DNSSECチュートリアル

2011年7月

株式会社日本レジストリサービス

Copyright © 2011 株式会社日本レジストリサービス

目次

- DNSキャッシュへの毒入れとDNSSEC
- DNSSECのしくみ
- DNSSEC導入に向けて
- DNSSECの鍵と信頼の連鎖
- DNSSECのリソースレコード(RR)
- 鍵更新と再署名
- BINDキャッシュサーバーでのDNSSECの設定
- 鍵生成と署名作業
- BIND権威サーバーでのDNSSECの設定
- スマート署名 (Smart signing)
- DNSSEC化によるDNSデータの変化
- DNSSECのリスク
- DNSSECのまとめ

Copyright © 2011 株式会社日本レジストリサービス

DNSキャッシュへの毒入れと DNSSEC

Copyright © 2011 株式会社日本レジストリサービス

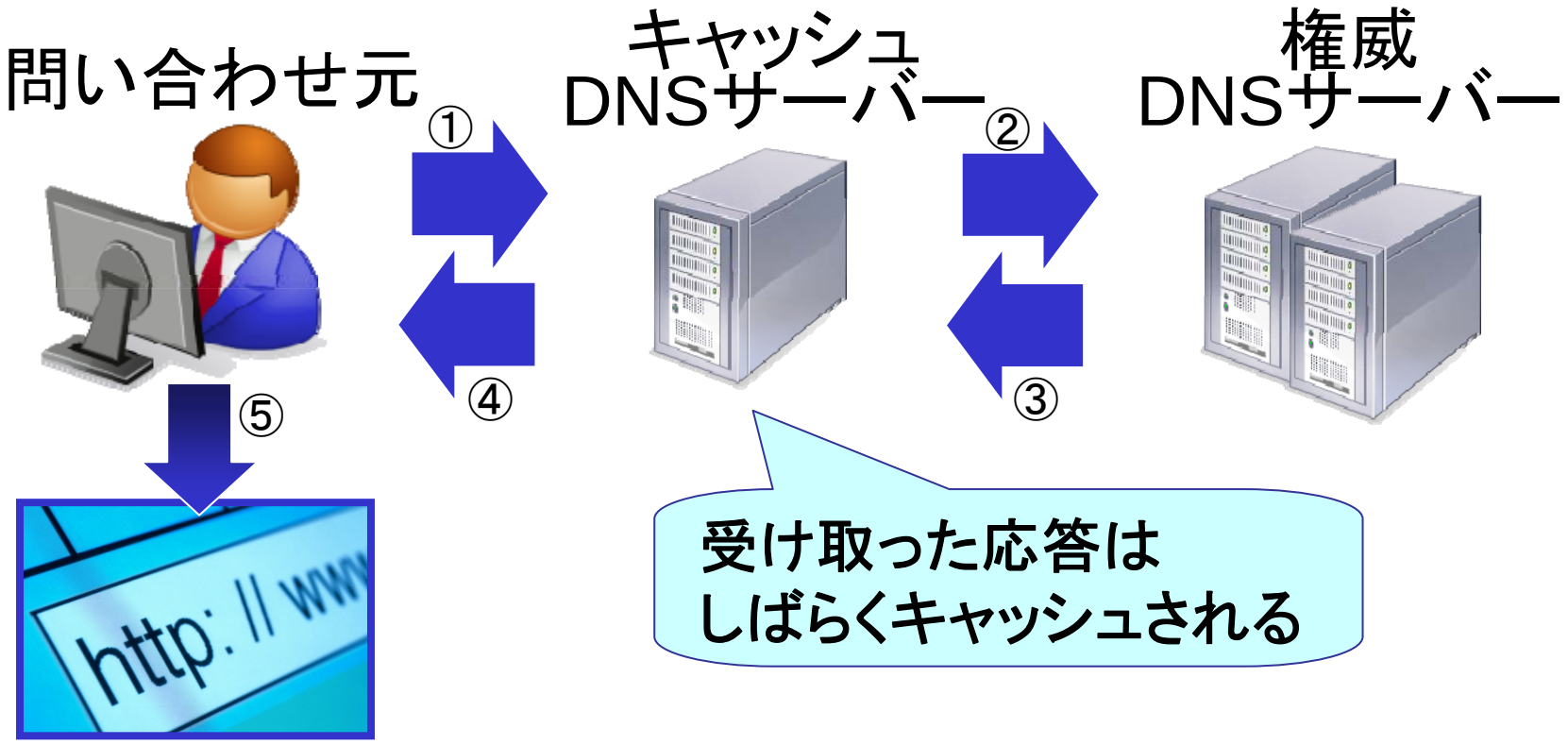
DNSへの毒入れ (キャッシュポイズニング)

- 予めキャッシュDNSサーバーに偽の情報を覚えこませ、ユーザーが正しいアクセスを行ったつもりでも、偽装サイトへ誘導する手法
  - フィッシングの為の攻撃手法の一つ

DNS最大級のリスク

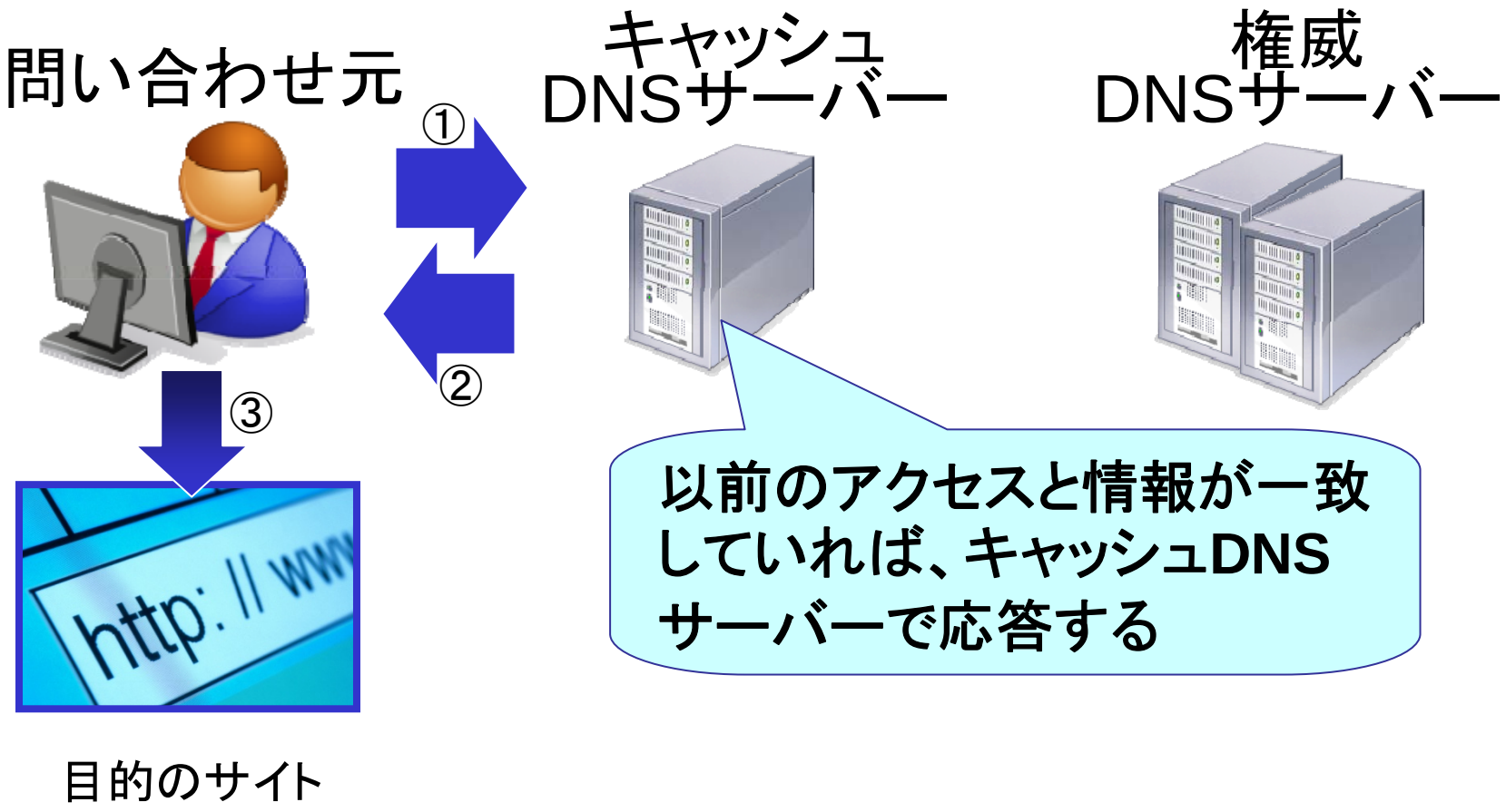
Copyright © 2011 株式会社日本レジストリサービス

DNSの正常な流れ (1回目のアクセス)



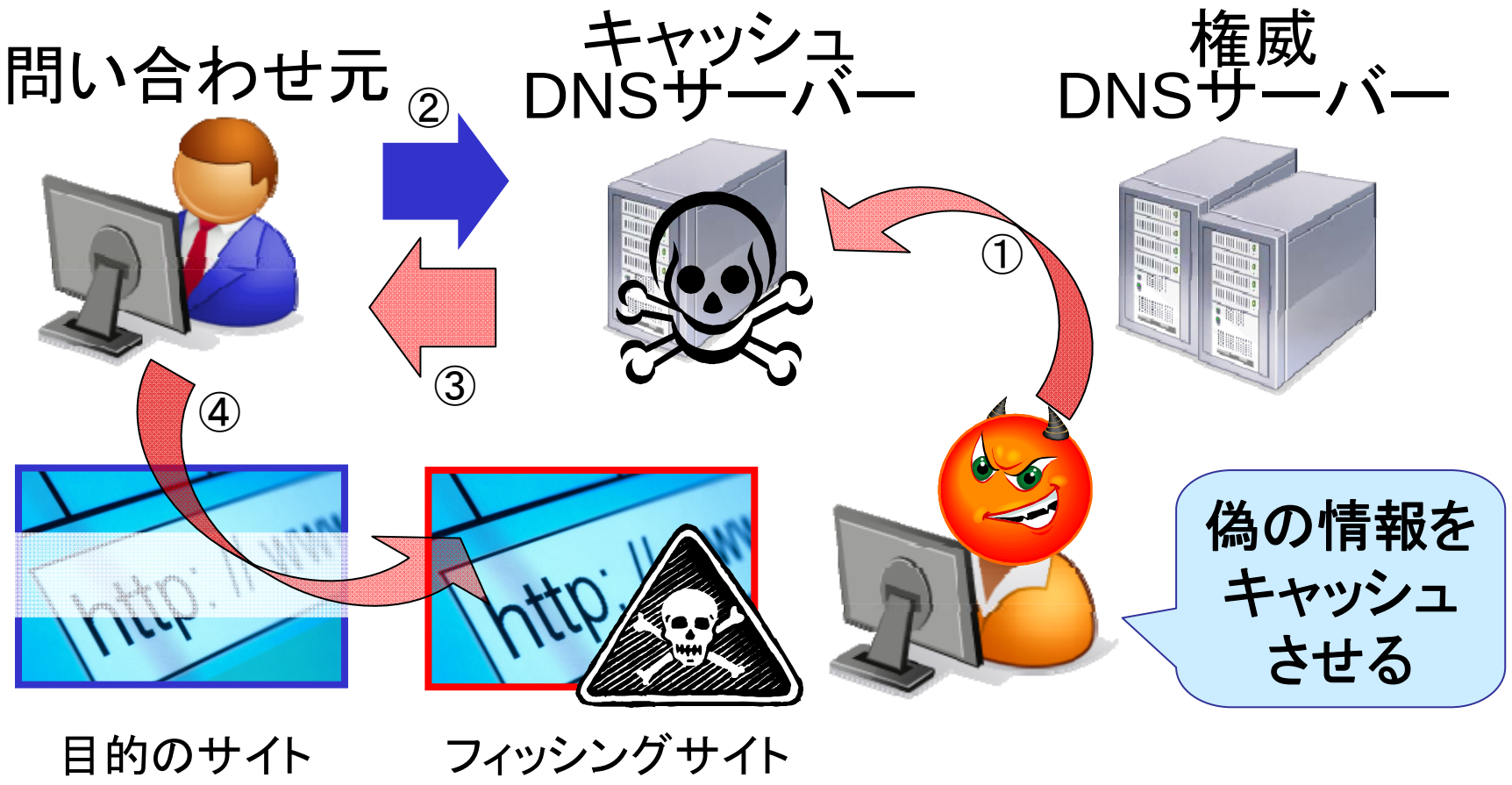
Copyright © 2011 株式会社日本レジストリサービス

DNSの正常な流れ(2回目以降)



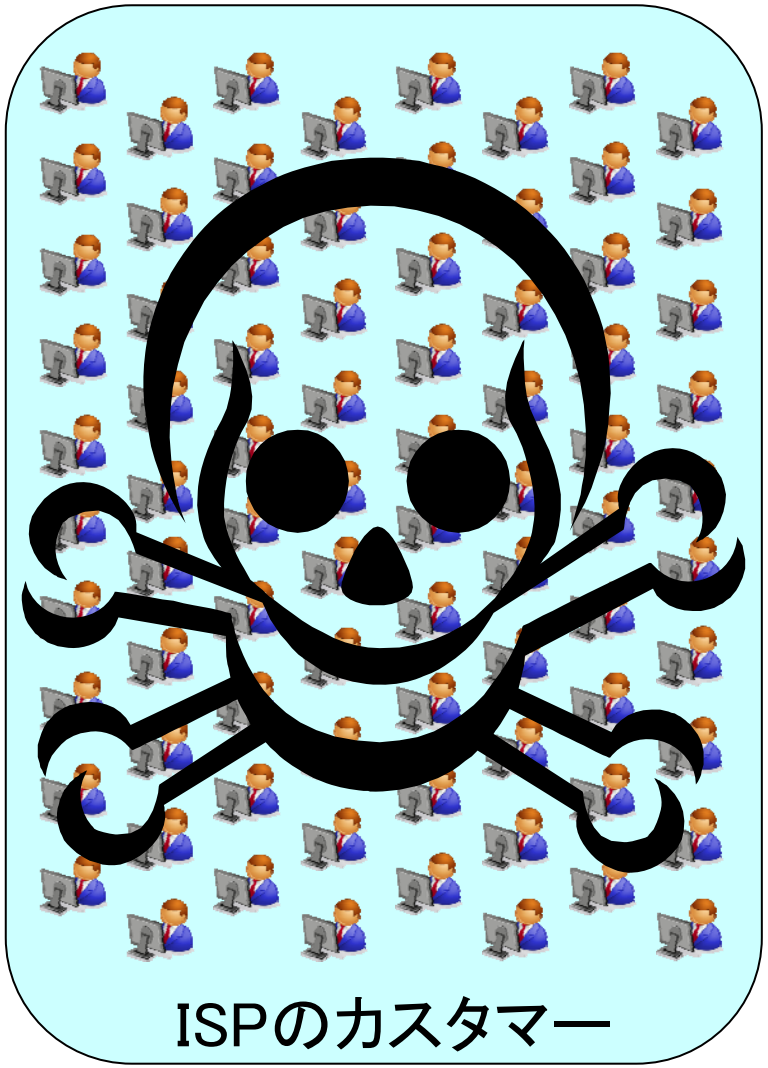
Copyright © 2011 株式会社日本レジストリサービス

DNSへの毒入れ攻撃

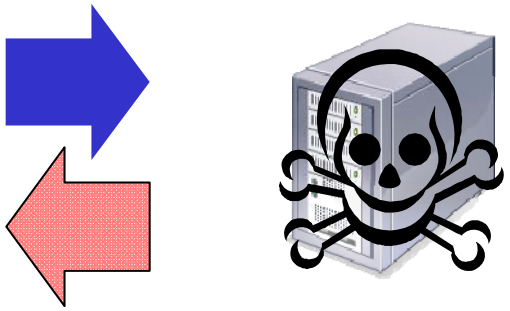


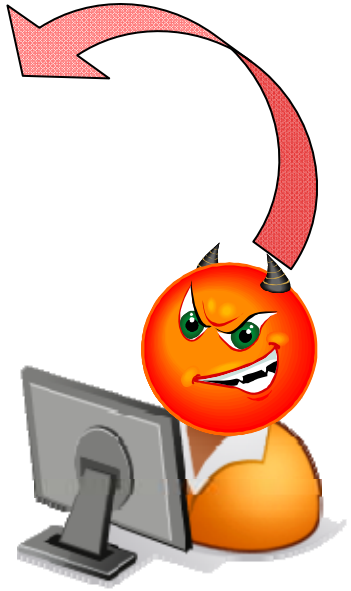
Copyright © 2011 株式会社日本レジストリサービス

ISPのキャッシュDNSサーバーが狙われたら



ISPのキャッシュ
DNSサーバー





カスタマー全員が
被害に会う

Copyright © 2011 株式会社日本レジストリサービス

毒入れ攻撃

- ユーザーは正常なアクセスを行っているつもりでも、フィッシングサイトに誘導される
  - 攻撃されたことに気づきにくい
- 同じキャッシュDNSサーバーのユーザー全員が影響を受ける
  - 大手ISPのキャッシュDNSサーバーが攻撃されると被害は甚大
- 攻撃そのものの検出が容易ではない
  - キャッシュへの毒入れは、見た目は通常のDNSパケットであるため、正常な応答と攻撃の区別が簡単ではない

Copyright © 2011 株式会社日本レジストリサービス

毒入れへの対策

- 暫定的な対策
  - 攻撃成功確率を下げるパッチや、その手法を取り込んだ実装の採用
    ⇒ 対症療法であり、執拗な攻撃には無力
- 根本的な対策
  - 毒入れはDNSプロトコルそのもののぜい弱性
  - 完全対処にはDNSのセキュリティ面でのプロトコル拡張が必要
    ⇒ このための技術が**DNSSEC**

Copyright © 2011 株式会社日本レジストリサービス

DNSSECのしくみ

Copyright © 2011 株式会社日本レジストリサービス

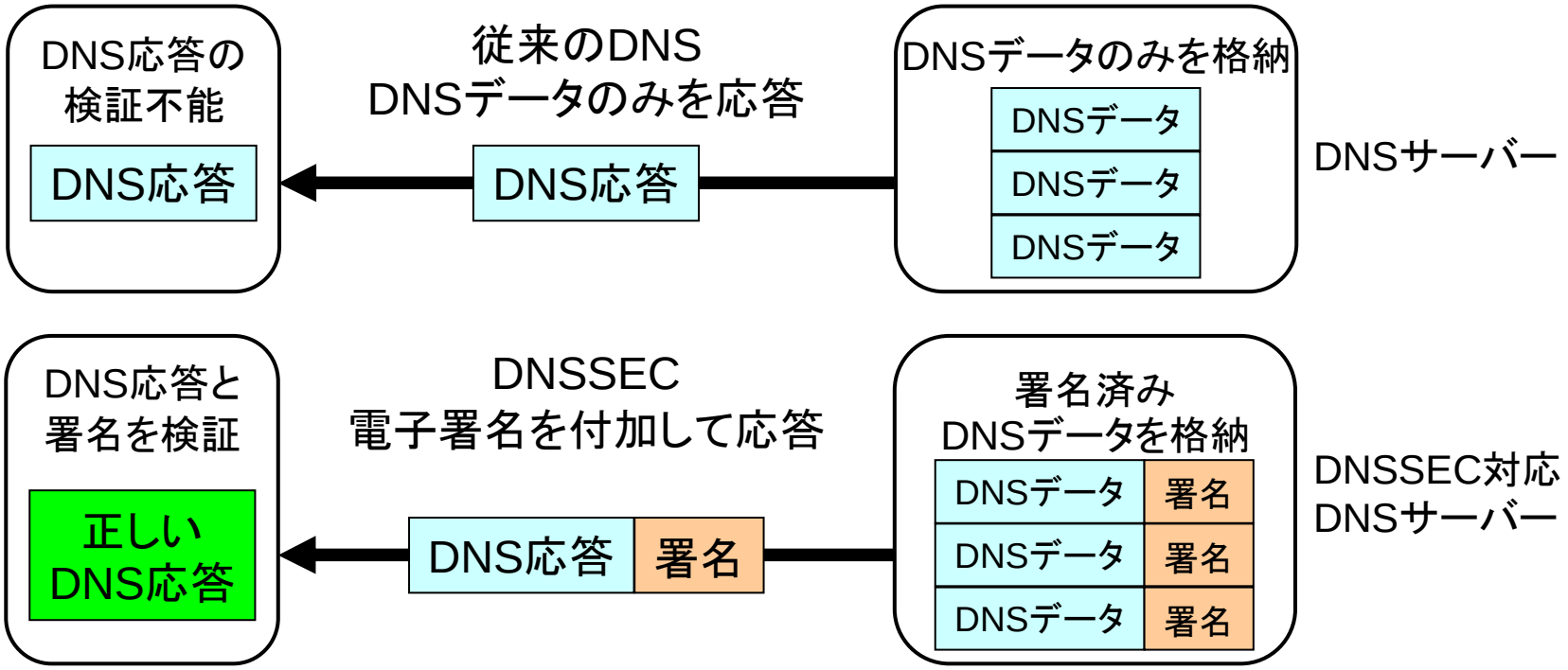
DNSSECとは

- DNSセキュリティ拡張(DNS SECurity Extensions)
  - **公開鍵暗号**の技術を使い、検索側が受け取ったDNSレコードの出自・完全性(改ざんのないこと)を検証できる仕組み
  - 従来のDNSとの**互換性を維持した拡張**
  - Kaminsky型攻撃手法の発覚を1つの契機に、多くのTLDが導入開始あるいは導入予定
- キャッシュへの毒入れを防ぐことができる現実解
  - 他の技術も存在するが標準化が成されていない

Copyright © 2011 株式会社日本レジストリサービス

従来のDNS vs DNSSEC

- DNSサーバーが応答に電子署名を付加し出自を保証
- 問合せ側でDNS応答の改ざんの有無を検出できる



Copyright © 2011 株式会社日本レジストリサービス

DNSSECのスコープ

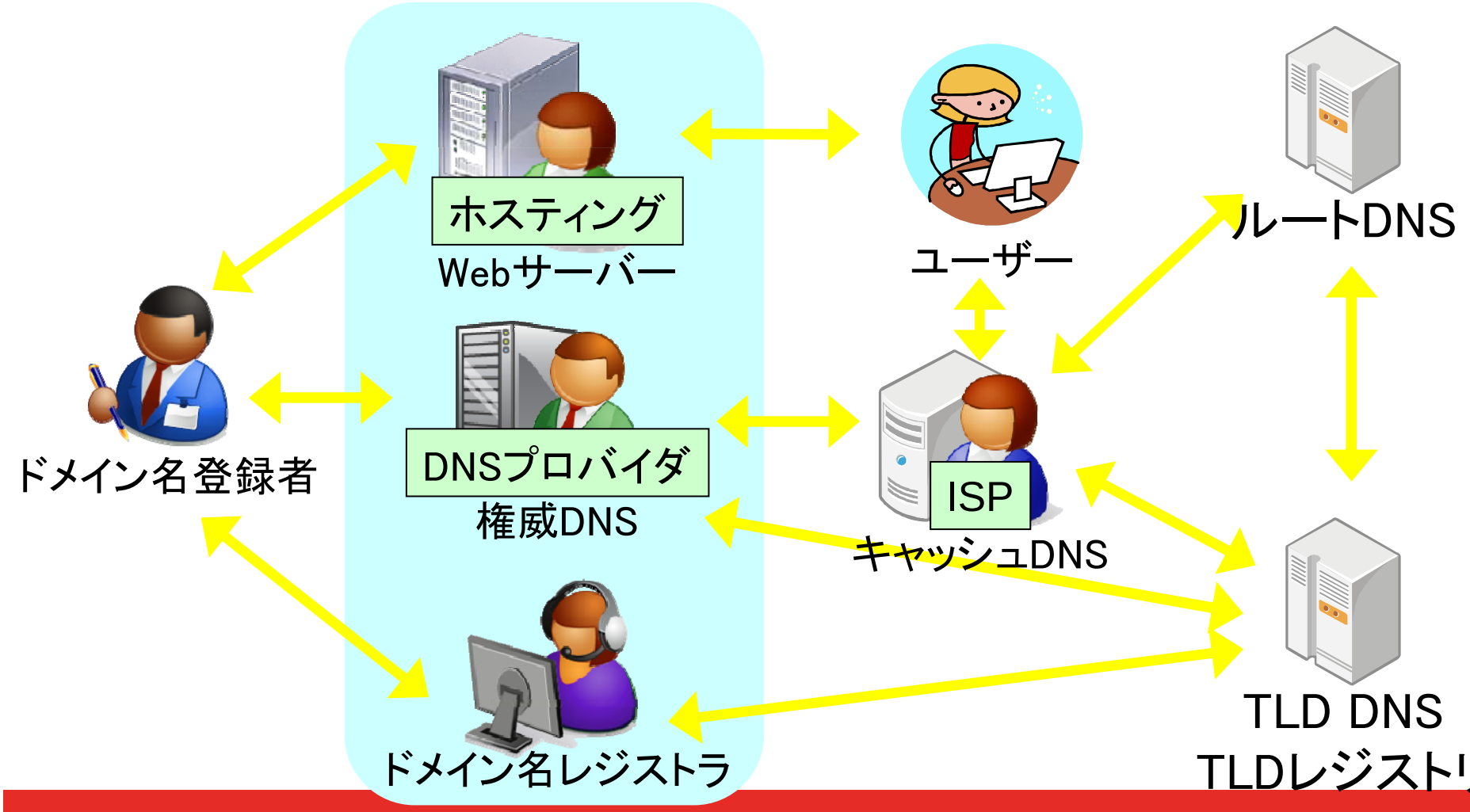
- 対象としているもの
  - DNS問合せに対する応答が、ドメイン名の正当な管理者からのものであることの確認
    ⇒ **出自の保証**
  - DNS問合せに対する応答における、DNSレコードの改変の検出
    ⇒ **完全性の保証**
- 対象としていないもの
  - 通信路におけるDNS問合せと応答の暗号化
    ※DNSレコードは公開情報という考え方から

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC導入に向けて

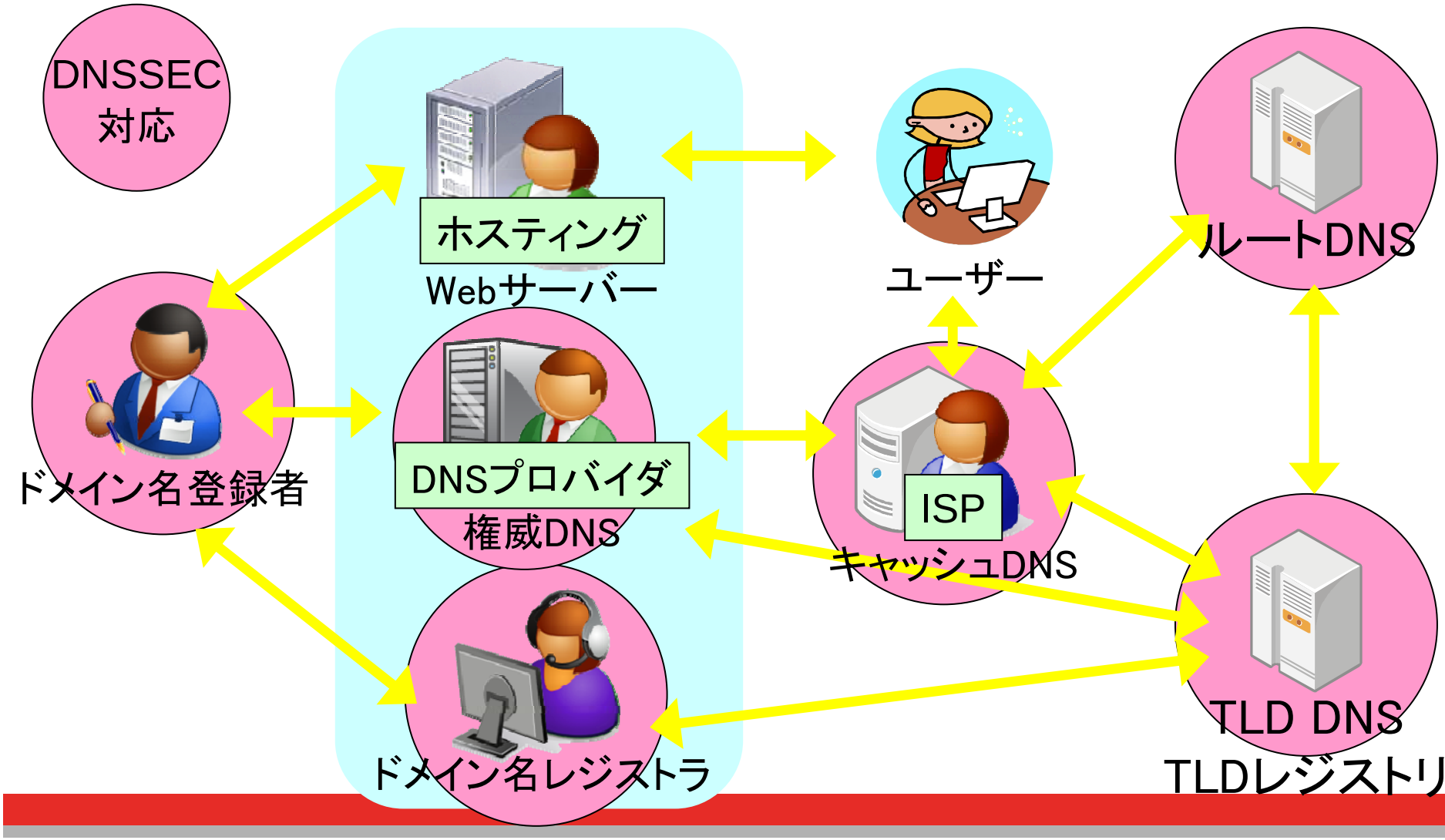
Copyright © 2011 株式会社日本レジストリサービス

DNS関係者と各情報の流れ



Copyright © 2011 株式会社日本レジストリサービス

DNSSEC対応が必要な関係者



Copyright © 2011 株式会社日本レジストリサービス

DNSSEC対応作業の概要

- ドメイン名登録者
  - DNSSEC導入の決定
- ドメイン名レジストラ
  - 鍵情報の上位レジストリへの取次ぎ
- TLD DNS、ルートDNS
  - 権威DNSサーバーのDNSSEC対応化
  - ゾーンへの署名
- DNSプロバイダ
  - 権威DNSサーバーのDNSSEC対応化
  - 秘密鍵・公開鍵を作成し、ゾーンに署名
- ISP
  - キャッシュDNSサーバーのDNSSEC対応化
  - (キャッシュDNSサーバーでの)署名の検証

Copyright © 2011 株式会社日本レジストリサービス

世界のDNSSEC導入状況 (2011年6月現在)

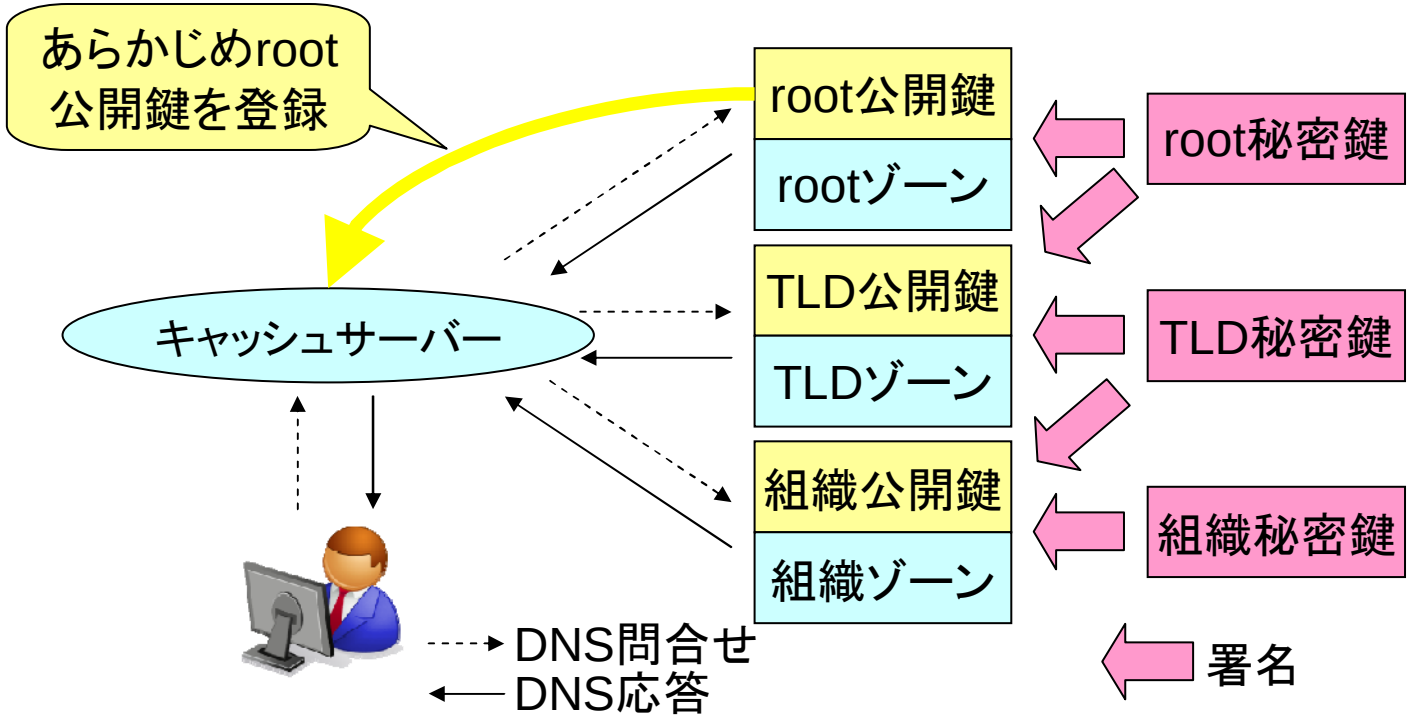
- rootゾーン
  - 2010年7月15日よりDNSSECの正式運用開始
- ccTLDの状況
  - 導入済: .se (スウェーデン) .eu (欧州連合) **.jp (日本)** .us (アメリカ) .in (インド) .de (ドイツ) など
  - 導入予定: .ca (カナダ) .cn (中国) .kr (韓国) など
- gTLDの状況
  - 導入済み: .org .com .net .edu .asia .cat .info .biz .museum など
  - 導入予定: .aero .mobi

Copyright © 2011 株式会社日本レジストリサービス

DNSSECの鍵と信頼の連鎖

Copyright © 2011 株式会社日本レジストリサービス

DNSSECの信頼の連鎖の概念図



- 秘密鍵で、自ゾーンと下位ゾーンの公開鍵に署名
- root公開鍵をキャッシュサーバーに登録することで、rootから組織ゾーンまでの信頼の連鎖を確立

Copyright © 2011 株式会社日本レジストリサービス

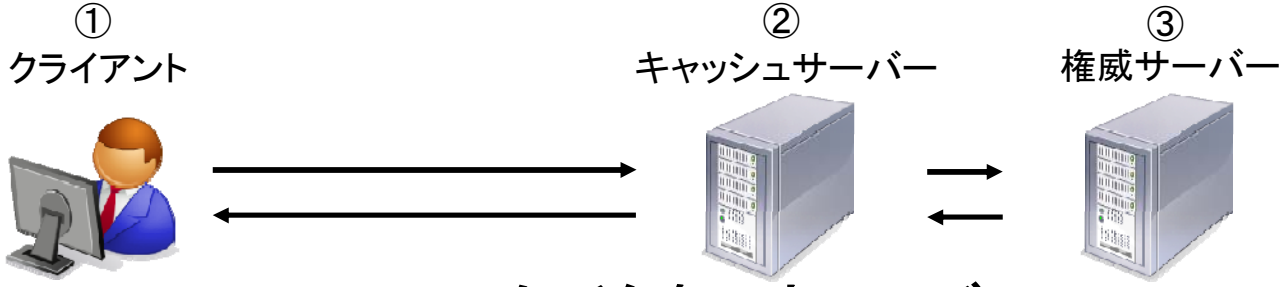
用語：バリデータ(Validator)

- DNSSECにおいて、バリデータは署名の検証を行うもの(プログラム、ライブラリ)を指す
- バリデータの所在
  - キャッシュサーバーが署名検証を行う場合、キャッシュサーバーがバリデータそのもの
    ⇒ 現状、もっとも一般的なDNSSECのモデル
  - WEBブラウザ等のDNS検索を行うアプリケーションが直接署名検証を行うモデルも考えられる

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC化による名前解決モデルの変化

- 従来のDNSでの名前解決モデル



- DNSSECでの名前解決モデル

  - 多くの場合バリデータは②に実装
  - バリデータが①に実装されていても問題ない

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC利用する2種類の鍵とDS

- 2種類の鍵
  - ZSK (Zone Signing Key)
    ゾーンに署名するための鍵
  - KSK (Key Signing Key)
    ゾーン内の公開鍵情報に署名するための鍵
- DS (Delegation Signer）
  - 上位ゾーンに登録するKSKと等価な情報

Copyright © 2011 株式会社日本レジストリサービス

ZSK

- 比較的暗号強度の低い鍵
  - 例えばRSAで1024bit等の鍵を使う
- 暗号強度が低い
  - 署名コストが低いため、大規模ゾーンの署名にも適応できる
  - 安全確保のため、ある程度頻繁に鍵を更新する必要がある
- 鍵更新は親ゾーンとは関係なく独立で行える

Copyright © 2011 株式会社日本レジストリサービス

KSK

- 比較的暗号強度の高い鍵
  - 例えばRSAで2048bitの鍵を使う
- 暗号強度が高い
  - 利用期間を長くできるため、鍵更新の頻度を低くできる
  - 署名コストは高いが、少数の鍵情報のみを署名対象とするため問題にはならない
- KSK公開鍵と暗号論的に等価な情報(DS)を作成し、親ゾーンに登録する
  - **KSKを変更する場合、同時にDSも更新する**

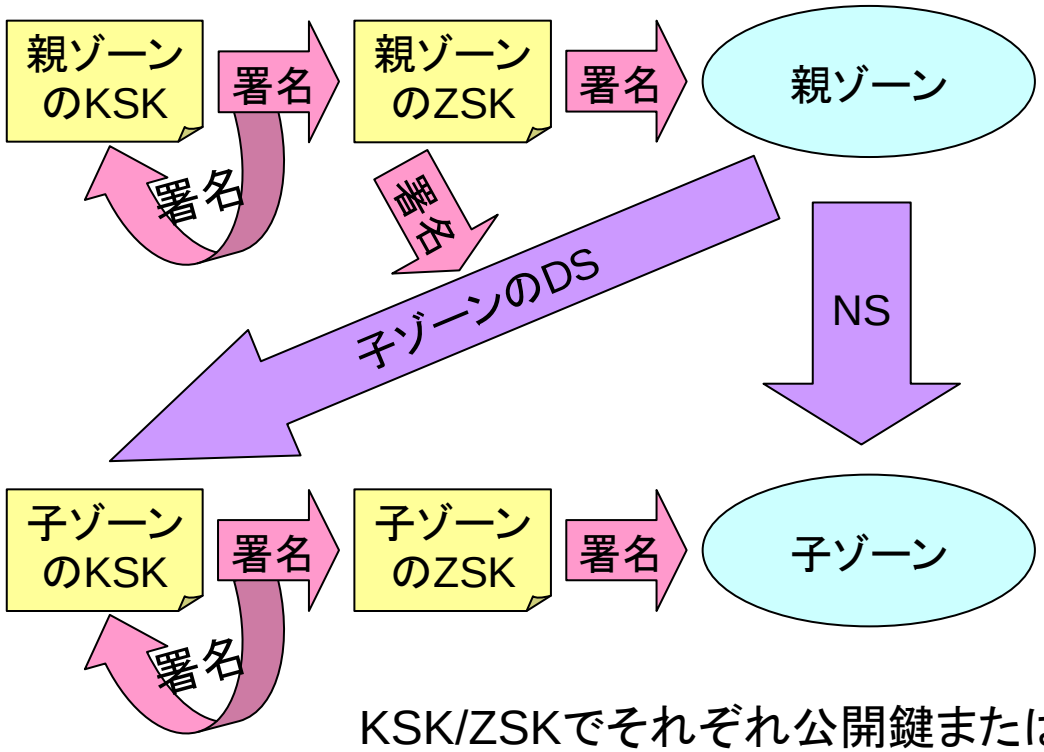
Copyright © 2011 株式会社日本レジストリサービス

DS

- KSK公開鍵を、SHA-1/SHA-256等のハッシュ関数で変換したDNSレコード
  ⇒ KSK公開鍵と等価の情報
- 親ゾーンの委任ポイントに、
  NSと共に子ゾーンのDS情報を登録
  - 親ゾーンの鍵でDSに署名してもらうことで、信頼の連鎖を形成する

Copyright © 2011 株式会社日本レジストリサービス

DNSSECの信頼の連鎖



KSK/ZSKでそれぞれ公開鍵またはDSが秘密鍵によって署名される。

- 公開鍵暗号による信頼の連鎖を形成
- キャッシュサーバーが、KSKの公開鍵を使って署名を検証
  ⇒**トラストアンカー**
  - キャッシュサーバーにはrootゾーンのKSK公開鍵を登録する

Copyright © 2011 株式会社日本レジストリサービス

DSとNSの本質的な違い

- NS : 委任先DNSゾーンデータが存在する(可能性のある)**サーバーを指し示す**
- DS: 委任先**DNSゾーンデータを直接指し示す**
  - DSは子ゾーンのKSK公開鍵と等価な情報
- NSの指し示すドメイン名がDNSSEC非対応であってもDNSSECの検証は問題無い

```
jpゾーンでの例
example.jp.  IN NS  ns0.example.ad.jp.
example.jp.  IN DS  2260 8 2 CC83B074566..............
```

  - example.ad.jpドメイン名はDNSSEC対応していなくても、example.jpドメイン名はDNSSEC検証可能

Copyright © 2011 株式会社日本レジストリサービス

DNSSECの
リソースレコード(RR)

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC関係のRR一覧

- DNSKEY　KSK・ZSK公開鍵の情報
- RRSIG　各RRsetへの署名
- DS　KSK公開鍵のハッシュ値を含む情報(親ゾーンに登録)
- NSEC　次のRRへのポインタと存在するレコード型の情報
- NSEC3　NSECを改良したもの(後述)
- NSEC3PARAM　NSEC3に必要な情報

Copyright © 2011 株式会社日本レジストリサービス

DNSKEY RR

- KSKとZSKの公開鍵を示すRR
  - オーナー名はゾーン頂点(=ゾーン名)
  - KSKとZSKを必要に応じて複数(後述)設定

```
example.jp. IN DNSKEY
    256 3 5 AwEAAeNO41ymz+Iw(行末まで省略)
     ①  ②③              ④
```

① フラグ(256:ZSK、257:KSK)

② プロトコル番号 (3のみ)

③ DNSSECアルゴリズム番号

④ 公開鍵(Base64で符号化)

Copyright © 2011 株式会社日本レジストリサービス

DNSSECアルゴリズム番号(抜粋)

番号	略称	参照
5	RSASHA1	[RFC3755] [RFC3110]
7	NSEC3RSASHA1	[RFC5155]
8	RSASHA256	[RFC5702]
10	RSASHA512	[RFC5702]

注) 5と7に差は無く、NSECとNSEC3 (後述) で使い分ける

DNSSECのDNSSECアルゴリズム番号一覧

http://www.iana.org/assignments/dns-sec-alg-numbers

Copyright © 2011 株式会社日本レジストリサービス

DS RR

- DS - Delegation Signer
  - 子ゾーンのKSKの正当性を親ゾーンで承認
  - 親ゾーンにのみ記述する唯一のRR

```
example.jp. IN DS 63604 5 1 DF…(16進数40文字)
example.jp. IN DS 63604 5 2 E8…(16進数64文字)
                  ①    ② ③        ④
```

① 鍵のID (16bit)
② DNSSECアルゴリズム番号
③ ハッシュのアルゴリズム(1:SHA-1，2:SHA-256)
④ ハッシュ化したKSK公開鍵

Copyright © 2011 株式会社日本レジストリサービス

RRSIG RR

- 各RRへの署名で、RRSet毎に存在する

```
ns.example.jp. IN RRSIG A 5 3 86400
                         ① ② ③   ④
    20091208144031 20091108144031 40002 example.jp.
          ⑤              ⑥          ⑦        ⑧
        NiVihYAIZBEwfUUAbPazDRIbvhNH8S(以下省略)
                              ⑨
```

① 署名対象のRRの種類
　　ここではns.example.jpのA RR

② DNSSECアルゴリズム番号

③ ラベルの数
”ns.example.jp”だと3、”*.example.jp”だと2

Copyright © 2011 株式会社日本レジストリサービス

RRSIG RR(続き)

④ 署名対象RRのTTL
⑤ 署名の満了時刻
⑥ 署名の開始時刻
⑦ 鍵のID
⑧ ドメイン名
⑨ 署名

署名は、元のRRの全て(TTL、クラス等を含む)と、RRSIGの署名そのものを除いた残りを含めて計算する

Copyright © 2011 株式会社日本レジストリサービス

DNSSECにおける不在証明

- DNSSECではドメイン名が存在しない場合、**存在しないこと**を**証明**する必要がある
  - 万が一存在しないドメイン名を偽装されたときのために、存在するのかどうかを検証できる仕組みが必須
- 存在するRRは署名(RRSIG RR)を付加して検証することで存在を証明できる
  ⇒ 存在しないRRは署名不能

Copyright © 2011 株式会社日本レジストリサービス

ハンバーガーのパティの有無

- パティ(肉)が有る
  - パティの存在を判断できる
- パティが無く、バンズ(パン)も無い
  - パティが無いかどうか判断不能
    - ⇒ 単純に配膳が遅れているだけ?
- クラウン(バンズ上部)とヒール(バンズ下部)があるのにパティが無い
  - クラウンとヒールの存在が判断できる
    - ⇒ パティが存在しないことを確実に判断できる

Copyright © 2011 株式会社日本レジストリサービス

NSEC RR

- NSECは存在しないものを証明(署名検証)するためのRR
  - 存在するレコードすべてを整列し、次のレコードへのリストを生成することで、存在しないものを証明する
  - NSEC RRにRRSIGを付加し署名検証を行う

Copyright © 2011 株式会社日本レジストリサービス

NSEC RRの例

- sec2.example.jpを問合せた場合の応答

```
sec1.example.jp.  IN  NSEC
   sec3.example.jp. NS DS RRSIG NSEC
```

(権威セクションで応答)

  - sec1.example.jp の次(アルファベット順)のドメイン名は sec3.example.jp
    ⇒ sec2.example.jp は存在しないことを示す
  - sec1.example.jp には NS, DS, RRSIG, NSEC のRRが存在する
    ⇒ NSECはRR TYPEの不存在も証明する

Copyright © 2011 株式会社日本レジストリサービス

NSEC3 RR

- NSECを使った不在証明では、NSEC RRを辿れば、完全なゾーンデータを入手できる
  ⇒NSEC方式はゾーンデータの公開と等価
  - Walker(DNSSEC Walker)というツールで、NSEC方式のDNSSEC化ゾーンのデータを入手可能
- **NSEC3** (RFC 5155)
  - ドメイン名を一方向性ハッシュ関数でハッシュ化したものを整列する

Copyright © 2011 株式会社日本レジストリサービス

NSEC3 RRの例

```
4HTJTU7UP56274L1C00Q9MLPHG2A2H85.example.jp.
 IN NSEC3
 1 0 3 123ABC ←NSEC3の関連パラメータ
B0B790UE4SAE4QB4RTB3PJSIH6JAOB7R NS DS RRSIG
```

NSEC RRと比べると

- ラベルがハッシュ化されBase32でエンコード
  - 元のドメイン名は推測不能
- NSEC3の関連パラメータを付加

Copyright © 2011 株式会社日本レジストリサービス

NSEC3の関連パラメータ

- 前スライドのRR例

```
1 0 3 123ABC
① ② ③     ④
```

① ハッシュアルゴリズム(1:SHA-1 RFC5155)

② NSEC3 オプトアウトフラグ
(1ならオプトアウト、0はオプトアウトしていない)

③ 繰り返し

④ ソルト(16進数で表記。例は3バイト分のソルト)

Copyright © 2011 株式会社日本レジストリサービス

NSEC3のハッシュ値計算方法

- ハッシュの計算アルゴリズム
  ① 値に**ソルト**を結合
  ② 次に**ハッシュアルゴリズム**でハッシュ値を計算
  ③ ①、②を**繰り返し**で指定された回数適用する
- 計算の元になる値は、小文字で正規化したドメイン名(のワイヤーフォーマット)
  – nsec3hash (BIND 9.7系以降に付属)で計算可

Copyright © 2011 株式会社日本レジストリサービス

NSEC3でのオプトアウト

- 一部の委任先がDNSSEC化している場合
  - 主に.JP、.COM等のTLDで該当
- ゾーン内のレコード全てにNSEC3 RRを用意すると、それに付随するRRSIGを含めた計算コストが膨大となる
  - DNSSEC化していない委任情報に、署名を付加する必要性は薄い
- 必要のある委任先にのみNSEC3 RRを用意

Copyright © 2011 株式会社日本レジストリサービス

NSEC3PARAM RR

```
example.jp. IN NSEC3PARAM 1 0 3 123ABC
```

- ゾーン提供(権威サーバー)側が、NSEC3の計算を行うために必要なレコード
  - NSEC3のパラメータを抜き出したもの
  - オーナー名はゾーン頂点(ゾーン名)

Copyright © 2011 株式会社日本レジストリサービス

NSEC3での権威サーバーの応答

- 存在しない名前の検索を受けた権威サーバー
  - クエリ名のハッシュ値を計算
  - あらかじめ整列してあるNSEC3 RRの中から、前後に該当するものを署名と共に権威セクションで応答
  - 実際は複数のNSEC3を応答
    (説明省略　RFC5155 Section 7.2参照)
- NSEC3のオーナー名の問合せ
  - ドメイン名としては存在しないもの
    ⇒ 名前エラー(Name Error)を応答する

Copyright © 2011 株式会社日本レジストリサービス

NSECとNSEC3比較

	NSEC	NSEC3
ゾーンデータの秘匿性	無	有
ハッシュ値を求めるための計算コストの増加	無	有

- NSEC3はNSECに比較しドメイン名の秘匿性は高まるが、ハッシュ計算のためのコストが増加する
  ⇒ NSEC3とNSECは用途に応じて使い分ける
  - ゾーンデータを秘匿する必要が無い場合、NSECのほうが各DNSサーバーの負荷の増加を抑えられる

Copyright © 2011 株式会社日本レジストリサービス

www.example.jpのAの署名検証 (1)

① 上位からNSとDSを受け取る
- JPの権威サーバーからexample.jpのDS (DSの署名検証の解説は省略)とNSの情報を受け取る

② 当該ゾーンのDNSKEYを受け取る
- example.jpの権威サーバーから、example.jpのDNSKEY(複数)とRRSIG(複数)を受け取る

③ KSKを特定する
- KSKとDSの鍵ID、DNSSECアルゴリズム番号を比べ、KSKを特定

Copyright © 2011 株式会社日本レジストリサービス

www.example.jpのAの署名検証 (2)

④ KSKを認証する
- DSのハッシュアルゴリズムに従ってKSKのハッシュ値を計算し、DSにあるハッシュ値と比較してKSKを認証する

⑤ DNSKEYを認証する
- ②で受け取ったDNSKEYに付随したRRSIG(複数)の鍵IDからKSKの鍵IDと一致するものを識別し、署名検証を行いDNSKEYを認証する

⑥ www.example.jpのAを受け取る
- example.jpの権威サーバーから、AとRRSIG(1個以上)を受け取る

Copyright © 2011 株式会社日本レジストリサービス

www.example.jpのAの署名検証 (3)

⑦ RRSIGからZSKを識別する
- RRSIGの情報(鍵ID等)に一致するDNSKEY内にあるZSKを識別する

⑧ www.example.jpのAを認証する
- ZSKで署名を検証する

- 必ずしも処理はこの順番ではなく、実装に依存する
- 鍵IDが衝突した場合は実際に計算して識別する
- 署名検証の際、署名の有効期間、ドメイン名など他のRRSIGのパラメータもチェックされる
- DSやDNSKEY、RRSIG等は、署名検証後もTTLの有効時間キャッシュする

Copyright © 2011 株式会社日本レジストリサービス

鍵更新と再署名

Copyright © 2011 株式会社日本レジストリサービス

鍵更新

- 鍵更新: Key rollover
  - 同じ鍵を長期間使い続けると、様々なリスクが生じる
  - 不注意、偶発的事故、鍵の盗難、暗号解読等
- リスクを最小に抑えるため、DNSSEC対応ゾーンの運用では定期的な鍵更新(鍵の交換)を行う
  - JPゾーンの場合、KSKを年次で、ZSKを月次で更新する運用を行っている

Copyright © 2011 株式会社日本レジストリサービス

鍵更新時に留意すべきこと

- 鍵更新は、DNSSECの信頼の連鎖が途切れないよう、注意深く作業する必要がある
  - 鍵情報(DSやDNSKEY)と署名(RRSIG)はDNSのレコードである
    ⇒ キャッシュサーバーはこれらを**キャッシュ**する
- キャッシュしている情報と、あらたにキャッシュサーバーが受け取る情報の整合性を確保
- 2種類の鍵更新手法
  - 事前公開法 (Pre-Publish Key Rollover)
  - 二重署名法 (Double Signature Key Rollover)

Copyright © 2011 株式会社日本レジストリサービス

ZSKの更新:事前公開法 (1/2)

① DNSKEYに新旧のZSKを登録する
- 新ZSKを作成し、旧ZSKと共にDNSKEYに登録し(この状態でKSKを含めてDNSKEYは最低3個)、旧ZSKでゾーンを署名
- DNSKEYのTTL時間(+セカンダリの転送時間)待つ
- 全てのキャッシュサーバーが新旧のZSKを含んだDNSKEYをキャッシュするようになり、旧RRSIGでも新RRSIGでも署名を検証できるようになる

② ゾーンの署名鍵を新ZSKに切り替える
- ゾーン内の最長のTTL時間(+セカンダリの転送時間)待つ
- 全てのキャッシュサーバーから旧ZSKで署名したRRSIGが無くなる

Copyright © 2011 株式会社日本レジストリサービス

ZSKの更新:事前公開法 (2/2)

③ 旧ZSKをDNSKEYから削除する

– DNSKEYは新ZSKとKSKの状態になる

	初期状態	①	②	③
DNSKEY	KSK 旧ZSK	KSK 旧ZSK **新ZSK**	KSK 旧ZSK **新ZSK**	KSK **新ZSK**
RRSIG	旧ZSKでの署名	旧ZSKでの署名	**新ZSKでの署名**	**新ZSKでの署名**

Copyright © 2011 株式会社日本レジストリサービス

ZSKの更新: 二重署名法

① 新ZSKを作成し、新旧両方のZSKでゾーンを署名
  – ゾーン内の最大TTL時間(+セカンダリの転送時間)待つ

② DNSKEYのZSKの旧ZSKを削除し、新ZSKでゾーンを署名

	初期状態	①	②
DNSKEY	KSK 旧ZSK	KSK 旧ZSK **新ZSK**	KSK **新ZSK**
RRSIG	旧ZSKでの署名	旧ZSKでの署名 **新ZSKでの署名**	**新ZSKでの署名**

時間の流れ

Copyright © 2011 株式会社日本レジストリサービス

ZSKの更新: メリット・デメリット

- 事前公開法
  - ○ ゾーンへの署名を2回行う必要が無い
  - × ZSKの公開時間が長くなるため、暗号解読攻撃のリスクが高まる(ZSKは鍵長が短い)
  - △ 初期状態から数えて4ステップ必要
  - ○ 次のZSKを常時公開することで、手順を簡略化可能
- 二重署名法
  - ○ 初期状態から数えて3ステップで終了する
  - × ゾーンへの署名を2回行う必要がある
  - × 鍵更新期間中(①の状態)はDNSデータが大きくなる

Copyright © 2011 株式会社日本レジストリサービス

KSKの更新:二重署名法(1/2)

予め親のDSのTTL時間を調べておく

① 新KSKを作成し、DNSKEYに登録して、DNSKEYを新KSKと旧KSKで署名する
- DNSKEYのTTL設定値の時間分待つ

② 親ゾーンのDS登録を旧から新に切り替える
- 親側のDSの切り替え作業を待ち、その後DSのTTL時間分待つ

③ 旧KSKを削除する

Copyright © 2011 株式会社日本レジストリサービス

KSKの更新:二重署名法(2/2)

	初期状態	①	②	③
親ゾーンのDS	旧DS	旧DS	**新DS**	**新DS**
自ゾーン DNSKEY	旧KSK ZSK	旧KSK **新KSK** ZSK	旧KSK **新KSK** ZSK	**新KSK** ZSK
自ゾーン DNSKEYの RRSIG	旧KSKでの署名 ZSKでの署名	旧KSKでの署名 **新KSKでの署名** ZSKでの署名	旧KSKでの署名 **新KSKでの署名** ZSKでの署名	**新KSKでの署名** ZSKでの署名

時間の流れ

Copyright © 2011 株式会社日本レジストリサービス

KSKの更新：事前公開法

予め親のDSのTTL時間を調べておく

① 新KSK(と新DS)を作成し親ゾーンに新旧2つのDSを登録する
  - 親ゾーンのDS登録を待つ、さらにDSのTTL時間分待つ

② 旧KSKを破棄し、新KSKでDNSKEYに署名する

③ 親ゾーンのDS登録を新DSのみにする

Copyright © 2011 株式会社日本レジストリサービス

KSKの更新:事前公開法

	初期状態	①	②	③
親ゾーンのDS	旧DS	旧DS **新DS**	旧DS **新DS**	**新DS**
自ゾーン DNSKEY	旧KSK ZSK	旧KSK ZSK	**新KSK** ZSK	**新KSK** ZSK
自ゾーン DNSKEYの RRSIG	旧KSKでの 署名 ZSKでの 署名	旧KSKでの 署名 ZSKでの 署名	**新KSKでの 署名** ZSKでの 署名	**新KSKでの 署名** ZSKでの 署名

Copyright © 2011 株式会社日本レジストリサービス

KSKの更新：メリット・デメリット

- 二重署名法
  - 親ゾーンとのDSのやり取りが1回で済む
  - ZSKと違い、署名がDNSKEYにのみ作用するので、ゾーンデータの肥大化や署名コストは問題にならない
- 事前公開法
  - 親ゾーンとDSのやり取りが2回必要となる

Copyright © 2011 株式会社日本レジストリサービス

鍵の更新間隔

- 運用面での鍵の更新間隔の実用的な値
  - KSK　13ヵ月　　12ヶ月で鍵更新
  - ZSK　～3ヵ月
- KSKの更新はDSの登録変更作業を伴うため、ドメイン名登録の更新にあわせるのが現実的
- ZSKはKSKのような制約は無く、自ゾーン内で処理が完結するため、運用面での負荷を考慮しながら期間を短めに設定する

Copyright © 2011 株式会社日本レジストリサービス

ゾーンの再署名

- 署名の有効期限が長すぎるのは望ましくない
  - 万が一の事態(鍵の盗難等)において速やかに対応するためには、署名期間は短いほうがよい
  - 有効期限が数分の署名も技術的には可能だが、休日等の対応を考慮すると現実性に欠ける
- 署名の有効期限に達する前に署名の有効期限を更新するために、**ゾーン全体の再署名**が必要となる
  - DNSSEC運用では、ゾーン情報に変更がなくても定期的再署名を行う必要がある

Copyright © 2011 株式会社日本レジストリサービス

NSEC3固有の問題

- NSEC3では、同じハッシュ値を使い続けると辞書攻撃により秘匿している情報が解析されるリスクがある
  - ゾーンの再署名時にソルトを変更し、ハッシュ値を変えるのが望ましい

Copyright © 2011 株式会社日本レジストリサービス

短すぎる署名期間の問題点

- 署名期間がRRのTTLより短い
  - キャッシュしたRRの署名が無効になる事態が発生する
    ⇒ 署名の有効期間はTTLより長い必要がある
- 署名期間がSOAのExpireより短い
  - セカンダリサーバーでゾーンが有効にも関わらず署名が無効になる事態が発生する可能性がある
    ⇒SOAのExpireは署名期間より短い必要がある

Copyright © 2011 株式会社日本レジストリサービス

鍵管理

- KSK秘密鍵が漏洩すると問題
  - KSKの更新は、DSの登録変更作業が必要なため、対処に時間がかかる
- ZSKはKSKに比べリスクは小さい
  - 万が一漏洩した場合でも、KSKに比べ短時間で更新できる
- いずれにしても鍵管理は十分厳重に行う

Copyright © 2011 株式会社日本レジストリサービス

BINDキャッシュサーバーでのDNSSECの設定

Copyright © 2011 株式会社日本レジストリサービス

古くて新しいDNSSEC

RFC	発行年	概要	対応BIND
2065	1997年	DNSSEC最初のRFC	
2535	1999年	RFC 2065の改良版	9.**2**系まで
3658	2003年	DS RRの登場	9.3系から
4033 4034 4035	2005年	現行のDNSSEC方式の基本 (**NSEC**方式)	9.3系から
5011	2007年	トラストアンカーの自動更新	9.**7**系から
5155	2008年	**NSEC3**方式(NSEC方式の改良)	9.**6**系から
5702	2009年	DNSKEY,RRSIGの**SHA-2**対応	9.**7**系から

Copyright © 2011 株式会社日本レジストリサービス

BINDキャッシュサーバーでのDNSSECの設定

- バージョンは9.7系以降の最新のものを使う
  - JPやrootゾーンではアルゴリズムに**RSASHA256**を採用している
- 通常のキャッシュサーバーの設定に、署名の検証を行う設定を追加する
  - named.conf の options 部分以下を追加する

    dnssec-enable yes;
    dnssec-validation yes;
  - 署名の検証に必要な情報を登録する
    ⇒ トラストアンカーの登録

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC対応のオプション

- dnssec-enable
  - DNSSEC対応にするかどうかのオプション
  - BIND 9.4以降のデフォルト　yes
- dnssec-validation
  - DNSSECの署名検証を行うかどうかのオプション
  - BIND 9.4のデフォルト　no
  - BIND 9.5以降のデフォルト　yes

```
options {
    ....
    dnssec-enable        yes;    // BIND 9.7 であれば
    dnssec-validatioin   yes;    // 設定しなくてもよい
    ....
};
```

Copyright © 2011 株式会社日本レジストリサービス

rootゾーンの公開鍵

- rootゾーンの公開鍵関連情報の入手先
  https://data.iana.org/root-anchors/
- rootゾーンの公開鍵
  https://data.iana.org/root-anchors/root-anchors.xml
  - ただし**DS情報のみ**
  - BINDで設定するトラストアンカーはKSK公開鍵
    ⇒ KSK公開鍵を入手する必要がある(後述)
  - DNSSEC Trust Anchor Publication for the Root Zone
    https://data.iana.org/root-anchors/draft-icann-dnssec-trust-anchor.html
- 他に、PGPの公開鍵、署名等がある

Copyright © 2011 株式会社日本レジストリサービス

root-anchors.xmlの内容

```
<?xml version="1.0" encoding="UTF-8"?>
<TrustAnchor
   id="AD42165F-3B1A-4778-8F42-D34A1D41FD93“
   source="http://data.iana.org/root-anchors/root-anchors.xml">
 <Zone>.</Zone>
 <KeyDigest id="Kjqmt7v“ validFrom="2010-07-15T00:00:00+00:00">
  <KeyTag>19036</KeyTag>
  <Algorithm>8</Algorithm>
  <DigestType>2</DigestType>
  <Digest>
   49AAC11D7B6F6446702E54A1607371607A1A41855200FD2CE1CDDE32F24E8FB5
  </Digest>
 </KeyDigest>
</TrustAnchor>
```

Copyright © 2011 株式会社日本レジストリサービス

rootのKSK公開鍵の入手

- rootゾーンのKSK公開鍵を入手

```
$ dig . dnskey | grep -w 257 > root-ksk.key
```

  - 257は現在有効なKSK (ZSKは256)

- KSK公開鍵からDSを生成

```
$ dnssec-dsfromkey -2 -f root-ksk.key .
. IN DS 19036 8 2 49AAC11D<中略>F24E8FB5
```

- root-anchors.xmlと比較し、差異の無いことを確認

注意：実際はPGP鍵で検証するのが望ましい

「DNSSECを利用するリゾルバーのための トラストアンカーの設定方法について 」
http://dnssec.jp/wp-content/uploads/2011/02/20110124-techwg-dnssec-trustanchor-install-howto-2.pdf

Copyright © 2011 株式会社日本レジストリサービス

BINDへトラストアンカーの登録

- named.conf にトラストアンカーを登録
  - root-ksk.keyから “<TTL> IN DNSKEY” を除いて trusted-keysに設定
- trusted-keysの書式
  - ドメイン名 数字 数字 数字 公開鍵
  - 公開鍵は “ “ で囲み、空白、TAB、改行等があってもよい
  - 複数ドメイン名の設定が可能

```
trusted-keys {
     “.“     257 3 8
             “AwEAAagAIKlVZrpC6Ia7gEzahOR+9W29euxhJhVVLOyQ
             bSEW0O8gcCjFFVQUTf6v58fLjwBd0YI0EzrAcQqBGCzh
             <中略>
             LmqrAmRLKBP1dfwhYB4N7knNnulqQxA+Uk1ihz0==“ ;
};
```

Copyright © 2011 株式会社日本レジストリサービス

キャッシュサーバーの動作確認

- named.conf の変更が終わったら、キャッシュサーバー用のnamedを再起動する
- digコマンドでDNSSEC対応ゾーンの確認
  - “.” の SOAが確実
  - 代表的なDNSSEC対応のドメイン名
    jprs.jp, www.iana.org, www.isc.org等
- +dnssec オプション
  - DNSSEC応答を受け取るためのオプション

Copyright © 2011 株式会社日本レジストリサービス

キャッシュサーバーへのdigの結果

```
$ dig +dnssec @127.0.0.1 jprs.jp a

; <<>> DiG 9.7.3 <<>> +dnssec @127.0.0.1 jprs.jp a
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1601
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 11

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;jprs.jp.                       IN      A

;; ANSWER SECTION:
jprs.jp.                54883   IN      A       202.11.16.167
jprs.jp.                54883   IN      RRSIG   A 8 2 86400 20110705035730 20110605035730 30789 jprs.jp. NwQZLPVhWGpO2JfGipy <略>

;; AUTHORITY SECTION:
jprs.jp.                54883   IN      NS      ns02.jprs.jp.
jprs.jp.                54883   IN      NS      ns01.jprs.jp.
jprs.jp.                54883   IN      NS      ns03.jprs.jp.
jprs.jp.                54883   IN      RRSIG   NS 8 2 86400 20110705035730 20110605035730 30789 jprs.jp. aye/3Z0hJEtCFTgd6w <略>
<以下略>
```

Copyright © 2011 株式会社日本レジストリサービス

digの結果のflagsフィールド

- flags: qr rd ra **ad**;
  - DNSにおけるさまざまな状態を表すフラグ
- DNSSECに関係するflag
  - ad: Authentic Data
    署名が検証できた正しいデータであることを示す
  - cd: Checking Disabled
    署名のチェックを行っていない状態を示す
- 署名の検証を行わない場合は+cdを指定

```
$ dig @127.0.0.1 +cd www.isc.org a
    flags: qr rd ra cd;
```

  - BINDはtrusted-keysを設定すると内部では**必ず署名の検証を行う**

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC検証の失敗

- トラストアンカーの設定を誤った場合

```
$ dig  +dnssec . soa
status: SERVFAIL
```

⇒ 答えが得られない

- 署名検証に**失敗した**場合、名前解決**不能**

⇒ DNSSEC最大の**リスク**

Copyright © 2011 株式会社日本レジストリサービス

トラストアンカーの自動更新

- rootのKSKが更新された場合、BIND等で設定しているトラストアンカーの更新が必要
  - 定期的な更新作業を要求される
- トラストアンカーの自動更新機能
  - RFC 5011対応 - BIND 9.7の新機能の一つ
- rootのKSK管理は、RFC 5011に準拠
  - RFC 5011対応の設定を行えば、トラストアンカーの更新作業を自動化できる

Copyright © 2011 株式会社日本レジストリサービス

RFC 5011対応の設定(1/2)

- managed-keysにトラストアンカーを設定
  - trusted-keysの替わりにmanaged-keysを使う
  - 別ドメイン名であれば併用も可能
- managed-keysの書式
  - ドメイン名 initial-key 数字 数字 数字 公開鍵
  - 「initial-key」を追加しあとはtrusted-keysと同じ
  - 複数ドメイン名の設定が可能

```
managed-keys {
     “.“     initial-key 257 3 8
             “AwEAAagAIKlVZrpC6Ia7gEzahOR+9W29euxhJhVVLOyQ
             bSEW0O8gcCjFFVQUTf6v58fLjwBd0YI0EzrAcQqBGCzh
             <中略>
             LmqrAmRLKBP1dfwhYB4N7knNnulqQxA+Uk1ihz0==“ ;
};
```

Copyright © 2011 株式会社日本レジストリサービス

RFC 5011対応の設定(2/2)

- managed-keysでは、2つのファイルがワーキングディレクトリに作成される
  managed-keys.bind　　KSKの履歴
  managed-keys.bind.jnl　上記のジャーナルファイル
  – ディレクトリはmanaged-keys-directoryで変更可
- ディレクトリのパーミッションを、namedの実行権限で読み書きできるように設定する

```
$ ps auxc | fgrep named
bind      932  0.0  0.8 48576 31844 ?? Ss 29Jul10 2:14.61 named
$ ls -ld /var/run/named
drwxr-xr-x 2 bind wheel 512 Jul 30 12:24 /var/run/named
```

- 運用事例が少ないため、RFC 5011の**過信は禁物**

Copyright © 2011 株式会社日本レジストリサービス

鍵生成と署名作業

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC鍵の作成: dnssec-keygen

- -a 鍵生成アルゴリズムの指定
  - RSASHA1、NSEC3RSASHA1、RSASHA256、RSASHA512などを指定する
- -b ビット長
  - ZSK: 上記アルゴリズムの場合 1024ビット以上
  - KSK: 上記アルゴリズムの場合 2048ビット以上
- -f ksk
  - KSKを作成する場合に指定
- 最後に名前(ゾーン名)を指定

Copyright © 2011 株式会社日本レジストリサービス

KSKとZSKの生成

- ZSKの生成

```
# dnssec-keygen -a RSASHA256 -b 1024
                example.jp > zsk-example.jp
```

  - 鍵のファイル名を表示するので、その結果を保存する
    Kexample.jp.+008+52863

  3桁の数字はアルゴリズム、5桁は識別子(ID)

- 1組の鍵ファイルができる

  Kexample.jp.+008+52863.key　　⇒公開鍵
  Kexample.jp.+008+52863.private　　⇒秘密鍵

- KSKの生成

```
# dnssec-keygen -a RSASHA256 -b 2048 -f ksk
                example.jp > ksk-example.jp
```

Copyright © 2011 株式会社日本レジストリサービス

鍵ファイルの中身の例(秘密鍵)

```
Private-key-format: v1.3
Algorithm: 8 (RSASHA256)
Modulus:
    yRDBKqRGEZUj6gAB2rd4SzNfcHb9sQMU9rq/BHhTs7FHxt90ey7cG2rLrh28l4AY4tFWGSKBN4bPQhVWXvT69zqu4jBuy7Gg7j2Rs
    +Eb+WYGUAyUWQMQ9MvPGOTD0tImE4m5kjUTnoziS/GFDSMj8GXs4rHm1rUWyClf9lv2Ru0=
PublicExponent: AQAB
PrivateExponent:
    rhnf6biNI7RsgLa45FZxx0wYnB2s1pXAlVRnCsvWToZ3jHD5P6D33pW/AGmnX9f/tIdnciQ6l4YX+TTYsSiYFemvG4TRZcx91rY5J
    VVxBL91kHPqTiNwdrdM5mcKE3835SFK/XEInPYHErAGi2Gz3bQGlk2dPGjQG6ai5ua1x9E=
Prime1: 9Gy+ms2q0EYpHVb4drHR25l4HzC2xpdgH77/0pwtftZSg7NZixb2YI9ULRy38y+57YtZ2AZ4s51QzgGHPt2xUw==
Prime2: 0pZZH5hPkkjNFVTBKQc0nve3Pd/FnS7GmlOhq2NhXQEYexRkYt0R2VYfV+GYgszuooOXqjRWi0eI1G/uMfPevw==
Exponent1: lxLboJz8Ne0Xnn3R5rMzzbKGz2hxoD+R9y07u7YyXJIlwCdLci/IKpiMY7G7dMEL/2nBJ0egtQvIFPxW1qF55w==
Exponent2: CxRN7BOfXBrob07eOsJeSl7ODTtQskxbtpLf1pyL6tC78P3JqknnPoABdiYwV/FgPLyfphzK0NkaodKhvY8PEQ==
Coefficient: 8q2G3F5fagdIvejnm6Jt7kXD5EYI3LXOVGwDYgZOLH6vPF/Eh5952Q9ivSr7qNqyjWzqMEP1fpET4uhRLizy5Q==
Created: 20101124065848
Publish: 20101124065848
Activate: 20101124065848
```

- BIND 9.7系のdnssec-keygenで作成
  - 9.6系まではformatがv1.2で、v1.2はBIND 9.7系のツールでも扱えるが、v1.3は9.7系(以降)のツールのみ対応
  - RSASHA256やRSASHA512はBIND 9.6.2以降で対応

Copyright © 2011 株式会社日本レジストリサービス

鍵ファイルの中身の例(公開鍵)

```
; This is a zone-signing key, keyid 52863, for example.jp.
; Created: 20101124065848 (Wed Nov 24 15:58:48 2010)
; Publish: 20101124065848 (Wed Nov 24 15:58:48 2010)
; Activate: 20101124065848 (Wed Nov 24 15:58:48 2010)
example.jp. IN DNSKEY 256 3 8
   AwEAAckQwSqkRhGVI+oAAdq3eEszX3B2/bEDFPa6vwR4U7OxR8bfdHsu
   3Btqy64dvJeAGOLRVhkigTeGz0IVVl70+vc6ruIwbsuxoO49kbPhG/lm
   BlAMlFkDEPTLzxjkw9LSJhOJuZI1E56M4kvxhQ0jI/Bl7OKx5ta1Fsgp
   X/Zb9kbt
```

- BIND 9.7系のdnssec-keygenで作成
  - –“;”のコメント部については後述

Copyright © 2011 株式会社日本レジストリサービス

dnssec-keygenの注意点

- KSKとZSKの区別に注意する
  - 2組の鍵ファイル(計4個)ができ、
    見た目での識別は困難
- 実行時に鍵ファイル名を保存すると良い

```
# dnssec-keygen -f ksk .. > ksk-...
# dnssec-keygen ......... > zsk-...
```

Copyright © 2011 株式会社日本レジストリサービス

ゾーンへの署名：dnssec-signzone

- 署名対象ゾーンファイル、ZSK、KSKを準備
  - 同じディレクトリに用意し、ゾーンファイルは
    ゾーン名とファイル名を一致させると便利

example.jp

Kexample.jp.+008+56449.key	KSK公開鍵
Kexample.jp.+008+56449.private	KSK秘密鍵
Kexample.jp.+008+52863.key	ZSK公開鍵
Kexample.jp.+008+52863.private	ZSK秘密鍵

Copyright © 2011 株式会社日本レジストリサービス

ゾーンへの署名(続き)

- ゾーンファイルにKSK、ZSKの公開鍵を登録
  - 公開鍵をまとめたファイルを用意し、$INCLUDE文を利用してゾーンファイルから参照する

```
# cat `cat ksk-example.jp`.key
      `cat zsk-example.jp`.key > example.jp.keys
```

```
;ゾーンファイル中でkeyファイルを参照
$INCLUDE example.jp.keys
```

- SOAシリアル値の管理は、dnssec-signzoneの -N オプションにまかせるのがベター

Copyright © 2011 株式会社日本レジストリサービス

署名前のゾーンファイル example.jp

```
$TTL    1D
$INCLUDE example.jp.keys
@       IN      SOA     ns root (
                        1       ; Serial
                        10800   ; Refresh
                        3600    ; Retry
                        3600000 ; Expire
                        1800 )  ; Minimum TTL
                NS      ns
                MX      10 mail
;
ns              A       192.0.2.17
www             A       192.0.2.18
mail            A       192.0.2.19

sub1            NS      ns.sub1
ns.sub1         A       192.0.2.49

sec3            NS      ns.sec3
ns.sec3         A       192.0.2.65
$INCLUDE ../sec3.example.jp/dsset-sec3.example.jp.

sub3            NS      ns.sub3
ns.sub3         A       192.0.2.81
```

Copyright © 2011 株式会社日本レジストリサービス

署名の実行

- dnssec-signzone -H <繰り返し回数> -3 <salt>
  -N <SOAのシリアル値> -k <KSK>
  <ゾーンファイル> <ZSK>

```
# dnssec-signzone –H 3 -3 123ABC –N unixtime
    –k `cat ksk-example.jp`
    example.jp `cat zsk-example.jp`
```

  - -3はNSEC3方式を選びソルトを指定するオプション
  - 秘密鍵を明示的に指定する必要は無い
- 出力ファイル
  - example.jp.signed　署名済みのゾーン
  - dsset-example.jp.　ゾーンへのDS RR

Copyright © 2011 株式会社日本レジストリサービス

署名済みのゾーンファイル(抜粋)

```
; File written on Wed Nov 24 16:07:14 2010
; dnssec_signzone version 9.7.2-P2
example.jp.             86400   IN SOA  ns.example.jp. root.example.jp. (
                                        1290582434 ; serial
                                        <中略>
                                        )
                        86400   RRSIG   SOA 8 2 86400 20101224060714 (
                                        20101124060714 52863 example.jp.
                                        WcEe7OoXanQAPuS8pTwYJ1wLRFkC75/2kwii
                                        <中略>
                                        8NmbrA3wlhoCqwEBAeQX+ZhKZtQ= )
                        86400   NS      ns.example.jp.
                        86400   RRSIG   NS 8 2 86400 20101224060714 (
                                        20101124060714 52863 example.jp.
                                        bLcTvbBVftz6NBTSzvj8Yn0G0PyMbTYfzEvd
                                        <中略>
                                        Q2kGOoVtpxJoG69hOod036w+Yj8= )
                        86400   MX      10 mail.example.jp.
                        86400   RRSIG   MX 8 2 86400 20101224060714 (
                                        20101124060714 52863 example.jp.
                                        OiKiNz7CGAnBXdgfaUm23+/VOGaLfgMk6RYB
                                        <中略>
                                        0D3RCXYCUoGtKdCswIM77w0U2GE= )
                        86400   DNSKEY  256 3 8 (
                                        AwEAAckQwSqkRhGVI+oAAdq3eEszX3B2/bED
                                        <中略>
                                        4kvxhQ0jI/Bl7OKx5ta1FsgpX/Zb9kbt
                                        ) ; key id = 52863
                        86400   DNSKEY  257 3 8 (
                                        AwEAAauHCuMQzCBUaaQLNf/FPRGqcPupOdYU
                                        <中略>
                                        WnegM3YXJyvpSS0gZ9ykoo1Reqa/94XL9HCl
                                        vIZUhkdCzjyo/0EKdgBt5IU=
                                        ) ; key id = 56449
```

Copyright © 2011 株式会社日本レジストリサービス

DSの登録

- dsset-example.jpの内容を親ドメインに登録
  - ゾーンの署名時に生成されたもの
  - 内容の例

```
example.jp. IN DS 56449 8 1 6EFE<中略>4BE
example.jp. IN DS 56449 8 2 7D29<中略>F32852 DE98ED8C
```

- DSレコードはKSKの公開鍵からも生成可能
  - dnssec-dsfromkeyコマンドを使用する
- **登録は権威サーバーの設定完了後に行う**

Copyright © 2011 株式会社日本レジストリサービス

BIND権威サーバーでの DNSSECの設定

Copyright © 2011 株式会社日本レジストリサービス

BINDの設定：権威サーバー(1/2)

- DNSSECを有効にする
  - named.conf の options 部分に dnssec-enable yes; を追加

```
options {
        <省略>
        dnssec-enable yes; // BIND 9.4以降は
                           // デフォルトが yes;
        <省略>
};
```

Copyright © 2011 株式会社日本レジストリサービス

BINDの設定：権威サーバー(2/2)

- ゾーンファイルを署名済みのものに変更

```
zone "example.jp" {
        type master ;
        // file "example.jp.zone" ;
        file "example.jp.signed" ;
} ;
```

- namedを再起動

```
rndc reload
```

Copyright © 2011 株式会社日本レジストリサービス

権威サーバーの動作確認

- digコマンドでDNSSEC対応ゾーンの確認

```
# dig +dnssec +norec @127.0.0.1 www.example.jp a
```

+dnssec DNSSECを有効にする問合せ

このオプションなしでは通常の(DNSSECでない)ものと同じ結果が返る

+norec　非再帰問合せ

キャッシュサーバーから権威サーバーへの問合せと同じ形式の問合せ

Copyright © 2011 株式会社日本レジストリサービス

権威サーバーへのdigの結果(1/2)

```
$ dig @127.0.0.1 +norec www.example.jp a

; <<>> DiG 9.7.2-P2 <<>> @127.0.0.1 +norec www.example.jp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59143
;; flags: qr aa ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;www.example.jp.                        IN      A

;; ANSWER SECTION:
www.example.jp.         86400   IN      A       192.0.2.18

;; AUTHORITY SECTION:
example.jp.             86400   IN      NS      ns.example.jp.

;; ADDITIONAL SECTION:
ns.example.jp.          86400   IN      A       192.0.2.17

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed Nov 24 16:23:02 2010
;; MSG SIZE  rcvd: 81
```

+dnssec無しの
digの応答

Copyright © 2011 株式会社日本レジストリサービス

権威サーバーへのdigの結果(2/2)

```
$ dig @127.0.0.1 +dnssec +norec www.example.jp

; <<>> DiG 9.7.2-P2 <<>> @127.0.0.1 +dnssec +norec www.example.jp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21018
;; flags: qr aa ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.example.jp.                        IN      A

;; ANSWER SECTION:
www.example.jp.         86400   IN      A       192.0.2.18
www.example.jp.         86400   IN      RRSIG   A 8 3 86400 20101224060714 20101124060714 52863 example.jp.
      r8cI/2zMc5fCtlie6pjZoaI0Oo0myBWzir3ykFpDpkMGWxaF1pHcCCzK ogZXfDvW1PMKP3Hc+TshZThW+FuCY1EnCsPzl5n1Rvl39tsP83ZHFGZ6
      PKH7FsxGZUFtwa0cyILkcKRF7BPvrITCk+y0oWivzDr1LHGR+5F6hx1p 4ac=

;; AUTHORITY SECTION:
example.jp.             86400   IN      NS      ns.example.jp.
example.jp.             86400   IN      RRSIG   NS 8 2 86400 20101224060714 20101124060714 52863 example.jp.
      bLcTvbBVftz6NBTSzvj8Yn0G0PyMbTYfzEvdwpL0WYmd6zbDiX3ZGXSv EymEaWi8CDzmnbQqZ5VM5dCAe5IaddZqHgAdeJJ2MRZCu2OxDdCjwEne
      pNnFnu0TXCVyP6Dipq61mcc+2lZHLakzQ2kGOoVtpxJoG69hOod036w+ Yj8=

;; ADDITIONAL SECTION:
ns.example.jp.          86400   IN      A       192.0.2.17
ns.example.jp.          86400   IN      RRSIG   A 8 3 86400 20101224060714 20101124060714 52863 example.jp.
      vGOD3t5bklnTeoZwDeXjKcZAeXFblB+qfmZzz3P7+JUFA7/1NjQGvWwO dqtGlBznFuTl+7lediOJnf9zDaJjJI7dobv10Nb3Wy/1QmZHw3hAmbcm
      64DgbN/004j1HUORp30UgB59/Esb8HFARQQcskRAxa7iq1gdTm5dH5oa PB0=

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed Nov 24 16:28:36 2010
;; MSG SIZE  rcvd: 602
```

+dnssecありでは、RRSIG RRを加えたものが返る

Copyright © 2011 株式会社日本レジストリサービス

DO(DNSSEC OK)ビット

- dig の +dnssec オプション
  - 問合せでEDNS0を使い、DOビットをONにすると共に512バイトを超えるサイズのDNSパケットを受けられることを宣言する
  - DNSSECでは**EDNS0のサポートは必須**
- DOビット
  - DNSSEC OK ⇒ DNSSECの応答を受ける ⇒ DNSSECを要求する
  - 権威サーバーは、問合せのDOビットがONであれば、DNSSECの情報を含んだ応答を返す

Copyright © 2011 株式会社日本レジストリサービス

スマート署名(Smart signing)
BIND 9.7での新機能

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC for Humans

- BIND 9.7から導入された、DNSSECの設定をより簡単に行う一連の機能
  - スマート署名(Smart signing)
  - 全自動ゾーン署名
  - RFC 5011への対応
  - Dynamic Update設定の簡素化
  - DLVの自動設定
- スマート署名
  - KSKやZSKの鍵管理の自動化

Copyright © 2011 株式会社日本レジストリサービス

スマート署名
example.jpをNSEC3方式で署名

```
# ls
example.jp
# dnssec-keygen -3 example.jp
Generating key pair....++++++ ..................++++++
Kexample.jp.+007+31760
# dnssec-keygen -3 -f ksk example.jp
Generating key pair.....................+++ ........+++
Kexample.jp.+007+22740
# dnssec-signzone -3 aabbcc -S example.jp
Fetching ZSK 31760/NSEC3RSASHA1 from key repository.
Fetching KSK 22740/NSEC3RSASHA1 from key repository.
Verifying the zone using the following algorithms: NSEC3RSASHA1.
Zone signing complete:
Algorithm: NSEC3RSASHA1: KSKs: 1 active, 0 stand-by, 0 revoked
                         ZSKs: 1 active, 0 stand-by, 0 revoked
example.jp.signed
# ls
Kexample.jp.+007+22740.key      dsset-example.jp.
Kexample.jp.+007+22740.private  example.jp
Kexample.jp.+007+31760.key      example.jp.signed
Kexample.jp.+007+31760.private
```

Copyright © 2011 株式会社日本レジストリサービス

スマート署名

- dnssec-signzoneの-Sオプション
  - 鍵情報を自動的に取り込むため、ゾーンファイルは通常のゾーンファイル(非DNSSEC)のままでよい
- dnssec-keygenのデフォルト値
  - アルゴリズム　RSASHA1
    「-3 」を指定した場合　NSEC3RSASHA1
  - 鍵のbit長:　ZSK: 1024　KSK: 2048
- -K　鍵を保存するディレクトリ(Key repository)を指定するオプション
  - 他の鍵関連コマンドでも指定できる

注意: アルゴリズムを指定する場合、鍵のbit長も指定する

例) RSASHA256での鍵の生成

- dnssec-keygen -a RSASHA256 -b 1024 example.jp
- dnssec-keygen -a RSASHA256 -b 2048 - f ksk example.jp

Copyright © 2011 株式会社日本レジストリサービス

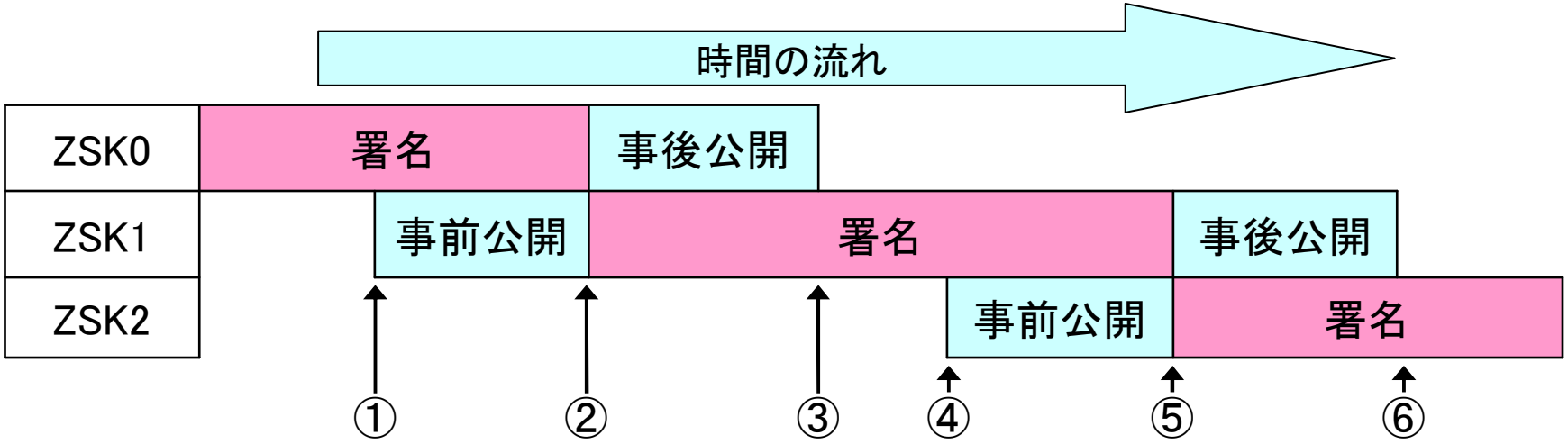
日付情報を指定するオプション

- dnssec-keygenで鍵生成時に指定
  - -P　ゾーンへの出力時刻(Publicatijon date)
  - -R　鍵の破棄時刻(Revocation date)
  - -A　署名鍵としての使用開始時刻(Activation date)
  - -I　署名鍵使用終了時刻(Inactivation date)
  - -D　ゾーンからの削除時刻(Deletion date)
  - -Pと-Aのデフォルトは now (現在時刻)
  - 相対時刻又は絶対時刻で指定する
    絶対時刻: YYYYMMDD 又は YYYYMMDDHHMMSS
    相対時刻: +数字 又は -数字　　'y', 'mo', 'w', 'd', 'h', or 'mi'
    (年、月、週、日、時間、分)
    を指定可能
- dnssec-settime - 日付情報を変更するコマンド

注意: 各時刻は**dnssec-signzoneの実行時**に評価される

Copyright © 2011 株式会社日本レジストリサービス

事前公開法でのそれぞれのタイミング



ZSK0: ②署名鍵使用終了 ③ゾーンから削除
ZSK1: ①ゾーンへ出力 ②署名鍵使用開始
⑤署名鍵使用終了 ⑥ゾーンから削除
ZSK2: ④ゾーンへ出力 ⑤署名鍵使用開始

	①	②	③	④	⑤	⑥
ZSK0		−I	−D			
ZSK1	−P	−A			−I	−D
ZSK2				−P	−A	

Copyright © 2011 株式会社日本レジストリサービス

日付を指定して鍵を作成(1/2)

```
# mkdir keys                                                    ①
# dnssec-keygen -K keys -f ksk example.jp                       ②
Kexample.jp.+005+45154
# dnssec-keygen -K keys -P now -A now –I +30d -D +31d example.jp  ③
Kexample.jp.+005+20076
# dnssec-keygen -K keys -P now -A +30d –I +60d -D +61d example.jp  ④
Kexample.jp.+005+45870
# dnssec-signzone -K keys –N unixtime -S example.jp             ⑤
Fetching KSK 45154/RSASHA1 from key repository.
Fetching ZSK 20076/RSASHA1 from key repository.
Fetching ZSK 45870/RSASHA1 from key repository.
Verifying the zone using the following algorithms: RSASHA1.
Zone signing complete:
Algorithm: RSASHA1: KSKs: 1 active, 0 stand-by, 0 revoked
                    ZSKs: 1 active, 1 stand-by, 0 revoked
example.jp.signed
# ls -F                                                         ⑥
dsset-example.jp.       example.jp.signed
example.jp              keys/
```

Copyright © 2011 株式会社日本レジストリサービス

日付を指定して鍵を作成(2/2)

① 鍵用のディレクトリ(keys)を作成
② KSKを作成
③ 最初に使うZSKを作成
　すぐに署名を開始し、30日後に署名を停止、31日後にゾーンから削除
④ 2番目に使うZSKを作成
　すぐにゾーンに出力、30日後に署名を開始、60日後に署名を停止、61日後にゾーンから削除
⑤ ゾーンへの署名
　KSKは1個、ZSKは1個が署名用、1個が事前公開用
　この例ではNSEC方式を採用し、SOAのシリアルはunixtimeを使っている
⑥ 鍵はkeysディレクトリ内にある

Copyright © 2011 株式会社日本レジストリサービス

署名と鍵更新の自動化

- cron等を利用して定期的に実行
  - dnssec-keygen で -P, -A, -I, -D を適正に設定したZSKを作成
  - dnssec-signzone -S で再署名しゾーンをリロード
    ⇒ 鍵更新、再署名の自動化が可能になる
- KSKについても同様の処理が可能
  - 但しDSの更新は、親ゾーンとのやり取りが必要なため、完全な自動化は難しい

注意:
dnssec-keygenで-Aを指定する場合、必ず-Pも指定する
⇒ BIND 9.7.3時点での不具合

Copyright © 2011 株式会社日本レジストリサービス

週を単位とした自動化設定例

- スケジュールの設定
  - 月曜日 9:00 に鍵を作成し、すぐに事前公開を行う
  - 水曜日に署名鍵を切り替え、古い鍵は金曜日に削除
  - 毎日10:00に再署名を行う
- 月曜日朝9:00の鍵生成

```
dnssec-keygen -K /var/named/keys -P now -A +2d
    -I +9d -D +11d -a RSASHA256 -b 1024 example.jp
```

- 毎朝10:00の再署名とゾーンのリロード

```
dnssec-signzone -K /var/named/keys -S -3 <ソルト>
             -N unixtime -H <繰り返し回数> example.jp
rndc reload example.jp
```

注意：ソルト・繰り返し回数は別途生成する

Copyright © 2011 株式会社日本レジストリサービス

運用面から見たスマート署名

- ゾーンファイルの変更履歴はとりやすい
- 署名完了後のゾーンファイルをnamedに読み込ませるため、ある程度確実な運用ができる
- dnssec-signzoneには差分署名の機能が無いため、大きなゾーンファイルに対しては、ゾーン変更時の署名の負荷が大きい
  - ゾーン数が多い場合やレコード数が多い場合は**OpenDNSSEC**が有利

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC化による DNSデータの変化

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC有無による jprs.jpの検索

- DNSSEC無し

```
$ dig +norec @a.dns.jp jprs.jp a | grep SIZE
;; MSG SIZE  rcvd: 186      (親のNSに問合せ)
$ dig +norec @ns01.jprs.jp jprs.jp a | grep SIZE
;; MSG SIZE  rcvd: 202      (自分自身のNSに問合せ)
```

- DNSSEC有り

```
$ dig +norec +dnssec @a.dns.jp jprs.jp a | grep
  SIZE
;; MSG SIZE  rcvd: 443      (親のNSに問合せ)
$ dig +norec +dnssec @ns01.jprs.jp jprs.jp a |
  grep SIZE
;; MSG SIZE  rcvd: 1382     (自分自身のNSに問合せ)
```

Copyright © 2011 株式会社日本レジストリサービス

権威サーバーへのdigの結果(1/2)

- +dnssec無しのdigの応答

```
$ dig +norec  @ns01.jprs.jp jprs.jp a

; <<>> DiG 9.7.3 <<>> +norec @ns01.jprs.jp jprs.jp a
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 35397
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 5

;; QUESTION SECTION:
;jprs.jp.                       IN      A

;; ANSWER SECTION:
jprs.jp.                86400   IN      A       202.11.16.167

;; AUTHORITY SECTION:
jprs.jp.                86400   IN      NS      ns02.jprs.jp.
jprs.jp.                86400   IN      NS      ns03.jprs.jp.
jprs.jp.                86400   IN      NS      ns01.jprs.jp.

;; ADDITIONAL SECTION:
ns01.jprs.jp.           86400   IN      A       202.11.17.107
ns01.jprs.jp.           86400   IN      AAAA    2001:df0:8:6::10
ns02.jprs.jp.           86400   IN      A       202.11.16.227
ns02.jprs.jp.           86400   IN      AAAA    2001:df0:8:20::10
ns03.jprs.jp.           86400   IN      A       61.200.83.204

;; Query time: 4 msec
;; SERVER: 2001:df0:8:6::10#53(2001:df0:8:6::10)
;; WHEN: Wed Jun  8 19:44:51 2011
;; MSG SIZE  rcvd: 202
```

Copyright © 2011 株式会社日本レジストリサービス

権威サーバーへのdigの結果(2/2)

- +dnssec有り 各RRにRRSIG RRを加えたものが返る

```
; <<>> DiG 9.7.3 <<>> +norec +dnssec @ns01.jprs.jp jprs.jp a
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 55599
;; flags: qr aa; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 11

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;jprs.jp.                       IN      A

;; ANSWER SECTION:
jprs.jp.                86400   IN      A       202.11.16.167
jprs.jp.                86400   IN      RRSIG   A 8 2 86400 20110705035730 20110605035730 30789 jprs.jp. NwQZLPVhW

;; AUTHORITY SECTION:
jprs.jp.                86400   IN      NS      ns03.jprs.jp.
jprs.jp.                86400   IN      NS      ns02.jprs.jp.
jprs.jp.                86400   IN      NS      ns01.jprs.jp.
jprs.jp.                86400   IN      RRSIG   NS 8 2 86400 20110705035730 20110605035730 30789 jprs.jp. aye/3Z0h

;; ADDITIONAL SECTION:
ns01.jprs.jp.           86400   IN      A       202.11.17.107
ns01.jprs.jp.           86400   IN      AAAA    2001:df0:8:6::10
ns02.jprs.jp.           86400   IN      A       202.11.16.227
ns02.jprs.jp.           86400   IN      AAAA    2001:df0:8:20::10
ns03.jprs.jp.           86400   IN      A       61.200.83.204
ns01.jprs.jp.           86400   IN      RRSIG   A 8 3 86400 20110705035730 20110605035730 30789 jprs.jp. upG+W1QiO
ns01.jprs.jp.           86400   IN      RRSIG   AAAA 8 3 86400 20110705035730 20110605035730 30789 jprs.jp. O2aPPk
ns02.jprs.jp.           86400   IN      RRSIG   A 8 3 86400 20110705035730 20110605035730 30789 jprs.jp. bZ/jdOKkL
ns02.jprs.jp.           86400   IN      RRSIG   AAAA 8 3 86400 20110705035730 20110605035730 30789 jprs.jp. sahymK
ns03.jprs.jp.           86400   IN      RRSIG   A 8 3 86400 20110705035730 20110605035730 30789 jprs.jp. tD5/K1HQA


;; Query time: 7 msec
;; SERVER: 2001:df0:8:6::10#53(2001:df0:8:6::10)
;; WHEN: Wed Jun  8 19:45:46 2011
;; MSG SIZE  rcvd: 1382
```

注意：RRSIGは行の途中まで、残り省略

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC有無による 存在しないドメイン名の検索

- example.jp +dnssec無し

```
$ dig +norec @a.dns.jp example.jp a | grep SIZE
;; MSG SIZE  rcvd: 75
```

- example.jp +dnssec有り

```
$ dig +norec +dnssec @a.dns.jp example.jp a | grep
  SIZE
;; MSG SIZE  rcvd: 987
```

注意: example.jpは存在しないドメイン名

Copyright © 2011 株式会社日本レジストリサービス

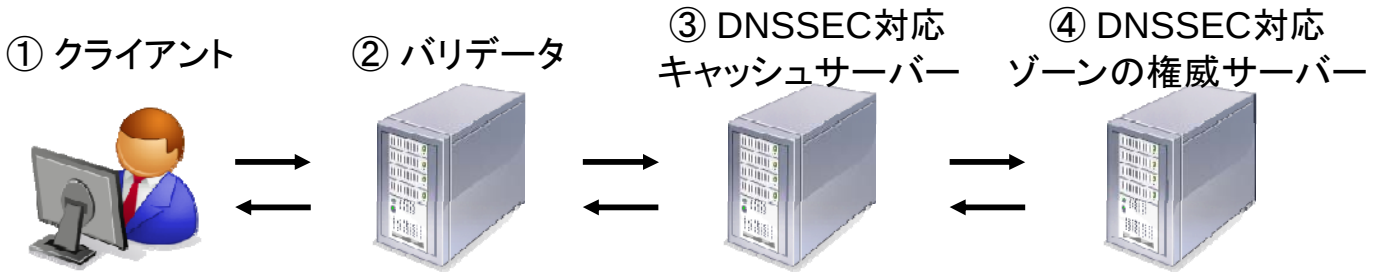
DNSSEC対応になると

- 署名の付加によりゾーンデータが大きくなる
  - 5～10倍程度(鍵のbit長に依存)
  - プライマリが署名すると、セカンダリにもインパクトがある
- DNS応答パケットのサイズが大きくなる
  - DNSトラフィックが増える
  - キャッシュサーバーのキャッシュ効率が落ちる
  - 特に存在しない名前の検索では顕著
- DNSSEC対応のキャッシュサーバーの実装では、**DNSSEC設定を行わなくてもDOビットはON**となる

Copyright © 2011 株式会社日本レジストリサービス

何故DOビットは常時ONなのか

- キャッシュサーバー自身では署名検証を行わなくても、他の署名検証を行うもの(**バリデータ**)のために、署名があれば対象レコードと同時にキャッシュ



DNSSECにおいて署名検証が独立したモデル

②が署名検証を行うキャッシュサーバーで③をフォワード先として指定している場合や、②は存在せず①のクライアントが直接署名検証を行う場合等

Copyright © 2011 株式会社日本レジストリサービス

DOビットが常時ONのインパクト

- 問合せ1回あたりの応答パケットが大きくなる
  - DNSのトラフィックが増加する
- キャッシュサーバーではキャッシュに必要なメモリ量が増える
  - 場合によっては、キャッシュできるレコード数が減り、キャッシュ効率に影響する
- 手元のDNSSEC設定とは関係なく、DNSSECの普及度によって影響する

Copyright © 2011 株式会社日本レジストリサービス

DNSSECのリスク

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC化による負荷の増加(1)

- 権威DNSサーバー側
  - 署名を作成するための負荷
    ⇒ 但し、DNSサーバーと別サーバーでの処理が可能
  - 署名が負荷によるDNSデータを保持するためのメモリ量の増加
  - 応答にRRSIGを付加するための処理
- キャッシュDNSサーバー側
  - 署名検証の負荷
    但し署名検証は、キャッシュ時に行われ、キャッシュ済みであれば、改めて署名検証することは無い
  - 署名データもキャッシュするためメモリ使用量の増加

Copyright © 2011 株式会社日本レジストリサービス

DNSSEC化による負荷の増加(2)

- キャッシュ・権威DNSサーバー共通の負荷
  - NSEC3方式の場合、ハッシュ計算コスト

- DNSSEC対応によって
  - 同じサーバーであれば処理能力は3～5割程度減少する ⇒ 条件によって大きく変化する
  - メモリ使用量は5～10倍程度増加
    ⇒ 署名検証する・しないとは独立に起きる

Copyright © 2011 株式会社日本レジストリサービス

DNSSECの署名検証の失敗

- 署名検証に失敗した場合、名前解決不能
  - DNSSECは嘘を識別する技術
    ⇒ 正しいものを探し出す技術ではない
- 署名検証が失敗する主な要因
  - 鍵を取り違えた
  - 上位に登録するDSを誤った
  - 署名開始前の鍵を作った時点で上位にDSを登録した
  - 署名の有効期間を過ぎた
  - etc...

Copyright © 2011 株式会社日本レジストリサービス

サーバー時刻の同期の必要性

- 署名には有効期間があり有効期間外は無効
  - 有効期間の開始時刻、終了時刻は絶対時刻
  - 署名が正しくてもサーバーの時刻が極端に違うと、署名検証に失敗する
- DNSSEC運用を行う場合、サーバーの時刻を正しく合わせる必要がある
  - NTPなどを利用するのが確実
  - 実用上は、分程度まで合っていれば問題ない

Copyright © 2011 株式会社日本レジストリサービス

DNSSECのまとめ

Copyright © 2011 株式会社日本レジストリサービス

従来(DNSSEC無し)との比較(1)

- ゾーン管理(権威サーバー)側
  - ZSKとKSKを作成し管理する必要がある
  - ゾーンに署名を行う必要がある
  - 定期的に鍵の更新を行う必要がある
  - 子ゾーンでは親ゾーンにDSの登録作業を行う必要がある(KSKを変更する度に必要)
- 鍵管理の手間と、ゾーン署名のコストが増えるが、ある程度は自動化可能

Copyright © 2011 株式会社日本レジストリサービス

従来(DNSSEC無し)との比較(2)

- キャッシュサーバー側
  - DNSSEC機能を有効にし、トラストアンカーを設定する
  - 必要に応じてトラストアンカーを更新する
  - 署名検証による負荷の増大の懸念がある
- 双方でサーバーの時刻を正しく設定する

Copyright © 2011 株式会社日本レジストリサービス

DNSSECまとめ

- DNSSECは、公開鍵暗号技術を利用した署名によるDNSデータ保護のしくみ
  - KSKとZSKの2つの鍵を使う
  - 親ゾーンにはNSに加えてDSを登録する
  - rootゾーンのKSKの公開鍵を使って署名を検証
  - 定期的な鍵の更新と再署名とが必要

Copyright © 2011 株式会社日本レジストリサービス

御清聴ありがとうございました

Copyright © 2011 株式会社日本レジストリサービス

ご利用にあたって

著作権について

当コンテンツの著作権は、株式会社日本レジストリサービス（JPRS）に帰属しています。

ご利用にあたっては、次の条件によるものとします。

- 再配布は、原則として自由です。
  ただし、次の事項をお守りください。
  (1) ご利用にあたっては、内容の改変およびコンテンツの著作権表示の改変または削除は行わず、必ず出所を明示してください。
  (2) JPRSの利益に反するご利用は行わないで下さい。
- 当コンテンツは、営利目的でのご利用はできません。
- 引用・転載・複製した著作物を発行する場合には、JPRSへ一部お送りいただけますようお願いいたします。

ご注意

- 掲載内容は2011年7月時点のものです。
- DNSSECの導入に当たっては、「JPドメイン名におけるDNSSEC運用ステートメント（JP DPS）」もあわせてご覧ください。
  ■JPドメイン名におけるDNSSEC運用ステートメント（JP DPS）
  <https://jprs.jp/doc/dnssec/jp-dps-jpn.html>
- 当コンテンツは以下のURLにて公開しています。
  ■ドメイン名やDNSの解説コラム
  <http://jpinfo.jp/topics-column/>

各種商標または登録商標について

当コンテンツに記載されている会社名、サービス名、商品名は各社の商標または登録商標です。

当コンテンツの文中や図版において、TMや(R)などの記号を使用していない場合があります。

お問合せについて

当コンテンツの技術的内容に関するお問い合わせ・サポートは受け付けておりません。

その他のお問合せにつきましては、info@jprs.jpまでお問合せください。

Copyright © 2011 株式会社日本レジストリサービス