

DNSSECの基礎概要

2012年11月21日

Internet Week 2012 DNSSECチュートリアル

株式会社日本レジストリサービス (JPRS)

船戸正和

本チュートリアルの内容

- DNSSECの導入状況
- DNSキャッシュへの毒入れと対策
- DNSSECのしくみ
- 鍵と信頼の連鎖
- DNSSECのリソースレコード(RR)
- 対応が必要な関係者
- まとめ

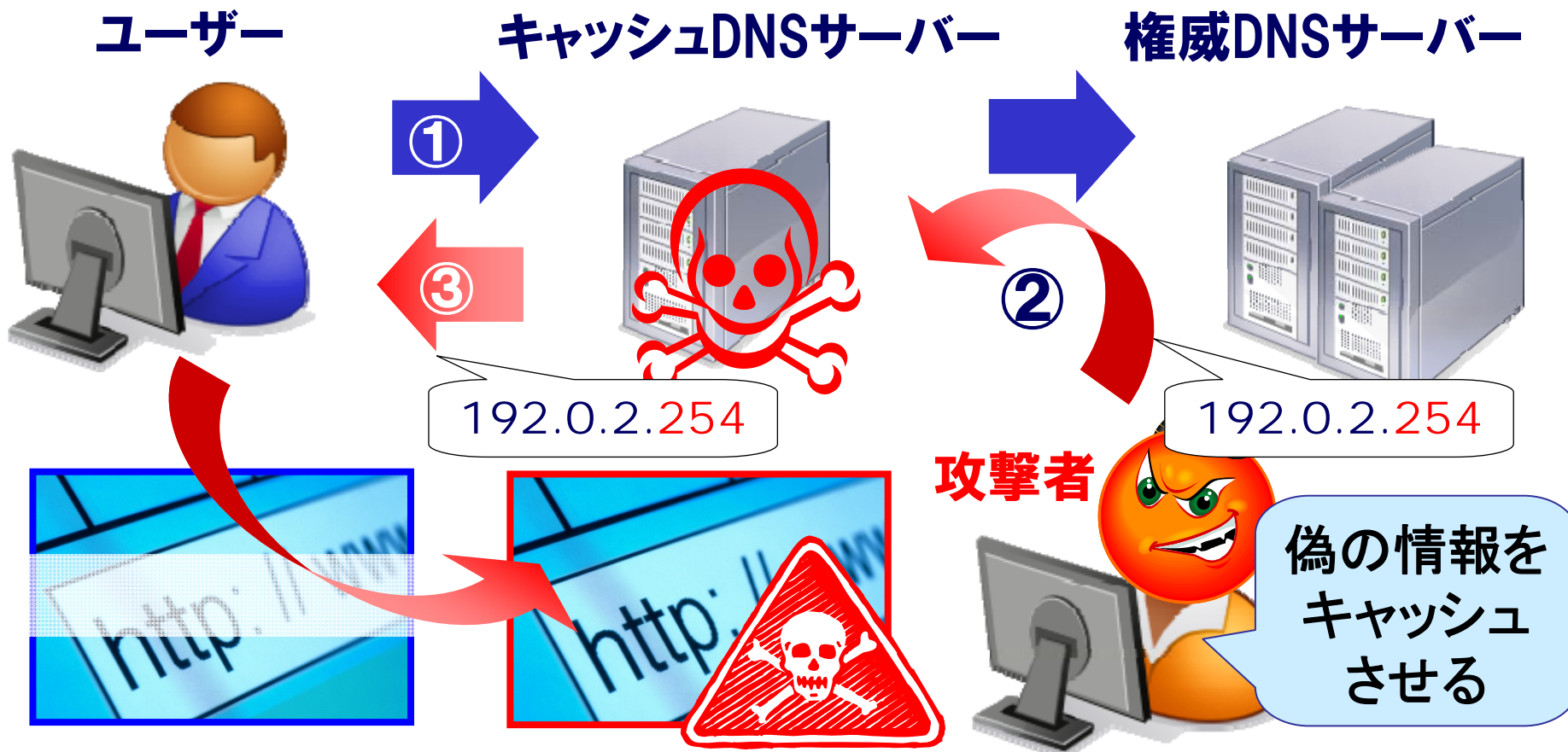
DNSSECの導入状況

DNSSECの導入状況

- rootゾーンにおけるDNSSEC
 - 2010年7月15日より正式運用
- TLDの導入状況
 - 既に多くのTLDで運用されている
 - gTLD
 - .com, .net, .org, .info, .asia...
 - 新gTLDではDNSSEC対応が必須項目
 - ccTLD
 - .se, .bg, .br, .us, .ch, .eu, .cz, .uk, .cc...
 - .jpは2011年1月16日より正式運用
 - 最新の導入状況
 - <http://www.ohmo.to/dnssec/maps/>

DNSキャッシュへの 毒入れと対策

DNSキャッシュへの毒入れ



**URLが本物のサイトと同じであるため
ブラウザのアドレスバーを見ただけでは判別できない**

毒入れ攻撃への対策

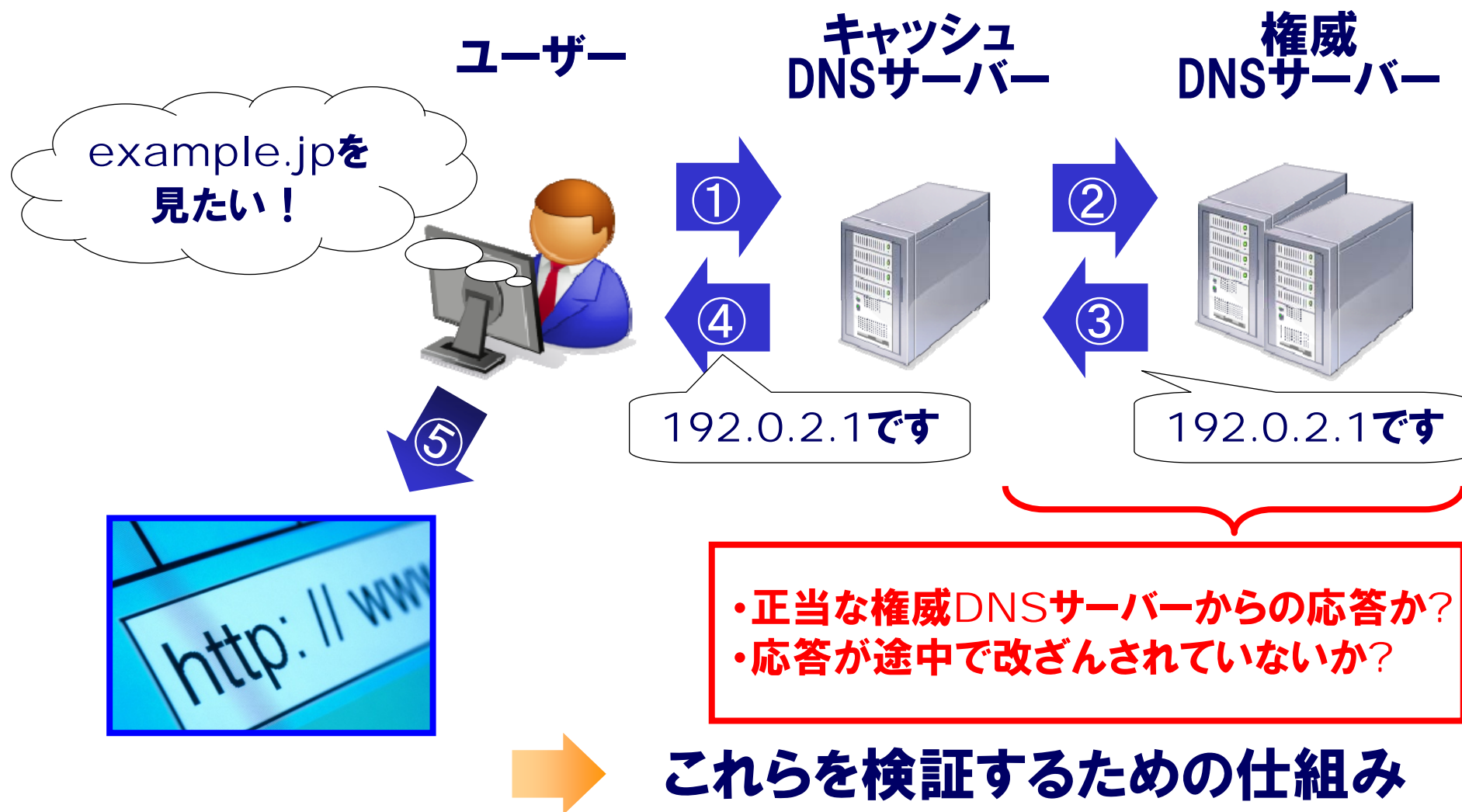
- 攻撃が成功する確率を下げる
 - 問い合わせポートのランダム化
 - 適切なアクセス制限を行う
 - etc...

→ 上記は現在すでに実施されている対策だが、根本的な対策ではない
- 根本的な対策
 - DNS応答が偽造/改ざんされていないことを受信側で確認可能にする

→ **DNSSECはこのための技術として開発された**

DNSSECのしくみ

DNSSECとは

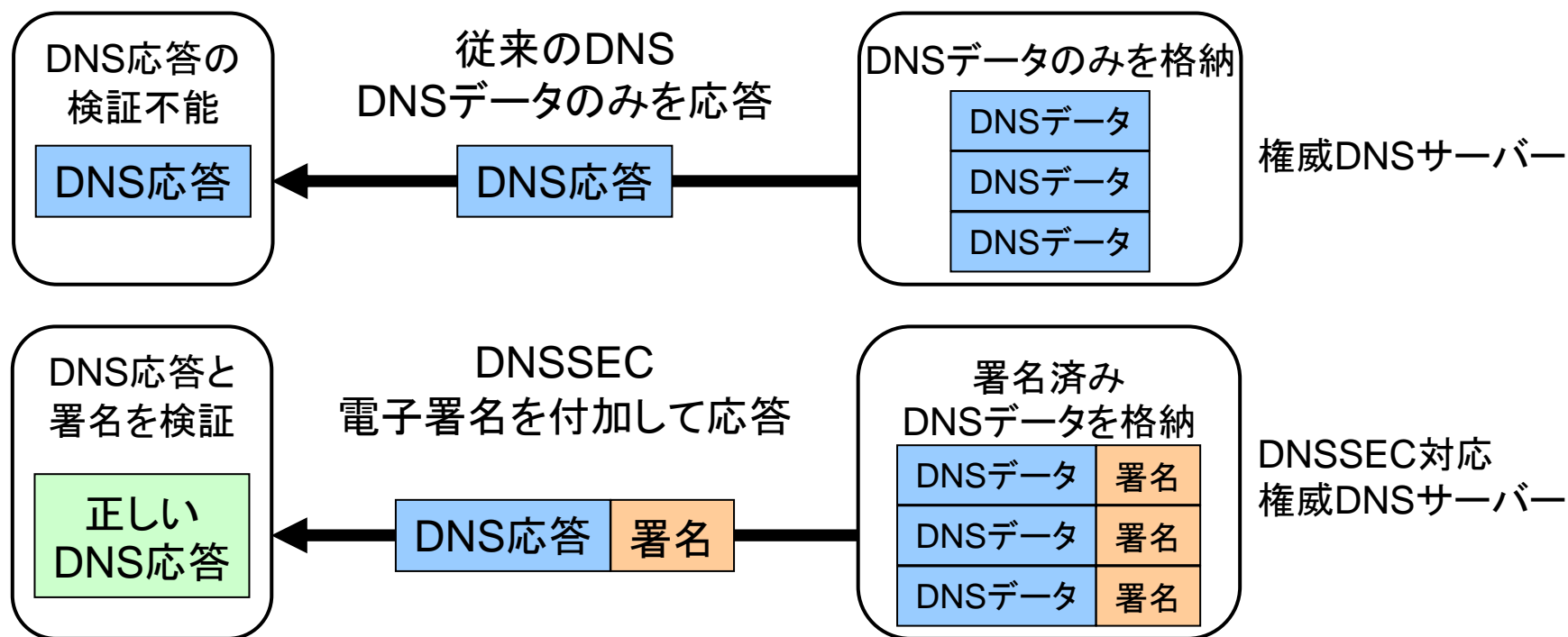


DNSSECとは

- DNSセキュリティ拡張
 - DNS SECURITY Extensions
- 受け取ったDNSレコードの出自・完全性(改ざんや欠落のないこと)を検証できる仕組み
- DNS応答に公開鍵暗号技術を用いた電子署名を付加
- 従来のDNSとの互換性を維持

従来のDNS vs DNSSEC

- 権威DNSサーバーで電子署名を付加し出自を保証
- 受信側でDNS応答の改ざん・欠落の有無を検出できる

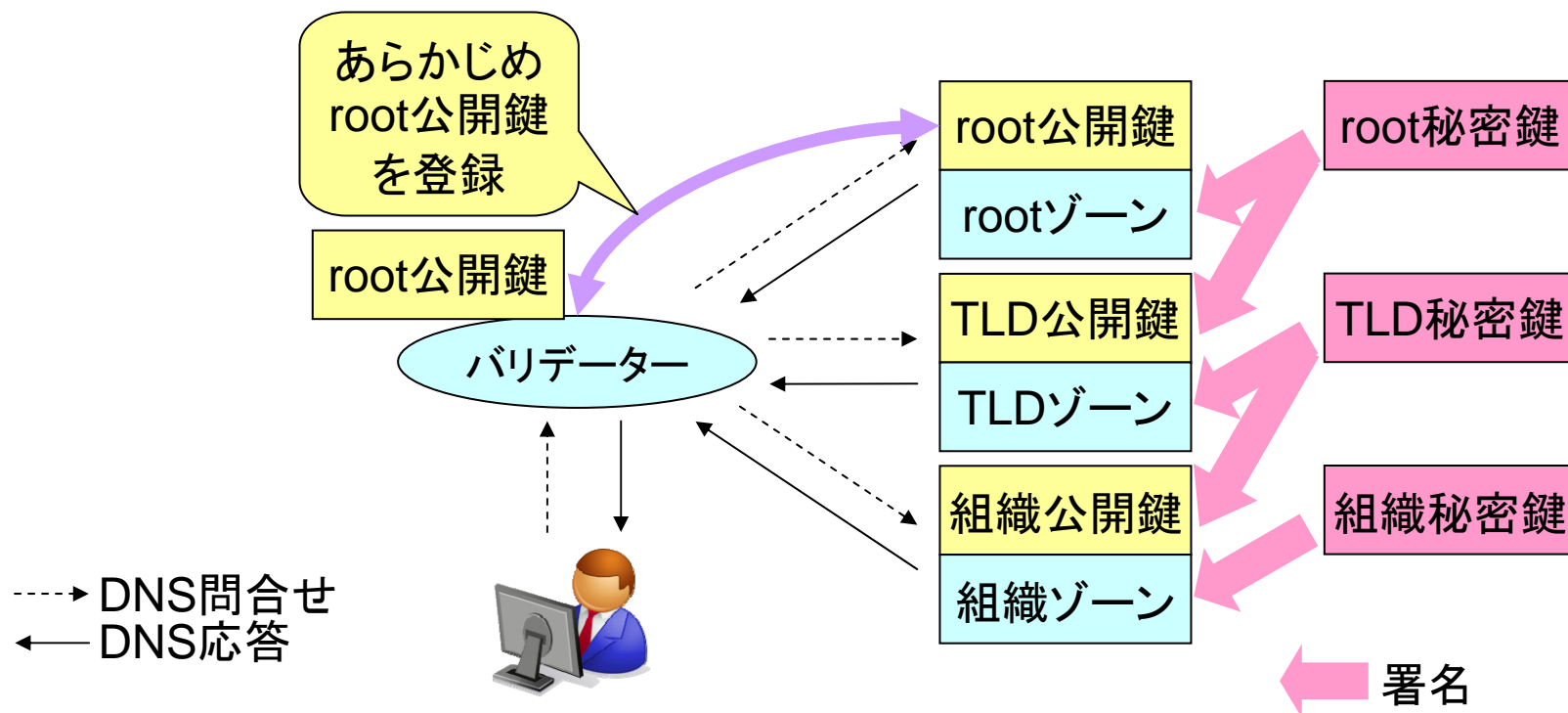


DNSSECの対象範囲

- 対象としているもの
 - 出自の保証
 - 正当な発信者からの応答であること
 - 完全性の保証
 - DNS応答が改ざん・欠落していないこと
- 対象としていないもの
 - 通信内容の暗号化
 - 通信内容を暗号化し、問い合わせ内容を秘匿すること

鍵と信頼の連鎖

DNSSECにおける信頼の連鎖の概念



- 秘密鍵で、自ゾーンと子ゾーンの公開鍵に署名
- root公開鍵をバリデータ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構築

2種類の鍵とDSレコード

- 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)
 - 親ゾーンにリソースレコードとして登録
 - 子ゾーンのKSKと等価な情報

ZSK (Zone Signing Key)

- 比較的暗号強度の低い鍵
 - RSAの場合、鍵長1024bit程度
- 署名コストが低い
 - 大規模ゾーンの署名にも適応できる
 - 安全確保のため、ある程度の周期で鍵を更新する必要がある
 - 現在の.jpでは1ヶ月ごとに更新
- 鍵更新は親ゾーンとは関係なく独立に行える
 - 2種類の鍵を使うメリット

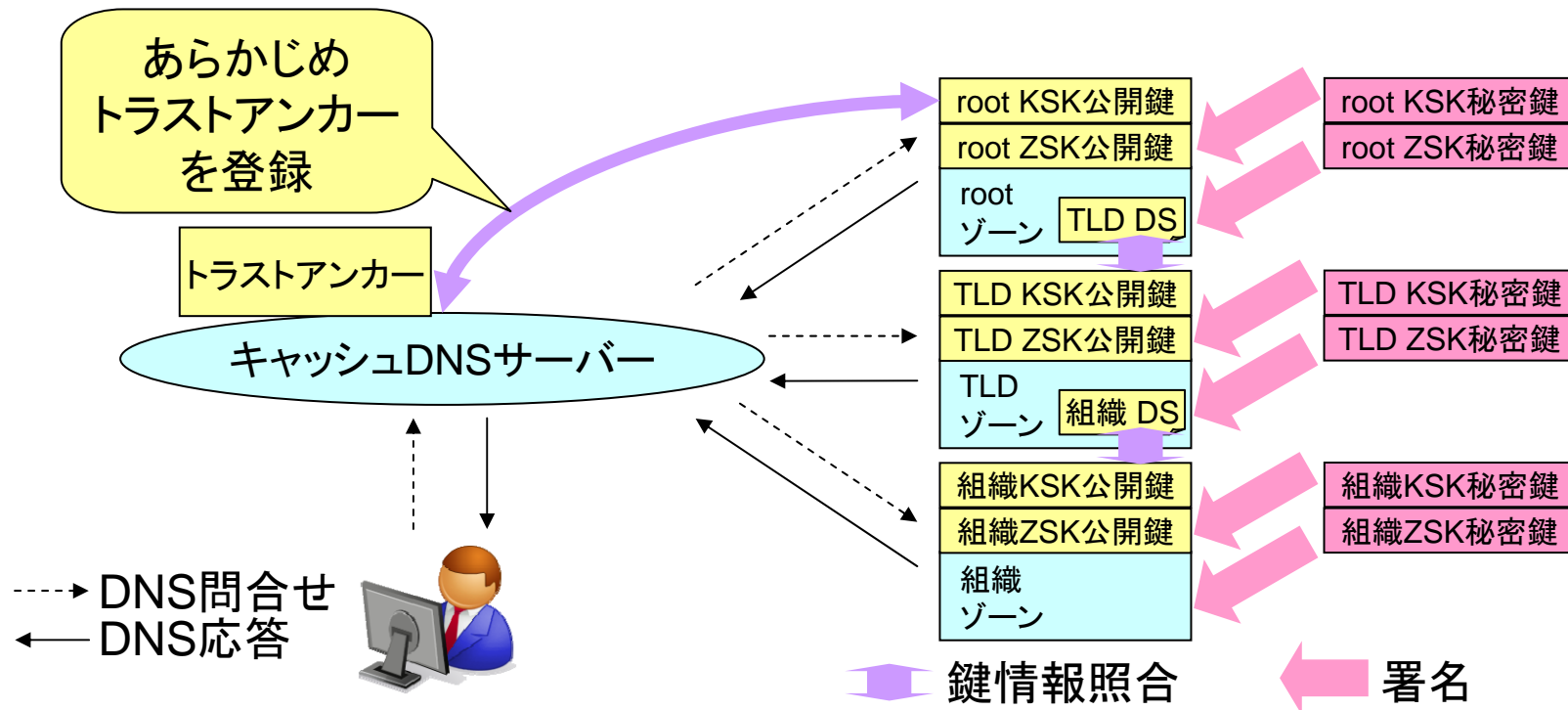
KSK (Key Signing Key)

- 比較的暗号強度の高い鍵
 - RSAの場合、鍵長2048bit程度
- 利用期間を長くできる
 - 鍵更新の頻度を低くできる
 - 現在の.jpでは1年ごとに更新
 - 署名コストは高いが、ゾーン内の公開鍵情報のみを署名対象とするため、全体のコスト増加は少ない
- 信頼の連鎖を構築するため、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必要がある
 - **KSKを変更する場合、同時にDSも更新する必要あり**

DS (Delegation Signer)

- KSK公開鍵を、SHA-1/SHA-256等のハッシュ関数で変換したリソースレコード
→KSK公開鍵と暗号論的に等価の情報
- 親ゾーンの委任ポイントに、NSと共に子ゾーンのDSを登録
 - 親ゾーンのZSKでDSに署名することにより、信頼の連鎖を構築

DNSSECの信頼の連鎖



- 親ゾーンのZSKでDSに署名することにより、信頼の連鎖を構築
 - 親ゾーンに登録したDSと子ゾーンのKSKの公開鍵の情報を照合
- バリデーターがDNS応答の署名を検証
 - 通常の場合、キャッシュDNSサーバーがバリデーターとなる
 - キャッシュDNSサーバーにrootゾーンのKSK公開鍵またはDSを登録する
→ **トラスタンカー**

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で指し示すのに使われているドメイン名がDNSSEC非対応であってもDNSSECの検証には影響しない
- 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ドメイン名がDNSSEC対応していなくても、example.jpドメイン名はDNSSEC検証可能

DNSSECの リソースレコード(RR)

DNSSEC関係のRR一覧

- DNSKEY 公開鍵の情報(ZSK・KSK)
- 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(不在証明(後述)に利用)
- NSEC3 NSECを改良したもの(後述)
- NSEC3PARAM NSEC3の生成に必要な情報

DNSKEY RR

- 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 8 AwEAAeNO41ymz+Iw(行末まで省略)
  ① ② ③ ④
```

- ① フラグ(256:ZSK、257:KSK)
- ② プロトコル番号(3:DNSSEC)
- ③ DNSSECアルゴリズム番号(8:RSASHA256)
- ④ 公開鍵(Base64で符号化)

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>

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 8 1 DF...(16進数40文字)
example.jp. IN DS 63604 8 2 E8...(16進数64文字)
                ①    ② ③    ④
```

- ① 鍵タグ(16bit)
- ② DNSSECアルゴリズム番号
- ③ ダイジェストアルゴリズム(1:SHA-1, 2:SHA-256)
- ④ KSK公開鍵のダイジェスト(ハッシュ値)

RRSIG RR

- 各RRへの署名で、RRSet毎に存在する
 - RRSet:同一RRの集合

```

ns.example.jp. IN RRSIG A 8 3 86400
                        ① ② ③ ④
                20121122120000 20121119000000 40002 example.jp.
                        ⑤ ⑥ ⑦ ⑧
                NiVihYAIZBEwfUUAbPazDRiBvhNH8S(以下省略)
                        ⑨
  
```

- ① 署名対象のRRSet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

RRSIG RR(続き)

```

ns.example.jp. IN RRSIG A 8 3 86400
                        ① ② ③ ④
20121122120000 20121119000000 40002 example.jp.
                        ⑤ ⑥ ⑦ ⑧
NiVihYAIZBEwfUUAbPazDRibvhNH8S(以下省略)
                        ⑨
  
```

- ④ 署名対象RRのTTL
- ⑤ 署名の満了時刻(UTC)
- ⑥ 署名の開始時刻(UTC)
 - “YYYYMMDDHHMMSS”形式で表記
 - UTCであることに注意

RRSIG RR(続き)

```

ns.example.jp. IN RRSIG A 8 3 86400
                        ① ② ③ ④
20121122120000 20121119000000 40002 example.jp.
                        ⑤ ⑥ ⑦ ⑧
NiVihYAIZBEwfUUAbPazDRiBvhNH8S (以下省略)
                        ⑨
  
```

⑦ 鍵タグ

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
- 署名には有効期間があるため、満了前に再署名が必要

不在証明

- 存在しない名前の偽装を防止するため「存在しない」ことを証明する必要がある
 - そのドメイン名は存在しない
 - そのRRSetは存在しない
 - DNSSECではRRSetに署名して検証している
 - 存在しないものには署名が付加できない
 - そこでDNSSECでは、存在によって不存在を証明することにした
- ➔このためのRRとしてNSEC RR/NSEC3 RRが使われる

NSEC RR

- NSEC RRの例

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

- 上記の意味

- アルファベット順でsec1.example.jpの次に存在するのはsec3.example.jp
- sec1.example.jpにはNS, DS, RRSIG, NSECの各RRが存在する

- 使用例

- sec2.example.jpのAレコードの問い合わせに対する応答
 - sec1.example.jpとsec3.example.jpの間に名前が存在しない
- sec1.example.jpのAレコードの問い合わせに対する応答
 - sec1.example.jpにはAレコードが存在しない

- 問題点

- NSEC RRの「次の名前」連鎖をたどることにより、外部からすべてのゾーンデータを入手できてしまう
- ➔ゾーンデータを取りにくくするための工夫が必要

NSEC3 RR

- NSEC3 RRの追加導入
 - ドメイン名を一方向性ハッシュ関数で処理することによりゾーンデータを取りにくくする
- NSEC3 RRの例

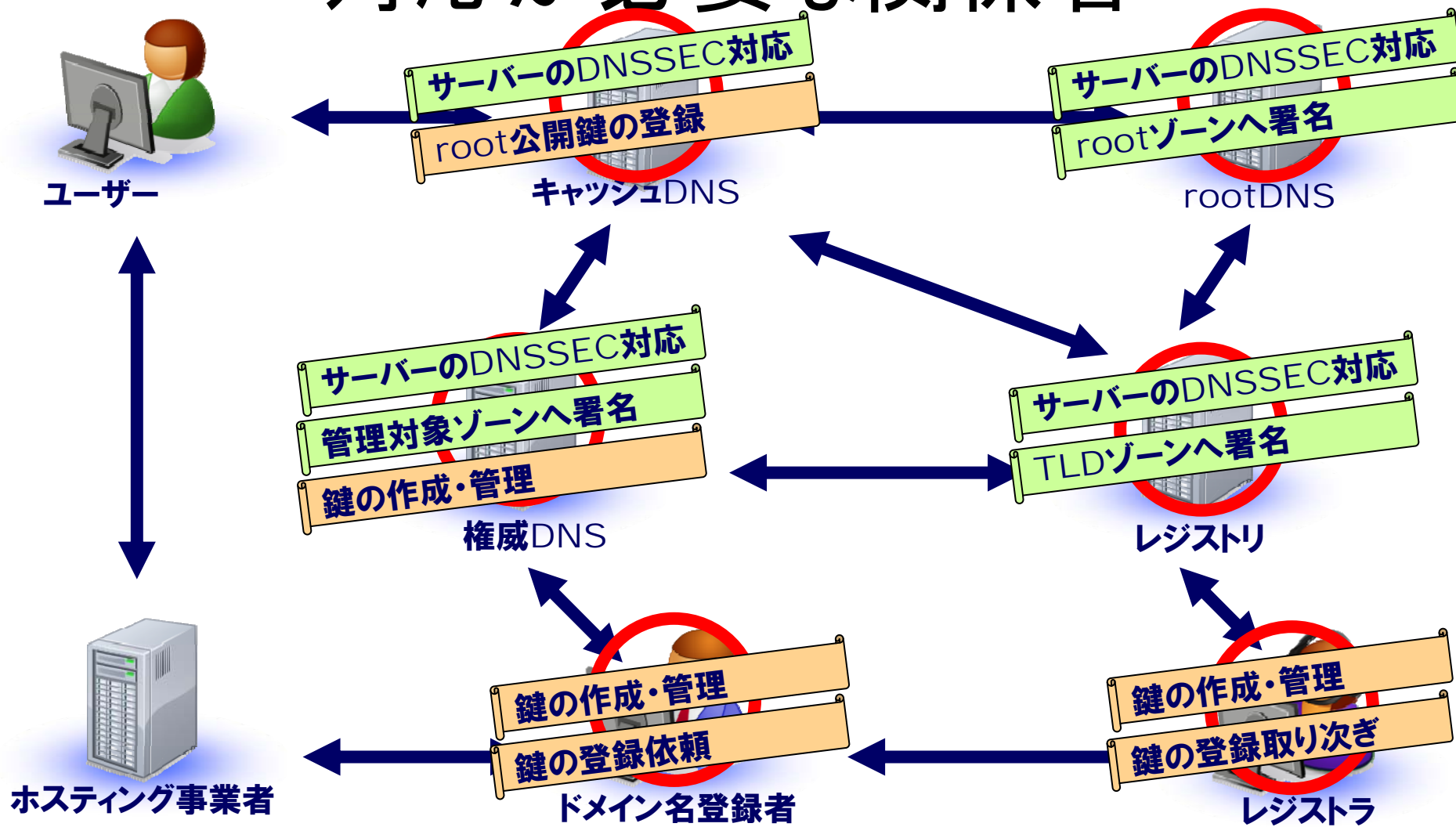
```
4HTJTU7UP56274L1C00Q9MLPHG2A2H85.example.jp.  
  IN NSEC3 1 0 3 123ABC B0B790UE4SAE4QB4RTB3PJSIH6JAOB7R  
  NS DS RRSIG
```

- NSECと比較しドメイン名の秘匿性は高まるが、ハッシュ値計算のためのコストが増加する
 - NSEC3とNSECはゾーンごとに使い分けることが可能
 - ゾーンデータを取りにくく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
- NSEC3 RRの生成は権威DNSサーバーで動的に行われる
 - NSEC3PARAM RRが必要
- NSEC3PARAM RRの例

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

対応が必要な関係者

対応が必要な関係者



まとめ

まとめ - 基礎概要 -

- DNSSECは、公開鍵暗号技術を利用した署名によるDNS応答検証のためのしくみ
 - 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公開鍵情報を使って署名を検証
 - DNSKEY RRまたはDS RR
 - 定期的な鍵の更新と再署名が必要

