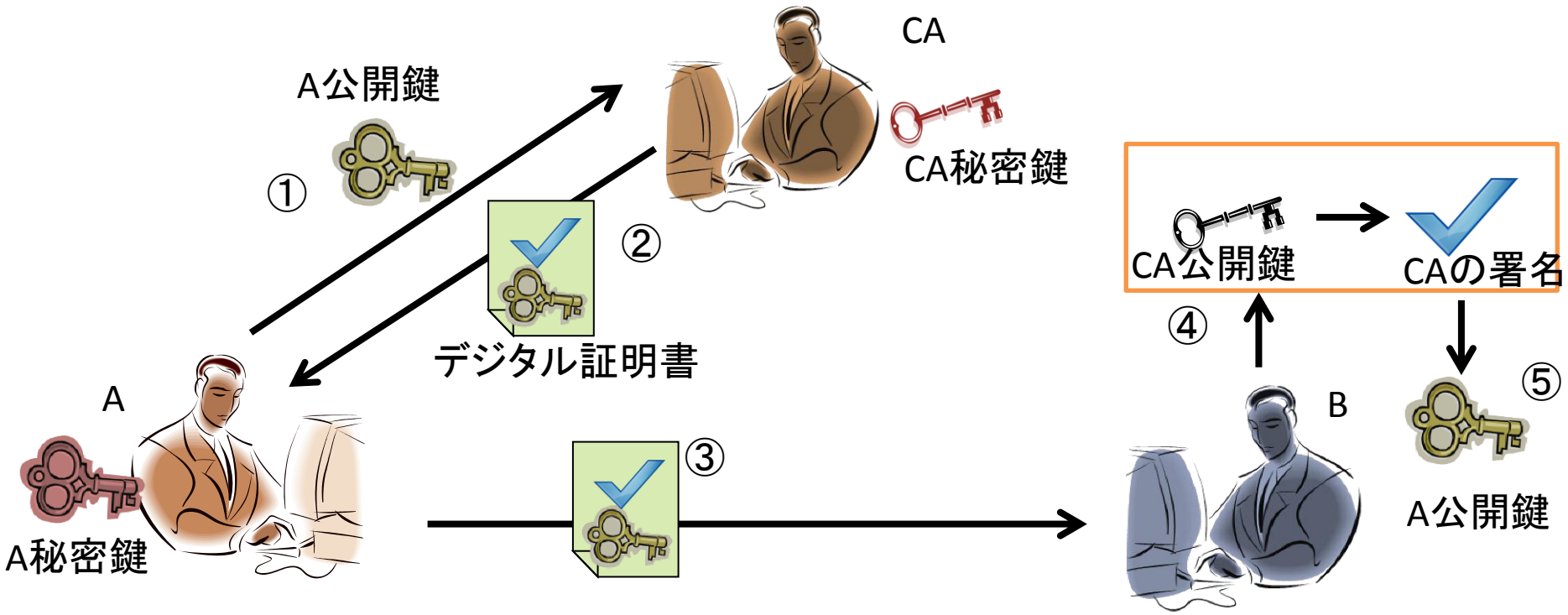
- 最上位CAは自身で証明書に署名する
- 証明書以外の信頼できる証が必要



自己署名証明書

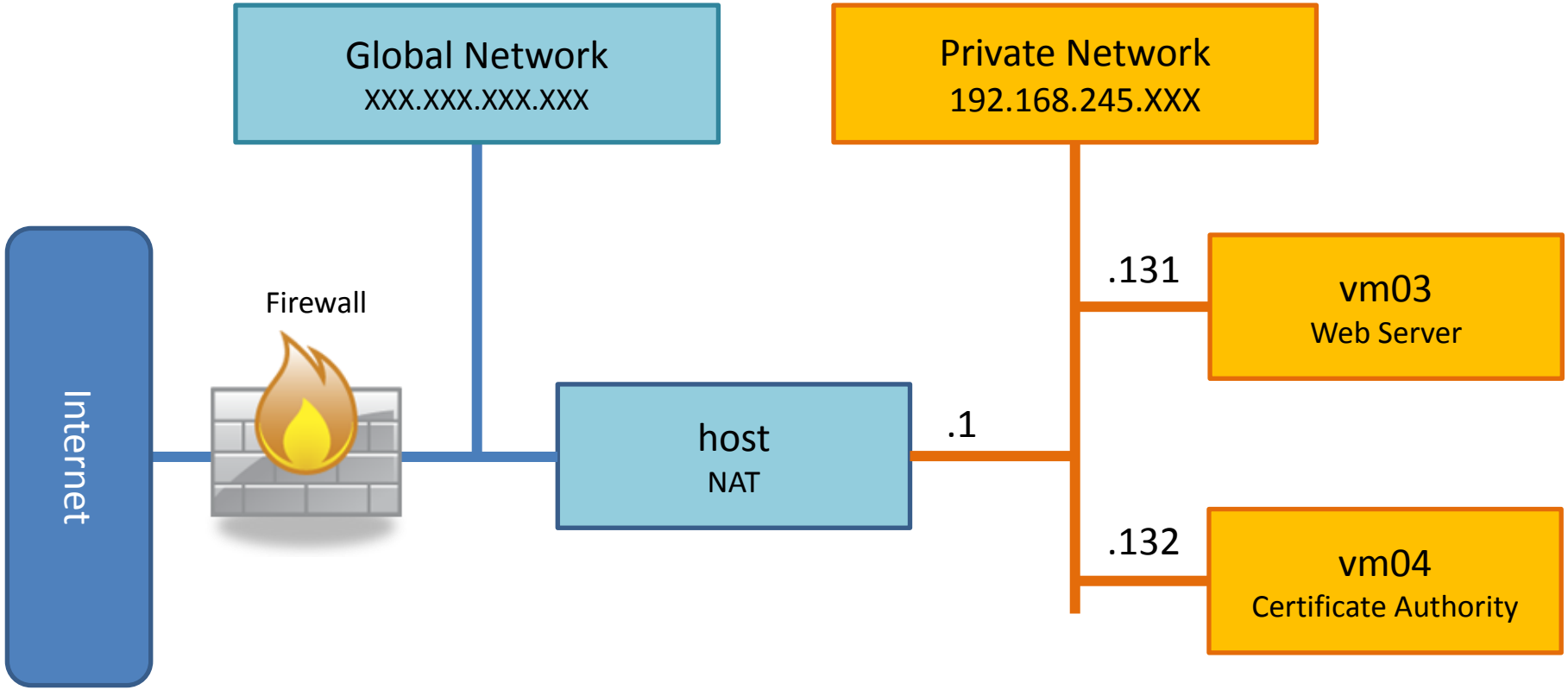
- 証明書内の公開鍵に対応する秘密鍵で署名した証明書
  - 最上位CA(ルートCA)はこれを発行することになる
    - 自身の秘密鍵で署名して証明書を発行
- 通常は第三者(CA)が署名を行う
  - Aの公開鍵に対して、CAが署名して証明書を発行

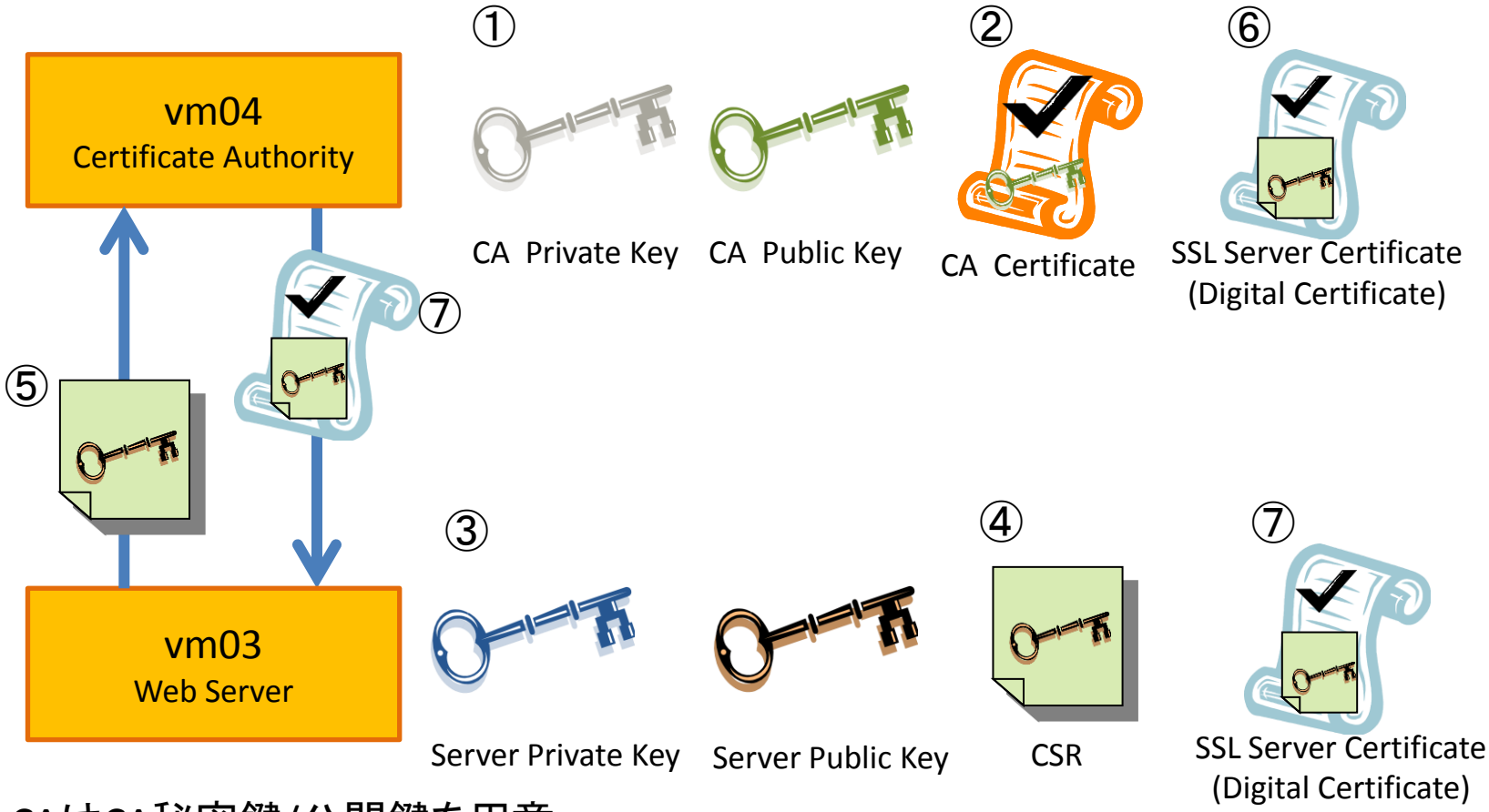
PKI



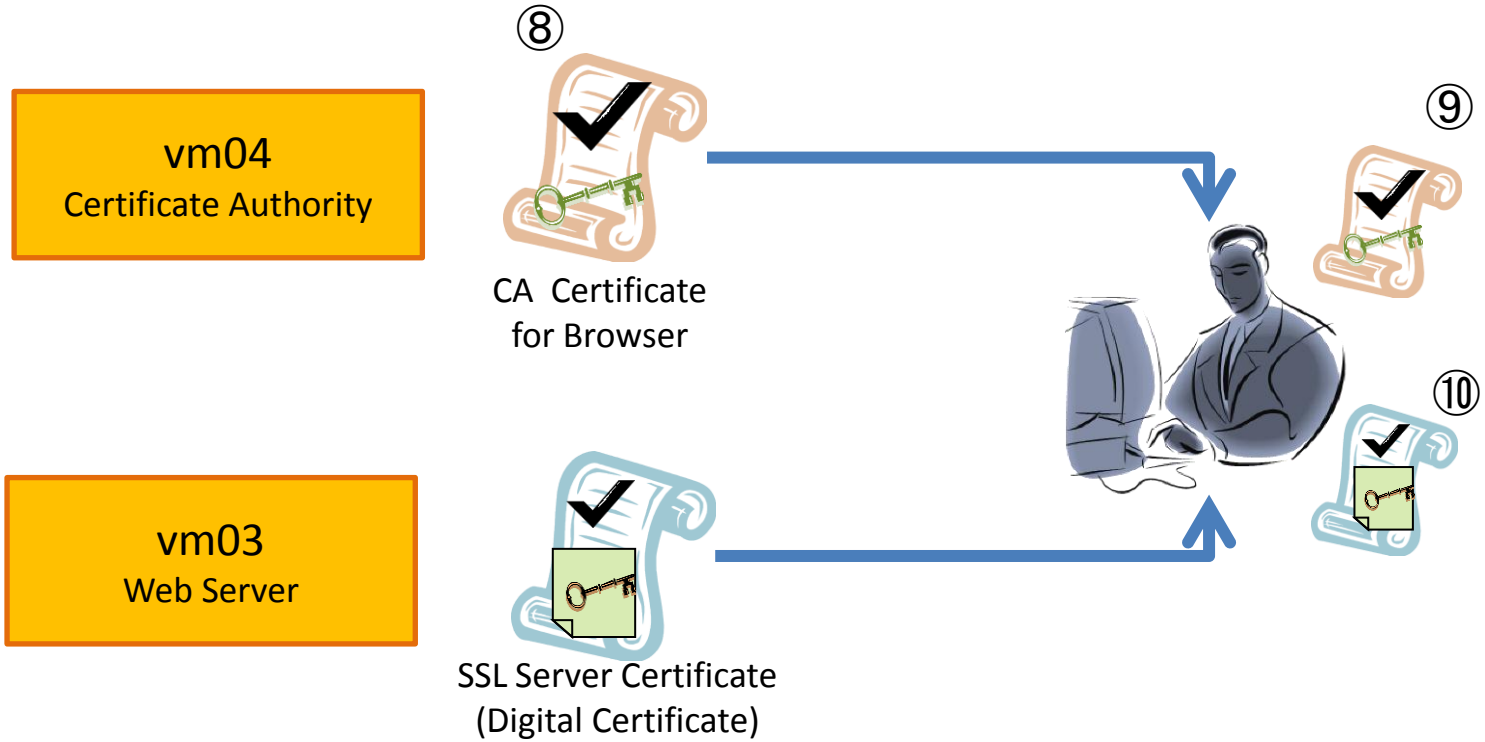
1. Aの公開鍵をCAに登録
2. CAがAの公開鍵に自身の署名をして返信（Aはデジタル証明書を受け取る）
3. Bと通信を開始前にAのデジタル証明書をBに渡しておく
4. Bは、Aの証明書の署名を、予め持っていたCAの公開鍵で検証
5. 検証がOKなら、Bは信頼できるAの公開鍵を得たことになる
6. Aは秘密鍵を、BはAの公開鍵を持っているのでセキュアな通信ができる

SSLを用いたWebサイトの構築例

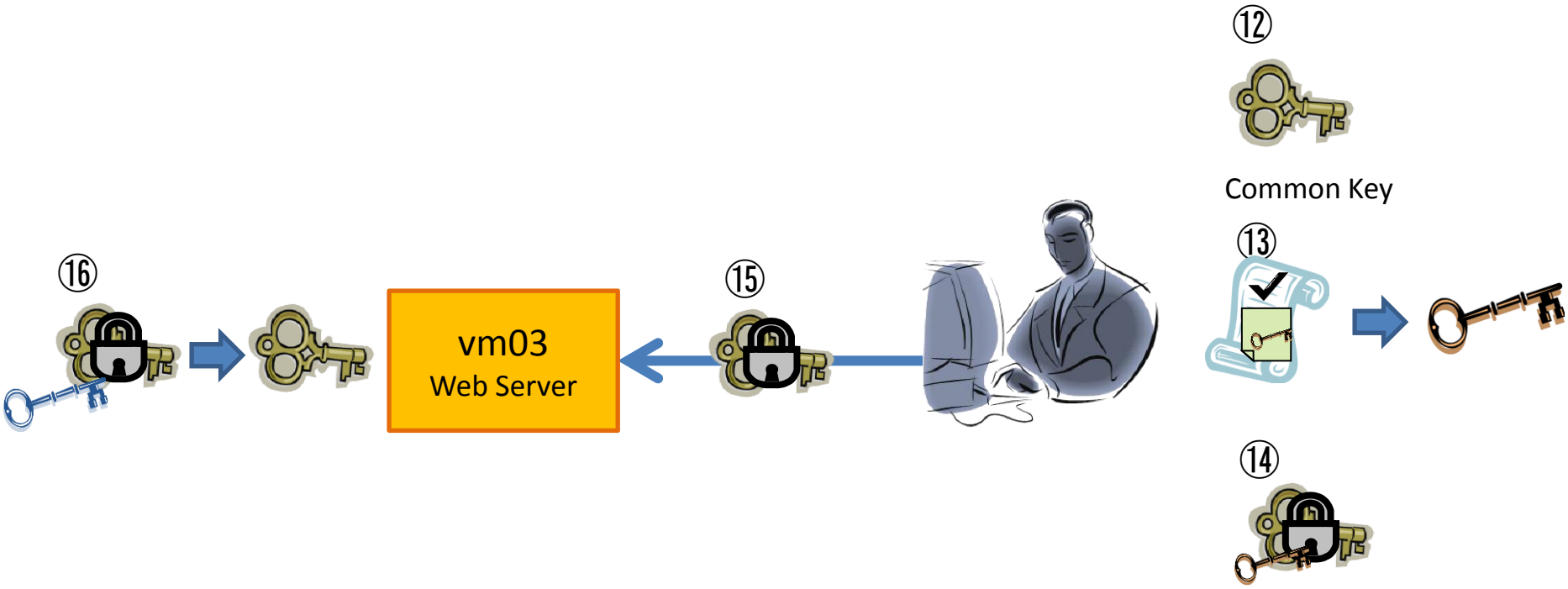




1. CAはCA秘密鍵/公開鍵を用意
2. CAはCA証明書(CA秘密鍵による署名とCA公開鍵を含む)を作成
3. SSLを利用したいWebサーバ(以下、サーバ)はサーバ秘密鍵/公開鍵を用意
4. サーバはCSR(Certificate Signing Request ; サーバ公開鍵を含む)を作成
5. サーバはCAにCSRを渡す
6. CAはSSLサーバ証明書(デジタル証明書 ; CA秘密鍵による署名とCSRを含む)を作成
   - 本人証明データの確認、ドメイン認証、組織認証を行い、CSRの正当性を確認
7. CAはサーバにSSLサーバ証明書を送る



8. CAはブラウザ用のCA証明書(PEMからDERに変換したもの)を作成してユーザへ配布できる状態にする
9. ブラウザはCA証明書をインポートする
10. サーバはブラウザからアクセスがあった時、ブラウザへSSLサーバ証明書(デジタル証明書)を送る
11. ブラウザは、受け取ったSSLサーバ証明書が信頼されたCAが発行したものかを検証する
    - ブラウザにインストールされているCA証明書内の公開鍵を使用し、SSLサーバ証明書についているCAの署名を検証
    - CA証明書がインポートされていない場合は警告が出る



12. ブラウザは共通鍵を作成
13. ブラウザはSSLサーバ証明書からサーバ公開鍵を取り出す
14. ブラウザはサーバ公開鍵を使用して共通鍵を暗号化する
15. ブラウザは暗号化した共通鍵をサーバに送る
16. サーバは受けとった共通鍵をサーバ秘密鍵で復号する
17. サーバ・ブラウザ間で共通鍵を使用して暗号化された通信が実現する

参考文献

- Simon Garfinkel, Gene Spafford, 安藤進訳, 1998年, Webセキュリティ＆コマース, 株式会社ウッドボックス
- 久米原栄, 2002年, UNIX Network セキュリティ管理, ソフトバンクパブリッシング株式会社
- 日本アイ･ビー･エム システムズ･エンジニアリング株式会社, 2004年, グリッド･コンピューティングとは何か Globus Toolkitではじめるグリッドの基礎, ソフトバンクパブリッシング株式会社
- 日本ベリサイン株式会社
  - https://www.verisign.co.jp/
- 暗号と署名の話
  - http://d.hatena.ne.jp/smoking186/20080302/1204451511