NRIセキュアテクノロジーズ

DNSSEC 2011 スプリングフォーラム

鍵のロールオーバについて

2011年04月20日

NRIセキュアテクノロジーズ株式会社
MSS事業本部MSS事業一部
ITセキュリティアナリスト

中島　智広

〒240-0005
横浜市保土ヶ谷区神戸町134 NRIタワー

背景と目的

■背景

- DNSSECは公開鍵暗号によるデジタル署名を用いたDNSのセキュリティ拡張
- デジタル署名技術でなりすまし(キャッシュ汚染攻撃)を防ぐ
- 適切なDNSSEC運用のためには鍵のロールオーバ(更新)が必要とされる

■目的

- DNSSECの運用における鍵のロールオーバの必要性と、
  ロールオーバ方式や考慮点を理解することでDNSSECの安定的な運用に資する

眠くなるかもしれませんが、頭に置いて頂きたい点をお話しします

(復習)DNSSECで用いられる鍵とDS

- ゾーン署名鍵(ZSK:Zone Signing Key)
  - ゾーンを署名する鍵
  - 署名コストを低くするため比較的暗号強度の低い鍵長を用いる(1024bitを推奨)
  - 安全性確保のため高い頻度で更新を行う必要がある(一ヶ月周期を推奨)
- 鍵署名鍵(KSK:Key Signing Key)
  - 鍵を署名し、親子ゾーン間で鍵の信頼の連鎖を形成するための鍵
  - 更新頻度を低くするため暗号強度の高い鍵長を用いる(2048bitを推奨)
  - 暗号強度が高いため更新頻度は低くてもよい(一年周期を推奨)
- DS(Delegation Signer)
  - 親ゾーンに登録する暗号論的にKSKと等価な情報
  - DNSメッセージがあふれるのを抑制するためKSKからハッシュ関数を用いて生成

1.鍵のロールオーバの必要性

2.鍵のロールオーバの基本と要求

3.鍵のロールオーバ方式

4.鍵の状態遷移に基づくタイミングの整理

5.鍵のロールオーバの実装例

6. DNSSECの有効化/無効化

7. 緊急ロールオーバとStandby Key

8.まとめ

鍵のロールオーバの必要性

■署名(RRSIG)には有効期限があるが鍵(DNSKEY)に有効期限はない
➔ロールオーバせず同じ鍵を使い続けて運用することもできる

```
www.dnssec.jp.  IN A     www.xxx.yyy.zzz
                IN RRSIG  A 8 4 86400 20110517173250 (・・・
dnssec.jp.      IN DNSKEY 256 3 8 ABCDEFGHIJKLMNOPQR・・・
                IN DNSKEY 257 3 8 ABCDEFGHIJKLMNOPQR・・・
```

有効期限

■しかし、下記への対応のためには鍵のロールオーバが必要

- 同じ暗号鍵を使い続けることによる安全性の低下
- 暗号アルゴリズムの危殆化と新アルゴリズムへの移行
- 暗号鍵の漏出/紛失時の緊急オペレーション

適切なDNSSEC運用のために鍵のロールオーバは必要不可欠

暗号アルゴリズムの危殆化

- 時間の経過により当初想定されていたよりも低いコストで、暗号アルゴリズムのセキュリティ上の性質が危うくなる現象
  [原因]
  - 計算機性能の向上
  - 計算機モデルの変化
  - 攻撃手法の進歩
- 一般に危殆化は段階的に進行するため緊急対応が必要となることは稀
- DNSSECはRSA-SHA256/512への対応が完了しているため、暗号アルゴリズムのロールオーバはしばらく発生しないと考えられる
- (余談)暗号アルゴリズム自身に問題は無くとも
  暗号モジュールの実装に脆弱性が発見されることもある

鍵の漏出/紛失

■鍵の漏出

- 問題点：鍵を悪用されることでキャッシュ汚染攻撃が可能となる
- 解決策：速やかに新しい鍵へロールオーバを行う

■鍵の紛失

- 問題点：現在の鍵を用いて署名の更新ができないため
  現在の署名の有効期限内に対策ができなければ検証に失敗(=名前解決不可)
- 解決策：基本的にはDNSSECをいったん無効化するしかないが、
  新しい鍵へのロールオーバで対応できるケースもある

(参考事例).ukのハードウェア障害によるZSK更新失敗

- 鍵を保管・管理するHSMの故障により鍵を利用できなくなった

目次

鍵のロールオーバの基本

■正しく検証可能な鍵と署名のペアの維持が必要

- 正しく検証可能な鍵と署名のペアが
  一組以上ある状態を維持しながら古い鍵と署名を削除する
- 正しく検証可能な鍵と署名のペアがなくなると
  バリデータで検証に失敗しServFail(=名前解決不可)となってしまう

■バリデータはキャッシュされたデータを用いて検証を行う可能性があるため鍵と署名のキャッシュ時間(TTL)の考慮が必要

- 新しい鍵を登録してからも一定時間経過するまでは古い鍵でも検証可能とする
- ロールオーバ完了後も一定時間経過するまでは古い鍵もしくは署名を削除しない

単純に古い鍵と署名を新しく置き換えるだけでは駄目

ロールオーバ方式への要求

■ZSKロールオーバ

- 複数の鍵を用いて複数の署名を作成することは避けたい
  - ゾーンファイルの全てのレコードに署名するため処理負荷が大きくなる
  - 署名が複数になるとレコードの応答サイズが大きくなる

■KSKロールオーバ

- DSの変更はレジストラへの申請といった手間がかかるため回数を少なくしたい
- KSKはDNSKEYに対してのみ署名を行うため処理負荷は問題とならない

鍵や署名を公開する順番やタイミングによって複数の方式があるが、
上記要求を満たす方式の使用が望ましい

1.鍵のロールオーバの必要性

2.鍵のロールオーバの基本と要求

3.鍵のロールオーバ方式

4.鍵の状態遷移に基づくタイミングの整理

5.鍵のロールオーバの実装例

6. DNSSECの有効化/無効化

7. 緊急ロールオーバとStandby Key

8.まとめ

推奨される鍵のロールオーバ方式

■ZSKロールオーバ 推奨方式

Pre-Publication(事前公開方式)

① 新しい鍵を公開する
② 新しい鍵でのみ署名を作成し、古い鍵で作られた署名を削除する
③ 古い鍵を削除する

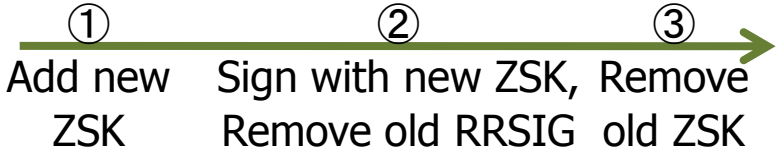
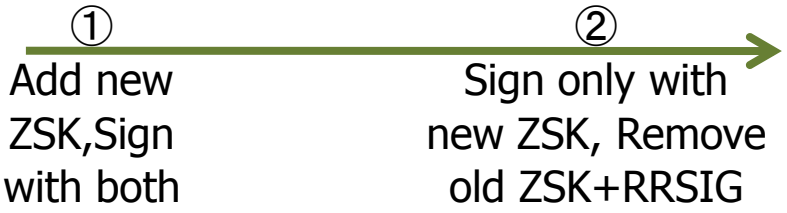
■KSKロールオーバ 推奨方式

Double-Signature(二重署名方式)

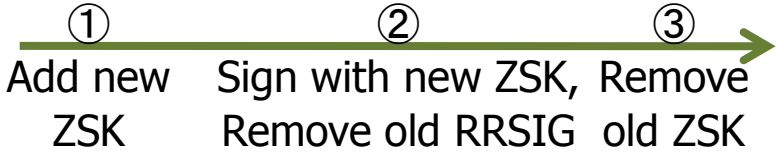
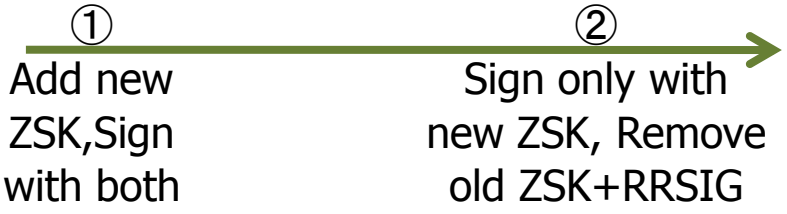
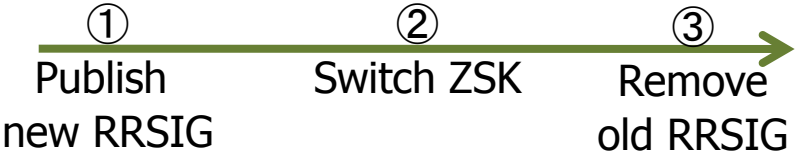
① 子ゾーンに新しい鍵と署名を追加登録
② 親ゾーンへ新しいDSを登録(上書き)
③ 古い鍵と署名を子ゾーンから削除

これだけ知っていれば十分ですが、次項より他との比較を紹介

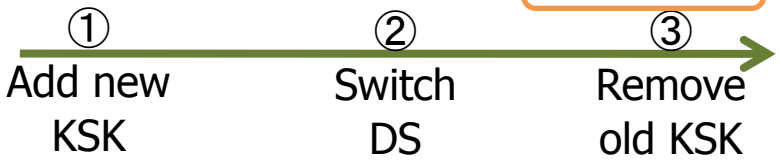
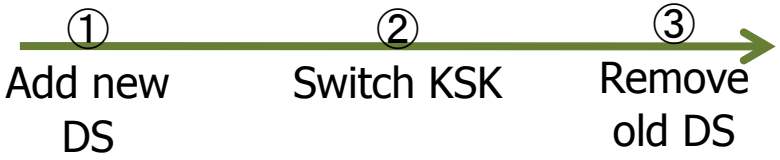
ZSKロールオーバ方式（手順）

方式	手順
Pre-Publication 推奨方式  	① 新しい鍵を公開する ② 新しい鍵でのみ署名を作成し、古い鍵で作られた署名を削除する ③ 古い鍵を削除する
Double-Signature  	① 新しい鍵と署名を同時に公開する 古い鍵と署名はそのまま残す ② 古い鍵と古い鍵で作られた署名を同時に削除する
Double-RRSIG  	① 新旧の鍵で作成された2つの署名を公開する ② 新しい鍵を公開しつつ古い鍵を削除する ③ 古い鍵で作成された署名を削除する

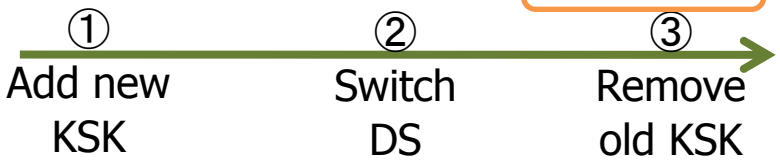
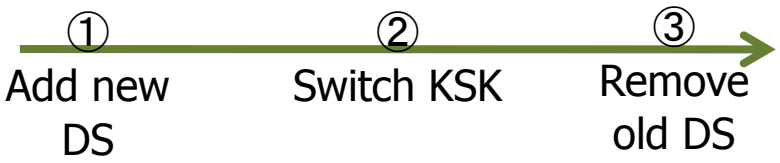
ZSKロールオーバ方式（長所と短所）

方式	長所	短所
Pre-Publication 推奨方式  	DNSSECのデータ量を最小にとどめることができ、パフォーマンスへの影響を抑えることができる	鍵の事前登録を行う点で3つの方式の中では複雑
Double-Signature  	3つの方式の中で最も手順が少なくシンプル	新旧の鍵で二重に署名を行うため、DNSSECのデータ量が大きくなる
Double-RRSIG  	なし	新旧の鍵で二重に署名を行うため、DNSSECのデータ量が大きくなり、かつ手順が複雑

KSKロールオーバ方式（手順）

方式	手順
Double-Signature 推奨方式  	① 子ゾーンに新しい鍵と署名を登録(追加) ② 親ゾーンへ新しいDSの登録申請を行う(上書き) ③ 古い鍵と署名を子ゾーンから削除
Double-DS  	① 親ゾーンに新しいDSを登録申請を行う(追加) ② 子ゾーンに新しい鍵と署名を登録 ③ 親ゾーンから古いDSの削除申請を行う
Double-RRset  	① 子ゾーンに新しい鍵を登録し 親ゾーンに新しいDSの登録申請を行う ② 子ゾーンから古いKSKを削除する 親ゾーンに古いDSの削除申請を行う

KSKロールオーバ方式（長所と短所）

方式	長所	短所
Double-Signature 推奨方式  	親ゾーンの更新が1回で済む	2つのKSK RRsetが含まれるためパケットサイズが大きくなる
Double-DS  	KSK RRsetのパケットサイズを最小にできる	親ゾーンの更新が2回
Double-RRset  	親ゾーンと子ゾーンの作業をほぼ同じタイミングで実施でき、業務フローを作りやすい	2つのKSK RRsetが含まれるためパケットサイズが大きくなる

1.鍵のロールオーバの必要性

2.鍵のロールオーバの基本と要求

3.鍵のロールオーバ方式

4.鍵の状態遷移に基づくタイミングの整理

5.鍵のロールオーバの実装例

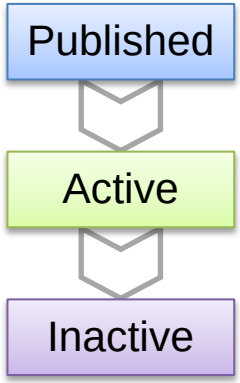
6. DNSSECの有効化/無効化

7. 緊急ロールオーバとStandby Key

8.まとめ

鍵の状態遷移に基づくタイミングの整理

- 鍵のロールオーバの適切なタイミングは、
  複数のパラメータを考慮すると複雑でわかりにいが、
  鍵の状態遷移に基づいて整理するとわかりやすくなる
- 鍵の状態遷移の定義は実装やドキュメントで名称や粒度が異なるが
  今回は下記の3状態を用いて推奨ロールオーバ2方式における
  鍵のロールオーバの適切なタイミングを整理する

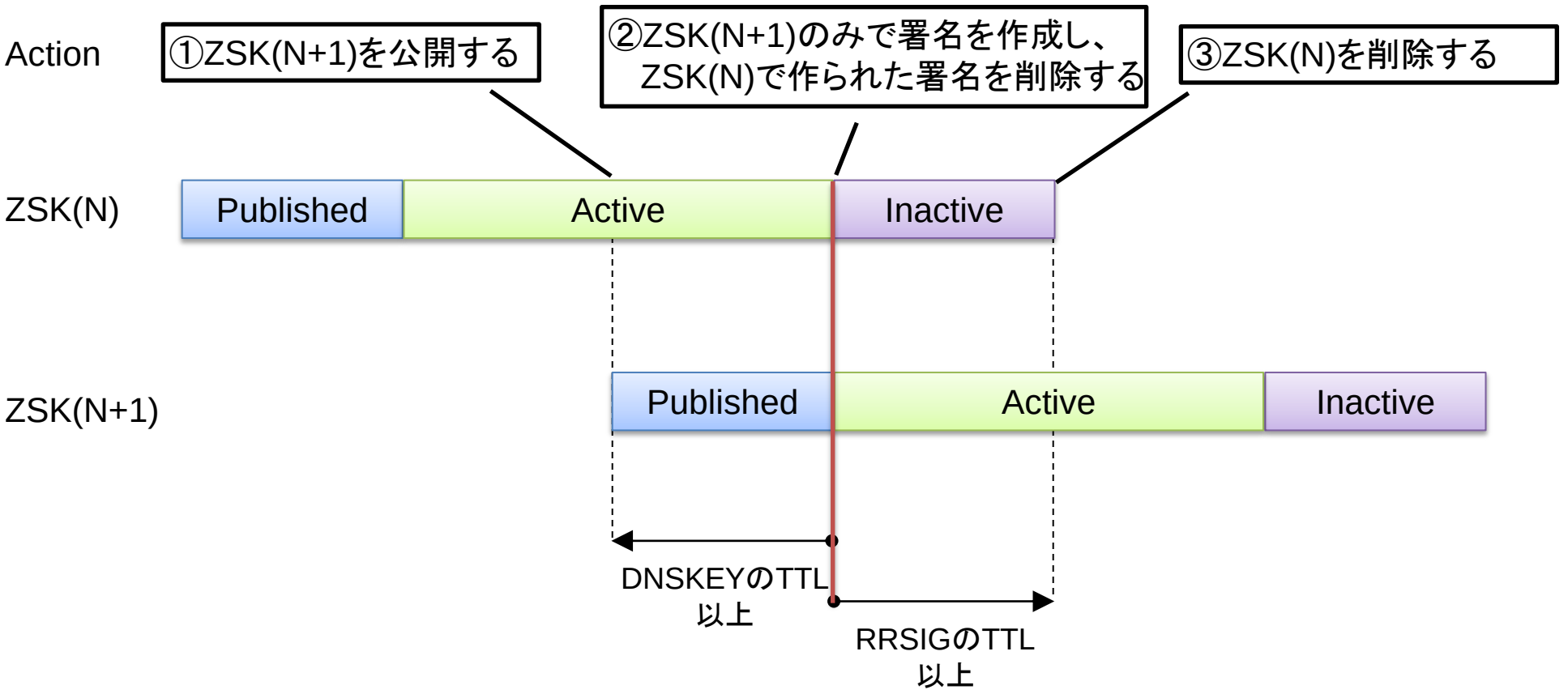


鍵がゾーンファイル内に事前公開された状態

鍵がゾーンファイル内にあり、かつ署名に用いられている状態

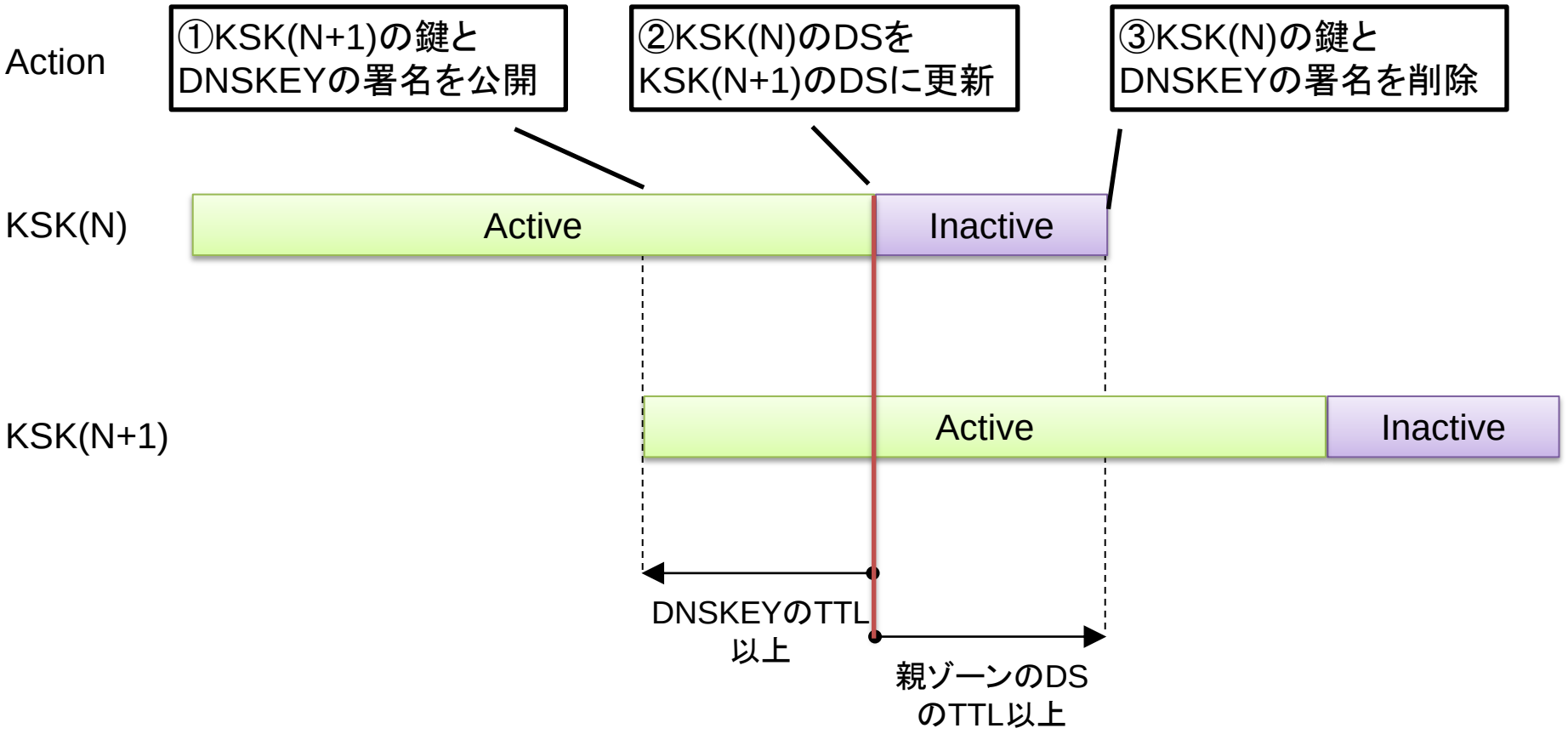
鍵がゾーンファイルに登録されているが署名には用いられていない状態

ZSKのロールオーバ（Pre-Publication方式）



署名に用いるZSKの入れ替え前後で適切な待ち時間が必要

KSKのロールオーバ(Double-Signature方式)



親ゾーンのDS更新の前後で適切な待ち時間が必要

鍵の状態遷移で考慮しなければならない時間

- TTL(キャッシュ保持期間)
  - DNSKEYのTTL
  - RRSIGのTTL
  - 親ゾーンのDSのTTL
- 下記について簡略化のため省略したが必要な場合考慮した方がよい
  - 親ゾーンの更新に要する時間
    - レジストラにDSの登録を申請してから反映されるまでの時間
  - ゾーン転送に要する時間
    - SOAレコードのReflesh値
    - ゾーン転送に失敗した場合のリトライ時間

環境にあわせて余裕をもって必要な待ち時間を定める

1.鍵のロールオーバの必要性

2.鍵のロールオーバの基本と要求

3.鍵のロールオーバ方式

4.鍵の状態遷移に基づくタイミングの整理

5.鍵のロールオーバの実装例

6. DNSSECの有効化/無効化

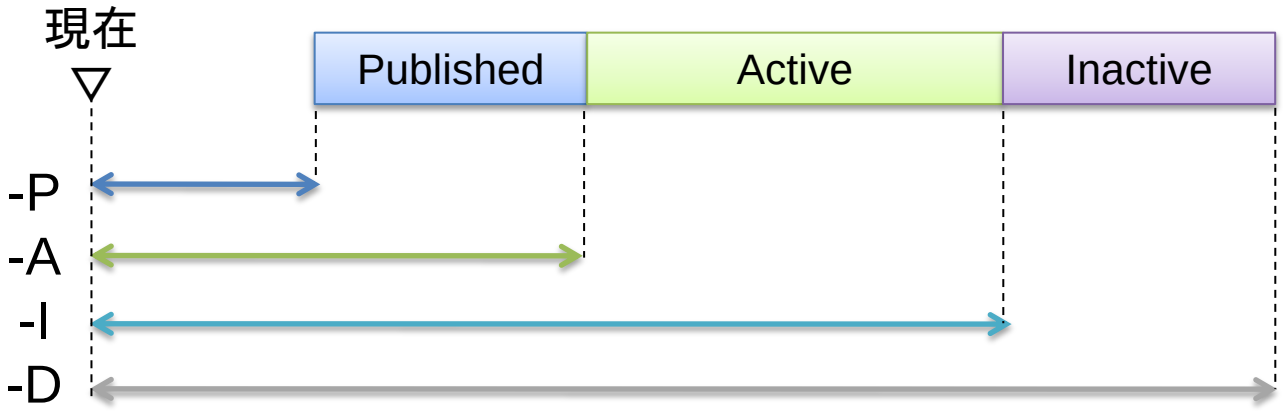
7. 緊急ロールオーバとStandby Key

8.まとめ

ISC BIND SmartSigning

- 鍵ファイル作成時に時刻情報を付加し、
  これに基づいて自動的に鍵と署名の登録を行う仕組み
- dnssec-signzone -Sを実施したタイミングで有効な鍵を用いる
  例)1日後にゾーンに登録、2日後から30日間署名に利用され
  その1日後にゾーンから削除される鍵を作成し、Smart Signingを実施

```
# dnssec-keygen  -a RSASHA256 -b 1024 -q ￥
        -P 1D -A 2D -I +32D -D +33D example.jp
# dnssec-signzone -S example.jp
```



ISC BIND SmartSigningを用いた実装例

■初回

```
# dnssec-keygen  -a RSASHA256 -b 1024 -q ¥
        -P now -A now -I +30D -D +31D example.jp
# dnssec-keygen - a RSASHA256 -b 1024 -q ¥
        -P now -A 30D -I +60D -D +61D example.jp
```

■以後毎日

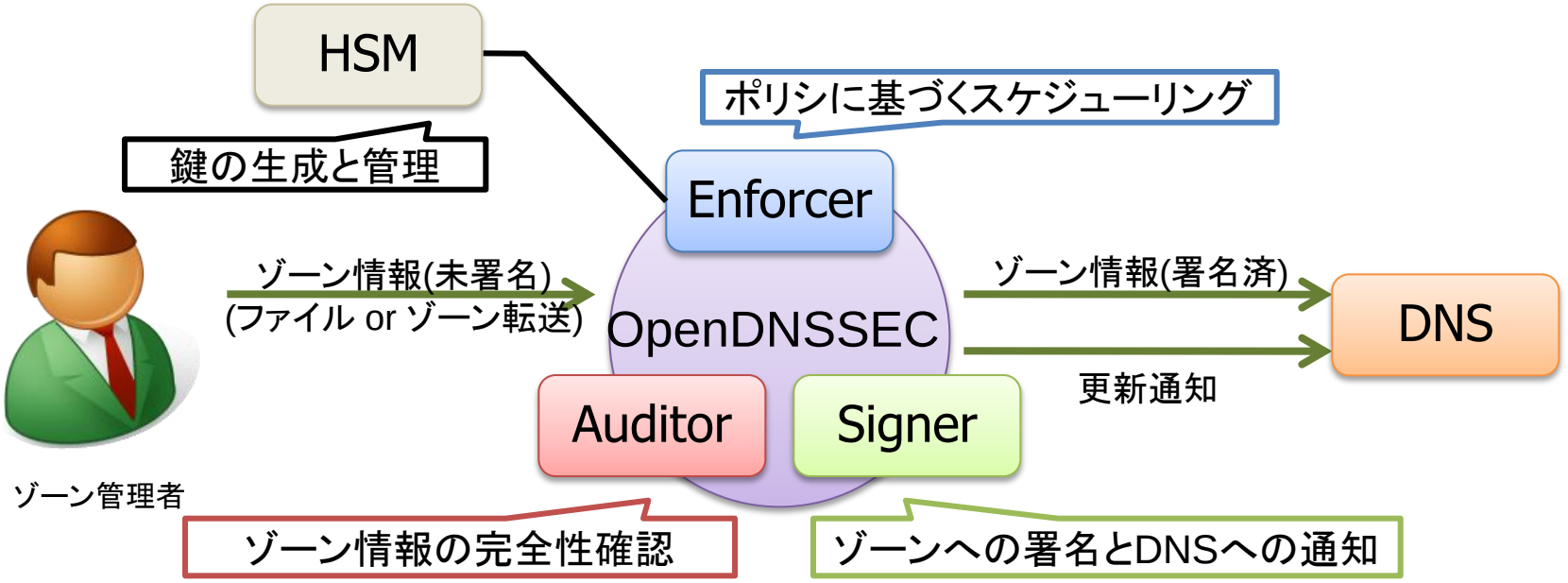
```
# dnssec-signzone -S example.jp
# rndc reload example.jp
```

■以後30日毎

```
# dnssec-keygen - a RSASHA256 -b 1024 -q ¥
        -P now -A 30D -I +60D -D +61D example.jp
```

OpenDNSSEC

■DNSSEC運用の全過程を自動化する運用支援ツール



OpenDNSSECを用いた実装例

- Key And Security Policy(kasp.xml)

```
<Key>
  <TTL>PT3600S</TTL>
  <RetireSafety>PT3600S</RetireSafety>
  <PublishSafety>PT3600S</PublishSafety>
  <Purge>P1D</Purge>
  <ZSK>
    <Algorithm length="1024">7</Algorithm>
    <Lifetime>P30D</Lifetime>
    <Repository>SoftHSM</Repository>
    <Standby>0</Standby>
  </ZSK>
</Key>
```

ポリシを定義するだけで自動的に鍵作成・署名を行う

1.鍵のロールオーバの必要性

2.鍵のロールオーバの基本と要求

3.鍵のロールオーバ方式

4.鍵の状態遷移に基づくタイミングの整理

5.鍵のロールオーバの実装例

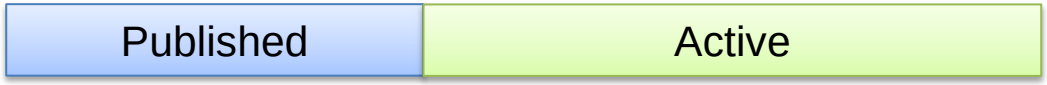
6. DNSSECの有効化/無効化

7. 緊急ロールオーバとStandby Key

8.まとめ

DNSSECの有効化/無効化

- DNSSEC導入後も必要に応じて有効化/無効化する可能性がある
  - トラブル時にやむなく
  - レジストラ移転
- 無効化する際の注意点
  - 親ゾーンからDSを削除後、DSのTTLが経過して初めて無効となる
  - DSのTTLが経過するまで子ゾーンから鍵と署名を消してはならない
- 有効化する際の注意点
  - 無効化時にはDNSKEYが存在しないため、
    ネガティブキャッシュのTTL分だけ事前公開が必要



SOAレコードのTTL
とSOAレコードの
minimum値の最小値

1.鍵のロールオーバの必要性

2.鍵のロールオーバの基本と要求

3.鍵のロールオーバ方式

4.鍵の状態遷移に基づくタイミングの整理

5.鍵のロールオーバの実装例

6. DNSSECの有効化/無効化

7. 緊急ロールオーバとStandby Key

8.まとめ

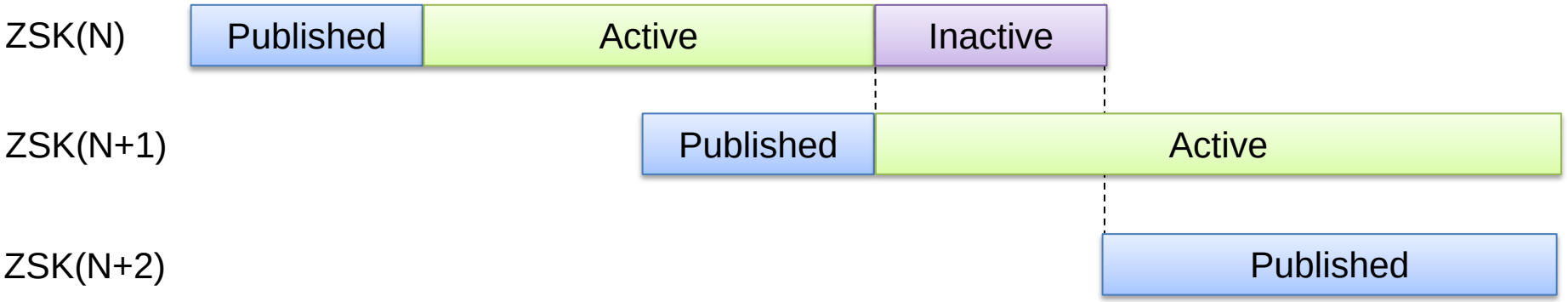
緊急ロールオーバとStandby Key

- 次の鍵の準備ができていない状態で
  緊急でロールオーバを行う必要に迫られた場合、
  鍵生成から着手すると時間が大幅にかかってしまう
  - 鍵生成、鍵の登録、TTL時間の経過、鍵とDSの更新
  - 特にKSKの緊急ロールオーバの場合親ゾーンのTTLが長く時間を要する
- Standby Keyは事前に用意しておいた予備鍵を用いることで
  ロールオーバ時間を短縮する手法及びそのために用いる鍵
- ロールオーバ方式によってStandby Keyの使用可否がある
  ➔使用可能な方式
  - ZSKロールオーバ
    - Pre-Publication
  - KSKロールオーバ
    - Double-Signature
    - Double-DS

あらかじめ鍵を準備することで緊急ロールオーバを短縮する

Standby Keyを用いた緊急ロールオーバの例（ZSK）

■準備



- 新しい鍵を利用開始する直前に公開するのではなく、古い鍵の削除時にさらに新しい鍵を公開しておくことでいつでもロールオーバに利用可能な状態としておく

■緊急ロールオーバ

- すでに公開済みの鍵を利用してロールオーバを行う

1.鍵のロールオーバの必要性

2.鍵のロールオーバの基本と要求

3.鍵のロールオーバ方式

4.鍵の状態遷移に基づくタイミングの整理

5.鍵のロールオーバの実装例

6. DNSSECの有効化/無効化

7. 緊急ロールオーバとStandby Key

8.まとめ

まとめ

- 適切なDNSSEC運用のためには鍵のロールオーバが必要
- ロールオーバ方式は複数あるが推奨される方式は各々一つ
  - ZSK: Pre-Publication(事前公開方式)
  - KSK: Double-Signature(二重署名方式)
- ロールオーバのタイミングは
  鍵の状態遷移に基づいて整理するとわかりやすくなる
- 実装においては可能な限り自動化を推奨
- DNSSECを有効化/無効化する際もキャッシュの状態に注意
- Standby Keyを用いた緊急ロールオーバは検討する価値有り

各組織でポリシを決めて安定的なロールオーバ運用を行いましょう

NRIセキュアテクノロジーズ