# DNSSEC における鍵管理


平成 23 年 3 月

DNSSEC ジャパン

# DNSSEC における鍵管理

（目次）

# 1. はじめに

この資料は、DNSSEC における鍵管理の基本的なライフサイクルを説明し、ガイドラインを提供することを目的としている。対象読者としては、DNSSEC の導入を実施した、あるいは検討中の DNS サーバー管理者を想定している。従って DNS に関する知識と DNSSEC に関する基本的な理解を前提としている。ここでセキュリティ上の推奨慣行やガイドラインに言及するが、重要性の高いゾーン（TLD=トップレベルドメインや金融機関のサイト等）に対しては不充分な可能性があることに注意されたい。

この資料では、まず DNSSEC における暗号鍵の重要性と脅威である攻撃手法、そして使用される鍵の種類を説明したあと、鍵のライフサイクル各フェーズにおける考慮点と注意事項を解説する。

## 1.1. 注意事項

- 免責事項

本ドキュメントの内容は保証されたものではない。下記 Web サイトの免責事項を確認いただいた上で、本ドキュメントを使用されたい。

http://dnssec.jp/?page_id=16

- 問い合せ先

本ドキュメントに関する改善点等のコメントは下記事務局までご連絡ください。

**DNSSEC ジャパン事務局　＜sec@dnssec.jp＞**

## 1.2. 謝辞

本ホワイトペーパーを作成するに当たり、貴重な時間を割いてご協力いただきました以下の皆様に深く感謝いたします。

会社名(五十音順)

- 株式会社サリオンシステムズリサーチ
- 東京エレクトロン デバイス株式会社

# 2. DNSSEC における暗号鍵の重要性

DNSSEC では、暗号鍵を使用してリソースレコードに電子署名を付与する。クライアントにおいて取得したレコードの電子署名を検証することにより、DNS レスポンスの完全性をチェックし、改竄されていないことを確認できる。従って DNSSEC の信頼モデルはリソースレコードへの電子証明に使われる秘密鍵（private keys)に依存することになり、この信頼モデルの正当性を保証するためにはこれらの鍵が保護されている必要がある。秘密鍵が侵害された場合の影響として、主に 2 点が考えられる。

- 正規のゾーン管理者が鍵を使用できなくなる。
- 攻撃者がゾーン管理者になりすます。

前者はサービス拒否（DoS)攻撃となりうる。その結果、代わりの鍵を生成し、ゾーン情報を期限切れの前に再署名し、KSK の場合には上位ゾーンの DS レコードを更新する必

要が出てくる。対象ゾーンのセキュリティ・ポリシーにも依存するが、多大な時間と工数が要求される可能性がある。

後者では、攻撃者が対象ゾーンのなりすましを行い、虚偽のリソースレコードを作成し、下位ゾーンを捏造することが可能になる。その結果、DNS レスポンスの偽造が可能となり、そのゾーンのセキュリティレベルを、事実上署名のないゾーンと同じレベルに引き下げることになる。

## 3. DNSSEC の暗号鍵に対して考えられる攻撃内容と対策

DNSSEC で使われる秘密鍵を侵害しうる攻撃は複数の異なる手法が存在する。ここでは代表的な手法を簡単に紹介し、後述する対策技術への参照を提供する。

### 3.1. ブルートフォース攻撃

ブルートフォース攻撃は、可能性のある鍵を次々に試し、正しい鍵を見つけるまで続ける。ブルートフォース攻撃は多大な計算資源と時間を必要とするので、鍵長が適切であれば、通常実用的ではない。DNSSEC における鍵長の選択についての詳細は、鍵生成の項を参照のこと。

### 3.2. アルゴリズムの脆弱性

安全とされていたアルゴリズムが暗号技術研究の結果として脆弱であると証明されることがある。対処としては、新規システムにおいては脆弱とされたアルゴリズムを使用せず、既存システムにおいては早急に置き換えることが必要となる。DNSSEC の鍵と署名アルゴリズム選択については鍵生成の項を参照のこと。

### 3.3. 不充分な情報エントロピー

安全な鍵の条件は簡単に再生成できないことである。鍵長の長い暗号鍵の生成には情報エントロピー（ランダム性）の高品質な生成源が必要で、通常は専用の乱数発生器が使用される。もし充分なエントロピーが得られなければ、つまり乱数発生器が簡単に予測可能であれば、生成される鍵は簡単に再生成できることになる。予測可能な乱数生成器がインターネット上の多くのシステムに影響を与えた最近の事例として、Debian OpenSSL の 2008 年に発見された脆弱性がある[1]。DNSSEC の鍵生成における充分なエントロピーの確保については、鍵生成の項を参照のこと。

### 3.4. 物理的な攻撃や盗難

高品質な鍵をうまく生成できたとしても、どこかに保存する必要がある。リムーバブル・メディアに保存される場合、メディアが盗難にあう可能性があり、攻撃者にゾーンの鍵へのアクセスを与えることになる。ハードディスクに保存される場合、鍵は通常暗号化のうえ保存されるが、暗号操作（ゾーンへの署名など）を行う際には復号される必要があり、平文でメモリー内に存在することになる。従って DNSSEC の鍵を保存するサーバーに不正なアクセスがあれば、鍵が暗号化のうえ保存されていても攻撃者によって復元される可能性がある。鍵を盗難から守る方法については、鍵の保存の項を参照のこと。

### 3.5. 破壊

もし鍵が充分に保護されていても、攻撃やハードウェア障害あるいは天災などによって、保存メディアが破壊されるおそれがある。その結果、最善でも鍵の緊急ロールオーバーが必要になり、サービス停止を招くこともある。対策は単純で、鍵のバックアップとなる。ただし、適切な実装となると、多少考慮が必要になる。DNSSEC の鍵バックアップについては、鍵の保存の項を参照のこと。

## 4. DNSSEC における鍵の種類

DNSSEC 検証プロトコルで区別されているわけではないが、一般に受け入れられている慣行として、ゾーン署名に使う鍵（zone-signing keys, ZSK）と鍵の署名に使う鍵（key-signing key, KSK）を分けて用意する。ZSK はゾーン内のリソースレコードに署名するために用いられ、KSK は ZSK を含むゾーン頂点の鍵セットに署名するために用いられる。KSK を使って DS（Delegation Signer, 委任署名者）レコードを上位ゾーンに追加することで、信頼の連鎖を構築する。上位ゾーンの DS レコードが上位の ZSK で署名されていれば、その DS レコードが有効で、該当ゾーンの KSK を信頼できることが明らかになる。以下に KSK と ZSK の主な相違点を挙げる。

- KSK は鍵セットへの署名だけに用いられ、鍵セットは他の DNS レコードに比べて更新頻度が少ない。
- ZSK はインターネットに接続されたサーバーに保存され、頻繁に使用される。侵害の危険がより大きい。
- ZSK のロールオーバー（新しい鍵の生成とそれによるゾーンへの再署名）は上位での操作を必要としないが、KSK のロールオーバーは必要とする。つまり、上位ゾーンへの新たな KSK に対する DS レコードの追加が必要となる。

KSK と ZSK は若干異なるライフサイクルを持ち、二者のうちで KSK はより重要であり、侵害された場合の置き換えに困難が伴う。この資料では鍵への署名とゾーン署名には異なる鍵があることを前提としており、操作やガイドラインが鍵の種類によって異なるべき場合には明記する。

## 5. DNSSEC における鍵のライフサイクル

すべての暗号鍵と同様に DNSSEC の暗号鍵も、生成され、保存及び保護され、暗号操作実行に使用され、定期的に更新され、いつかは使われなくなれば失効処理および廃棄される必要がある。DNS が、複数ノードから構成される階層構造とキャッシュ機能を持つことから、DNSSEC における鍵の更新（鍵のロールオーバーと呼ばれる）はそれ自体相当に複雑なテーマとなる。この資料では鍵のロールオーバーに関する詳細には立ち入らない。鍵のロールオーバー操作に関する概要と説明については、RFC 4641 を参照されたい[2]。鍵の失効と廃棄に関しては、KSK がトラストアンカーとして使われる時に必

要となる。トラストアンカーの自動的な更新と失効についてもこの資料では扱わないので、それらの詳細は RFC 5011 を参照されたい[3]。

以下の項において、上記で言及した基本的な鍵のライフサイクルについて、より詳細を説明する。ライフサイクルの各フェーズについて、簡単な説明を行い、考えられる主な脅威を紹介したうえで、それら脅威を回避あるいは軽減するためのガイドラインを提案する。

## 5.1. 鍵生成

DNSSEC における鍵を生成する際には、主に以下 2 種類のパラメーターを考慮する必要がある。

- 暗号アルゴリズム
- 鍵長

### 5.1.1. 暗号アルゴリズム

DNSSEC のリソースレコードに署名を付与するための暗号アルゴリズムは、現在 2 種類が利用可能で、RSA と DSA がある。この 2 種類のうちでは、RSA がより歴史が長く、実装でもより多く使われている。DNSKEY レコードでは署名アルゴリズムも指定する[4]ので、DNS レコードに署名する際に使われるハッシュアルゴリズムも、鍵生成の際に指定しなければならない。現在の DNSSEC 仕様では、MD5、SHA-1、および SHA-2 のファミリー（SHA-224、SHA-256、SHA-384、SHA-512）を暗号学的ハッシュとしてサポートしている。MD-5 と SHA-1 に対しては有効な暗号解読攻撃が存在するため、SHA-2 アルゴリズムのいずれかを使用することが推奨される。新規の実装では、DNSSEC の署名アルゴリズムとしては RSA/SHA-256 を使うのが良い。

### 5.1.2. 鍵長

DNSSEC の署名に使う鍵長の選択には、鍵の種類と有効期間を考慮する必要がある。暗号解読の現状としては、6～7 年までの有効期間であれば、1024 ビットの鍵が安全に利用できると考えられる[5]。より重要な鍵、例えば TLD の KSK やロールオーバーが困難な鍵については、より長い鍵長が適する可能性がある。次によく使われている RSA の鍵長は 2048 ビットであり、重要度の高い鍵についてはこちらの使用を勧める。

### 5.1.3. 乱数発生器

安全な鍵の生成には、良いエントロピー源が必要となる。重要度の高い鍵に対しては、専用の乱数ハードウェアの使用が望ましい。例えば、HSM (ハードウェア・セキュリティ・モジュール)や TPM (トラステッド・プラットフォーム・モジュール)がある。その他の実装ではほとんどの場合、ソフトウェアの乱数発生器が代わりに使われる。最近の UNIX 系 OS では、/dev/random として提供されているものがある。DNSSEC の鍵生成ツール、例えば BIND の dnssec-keygen コマンドは、もし利用可能であれば/dev/random を利用して高品質な鍵を生成する機能を備えている。しかしもし充分なエントロピーが利用できない場合、鍵生成に多大な時間がかかる場合がある。より高速な/dev/urandom を使いたい場合もあ

るかもしれないが、/dev/urandom は良いエントロピー源ではなく、DNSSEC の鍵生成には使うべきではない。

#### 5.1.4. ソフトウェアの脆弱性

アルゴリズムや鍵長、そしてランダム性の発生源を慎重に選択したとしても、DNSSEC の鍵生成に使うソフトウェアの脆弱性が脆弱な鍵の生成につながることもある。DNSSEC の鍵生成の際には、既知の脆弱性が使用する暗号ライブラリ（代表的には OpenSSL）や鍵生成のユーティリティ（DNS サーバーソフトウェアとともに配布されているものなど）に存在しないことを確認すべきである。

### 5.2. 鍵の保存とアクセス

#### 5.2.1. 概要

BIND ネームサーバーで標準的に提供される DNSSEC のツールは、DNSSEC の鍵をサーバー上のファイルとして保存する。従って秘密鍵の保護も、サーバーへのアクセス権と鍵へのアクセスに使われるアカウントのみに依存する。鍵の重要度により、ファイルに保存された鍵の保護については異なる手段が考えられる。

- 鍵をサーバーに保存しない。鍵はリムーバブル・メディアに保存し、安全な場所（金庫など）に保管しておき、署名が必要なときのみ使用する。これは特に使用頻度の少ない KSK に有効である。
- 鍵をディスクに保存するのであれば、秘密鍵のファイルを保持するマシンをオフラインにする。ゾーンへの署名はそこで実施し、運用サーバーへ何らか安全な方法で転送する。注意点として、ゾーンファイルのマスターファイルも、オフラインのマシンで管理すべきである。ただしこの方法は、動的に更新されるべきゾーンに対しては現実的ではない。
- 鍵のファイルに適切なファイルパーミッションを設定する。適切な管理グループに属するユーザーのみが鍵のファイルにアクセスできることを確認する。もし可能であればファイルへのアクセスを監査する。

BIND の最近のバージョン（9.6 以降）やサードパーティのツール、例えば OpenDNSSEC では、専用の暗号モジュール、例えば HSM (ハードウェア・セキュリティ・モジュール)に DNSSEC の鍵を保存する機能を提供している。鍵をハードウェアに保存することには次のようなメリットがある。

- 鍵がハードウェアの中にのみ存在するため、仮に攻撃者がサーバーへ管理者権限でアクセス可能になったとしても、鍵を取り出して盗むことはできない。
- 暗号モジュールは通常物理的保護機能を持っており、仮に攻撃者が暗号モジュールへの物理的アクセスが可能になったとしても、鍵を侵害できない。
- 暗号モジュールには高品質の乱数発生器を備えているものがあり、高品質の鍵生成が保証される。

### 5.2.2. 暗号モジュール

重要度の高い鍵（例えば KSK）に対しては、暗号モジュールの利用を検討すべきであり、それによってソフトウェアだけでは達成できない鍵の保護レベルが得られる。暗号モジュールには複数の種類があり、提供するセキュリティのレベルが異なる。

- IC カード
- TPM (トラステッド・プラットフォーム・モジュール)
- HSM (ハードウェア・セキュリティ・モジュール)

暗号モジュールのセキュリティレベルは、米国立標準技術研究所が FIPS 140-2 として評価基準を定めており、認定制度 CMVP を運用している[6]。

IC カードは暗号処理機能を統合しており、秘密鍵の操作をカード上で実行できる。従って鍵がホスト上で平文の状態で存在することはなくなる。IC カードは基本的な侵害保護機能も提供しており、鍵を簡単に取り出したり改竄することはできない。鍵へのアクセスはパスフレーズで保護されているため、カードを手に入れても対応するパスフレーズを知っている場合のみ鍵を利用できる。IC カードの利点は入手しやすく、比較的安価で、交換も容易なところにある。物理的にサーバーから取り外し、安全な場所に保管し、署名が必要なときのみ使う事ができる。IC カードの主な注意点は、処理速度が遅いことである。このため、使用頻度が高かったり高いスループットが必要な場合には適さない。従って IC カードが最も適するのは、使用頻度が低く署名生成数も少ないシステムの KSK となる。

TPM は最近のコンピューターのマザーボードに搭載されていることが多い。機能としては、暗号鍵の安全な生成と、ハードウェアの乱数生成器を提供する。DNS サーバー機に搭載されている場合もあるので、DNSSEC の鍵の生成と保存に利用することは比較的容易な可能性がある。ただし TPM にアクセスするソフトフェアは、限られたプラットフォームでのみ提供されている（ほとんどが Microsoft Windows）ので、UNIX ベースの DNS サーバーでの利用可能性は限られる。また構成にもよるが、TPM に保持された鍵は取り出しできない場合があり、そうするとバックアップや別のサーバーへの移行が困難な場合がある。

HSM は、ほとんどの場合、プラグインするボード型か、単体のネットワーク接続機器として提供されている。通常高いセキュリティを持つ鍵ストレージと、ハードウェアの乱数生成器、そして IC カードや TPM と比べて極めて高い署名スループット性能を提供する。HSM 専用機が適するのは、複数の署名鍵を安全に管理しなければならず、使用頻度も高い場合だ。HSM の主な注意点は、コストが高いことと、導入や運用に必要となる工数である。DNSSEC 用に HSM を使うことで数々のセキュリティ上のメリットが得られるが、考

え得るリスクと(主に経済的な)影響を慎重に検討する必要がある。

## 5.3. 鍵のバックアップ

DNSSEC の鍵自体には価値はなく、何らかの場合で失われた場合には新しいものを簡単に生成できる。ZSK の場合にはほんの数分で実行できるだろうが、ただし KSK では相当に複雑かつ時間のかかる手順が必要な場合もある。重要な鍵を置き換える際の複雑な運用を回避するためには鍵のバックアップが推奨される。

鍵がファイルに保存されている場合には、バックアップ処理は鍵ファイルのコピーを作成することになる。重要なのは、バックアップメディアから鍵が流出することを避けるために、バックアップに対してもオリジナルの鍵ファイルと同等のセキュリティが求められることである。バックアップは安全な場所に保管されるべきであり、バックアップに対するアクセスは適切に監査され制御されなければならない。

HSM を使う場合には、適切な初期化に注意を払い、バックアップを可能にする。鍵のバックアップは、使用する HSM のバックアップ手順に準じ、暗号化されたファイルや IC カードに作成する。必要なセキュリティレベルを達成するため、鍵ファイルに対する充分強力なパスフレーズやバックアップカードに対する PIN を使用しなければならない。さらなる保護のため、鍵のファイルやバックアップカードは、安全な場所に保管する必要がある。

---


参照文献

1 http://www.debian.org/security/2008/dsa-1571
2 http://jprs.jp/tech/material/id/draft-ietf-dnsop-rfc4641bis-04-ja.txt
3 http://jprs.jp/tech/material/rfc/RFC5011-ja.txt
4 http://jprs.jp/tech/material/rfc/RFC4034-ja.txt
5 NIST Special Publication 800-131A http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf
6 http://csrc.nist.gov/groups/STM/cmvp/index.html