Root KSK 更新に対応する方法

東京大学　総合文化研究科

石原知洋

概要

- Root KSK Rollover とは？
- 更新方法
- 自動更新：　RFC5011: Automated Updates of DNS Security (DNSSEC) Trust Anchors “DNSSEC トラストアンカーの自動更新”

Root KSK 更新とは？

- DNSSEC の（というより世の中の）鍵は定期的な更新が必要
  - DNS ルートサーバの鍵も例外ではありません！
- 今までの講習で trust-anchor を設定したと思いますが・・・・
  - trust-anchor はルートの KSK 公開鍵（DNSKEY）なので、これも変わります

KSK 更新に追従できないと・・・

- 名前が引けなくなる
- Trust-anchor 以下のすべてのドメインについて検証失敗で ServFail を返すようになります

追従の方法

- 手動と自動の二種類の方法がある

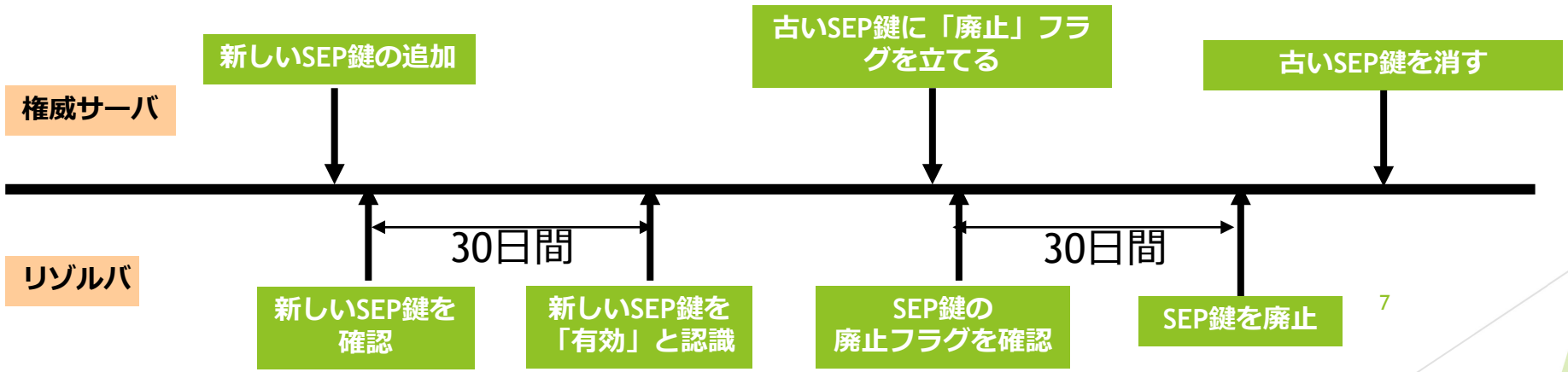
- 手動→設定ファイルのトラストアンカーを新しい物に書き換える
- 自動→リゾルバサーバ上で RFC5011 機構を動作させる

手動での設定

- 以前の講義で説明された（であろう）トラストアンカーの設定を書き換える
- Bind ならば named.conf の trusted-keys ディレクティブで指定
- Unbound ならば /etc/unbound/root.key に鍵を置く
  - unbound-anchor コマンドを使ってもよい
- 新しい鍵が正しいことは DNS/DNSSEC 以外の方法で担保する
  - Unbound では別途トラストアンカー検証用の証明書を持っているため、そちらを使って検証可能

自動更新：Automated Updates of DNS Security (DNSSEC) Trust Anchors （RFC5011）

- DNSSEC で使われるトラストアンカーの更新について定めた仕様
- 新しいSEP鍵を追加した際には、新旧両方のSEP鍵をそれぞれの鍵で署名
  - 古いSEP鍵を持っている検証者は追加されたSEP鍵を検証できる
- 追加したSEP鍵を30日間存在していることを確認した後に新しいSEP鍵を有効化する
- 削除時もSEP鍵に削除フラグを付け、30日間削除フラグを確認し続けて初めて削除



BIND の設定方法

- named.conf 内、Managed-keys ディレクティブで設定する
- 内容は DNSKEY レコードそのもの
- Trusted-keys とは “initial-key” と書く部分のみ違う

```
managed-keys {
        "."  initial-key  257 3 5
          "AwEAAcxDNuXWPybYlz5OvFFNSSZzC290IPIg1H+stlsIJ74VZS2
             (中略)
          ExPL";
};
```

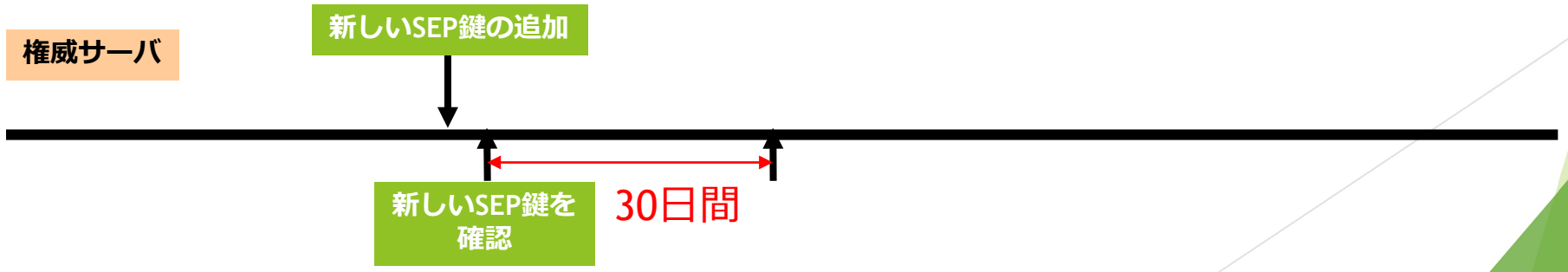
設定後の動き

- Working ディレクトリの中に managed-keys.bind と、managed-keys.bind.jnl の２種類のファイルができる
  - .jnl はバイナリのジャーナルファイル、定期的に .bind に書き出される
- 鍵が trusted と認識された時間が記録される

```
$ORIGIN .
$TTL 0  ; 0 seconds
@                       IN SOA  . . (
                                5          ; serial
                                0          ; refresh (0 seconds)
                                0          ; retry (0 seconds)
                                0          ; expire (0 seconds)
                                0          ; minimum (0 seconds)
                                )
                        KEYDATA 20151103081348 20151103061348 19700101000000 257 3 5 (
                                (中略)
                                HIOk9UtlA95+X+SN/QiL0A4fmS0/0Fj9VbJP4Go2ExPL
                                ) ; KSK; alg = RSASHA1; key id = 64489
                                ; next refresh: Tue, 03 Nov 2015 08:13:48 GMT
                                ; trusted since: Tue, 03 Nov 2015 06:13:48 GMT
```

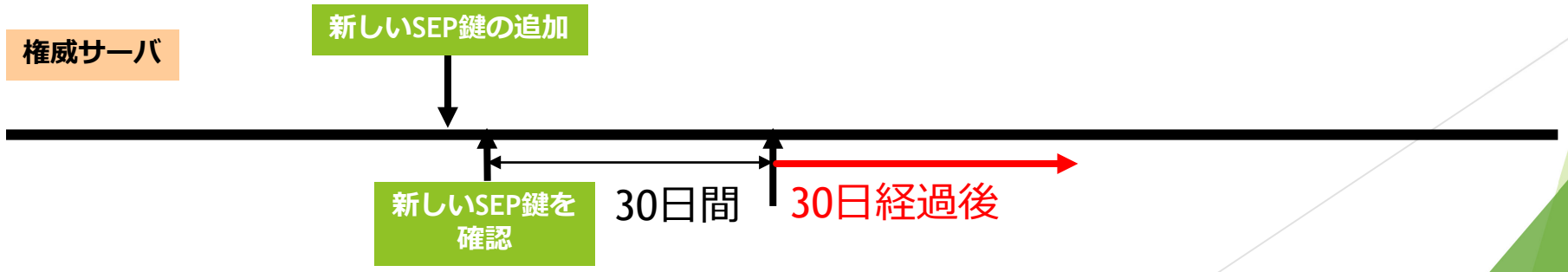
鍵の追加が確認された状態：BIND

```
@                   IN SOA  . . (
                                (略)
                        )
                    KEYDATA 20151105045507 20151104035439 19700101000000 257 3 5 (
                            (略)
                        ) ; KSK; alg = RSASHA1; key id = 64489
                        ; trusted since: Wed, 04 Nov 2015 03:54:39 GMT
                    KEYDATA 20151105045507 20151205035507 19700101000000 257 3 5 (
                            (略)
                        ) ; KSK; alg = RSASHA1; key id = 36058
                        ; next refresh: Thu, 05 Nov 2015 04:55:07 GMT
                        ; trust pending: Sat, 05 Dec 2015 03:55:07 GMT
```



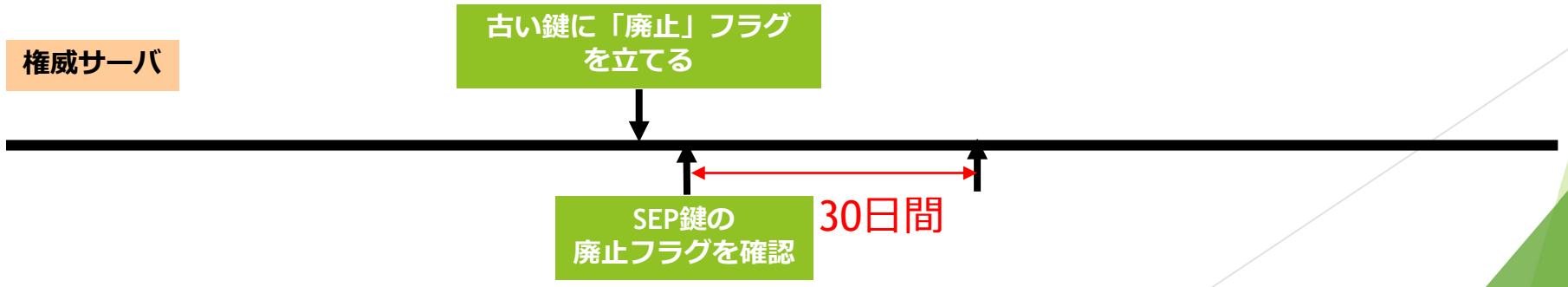
追加された鍵が承認された状態：BIND

```
@                   IN SOA  . . (
                                (略)
                        )
                    KEYDATA 20151105045507 20151104035439 19700101000000 257 3 5 (
                                (略)
                        ) ; KSK; alg = RSASHA1; key id = 64489
                        ; trusted since: Wed, 04 Nov 2015 03:54:39 GMT
                    KEYDATA 20151105045507 20151205035507 19700101000000 257 3 5 (
                                (略)
                        ) ; KSK; alg = RSASHA1; key id = 36058
                        ; next refresh: Thu, 05 Nov 2015 04:55:07 GMT
                        ; trusted since: Sat, 05 Dec 2015 03:55:07 GMT
```



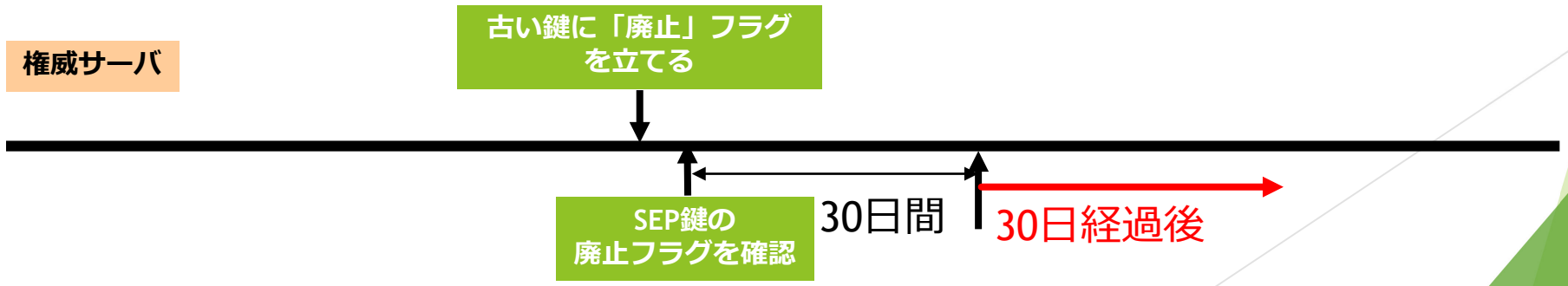
鍵の破棄が確認された状態：BIND

```
@                       IN SOA  . . (
                                    (略)
                                )
                        KEYDATA 20151105045507 20151104035439 19700101000000 257 3 5 (
                                (略)
                                ) ; revoked KSK; alg = RSASHA1; key id = 64617
                                ; trusted since: Wed, 04 Nov 2015 03:54:39 GMT
                                ; removal pending: Wed, 06 Jan 2016 04:11:14 GMT
                        KEYDATA 20151105045507 20151205035507 19700101000000 257 3 5 (
                                (略)
                                ) ; KSK; alg = RSASHA1; key id = 36058
                                ; next refresh: Thu, 05 Nov 2015 04:55:07 GMT
                                ; trusted since: Sat, 05 Dec 2015 03:55:07 GMT
```



鍵の破棄が承認された後
BIND

```
@                   IN SOA  . . (
                                (略)
                            )
                            KEYDATA 20151105045507 20151205035507 19700101000000 257 3 5 (
                            (略)
                            ) ; KSK; alg = RSASHA1; key id = 36058
                            ; next refresh: Thu, 05 Nov 2015 04:55:07 GMT
                            ; trusted since: Sat, 05 Dec 2015 03:55:07 GMT
```



Unbound の設定方法

- トラストアンカーのファイルを作成

```
.       3600    IN      DNSKEY  257 3 5 AwEAAcxDNuXWPybYlz5OvFFNSSZzC
                                        （中略）
                                        +X+SN/QiL0A4fmS0/0Fj9VbJP4Go2ExPL
```

- Unbound.conf 内で上記ファイルを auto-trust-anchor-file で指定
  - Unbound が鍵をサーバから取得すると、id や確認時間などの情報が上記ファイルに追加される

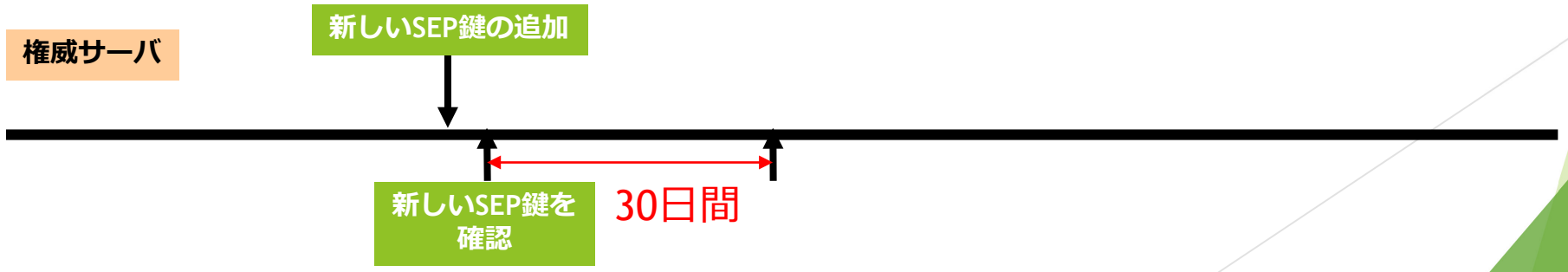
設定後の動き

- 設定ファイルがサーバから鍵を取得した際の情報を加えて更新される
- 鍵が trusted と認識された時間が記録される

```
; autotrust trust anchor file
;;id: . 1
;;last_queried: 1446533676 ;;Tue Nov  3 15:54:36 2015
;;last_success: 1446533676 ;;Tue Nov  3 15:54:36 2015
;;next_probe_time: 1446537078 ;;Tue Nov  3 16:51:18 2015
;;query_failed: 0
;;query_interval: 3600
;;retry_time: 3600
.       3600    IN      DNSKEY  257 3 5 AwEAAcx（中略）XWo2ExPL ;
        {id = 64489 (ksk), size = 1024b}
         ;;state=2 [  VALID  ] ;;count=0
         ;;lastchange=1446533466 ;;Tue Nov  3 15:51:06 2015
```

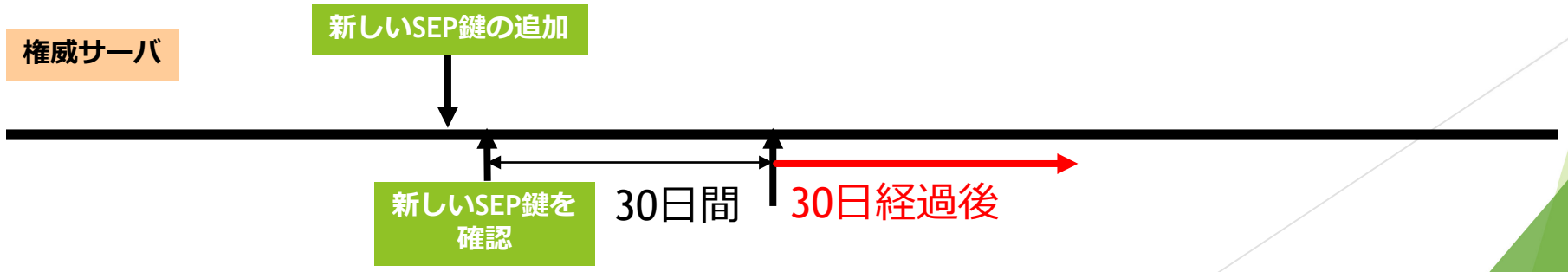
鍵の追加が確認された状態：Unbound

```
… ;;last_queried: 1446623358 ;;Wed Nov  4 16:49:18 2015
;;last_success: 1446623358 ;;Wed Nov  4 16:49:18 2015
;;next_probe_time: 1446626760 ;;Wed Nov  4 17:46:00 2015
;;query_failed: 0
;;query_interval: 3600
;;retry_time: 3600
.       3600    IN      DNSKEY  257 3 5 LhK（中略）HFx9
        ;{id = 36058 (ksk), size = 1024b}
        ;;state=1 [ ADDPEND ] ;;count=1 ;;lastchange=1446623358 ;;Wed Nov  4 16:49:18 2015
.       3600    IN      DNSKEY  257 3 5 Aw2（中略）ExPL
        ;{id = 64489 (ksk), size = 1024b}
        ;;state=2 [  VALID  ] ;;count=0 ;;lastchange=1446533466 ;;Tue Nov  3 15:51:06 2015
```



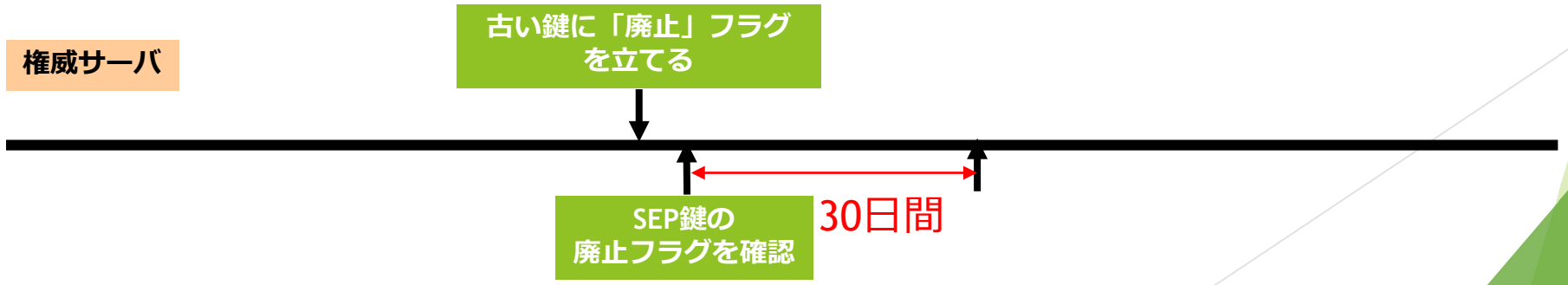
追加された鍵が承認された状態：Unbound

```
… ;;last_queried: 1449221408 ;;Fri Dec  4 18:30:08 2015
;;last_success: 1449221408 ;;Fri Dec  4 18:30:08 2015
;;next_probe_time: 1449224787 ;;Fri Dec  4 19:26:27 2015
;;query_failed: 0
;;query_interval: 3600
;;retry_time: 3600
.       3600    IN      DNSKEY  257 3 5 LhK（中略）HFx9
        ;{id = 36058 (ksk), size = 1024b}
        ;;state=1 [  VALID  ] ;;count=1 ;;lastchange=1449221408 ;;Fri Dec  4 18:30:08 2015
.       3600    IN      DNSKEY  257 3 5 Aw2（中略）ExPL
        ;{id = 64489 (ksk), size = 1024b}
        ;;state=2 [  VALID  ] ;;count=0 ;;lastchange=1446533466 ;;Tue Nov  3 15:51:06 2015
```



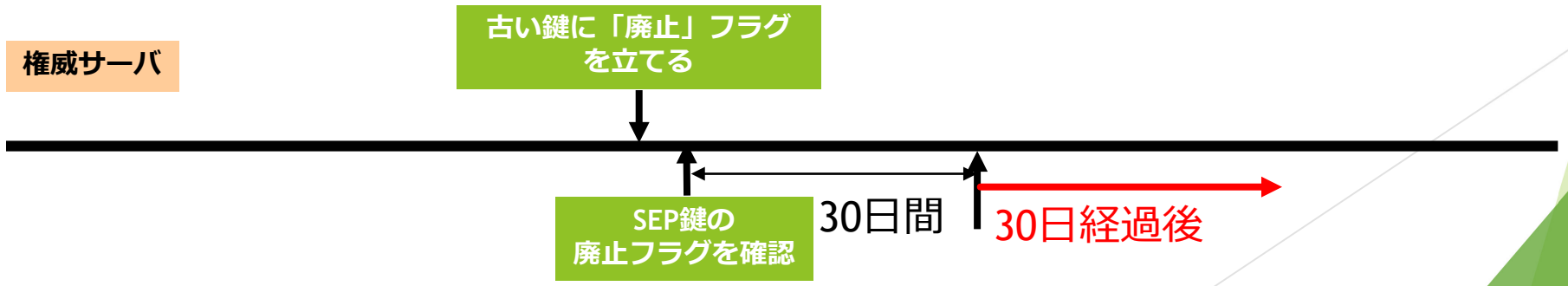
鍵の破棄が確認された状態：Unbound

```
… ;;last_queried: 1449307923 ;;Sat Dec  5 18:32:03 2015
;;last_success: 1449307923 ;;Sat Dec  5 18:32:03 2015
;;next_probe_time: 1449311205 ;;Sat Dec  5 19:26:45 2015
;;query_failed: 0
;;query_interval: 3600
;;retry_time: 3600
.       3600    IN      DNSKEY  257 3 5 LhK（中略）HFx9
        ;{id = 36058 (ksk), size = 1024b}
        ;;state=1 [  VALID  ] ;;count=1 ;;lastchange=1449221408 ;;Fri Dec  4 18:30:08 2015
.       3600    IN      DNSKEY  257 3 5 Aw2（中略）ExPL
        ;{id = 64617 (ksk), size = 1024b}
        ;;state=4 [ REVOKED ] ;;count=0 ;;lastchange=1449307923 ;;Sat Dec  5 18:32:03 2015
```



鍵の破棄が承認された後
Unbound

```
… ;;last_queried: 1452045559 ;;Wed Jan  6 10:59:19 2016
;;last_success: 1452045559 ;;Wed Jan  6 10:59:19 2016
;;next_probe_time: 1452049132 ;;Wed Jan  6 11:58:52 2016
;;query_failed: 0
;;query_interval: 3600
;;retry_time: 3600
.       3600    IN      DNSKEY  257 3 5 Aw2（中略）ExPL
        ;{id = 36058 (ksk), size = 1024b}
        ;;state=2 [  VALID  ] ;;count=0 ;;lastchange=1449221408 ;;Fri Dec  4 18:30:08 2015
```



起動時に新しい鍵があった場合

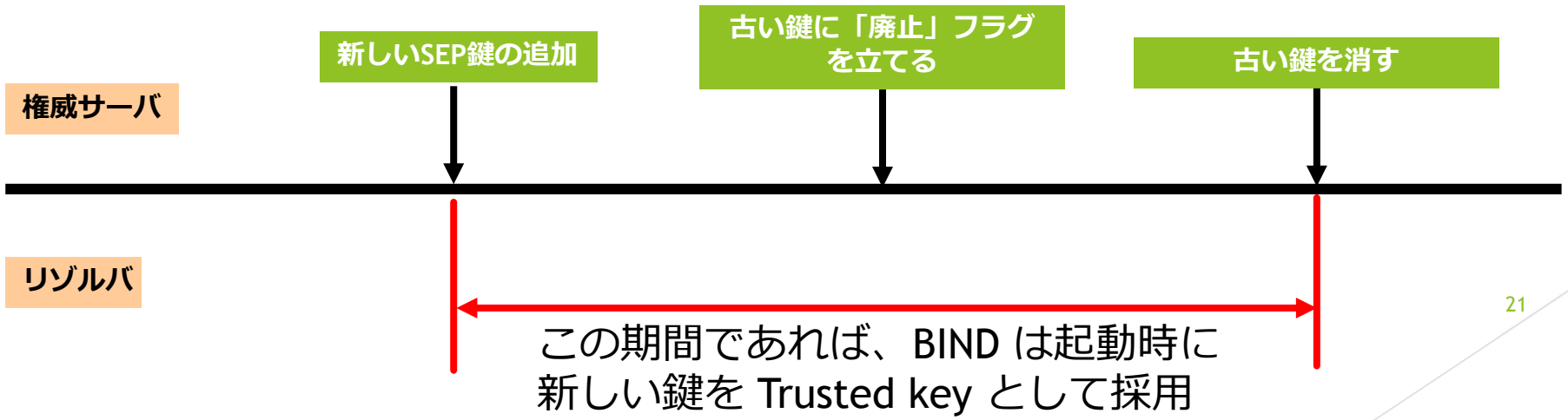
- 初期鍵を設定して起動した際に
  - すでに新しい鍵があった場合
  - 設定した鍵が REVOKE されていた場合
  - 設定した鍵がすでになかった場合

  にはどうなる？

- RFC5011 で適切に鍵交換がされている場合は確認した鍵が保存されるため、単にホストやプロセスを再起動したときの話ではない

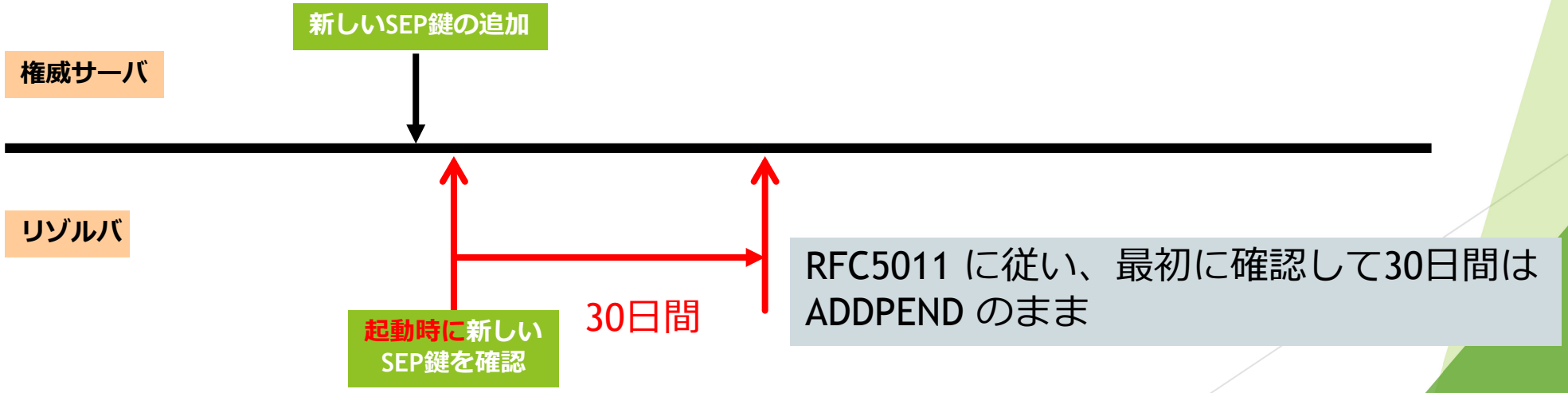
起動時に新しい鍵があった場合：BIND (9.9.8) の挙動

- 起動時にすでに新しい KSK が足されていると、その瞬間に trusted になる
- 検証側が古い鍵しかもっておらず、かつその鍵に REVOKE ビットが立っていても、その鍵で新しい鍵を検証して採用



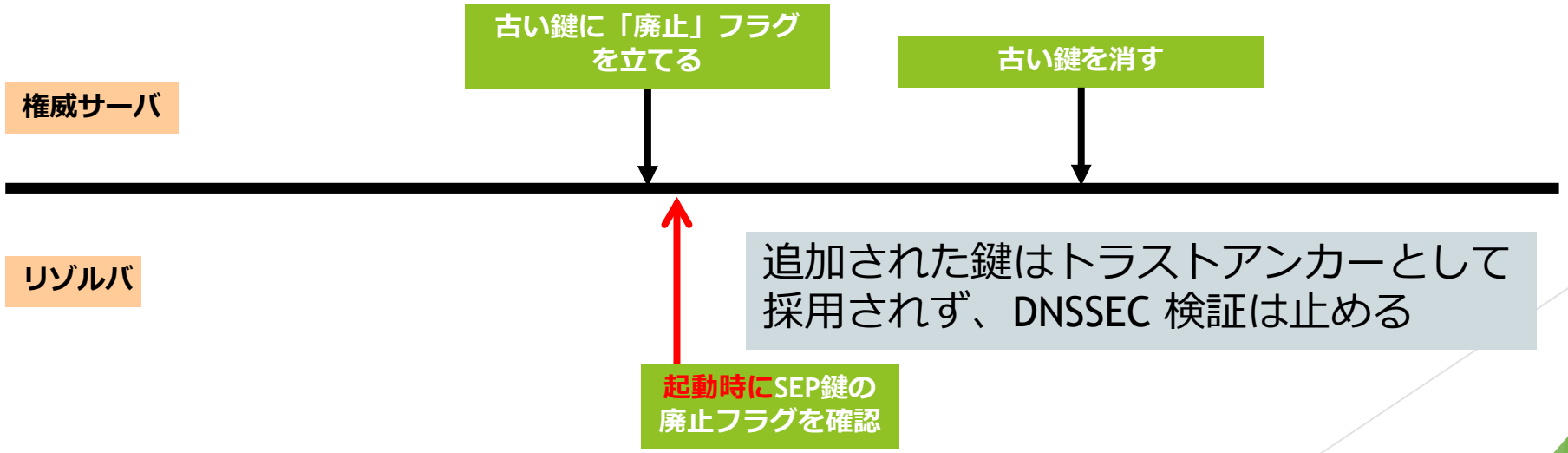
起動時に新しい鍵があった場合 unbound (1.5.6) の挙動（1）

- 起動時にすでに新しい KSK が足されていても、その瞬間には trusted にならない



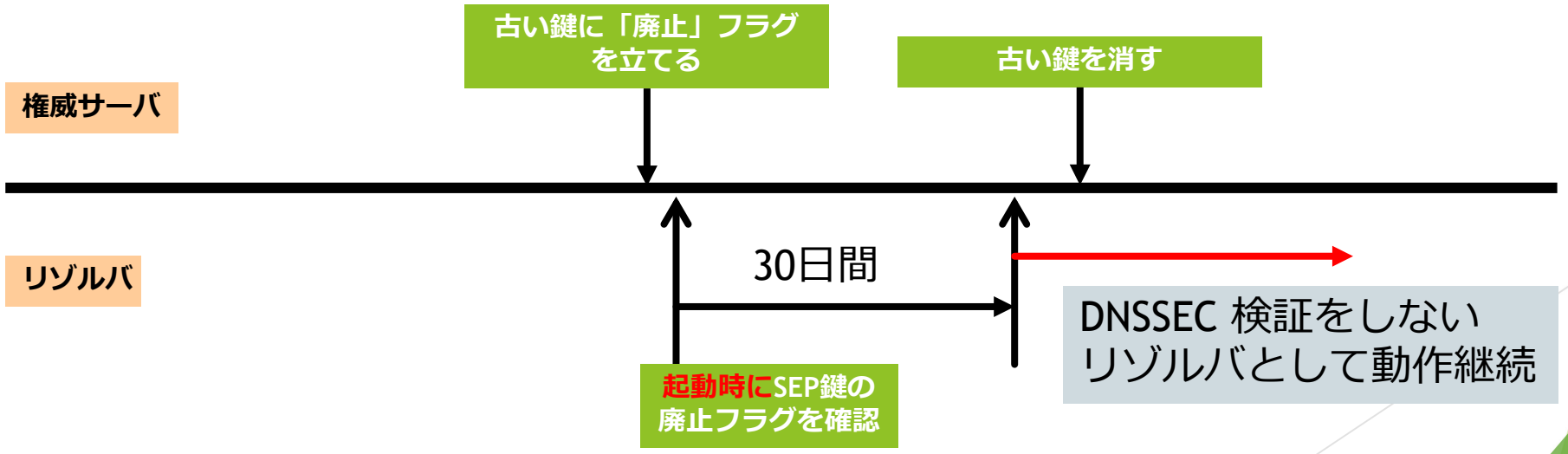
起動時に新しい鍵があった場合 unbound (1.5.6) の挙動（2）

- 所持しているトラストアンカーにサーバ側で REVOKE ビットが立っていた場合、その鍵を使って新しい鍵を取得しない
- 所持しているトラストアンカーを REVOKE 扱いにする
- この際、DO bit を出して問い合わせを出しても、AD bit 無しでスタブに返答が返される



起動時に新しい鍵があった場合 unbound (1.5.6) の挙動（3）

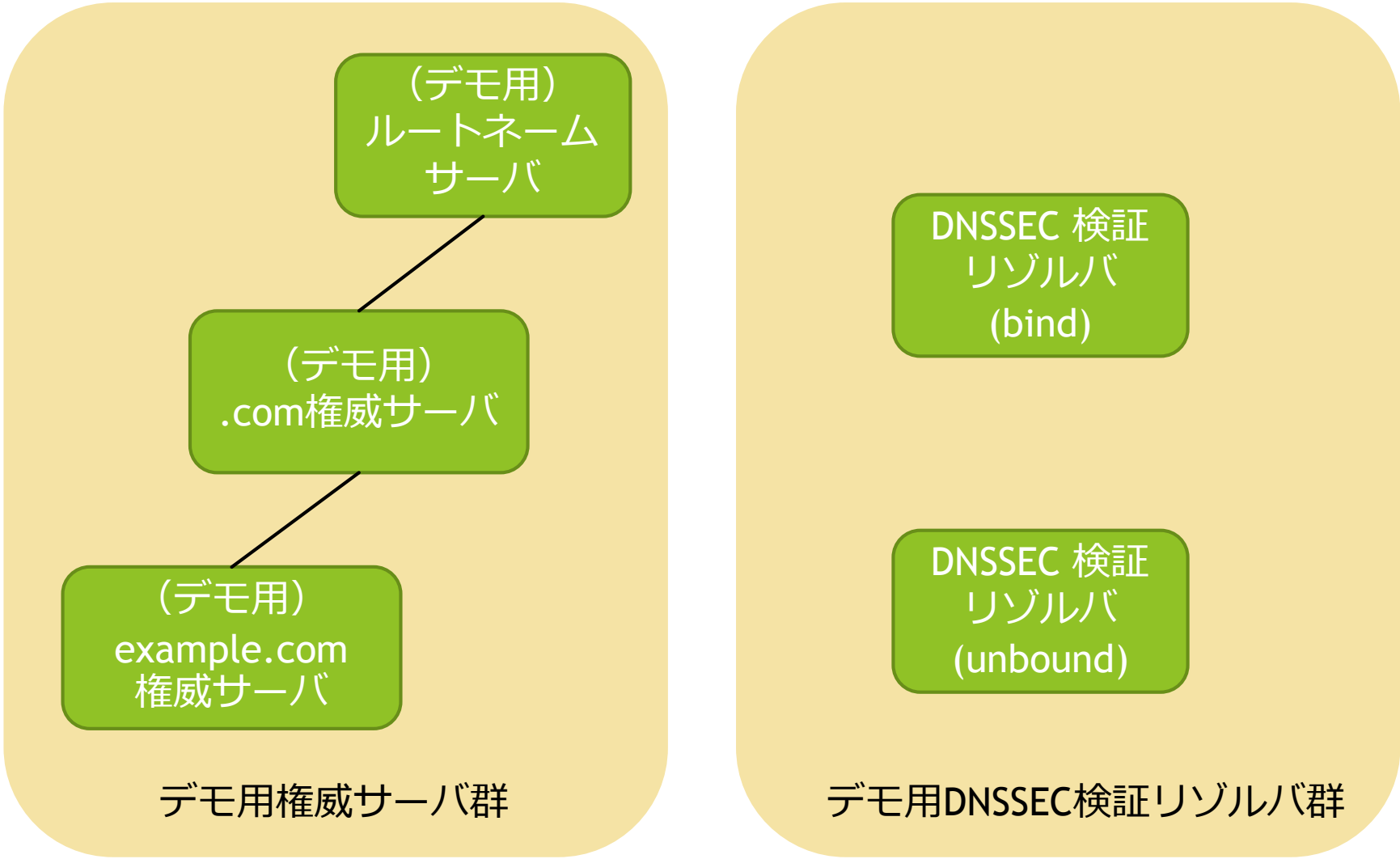
- その後、30日以上経過しても root.key は REVOKED のマークのまま保存
- DNSSEC 検証をしない状態で動作継続



Key Rollover のデモ

- 初期設定・DNSSEC 検証
- 新しい鍵の追加、各リゾルバの状態
- 30日経過後、各リゾルバの状態
- 鍵に REVOKE フラグを付与
- 鍵の削除を確認

デモ用構成


（デモ用）
ルートネーム
サーバ
（デモ用）
.com権威サーバ
（デモ用）
example.com
権威サーバ
デモ用権威サーバ群
DNSSEC 検証
リゾルバ
(bind)
DNSSEC 検証
リゾルバ
(unbound)
デモ用DNSSEC検証リゾルバ群

Key Rollover のデモ（再）

- 初期設定・DNSSEC 検証
- 新しい鍵の追加、各リゾルバの状態
- 30日経過後、各リゾルバの状態
- 鍵に REVOKE フラグを付与
- 鍵の削除を確認

実際の運用について

- とはいえ、いきなり使うのは心配！
  - 歴史上 Root KSK されたことはない！
  - RFC5011 自体、（ほぼ）まだ誰にも使われていない機構
  - 機構がそうなので、実装も枯れている状態とは言い難い
  - 各ステート遷移のタイマが非常に長いので、検証も苦労する
    - 今回は無理やり時間を進めて検証
    - 実環境とまったく同じ条件で検証となると３カ月もかかる！

→　手動と自動のハイブリッド

RFC5011 専用サーバの運用（主に ISP 向け）

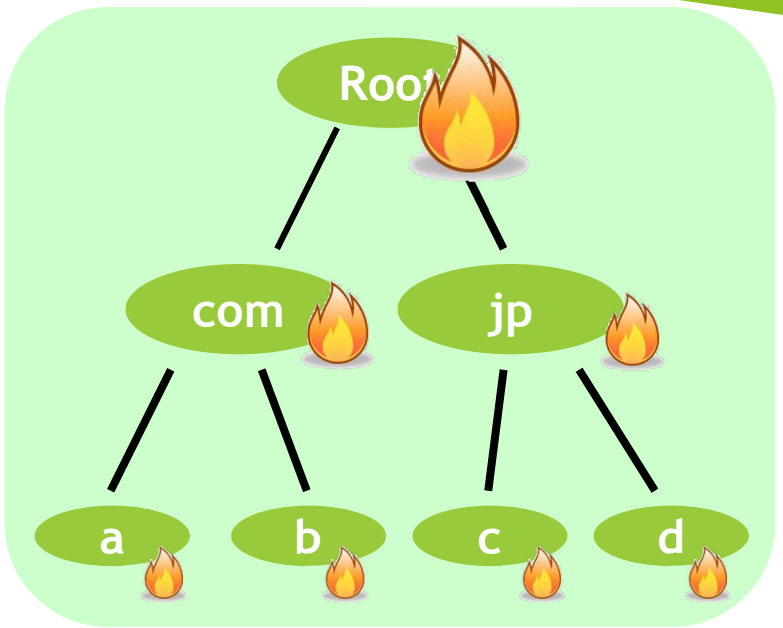
- RFC5011 の key rollover に特化したサーバを構築
  - ユーザにはサービス提供しない
- リゾルバサービスを提供するサーバには、 RFC5011 専用サーバで取得したトラストアンカーを手動で移植
- 何度か運用して、RFC5011 が大丈夫そうだと判断できたら本番環境に入れる？
  - でも、Root KSK Rollover の頻度はどれぐらいだろうか・・・
- ただし、DNS 検証サーバを積んでるネットワーク製品ベンダは RFC5011を採用してほしい
  - 期限切れトラストアンカーを持っているクライアントはルートネームサーバに大量のクエリを出す可能性が高い

注意点（ISP とベンダ向け）

- RFC5011 の仕様上、古い鍵を廃止して30日以上経過したらもう新しい鍵を検証できない
- 長い期間オンラインになっていない機器などは正しく DNSSEC 検証できない
  - 店舗在庫になっている機器等
- 正しく DNSSEC 検証できないと、（正規にキャッシュされないので）問い合わせが増加する
  - Root や TLD へのクエリが増加する

問題点

名前が ServFail エラーで引けないため、
ネガティブキャッシュをされず、
クライアントが再試行するたびにクエリが
発生する



権威サーバ

通常のリゾルバは正しく検証でき、
結果としてレコードもキャッシュ
されるため問い合わせは多くない

リゾルバ
リゾルバ

失効した鍵を持つ
検証サーバ

リゾルバ
リゾルバ
リゾルバ
リゾルバ

有効な鍵を持つ
検証サーバ

クライアント

対策

- 手動で入れ替えるしかありません
  - いくつかのアプライアンスは firmware update で対応
- が、自分自身で名前が引けないので別の Resolver があるネットワークでイメージをダウンロードする必要がある
- unbound-anchor はこの問題を回避するため、DNSSEC 検証をせずに名前解決をするようにできている（データ自体は手持ちの公開鍵で検証する）

まとめ

- Root KSK Rollover とは？
- 更新方法　：手動 or 自動
- 自動更新：　RFC5011: Automated Updates of DNS Security (DNSSEC) Trust Anchors “DNSSEC トラストアンカーの自動更新”
  - Bind での設定方法
  - Unbound での設定方法
- 注意点