NIST
National Institute of
Standards and Technology
Technology Administration
U.S. Department of Commerce

Special Publication 800-81
国土安全保障省による後援

セキュアなドメインネームシステム（DNS）の導入ガイド

米国国立標準技術研究所による勧告

Ramaswamy Chandramouli
Scott Rose

この文書は下記団体によって翻訳監修されています
独立行政法人 情報処理推進機構
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN
NRI SECURE TECHNOLOGIES

NIST Special Publication 800-81

国土安全保障省による後援

セキュアなドメインネームシステム（DNS）の導入ガイド

米国国立標準技術研究所による勧告

Ramaswamy Chandramouli
Scott Rose

コンピュータセキュリティ

米国国立標準技術研究所
情報技術ラボラトリ
コンピュータセキュリティ部門
Gaithersburg, MD 20899-8930

2006 年 5 月

米国商務省　長官

Carlos M. Gutierrez

技術管理局　技術担当商務次官

Robert C. Cresanti

米国国立標準技術研究所　所長

William Jeffrey

コンピュータシステム技術に関する報告書

米国国立標準技術研究所（NIST: National Institute of Standards and Technology、以下、NIST と称す）の情報技術ラボラトリ（ITL:Information Technology Laboratory）は、国家の測定および標準に関する基盤において技術的リーダーシップを提供することにより、米国の経済と公共福祉に貢献している。情報技術ラボラトリは、テストの実施、テスト技法の開発、参照データの作成、実装によるコンセプト実証、技術的分析を通じて、情報技術の開発と生産的利用の発展に努めている。情報技術ラボラトリの責務は、連邦政府の情報システムにおいて、費用対効果の高いセキュリティを施し、国家安全保障にかかわらない情報のプライバシーを確保するための、技術的、物理的、管理的および運用のための標準とガイドラインを策定することにある。NIST Special Publication 800 シリーズでは、情報システムセキュリティにおける情報技術ラボラトリの調査、ガイドライン、普及活動ならびに産業界、政府機関および教育機関との共同活動について報告する。

この文書中で特定される商業的組織、装置、資料は、実験手順または概念を適切に説明するためのものである。したがって、NIST による推薦または保証を意味するものではなく、これらの組織、資料、または装置が、その目的に関して得られる最善のものであると意味しているわけでもない。

謝辞

本書執筆陣である Ramaswamy Chandramouli、Scott Rose（ともに NIST）は、この文書の草稿をレビューしてくれた同僚に対し感謝の意を表したい。参考になるご意見をくださり、この文書で引用している文書のいくつかを紹介してくださった、政府 DNSSEC 作業部会（Government DNSSEC Working Group）のメンバーにも深く感謝する。また、Tim Grance（NIST、コンピュータセキュリティ部門）および Doug Montgomery（NIST、高度ネットワーク技術部門）によるこのプロジェクトを通じてのリーダーシップと指導にも感謝する。本書の執筆を後援してくれた Douglas Maughan（国土安全保障省）に特別に感謝の意を表する。

目次

付録

図

表

要旨

インターネットは、何億人ものユーザが利用する世界最大のコンピュータネットワークである。ユーザ側からみると、このネットワーク上のノードやリソースはそれぞれ一意の名前、すなわち www.nist.gov などのドメイン名によって識別される。しかし、インターネット上で通信パケットを配送するネットワーク機器の側からみると、リソースを表す一意の識別子は、172.30.128.27 のようなインターネットプロトコル（IP：Internet Protocol）アドレスである。インターネットリソースに、IP アドレスではなくユーザにとって使い勝手のよいドメイン名でアクセスするには、ドメイン名から IP アドレスへの変換と、その逆の変換を行う仕組みが必要である。この変換処理が、ドメインネームシステム（DNS：Domain Name System）の主な仕事である。

DNS のインフラストラクチャは、地理的に世界中に分散している、計算処理と通信を行うエンティティから成る。そこには、.gov や.com などの 250 のトップレベルドメインと、nist.gov や ietf.org などの数百万の第 2 レベルドメインが存在する。これに応じて、DNS インフラストラクチャには多数のネームサーバがあり、各ネームサーバにはドメイン名空間のごく一部に関する情報が格納されている。DNS インフラストラクチャは、関与するさまざまな要素の間で連携をとることによって機能する。DNS によって提供されるドメイン名データは、インターネット上のあらゆるコンピュータがどの場所からでも利用できることを目的としている。

この文書は、組織内において DNS のセキュリティを保護するための導入ガイドラインである。DNS データは、公開されることが前提であるため、一般からアクセス可能な IT リソースに関する DNS データの機密性は、セキュリティ目標にはならない。DNS の主なセキュリティ目標は、データの完全性維持と情報源の真正性確認を実現することであり、これはドメイン名情報の真正性の保証と、伝送中のドメイン名情報の完全性維持のために必要となる。この文書では、データの完全性維持と情報源の真正性確認に関するさまざまな指針を示す。また、DNS サービスとデータの可用性も非常に重要である。DNS の構成要素はしばしば、その DNS 構成要素が処理するドメイン名を持つリソースへのアクセスを妨害することを狙ったサービス運用妨害攻撃を受ける。この文書では、さまざまな DNS 構成要素の脆弱性を悪用する数々のサービス運用妨害攻撃を防ぐための、DNS 導入の際の設定に関するガイドラインを示す。

DNS は、他のあらゆる分散型コンピュータシステムと同じタイプの脆弱性（プラットフォームレベル、ソフトウェアレベル、およびネットワークレベル）の影響を受けやすい。しかし、DNS は世界規模のインターネットの基盤システムであるため、多くの分散型コンピュータシステムにはみられない次のような特性がある。

+ 明確なシステム境界がない。参加するエンティティは、地理上またはネットワークトポロジ上の境界ルールの制約を受けない。

+ データの機密性保持の必要がない。データは、地域や所属を問わず、あらゆるエンティティがアクセス可能でなければならない。

このような特性のため、なりすましやメッセージの改ざんといった従来のネットワークレベルの攻撃や、ホストで管理され、配布されるデータの完全性の侵害は、次のようにまったく異なる機能的影響をもたらす。

+ DNS ノードの身元識別情報を偽装したなりすましノードは、そのノードが情報を提供するインターネットリソース群（すなわち、そのノードがサービスを提供するドメイン）のサービスへのアクセスを拒否できる。この拒否は、一定のクライアントだけでなく、それらのリソースにアクセスする必要のあるすべてのクライアントを含む世界全体が対象となる。

+ なりすましや侵入者によって偽装された DNS 情報により、その DNS 情報の一部を提供する DNS ノード（つまり、組織のユーザにインターネットアクセスサービスを提供するネームサーバ）の情報キャッシュが汚染され、それが結果として、その DNS ノードによるホストの名前解決によってアクセス可能となるリソースに対するサービス運用妨害となる。

+ 信頼できる出所にある DNS 情報や、いくつかの過去のクエリから情報を蓄積している中間の情報キャッシュにある DNS 情報の完全性が損なわれた場合、連鎖的な DNS 情報取得プロセスが損なわれる可能性がある。この結果、DNS の名前解決機能に対するサービス運用妨害、あるいは有害で違法なリソースへのユーザの誘導につながる可能性がある。

+ DNS システムによってホストされている名前解決データが DNS 規格で定義されているデータ内容の要件に従っていないと、DNS システムの負荷が高まったり、インターネットリソースへのサービス運用妨害につながる古いデータを提供したりするなど、悪影響が生じることが考えられる。ほとんどのソフトウェアでは、プログラムデータが独立であることにより（通常のデータベース管理システムのように）、誤ったデータによる悪影響に対して、ある程度のバッファができる。DNS の場合は、データ内容によってシステム全体の完全性が左右される。

こうした機能的影響に基づき、この文書で示すセキュアな DNS のための導入ガイドラインは、次の一般的な推奨事項と DNS 固有の推奨事項で構成されている。

+ オペレーティングシステム、アプリケーションパッチ、プロセス隔離、ネットワークフォールトトレランスなど、DNS ホスティング環境をセキュリティ保護するための適切なシステムおよびネットワークセキュリティ管理策を実施すること。

+ DNS の名前解決データの更新やデータの複製など、組織の管理下にある DNS ノードが関与する DNS トランザクションを保護すること。IETF（Internet Engineering Task Force）の TSIG（Transaction Signature）仕様に沿って、共有秘密に基づくハッシュベースのメッセージ認証コードを使用してトランザクションを保護すること。

+ 世界規模のインターネット上の任意の DNS ノードが関与する可能性のある、随所で発生する DNS のクエリ／レスポンスのトランザクションは、IETF の DNSSEC（Domain Name System Security Extensions）仕様に沿って、非対称暗号に基づくディジタル署名を使用して保護すること。

+ DNS システムのパフォーマンスと完全性維持のバランスを適切に保つことが可能な完全性制約を使用して、DNS の名前解決データの内容を規制すること。

この文書では、DNS ネームサーバのセキュリティ保護のための推奨事項を示す。推奨事項の一部は、ゾーン情報のための DNSSEC（DNS Security Extensions）の導入に関するものである。セキュリティのこの部分を実現するための基本手順は、以下の通りである。

+ DNSSEC 対応ネームサーバをインストールする（7.2.1 項参照）

+ 完全性エラーがないか、ゾーンファイルをチェックする（セクション 10 参照）

+ ゾーンごとに非対称鍵のペアを生成し、それらをゾーンファイルに含める（9.2 項および 9.3 項参照）

+ ゾーンに署名する（9.6 項参照）

+ 署名付きゾーンをサーバにロードする

+ DNSSEC 処理を行うようにネームサーバを設定する（9.1 項参照）

+ （任意で）セキュアな委任のために公開鍵のコピーを親に送信する。

このガイドの大部分は、権威ネームサーバに重点を置いているが、DNS キャッシュサーバのための DNSSEC 導入の基本手順は、以下の通りである。

+ DNSSEC 対応リゾルバの実装をインストールする（7.2.1 項参照）
+ 管理者が有効性確認を必要としているゾーンに対して、1 つまたは複数のトラストアンカを取得する（9.7 項参照）
+ DNSSEC 処理を行うようにリゾルバを設定する（9.1 項参照）

残りのセクションでは、ネームサーバのセキュアな設定と運用のための推奨事項を取り上げる。

（本ページは意図的に白紙のままとする）

1. はじめに

1.1 作成機関

米国国立標準技術研究所(NIST: National Institute of Standards and Technology、以下 NIST と称する)は、2002 年施行の連邦情報セキュリティマネジメント法(FISMA: Federal Information Security Management Act、以下 FISMA と称す)、公法 107-347 に基づくその法的責任を推進するために、この文書を作成した。

NIST は、すべての連邦政府機関の運営および資産に適切な情報セキュリティをもたらすために、最低限の要求事項を含んだ標準およびガイドラインを作成する責務があるが、このような標準およびガイドラインは国家安全保障にかかわるシステムには適用されない。このガイドラインは、行政管理予算局(OMB; Office of Management and Budget)Circular A-130、第 8b(3)項、『政府機関の情報システムの保護(Securing Agency Information Systems)』の要求事項に一致しており、これは A-130 の付録 IV「重要部門の分析」で分析されているとおりである。補足情報は、A-130、付録 III に記載されている。

このガイドラインは連邦政府機関が使用する目的で作成されている。非政府組織が自由意志で使用することもでき、著作権の制約はない(翻訳者注:著作権に関するこの記述は、SP800-81 の英語の原文のことを言っており、日本語へ翻訳した本書の著作権は、独立行政法人情報処理推進機構および NRI セキュアテクノロジーズ株式会社に帰属する)。

本文書における一切は、商務長官が法的権威に基づき連邦政府に対して義務および拘束力を与えた標準および指針を否定するものではない。また、これらの指針は、商務長官、行政管理予算局長、または他のすべての連邦政府当局者の既存の権威に変更を加えたり、これらに取って代わるものと解釈してはならない。

1.2 目的と範囲

この文書は、組織におけるドメインネームシステム(DNS:Domain Name System)サービスの安全な導入について理解を深められるよう、組織を支援することを目的としている。組織内の DNS の各種側面のセキュリティ保護について、運用環境および関連する脅威の分析結果に基づく実践的な指針を示す。

現時点では、DNS はほとんどの攻撃の主たる標的ではないが、ホストでのセキュリティ意識が高まり、アプリケーションがネットワーク操作において DNS インフラストラクチャを頼りにし始めるにつれて、DNS インフラストラクチャがより魅力的な標的となることが考えられる。DNSSEC の最終的な目標は、インフラストラクチャ側でドメインツリー全体に全面的に DNSSEC が導入されることと、DNSSEC によって提供されるサービスを要求する可能性のあるアプリケーションに実装されることである。現在、DNS ドメインツリーにおいて DNSSEC 機能を提供する運用ノードはない。したがって全面導入に向けての第 1 歩は、高いセキュリティが要求されるドメインサブツリーに対して DNSSEC 機能を提供することである。DNS インフラストラクチャにおいて DNSSEC 機能が広く普及すれば、アプリケーション開発者は DNSSEC に対応したアプリケーションを開発し、DNSSEC をネットワークセキュリティの手段として使用できるようになる。

この文書では、DNSSEC の導入は、個々のホストではなく DNS インフラストラクチャを対象としている。しかし、インフラストラクチャのセキュリティが高まるにつれて、必然的に、DNSSEC は DNS クエリを行う個々のクライアントにまで適用されるようになる。DNSSEC は、下位互換性を考慮して設計されているため、現行のネットワークアプリケーションは、上流のサーバが DNSSEC を使用していれば、DNSSEC によるセキュリティを得ることができるが、将来はすべてのシステム(DNS ネームサ

ーバとクライアント）が DNSSEC 仕様とこの文書で解説する動作の少なくとも一部は実行できるようになることが望ましい。

1.3 対象とする読者

この文書は、DNS 導入の管理者、および DNS 関連業務に関して責任を持つコンピュータセキュリティ担当者とシステム管理者を対象に作成された。

1.4 構成

この文書のこれ以降の内容は、次の 10 セクションで構成されている。

+ セクション 2 では、DNS および DNS インフラストラクチャの概要を説明する。また、DNS のセキュリティ目標についても説明し、この文書で取り上げる DNS の各種側面の概略も述べる。
+ セクション 3 では、DNS データを保持するゾーンファイル、DNS サービスを提供するネームサーバとリゾルバなど、DNS の基本的な構成要素をいくつか紹介する。
+ セクション 4 では、各種の DNS トランザクションを明確に定める。
+ セクション 5 では、DNS ホスティング環境にかかわる脅威、セキュリティ目標、保護策について述べる。セクション 6 では、DNS トランザクションについて同種の情報を提供する。
+ セクション 7 および 8 では、DNS ホスティング環境と DNS トランザクション（DNS クエリ／レスポンスを除く）のセキュリティ保護のガイドラインを示す。
+ セクション 9 では、DNS のクエリ／レスポンスのトランザクションのセキュリティ保護のための推奨事項を示す。
+ セクション 10 では、DNS を通じた情報の露出を最小限に抑えるための指針に重点を置く。
+ セクション 11 では、DNS セキュリティ管理業務のガイドラインを示す。

また、付録には参考情報を掲載している。付録 A には、重要な DNS セキュリティ用語の定義を掲載している。付録 B および C には、それぞれ略語一覧と参考文献を掲載している。

2. ドメインネームシステムのセキュリティ保護

この文書では、組織におけるドメインネームシステム（DNS：Domain Name System）のセキュリティを保護するための導入ガイドラインを示す。導入ガイドラインは、すべての DNS 構成要素についてセキュリティ目標と保護策を分析した結果から得られたものである。セキュリティ目標の根拠と導入ガイドライン策定の方法は、次のとおりである。

+ 各 DNS 構成要素のセキュリティ目標は、運用環境および関連する脅威の分析に基づいて策定される。
+ 各 DNS 構成要素を安全に導入するためのガイドラインは、ポリシーまたはベストプラクティスに基づく構成オプションとチェックリストの組み合わせを通じて提供される。

この文書で導入ガイドラインを示すために使用した主なセキュリティ仕様（および関連する仕組み）は、次のとおりである。

+ RFC（Request for Comments）4033、4034、4035 および 3833[6,7,8,1][1]に規定されている Internet Engineering Task Force（IETF）Domain Name System Security Extensions（DNSSEC）仕様。
+ RFC 2845 および 3007 [12,9][2]で規定されている IETF Transaction Signature（TSIG）仕様。

2.1 ドメインネームシステム（DNS）とは何か

インターネットは、5 億 8 千万人ものユーザを抱える世界最大のコンピュータネットワークである。ユーザ側からみると、このネットワーク上のノードやリソースはそれぞれ一意の名前、すなわちドメイン名によって識別される。インターネットリソースの例をいくつか示す。

+ Web サーバ。Web サイトへのアクセス用
+ メールサーバ。電子メールメッセージの配信用
+ アプリケーションサーバ。ソフトウェアシステムとデータベースへのリモートからのアクセス用

しかし、インターネット上で通信パケットを配送するネットワーク機器（ルータなど）の側からみると、一意のリソース識別子は、ドットで区切られた 4 つの数字の並び（たとえば 123.67.43.254）として表されるインターネットプロトコル（IP）アドレスである。インターネットリソースに、IP アドレスではなくユーザにとって使い勝手のよいドメイン名でアクセスするには、ドメイン名から IP アドレスへの変換と、その逆の変換を行う仕組みが必要である。この変換処理が、*ドメインネームシステム*（DNS：*Domain Name System*）と呼ばれるエンジンの主な仕事である。

ユーザがクライアントプログラムやユーザプログラム（たとえば Web ブラウザ）を使用してインターネットリソース（たとえば Web サーバ）にアクセスするには、ドメイン名を入力する。Web サーバに接続して該当する Web ページを取得するには、ブラウザは Web サーバのドメイン名に対応する IP アドレスを必要とする。ブラウザは、DNS にこの情報を要求する。ドメイン名と IP アドレスを対応付ける

---

[1] RFC 4033、『*DNS Security Introduction and Requirements*』は、http://www.ietf.org/rfc/rfc4033.txt.から入手できる。RFC 4034、『*Resource Records for the DNS Security Extensions*』は http://www.ietf.org/rfc/rfc4034.txtから入手できる。RFC 4035、『*Personnel Modifications for the DNS Security Extensions*』は http://www.ietf.org/rfc/rfc4035.txtから入手できる。RFC 3833、『*Threat Analysis of the Domain Name System (DNS)*』は、http://www.ietf.org/rfc/rfc3833.txtから入手できる。

[2] RFC 2845、『*Secret Key Transaction Authentication for DNS (TSIG)*』は http://www.ietf.org/rfc/rfc2845.txtから入手できる。RFC 3007、『*Secure Domain Name System (DNS) Dynamic Update*』は、http://www.ietf.org/rfc/rfc3007.txtから入手できる。

この機能は、*名前解決*と呼ばれる。DNS が名前解決機能を遂行するために使用するプロトコルは、DNS プロトコルと呼ばれる。

上述の DNS 機能は、次の基本要素で構成される。1 つは、ドメイン名とその IP アドレスを格納するためのデータリポジトリである。ドメイン名の数は膨大であるため、拡張性とパフォーマンスを考慮すると、リポジトリは分散させるべきだということがわかる。ドメイン名は、場合によってはフォールトトレランスのために複製する必要もある。2 つめは、このリポジトリを管理し、名前解決機能を提供するソフトウェアである。この 2 つの機能（ドメイン名リポジトリの管理と名前解決サービスの提供）は、DNS の主たる構成要素である*ネーム サーバ*によって提供される。ネームサーバには多数のカテゴリがあり、提供するデータや実行する機能の種類に応じて区別される。DNS ネームサーバによって提供されるサービスにユーザプログラムに代わってアクセスするために、*リゾルバ*と呼ばれるもう 1 つの構成要素がある。リゾルバは、その機能性に応じて大きく 2 つに分類される（キャッシング／再帰的／リゾルビィングネームサーバ(Resolving Name Server)と、スタブ（キャッシングを行わない）リゾルバ）[3]。通信プロトコル、各種の DNS 構成要素、構成要素の設定に適用するポリシー、ドメイン名の作成、格納および使用の手順が DNS インフラストラクチャを形成する。

2.2　DNS インフラストラクチャ

DNS のインフラストラクチャは、地理的に世界中に分散している、計算処理と通信を行うエンティティから成る。この DNS インフラストラクチャを理解するためには、まず、ドメイン名の編成の背後にある構造を見てみる必要がある。ドメイン名空間（すべてのドメイン名で構成される空間）は、階層構造に編成されている。階層の最上位レベルは*ルートドメイン*であり、ドット（“.”）で表される。階層の次のレベルは、*トップレベルドメイン*（TLD：Top Leve Domain、以下 TLD と称す）と呼ばれる。ルートドメインは 1 つしかないが、TLD は多数存在する。TLD はそれぞれ、ルートドメインの子ドメインと呼ばれる。この文脈では、ルートドメインは TLD よりも 1 つ上のレベルにあるという意味で親ドメインとなる。さらに TLD もそれぞれ複数の子ドメインを持つことができる。TLD の子は*第 2 レベルドメイン*、または*エンタープライズレベルドメイン*と呼ばれる。

ドメイン名表記では、ルートドメインの記号は通常は省略する。たとえば、marketing.example.com というドメイン名を例にとって考える。このドメイン名の一番右のラベル（“com”）は、TLD である。その左のラベル（“example”）は、第 2 レベルドメイン（エンタープライズドメイン）である。一番左のラベル（“marketing”）は、第 3 レベルドメインである。第 4 レベルドメイン、第 5 レベルドメインのように続けることもできる。marketing.example.com の個々のラベルはドメイン（TLD、第 2 レベルドメイン、第 3 レベルドメインなど）と呼ばれることから、現在のレベルから TLD までのすべてのラベルを列挙したものは、*完全修飾ドメイン名*（FQDN：Fully Qualified Domain Name、以下 FQDN と称す）と呼ばれる。だたしこの文書では、FQDN を指すときは単に「ドメイン名」と表記し、個々のラベルを識別するときは、レベル名を使用する。

ルートドメインは、1 つだけである。TLD は 250 以上あり、次の 3 種類に分類される。

+ **国コード TLD（ccTLD：Country-code TLD）**。国と地域に関連付けられるドメイン。240 以上の ccTLD が存在する。例：.uk、.in、.jp など。
+ **スポンサー付き一般 TLD（gTLD：generic TLD）**。特定の利益共同体を表すスポンサー付きの特殊なドメイン。例：.edu、.gov、.int、.mil、.aero、.coop、.museum。
+ **スポンサーなし一般 TLD（gTLD）**。後援組織のないドメイン。
例：.com、.net、.org、.biz、.info、.name、.pro。

[3] キャッシング／再帰的／レゾルビィングネームサーバは、リゾルバとネームサーバの両方の役割を果たすので、リゾルバに分類される。

数百万のエンタープライズレベル（第 2 レベルドメインまたはそれ以下）ドメインが存在する。実際、2004 年 9 月時点で 6 千万を超えるドメイン名が登録されていた。DNS 名前空間階層の一部を図 2-1 に示す。

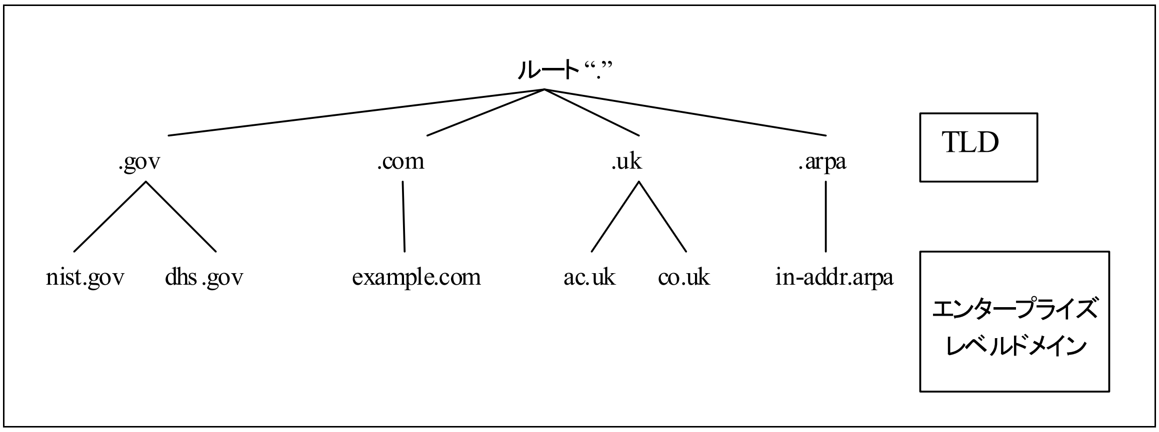

図 2-1. DNS 名前空間の階層の一部

DNS インフラストラクチャには、多数のネームサーバが存在する。各ネームサーバには、ドメイン名空間の一部に関する情報が格納されている。ネームサーバは、ドメイン名空間の最初の 3 つのレベルの範囲まではレベルと関連付けられている。ルートレベルには、13 のネームサーバが関連付けられており、これらは*ルートサーバ*と呼ばれる。現在 2 つのルートサーバが合衆国の民間企業 VeriSign によって運用されている。残りは、世界中のほかの組織によって、インターネットコミュニティへのサービスとして運用されている。TLD に関連付けられたネームサーバを運用する組織は、*レジストリ*と呼ばれる。一般に、ccTLD はそれぞれの国の指定レジストリによって運用され、gTLD はグローバルなレジストリによって運用されている。たとえば、VeriSign は現在.com と.net の各 TLD のネームサーバを管理している。非営利組織の PIR（Public Internet Registry）は、.org TLD のネームサーバを管理している。またもう 1 つの非営利組織の EDUCAUSE は、.edu TLD のネームサーバを管理している。ただしこれらのレジストリ組織はすべて変わる可能性もある。ドメイン登録者は、該当するレジストリの代表窓口（レジストラと呼ばれる）を把握しているべきである。エンタープライズレベルおよびそれよりも下位のドメインに関連付けられているネームサーバは、そのドメインを所有する組織によって直接運用されているか、あるいはインターネットサービスプロバイダ（ISP）などのサービスプロバイダに運用が委託されている。

DNS インフラストラクチャは、関与するさまざまな主体（ルートサーバを管理する組織、TLD を運用するレジストリなど）のあいだの連携によって機能している。非営利の民間法人である Internet Corporation for Assigned Names and Numbers（ICANN）は、DNS のさまざまな側面の技術的な調整役の役割を果たしている。たとえば、ICANN はルートサーバの管理のためのポリシーを策定している。ICANN はまた、新しい TLD の作成権者でもある。ICANN は 1998 年、米国商務省によって設立された。

ドメイン名（TLD の下のエンタープライズレベルドメインに限る）の登録を希望するユーザ（個人または法人）は、レジストラと呼ばれる認定機関に問い合わせる必要がある（対象となる TLD によっては手数料が必要）。*レジストラ*は、エンドユーザ向けの特定の TLD（場合によっては TLD のサブドメイン、たとえば co.uk など）にドメイン名を登録する権限を与えられている企業である。レジストラは世界中に存在する。レジストラは、ユーザから登録依頼を受けると、対応する TLD（または TLD の下のサブドメイン）を管理するレジストリを調べることにより、名前が使用可能かどうかを確認する。

名前が使用可能であれば、レジストラはその名前を該当するレジストリに登録する。その後、レジストリが新しい名前をそのレジストリデータベースに追加し、DNS に新しい名前を公開する。ドメインによっては（たとえばいくつかの国コードと gTLD）、同一組織がそのドメインのレジストリとレジストラの役割を果たすこともある。ドメイン保有者に代わって名前を登録する中間組織は存在しない。

エンタープライズレベルのドメイン名を登録し、取得する組織はしばしば、さまざまな部門に関連付けられたインターネットリソースを正しく識別できるように、子ドメインを作成する必要が生じる。たとえば、ドメイン名 example.com の所有者であれば、組織の発送部門に関連付けられたリソースを作成、識別するためにサブドメイン shipping を作成することが考えられる。同様に、組織のすべてのインターネットリソースを正しく識別するために、ほかにも多くのサブドメイン（この文脈では第 3 レベルドメイン）を作成することができる。しかし、1 つの組織（すなわち第 2 レベルドメインの所有者）において、第 3 レベルドメインは多数あるが、そのドメインごとのインターネットリソース（Web サーバ、アプリケーションサーバなど）は少数であることが多い。したがって、このような第 3 レベルおよびそれよりも下位のドメインのそれぞれに対して個別のネームサーバを割り当てるのは経済的ではない。さらに、ある組織の主要ドメイン（第 2 レベルドメインやエンタープライズレベルドメインなど）とそのすべてのサブドメインに関する情報を 1 つのリソースにグループ化し、1 つの単位として運用するのが管理上は便利である。

このグループ化を容易にするために、DNS ではゾーンという概念を定義している。ゾーンは、ドメイン全体であっても特定のドメインとその 1 つまたは複数のサブドメインであってもよい。ゾーンとは、ネームサーバ内の設定可能な要素であり、このゾーンの下で、1 つのドメインと選択した一連のサブドメインに関するすべてのインターネットリソースの情報が記述される。つまり、ドメインが DNS 名前空間の構造上の基本要素であるのと同様に、ゾーンは DNS 名前空間の管理上の基本要素である。結果として、ゾーンという用語は、独立した管理エンティティとして管理されている 1 つのドメインを指す場合にも一般的に使われている（たとえばルートゾーン、.com ゾーンなど）。この文書では、ゾーンという用語は、1 つまたは複数のドメインに関するドメイン名情報が含まれ、1 つの管理エンティティとして管理されているリソースエンティティを指す場合に使用する。つまりゾーンは、ドメイン名情報が格納されている、導入済みのネームサーバ内部の設定可能なリソースである。

DNS インフラストラクチャ（ネームサーバとリゾルバ）、ドメイン名、ゾーン、各種レベルのネームサーバ（ルートサーバ、TLD サーバ）およびリゾルバの全般的な知識を得たところで、今度は名前解決機能をさらに詳しく定義していくことができる。Web ブラウザにユーザが www.example.com という URL を入力すると、ブラウザプログラムは*スタブリゾルバ*という種類のリゾルバに問い合わせを行う。するとスタブリゾルバは、ローカルのネームサーバ（*再帰的ネームサーバ*または*リゾルビィングネームサーバ*と呼ばれる）に問い合わせを行う。リゾルビィングネームサーバは、そのキャッシュを調べて、アクセスされたインターネットリソース（たとえば www.marketing.example.com）の IP アドレスを提供するための有効な情報があるか確認する（情報は、この文書で後述する基準に基づいて有効であると判断される）。情報がなければ、リゾルビィングネームサーバはキャッシュを調べて、marketing.example.com ゾーンのネームサーバに関する情報があるか確認する（これはリソース www.marketing.example.com を含むと想定されるゾーンであるため）。そのネームサーバの IP アドレスがキャッシュにあれば、リゾルバの次の問い合わせ先はそのネームサーバに向けられる。marketing.example.com のネームサーバの IP アドレスがキャッシュにない場合、リゾルバは marketing.example.com よりも 1 つ上位のレベルのゾーン（すなわち example.com）のネームサーバの情報があるか確認する。example.com のネームサーバ情報がなければ、次の検索はキャッシュ内で.com ゾーンのネームサーバを探すことになる。

キャッシュを（上記のとおり）完全に検索してもなお必要な情報が得られない場合、リゾルビィングネームサーバは、DNS 名前空間階層の最上位のネームサーバ（すなわちルートサーバ）に問い合わせることにより検索を開始する以外に手段がない。キャッシュの検索に成功した場合、リゾルビィン

グネームサーバは、ルートゾーンよりも下位のレベル（この場合は marketing.example.com、example.com、または.com）においてネームサーバに問い合わせをしなければならない。ルートサーバから始まる一連の反復クエリには、より下位レベルのサーバから始まるクエリが包含されるため、このセクションでは、エンタープライズレベルのリゾルビィングネームサーバによってルートサーバへ送られたクエリからの名前解決プロセスについて説明する。

ルートサーバへの問い合わせは、DNS 内のどのネームサーバにも通常存在する「ルートヒント」と呼ばれるファイルによって行われる。ルートヒントファイルには、13 のルートサーバのうちの 1 つ以上の IP アドレスが格納されている。ルートサーバには、その子ゾーン（すなわち TLD）のネームサーバに関する情報が保持されている。TLD（たとえば.com）には、その子ゾーン（たとえば example.com）に関するネームサーバ情報が保持されている。ゾーンに保持されている、子ゾーンに関するネームサーバ情報のことを、*委任情報*と呼ぶ。委任情報は、ゾーンがドメイン名階層においてそれよりも下位のリソースに対する名前解決リクエストを転送するために使用する情報である。この例では、名前解決リクエストが第 3 レベルドメインのリソースに関するものであるため、ルートサーバはリクエストを下位レベルのネームサーバに転送する必要がある。この委任情報の送信に関わるリゾルビィングネームサーバに対するレスポンスは、参照と呼ばれる。*参照*は、リクエストに関連する TLD ゾーン（すなわち.com ゾーン）のネームサーバの名前と IP アドレスを提供する（クエリが marketing.example.com のリソースに対するものであるため）。この参照を用いて、リゾルビィングネームサーバはクエリを作成し、.com ゾーンのネームサーバへ送る。このサーバは、example.com のネームサーバに対する参照を返す。marketing.example.com ドメインが example.com のゾーンに含まれている場合、example.com のネームサーバに問い合わせることにより、リソース www.marketing.example.com の IP アドレスが得られる。名前解決プロセス（キャッシュ検索なし）を図 2-2 に図示する。

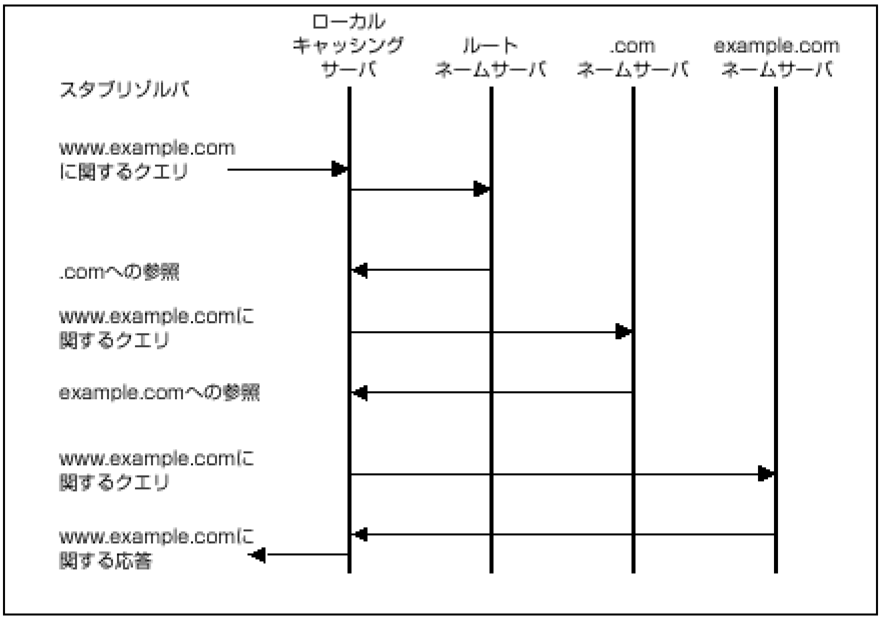

図 2-2.名前解決プロセス（キャッシュ検索なし）

名前解決プロセスの説明で明確にしたとおり、ネームサーバは次の機能を実行する。

+ 子ゾーンへの参照を返す。
+ ドメイン名から IP アドレスへのマッピング（*ドメイン名の解決*と呼ばれる）、または IP アドレスからドメイン名へのマッピング（*逆引き解決*とも呼ばれる）を返す。ただし実のところ、これは異なる種類のデータを得るための標準的なクエリである。

+ 存在しない DNS エントリのクエリに対しては、エラーメッセージを返す。

ネームサーバは、この 3 つの機能を、*ゾーンファイル*と呼ばれる同一の DNS データベースを用いて実行する。*委任情報*と呼ばれる情報には、親ゾーン内の子ゾーンのネームサーバ情報が含まれており、参照機能を実行する際に使われる。マッピング機能は、ゾーンファイル内の*権威情報*と呼ばれる分類に属する情報によって実行され、そのゾーン内のリソースを列挙したレコード一式と、そのドメイン名および IP アドレスによって提供される。リソースはそのゾーンに属するため、提供される情報は実際に権威を持つとみなされる。したがって、ゾーンファイルには 2 つのカテゴリの情報が含まれている－権威情報（ゾーン内のすべてのドメインのすべてのリソースに関する情報）と、委任情報（子ゾーンのネームサーバに関する情報）である。ゾーンファイル内で委任情報が置かれている場所は、*委任点*と呼ばれる。ゾーンファイルのレベルは、そのファイルが権威を持つ情報の対象となるドメインの中で最上位ドメインのレベルと等しい。先述の例では、example.com のネームサーバのゾーンファイルは、エンタープライズレベルのゾーンファイルであり、対応するネームサーバはエンタープライズレベルネームサーバと呼ばれる。

2.3 DNS 構成要素とセキュリティ目標

セキュリティ目標を決定する前に、DNS の構成要素を指定する必要がある。DNS には次の要素が含まれる。

+ ネームサーバとリゾルバが存在するプラットフォーム（ハードウェアとオペレーティングシステム）
+ ネームサーバおよびリゾルバソフトウェア
+ DNS トランザクション
+ DNS データベース（ゾーンファイル）
+ ネームサーバとリゾルバ上の設定ファイル

メディアアクセスレベルのネットワーク技術（すなわち、スタブリゾルバとローカルリゾルビングネームサーバを接続する Ethernet ネットワーク）は、DNS の定義には含まれない。

機密性、完全性、可用性、および情報源の真正性確認は、あらゆる電子システムに共通のセキュリティ目標である。しかし、DNS は一般公開されているインターネットリソースの名前解決情報を提供することを期待されている。セキュアチャネルを通じて内部の DNS ネームサーバが提供する内部リソース（たとえば、ファイアウォールの内側のサーバ）に関する DNS データを除き、公開 DNS ネームサーバが提供する DNS データには機密性はないとみなされている。したがって、機密性は DNS のセキュリティ目標には含まれない。情報の真正性を保証し、伝送中の情報の完全性を維持することは、DNS が名前解決サービスを提供しているインターネットを効率良く機能させるために重要である。したがって、完全性と情報源の真正性確認が、DNS の主たるセキュリティ目標である。

ネームサーバプラットフォームや、DNS 構成要素（ネームサーバソフトウェアやリゾルバソフトウェアなど）が置かれている基盤ネットワークの多様性のため、あらゆる種類のサービス運用妨害攻撃を防ぐことは現実的ではない。ただし、ネームサーバプラットフォーム、ゾーンファイルのデータ、特定の DNS トランザクションのためのアクセス制御設定の脆弱性を悪用したサービス運用妨害攻撃を防ぐために、いくつかの基本的なガイドラインに従う必要がある。

2.4　この文書の焦点

ネームサーバには、3 つの主要なレベルがある。すなわち、ルートネームサーバ、TLD ネームサーバ、エンタープライズレベルネームサーバである。この文書では、TLD レベルの.govドメイン（ゾーン）、および.gov TLD の配下にあるすべてのエンタープライズレベルネームサーバのセキュリティを保護するための導入ガイドラインを提示する。つまりこのガイドラインは、軍関連を除くすべての米国連邦政府機関における、すべてのネームサーバのセキュアな設定と運用を対象としている。対象読者は主に、これらのネームサーバの設定と運用について責任を持つゾーン管理者である。ただし、このガイドラインはあらゆるエンタープライズレベルゾーン（たとえば、mit.edu など）にも同様に適用可能である。

この文書のガイドラインが適用するセキュリティメカニズムは、IETF の DNSSEC および TSIG 仕様に適合している。この文書のガイドラインでは、次に示す DNS 構成要素／関連操作に対するポリシー、設定オプション、チェックリストを取り上げている。

+ DNS ホスティング環境
  - ホストプラットフォーム（OS、ファイルシステム、通信スタック）
  - DNS ソフトウェア（ネームサーバ、リゾルバ）
  - DNS データ（ゾーンファイル、設定ファイル）
+ DNS トランザクション
  - DNS クエリ／レスポンス
  - ゾーン転送
  - 動的更新
  - DNS 通知
+ セキュリティ管理業務
  - アルゴリズムと鍵サイズの選択（TSIG および DNSSEC）
  - 鍵管理（生成、格納、使用）
  - 公開鍵の公開方法およびトラストアンカの設定
  - 鍵のロールオーバ（計画的な実施と緊急時の実施）

（本ページは意図的に白紙のままとする）

3. DNS データと DNS ソフトウェア

DNS の 2 つの主要なソフトウェア構成要素は、ネームサーバとリゾルバである。ネームサーバの主な機能は、ドメイン情報が格納されているデータベース（ゾーンファイルと呼ばれる）を提供することと、名前解決クエリに対して、権威を持つレスポンスまたは参照を通じて応答することである。リゾルバソフトウェアの主な機能は、名前解決クエリ（単体または一連のもの）を作成することである。主要な DNS データは、ゾーンファイルである（設定ファイルは、また別の種類の DNS データである）。3.1 項では、ゾーンファイルの構成について説明する。3.2 項および 3.3 項は、各種のネームサーバとリゾルバの機能をそれぞれ取り上げる。ネームサーバとリゾルバについては、エンタープライズレベルの範囲で説明しており、ルートレベルと TLD レベルにおけるネームサーバとリゾルバには該当しない場合がある。

3.1 ゾーンファイル

ゾーンファイルには、そのゾーン内の各種リソースに関する情報が含まれている。各リソースに関する情報は、リソースレコード（RR）と呼ばれるレコードで表される。ゾーンには複数のドメインが含まれ、各ドメイン内に複数のタイプのリソースが存在することが考えられるため、各 RR にはこれを識別するためのフィールドがある。具体的には、RR は次に示す主要なフィールドで構成されている。

+ **所有者名（Owner name）:**ドメイン名またはリソース名
+ **TTL:**存続可能時間（秒単位）
+ **クラス（Class）:**現在、IN（インターネットを表す）という 1 つのクラスのみ使用されている
+ **RRType:**リソースのタイプ
+ **RData:**リソースに関する情報（RRType による）

一般的な RRType としては、たとえば以下のものがある。

+ **A:**アドレス RRType。このタイプの RR は、ホスト名（FQDN によって識別される）の IP アドレスを示す。
+ **MX:**メールエクスチェンジャ RRType。このタイプの RR は、ドメインのメールサーバホスト名を示す。
+ **NS:**ネームサーバ RRType。このタイプの RR は、ドメインのネームサーバホスト名を示す。

IETF RFC 1035 に、DNS において有効な RRType の完全な形式が示されている[4]。共通の所有者名の下に RRType ごとに複数のリソースが存在することが考えられ（たとえば、ネームサーバとして機能するホストが複数あるなど）、なおかつクラス（CLASS）は 1 つのみ（すなわち IN）であるため、ゾーンファイル内で注目する情報（たとえば、ドメイン内のすべてのメールサーバ、[RRType = MX]）は、同じ所有者名、TTL、クラス、および RRType を含んだ RR の組み合わせのなかにある。同じ所有者名、クラス、RRType を持つ RR を組み合わせたものを、*RRSet* と呼ぶ。したがって、ゾーンファイルは論理的には複数の RRSet で構成されている。

[4] RFC 1035、『*Domain Names - Implementation and Specification*』は、http://www.ietf.org/rfc/rfc1035.txt から入手できる。

3.2　ネームサーバ

ネームサーバには、主に 2 つのタイプがある。権威ネームサーバと、キャッシングネームサーバである。*権威*とは、ゾーンに対して権威を持つことを示している。あるネームサーバが特定の 1 つまたは複数のゾーンの RR について権威を持つ情報源である場合、そのネームサーバは、その 1 つまたは複数のゾーンの権威ネームサーバと呼ばれる。ゾーンの権威ネームサーバは、そのゾーンのリソースを問い合わせる名前解決クエリに対して、自身のゾーンファイルの RR を使ってレスポンスを返す。これに対して*キャッシングネーム サーバ*（名前解決／再帰的ネームサーバとも呼ばれる）は、名前解決クエリ内のドメインの階層において権威ネームサーバへの一連のクエリを通じてレスポンスを返すか、または過去のクエリを使用して構築されたレスポンスのキャッシュからレスポンスを返す。

3.2.1　権威ネームサーバ

権威ネームサーバには、2 つのタイプがある。マスタ（プライマリ）ネームサーバと、スレーブ（またはセカンダリ）ネームサーバである。フォールトトレランス向上のため、組織に複数のスレーブネームサーバが存在することが考えられる。*マスタネーム サーバ*には、ゾーン管理者によって手動で作成、編集されるゾーンファイルが含まれている。場合によっては、マスタネームサーバが、認可された DNS クライアントにゾーンファイルの動的更新を許可することもある。この機能を持つように設定されたマスタネームサーバは通常、*プライマリマスタネーム サーバ*と呼ばれる。*スレーブ（セカンダリ）ネーム サーバ*にも、ゾーンに関して権威を持つ情報が含まれているが、そのゾーンファイルはマスタネームサーバにあるファイルの複製である。複製は、マスタネームサーバからスレーブネームサーバへゾーンファイルの RR をすべて転送する、ゾーン転送と呼ばれるトランザクションを通じて行われる。名前解決クエリは、特定の RR を問い合わせるものであるため、ゾーン転送は実際には、「問い合わせ対象のゾーンのすべての RR」を意味するタイプコードである AXFR を持つ名前解決クエリの一種として扱われる。ゾーンファイルの内容がマスタネームサーバ側で変更されるたびに、スレーブネームサーバは DNS NOTIFY と呼ばれるトランザクションを通じて、変更について通知される。スレーブネームサーバは、このリクエストを受け取ると、マスタネームサーバに対してゾーン転送リクエストを開始する。

3.2.2　キャッシングネームサーバ

キャッシングネームサーバは、一般に組織内のローカルのネームサーバであり、組織のさまざまなクライアントに代わって名前解決機能を実行する。キャッシングネームサーバは、名前解決／再帰的ネームサーバとも呼ばれる。名前解決機能は、キャッシングネームサーバによって、スタブリゾルバからのクエリに応答して実行される。名前解決で行う検索プロセスには、自身のキャッシュの検索、一連の反復クエリを通じた権威ネームサーバへの再帰的な問い合わせ、あるいはこれらの組み合わせが含まれることがある。

特定のネームサーバを、権威ネームサーバと再帰的なネームサーバの両方の役割を果たすように構成することができる。この構成では、同一のネームサーバが、権威を持つゾーンに関するクエリに対して権威を持つ情報を提供する一方で、他のゾーンに関するクエリに対しては名前解決機能を実行する。名前解決機能を実行するには、ネームサーバは再帰的クエリをサポートする必要がある。再帰的クエリをサポートするサーバは、これをサポートしないサーバよりも、攻撃に対して脆弱である（次の項を参照）。結果として、権威を持つ情報が侵害される可能性がある。このため、1 つのネームサーバが権威ネームサーバと再帰的ネームサーバの機能を実行するように構成することは、よいセキュリティプラクティスとはいえない。

3.3 リゾルバ

Web ブラウザや電子メールクライアントのように、インターネットリソースにアクセスする必要のあるソフトウェアは、*クライアントリゾルバ*またはスタブリゾルバと呼ばれる DNS クライアントを利用する。スタブリゾルバは、インターネットにアクセスするソフトウェアが求めるリソースの名前解決クエリを作成し、それを組織内のキャッシング（名前解決）ネームサーバに送信する。スタブリゾルバは一般に、運用のための一定のフォールトトレランスを実現するために、複数のリゾルビィングネームサーバのリストで構成される。スタブリゾルバは、単に*リゾルバ*と呼ばれることが多い。スタブリゾルバからクエリを受信するキャッシング（名前解決）ネームサーバも、名前解決のための構成要素とキャッシング（ネームサーバ）のための構成要素を持っているため、（キャッシュからクエリに応答することができない場合に）権威ネームサーバに問い合わせるためにクエリを作成することから、やはり*リゾルバ*と呼ばれることがある。

4. DNSトランザクション

DNSトランザクションの最も一般的なタイプは次のとおりである。

+ DNS クエリ／レスポンス
+ ゾーン転送
+ 動的更新
+ DNS NOTIFY

このセクションでは、これらの各トランザクションについて説明する。

4.1 DNS クエリ／レスポンス

これは DNS において最も一般的なトランザクションである。DNS クエリはリゾルバから発信され、宛先は権威ネームサーバ、あるいはキャッシングネームサーバである。最も一般的なクエリは、所有者名または RRType に基づく RR の検索である。レスポンスは、単一の RR、RRSet、またはエラーメッセージである。

先述のとおり、クエリには反復型と再帰型の 2 つのタイプがある。反復クエリを送信する DNS リゾルバは、クエリへの最終的な回答を得るために複数の参照を行う必要が生じる可能性があるため、それらが提供するレスポンスの種別に関してより堅牢である傾向がある。リゾルバが DNS キャッシュも持っている場合、DNS 内のネームサーバの全体像とレスポンスを、過去のクエリから組み立てることができる。これを使用して以後のクエリに対する応答時間を短縮できる。再帰的クエリは通常、複雑な DNS 操作を処理する機能のないスタブリゾルバから送信される。スタブリゾルバは、そうした機能がない代わりに、上流の DNS エンティティ（通常は一連のスタブリゾルバに代わって反復クエリを送信する、キャッシュを持つネームサーバ）を利用して、クエリを実行して、最終的な回答を返す。

DNS クエリは、単一の UDP パケットで送信される。レスポンスも同様に、通常は単一の UDP パケットであるが、データサイズは切り捨てられることがあり、その場合は通常、TCP を使ってクエリを再発行することになる。リソース消費のオーバーヘッドが低いことから UDP が望ましいが、DNS 管理者は、レスポンスのゾーンデータの大部分が、切り捨てられた DNS レスポンスで占められないようにする必要がある。

DNS クエリは平文で送信され、受け取るレスポンスは正しく、信頼できる送信元からのものと想定される。その結果、能動的な攻撃者がクエリ元のクライアントへのレスポンスを横取り（および改ざん）したり、偽造したりすることが可能である。6.1 項で、DNS クエリ／レスポンストランザクションに対する脅威について、さらに詳しく説明する。8.1.1 項およびセクション 9 では、これらの脅威に対応するためのセキュリティ対策案を提示する。

4.2 ゾーン転送

*ゾーン転送*とは、（セカンダリ）スレーブサーバが、ゾーンファイルの内容全体を（プライマリ）マスタサーバのファイルに基づいて更新する手段である。このプロセスにより、セカンダリネームサーバは、ゾーンファイルをプライマリネームサーバと同期させることができる。ゾーン転送トランザクションは、セカンダリネームサーバからプライマリネームサーバへのクエリとして開始する。ゾーン転送クエリは、DNS クエリとは対照的に、特定の所有者名や RRType の RR をリクエストするのではなく、ゾーンのすべての RR をリクエストする。ゾーン転送クエリは、DNS NOTIFY メッセージ（4.4 項参照）に

応えて、または、ゾーンの SOA（Start of Authority）RR の RData フィールドの Refresh データ項目の値に基づいて、セカンダリネームサーバから発信される。

ゾーン転送プロセスは、通常の DNS クエリよりも大量の情報を開示し、メッセージのリソース使用量も増えるため、さまざまなセキュリティに影響する。ゾーン転送トランザクションに対する脅威については、6.2 項で説明する。保護メカニズムについては、8.1.2 項および 8.2.5 項で述べる。

4.3　動的更新

組織で IP ネットワークベースのリソース（すなわち、データベースサーバ、Web サーバ、メールサーバ、場合によってはネームサーバも）の追加、削除、移動を行うたびに、リソースが置かれているドメインについての情報を保持するゾーンファイルに、対応する変更を加える必要が生じることがある。DNS の初期の時代は、こうした変更は DNS ゾーン管理者が手動で行っていた。しかし変更量が多くなり頻度も増えると、手動更新プロセスでは不十分であることがわかり、動的更新の概念が導入された。量と頻度とは別に、アプリケーションプログラムを介した DNS ゾーンファイルへの即時の自動更新を必要とするアプリケーションがいくつかある。このようなアプリケーションの例としては、認証局（CA：Certificate Authority）サーバ、DHCP(Dynamic Host Configuration Protocol: 動的ホスト設定プロトコル) サーバおよびインターネットマルチキャストアドレスサーバがある。

いくつかの例では、DNS サーバが公開鍵証明書のリポジトリとして使われている。これらの場合、CA サーバはユーザから公開鍵を受け取り、証明書を生成するための秘密鍵でそれに署名し、この証明書を CERT タイプの RR（CERT RR）として DNS サーバに格納する。DHCP サーバは、リクエスト元のホストに IP アドレスを動的に割り当て、この情報（ホストと新しく割り当てられた IP アドレス）を A タイプの RR として追加する。また、DHCP サーバは、ホストによって IP アドレスが返された後、この A タイプの RR の削除も行う。インターネットマルチキャストサーバは、インターネットマルチキャスト IP アドレス空間から新しいマルチキャスト IP アドレスを選択し、新しく生成されたマルチキャストグループにそれを割り当てる。次にサーバは、ユーザが IP アドレスではなくドメイン名を使ってマルチキャストグループに加わることができるように、この情報（マルチキャストグループのドメイン名、新しく選択されたマルチキャストアドレス）を A タイプの RR として DNS データベースに追加する。

RFC 2136 [7]に、後に BIND 8 に実装され、以降すべてのバージョンに引き継がれている動的更新メカニズムのロードマップがまとめられている[5]。

動的更新機能により、ゾーンファイルにおける RR の追加と削除の操作が提供される。既存の RR の内容の更新は、古いレコードの削除と、内容が変更された新規レコードの追加によって達成できるため、独立した更新という操作は提供されない。

動的更新機能に含まれる一連の操作は、次のもので構成される。

+ 既存のドメインの個々の RR の追加または削除
+ 既存のドメインの特定の RRset（同じ所有者名、クラス、RRType を持つリソースレコード群[たとえば、当然ながら共通クラスが IN であるドメイン／所有者名 example.com の RRType が NS である RR の集合）の削除。
+ 既存のドメイン（指定されたドメイン名のすべてのリソースレコード[たとえば、ドメイン example.com のすべての RR]）の削除。

[5] RFC 2136、『*Dynamic Updates in the Domain Name System (DNS UPDATE)*』は、http://www.ietf.org/rfc/rfc2136.txtから入手できる。

+ 新しいドメインの追加（新しいドメインの 1 つ以上の RR[たとえば、新しいドメイン NYBranch.example.com の A タイプの RR の追加]）。

動的更新を許可されている DNS ゾーンは、多くの悪意ある攻撃に対して無防備である。起こり得る攻撃の網羅的なリストを 6.3 項で取り上げる。これらの攻撃を制限あるいは回避するための解決案を、8.1.3 項、8.2.6 項、8.2.7 項および 8.2.8 項に示す。

4.4 DNS NOTIFY

プライマリ（マスタ）DNS サーバのゾーンファイルに変更が生じるたびに、それと同じデータを持っていると想定されているセカンダリ（スレーブ）DNS サーバは、変更があったことを通知される必要がある。この通知は、セカンダリ DNS サーバに対してゾーン転送を開始するように合図する *DNS NOTIFY メッセージ*を通じて行われる（4.2 項参照）。DNS NOTIFY メッセージは、セカンダリサーバがプライマリサーバと同期をとるための、もう 1 つの手段（セカンダリサーバが、SOA の更新値（Refresh）のタイムアウトに基づいて変更がないか、プライマリサーバをポーリングする）と比べて効率的かつ迅速な手段である。

DNS NOTIFY メッセージの送信は、*通知操作*と呼ばれ、BIND 9.x のデフォルトの操作である。では、DNS ネームサーバソフトウェアは、どのようにして DNS NOTIFY メッセージの送信先とすべきサーバを認識するのだろうか。BIND 9.x では、デフォルトでは、ゾーンの NS RR に定義されているサーバに通知することになっている。ゾーン管理者が DNS NOTIFY メッセージの送信先として希望するサーバがほかにもある場合（たとえば、ステルス型スレーブサーバ）、DNS 管理者は、BIND 設定ファイルにサーバの IP アドレスを追加できる。また管理者は、設定オプションを通じて、このネームサーバのサービスを受ける特定の 1 つのゾーン、またはすべてのゾーンへの通知メッセージの送信を止めることもできる。

セカンダリサーバが DNS NOTIFY メッセージを受け取ると、サーバは関連するゾーンの更新値をゼロにリセットし、これによりゾーン転送が開始される。ゾーン更新の場合と同様、SOA RR のゾーンのシリアル番号がインクリメントされていなければ（10.1 項）、ゾーン転送は行われない。この処理により、ゾーンへの変更をすべてのネームサーバに迅速に伝えることができる。

DNS NOTIFY メッセージが引き金となってゾーン転送が行われるため、DNS NOTIFY メッセージが偽造された場合、不要なゾーン転送が行われ、それよってサービス運用妨害が生じる可能性がある。このような偽造通知を最小限に抑えるための解決案は、8.1.4 項に示す。

（本ページは意図的に白紙のままとする）

5. DNS ホスティング環境――脅威、セキュリティ目標、保護策

DNS ホスティング環境は、次の要素で構成される。

+ ホストプラットフォーム（オペレーティングシステム[OS]、ファイルシステム、通信スタック）
+ DNS ソフトウェア（ネームサーバ、リゾルバ）
+ DNS データ（ゾーンファイル、設定ファイル）

このセクションでは、ホスティング環境のこれらの部分に対する脅威と推奨される保護策について述べる。

5.1 ホストプラットフォームに対する脅威

DNS ソフトウェアをホストするプラットフォームに対する脅威は、インターネット上のホストが直面する脅威と変わらない。これらの一般的な脅威とその影響を、特に DNS ホストの観点から以下に示す。

+ **脅威 T1**：DNS ホスト上の OS、システムソフトウェア、アプリケーションソフトウェアはすべて、バッファオーバフローのような攻撃に対して脆弱であり、その結果、名前解決サービス運用の妨害を引き起こすことが考えられる。
+ **脅威 T2**：DNS ホスト（スタブリゾルバ、キャッシング／名前解決／再帰的ネームサーバ、権威ネームサーバなど）上の TCP/IP スタックが、パケットフラッド攻撃（SYNC や smurf など）を受けた場合、通信障害が発生することが考えられる。アプリケーション層における同等の攻撃は、偽造された大量の DNS クエリを送信して権威ネームサーバやリゾルビィングネームサーバを飽和させることである。
+ **脅威 T3**：DNS ホストが接続されているローカルエリアネットワーク（LAN）セグメントへのアクセス許可を持つ悪意ある侵入者が、DNS メッセージの流れを中断させるアドレス解決プロトコル（ARP）スプーフィング攻撃を開始することが考えられる[6]。
+ **脅威 T4**：通信を可能にするプラットフォームレベルの設定ファイル（たとえば、UNIX プラットフォームでは resolv.conf と host.conf）は、ウイルスとワームによって破壊されたり、ファイルレベルの保護が不十分なために無許可の変更を受けたりすることがあり、その結果、DNS ホスト間（たとえば、スタブリゾルバとリゾルビィングネームサーバ間、リゾルビィングネームサーバと権威ネームサーバ間など）で通信が断絶することが考えられる
+ **脅威 T5**：DNS 固有の設定ファイル（named.conf、root.hints など）、データファイル（ゾーンファイル）、暗号鍵を含むファイルは、ウイルスやワームによって破壊されたり、ファイルレベル保護が不十分なために無許可の変更を受けたりすることがあり、その結果、名前解決サービスが適切に機能しなくなることが考えられる。
+ **脅威 T6**：DNS クライアントと同じ LAN 上の悪意のあるホストが、DNS レスポンスを横取りし、さらに改ざんできる可能性がある。これにより攻撃者は、クライアントを別のサイトに導くことができる。これがクライアントホストに対する攻撃の最初の行為になる可能性がある。

[6] これは厳密にはホストに対する脅威ではなく、ネットワークに対する脅威であり、DNS サーバを制限された LAN セグメント（たとえば、VLAN 経由など）に設置することで軽減される。一般的なネットワークレベルの脅威はこの文書の適用範囲から外れるが、この脅威は、DNS パラメータ（すなわち IP アドレス）に影響が及ぶために取り上げている。

5.2 DNS ソフトウェアに対する脅威

DNS ソフトウェア自体に対する脅威は、セキュリティに重大な影響を与える可能性がある。以下に、最も一般的なソフトウェアの問題と、それに対する脅威の影響を示す。

+ **脅威 T6**：DNS ソフトウェア（ネームサーバまたはリゾルバ）には、バッファオーバフローなど、サービス運用妨害につながる脆弱性があることが考えられる。

+ **脅威 T7**：DNS ソフトウェアには、その設定ファイル（named.conf など）、データファイル（ゾーンファイルなど）、および署名鍵（TSIG や DNSKEY など）を含んだファイルを対象とした無許可での読み取り／更新を防止するための十分なアクセス制御機能がない。このような機能は、脅威 T4 および T5 に示した OS のファイルレベルの保護の上位レベルで提供されており、後者に依存する場合がある。

5.3 DNS データ内容による脅威

DNS データは、ゾーンファイルと設定ファイルの 2 種類で成り立っている。どちらの種類の内容もセキュリティに影響を与える。この文書で解説するすべてのセキュリティ導入オプションは、設定ファイルの内容に関連している。ゾーンファイルの内容に起因するセキュリティへの影響については、セクション 10、「DNS の内容制御を通じて情報露出を最小限に抑えるためのガイドライン」で解説しており、影響のほとんどが、ゾーンデータの次の側面によるものである。

+ さまざまな RRType の RR の特定の鍵フィールドのパラメータ値

+ ゾーンファイルの特定の RR の存在

ゾーンファイルにある各種の不適切な内容は、次のようにさまざまなセキュリティ上の危険および潜在的な脅威をもたらす。

+ **脅威 T8—不完全な委任**：このエラーは、ネームサーバの FQDN または IP アドレスが子ゾーンで変更されているが、親ゾーンが委任情報（NS RR およびグルーレコード）を更新していない場合に生じる。この状況では、子ゾーンは到達不能になる（サービス運用妨害）。

+ **脅威 T9—ゾーンドリフトおよびゾーンスラッシュ**：プライマリネームサーバの SOA RR の Refresh（更新）フィールドと Retry（再試行）フィールドの値が高すぎるとき、ゾーンファイルが頻繁に変更されると、プライマリネームサーバとセカンダリネームサーバのあいだでデータの不一致が生じる場合がある。このエラーはゾーンドリフトと呼ばれ、セカンダリネームサーバ側のゾーンデータが不正確なものとなる。SOA RR の Refresh フィールドと Retry フィールドの値が低すぎると、セカンダリサーバはゾーン転送を頻繁に行うことになる。このエラーはゾーンスラッシュと呼ばれ、プライマリネームサーバとセカンダリネームサーバの負荷が増大する。このような不正確なデータや負荷の増加は、サービス運用妨害につながる可能性がある。

+ **脅威 T10—標的を定めた攻撃についての情報**：HINFO や TXT などの RR は、ソフトウェア（たとえば Web サーバやメールサーバなどのリソース）の名前とバージョンに関する情報を提供する。知識の豊富な攻撃者は、その情報を使ってそれらのソフトウェアのバージョンにおける既知の脆弱性を利用し、当該リソースに対して攻撃を開始することができる。

5.4 セキュリティ目標

DNS ホストプラットフォーム、DNS ソフトウェア、DNS データの保護に関する共通の目標は、完全性と可用性である。

5.5　ホストプラットフォームの保護策

DNS ホストプラットフォームに対する保護／脅威軽減の措置としては以下がある。

+ セキュア OS の実行
+ OS の安全な設定／導入

これらの措置については 7.1 項で説明する。

5.6　DNS ソフトウェアの保護策

以下に、ベストプラクティスに従った DNS ソフトウェアの保護策を示す。

+ 最新バージョンのネームサーバソフトウェアを実行する。前のバージョンの場合は適切なパッチを適用する
+ 限定された権限でネームサーバソフトウェアを実行する
+ ネームサーバソフトウェアを隔離する
+ 機能ごとに専用のネームサーバインスタンスを設ける
+ 指定外のホストからネームサーバソフトウェアを削除する
+ フォールトトレランスのために、権威ネームサーバをネットワークトポロジ上と地理上で分散して構築する
+ 物理的に同じネームサーバ上に異なる 2 つのゾーンファイルを設定する（*分割 DNS*）か、または異なるクライアントクラスごとに別々のネームサーバを設定して IT リソース情報の露出を制限する。

これらの措置については、7.2.1～7.2.8 項で説明する。

5.7　DNS データ内容制御—保護策

ゾーンファイル内の不適切な内容の制御は、セキュリティへの影響がないか内容を分析し、不適切な内容の有無をチェックする完全性制約を作成し、その完全性制約を満たしているかどうかゾーンファイルを検証することによって行う。したがって唯一の保護策は、必要な制約情報が記述され、ゾーンファイルに対して実行すれば、それらの制約に違反する内容を検出できる完全性チェックソフトウェアを開発することである。制約情報の編成を支援するために、ゾーンファイルの各種 RR における望ましいフィールド値（範囲やリスト）が必要である。これらの制約は、署名なしゾーンの RR に対してだけでなく、署名付きゾーン（DNSSEC 仕様を実装したゾーン）の追加の RR に対しても開発する必要がある。このためゾーンファイルの内容制御のための推奨事項については、セクション 9 での DNSSEC 展開ガイドラインの説明後、セクション 10 で説明する。セクション 10 ではこれらの追加の RR についても取り上げている。

DNS によって提供されるサービスは、ルータなどのネットワークインフラストラクチャの脆弱性に起因する脅威にも直面している。しかしネットワーク構成の問題は、この文書の対象外である。

（本ページは意図的に白紙のままとする）

6. DNSトランザクション——脅威、セキュリティ目標、保護策

DNSトランザクションに対する脅威は、トランザクションの種類によって異なる。DNS クライアント（スタブリゾルバやリゾルビィングネームサーバ）と DNS サーバ（キャッシング／リゾルビィングネームサーバや権威ネームサーバ）のあいだの名前解決クエリとレスポンス（DNS クエリ／レスポンス）は、インターネット上の任意のノードが関与することが考えられる。このため、それらに対する脅威はゾーン転送、動的更新、DNS NOTIFY といったトランザクションと比べて数も深刻度もはるかに大きい。一般に、ゾーン転送、動的更新、DNS NOTIFY に関与するノードはすべて単一の組織の管理ドメイン内にある。唯一の例外は、組織のプライマリまたはセカンダリネームサーバが、当該組織に代わって ISP や他の組織によって運営されている場合である。しかしこのようなケースでは通常、信頼関係がすでに存在するため、DNS ゾーン転送のために相互認証システムを構成するのは困難ではない。

6.1 DNS クエリ／レスポンスに対する脅威と保護策

DNS 名前解決クエリとレスポンス（DNS クエリ／レスポンス）は一般に、署名のない単一の暗号化されていない UDP パケットを使用する。DNS クエリ／レスポンスのトランザクションに対する既知の脅威については、IETF RFC 3833 に記載されており、次のように分類できる。

+ **脅威 T11**：偽造または偽装されたレスポンス
+ **脅威 T12**：レスポンスからのいくつかの RR の削除
+ **脅威 T13**：ゾーンファイル内のワイルドカード RR への展開ルールの間違った適用

6.1.1 偽造または偽装されたレスポンス

*偽造*または*偽装されたレスポンス*とは、正当な権威ネームサーバから期待されるものとは異なるレスポンスである。偽装レスポンスは、次のものから発信される可能性がある。

+ 侵害された権威ネームサーバ（リゾルビィングネームサーバから発信されたクエリの場合）
+ リゾルビィングネームサーバの汚染されたキャッシュ（スタブリゾルバから発信されたクエリの場合）

権威ネームサーバは、その OS や通信スタックに対するプラットフォームレベルの攻撃によって侵害される可能性がある（5.1 項参照）。

名前解決（キャッシング）ネームサーバのキャッシュは、次の攻撃によって汚染される可能性がある。

+ **パケット横取り。**この種の攻撃では、正当な権威ネームサーバからの実際のレスポンスがリゾルビィングネームサーバに到達する前に、攻撃者がリクエストを傍受し、権威ネームサーバになりすますことによりレスポンスを生成して送信する。
+ **ID 推測とクエリ予測。**この種の攻撃では、攻撃者は DNS リクエストメッセージのヘッダの ID フィールドを推測し（このフィールドは、長さが 16 ビットしかないため、総当たりによる推測が可能）、場合によっては QNAME と QTYPE（所有者名と RRType）も推測する。攻撃者は次に、ネームサーバになりすまして偽装データをレスポンスとしてネットワークに投入する。
+ **侵害された権威ネームサーバから蓄積されたレスポンス。**侵害された権威ネームサーバは、操っている攻撃者により、リゾルビィングネームサーバからのクエリに対して偽装レスポンスを送信するよう指示される。

キャッシュが汚染されたリゾルビィングネームサーバのサービスを受けるシステムへの影響は次のとおりである。

+ **サービス運用妨害。**アドレスレコード（A RR）のようないくつか重要な RR が偽造されると、この情報を必要とするシステムは意図したノードとの接続を確立できなくなる。
+ **キャッシュ汚染を通じたクライアントのリダイレクト。**クライアントのリダイレクトは、RDATA 要素に名前が格納されている DNS RR の選択的な汚染によって行われる。このような RR の例としては、CNAME、NS、MX がある。これらの名前の名前解決（すなわち IP アドレス）情報は、追加の情報セット（代理レスポンスについて論じるときには、*グルーレコード*）のなかにある。通常、リゾルビィングネームサーバはこれらの必要な A/AAAA RR をフォローアップクエリ（*トリガされたクエリ*とも呼ばれる）を通じて取得する。これらのフォローアップクエリによってネットワークに流れ込むレスポンスは、攻撃者にとっては偽造レコードを挿入するさらなるチャンスとなる。まず、攻撃者は攻撃者が選んだ任意の名前を、選択した RR の RDATA 部分に入れることができる。次に、フォローアップクエリへの応答として送信された対応するグルーレコードに、（攻撃者が選んだ）サーバの IP アドレスを挿入できる。2 つ関連するレスポンスに対するこの種の攻撃は、*名前連鎖攻撃*と呼ばれる。リゾルビィングネームサーバのこのようなキャッシュ汚染の全体的な影響は、リゾルビィングネームサーバのサービスを利用しているいくつかのクライアントが間違った方向へ導かれることである。攻撃者が選んだノードへユーザをリダイレクトすることにより、攻撃者はパスワードのような機密情報を取得できる可能性がある。

6.1.2 RR の削除

レスポンスへの偽造データまたは偽装データの投入のほかに、攻撃者がレスポンスから RR を削除する場合もある。このような行為は、名前解決クエリの失敗とサービス運用妨害につながる可能性がある。

6.1.3 ワイルドカード RR への展開ルールの間違った適用

多くのゾーンは、ゾーンファイルのデータの量を節約するためにワイルドカード RR を使用する。ワイルドカードパターンは、名前解決クエリに対するレスポンスを生成する際に、その場で RR を合成するために使われる（合成ルールは、IETF RFC 1034 [4]の 4.3.2 項にまとめられている）[7]。合成ルールがネームサーバで間違って適用されると、組織の既存のリソースに関連付けられた RR が生成されず、DNS レスポンスにおいても使用できない場合がある。これもまたサービス運用妨害につながる。

6.1.4 DNS クエリ／レスポンスの脅威に対する保護策—DNSSEC

DNS クエリ／レスポンスに関する主な脅威（レスポンスの偽造やレスポンスの失敗など）の根本的な特色は、レスポンスで返される DNS データの完全性である。したがって、セキュリティ目標は、受け取った各レスポンスの完全性を検証することである。完全性検証の不可欠な要素は、有効なデータが正しい情報源から発信されたことを検証することである。情報源に対する信頼を確立することを、*データ源の真正性確認*という。したがって、DNS クエリ／レスポンストランザクションを保護するために必要なセキュリティ目標、およびその結果のセキュリティサービスは、データ源の真正性確認とデータ完全性の検証である。

これらのサービスは送信元との信頼を確立し、その送信元から送信されたデータの署名を検証することによって提供される。DNS インフラストラクチャにおけるディジタル署名メカニズムの仕様は、

[7] RFC 1034、『*Domain Names - Concepts and Facilities*』は、http://www.ietf.org/rfc/rfc1034.txtから入手できる。

IETF の DNSSEC 標準にある。DNSSEC に関わる目標、追加の RR、および DNS メッセージ内容については、RFC 4033、4034、4035 で規定されている。DNSSEC では、送信元の（署名検証のための）公開鍵に対する信頼は、第三者あるいは（公開鍵基盤[PKI]連鎖のように）第三者の連鎖に任せるのではなく、信頼できる 1 つのネームサーバ（ルートネームサーバなど）から現在のレスポンスの送信元まで、親による子の公開鍵の署名の連鎖的な検証を通じて、信頼の連鎖を築くことによって確立される。信頼できるネームサーバの公開鍵のことを、*トラストアンカ*と呼ぶ。

送信元の真正性確認後、DNSSEC が要求する次のプロセスは、レスポンスの完全性検証である。このため、レスポンスは、要求された RR だけでなく、それに対応する完全性検証要素（authenticator）によって構成される必要がある。DNSSEC では、この完全性検証要素は RRSet のディジタル署名である。RRSet のディジタル署名は、RRSIG と呼ばれる特殊な RRType を通じてカプセル化される。DNS クライアントは、（信頼が確立されたばかりの）送信元の信頼できる公開鍵を使って、そのディジタル署名を検証してレスポンスが正当なものか偽装されたものかを検知する。

クエリに対応する RR が実際にゾーンファイルにないことと、送信中に削除されていないことを確認するために、DNSSEC メカニズムは RR が存在しないことを確認する手段を提供する。このメカニズムは、所有者名とゾーンファイル内にあるその次の名前に対応する RRType を列挙する、NSEC RR と呼ばれる特殊な RR を生成する。DNSSEC はこの特殊な RR とその署名をリゾルビィングネームサーバに送信する。この署名を検証することにより、DNSSEC 対応のリゾルビィングネームサーバは、ゾーン内に存在する所有者名と、この所有者名側にある、権威を持つ RRType を判別できる。

ワイルドカード RR の展開ルールの間違った適用による脅威から保護するために、DNSSEC メカニズムは有効性が確認されているワイルドカード RR と NSEC RR を比較することによって、ネームサーバが応答の生成時にワイルドカード展開ルールを正しく適用していることを検証する手段を提供している。

DNSSEC は、DNS クライアントが DNSSEC 署名検証を行う限り、インターネットベースのリソースの代理で動作している DNS クライアントへの名前解決レスポンスの完全性を保証できる。ただし多くの場合、これらの DNS クライアントは DNSSEC 対応ではないスタブリゾルバである。署名検証が DNS クライアントに名前解決サービスを提供するリゾルビィングネームサーバによって実行される場合、レスポンスデータのエンドツーエンドの完全性は、リゾルビィングネームサーバとスタブリゾルバのあいだの通信チャネルを保護することによってのみ保証できる。

IETF の設計基準では、DNS データは一般公開されるものとみなしている。このため、機密性は DNSSEC のセキュリティ目標には含まれない。DNSSEC は、サービス運用妨害の脅威に対する直接の保護を提供することを意図したものではないが、メッセージの完全性と情報源の真正性確認によって間接的にこれを行う。また、名前解決クエリとレスポンスは、公共インターネットの数百万ものノードを経由するため、DNSSEC は通信チャネルセキュリティも提供しない。DNSSEC はまた、従来の DNS には存在しなかった新しいタイプの弱点をもたらす可能性もある。DNSSEC がネガティブレスポンスを実行する仕組みの副産物を使って、クライアントはゾーン内のすべての名前をマップできる。攻撃者は標的ゾーン内のすべてのドメイン名と IP アドレスを記載した「地図」を取得できるため、これは攻撃の糸口になり得る。したがって、ゾーンには管理者が公開したいゾーンデータだけが含まれるようにすることが望ましい。内部 DNS の場合、分割 DNS（7.2.7 項参照）のようなものを導入することが考えられる。

6.2 ゾーン転送に対する脅威と保護策

ゾーン転送は、組織が提供する DNS サービスにある程度のフォールトトレランスを持たせる目的で、ゾーンファイルを複数のサーバ上に複製するために実行する。ゾーン転送からの脅威は、IETF RFC を通じて正式に明文化されてはいない。しかしいくつかの脅威が想定される。最初の脅威であ

るサービス運用妨害は、あらゆるネットワークトランザクションに共通である。2 番目の脅威はあらゆるネットワークパケットに共通である。

+ **脅威 T14—サービス運用妨害**：ゾーン転送は、ゾーン全体の転送を伴うため、通常の DNS クエリと比べ、ネットワークリソースに大きな負担をかける。組織のネームサーバに対する誤った、または悪意のある頻繁なゾーン転送リクエストは、マスタゾーンサーバを過負荷にし、正当なユーザへのサービス運用妨害につながる可能性がある。
+ **脅威 T15**：ゾーン転送のレスポンスメッセージは、改ざんされる可能性がある。

ゾーン転送リクエストを出すことを許可するサーバを既知の一連のサーバだけに限定すれば、サービス運用妨害は最小限に抑えられる。このような制限をプライマリネームサーバに設定するには、それらのエンティティを識別する手段が必要である。BIND などのネームサーバソフトウェアは元々、ゾーン転送リクエストを指定された一連の IP アドレスに限定するための設定機能を提供していた。しかし、IP アドレスは偽装が可能なため、この設定モードはゾーン転送アクセスを制限する十分な手段にはならない。

その後 IETF は*トランザクション署名*（TSIG）と呼ばれる別の仕組みを開発した。この仕組みでは、サーバの相互の識別は共有秘密鍵に基づいて行われる。ゾーン転送に関与するサーバの数は限られているため（一般に、組織の同一の管理ドメイン内のネームサーバに限定される）、共有秘密鍵に基づく二者間の信頼モデルはほとんどの組織（非常に大規模な組織を除く）にとって十分であると考えられる。TSIG では、共有秘密鍵は相互認証のためだけでなく、ゾーン転送リクエストおよびレスポンスの署名にも使用されるものと定めている。したがって、TSIG はゾーン転送のレスポンスメッセージの改ざん（脅威 T15）に対抗する保護策を提供する。ゾーン転送メッセージ内の DNS データのみ（ペイロード）の保護は、DNSSEC 署名付きゾーンからの RR に付随する署名レコードの検証を通じて保証できる。しかしこれらの署名は、ゾーンファイルのすべての情報を対象としたものではない（たとえば委任情報など）。さらに、検証できるのは個々の RRset のみであって、ゾーン転送のレスポンスメッセージ全体ではない。

非対称鍵暗号（公開鍵暗号）を用いた DNS トランザクションの真正性確認の方法は、さらにもう 1 つある。SIG(0) RR の書式はリソースレコード署名（RRSIG）RR（9.2.1 項参照）と似ており、（共有秘密鍵ではなく）DNS に格納されている公開鍵を用いて有効性を確認できる。SIG(0)の使用は、計算量の点で高価になることがあるが、事前の信頼関係を必要とせずに SIG(0)による署名付きメッセージを使用できるという利点がある。しかし、ほとんどのゾーン転送は、すでに関係が確立している当事者間で行われるため、ゾーン転送トランザクションの真正性確認のために TSIG を実装するのは容易であると考えられる。

6.3 動的更新に対する脅威と保護策

動的更新では、権威ネームサーバ内のゾーンデータに対し DNS クライアントがリアルタイムで変更を加える。動的更新を実行するクライアントは一般に、CA サーバ、DHCP サーバ、およびインターネットマルチキャストアドレスサーバである。ゾーン転送トランザクションと同様、動的更新トランザクションに関する脅威は、IETF による RFC を通じた正式な文書化はなされていない。以下に、動的更新ではデータ更新リクエストがネットワークを介して伝送されるという事実に基づき、想定されるいくつか共通の脅威を示す。

+ **脅威 T16—無許可の更新**：無許可の更新は、ゾーンデータの内容にいくつかの悪影響をもたらす可能性がある。悪影響をもたらすデータ操作には次のものがある：(a) 不正なリソースの追加（有効なゾーンファイルへの新規の FQDN および RR）、(b) 正当なリソースの削除（FQDN 全体または特定の RR）、(c) 委任情報の変更（子ゾーンを指す NS RR）。

+ **脅威 T17**：動的更新リクエスト内のデータは、改ざんされる可能性がある。
+ **脅威 T18—リプレイ攻撃**：更新リクエストメッセージが捕捉されて後で再送信される可能性がある。これは不適切な更新の原因になる。

脅威 T16 および T17 は、関与するエンティティを認証し、メッセージの改ざんを検知する手段を提供することにより対抗できる。ゾーン転送の場合は、これらのセキュリティ目標は TSIG/SIG(0)メカニズムによって達成されるため、動的更新の保護についても同じ TSIG/SIG(0)メカニズムが指定される。動的更新メッセージの場合、その必須フィールドがある程度のリプレイ攻撃（脅威 T18）保護を備えているが、TSIG/SIG(0)では動的更新リクエストにタイムスタンプフィールドを含めることにより、リプレイ攻撃から保護するための追加のメカニズムを提供している。この署名付きタイムスタンプにより、サーバは動的更新リクエストのタイミングが、設定で指定されている許容される時間の範囲内であるかを判断できる。

ゾーン転送よりも動的更新に対して SIG(0)保護メカニズムを使用するほうが道理にかなっていることがある。動的更新トランザクションは、必ずしも事前のセキュリティ関係があるとはかぎらない当事者間で発生することもあれば、起動処理の一部として行われることもある。したがって、共有秘密による TSIG の使用は不可能だが、DNS に格納されている鍵を使用する SIG(0)による真正性確認は可能であると考えられるのである。

6.4 DNS NOTIFY に対する脅威と保護策

DNS NOTIFY は、プライマリ（マスタ）ネームサーバによってセカンダリ（スレーブ）ネームサーバに送信されるメッセージであり、これによりセカンダリサーバは、更新操作（すなわち、シリアル番号を調べるための SOA RR のクエリ）を開始し、ゾーンの更新が行われていればゾーン転送を実行する。NOTIFY メッセージは単なる合図であるため、メッセージを処理する際のセキュリティリスクはごくわずかである。考慮すべき主なセキュリティリスクは次のとおりである。

+ **脅威 T19—偽りの NOTIFY メッセージ**：セカンダリネームサーバがプライマリネームサーバ以外の送信元から、偽りの DNS NOTIFY メッセージを受け取る可能性がある。

ゾーン転送は、更新されたゾーンがプライマリサーバ上にある場合にのみ発生するため、偽りの DNS NOTIFY メッセージを受け取った場合の唯一の影響は、セカンダリネームサーバの負荷の増大である。この脅威による影響は小さいため、必要な唯一の保護策は、セカンダリネームサーバに対して、DNS NOTIFY メッセージを組織のプライマリネームサーバからのみ受け取るように設定することである。ただし、一連のホストのあいだのすべての通信で TSIG を使用するように設定している場合、NOTIFY メッセージでも TSIG が使用される。

6.5 脅威のまとめ

表 6-1 に、DNS トランザクションとそれらに対する脅威とセキュリティ目標、および目標を達成するための IETF のセキュリティメカニズムの仕様をまとめる。

表6-1. DNS トランザクションに対する脅威とセキュリティ目標

DNS トランザクション	脅威	セキュリティ目標	IETF セキュリティ仕様
DNS クエリ／レスポンス	(a)偽造レスポンスまたは偽装レスポンス (b) レスポンス内のレコード（RR）の削除 (c) ワイルドカードの展開ルールの間違った適用	(a) データ源の真正性確認 (b) データ完全性の検証	DNSSEC
ゾーン転送	(a) サービス運用妨害 (b) メッセージの改ざん	(a) 相互認証 (b) データ完全性の検証	TSIG
動的更新	(a) 無許可の更新 (b) メッセージの改ざん (c) リプレイ攻撃	(a) 相互認証 (b) データ完全性の検証 (c) 署名付きタイムスタンプ	TSIG または SIG(0)
DNS NOTIFY	(a) 偽りの通知	(a) 負荷の増大によるサービス運用妨害を回避すること	どのホストからこのメッセージを受信できるかを指定する TSIG または SIG(0)

7. DNS ホスティング環境のセキュリティ保護のためのガイドライン

DNS ホスティング環境のセキュアな設定のためのガイドラインは、次の項目に分類される。

+ DNS ホストプラットフォームのセキュリティ保護
+ DNS ソフトウェアのセキュリティ保護
+ ゾーンファイルの内容制御

ガイドラインのほとんどは、BIND DNS ネームサーバソフトウェアを念頭に提供されている。

7.1 DNS ホストプラットフォームのセキュリティ保護

ネームサーバソフトウェアが稼動するプラットフォームは、適切にセキュリティ保護された OS 上に構築されるべきである。DNS のほとんどは、UNIX 系または Windows 系の OS で稼動する。このようなシナリオの下では、次のことを確実にする必要がある。

+ OS の最新のパッチがインストールされている。
+ ネームサーバソフトウェアに該当するアプリケーションプロファイルに関して確認されている脆弱性[18]に基づき、OS の設定について CERT®/CC や NIST の NVD メタベースによって公表されている推奨対策がとられている。特に、ネームサーバソフトウェアを実行するホストは他のサービスを提供してはならず、DNS トラフィックにのみ応答するように設定すべきである。つまり、これらのホストに対して唯一許可される送受信メッセージは、53/udp と 53/tcp でなければならない。

7.2 DNS ソフトウェアのセキュリティ保護

DNS ソフトウェアの保護策には、バージョンの選択、パッチのインストール、限定的な権限での実行、実行環境での他のアプリケーションの制限、各機能に対する専用インスタンスの用意、ソフトウェアがインストールされるホストの組み合わせの制御、ネットワーク内の配置、ゾーンファイルデータの論理的／物理的分割またはクライアントの種類ごとに 2 つのネームサーバソフトウェアインスタンスを実行することによる情報の露出制限などが含まれる。

7.2.1 最新バージョンのネームサーバソフトウェアの実行

より新しいバージョンのネームサーバソフトウェアには、BIND ソフトウェアの場合は特に、以前のバージョンで見つかった脆弱性は存在しないのが一般的である。なぜなら、より新しいバージョンにはそうした脆弱性に対処する設計変更が施されているためである。もちろん、こうした脆弱性はすでに悪用されており（つまり何らかの形態の攻撃が行われた）、悪用の性質に関する十分な情報がもたらされている。つまり、理論上は最新バージョンが最も安全なバージョンであるため、最新バージョンの BIND ソフトウェアを実行することが業務判断としては賢明である。ソフトウェアが最新バージョンであったとしても、それをデフォルトモードで実行するのは安全ではない。セキュリティ管理者は、最新バージョンにおける新しいセキュリティ設定についてよく理解しておく必要がある。

インストール環境によっては、BIND ソフトウェアの最新バージョンにすぐには切り替えられないこともある。そのような状況では、管理者は運用中のバージョンで確認されている脆弱性と、それに関するセキュリティパッチ[19]を見逃さずに対応する必要がある。

チェックリスト項目 1：ネームサーバソフトウェアのアップグレード版をインストールしたら、新しいセキュリティ機能を利用するために管理者は設定パラメータに対して必要な変更を行うこと。

チェックリスト項目 2：最新バージョンを実行しているか旧バージョンを実行しているかにかかわらず、管理者は組織で運用中のバージョンの脆弱性、悪用行為、セキュリティ修正、パッチについて認識していること。次の措置が推奨される。

– 「bind-announce」という ISC のメーリングリストを購読する。

– BIND の脆弱性情報ページ（*http://www.isc.org/products/BIND/bind-security.html*）を定期的に参照する。

– CERT®/CC の Vulnerability Notes Database（*http://www.kb.cert.org/vuls/*）や NIST の NVD メタベース（*http://nvd.nist.gov/*）を参照する。

7.2.2　BIND のバージョンクエリの無効化

BIND の機能に、ある特別なクエリがサーバに送信されると、実行中のサーバデーモンのバージョン番号を返す機能がある。このクエリは、クエリタイプ TXT、クエリクラス Chaos（CH）として、文字列「version.bind」を問い合わせるものである。この情報は、既知の弱点を持った BIND の特定のバージョンを探している攻撃者に利用される可能性がある。BIND の設定ファイル（`/etc/named.conf`）に次のコマンドを追加することにより、この種のクエリを拒否するように BIND を設定できる。

```
options {
      version none;
};
```

チェックリスト項目 3：システムで実行中の BIND のバージョンに関する情報が公開されるのを防ぐために、「version.bind」を求めるクエリを拒否するようにネームサーバを設定すること。

7.2.3　限定された権限でのネームサーバソフトウェアの実行

ネームサーバソフトウェアが特権ユーザ（たとえば UNIX システムの root）として実行されている場合、ソフトウェアへのいかなる侵入も、ネームサーバプラットフォームにあるリソースに対して致命的な結果をもたらす可能性がある。具体的には、ソフトウェアに侵入したハッカーは無制限のアクセス許可を取得し、それにより任意のコマンドの実行や任意のファイルの変更または削除を行える可能性がある。したがって、侵入から生じる損害を食い止めるために、アクセス許可が指定のディレクトリに限定されている非特権ユーザとしてネームサーバソフトウェアを実行する必要がある。

BIND の chroot コマンドを使用して、ネームサーバソフトウェアを実行するべき適切なユーザ名を指定することができる。これが、この手法が「chroot 監獄内での DNS サーバの実行」として知られている理由である。以下にコマンドの例（コマンドラインからの入力）を示す。

```
chroot -u named –g other –t /var/named
```

オプションの意味は次のとおりである。

> *–u* には、ユーザ ID を指定する。ネームサーバソフトウェアは、開始後にこの ID に変更される。このユーザアカウントは、chroot コマンドの発行前に作成しておくべきである。
>
> *–g* には、グループ ID を指定する。ユーザ ID はこのグループに割り当てられる。*–g* を指定しなければ、ネームサーバはユーザ ID のプライマリグループの下で実行される。
>
> *–t* には、ネームサーバソフトウェアの所有者が権限を持つディレクトリを指定する。

7.2.4 ネームサーバソフトウェアの隔離

DNS ソフトウェア（BIND など）がセキュアな OS で実行されていたとしても、そのプラットフォーム上のほかのソフトウェアプログラムの脆弱性により、DNS ソフトウェアのセキュリティが侵害される可能性がある。したがって、DNS ソフトウェアが稼動するプラットフォームには、OS とネットワークサポートに必要なプログラム以外は置かないようにする必要がある。

7.2.5 各機能専用のネームサーバインスタンスの設定

権威ネームサーバは、自身のゾーンファイルから RR を提供する。この機能は*権威機能*と呼ばれる。キャッシュから（直接的に、あるいは反復クエリを通じて動的にキャッシュを構築することにより）RR を提供することを*名前解決機能*と呼び、これがリゾルビィングネームサーバのレスポンスの提供方法である。ネームサーバインスタンスは、権威ネームサーバかリゾルビィングネームサーバ、あるいはその両方として設定できる。しかし、キャッシュ汚染（6.1 項参照）などの攻撃のため、リゾルビィングネームサーバは権威ネームサーバのセキュリティポリシーとは異なるセキュリティポリシーの下で実行すべきである。したがって、ネームサーバインスタンスは常に、権威ネームサーバまたはリゾルビィングネームサーバのどちらかとして設定する必要がある。

権威ネームサーバは、権威を持っている情報の対象となるゾーンの名前解決を提供することのみを目的としている。したがって、セキュリティポリシーは、このタイプのネームサーバでは再帰を無効にする必要がある。再帰を無効にすると、権威ネームサーバは、他のネームサーバに代わってクエリを送信することや、レスポンスを使ってキャッシュを構築することができなくなる。この機能を無効にすることで、キャッシュ汚染の脅威は排除される。BIND では、BIND 設定ファイルで options 文を次のように使用することによって再帰を無効にできる。

```
options {
      recursion no;
};
```

リゾルビィングネームサーバは、内部クライアントに対して名前解決サービスを提供する（クライアントに代わって名前解決クエリを処理する）ことのみを目的としている。したがって、ネームサーバソフトウェアの設定ファイルの各種設定オプションを通じて、この種の対話（*トランザクション*ともいう）を指定のホストに限定することにより、リゾルビィングネームサーバを確実に保護できる。

7.2.6 指定外のホストからのネームサーバソフトウェアの削除

DNS ソフトウェアは、ネームサーバとして指定されていないホスト上では実行も存在もしてはならない。このようなケースは DNS BIND ソフトウェアの場合は起こり得る。UNIX の多くのバージョン（Solaris や Linux も含めて）で BIND がデフォルトでインストールされるという事実があるためである。したがって、セキュリティ監査の一環として組織のワークステーションおよびサーバ上のソフトウェア

の目録を作成する際に、BIND のインストールを探し、ネームサーバとして機能していないホストからは削除する必要がある。

7.2.7 ネットワーク上および地理上での権威ネームサーバの分散

ほとんどの組織には、1 つの権威プライマリネームサーバと、複数の権威セカンダリネームサーバが存在する。組織においてこれらの権威ネームサーバは、異なるネットワークセグメントに設置することが重要である。分散することにより、特定のルータやスイッチが故障した場合だけでなく、あるネットワークセグメント全体が攻撃に巻き込まれている場合にも権威ネームサーバの可用性が保証される。ネットワークベースの分散だけでなく、地理的にも権威ネームサーバを分散させる必要がある。つまり、異なるネットワークセグメントに設置するだけでなく、権威ネームサーバをすべて同じ建物内には設置しないようにする。いくつかの組織が採用している 1 つの方法は、いくつかの権威ネームサーバを自分たちの施設内に置き、残りを ISP のデータセンターや提携組織に設置するやり方である。

チェックリスト項目 4：組織の権威ネームサーバはネットワーク上と地理上の両方で分散させるべきである。ネットワークベースの分散には、すべてのネームサーバを 1 つのルータやスイッチ、1 つのサブネット、あるいは 1 つの専用回線のもとに設置しないようにすることが含まれる。地理的な分散には、すべてのネームサーバを物理的に同じ場所には設置せずに、少なくとも 1 つのセカンダリサーバを遠隔地に設置することが含まれる。

7.2.8 ゾーンファイルの分割による情報露出の制限

組織の権威ネームサーバは、外部と内部の両方のクライアントからリクエストを受信する。多くの場合、外部クライアントが受信する必要があるのは、公開のサービス（公開 Web サーバ、メールサーバなど）に関連する RR のみである。内部クライアントは、公開のサービスと内部ホストに関連する RR を受信する必要がある。したがって、これらの RR を提供するゾーン情報を、外部クライアント用と内部クライアント用にそれぞれ物理的に異なるファイルに分けることができる。ゾーンファイルのこのような実装方法は、*分割 DNS* と呼ばれる。

分割 DNS にはいくつか不利な点がある。1 つは、リモートホスト（たとえば出張先からノート型 PC を使って組織に接続する場合など）が内部の名前解決 DNS サーバを使用できないことがあるため、内部ホストを認識できない場合があることである。もう 1 つは、内部ホスト情報が（事故や攻撃により）ファイアウォールの外側に漏れる可能性が生じることである。結果として分割 DNS を設定する目的にそぐわなかったり、内部と外部のホストの FQDN が同じながらも IP アドレスが異なるために混乱を招いたりする。分割 DNS を適切なアクセス制御の代替手段として捉えてはならない。

チェックリスト項目 5：分割 DNS の実装の場合、少なくとも 2 つの物理ファイル、すなわち*ビュー*が存在すること。1 つは、ファイアウォールの内側に設定されているホスト専用に名前解決を提供する。そのファイルは、ファイアウォールの外側のホストの RRset も含むことができる。もう 1 つのファイルまたはビューは、ファイアウォールの内側のホストではなく、ファイアウォールの外側または DMZ に設置されているホストの名前解決を提供する必要がある。

BIND を使用して分割 DNS を設定するには、BIND 設定ファイルで view 文を使用する。たとえば、内部クライアント用にゾーン「sales.mycom.com」の権威ビューを設定するには次のように記述する。

```
view “insider” {
     match-clients { internal_hosts; };
     recursion no;
     zone sales.mycom.com {
          type master;
          file “sales_internal.db”;
     };
};
```

この文のファイル「sales_internal.db」には、ゾーン「sales.mycom.com」の権威を持つ情報が含まれており、その情報の提供先を「internal_hosts」というアドレスリストから送信されたクエリに限定している。8.1.1 項で、このリストの設定方法の詳細を述べる。ゾーンは同じであってもネットワーク外部から送信されるクエリに対して外部ホストによるビューのみを返すようにするには、同じ設定ファイルで次の view 文を使用する。

```
view “outsider” {
     match-clients { any; };
     match-destinations { public_hosts; };
     recursion no;
     zone sales.mycom.com {
          type master;
          file “sales_external.db”;
     };
};
```

この文は、直前に取り上げた文にひじょうによく似ているが、ゾーンビューを持つファイルは「sales.external.db」であり、クライアントリストは「any」と指定されている。これは、ネットワークの外側と内側からのクエリは両者とも、「sales.mycom.com」の同じ外部ビューを見ることができることを意味する。これらの文により、内部クライアントは「sales.mycom.com」の内部ホストと外部ホストの両方を見ることができ、外部ホスト（同じネットワーク上にはない）は宛先アドレスがアドレスリスト「public_hosts」と一致するゾーンの外部ビューに含まれている DNS 情報のみをみることができる。

7.2.9 各種クライアント用にネームサーバを導入することによる情報露出の制限

組織では、さまざまなタイプのクライアントに対して同じ権威ネームサーバの組み合わせでサービスを提供するのではなく、権威ネームサーバの 2 つの異なる組み合わせを設定することも考えられる。1 つの組み合わせは*外部ネームサーバ*と呼ばれ、DMZ 内に設置できる。これらは外部クライアントがアクセスできる唯一のネームサーバであり、公開のサービスを提供するホスト（外部 Web ページや B2C サービスを提供する Web サーバ、メールサーバなど）に関する RR を提供する。もう 1 つの組み合わせは*内部ネームサーバ*と呼ばれ、ファイアウォールの内側に設置される。これらは外側からは到達できないように設定する必要があり、そのため内部クライアント専用にネームサービスを提供する。両方のアーキテクチャの選択肢（すなわちネームサーバの 2 つの異なる組み合わせと、分割 DNS）の目的は、外側の世界に内部ホストの IP アドレスが知られるのを防ぐことである。この設定は、BIND にあるビュー機能のない DNS サーバソフトウェアを使用している組織にとっては、利用可能な唯一の選択肢であると考えられる。

7.3　ゾーンファイルの内容制御

5.7 項で述べたように、DNS ゾーンファイルの内容制御のための唯一の保護策は、ゾーンファイルの完全性チェッカを使用することである。ゾーンファイルの完全性チェッカを用いた完全性チェックの有効性は、チェッカに組み込まれている制約のデータベースに依存する。したがってその導入手順は、正しいロジックでの制約の策定から成り、その制約の論理述語の真偽値を唯一決定するのは、さまざまな RRType の形式の特定の鍵フィールドのパラメータ値である。これらのパラメータ値の選択が導入ガイドラインを形成する。これについては、セクション 10 で解説する。

7.4　推奨事項のまとめ

以下に、このセクションで取り上げた主な推奨事項をまとめる。

+ **チェックリスト項目 1**：ネームサーバソフトウェアのアップグレード版をインストールしたら、新しいセキュリティ機能を利用するために管理者は設定パラメータに対して必要な変更を行うこと。

+ **チェックリスト項目 2**：最新バージョンを実行しているか旧バージョンを実行しているかにかかわらず、管理者は組織で運用中のバージョンの脆弱性、悪用行為、セキュリティ修正、パッチについて認識していること。次の措置が推奨される。

  – 「bind-announce」という ISC のメーリングリストを購読する

  – BIND の脆弱性情報ページ（*http://www.isc.org/products/BIND/bind-security.html*）を定期的に参照する。

  – CERT®/CC の Vulnerability Notes Database（*http://www.kb.cert.org/vuls/*）や NIST の NVD メタベース（*http://nvd.nist.gov/*）を参照する。

+ **チェックリスト項目 3**：システムで実行中の BIND のバージョンに関する情報が公開されるのを防ぐために、「version.bind」を求めるクエリを拒否するようにネームサーバを設定すること。

+ **チェックリスト項目 4**：組織の権威ネームサーバはネットワーク上と地理上の両方で分散させるべきである。ネットワークベースの分散には、すべてのネームサーバを 1 つのルータやスイッチ、1 つのサブネット、あるいは 1 つの専用回線のもとに設置しないようにすることが含まれる。地理的な分散には、すべてのネームサーバを物理的に同じ場所には設置せずに、少なくとも 1 つのセカンダリサーバを遠隔地に設置することが含まれる。

+ **チェックリスト項目 5**：分割 DNS の実装の場合、少なくとも 2 つの物理ファイル、すなわち*ビュー*が存在すること。1 つは、ファイアウォールの内側に設定されているホスト専用に名前解決を提供する。そのファイルは、ファイアウォールの外側のホストの RRset も含むことができる。もう 1 つのファイルまたはビューは、ファイアウォールの内側のホストではなく、ファイアウォールの外側または DMZ に設置されているホストの名前解決を提供する必要がある。

8. DNSトランザクションのセキュリティ保護のためのガイドライン

セクション 6 では、さまざまな DNSトランザクションに対する脅威、セキュリティ目標、保護策の概略を述べた。このセクションでは、これらの保護策の実装に必要な手順と、実装に合わせた運用面でのベストプラクティスについて説明する。以下に、表 6-1 の保護策をまとめ、保護策の種類ごとに見出しを付けて詳しく説明する。

+ **IP アドレスに基づいてトランザクションエンティティを制限する。**この種の実装では、ネームサーバソフトウェアが提供している該当するアクセス制御文で IP アドレスを指定することにより、DNSトランザクションに参加する DNS ネームサーバとクライアントを信頼できる一連のホストに限定する。IP に基づくアクセス制御文による保護策は、IP の偽装などの攻撃によって迂回される可能性がある。したがって、この解決策は脅威の影響の度合いが高い DNS クエリ／レスポンス、ゾーン転送、動的更新のトランザクションに対しては推奨されない。しかし、唯一の脅威が偽りの通知（ゾーン転送さえ引き起こさない可能性がある）である DNS NOTIFYトランザクションに対しては、IP アドレスに基づくアクセス制御があれば十分であろう。この解決策は、一般的には推奨されないが、ハッシュベースのメッセージ認証コードによるトランザクション保護が実装される場合、名前付きの鍵に基づくホストの識別に同様のアクセス制御文が使用されるため、8.1 項で IP アドレスを使用したアクセス制御の仕組みについて説明する。このアプローチは、すべての DNSトランザクションに対して実装されてきた。

+ **ハッシュベースのメッセージ認証コード（TSIG 仕様）によるトランザクション保護。**この保護策では、トランザクション保護はハッシュベースのメッセージ認証コード（HMAC）の生成と検証を通じて実施される。これらのコードは RRType が TSIG の特別な RR に埋め込まれるため、HMAC を使用した DNSトランザクションの保護についてまとめた仕様は、DNS コミュニティでは TSIG と呼ばれている。TSIG の仕様は、RFC 2845 および 3007 に記述されている。ゾーン転送および動的更新のトランザクションの保護に対する TSIG 仕様の適用については、8.2 項で述べる。

+ **非対称ディジタル署名（DNSSEC 仕様）によるトランザクション保護。**この保護策は通称 DNS セキュリティ拡張（DNSSEC）と呼ばれ、一連の RFC（4043、4044、4045）に記述されている。DNSSEC が提供する中核サービスは、データ源の真正性確認と完全性の保護である。DNSSEC は主に、DNS クエリ／レスポンストランザクションによって取得する DNS 情報をセキュリティ保護するために使われる。DNSSEC の導入に関する問題については、セクション 9 で述べる。

8.1 IP アドレスに基づくトランザクションエンティティの制限

BIND 9.x など、DNS ネームサーバ実装のいくつかはアクセス制御文を提供しており、これを通じて対象の DNSトランザクションに参加できるホストを指定できる。ホストは、それらの文の中で IP アドレスまたは IP サブネット参照（*IP プレフィックス*とも呼ばれる）を使用して指定できる。

この IP アドレスまたは IP プレフィックスを含んでいるリストを、*アドレスマッチリスト*と呼ぶ（アドレスマッチリストは、8.1.1 項で述べるように、IP アドレスと IP プレフィックス以外の情報も含めることができる）。アドレスマッチリストは、BIND 設定ファイルにおいて使用できる各種のアクセス制御文の中で引数として使用される。DNSトランザクションのタイプごとに異なるアクセス制御文がある。各種のアクセス制御文の構文と、それらが使用される DNSトランザクションを表 8-1 に示す。

表 8-1. DNS トランザクション用のアクセス制御文の構文

アクセス制御文の構文	DNS トランザクション
allow-query { address_match_list }	DNS クエリ／レスポンス
allow-recursion { address_match_list }	再帰的クエリ
allow-transfer { address_match_list }	ゾーン転送
allow-update { address_match_list }	動的更新
allow-update-forwarding { address_match_list }	動的更新
allow-notify { address_match_list }	DNS NOTIFY
blackhole { address_match_list }	ブラックリストに載せるホスト

以下に、各アクセス制御文の用途を示す。

+ **allow-query:**ネームサーバ（ネームサーバ全体またはネームサーバ内の特定のゾーン）に対するクエリが許可されるホストのリストを指定する。

+ **allow-recursion:**ネームサーバ（ネームサーバ全体またはネームサーバのサービスを受けている特定のゾーン）に対する再帰的クエリが許可されるホストのリストを指定する。

+ **allow-transfer:**ネームサーバ（ネームサーバ全体またはネームサーバ内の特定のゾーン）に対するゾーン転送リクエストが許可されるホストのリストを指定する。この文は主に、マスタネームサーバの設定に必要となる。

+ **allow-update:**動的更新リクエストが許可されるホストのリストを指定する。

+ **allow-update-forwarding:**動的更新リクエストの転送が（リクエストの送信元に関係なく）許可されるホストのリストを指定する。

+ **allow-notify:**ゾーンファイルに変更があったことを示す DNS NOTIFY メッセージの送信が許可されるホストのリストを指定する。このリストは、セカンダリスレーブネームサーバの設定にのみ関係する。

+ **blackhole:**このネームサーバに関するいかなるトランザクションも開始できないように（禁止する）ブラックリストに載せるホストのリストを指定する。この文は、サーバ全体を対象とする ACL 文である options 文でのみ使用される。

これらのアクセス制御文は、実際には、BIND 9.x の設定ファイルの *options* 文と *zone* 文のコンテキストで使うことのできる副文である（ただし **blackhole** を除く）。zone 文のなかで使った場合は、対象となる特定のゾーンに該当する DNS トランザクションのアクセス制限を指定する。options 文の一部として使った場合は、ネームサーバ全体に該当する DNS トランザクションのアクセス制限を指定する（ネームサーバは複数のゾーンをホストしていることが考えられるため）。

8.1.1 DNS クエリ／レスポンストランザクションエンティティの制限

allow-query 文（DNS クエリの送信元として受け付ける IP アドレス／サブネットを明示することで、DNS クエリ／レスポンストランザクションに対する制限を指定する）のサーバレベルとゾーンレベルの両方の使用例を示す（ゾーンは example.com）。

```
options {
     allow-query { 254.10.20.10; 239.10.30.29/25; };
};
```

```
zone “example.com.” {
      type master;
      file “zonedb.example.com”;
      allow-query { 192.249.249.1; 192.249.249.4; };
};
```

options 文や zone 文のなかで IP アドレスと IP プレフィックスのリストを指定すると、設定ファイルが乱雑になる可能性がある。さらに、IP アドレスと IP プレフィックスのリストは、ネームサーバ内の多くのアクセス制御文で同じであることが考えられ、リストに対して追加や削除が行われた場合に誤りが生じることもある。これらの問題を避けるために、BIND では*アクセス制御リスト*（ACL）と呼ばれる名前付きのアドレスマッチリストを作成できるようになっている。アクセス制御文の（アドレスマッチリスト引数のなかの）IP アドレス／IP プレフィックスのリストの代わりに ACL を使用できる。

ACL は、BIND 9.x では acl 文を使用して作成する。acl 文の一般構文は、次のとおりである。

```
acl acl-list-name {
      address_match_list
};
```

acl-list-name はユーザ定義文字列である（たとえば internal_hosts）。address_match_list には、IP アドレス、IP アドレスプレフィックス（サブネットを表す）、または暗号鍵のリストを指定できる。address_match_list に IP アドレスとサブネット参照を指定した acl 文の例を以下に示す。この例では、254.10.20.10 はホストの IP アドレスを表し、IP プレフィックス 239.10.30.0/24 はクラス C のサブネットを表す。

```
acl “internal_hosts” {
      254.10.20.10;
      239.10.30.0/24;
};
```

先述の options 文と zone 文において、IP アドレス／IP プレフィックスのリストの代わりに ACL、すなわち internal_hosts を使用すると、次のようになる。

```
options {
      allow-query { internal_hosts; };
};

zone “example.com.” {
      type master;
      file “zonedb.example.com”;
      allow-query { internal_hosts; };
};
```

アクセス制御文のアドレスマッチリストパラメータには、次の値を含めることができる。

+ IP アドレスまたは IP アドレスのリスト
+ IP プレフィックスまたは IP プレフィックスのリスト
+ ACL
+ 上記の 3 つの組み合わせ

ACL の定義は、DNS トランザクション制限の設定において重要な要素を形成する。したがって、DNS 管理者は、さまざまな DNS トランザクションに関する ACL を定義し、作成することがよい運用の仕方である。

> **チェックリスト項目 6**: 管理者は、各種の DNS トランザクションのそれぞれに対して、信頼できるホスト（またはブラックリストに載せるホスト）の名前付きリストを作成することが推奨される。一般に、次の種類の役割のホストは、該当する ACL に含めることを考慮すること。
>
> – 組織内の各ゾーンで定義される DMZ ホスト
>
> – ゾーン転送の開始が許可されているすべてのセカンダリネームサーバ
>
> – 再帰的クエリの実行が許可されている内部ホスト

IP アドレス、IP プレフィックス、ACL のほかに、アクセス制御文のアドレスマッチリストパラメータには、次の特殊な値を指定できる。

+ **none:**どのホストとも一致しない
+ **any:**すべてのホストと一致する
+ **localhost:**ネームサーバが稼動しているサーバ上のすべての IP アドレスと一致する
+ **localnets:**ネームサーバが稼動しているサーバ上のすべての IP アドレスとサブネットマスクと一致する

以下に、ACL を作成するコマンドの例と、options 文および zone 文での ACL の使用例をさらにいくつか示す。

```
acl “local_hosts” {
      254.10.20.10;
      239.10.30.29/25;
};

acl “fake-net” {
      0.0.0.0/8;
      1.0.0.0/8;
};

options {
      allow-query { any; };
      blackhole { fake-net; };
};

zone “example.com.” {
      type master;
      file “zonedb.example.com”;
      allow-query { local_hosts; };
};
```

上記の named.conf（抜粋）では、local_hosts と fake-net という 2 つの ACL を指定している。あらゆるホストからの DNS クエリがサーバレベルで許可されている。fake-net に含まれているホストからのト

ランザクションは許可されていない。zone 文（ゾーン固有の指定）の下に指定されている制限はすべて、options 文（サーバ全体の指定）の下に指定されている制約よりも優先するため、ゾーン example.com に対するクエリは、ACL local_hosts に含まれているホストのみが開始できる。

ACL 文で鍵素材を使用することもできる。これは、共有鍵（または鍵ペア）を認識（および使用）しているホストだけがやり取りできることを示す。ACL で鍵がどのように使われるかについては、8.2.3 項で説明する。

8.1.1.1　再帰的クエリの制限（DNS クエリ／レスポンスの特殊ケース）

権威ネームサーバは、自身のデータから名前解決サービスを提供し、このサービスを任意の DNS クライアントに提供するものと想定されている。したがって、限定された一連のホストからのクエリを受け付けるように権威ネームサーバを設定するのは無意味である。権威ネームサーバに対する現実的なセキュリティ保護は、クエリの再帰機能を無効にして、権威ネームサーバがほかの（侵害されている可能性のある）ネームサーバに対してクエリを行って自身のキャッシュを汚染しないようにすることである。ローカルの名前解決／再帰的ネームサーバを、内部ホストからのみクエリを受け付けるように設定して、キャッシュ汚染だけでなくサービス運用妨害攻撃からも保護できる。しかし、権威を持つサービスと名前解決サービスに対して別々の専用のサーバを割り当てるのが経済的に不可能で、リゾルビィングネームサーバが 1 つまたは複数のゾーンの権威サーバとして機能しなければならない状況もある。このような場合、BIND 9.x ネームサーバにおいては次のような方策をとることができる。

+ サーバによって受け付けられるすべてのクエリを、内部クライアントのうち指定した一連の IP アドレスに制限したあと、この指定を権威を持つゾーンに対してのみオーバーライドして、すべての DNS クライアントがそのゾーンのリソースの情報を取得できるようにする。
+ 直接の設定オプションを通じて、再帰的クエリを、内部クライアントのうち指定した一連の IP アドレスに制限する。
+ ビューを定義することにより、異なるクライアントに異なるレスポンス（データ）を提供する。

<u>サーバレベルの制限に、権威ゾーンに対するオーバーライド:</u>

この方策では、次のように、ネームサーバへのクエリの送信を許容する内部クライアントの組み合わせを acl 文によって指定する。

```
acl internal_hosts {192.158.43.3; 192.158.43.6;
192.158.44.56;};
```

サーバ全体に関するオプションでは、すべてのクエリをこれらのクライアントに限定する。

```
options {
      allow_query { internal_hosts; };
};
```

このオプションを、このネームサーバが権威を持っているゾーンを指定することによってオーバーライドできる（その結果、すべてのクライアントからのそのゾーンに関するクエリを許可する）。

```
zone “example.com” {
      type master;
      file “zonedb.example.com”;
```

```
        allow_query { any; };
};
```

<u>すべての再帰的クエリを指定した一連の IP アドレスに限定:</u>

次のようにサーバ全体に対する制限を指定する:

```
options {
        allow_recursion { internal_hosts; };
};
```

<u>ビューを通じての再帰クエリの制限:</u>

ビューを作成する目的は、再帰的クエリがサポートされるクライアント（IP アドレスにて指定）およびゾーンと、再帰的クエリがサポートされないクライアントおよびゾーンの組み合わせによって構成される、論理的なパーティションを作成することである。次の例では、ビューrecursion_view を使用して再帰的クエリの送信が許可される IP アドレスとゾーンの範囲を定義している。no_recursion_view は、再帰的クエリが許可されない対象である。

```
view recursion_view {
        match-clients { internal_hosts; };
        recursion yes;
};

view no_recursion_view {
        match-clients { any; };
        recursion no;
};
```

8.1.2 ゾーン転送トランザクションエンティティの制限

権威ネームサーバ（特にプライマリネームサーバ）には、アクセス制御文の allow-transfer を使用して、ゾーン転送リクエストの送信元として受け付けるホストのリストを指定する必要がある。これらの制限は、サービス運用妨害の脅威と、内部リソース情報の無差別の配布から生じる悪用の可能性に対処するものである。知る必要があるかを基準にすると、ゾーンファイルを定期的に更新する必要のあるネームサーバは、セカンダリネームサーバだけである。したがって、プライマリネームサーバからのゾーン転送は、セカンダリネームサーバに限定すべきである。セカンダリネームサーバ側では、ゾーン転送は完全に無効にすべきである。allow-transfer 文へのアドレスマッチリスト引数は、セカンダリネームサーバの IP アドレスとステルスセカンダリサーバの IP アドレスで構成すべきである。

以下に、3 つのセカンダリネームサーバを含んだ「valid_secondary_NS」という ACL を作成する例を示す。

```
acl “valid_secondary_NS” {
        224.10.229.5;
        224.10.235.6;
        239.10.245.25;
};
```

zone 文と options 文のなかで allow-transfer 文を使用できる。この文を zone 文のなかで使用すると、当該ゾーンにおけるゾーン転送を制限でき、options 文の中で使用すると、ネームサーバのすべてのゾーンにおけるゾーン転送を制限できる。

以下に、サーバレベルの allow-transfer 文を示す。

```
options {
      allow-transfer { “valid_secondary_NS”; };
};
```

以下に、ゾーンレベルの allow-transfer 文を示す。

```
zone “example.com” {
      type master;
      file “zonedb.example.com”;
      allow-transfer { “valid_secondary_NS”; };
};
```

これらの文は、プライマリネームサーバに適用される。セカンダリネームサーバとステルスセカンダリネームサーバでは、ゾーン転送は次のように無効にすべきである。

```
zone “example.com” {
      type slave;
      masters { 224.239.5.1; } ;
      file “zonedb_bak.example.com”;
      allow-transfer { none; };
};
```

8.1.3 動的更新トランザクションエンティティの制限

ゾーンファイルの動的更新は、そのゾーンのプライマリネームサーバ上（すなわちマスタゾーンファイルが置かれている場所）にあるゾーンファイルのコピーのみを対象とすることができる。BIND 8 と BIND 9 では、デフォルトでは動的更新ができないようになっている。BIND で次のどちらかの文を使用して、動的更新の許可または禁止を指定できる。

+ allow-update
+ update-policy（BIND 9 バージョンでのみ使用可能）

この 2 つの文は、サーバレベルではなくゾーンレベルでのみ指定できる。つまり、これらの文は zone 文のなかの副文である。allow-update 文により、IP アドレスと共有秘密（*TSIG 鍵*とも呼ばれる）に基づいて動的更新の制限を指定できる。この項では、IP アドレスだけを用いた allow-update 文の使い方を取り上げる。TSIG 鍵を用いた allow-update 文の使い方は、8.2.6 項で説明する。

update-policy 文では、TSIG 鍵にのみ基づいて動的更新の制限を指定できるが、より細かい粒度での指定が可能である。allow-update 文は、ゾーンの全レコードを対象とした更新アクセス許可を意味するが、update-policy 文は更新アクセス許可を 1 つまたは複数の指定した RRType（たとえば A RR）に制限する場合に使用できる。

allow-update 文を使用するには、アドレスマッチリストを作成する必要がある。以下に、IP アドレスが 1 つのみの ACL DU_Allowed_List を作成するコマンドを示す。

```
acl “DU_Allowed_List” {
      192.249.12.21;
};
```

この ACL DU_Allowed_List(ゾーン example.com の内容を更新するために、動的更新リクエストの送信が許可されるホストの IP アドレスのリスト)を、zone 文内の allow-update 文のなかで次のように使用する。

```
zone “example.com” {
      type master;
      file “zonedb.example.com”;
      allow-update { “DU_Allowed_List”; };
};
```

動的更新リクエストは一般に、IP アドレスをホストに動的に割り当てる DHCP サーバのようなホストから発信される。IP アドレスを新しいホストに割り当てたら、そのホストは(A RR を作成して)FQDN から IP アドレスへのマップ情報と、(PTR RR を作成して)アドレスから FQDN へのマップ情報を、そのゾーンのプライマリ権威ネームサーバに格納する必要がある。この情報の作成が動的更新を通じて行われる。

8.1.4 DNS NOTIFY トランザクションエンティティの制限

サーバ間のゾーン転送を設定したら、セカンダリネームサーバがゾーンファイルデータへの変更について通知メッセージを通じて知らされることを確認した方がよい。デフォルトでは、プライマリネームサーバがゾーンファイル内で変更を検出するたびに通知メッセージが送信される。プライマリネームサーバは、ゾーンの NS RRSet にリストされているすべてのネームサーバに DNS NOTIFY メッセージを送る(これらのサーバは、このゾーンにおいて認識されているセカンダリネームサーバであるため)。DNS 管理者は、この通知を有効にしておくべきである。こうすることで、更新をセカンダリネームサーバに迅速に伝えられるようになるためである。ただし特定のゾーンに対してこの機能を無効にしておく必要がある場合は、当該ゾーンの zone 文で notify 文を使用する。

```
zone “example.com” {
      type master;
      notify no;
      file “zonedb.example.com”;
};
```

ゾーン管理者が DNS NOTIFY メッセージの送信先として希望するサーバがほかにある場合(たとえば、ステレススレーブサーバ)は、次のように「also-notify」という副文を zone 文に追加し、追加のサーバの IP アドレスをそのパラメータ値として指定する

```
zone “example.com” {
      type master;
      also-notify { 192.168.25.2; };
      file “zonedb.example.com”;
};
```

DNS NOTIFY メッセージの受信者、すなわちセカンダリネームサーバは、デフォルトではプライマリネームサーバからのみ通知メッセージを許可する(先述のとおり、セカンダリネームサーバはそのプライマリネームサーバのことを zone 文内の master 文を通じて認識する)。セカンダリネームサーバがほかのホストからも通知メッセージの受け取りを希望する場合、zone 文に「allow-notify」文を追加し、次のように対象サーバの IP アドレスをこの副文のなかで指定する。

```
zone “example.com” {
      type slave;
      allow-notify { 193.168.25.4; };
```

```
        file "zonebak.example.com";
        masters { 192.168.25.1; };
};
```

8.2 ハッシュベースのメッセージ認証コード(TSIG)によるトランザクション保護

ハッシュベースのメッセージ認証コード(HMAC)を通じてメッセージの送信元とその完全性を認証するプロセスは、TSIG という総称で知られる一連の DNS 仕様を通じて指定される。HMAC という言葉は、鍵付きハッシュ関数を使用することによって生成されるメッセージ認証コードを示す場合と、ハッシュ関数自身を示す場合がある。HMAC は RFC 2104[14][8]で定められており、NIST 文書 FIPS 198 [15]に概要が述べられている[9]。

HMAC 関数は、メッセージ入力と秘密鍵という 2 つのパラメータを使用し、*メッセージ認証コード*(MAC:message authentication code)または*ハッシュ*と呼ばれる出力を生成する。メッセージの送信者は、HMAC 関数を使用して MAC を生成し、この MAC とメッセージを受信者へ送信する。同じ秘密鍵を共有する受信者は、その鍵と送信者が使用した HMAC 関数を使用して、受信したメッセージの MAC を計算する。受信者は次に、計算した MAC と受信した MAC を比較し、2 つの値が一致していればメッセージが正しく受信されたことと、送信者が同じ秘密鍵を共有するユーザのコミュニティに属していることを確認できる。このようにして、メッセージ発信元認証と完全性検証が 1 回のプロセスで実行される。

ハッシュアルゴリズム(ハッシュ関数のプリミティブを形成する)は、任意のサイズのメッセージから固定サイズの MAC(ハッシュ)を生成する。RFC 2845 [12]に定められている TSIG の HMAC 関数が提供するハッシュアルゴリズムはただ1つ、HMAC-MD5 というアルゴリズム識別子を持つ MD5 である。さらに 2 つのハッシュアルゴリズム(SHA-1 および SHA-256)をサポートすることが、TSIG に関する IETF 文書案 [13] において実装者に対して勧告されている。

共有秘密を使用した HMAC を通じてのトランザクション保護は、拡張性のある解決策ではない。TSIG 仕様が主にゾーン転送トランザクションと動的更新トランザクションにのみ使用されているのは、このためである。これらの DNS トランザクションは、同じ管理ドメイン内のサーバ間、あるいは以前にやり取りが確立しているドメイン内のサーバ間で行われる。

DNS メッセージの送信者によって生成された MAC(ハッシュ)値は、その DNS メッセージに追加された *TSIG レコード*と呼ばれる新しい RR に置かれる。この TSIG レコードには、生成されたハッシュのほかに次の情報が含まれる。

+ ハッシュアルゴリズムの名前
+ 鍵名
+ ハッシュが生成された時間(タイムスタンプ)
+ 「誤差」。生成時刻の前後の差分として使用する秒数(通常は 5)。この時間内であれば、その TSIG 署名は正当と見なすべきである。ホスト間のクロックのずれの可能性を考慮するために使用される。

タイムスタンプフィールドは、MAC が生成された時間を示す。このフィールドの目的は、リプレイ攻撃から(システムを)保護することである。リプレイ攻撃では、攻撃者は MAC を含んだパケットを捕

---

[8] RFC 2104、『*HMAC: Keyed-Hashing for Message Authentication*』は http://www.ietf.org/rfc/rfc2104.txtから入手できる。

[9] FIPS 198、『*The Keyed-Hash Message Authentication Code (HMAC)*』は http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdfから入手できる。

捉し、しばらくしてからそれを送信することが考えられる。このようなことが生じないようにするために、受信者は MAC の生成時刻と現在のクロック時刻を見て、その MAC が「誤差」を考慮して計算された「許容可能な有効時間」内に生成されたものかを検証する。

「誤差」フィールドは、MAC 生成時刻後、メッセージを正当とみなせる時間の範囲を示す。MAC 生成ホストと検証側のホストのあいだのクロックのずれ（不一致）を考慮して、MAC 生成時刻に対して「誤差」を適用する（短い秒数の加算または減算を行う）ことによって計算される。

TSIG に基づくセキュアなトランザクションを行うために、送信者は DNS メッセージ全体と秘密鍵のハッシュを計算し、結果をメッセージの最後の TSIG RR のなかでエンコードする。受信者側では、TSIG レコードは DNS メッセージから取り除かれて処理される。受信者が TSIG レコードを用いて受信した DNS メッセージの完全性を確認するプロセスを、*検証*という。検証プロセスでは、ハッシュアルゴリズム名を使用してハッシュ関数を識別し、鍵名を使用して TSIG レコードの有効性を確認するための鍵を識別する。誤差の数値は、署名側と検証側のクロックの不一致の可能性を考慮して、署名の時刻に対して加算または減算される。つまり誤差は、生成時刻に基づいて計算された MAC の有効期間に対する許容限度を示すものである。

TSIG レコードの鍵名を送信する目的は、DNS メッセージの検証者（受信者）が正しい鍵を使って検証できるようにすることである。受信者はまた、その鍵名が実際に送信者と共有している鍵の 1 つであることも確認できる。TSIG レコードの「時刻署名された」フィールド、つまりタイムスタンプフィールドの目的は、メッセージ受信者に MAC の生成時刻を知らせることである。受信者はこの値と受信側システムの現在のクロック時刻を比較し、MAC が TSIG レコード自身の一部として指定された許容時間内に生成されたものであることを確認する。タイムスタンプを使用する目的は、リプレイ攻撃を防ぐことである。現在時刻に対して生成時刻を正しく検証するためには、トランザクション参加者のシステムクロックが同期している必要がある。この目的には、NTP（Network Time Protocol）などのプロトコルを使用できる。

検証プロセスは、受信者による適切な秘密鍵の取得、受信した DNS メッセージの独自のハッシュの生成、生成したハッシュと受信したハッシュ（TSIG レコードにある）の比較から成る。この検証プロセスでは、受信側のネームサーバは次の有効性確認を行う。

+ メッセージが（秘密鍵を共有する）認証された発信元からのものとして検証済みであること。
+ メッセージが送信中に改ざんされていないこと（ハッシュ値の照合によって検証される）。

発信元認証は身元偽装の対策であり、データ完全性チェックは送信中のデータの破損と改ざんに対処するのに役立つ。

BIND バージョン 8.2 は TSIG 機能を導入した最初のバージョンであり、以降のすべてのバージョンにこの機能がある。BIND 9.x の TSIG のサポートには、ゾーン転送と動的更新のトランザクションのセキュリティ保護が含まれている。

DNS トランザクションで TSIG を使用できるように環境を設定するには、次の準備が必要である。

+ DNS トランザクションに参加するネームサーバ（プライマリおよびセカンダリ）のシステムクロックを（NTP などを通じて）同期化する。
+ 十分な乱雑さを持って、必要な長さの鍵を生成できる秘密鍵生成ユーティリティを用意する。鍵ファイル（秘密鍵文字列を含んだファイル）は、トランザクションに参加している 2 つのサーバに対して安全に伝達されなければならない。

+ 適切な文を通じて、鍵情報を設定ファイルにて指定する（たとえば、BIND 9.x の named.conf 設定ファイルの key 文と server 文など）。

鍵生成プロセスについては 8.2.1 項で説明する。鍵を定義するコマンドと、すべての DNS トランザクションに対してその鍵を使用するようにネームサーバに指示するコマンドについては、それぞれ 8.2.2 項および 8.2.3 項で説明する。ネームサーバにおける鍵ファイル作成と鍵定義のためのチェックリストは、8.2.4 項に示す。TSIG で指定されている HMAC を使用したゾーン転送トランザクションと動的更新トランザクションの保護については、8.2.5 項と 8.2.6 項でそれぞれ取り上げる。

8.2.1 鍵の生成

認証されたメッセージを通じてゾーン転送（リクエストとレスポンス）を許可するには、ネームサーバの各ペアに対して鍵を生成する必要がある。その鍵は、動的更新、DNS クエリ、レスポンスなどの他のトランザクションのセキュリティ保護にも使用できる。DNSSEC で使われるほとんどの鍵生成ユーティリティが生成するバイナリ鍵文字列は、base 64 でエンコードされている。BIND 9.x で鍵を生成するプログラムは dnssec-keygen である。dnssec-keygen プログラムを呼び出して秘密鍵を生成するコマンドの例（このプログラムでは、公開鍵などほかの種類の鍵も生成できる）を以下に示す。

```
dnssec-keygen -a HMAC-SHA1 –b 128 -n HOST ns1-ns2.example.com.
```

コマンドオプション（パラメータ）のそれぞれの意味は次のとおりである。

-a オプション：鍵を使用するハッシュアルゴリズムの名前（HMAC-SHA1）

-b オプション：鍵の長さ（128 ビット）

-n オプション：鍵の種類（この場合は HOST）

最後のパラメータ：鍵の名前（ns1-ns2.example.com）

dnssec-keygen プログラムは、それぞれに鍵文字列を含む次のファイルを生成する。

```
Kns1-ns2.example.com.+157+34567.key
Kns1-ns2.example.com.+157+34567.private
```

プログラムが鍵のペア（公開鍵と秘密鍵）を生成すると、*key* という拡張子の付いたファイルには公開鍵文字列、*private* という拡張子の付いたファイルには秘密鍵が格納される。この例では秘密鍵だけを生成しているため、どちらのファイルの鍵文字列も TSIG の実装に対しては同じである。次に、これらのファイルのいずれかの鍵文字列が鍵ファイルと呼ばれるファイルにコピーされる。その後、このファイルは key 文のなかの include 文を使用して参照される。

8.2.2 通信を行うネームサーバでの鍵の定義

dnssec-keygen ユーティリティによって生成する鍵は、通信する 2 つのサーバ（通常はプライマリネームサーバとセカンダリネームサーバ）の named.conf 設定ファイル内で定義する必要がある。これは、BIND の key 文を使用して行う。

```
key “ns1-ns2.example.com.” {
      algorithm hmac-sha1;
      include “/var/named/keys/secretkey.conf”;
};
```

ここでファイル secretkey.conf には、(この例では)キーワード secret と実際の鍵文字列が格納される。

```
secret “MhZQKc4TwAPkURM==“
```

8.2.3 すべてのトランザクションでの鍵の使用をネームサーバに指示する

以下に、すべてのトランザクション(DNS クエリ/レスポンス、ゾーン転送、動的更新など)で鍵を使用するようにサーバに指示するコマンドを示す。

```
server 192.249.249.1 {
     keys { ns1-ns2.example.com.; };
```

同じ文を acl 文のエントリとしても使用できる。

```
acl key_acl {
     ns1-ns2.example.com.; };
```

8.2.4 鍵ファイル生成と鍵構成プロセスのためのチェックリスト

チェックリスト項目 7:TSIG 鍵の長さは 128 ビット以上あること。

チェックリスト項目 8:通信するホストのペアごとに一意の TSIG 鍵を生成すること(つまり、プライマリネームサーバとのトランザクションを認証するセカンダリネームサーバごとに別々の鍵を生成する、など)

チェックリスト項目 9:ネームサーバ上の鍵ファイルに鍵文字列をコピーした後、dnssec-keygen プログラムによって生成される 2 つのファイルには、サーバ管理者アカウント(たとえば UNIX の root)のみアクセス可能とすべきである。あるいは削除するとさらによい。これらのファイルの紙のコピーも破棄すること。

チェックリスト項目 10:鍵ファイルは、鍵を生成したネームサーバの通信相手のネームサーバへ、ネットワーク経由で安全に送信すること。

チェックリスト項目 11:設定ファイル(UNIX 上の BIND の場合、通常は/etc/named.conf)内で TSIG 鍵を記述する文(鍵名[ID]、署名アルゴリズム、鍵文字列)には、鍵文字列を直接含めないこと。設定ファイルに鍵文字列があると、ゾーン管理者以外でも設定ファイルを読めるようになっている必要のある環境では、鍵の侵害のリスクが増加する。代わりに、別の鍵ファイルに鍵文字列を定義し、設定ファイルの key 文の include ディレクティブを通じて参照すべきである。TSIG 鍵ごとに別々の鍵ファイルを用意すること。

チェックリスト項目 12:鍵ファイルは、ネームサーバソフトウェアを実行するアカウントが所有すること。鍵ファイルの読み取りや編集は、ネームサーバソフトウェアを実行しているアカウントだけが行えるように、許可ビットを設定すること。

チェックリスト項目 13:サーバのペア間のメッセージの署名に使用する TSIG 鍵は、やり取りをしている両方のサーバの server 文のなかで互いを指すように指定すること。これは、特定のトランザクションのリクエストメッセージとトランザクションメッセージの両方が署名され、セキュリティ保護されるのを確実にするために必要である。

あるゾーンにおいて秘密鍵を共有するサーバのペアのそれぞれの設定ファイルのなかで、（BIND 設定ファイルの server 文を通じて）サーバ間のすべてのやり取りに使用する鍵の名前を指定する必要がある。

8.2.5 TSIG を使用したゾーン転送のセキュリティ保護

ゾーン転送トランザクションに参加するサーバのペアは、key 文によって定義された鍵を使用するように指示されている必要がある（8.2.2 項）。このペアは一般に、プライマリネームサーバとセカンダリネームサーバから成る。名前付き鍵とゾーン転送リクエストメッセージを使用して MAC を送信するセカンダリネームサーバからのみ、プライマリネームサーバはゾーン転送リクエストを受け付けるように設定される。この設定は、zone 文内で副文 allow-transfer を使用して行う。以下に、ns1-ns2.example.com という鍵を使用するネームサーバからのみ example.com ゾーンのゾーン転送リクエストを許可するようにプライマリネームサーバに指定する allow-transfer 文の例を示す。

```
zone “example.com” {
      type master;
      file “zonedb.example.com”;
      allow-transfer { key {ns1-ns2.example.com.}; };
};
```

セカンダリネームサーバは、8.2.3 項に示した server 文を使用して、プライマリネームサーバ（IP アドレスは 192.249.249.1）へのゾーン転送リクエストにおいて鍵 ns1-ns2.example.com を使用するように指示されている。

8.2.6 TSIG または SIG(0)を使用した動的更新のセキュリティ保護

TSIG 鍵に基づく動的更新の制限は、BIND 8.2 以降のバージョンで、zone 文の allow-update 文によって指定できる。この文への引数には、キーワード key と、その後に TSIG 鍵を続けて指定する（BIND ネームサーバの設定ファイルにおける key 文の入力方法の詳細については、8.2.2 項を参照）。key 文が入力されていれば、zone 文に次の副文を追加して動的更新に秘密鍵を使用するように指定できる。

```
zone “example.com” {
      type master;
      file “zonedb.example.com”;
      allow-update { key dhcp-server.example.com.; };
};
```

文字列 dhcp-server.example.com.は FQDN に見えるが、実際には TSIG 鍵の名前を示している。この設定文の例の意味は、dhcp-server.example.com.という名前の鍵を所有するホストはすべて、プライマリ権威ネームサーバにある（ゾーン example.com の）ゾーンファイルに対する動的更新リクエスト（RR の追加、削除、変更）を送信できるということである。

動的更新メッセージの真正性確認に SIG(0)を使用するには、検証側のクライアントが鍵を取得できるように、使用する鍵の公開設定要素を先に DNS に格納しておく必要がある[10]。鍵の発行については 9.5 項を参照すること。必要であれば、先述の手順を実施してアクセスを制御する必要がある。それを行った後、（ネームサーバが SIG(0)と動的更新をサポートしていれば）更新側のネームサーバは鍵を取得し、動的更新リクエストを処理できる。

[10] 詳細については、次の動的更新の手引きのページの説明を参照：http://ops.ietf.org/dns/dynupd/secure-ddns-howto.html.

8.2.7　TSIG 鍵を使用した動的更新転送の制限の設定

動的更新は、プライマリ権威ネームサーバ上のゾーンファイルのコピーが唯一の「書き込み可能」なコピーであるため、このファイルに対してのみ許可される。これはかならずしも、プライマリ権威ネームサーバだけに動的更新リクエストの受け付けが許可されているという意味ではない。実際、BIND 9.1.0 以降ではセカンダリネームサーバが動的更新リクエストを受け付け、プライマリ権威ネームサーバに転送できるようになっている。このシナリオでは、セカンダリネームサーバが動的更新リクエストを転送する送信元について、送信元ホストの識別情報に基づいた制限がなければ、任意のホストからセカンダリネームサーバにリクエストを送信し、プライマリネームサーバに転送できてしまうため、プライマリネームサーバで指定されている動的更新制限が迂回されているのと同じである。この問題に対処するために、allow-update-forwarding という新しい副文が、動的更新の転送機能のある BIND で利用できるようになっている。TSIG 鍵を使用したこの allow-update-forwarding 文の例を以下に示す。

```
zone “example.com” {
     type slave;
     file “backupdb.example.com”;
          allow-update-forwarding { key dhcp-
server.example.com.; };
};
```

8.2.8　TSIG/SIG(0)鍵を使用した詳細な動的更新制限の設定

allow-update 文では、ゾーンレコードの内容ではなく、リクエストの送信元（IP アドレスや TSIG 鍵の保有によって識別される一連のホスト）に基づいて動的更新制限を指定する。ドメイン名／サブドメイン名と RR タイプ（A、MX、NS など）の組み合わせに基づいて動的更新アクセス制限（grant（許可）または deny（拒否））を指定するために、BIND 9 以降には zone 文内で使用する update-policy という副文が提供されている。update-policy 文は、TSIG 鍵に基づいてこれらの制限を適用する。つまり、update-policy 文では、そのドメイン／サブドメイン内のどのドメイン／サブドメインと RR タイプに対して、どの TSIG 鍵（または鍵保有者）が動的更新の実行を許可されるかを指定する。

update-policy 文の一般形式は、次のとおりである。

```
update-policy {
     (grant | deny) TSIGkey nametype name [type]
};
```

各要素の意味を以下に示す。

grant/deny—この後に続く組み合わせに対して動的更新を許可（grant）／拒否（deny）する。

TSIGkey—動的更新の真正性確認に使用する TSIG 鍵の名前。

nametype—指定可能な値とその意味は次のとおり。

name—この後の name フィールドに指定されたドメイン名に制限を適用する。

subdomain—この後の name フィールドに指定されたドメインのサブドメインに制限を適用する。

wildcard—この後の name フィールドにワイルドカード構文（すなわち*）を使用して指定された一連のドメインに制限を適用する。

self—TSIGkey フィールドに指定されている名前と同じ名前のドメインに制限を適用する（つまり、レコードの更新対象のドメイン名は、動的更新リクエストの認証に使用する鍵の名前と同じである）この使用法では、name フィールドの内容は冗長になるが、この文では使用する必要がある（つまり、name フィールドは空であってはならない）。

name—ドメインの名前を指定する場合に使用する。構文および対象となるドメインは、nametype フィールドの値に基づく（たとえば、nametype フィールドの値が subdomain ならば、指定されたドメイン名のすべてのサブドメインがこの文の対象となる）。

type—有効な任意の RRType（NSEC タイプを除く）、またはワイルドカードの「ANY」（ANY は NSEC タイプを除くすべての RR タイプを表す）を含むことができる任意指定のフィールド。指定がなければ、SOA、NS、RRSIG、NSEC を除くすべての RRType を表す。また、複数の RRType を空白で区切って指定することもできる（たとえば、A NS など）。

update-policy 文の例とその意味を以下に示す。

example.com 内に sales.example.com というドメインがあり、ネームサーバはそのドメイン名と同じ名前の TSIG 鍵（すなわち、sales.example.com）を使用するものとする。次のようにすると、sales.example.com からの動的更新はすべて、ゾーンファイル内のそのドメインのすべてのリソースレコードに限定される。

```
zone “example.com” {
     type master;
     file “zonedb.example.com”;
     update-policy { grant sales.example.com. self
sales.example.com.; };
};
```

次のようにすると、sales.example.com からの動的更新はすべて、そのドメインの A および MX RR タイプに限定される。

```
zone “example.com” {
     type master;
     file “zonedb.example.com”;
     update-policy {
       grant sales.example.com. self sales.example.com. A
MX; };
};
```

TSIG 鍵 sales.example.com を持つクライアントに、NEsales.example.com のサブドメインに関連する、ネームサーバレコード（RRType NS）以外のすべてのレコードの更新を許可するには、次のようにする。

```
zone “example.com” {
     type master;
     file “zonedb.example.com”;
     update-policy {
          deny sales.example.com. subdomain NE
sales.example.com. NS;
          grant sales.example.com. subdomain NE
sales.example.com. ANY; };
};
```

8.3 推奨事項のまとめ

以下に、このセクションで取り上げた主な推奨事項をまとめる。

+ **チェックリスト項目 6**：管理者は、各種の DNS トランザクションのそれぞれに対して、信頼できるホスト（またはブラックリストに載せるホスト）の名前付きリストを作成することが推奨される。一般に、次の種類の役割のホストは、該当する ACL に含めることを考慮すること。
  – 組織内の各ゾーンで定義される DMZ ホスト
  – ゾーン転送の開始が許可されているすべてのセカンダリネームサーバ
  – 再帰的クエリの実行が許可されている内部ホスト
+ **チェックリスト項目 7**：TSIG 鍵の長さは 128 ビット以上あること。
+ **チェックリスト項目 8**：通信するホストのペアごとに一意の TSIG 鍵を生成すること（つまり、プライマリネームサーバとのトランザクションを認証するセカンダリネームサーバごとに別々の鍵を生成する、など）
+ **チェックリスト項目 9**：ネームサーバ上の鍵ファイルに鍵文字列をコピーした後、dnssec-keygen プログラムによって生成される 2 つのファイルには、サーバ管理者アカウント（たとえば UNIX の root）のみアクセス可能とすべきである。あるいは削除するとさらによい。これらのファイルの紙のコピーも破棄すること。
+ **チェックリスト項目 10**：鍵ファイルは、鍵を生成したネームサーバの通信相手のネームサーバへ、ネットワーク経由で安全に送信すること。
+ **チェックリスト項目 11**：設定ファイル（UNIX 上の BIND の場合、通常は/etc/named.conf）内で TSIG 鍵を記述する文（鍵名[ID]、署名アルゴリズム、鍵文字列）には、鍵文字列を直接含めないこと。設定ファイルに鍵文字列があると、ゾーン管理者以外でも設定ファイルを読めるようになっている必要のある環境では、鍵の侵害のリスクが増加する。代わりに、別の鍵ファイルに鍵文字列を定義し、設定ファイルの key 文の include ディレクティブを通じて参照すべきである。TSIG 鍵ごとに別々の鍵ファイルを用意すること。
+ **チェックリスト項目 12**：鍵ファイルは、ネームサーバソフトウェアを実行するアカウントが所有すること。鍵ファイルの読み取りや編集は、ネームサーバソフトウェアを実行しているアカウントだけが行えるように、許可ビットを設定すること。
+ **チェックリスト項目 13**：サーバのペア間のメッセージの署名に使用する TSIG 鍵は、やり取りをしている両方のサーバの server 文のなかで互いを指すように指定すること。これは、特定のトランザクションのリクエストメッセージとトランザクションメッセージの両方が署名され、セキュリティ保護されるのを確実にするために必要である。

（本ページは意図的に白紙のままとする）

9. DNS クエリ／レスポンスのセキュリティ保護のためのガイドライン

6.1.4 項では、DNSSEC のデータ源の真正性確認、データ完全性の検証、および真正性の確認された不存在証明を用いることによる DNS クエリ／レスポンストランザクションの保護策について説明した。このセクションでは、DNSSEC 保護策に含まれるメカニズム、それらのメカニズムで行われる操作、およびそれらの操作について、チェックリストを使って安全に実行する方法について説明する。つまり、このセクションではネームサーバソフトウェアでサポートされる DNSSEC の操作を通して DNSSEC の機能を安全に導入するためのガイドラインを提供する。これらのガイドラインについては、9.3 項から 9.7 項で扱う。

DNS クエリ／レスポンストランザクションをエンドツーエンドで確実に保護するには、（DNSSEC 仕様で提供されるものとは別に）ローカルの DNSSEC 対応キャッシング／リゾルビングネームサーバとスタブリゾルバ間の通信経路のセキュリティ保護などの追加的な保護策が必要である。これらの保護策については 9.8 項で説明する。

ゾーン管理者は、DNSSEC 仕様によってゾーンファイルに導入された追加の RRset（特に、NSEC RRset）が存在する状況での動的更新のロジックを理解している必要がある。このロジックについては、9.9 項で説明する。

9.1 BIND での DNSSEC 処理の有効化

DNSSEC の署名付きゾーンを導入する前に、DNSSEC 処理を行うようにネームサーバを設定する必要がある。BIND でこの設定を行うには、named の設定ファイル（named.conf）の options 文に次の行を追加する。

```
options {
      dnssec-enable yes;
      };
```

再起動後、ネームサーバは DNS クエリ／レスポンストランザクションに対して DNSSEC 処理を実行するようになる。

チェックリスト項目 14：DNSSEC の署名付きゾーンまたはクエリの署名付きゾーンを導入するネームサーバは、DNSSEC 処理を行うように設定すること。

9.2 DNSSEC のメカニズムと操作

DNSSEC メカニズムには、2 つの主要なプロセスが関係している。1 つは署名の生成と提供であり、もう 1 つは署名の検証である。以下では、これらのプロセスについて説明する。

9.2.1 署名の生成と提供

このプロセスの最初の仕事は、ゾーンファイル内の各 RR に関連付けられたディジタル署名を生成することである。DNSSEC では、すべての RR に対して署名を生成する代わりに、RRset（同じ所有者名、クラス、および RRType を持つ RR のセット）に対する署名の生成が規定されている。このディジタル署名とその関連情報（使用される鍵の ID、署名の有効期限の開始日と終了日など）は、Resource Record Signature（RRSIG）という特別な RRType の RR にカプセル化される。実際の鍵文

字列と、（RRSIG 内の）署名を検証するために使われる公開鍵に関する関連情報は、DNSKEY という特別な RRType で提供される。もう 1 つの RRType である Next Secure（NSEC）は、特定のドメイン（所有者名）で使用できる RRType を（正規順序で[11]）列挙するために使用され、そのドメインに存在しない RRType を問い合わせるクエリに対して、真正性の確認された不存在証明を提供するために、この RRType の署名（RRSIG RR）が生成される。また、ゾーンがその子ゾーンの公開鍵の真正性を保証する必要がある場合のために、任意使用の RRType である Delegation Signer（DS）が用意されている。つまり、DS RR は子ゾーンの（ハッシュ）公開鍵を含んでいる RR に対する署名を保持している。DNSSEC 仕様によって導入されたこれら追加の RRType の詳しい構文は、RFC 4034 に規定されている。このうち最も重要なのは、実際の署名文字列を持つ RRSIG RR である。

RRSIG RR は、ほかの RR と同じように、所有者名、TTL、クラス、RRType、および RDATA の各フィールドを持つ。ディジタル署名とそのすべての関連情報は、RDATA フィールドに含まれている。RRSIG RR の RDATA フィールドのレイアウト（すべてのサブフィールドを含む）を図 9-1 に示す。その後に、各サブフィールドを簡単に説明する。

<table>
<tr><td>RRType Covered
（対象となる RRType）</td><td>Algorithm Code
（アルゴリズムコード）</td><td>Labels
（ラベル）</td></tr>
<tr><td colspan="3">Original TTL（元の TTL）</td></tr>
<tr><td colspan="3">Signature Expiration（署名の有効期限）</td></tr>
<tr><td colspan="3">Signature Inception（署名の開始日時）</td></tr>
<tr><td>Key Tag（鍵タグ）</td><td colspan="2">Signer's Name（署名者名）</td></tr>
<tr><td colspan="3">エンコードされた署名</td></tr>
</table>

図 9-1. RRSIG RR の RDATA フィールドのレイアウト

「RRType Covered（対象となる RRType）」フィールドは、この RRSIG に含まれる署名の対象となる RRset のタイプである。「Algorithm Code（アルゴリズムコード）」フィールドは、署名の生成に使われている特定の暗号化アルゴリズムを表すために割り当てられた整数コードである。「Labels（ラベル）」フィールドと「Original TTL（元の TTL）」フィールドは、ラベルの数（FQDN 内のラベルの数）と、この署名の対象となっている RRset の TTL 値である。「Signature Expiration（署名の有効期限）」と「Signature Inception（署名の開始日時）」の値は、署名の有効期間（この RRSIG が対象ゾーンに対して有効であるとみなされる期間）を表す絶対時間値である。「Key Tag（鍵タグ）」フィールドと「Signer's Name（署名者名）」フィールドは、クライアントが署名の有効性を確認するのに必要な DNSKEY RR のハッシュと FQDN である。最後のフィールドは、エンコードされた署名そのものである。

通常の RR とともにこれら追加の RR を持つゾーンは、*署名付きゾーン*と呼ばれる。これらの署名付きゾーンをホストし、要求された RR とともに適切な署名（すなわち対応する RRSIG）がそのレスポンスに含まれるネームサーバは、*DNSSEC 対応ネームサーバ*と呼ばれる。

[11] DNS データの正規形式については、http://www.ietf.org/rfc/rfc4034.txt）から入手可能な RFC 4034、『*Resource Records for the DNS Security Extensions*』のセクション 6 を参照すること。

9.2.2 署名の検証

署名付きゾーンから発信されるレスポンスは、*署名付きレスポンス*と呼ばれる。署名付きレスポンスに含まれる署名を検証する機能を持つリゾルバは、*DNSSEC 対応検証機能付きリゾルバ*と呼ばれる。リゾルバが（レスポンスとともに送信された）ゾーンの公開鍵を使ってゾーンの（レスポンス内の）RRset に関連付けられた署名を検証するには、その公開鍵に対する信頼を事前に確立しなければならない。DNSSEC では、*信頼の連鎖の構築*と呼ばれるサブプロセスをリゾルバに行わせることによって、この要件に対応する。このサブプロセスでは、リゾルバが信頼できる公開鍵（*トラストアンカ*と呼ばれる）の既知のリストから処理を開始し、DNS の名前空間階層を使って構築した公開鍵の連鎖をたどることにより、特定のレスポンスに含まれるゾーンの公開鍵に対する信頼を確立する。リゾルバ内のトラストアンカは、理想的にはルート公開鍵（ルートサーバが DNSSEC 対応である場合）または階層の下位にあるゾーンの公開鍵で構成される。リゾルバ内のトラストアンカリストの構築は、DNS トランザクションを通してではなく、帯域外のメカニズムを使って行われる。

上記の DNSSEC プロセスには、いくつかのネームサーバの操作とリゾルバの操作が関係している。ネームサーバの操作は次のとおりである。

+ DNSSEC-OP1：公開鍵と秘密鍵のペアの生成
+ DNSSEC-OP2：秘密鍵のセキュアな格納
+ DNSSEC-OP3：公開鍵の配布
+ DNSSEC-OP4：ゾーン署名
+ DNSSEC-OP5：鍵のロールオーバ（鍵の変更）
+ DNSSEC-OP6：ゾーン再署名

リゾルバの操作は次のとおりである。

+ DNSSEC-OP7：トラストアンカの構成
+ DNSSEC-OP8：信頼の連鎖の確立と署名の検証

ネームサーバの DNSSEC-OP1 から DNSSEC-OP4 までの操作とリゾルバの操作（DNSSEC-OP7 と DNSSEC-OP8）は、DNSSEC の導入前に行われるか、またはセキュアな操作（署名付きレスポンスの提供と署名の検証）のために行われるため、このセクションで取り上げる。ネームサーバの残りの操作である鍵のロールオーバ（DNSSEC-OP5）とゾーンの再署名（DNSSEC-OP6）は、全面的な運用の DNSSEC の導入後に定期的に行われるため、セクション 11 で取り上げる。

9.3 公開鍵と秘密鍵のペアの生成（DNSSEC-OP1）

DNSSEC では、非対称鍵を使ったディジタル署名の生成と検証が規定されている。このため、公開鍵と秘密鍵のペアを生成する必要がある。DNSSEC 仕様では異なるタイプの鍵は要求されていない（1 つの鍵ペアしか要求されていない）が、鍵のロールオーバ（鍵の変更）やゾーンの再署名などの日常的なセキュリティ管理業務を簡単にするためには、試験的な導入の経験から、少なくとも 2 つのタイプの鍵が必要であることがわかっている。一方の鍵タイプは、*鍵署名鍵（KSK：key signing key）*と呼ばれる。この鍵（具体的には、鍵ペアのうち秘密のほう。KSK 秘密鍵と呼ばれる）は、ゾーンファイル内の鍵セット（すなわち DNSKEY RRSet）の署名にのみ使用される。もう一方の鍵タイプは、*ゾーン署名鍵（ZSK：zone signing key）*と呼ばれ（その秘密のほうの鍵は、ZSK 秘密鍵と呼ばれ）、ゾーン内の（DNSKEY RRSet を含む）すべての Rrset に署名するために使用される。KSK 鍵と ZSK

鍵は、これらの公開されるほうの鍵（この場合は KSK 公開鍵と呼ばれる）を表す DNSKEY RR のセキュアエントリポイント（SEP：Secure Entry Point）フラグビットをセットすることにより、管理上区分される。

2 つのタイプの鍵ペアを作成することの背景にあるロジックは、鍵のタイプごとに別々の機能セットを提供することにより、鍵のロールオーバやゾーンの再署名に関わる作業全体の複雑さを低減することである。したがって、KSK（KSK 秘密鍵）は鍵セット（すなわち DNSKEY RRSet）に署名を加えるために使用され、認証された委任で使用するために親に送信される鍵タイプ（公開の構成要素である KSK 公開鍵）である。認証された委任は、子の KSK 公開鍵のハッシュを使って DS RR を生成し、親自身の ZSK を使って対応する署名（RRSIG RR）を生成することによって行われる。KSK（KSK 公開鍵）は、検証機能付きリゾルバで署名検証用の信頼の連鎖を確立するためのトラストアンカ（SEP 鍵）として使用される場合もある。

ZSK（ZSK 秘密鍵）は、ゾーンファイル全体（すべての RRset）に署名を加えるために使用される。この鍵の公開部分（ZSK 公開鍵）は、親に送信されず、常にゾーンに残される。

KSK と ZSK の鍵ペアの生成に関わる決定パラメータは、次のとおりである。

+ ディジタル署名アルゴリズムの選択
+ 鍵サイズの選択
+ 暗号周期（鍵を使用する期間）の選択

ディジタル署名アルゴリズムの選択は、既知の標準に含まれる推奨アルゴリズムに基づいて行われる。NIST のディジタル署名標準（DSS：Digital Signature Standard）（FIPS 186-2）では、以下の 3 つのアルゴリズムが選択肢として用意されている。

+ DSA（Digital Signature Algorithm：ディジタル署名アルゴリズム）
+ RSA
+ ECDSA（Elliptic Curve DSA：楕円曲線 DSA）

この 3 つのアルゴリズムのうち、RSA と DSA は広く普及しているため、DNSSEC 用のアルゴリズムの候補とみなされる。パフォーマンスの点では、RSA も DSA も署名生成の速度は同等であるが、DSA は署名の検証速度がひじょうに遅い。このため、このガイドラインに関する限り、RSA が推奨されるアルゴリズムである。SHA-1 を使った RSA は、DNSSEC とともに実装することが義務付けられている唯一の暗号化アルゴリズムである。ネームサーバとクライアントは、少なくとも RSA アルゴリズムを使用できることが期待される。1 つのゾーンで少なくとも 1 つの ZSK が RSA アルゴリズムを使用することが推奨される。

NIST の Secure Hash Standard（SHS）（FIPS 180-2）では、NIST の DSS のディジタル署名アルゴリズムを使ってディジタル署名を生成するためのアルゴリズムスイートの一部として使用することが承認されているハッシュアルゴリズムとして、SHA-1、SHA-224、SHA-256、SHA-384、および SHA-512 が規定されている[12]。実装の簡単さや入手のしやすさを考慮すると、SHA-1 が適切な候補として推奨される。ただし、最近、ハッシュ衝突攻撃により、理論上の強度が 11 ビット分減ったと報告されている。

---

12 FIPS 180-2、『*Secure Hash Standard (SHS)*』は http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf から入手できる。

鍵サイズの選択は、鍵が侵害されるリスクとパフォーマンスのトレードオフになる。パフォーマンスの変動要素は、署名の生成と検証の回数である。DNSKEY RR は DNS レスポンスの追加セクションで送信される可能性があるため、DNS レスポンスパケットのサイズも 1 つの要素である。KSK は鍵セット（DNSKEY RRSet）の署名にのみ使用されるため、パフォーマンスはそれほど大きな問題ではない。しかし、KSK はゾーンへのエントリポイントとなる鍵であるため、KSK の侵害は大きな影響を及ぼす可能性がある。侵害発生時に KSK をロールオーバするときには、多数の検証機能付きリゾルバのトラストアンカを更新しなければならない可能性がある。このため、KSK については鍵サイズを大きくすることが推奨される。鍵サイズを大きくすると、鍵が侵害される可能性が減少し、管理者による監視や事後措置が求められるロールオーバを頻繁に行う必要がなくなる。

ZSK はゾーン内のすべての RRset の署名に使用されるため、ZSK の鍵サイズの選択に関する限り、パフォーマンスは確実に影響要素の 1 つになる。しかし、ZSK はそのゾーンの RRset に署名を加えるためにのみ使用され、真正性の確認された委任を子ゾーンに提供するためには使用されないため、影響の範囲は 1 つのゾーンに限られる。したがって、ZSK には KSK より小さい鍵サイズを使用できる。

暗号周期（ロールオーバ周期）の選択は、鍵の露出のリスクによって決まる。KSK の場合は、生成される署名セットの量が多くない（KSK は DNSKEY RRSet のみに署名を加え、その RRset を変更する回数も少ないため）。鍵の露出が最低限であり（KSK 秘密鍵の推定に利用できるデータの量が少なく）、鍵のサイズも大きいことから、KSK の暗号周期は長く（通常は 1～2 年に）できる可能性がある。

ZSK の場合は、鍵の露出が多いため、鍵を推定されるリスクが高い。鍵の露出の多さは、生成される署名セットの数が非常に多いことの結果である（ZSK はゾーン内のすべての RRset に署名する。さらに、DNSKEY RRset に比べてほかの RRset は変更の頻度がはるかに高いため、新規に生成される署名の数が多い）。この要素と、鍵のサイズが比較的小さいことから、ZSK は KSK より頻繁に（通常は 1～2 か月で）ロールオーバしなければならない可能性がある。

チェックリスト項目 15：KSK の鍵が侵害されると DNS への影響が大きいため、KSK の鍵サイズは十分に大きくすること。

各タイプ（KSK と ZSK）の生成すべき鍵の数に関しては、署名に使用する ZSK のほかに余分の ZSK を生成することが推奨される。したがって、ゾーン管理者は DNSSEC の最初の導入時に鍵生成プログラムを使って 1 つの KSK と 2 つの ZSK を生成すべきである。一方の ZSK を現在有効な鍵として扱い、その秘密部（ZSK 秘密鍵）を署名の生成に使用する。もう一方の ZSK（ZSK 公開鍵）は、DNSKEY RRSet の一部とするものの、その秘密部（ZSK 秘密鍵）は RRset の署名には使用しない。この追加の ZSK は、非常事態（鍵の侵害など）発生時の即時ロールオーバを行うためにいつでも利用できる ZSK となり、また、この鍵が現在の暗号周期終了後にゾーンがロールオーバする鍵であることを検証機能付きリゾルバに伝えるある種の事前通知となる。DNSKEY RRSet にこの鍵が存在するだけで、検証機能付きリゾルバは、ロールオーバが発生しても鍵をすぐに署名検証に使用できるように、新しい鍵をキャッシュし、信頼を確立しておける。

KSK 鍵と ZSK 鍵に関して推奨されるディジタル署名アルゴリズムスイート、鍵サイズ、および暗号周期を表 9-1 に示す。

表 9-1. ディジタル署名アルゴリズム、鍵サイズ、および暗号周期

鍵タイプ	ディジタル署名アルゴリズムスイート	鍵サイズ	暗号周期（ロールオーバ周期）
鍵署名鍵（KSK）	RSA-SHA1	2048 ビット	12 か月（1 年）
ゾーン署名鍵（ZSK）	RSA-SHA1	1024 ビット	1 か月（30 日）

9.3.1 鍵ペアの生成（説明用の例）

DNSSEC 対応ネームサーバの実装はすべて、非対称鍵ペア（公開鍵と秘密鍵のペア）を生成するためのユーティリティプログラムを備えているはずである。このようなプログラムの使用例として、dnssec-keygen（BIND 9.3.x に付属）の例を以下に示す。

```
dnssec-keygen –a algorithm – b bits –n type [options] name
```

（-a パラメータの） *algorithm* には、次のいずれかのアルゴリズムを指定する。

+ RSASHA1
+ DSA

-b パラメータの *bits*（鍵サイズ）には、次の範囲の値を指定する。

+ RSASHA1 の場合は 512～4096
+ DSA の場合は 512～1024（64 の倍数にすること）

-n パラメータの *type* には、ZONE または HOST を指定する。

name には、鍵の所有者（通常はゾーン頂点のドメイン名）を指定する。

このコマンドによって、2 つのファイルが生成される。1 つは公開鍵を含むファイルであり、もう 1 つは対応する秘密鍵を含むファイルである。これらのファイルの一般名は次のとおりである。

```
K<domain_name>+algorithm_id+Key_id.key
K<domain_name>+algorithm_id+Key_id.private
```

domain_name は、コマンドラインで指定された *name* パラメータの値である。algorithm_id は、次のいずれかである。

3 – DSA

5 – RSASHA1

key_id は、プログラムによって生成される一意の鍵識別子である。

たとえば、RSASHA1 アルゴリズムを使って example.com というゾーンに署名を加える 1024 ビット長の ZSK 鍵セットを生成する場合は、次のコマンドを発行する。

```
dnssec-keygen –a RSASHA1 –b 1024 –n ZONE example.com
```

このコマンドにより、以下のファイルが生成され、それぞれに秘密鍵と公開鍵が格納される。

```
Kexample.com.+005+28345.private
Kexample.com.+005+28345.key
```

これらのファイル名のうち、005 が algorithm_id を示し、28345 が一意の鍵 ID を示す。

*.key ファイルでは、公開鍵の情報がゾーンファイルの RR と同じ構文で表現されている。Kexample.com.+005+28345.key ファイルの内容は次のようになる。

```
example.com  IN  DNSKEY  256  3  5  BQFG+KGJ7……….（Base64 でエンコードされた鍵文字列）
```

したがって、次のコマンドを使って、公開鍵を含むファイルの内容をゾーンファイル（zonedb.example.com）に簡単に追加することができる。

```
cat *.key >> /var/named/zonedb.example.com
```

公開鍵を含む DNSKEY RR をゾーンファイルに追加したあとは、ゾーンを実際に署名する前に SOA RR のゾーンのシリアル番号をインクリメントする必要がある。

9.4 秘密鍵のセキュアな格納（DNSSEC-OP2）

KSK と ZSK の鍵ペアに含まれる秘密鍵は、無許可のアクセスから保護すべきである。可能であれば、この秘密鍵をゾーンファイルのマスタコピーとともに、ネットワークからアクセスできない物理的にセキュアなコンピュータに（DNSSEC 対応ネームサーバからみて）オフライン状態で保存すべきである。秘密鍵を使って生成した署名は、（永続的なネットワークリンクではなく）動的に確立されたネットワーク接続を使って、ロードプロセスを介してプライマリ権威ネームサーバに転送すべきである。

この戦略は、DNSSEC 対応ネームサーバが動的更新をサポートしなければならない状況には適さない。動的更新トランザクションをサポートするには、DNSSEC 対応ネームサーバ（通常はプライマリ権威ネームサーバ）でゾーンファイルのマスタコピーとゾーン署名鍵に対応する秘密鍵（ZSK 秘密鍵）の両方をオンライン状態にして、更新された RRset の署名を即座に更新できるようにしなければならない。鍵署名鍵に対応する秘密鍵（KSK 秘密鍵）は、オフラインのままにしておくことができる。このシナリオでは、ZSK 秘密鍵を保護するために以下の措置を取る必要がある。

+ 許可されたゾーン管理者／オペレータ以外は、鍵生成プログラム／ユーティリティを起動するシェルにアクセスできないようにする。

+ 許可されたゾーン管理者／オペレータ以外は、秘密鍵ファイルが格納されているディレクトリ（通常は、ゾーンと同じ名前を持つディレクトリの下にあるサブディレクトリで、そのディレクトリに含まれるファイルの名前が BIND では K<zonename>.+AlgorithmID+<keytag>.private という構造になっているもの）にアクセスしたり、ディレクトリを表示したりできないようにする。

+ ミラーディスクの導入、またはテープ、CD／DVD、光媒体への定期的なバックアップによって、秘密鍵ファイルを格納するディレクトリの内容に対して十分なフォールトトレランスを提供する。バックアップ媒体も開示、改ざん、盗難から保護する。

+ 暗号化されたファイルシステムに秘密鍵を格納するという方法もある。

チェックリスト項目 16：DNSSEC 対応プライマリ権威ネームサーバが動的更新をサポートしない場合は、ZSK と KSK の両方に対応する秘密鍵をネームサーバに保管しないこと。動的更新がサポートされ

ている場合は、ZSK に対応する秘密鍵だけをネームサーバに保管し、適切なディレクトリ／ファイルレベルのアクセス制御リストまたは暗号化に基づく保護策を取ること。

9.5 公開鍵の発行（DNSSEC-OP3）とトラストアンカの設定（DNSSEC-OP7）

DNSSEC 対応リゾルバによる特定のゾーンのゾーンデータを検証する操作は、そのリゾルバがそのゾーン（ゾーンデータを検証するゾーン）の公開鍵またはその先祖（親および DNS ツリーにおいてその親の上位にあるゾーン）を認識して信頼することから始まる。検証するゾーン（example.com など）がセキュアであり、その親（.com）がセキュアでない場合、信頼点はそのゾーン自体から始まる。親（.com ゾーン）がセキュア（DNSSEC 対応）であり、その親の親（ルートゾーン）がセキュアでない場合、最初の信頼点はその親（.com ゾーン）である。ルートゾーンがセキュアである場合は、当然ながらルートゾーンが信頼の起点になる。

信頼の連鎖の開始点がどこ（エンタープライズレベル、TLD、ルートゾーン）であっても、その開始点の公開鍵は、その公開鍵に対応するネームサーバによって通知対象となる DNSSEC 対応リゾルバに通知されるべきである。DNSSEC 対応リゾルバに通知されるこれらの公開鍵は、*トラストアンカ*と呼ばれる。DNS には（X.509 公開鍵基盤のような）公開鍵の第三者認証を行う機能がないため、（DNS に対して）帯域外のプロセスによって公開鍵を配布しなければならない。この配布は、Web サイトや電子メールなどのチャネルを利用して行うことができる。

DNSSEC 対応リゾルバのトラストアンカリストに含まれるエントリは、ゾーンの署名付きレスポンスがセキュアと非セキュアのどちらに分類されるかを決定する。9.2.2 項で述べたように、リゾルバは署名の検証を行う前に、レスポンスを送信したゾーンの公開鍵に対する信頼を確立しなければならない。トラストアンカリストのどのエントリを使っても信頼の連鎖を構築できないためにその信頼を確立できない場合は、そのレスポンスは非セキュアであるとみなされる。信頼の連鎖を構築できないのは、DNS の名前空間に、署名付きゾーンが切れ目なく連続する階層ではなく、署名付きゾーンの孤島が含まれていることが原因である。署名付きゾーンの孤島のマッピング例（図 9-2）に関して、各トラストアンカがレスポンスのラベル付けに与える影響を表 9-2 に示す。

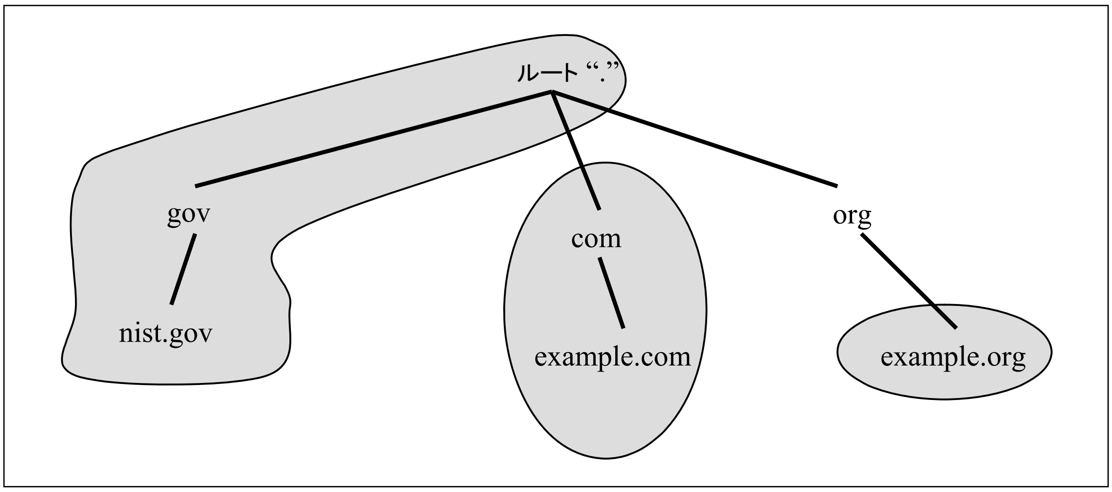

図 9-2. セキュリティの孤島のマッピング例

表 9-2. トラストアンカがレスポンスのラベル付けに与える影響

DNSSEC 対応リゾルバに導入されたトラストアンカ	以下のノードを対象とするクエリに対する DNSSEC レスポンスステータス		
	www.nist.gov	**www.example.com**	**www.example.org**
なし	非セキュア	非セキュア	非セキュア
ルート	セキュア	非セキュア	非セキュア
.com	非セキュア	セキュア	非セキュア
example.org	非セキュア	非セキュア	セキュア

9.6 ゾーンの署名（DNSSEC-OP4）

ゾーンファイルに署名すると、次の一連の操作が行われる。

+ ゾーンファイルがドメイン名の正規順序で並べ替えられる。
+ ゾーン内のすべての所有者名に対して NSEC RR が生成される。
+ KSK（KSK 秘密鍵）を使って DNSKEY RRSet への署名が行われる。この KSK は、DNSKEY RRType の RDATA 部分に含まれる SEP フラグを使って識別される。
+ ZSK（ZSK 秘密鍵）を使ってゾーン内のすべての RRset（DNSKEY RRset と NSEC RRset を含む）への署名が行われる。

チェックリスト項目 17：KSK を使った署名の生成は、オフラインに保存されている KSK 秘密鍵を使ってオフラインで行うこと。その後、DNSKEY RRSet を、その RRSIG RR とともにプライマリ権威ネームサーバに読み込むことができる。

ゾーン署名プログラム（BIND 9.3.x に付属の dnssec-signzone）を使ってゾーンに署名する例を次に示す。

```
dnssec-signzone -o <ゾーンの名前> -k <KSK を含むファイルの名前> <ゾーンファイルの場所> <ZSK を含むファイルの名前>
```

Kexample.com.+005+76425.key ファイルに格納されている KSK と、以前に生成された ZSK を使ってゾーン example.com のデータに署名をするには、次のコマンドを実行する。

```
dnssec-signzone -o example.com -k Kexample.com.+005+76425.key
\ /var/named/zonedb.example.com  Kexample.com.+005+28345.key
```

ゾーンファイルに署名を加えるプロセスは、各 RRset（同じ所有者名、クラス、RRType を持つ RR）のハッシュを生成すること、（ZSK 秘密鍵を使って）RRset の署名を生成すること、および RRSIG RR タイプの新しい RR にこの署名情報を捕捉することで構成される。KSK は DNSKEY RRSet にのみ署名を加えるのに対し、ZSK はゾーンファイル内のすべての RRset に署名をする。ゾーンデータの署名に使われる秘密鍵を持つエンティティは、*署名者*または*署名機関*と呼ばれる。ほとんどの場合、署名者はゾーンに対応するドメインである。DNSSEC 対応権威ネームサーバは、DNS クエリに対して、該当するゾーンデータとそのデータのディジタル署名を返す。受信者は、（署名者の公開鍵に対する信頼を確立したあとで）署名者の公開鍵を使って、受信したゾーンデータに関連付けられているディジタル署名を復号する。この復号により、保存されている元のゾーンデータのハッシュが得られ

る。受信者は、（署名者が使用したのと同じハッシュ関数アルゴリズムを使って）受信したゾーンデータのハッシュも生成する。このように算出したハッシュとディジタル署名から取得したハッシュを比較し、ハッシュが一致すれば、レスポンスのディジタル署名が検証されたことになる。

9.7 信頼の連鎖の確立と署名の検証（DNSSEC-OP8）

すべてのゾーンが署名付きゾーンになるまでは、ゾーンが署名されてもその親が署名されていないという状況が起こり得る。その場合、DNSSEC 対応リゾルバにとって唯一の信頼（保証）点は、ゾーンデータ署名者の事前構成されている公開鍵である。署名者の公開鍵の真正性を保証するソースがほかに存在しない場合、リゾルバが信頼できる唯一のソースは、リゾルバにその公開鍵が信頼されるように構成されているゾーン署名者自体である。その一方で、ゾーン署名者である X の子の公開鍵の真正性を保証する DNS ゾーン X が存在すれば、リゾルバは X を介してゾーン署名者である X の子に対する信頼を確立できる。署名されたゾーンデータを提供する署名者の公開鍵に対する信頼を確立するためにリゾルバが使用する DNS ツリー内のゾーンの連なりは、*信頼の連鎖*と呼ばれる。

信頼の連鎖は、たとえばこの例においては、DNS サイト X から始まり（起点とする）、サイト Y のゾーンデータで終わる。通常、DNS サイト X は、グローバル DNS ゾーンツリーにおいてゾーン Y の直接の親となるゾーン（*親ゾーン*と呼ばれる）である。親ゾーンは、子ゾーンの公開鍵のハッシュにディジタル署名を行うことによって、子ゾーンの真正性を保証する。このハッシュは、*Delegation Signer (DS) RR* と呼ばれる DNS RR に格納される。また、署名付きでないゾーンは子ゾーンの公開鍵のハッシュに署名するための（秘密鍵／公開鍵ペアの）秘密鍵を所有していないため、親ゾーンは署名付きゾーンでなければならない。信頼の連鎖が存在するかどうかによって、署名付きゾーンは次のいずれかになる。

+ **セキュリティの孤島**：自己署名されているゾーン。自己署名されている理由は、ゾーンの親が、署名付きゾーンではないか、またはセキュアな委任を設定することを望まないため、子ゾーンの公開鍵の真正性を保証できないことにある。この場合、信頼の連鎖は実際には存在しない。

+ **連鎖したセキュアゾーン**：ゾーンの親と（場合によっては）DNS ゾーンツリーの上位にある 1 つ以上の先祖がすべて署名されているゾーン。この署名付きゾーンの階層および関連する構成済みの鍵の階層では、通常リゾルバはこの署名付きゾーンの階層の最上位にあるゾーンを信頼の連鎖の起点とみなし、そのゾーンの鍵を選択する。信頼の連鎖の起点となるゾーンの公開鍵は、*トラストアンカ*と呼ばれる。実際には、複数の鍵がトラストアンカになる可能性がある。つまり、トラストアンカとは、リゾルバが最初に信頼し、リゾルバが検証しようとしている署名に対応するゾーンの公開鍵までつながる信頼の連鎖を構築するために使用できる鍵のグループを指す。信頼の連鎖が DNS ツリーをさかのぼってルートゾーンまでつながっている場合、当該ゾーンは「グローバルにセキュア」であるとされる。

セキュリティの孤島ゾーンに対するディジタル署名サービスによるゾーンデータの保護策は、次の基本的な作業で構成される。

+ 公開鍵／秘密鍵ペアの生成

+ 秘密鍵のセキュアな（必要であれば、ネームサーバから見てオフラインに）格納

+ ゾーンファイルへの DNSKEY RR の取り込みによる公開鍵の発行

+ ゾーンデータに対するディジタル署名の生成（ゾーン署名）

連鎖したセキュアゾーンでは、追加のタスクが行われる。ゾーンが信頼の連鎖に組み込まれるためには、その親に自身の公開鍵（KSK 公開鍵）を安全に（DNS チャネル以外の手段を使って）通知しなければならない。親は、子の公開鍵のハッシュを作成し、それを親ゾーンの DS RR と呼ばれる新しい RR に保存する。また、RRSIG RR を生成することにより、この DS RR に署名する。しかし、実際の状況では、生成される署名データの量に助けられて、どのような鍵でも十分な計算能力があれば割り出すことができるため、鍵を定期的に変更しなければならないことがわかっている。連鎖したセキュアゾーンでは、ゾーンの KSK が変更されたときに、必ずその親に新しい鍵が通知されなければならない。親は、新しい DS RR を生成し、それに再び署名しなければならない。この管理負担を軽減するための一般的な方策は、もう 1 つの鍵ペアである KSK を使用することである。KSK は、DNSKEY RRSet の署名にのみ使用される。ゾーンファイル内のほかの権威を持つ RRset は、すべて ZSK を使って署名される。KSK は、親に対して発行される鍵である。親は、親自身の ZSK を使って DS RR と RRSIG RR を生成する。KSK は、ZSK（1024 ビットなど）に比べて十分な長さ（2048 ビットなど）で作成されるため、それほど頻繁に変更する必要はない。ただし、鍵のロールオーバ（鍵の変更）の管理可能な頻度や、ゾーンによって提供される DNS 情報の重要度のため、管理者は ZSK と KSK に対して 2 つの異なる鍵ペアを使用しない場合がある。

要約すると、連鎖したセキュアゾーンで行われる追加の作業は次のとおりである。

+ セキュアゾーンからその親への KSK のセキュアな伝送。この伝送は、帯域外で行われ、DNS トランザクションを必要としない可能性がある。

+ 親は、子の KSK を特別な鍵セットファイルディレクトリに保存し、鍵のハッシュを生成し、そのハッシュを DS RR に保存する。親は、この DS RR に対するディジタル署名（RRSIG RR）も生成し、それをほかの委任情報に含める。

署名付きレスポンスを検証する必要があるゾーン（ターゲットゾーン）は、信頼の連鎖の葉ノードである。この操作の前提条件は、ゾーンの ZSK に対する信頼を確立することである。ゾーンの ZSK に対する信頼は、次の操作によって確立される。

+ 信頼が確認された、親からの参照

+ 子ゾーンの KSK の真正性確認

親からの真正性確認済み参照を理解するには、通常の DNS 参照プロセスを見てみる必要がある。通常の DNS クエリでは、子ゾーンのドメイン名を問い合わせるクエリに関する権威を持つ情報が存在しないゾーンは、NS RRset と関連する追加情報（NS RRset で提供されるサーバの IP アドレスを含む RR）を提供することにより、ヒントまたは参照を返す。通常の DNS クエリ処理では、この参照をたどることが妥当な処理手順である。しかし、このプロセスは信頼の連鎖の確立という点で十分でない。NS RRset、および関連するグルーRR に関する情報（またはそのほかの追加情報）は、権威を持つ情報源（すなわち子ゾーン）の公開鍵で署名されていないため、信頼できるものとみなすことができない。このような参照情報の真正性を確認するために、親は DS RR を通じて暗号ヒントを提供する。

DNSSEC 対応リゾルバが署名付きゾーンである example.com から受信した DNS レスポンスの有効性をどのように確認するかを示す例を検討する。リゾルバは、そのトラストアンカから始まる信頼の連鎖に従って、.com ゾーンの公開鍵を認証し、信頼している。したがって、リゾルバは.com ゾーンが提供する NS RR と DS RR を信頼している。NS RR とそれに対応するグルーRR から、リゾルバは example.com の権威ネームサーバが ns.example.com であると判断し、その IP アドレスも認識している。リゾルバは、これらの情報を使って ns.example.com にアクセスし、その DNSKEY RRSet から example.com の KSK を取得する。リゾルバはこの鍵のハッシュを生成し、それを親ゾーンである.com の DS RR に含まれるハッシュと比較する。これら 2 つのハッシュが一致すると、.com ゾーン

から example.com ゾーンへの参照の真正性が確認されたこととなり、example.com ゾーンの KSK の真正性も確認される。ZSK は example.com の KSK によって署名されているため、リゾルバは example.com の ZSK の真正性を確認できる方法で（ZSK を）取得できる。ZSK は、example.com のすべての RRset に署名しているため、そのゾーンから受信した署名付きレスポンスの有効性は、信頼できるようになった ZSK を使って確認できる。

このプロセスの説明からもわかるように、連鎖したセキュアゾーンの場合、親から受け取った委任情報は次の要素で構成される。

+ **子ゾーンの NS RR。**これらの NS レコードに関して権威を持つ情報源は子ゾーンであるため、提供される NS RR はヒントまたは参照と呼ばれる。
+ **グルーRR。**NS RR で指定されたサーバの場所（参照されたネームサーバの IP アドレス）を提供する。
+ **DS RR。**子ゾーンの KSK または ZSK のハッシュを提供する。
+ **RRSIG RR。**DS RR を対象とする（DS RR の署名）。

9.7.1 署名の検証結果の記録と伝達

一部のネームサーバは、一部のゾーンに関しては権威を持っているが、ほかのゾーンに関しては権威を持っていない可能性がある。ネームサーバが権威を持っていないゾーンでは、ネームサーバはそれらのゾーンに関する以前のクエリから RRset をキャッシュする機能を果たす。DNSSEC 対応キャッシング／リゾルビィングネームサーバのキャッシング機能には、それに加えて別の作業がある。これらの追加作業は、レスポンス RRset のセキュリティ状態の分類と、それに応じたレスポンス RRset のキャッシングに関するものである。RRset が取り得る 3 つの状態を以下に示す。

+ **セキュア。**DNSSEC 対応キャッシングネームサーバ（またはリゾルバ）は、RRset に対応する RRSIG の検証に成功した場合、その RRset をセキュアとして分類する。これは、その RRset からトラストアンカにいたるまでの有効な信頼の連鎖を形成できることを意味する。この RRset はキャッシュに格納され、データが（TTL に基づいて）最新ではないか、または（RRSIG の有効期間に基づいて）検証可能でなくなったとみなされたときに消去される。
+ **非セキュア。**DNSSEC 対応キャッシングネームサーバは、RRset がセキュアでないことが証明できる場合には、その RRset を非セキュアとして分類する。これは、DNSSEC のセキュリティ RR がレスポンスに含まれておらず、受信者がそのような RR を受信しないとわかっていたことを意味する。これは、署名なしゾーンへの委任（参照レスポンスに DS RR が含まれていない委任）を受け取ったときに発生する。非セキュア RRset は、セキュア RRset と同じ方法で処理されるが、エンドシステムのローカルポリシーによっては、レスポンス内のセキュアでない委任やデータを信頼しないように規定している可能性がある。
+ **偽装。**DNSSEC 対応キャッシングネームサーバは、署名（RRSIG RR）の検証に失敗するか、または署名に誤ったフィールドが含まれている（RRSIG の期限切れなど）場合に、レスポンスを偽装されたものとして分類する。これらの RRset の処理方法は、キャッシュポリシーによって決まる。これらの RRset は、破棄されるか、偽装であることが判明した RRset だけを含む特別な不良キャッシュに格納される。

DNSSEC 対応キャッシングネームサーバは、セキュリティ非対応リゾルバからクエリを受け取ると、そのキャッシュからセキュアデータまたは非セキュアデータを返す。このクエリは、セキュリティ非対応であるため、レスポンスには DNSSEC の RR タイプが含まれておらず、通常の DNS 処理だけが適用される。

DNSSEC 対応リゾルバは、DNSSEC 対応キャッシングサーバからセキュアデータまたは非セキュアデータを受信するが、このときの DNS レスポンスのヘッダには特定のビットがセットされている。このヘッダビットは、AD（Authenticated Data）ビットと呼ばれ、レスポンス内の RRset がキャッシングネームサーバによって実行されたすべてのセキュリティチェックに合格したことを示す。そのサーバのセキュリティ状況に応じて、クライアントはこのレスポンスを受け入れたり、独自のセキュリティチェックを実行したりする。

場合によっては、キャッシングネームサーバが偽装とみなしたデータをクライアントが要求することもある。この場合、クライアントは DNS メッセージヘッダの CD（Checking Disabled）ビットをセットしてクエリを送信する。このビットは、DNSSEC 対応キャッシングネームサーバに対して、エラーメッセージではなく、不良キャッシュに格納されている偽装データを使って応答するように指示する。この場合、クライアントシステムはレスポンス RRset に対して独自のセキュリティチェックを実行する必要がある。これらのタイプのレスポンスに関して、サーバは AD ビットをセットしないことにより、そのレスポンスがサーバによって実行されたセキュリティチェックの一部に合格していないことを示す。

9.8 DNS クエリ／レスポンスに対する追加的な保護策

DNS クエリ／レスポンストランザクションの保護に関する DNSSEC 仕様は、以下の通信を対象としている。

+ リモート権威ネームサーバからローカル（組織の）リゾルビィングネームサーバへの DNS レスポンス
+ リモートキャッシングネームサーバからローカルリゾルビィングネームサーバへの DNS レスポンス

しかし、DNS のクエリの大部分は（インターネットリソースへのアクセスを必要としているクライアントソフトウェアに代わって）スタブリゾルバから発信されるため、DNS レスポンスメッセージの保護をスタブリゾルバからリゾルビィングネームサーバへの経路にも拡張する必要がある。この経路（*DNS ラストホップ*ともいう）の保護策は、スタブリゾルバの性質とネットワークの構成によって決まる。

スタブリゾルバには、DNSSEC 非対応、DNSSEC 対応検証機能なし、および DNSSEC 対応検証機能付きがある。現在導入されているスタブリゾルバのほとんどは DNSSEC 非対応である。つまり、返された RRset に対応する署名を検証する機能を持たないだけでなく、真正性が確認された（署名が検証された）レスポンスと（ローカルリゾルビィングネームサーバを通過した）真正性の確認されていないレスポンスを区別することもできない。DNS クエリ／レスポンスに対して完全なエンドツーエンドの保護を行う場合は、これらのタイプのスタブリゾルバに対する最低限の要件として、DNS 名前解決サービスを提供するリゾルビィングネームサーバとスタブリゾルバを接続しているチャネルを通ってくるレスポンスの送信元の真正性確認とデータ完全性保護を行う機能が必要である。この機能は、TSIG で規定されている HMAC の手法を使ってこれらのタイプのスタブリゾルバに導入できる（HMAC は、ゾーン転送と動的更新のトランザクションを保護するために実装されるため）。あるいは、組織が IPsec（IP Security）などのほかのネットワークセキュリティメカニズムを使ってこの機能を提供することもできる。どのようなメカニズムを使ってラストホップ（リゾルビィングネームサーバからスタブリゾルバまで）のチャネルセキュリティを確保するにしても、この機能は DNSSEC 非対応スタブリゾルバだけでなく、DNSSEC 対応検証機能なしスタブリゾルバにも存在すべきである。DNSSEC 対応検証機能なしスタブリゾルバは、この信頼できる経路を利用して、受信したレスポンスメッセージのメッセージヘッダに含まれる AD ビットの設定を検査できる。これらのタイプのスタブリゾルバは、リゾルビィングネームサーバがレスポンスの Answer セクションと Authority セクションに含まれるすべてのデータについて署名の有効性を問題なく確認できたかどうかを判定するためのヒントとしてこのフラグビットを使用できる。

状況によっては、スタブリゾルバと DNS サービスを提供するリゾルビィングネームサーバとのあいだに信頼できる経路を確立することが現実的に不可能な場合がある。たとえば、リゾルビィングネームサーバが組織の管理ドメイン配下になく、ISP によって運営されている場合がある。このような状況で DNS クエリ／レスポンスに対するエンドツーエンドの保護を保証するには、DNSSEC 対応スタブリゾルバまたはメッセージ認証（TSIG または SIG（0））を使って信頼できるキャッシングサーバと通信できるスタブリゾルバを用意することに以外にない。DNSSEC 対応スタブリゾルバが独自の署名検証を行うことをローカルリゾルビィングネームサーバに示すには、クエリメッセージの CD（checking disabled）ビットをセットする。あるいは、特定のリゾルビィングネームサーバを使用し、TSIG／SIG（0）を使ってメッセージ認証を提供するように、スタブリゾルバを手動で設定する必要がある。

9.9 DNSSEC 対応ゾーンにおける動的更新

4.3 項では、動的更新の過程でゾーンファイルに対して行われるさまざまな論理操作を示した。4 つの論理操作は、RR の追加と RR の削除という 2 つの基本操作から成るとみなすことができる。RR の更新は、追加と削除という 2 つの基本操作を組み合わせたものとみなすことができる。RR の追加と削除では、非セキュアゾーンのゾーンファイルに含まれる残りの RR に対しては追加の操作はない。しかし、セキュアゾーンの場合、名前空間のすべてのギャップを埋めるために 1 つの NSEC RR（および対応する RRSIG RR）が存在する。

ゾーン内のすべての一意の所有者名に対して 1 つの NSEC RR が存在する。この NSEC RR は、正規順序（ゾーン内のドメイン名を辞書順に並べ替えることによって得られる順序）における次の所有者名を指し示している。正規順序における最後の所有者名に対応する NSEC RR は、ゾーン頂点名（つまり、ゾーン名）を指し示している。したがって、NSEC RR は概念的にはゾーン内の個々のドメイン名をたどる循環リンクを形成している。

example.com ゾーンに対する NSEC RR の編成と内容を考えてみる。以下は、このゾーンに含まれる個々のドメイン名の正規順序であるとする。

```
example.com.   IN  SOA  ns.example.com. admin.example.com.
( 12985 3600 2700 8000 3600 )
           IN  RRSIG ( SOA )
           IN  NS  ns.example.com.
           IN  RRSIG ( NS )
           IN  MX  mail.example.com.
           IN  RRSIG ( MX )

ns.example.com.  IN  A  192.253.101.7

mail.example.com.  IN  A  192.253.101.8

marketing.example.com.   IN  A  192.253.101.9
           IN  RRSIG ( A )
           IN  MX  mail.example.com.
           IN  RRSIG ( MX )

sales.example.com.   IN  NS  ns.example.com.
           IN  RRSIG ( NS )

www.example.com.   IN  A  192.253.101.10
           IN  RRSIG ( A )
```

このゾーンの名前ごとに見つかるドメイン名と RR タイプに関連する名前空間のギャップを埋める NSEC RR の疑似フォーマット（重要なフィールドのみを含む）は、次のとおりである。

```
example.com.  IN  NSEC  marketing.example.com. (NS SOA MX RRSIG
NSEC)

marketing.example.com.  IN  NSEC  sales.example.com. (A MX
RRSIG NSEC)

sales.example.com.  IN  NSEC  www.example.com. (NS RRSIG NSEC)

www.example.com.  IN  NSEC  example.com. (A RRSIG NSEC)
```

ゾーン内の個々のドメイン名と同じ数の NSEC RR がある。所有者名が example.com である NSEC RR は、正規順序における次のドメイン（marketing.example.com）を指し示している。marketing.example.com および sales.example.com の各ドメインに関する NSEC RR についても同じことがいえる。最後のドメイン名（www.example.com）に関する NSEC RR は、ゾーン内の最初のドメイン名（example.com）を指し示している。

（ゾーン内に存在しない）「package.example.com IN A」に対するクエリを受け取ると、権威サーバはその名前がゾーンに存在しないことを証明する NSEC RR を使って応答する。この場合、サーバからのレスポンスは、その名前が存在しないことを示す通常の DNS 応答と以下の要素とで構成される。

+ 「marketing.example.com.」と「sales.example.com」のあいだに権威を持つ名前が存在しないことを示す marketing.example.com. NSEC RR
+ クエリに一致するように展開できたワイルドカード名がゾーン内に存在しないことを証明する www.example.com. NSEC RR（ゾーン内の最後のドメイン）[13]
+ 先述の各 NSEC レコードに付随する真正性確認用の RRSIG RR

以下では、次の操作に対して必要となる NSEC RR の変更内容を示す。

+ 既存ドメインへの新しい RR タイプの追加
+ 既存ドメインからの RR タイプの削除
+ ゾーンへの新しいドメイン名の追加
+ ゾーンからのドメイン名の削除

既存ドメインへの新しい RR タイプの追加：

www.example.com に新しいメールホストが追加されたとする。この変更により、このドメイン名に MX RR を追加する必要がある。したがって、www.example.com の NSEC RR を次のように変更する必要がある。

```
www.example.com.  IN  NSEC  example.com. (A MX RRSIG NSEC)
```

既存ドメインからの RR タイプの削除：

[13] NSEC RR を使ったネガティブレスポンスについては、RFC 4035 を参照すること。

example.com がマーケティング部の従業員のために独立の電子メールセットを用意する必要がなくなったと判断したとする。この変更により、ドメイン marketing.example.com からメールサーバ(RRType=「MX」)を削除する必要がある。変更後の NSEC RR は次のようになる。

```
marketing.example.com.  IN  NSEC  sales.example.com. (A RRSIG
NSEC)
```

ゾーンへの新しいドメイン名の追加:

顧客がオンラインで商品を注文できるようにするために、websales.example.com という独立したドメインを追加することにした。別のネームサーバと一連の新しいホストも追加する。この新しいドメインのために、ゾーンファイルに対して以下の変更を加える必要がある。

+ 新しく追加するドメインに対応する新しい NSEC RR。この場合は、ドメイン名 websales.example.com に対応する NSEC RR を追加すべきである。また、その正規順序における位置を決定する必要がある。この NSEC RR は、新しい正規順序における次のドメイン名を指し示すように設定すべきである。以下に示すように、(RR タイプのリストフィールドの NS コードと A コードによって)ネームサーバとホストの存在をこの新しい RR に反映し、対応する RRSIG RR を生成する必要がある。また、RR リストには DS RR が示されていないため、新しい委任はセキュアではない。ドメイン内を参照するクライアントは、応答に DNSSEC の情報があることを期待すべきでない。

  ```
  websales.example.com.  IN  NSEC  www.example.com. (A NS RRSIG
  NSEC)
  ```

+ 追加するドメイン名の(正規順序における)直前にあたるドメイン名に対応する NSEC RR は、新しく追加したドメインを指し示すように変更しなければならない。ここでは、この NSEC RR が websales.example.com を指し示すべきである。新しく生成された NSEC RR を対象とする新しい RRSIG RR を生成する(ZSK で署名する)必要がある。

  ```
  sales.example.com.  IN  NSEC  websales.example.com. (A RRSIG
  NSEC)
  ```
+

ゾーンからのドメイン名の削除:

すべての販売を Web 経由で行うことを決定したとする。この機能に対応するためのホストを新しいドメインである websales.example.com に作成したため、sales.example.com のホスト上にあるアプリケーションの使用が中止され、それらのホストが不要になった。これは、1 つのドメインに属するすべての RR をゾーンから削除すべきであることを意味している。この変更では、以下の操作を行う必要がある。

+ 削除するドメインに対応する NSEC RR を削除すべきである。したがって、ここではドメイン sales.example.com に関する NSEC RR を削除する必要がある。

+ 削除するドメインの直前にあるドメイン名に対応する NSEC RR を、削除するドメインの直後にある(つまり、削除する NSEC RR が指し示す)ドメインを指し示すように変更する必要がある。ここでは、marketing.example.com に対応する NSEC RR が次のようにドメイン websales.example.com を指し示すべきである。

  ```
  marketing.example.com.  IN  NSEC websales.example.com. (A MX
  RRSIG NSEC)
  ```

9.10 推奨事項のまとめ

以下に、このセクションで取り上げた主な推奨事項をまとめる。

+ **チェックリスト項目 14**：DNSSEC の署名付きゾーンまたはクエリの署名付きゾーンを導入するネームサーバは、DNSSEC 処理を行うように設定すること。

+ **チェックリスト項目 15**：KSK の鍵が侵害されると DNS への影響が大きいため、KSK の鍵サイズは十分に大きくすること。

+ **チェックリスト項目 16**：DNSSEC 対応プライマリ権威ネームサーバが動的更新をサポートしない場合は、ZSK と KSK の両方に対応する秘密鍵をネームサーバに保管しないこと。動的更新がサポートされている場合は、ZSK に対応する秘密鍵だけをネームサーバに保管し、適切なディレクトリ／ファイルレベルのアクセス制御リストまたは暗号化に基づく保護策を取ること。

+ **チェックリスト項目 17**：KSK を使った署名の生成は、オフラインに保存されている KSK 秘密鍵を使ってオフラインで行うこと。その後、DNSKEY RRSet を、その RRSIG RR とともにプライマリ権威ネームサーバに読み込むことができる。

（本ページは意図的に白紙のままとする）

10. DNS 内容制御を通じて情報露出を最小限に抑えるためのガイドライン

DNS セキュリティ拡張では、送信元の真正性確認とデータの完全性保護のみ達成できる。DNS データは公開するという設計上の決定から、この保護策では機密性は提供されない。分割 DNS のような機能は、内部のネットワーク情報がグローバルなインターネット上に露出するのを防ぐ手段となるが、こうした機能は DNS プロトコル仕様に正式には含まれていない。

攻撃者が、DNS を通じて組織のネットワークに関して重要な情報を知り、その情報を利用して攻撃を開始できる可能性がある。公開サーバの IP アドレスなどのいくつかの要素に関しては、この可能性は避けられない。しかし、ネットワークへの露出を最小限に抑えるために、ゾーンファイルの生成時に DNS 管理者がとり得る手順がいくつかある。以下にこれらの手順を示す。この手順は、ゾーンの署名前に行うべきである。完全に機密を保持する必要のあるネットワーク情報は、そもそも DNS に公開すべきではない。

10.1 SOA RR のパラメータ値の選択

DNS 管理者が最初にとるべきアクションは、SOA リソースレコードデータの値が正しいか確認することである。この RR の値は、ゾーンのプライマリサーバとセカンダリサーバのあいだのやり取りを制御する。たとえば、セカンダリサーバが定期的にプライマリサーバからゾーン転送を実行すべきタイミングなどである。このデータには、クライアントリゾルバに DNSSEC 以外のネガティブレスポンスをキャッシュする期間を示す、最小 TTL 値フィールドも含まれている。以下に、これらのフィールドの推奨値をいくつか示す[14]。

+ **シリアル番号（Serial Number）**。SOA RDATA の*シリアル番号*は、ゾーンに変更が加えられ、ゾーン転送を実行する必要があることをセカンダリネームサーバに示すために使用される。このシリアル番号は、ゾーンデータに変更が加えられるたびにインクリメントする必要がある。

+ **更新値（Refresh Value）**。*更新値*は、セカンダリサーバに対して、ゾーン転送を行う間隔を秒数で示す。頻繁に更新されるゾーンに対しては、この値を小さくする必要がある（20 分から 2 時間）。頻繁には更新されないゾーンに対しては、大きな値を使用することもできる（2 時間から 12 時間）。署名付きゾーンに対しては、セカンダリゾーンに期限切れの RRSIG を持つゾーンが含まれないようにする必要があるため、この値は RRSIG の有効期間を超えてはならない。この値はプライマリサーバ側の帯域幅の制約に左右される場合もある。プライマリサーバが更新時に NOTIFY メッセージを発行した場合、セカンダリサーバはこの更新値がタイムアウトするのを待たずに、即座にゾーン転送を実行する。

+ **リトライ値（Retry Value）**。*リトライ値*は、セカンダリサーバがゾーン転送の試みが失敗した場合に、次にゾーン転送を実行するまでに待機すべき期間を示す。この値は、更新値の約数にする必要がある。更新値が上に示した範囲の場合、このフィールドが取り得る値の範囲は、5 分から 1 時間である。

+ **有効期限値（Expire Value）**。*有効期限値*は、セカンダリサーバが更新のためにプライマリサーバにアクセスできなくなった場合に、ゾーン情報を有効とみなせる時間の長さを示す。このフィールドによりセカンダリサーバは、ネットワーク障害が解決するまで稼動を続けることができる。この値は、ゾーンに変更が加えられる頻度と、ネームサーバ間の接続の信頼性に応じて異なる。この値は更新値の倍数で、可能であれば 2〜4 週間程度に設定する。

---

[14] 詳細については、http://www.ietf.org/rfc/rfc1912.txtから入手可能な RFC 1912、『*Common DNS Operational and Configuration Errors*』を参照すること。

+ **最小 TTL(Minimum TTL)**。*最小 TTL 値*は、ネガティブキャッシングの場合に使用するデフォルト値である。最小 TTL 値は、ゾーンにおける情報の変更頻度によって異なる。静的なゾーンの場合、この値は大きくてもかまわないが、動的なゾーンの場合は小さい値にすべきである。最小値は 5 分が妥当であり、推奨範囲は 30 分から 5 日である。

チェックリスト項目 18:ゾーンの SOA RR の更新値は、更新の頻度を考慮して選択すること。ゾーンが署名付きの場合、更新値は RRSIG の有効期間よりも小さい値であること。

チェックリスト項目 19:ゾーンの SOA RR のリトライ値は、更新値の 10 分の 1 であること。

チェックリスト項目 20:ゾーンの SOA RR の有効期限値は、2〜4 週間であること。

チェックリスト項目 21:TTL の最小値は、30 秒から 24 時間のあいだであること。

10.2 情報提供 RRType からの情報漏えい

DNS の RR には、ネットワーク、ホスト、サービスについての情報を人やアプリケーションに伝える手段となる種類の RR がいくつかある。具体的には、担当者(RP:Responsible Person)レコード、ホスト情報(HINFO:Host information)レコード、場所(LOC:Location)レコード、任意のテキスト文字列リソースレコード(TXT:Text string)などがある。これらのレコードタイプは、ユーザを信用して情報を提供することを目的としているが、ネットワークホストの悪用を企む攻撃者には、そのネットワークホストに関する知識を与えることにもなる。たとえば攻撃者は、セキュリティ上の弱点があることがわかっている OS やプラットフォームをリスト表示するホストを探すために、HINFO レコードを求めてクエリを実行する可能性がある。したがって、ゾーンにこれらのレコードタイプを含める前に、十分に注意する必要がある。実際のところ、これらはすべて除外しておくのが最善である。

TXT リソースレコードタイプに関しては、より細心の注意を払うべきである。DNS 管理者は、TXT RR に含まれているデータが情報漏えいとなるか、あるいは必要な情報であるかを判断する必要が生じる。こうした判断は、ケースバイケースで行う必要がある。

チェックリスト項目 22:DNS 管理者は、ゾーンファイルに、HINFO、RP、LOC、そのほか攻撃者にとって有用な情報を公開する可能性のある RR タイプ、あるいは分割 DNS を使用している場合はゾーンの外部ビューを含めるべきではない。

チェックリスト項目 23:DNS 管理者は、ゾーンファイルに TXT RR を追加する前に、TXT RR に含まれているデータに情報漏えいの可能性がないかを必ず確認すること。

10.3 鍵の侵害を最小限に抑えるための RRSIG の有効期間の使用

鍵の侵害の影響を最小限に抑えるためにゾーン管理者が採用できる最善の方法は、対象ゾーンとその親ゾーンの RRSIG の有効期間を制限することである。この方策により、侵害した鍵を攻撃者が利用し、レスポンスを偽造できる時間が制限される。ZSK を侵害した攻撃者は、その KSK の署名の有効期間内のみ鍵を使用できる。KSK を侵害した攻撃者は、委任元である親の DS RR を対象とする RRSIG の署名期間のあいだのみ、その鍵を利用できる。したがって、これらの有効期間を短くすることは有効であるが、それにより再署名が必要となる頻度は高くなる。

侵害された ZSK の影響を最小限に抑えるために、ゾーン管理者はゾーンの DNSKEY RRSet（そのゾーンの ZSK および KSK を含む RRSet）を対象とする RRSIG の署名の有効期間を 1 週間に設定すべきである。ZSK のロールオーバを実行せずに DNSKEY RRSet に再署名することもできるが、計画的な ZSK ロールオーバも一定の間隔で実行すべきである。

侵害された KSK の影響を防ぐために、委任元である親は DS RR を対象とする RRSIG の署名の有効期間を数日から 1 週間に設定すべきである。この再署名には、親の ZSK の頻繁なロールオーバを必要としないが、計画的な ZSK ロールオーバも一定の間隔で実行すべきである。

チェックリスト項目 24：ゾーンの DNSKEY RRSet を対象とする RRSIG の有効期間は、2 日から 1 週間の範囲であること。この値は、鍵の侵害から生じる脆弱性の存続期間の短縮に役立つ。

チェックリスト項目 25：委任された子を持つゾーンの有効期間は、委任された子の DS RR を対象とする RRSIG に対しては、数日から 1 週間であること。この値は、KSK の侵害から生じる、子ゾーンの脆弱性の存続期間の短縮に役立つ。

10.4 推奨事項のまとめ

以下に、このセクションで取り上げた主な推奨事項をまとめる。

+ **チェックリスト項目 18**：ゾーンの SOA RR の更新値は、更新の頻度を考慮して選択すること。ゾーンが署名付きの場合、更新値は RRSIG の有効期間よりも小さい値であること。
+ **チェックリスト項目 19**：ゾーンの SOA RR のリトライ値は、更新値の 10 分の 1 であること。
+ **チェックリスト項目 20**：ゾーンの SOA RR の有効期限値は、2～4 週間であること。
+ **チェックリスト項目 21**：TTL の最小値は、30 秒から 24 時間のあいだであること。
+ **チェックリスト項目 22**：DNS 管理者は、ゾーンファイルに、HINFO、RP、LOC、そのほか攻撃者にとって有用な情報を公開する可能性のある RR タイプ、あるいは分割 DNS を使用している場合はゾーンの外部ビューを含めるべきではない。
+ **チェックリスト項目 23**：DNS 管理者は、ゾーンファイルに TXT RR を追加する前に、TXT RR に含まれているデータに情報漏えいの可能性がないかを必ず確認すること。
+ **チェックリスト項目 24**：ゾーンの DNSKEY RRSet を対象とする RRSIG の有効期間は、2 日から 1 週間の範囲であること。この値は、鍵の侵害から生じる脆弱性の存続期間の短縮に役立つ。
+ **チェックリスト項目 25**：委任された子を持つゾーンの有効期間は、委任された子の DS RR を対象とする RRSIG に対しては、数日から 1 週間であること。この値は、KSK の侵害から生じる、子ゾーンの脆弱性の存続期間の短縮に役立つ。

（本ページは意図的に白紙のままとする）

11. DNS のセキュリティ管理業務のガイドライン

セクション 9 では、DNS クエリ／レスポンストランザクションの保護のために DNSSEC 機能を導入し、使用するための各種作業を取り上げた。このセクションでは、DNSSEC 対応のエンタープライズレベルゾーンにおける定期的なセキュリティ管理業務（およびチェックリスト）と、当該業務を安全に実施する方法を取り上げる。

11.1 計画的な鍵のロールオーバ（鍵の寿命）

ゾーン署名（ZSK）と鍵署名（KSK）に使用する鍵は、一定期間使用すると脆弱になる（突き止められやすくなる）ため、変更する必要がある（一般にムーアの法則、すなわちコンピュータの処理能力は時間の経過とともに指数的に高くなる、という予測に起因するが、そのほかにも以下に述べる要素も原因である）。秘密鍵の侵害は、任意のサイトがその秘密鍵を用いて偽装 RRSet に署名することにより、ゾーンを偽装できることを意味する(これにより、ゾーンファイルに署名する目的が損なわれる)。鍵のロールオーバは、計画的なイベントとして行われる場合と（計画的なロールオーバ）、緊急時対応の結果として行われる場合がある（緊急時のロールオーバ）。緊急時のロールオーバは、次のいずれかの理由により生じる。

+ ゾーンの秘密鍵が侵害された。
+ ゾーンの秘密鍵を喪失し、RRSIG の期限切れ前にゾーンを更新する必要が生じた。

計画的な鍵のロールオーバでは、対象の鍵を変更しなければならない期限（または変更頻度）を、いくつかの要因によって決定する。

+ ゾーンファイルのサイズが大きいほど、署名の生成対象となるデータセットが大きくなり、秘密鍵を突き止めるプロセスが容易になる。
+ 秘密鍵のサイズは小さいほど、解読が容易になる。

これらの要因に基づき、各ゾーンの ZSK および KSK の鍵のロールオーバの必要な頻度が定められる。先述のとおり、KSK（KSK 秘密鍵）は DNSKEY RRSet にのみ署名するために使用され、ZSK（ZSK 秘密鍵）は、ゾーンファイル全体に署名するために使用される。データ量のほかに、次のような状況でも、ZSK がはるかに高い頻度で使用される。

+ 新しい RR が追加されたとき（たとえば、新しいメールサーバが追加され、新しい MX RR がゾーンファイルに追加されたなど）。
+ 既存の RR の RDATA が変更されたとき（たとえば、サーバの IP アドレスが変更されたために、対応する A RR を差し替える必要があるなど）。
+ RRSIG RR の署名の期限が切れたとき。

処理するデータ量と使用頻度のため、ZSK 秘密鍵のサイズは、ディジタル署名生成処理によって消費される CPU サイクル全体の重要な要素となる。このため使用される ZSK は比較的小さい。

チェックリスト項目 26：KSK のロールオーバは、ZSK よりも低い頻度で行う必要がある。KSK に対して推奨されるロールオーバの頻度は、1 年に 1 回（RSA/SHA1 を使用した 2048 ビットのサイズの場合）である。これに対して ZSK のロールオーバは、毎月行う必要がある（RSA/SHA1 を使用した 1024 ビットの鍵サイズ）。

鍵のロールオーバによる DNS の残りの部分への影響は、セキュアゾーンが局所的にセキュアであるかグローバルにセキュア（信頼の連鎖の一部）であるかによって異なる。

鍵のロールオーバに必要な手順の詳細については、DNSSEC 操作に関する IETF の文書を参照すること[15]。

11.1.1 局所的なセキュアゾーンにおける鍵のロールオーバ

局所的にのみセキュアなゾーンには、ZSK のほか、クライアントリゾルバにおいて信頼できる鍵として構成されている KSK を持たせることが考えられる。局所的に署名されるゾーンに対するものでも、KSK があれば ZSK のロールオーバが容易になるが、いずれかの鍵がロールオーバされた場合には、いくつかの難題が生じる。あるゾーンで ZSK を変更し、変更されていないままの KSK がある場合、1 つだけ問題が生じる。それは、離れた場所にあるリゾルバのキャッシュやネームサーバのキャッシュに古い鍵が残っていることが考えられる場合の、新しい鍵の導入方法である。

この場合の解決策は、ロールオーバ前に新しい公開鍵を事前公開することである。DNS 管理者は、新しい鍵が署名の生成に使われる前に、それをゾーンファイルの DNSKEY RR として公開する必要がある。この手順を以下に示す。

+ 新しい鍵ペアを生成する。
+ 新しい鍵ペアの公開鍵をゾーンファイル（DNSKEY レコード）に追加する。
+ 現在有効な鍵ペアの秘密鍵と、KSK（もしあれば）を使用してゾーンに署名する。
+ DNSKEY RRSet の TTL、またはゾーン SOA レコードの MinTTL の期間（どちらか長いほう）が経過するまで待機する。
+ ゾーン鍵セットから古い DNSKEY RR を削除し、さらに期限切れの RRSIG RR を削除する。
+ 新しい ZSK を用いて DNSKEY RRSet に再署名する。

DNS 管理者にとって一番よいのは、ZSK のロールオーバを継続的に実施することであろう。管理者は最初の 3 つの手順を実行し、鍵セットから古い DNSKEY が削除されるまで無期限に待機でき、ゾーンの RRSIG が期限切れになると古い DNSKEY でゾーンを署名し続けることもできる。この手順により、管理者は緊急時の鍵ロールオーバをより効率よく実施できる（下記参照）。

[15] インターネット標準案『*DNSSEC Operational Practices*』（http://www.ietf.org/internet-drafts/draft-ietf-dnsop-dnssec-operational-practices-08.txt）を参照。

新しい公開鍵を事前公開するゾーンでは次の項目に従う必要がある:

チェックリスト項目 27: 公開鍵を事前公開するセキュアゾーンは、鍵のロールオーバの実行時期よりも、少なくとも TTL 期間 1 回分の期間だけ先立って行うこと。

チェックリスト項目 28: 古い公開鍵の削除後は、そのゾーンではゾーンファイル内のほかの鍵（DNSKEY RR）に基づいて新しい署名（RRSIG RR）を生成すること。

KSK のロールーバでは、セキュアゾーンではどのリゾルバがその公開鍵をトラストアンカとして格納しているかわからない。ネットワーク管理者は、公開鍵をトラストアンカとして格納したリゾルバ管理者と連絡をとるための帯域外の手段（電子メールなど）があれば、適切な警告を発信し、新しいトラストアンカを配布する信頼のおける手段を確立する必要がある。さもなければ DNS 管理者は、リゾルバ管理者に新しい KSK を認識してもらうための十分な時間を提供するために、時間に余裕をもって新しい KSK を事前公開する以外に手立てはない。

11.1.2 連鎖したセキュアゾーンにおける鍵のロールオーバ

グローバルにセキュアなゾーンは、2 つの鍵の組み合わせを使用する。すなわち、ZSK と KSK である。

11.1.3 連鎖したセキュアゾーンにおける ZSK 鍵のロールオーバ

グローバルにセキュアなゾーンにおける ZSK のロールオーバの手順は、局所的にセキュアなゾーンにおける ZSK 鍵のロールオーバと同じである（11.1.1 項を参照）。

11.1.4 連鎖したセキュアゾーンにおける KSK 鍵のロールオーバ

KSK（KSK 公開鍵）は、セキュアゾーンのセキュアな親が信頼を提供するための鍵である。この信頼は、子の KSK のハッシュが含まれている DS RR と呼ばれる新しい RR タイプを使用して提供される。委任元である親はこの DS RR に自身の ZSK を用いて署名する。あるゾーンでその KSK を変更すると、ゾーン自身と親の間の信頼関係は断たれる。この信頼関係を維持するために、KSK のロールオーバを行うゾーンでは次のことを行う必要がある。

+ 新しい KSK を生成し、それをゾーン鍵セットに追加する。
+ 新しい KSK と古い KSK（もうすぐ期限切れになる鍵）を用いてゾーン鍵セットに署名する。
+ 新しい KSK をその親へ、親がその真正性を確認できる方法で伝える。
+ さらに親は、新しい KSK のハッシュを含んだ新しい DS RR（これが古い DS RR と交換される）を生成し、新たに生成した DS RR に署名する必要がある。

親が既存の信頼の連鎖に基づいて新しい KSK（KSK2 と呼ばれる）の真正性を確認できるように、鍵のロールオーバを実行するゾーン（すなわち子ゾーン）は、既存の鍵の DNSKEY RR と新しい KSK2 の DNSKEY RR を一緒に使用して DNSKEY RRSet を生成する。次に、2 つの署名（2 つの RRSIG RR）を生成する。すなわち、既存の KSK の秘密鍵を使用する署名と、新しい KSK の秘密鍵を使用する署名である。その後、新しく生成した DNSKEY RRSet を RRSIG RR（KSK および KSK2 にそれぞれ対応する 2 つの署名）と併せて、親に送信する。親に送信する RR のセットを擬似的な形式で以下に示す。

```
example.com   DNSKEY  <key-id: 43543> /* 新しい KSK */
example.com   DNSKEY  <key-id: 78546> /* 既存の KSK */
example.com   DNSKEY  <key-id: 98342> /* ZSK */
example.com   RRSIG(DNSKEY) <signer:example.com signing-key:
78546>  /* 既存の KSK で署名された DNSKEY RRSet 全体 */
example.com   RRSIG(DNSKEY)  <signer:example.com signing-key:
43543>  /* 新しい KSK で署名された DNSKEY RRSet 全体 */
```

この情報を受け取ると、親は次のことを行う。

+ 信頼している KSK（すなわち 78456）が本当に、新しい KSK（KSK2）を含んだ新しく生成された DNSKEY RRSet に署名したことを検証する。これを行うために、新しく生成された DNSKEY RRSet、最初の RRSIG RR（署名鍵として 78456 と示されている）、および自身の DS RR を使用する。

+ さらに、2 番目の RRSIG RR（署名鍵として 43543 と示されている）を DNSKEY RRSet と突き合わせて検証することにより、KSK2 の真正性（子ゾーンが KSK2 の秘密鍵を所有している）も検証する。

+ 次に、新しい KSK（KSK2）のハッシュを含んだ新しい DS RR を生成する。

+ 親はその後、KSK2 の新しい DS RR を対象とする RRSIG を生成する必要がある。

これらの作業を親が実行すると、子の観点からは KSK ロールオーバプロセスは基本的に完了したことになる。親ゾーンではその後、自身のゾーンファイルを新しい DS RR で更新する必要がある。これは、ほかの RRSet の更新と同様であり、必要に応じて新しい RRSIG を生成する処理が含まれる。

+ 親ゾーンは新しい DS RR（DS2）を委任先の DS RRset に追加する。

+ 親は新しい DS RR が伝播されるまで（DS RRset の TTL の時間）待機する。

+ 伝播されたら、親は期限切れの DS RR を DS RRset から削除できる。これがきっかけとなりゾーンの再署名が行われる。

委任された子ゾーンは、親が適切な DS 更新を実行したことを確認し、KSK のロールオーバのあいだ、真正性確認の連鎖が変更されないことを確認するまで、期限切れの KSK を鍵セットに保持しておくべきである。委任元の親が子に対する委任情報を更新したことを子ゾーンが確認したら、子ゾーンの管理者は古い KSK を鍵セットから削除し、ZSK と新しい KSK を用いて鍵セットに再署名する。

11.2 鍵の緊急ロールオーバ

鍵の緊急ロールオーバは、ゾーン内の 1 つまたは複数の鍵（ZSK または KSK）が侵害された場合や、非公開の構成要素が失われ、再署名の必要が生じた場合に行われる。この種のロールオーバは計画されたものではないため、ゾーン管理者がこの種のロールオーバを実施する必要がある場合、真正性確認の連鎖が断たれる可能性は高くなる。

11.2.1 ZSK の緊急ロールオーバ

DNS 管理者は、1 つ（または複数）の ZSK が侵害された場合に ZSK の緊急ロールオーバを実施するには、複数の手順を実行する必要がある。この複数の手順による復旧プロセスは、ゾーン管理者が ZSK のロールオーバを継続的に実施すべきもう 1 つの理由である。ゾーン管理者は、ゾーン鍵セットの一部として新しい DNSKEY RR をゾーンにすでに公開している場合（11.1.1 項で示した最初の 3 つの手順）、次の手順は侵害された鍵によって異なる。

現在使用中の ZSK が侵害された場合、ゾーン管理者は直ちに新しい鍵にロールオーバできる。新しい鍵は、すでに一定期間公開されているため、TTL 値が経過するまで待つ必要はない。管理者は、古い RRSIG をゾーンから除去し、予定されていたロールオーバよりも前に、新しい ZSK で再署名するだけでよい。この再署名のあとも DNS の運用は、真正性確認の連鎖が断たれることなく正常に継続される。

新しい ZSK（次に使用する予定の ZSK）が侵害された場合は、ゾーン鍵セット内にあるその鍵を直ちに交換する必要がある。この新しい鍵は、RRSIG の生成にはまだ使用されていないため、置き換えはアトミックに行うこともできる。ゾーン鍵セットを再署名するだけでよい。また、1 回の更新で、侵害された鍵を削除し、新しい ZSK と置き換えるだけでもよい。

それでもなお、攻撃者が侵害された ZSK を使用して、ゾーンからのレスポンスを偽装する危険がある。この危険は、現在の KSK が有効で、KSK のロールオーバを開始しない限り存在する。ZSK が侵害された場合、ゾーン管理者はできるだけ早く KSK のロールオーバを開始すべきである。

11.2.2 KSK の緊急ロールオーバ

あるゾーンの KSK が侵害された場合、唯一の対応はその親と KSK のロールオーバを開始することである。ただし、この場合のロールオーバは、KSK の計画的なロールオーバの手順とは異なる。この場合、古い KSK 鍵が侵害されたことと、その鍵を使用したいかなる KSK ロールオーバメッセージも受け付けてはならないことを親ゾーンの管理者に注意喚起する手段が必要である。代替鍵は、子ゾーンの同一性を確認するためにほかの何らかのセキュアチャネルを使用して転送、検証する必要があり、1 つまたは複数の DNSSEC 非対応の真正性確認の手順を含む場合がある。この手順は、子ゾーンの最初の KSK を確立するときの手順と同じでもよい。

チェックリスト項目 29：DNS 管理者は、KSK の緊急ロールオーバを実行する必要が生じたときのために、直接の親ゾーンの緊急時の連絡先を用意しておくこと。

チェックリスト項目 30：親ゾーンは、子サブゾーンの緊急 KSK ロールオーバの場合の緊急時の連絡方法を、その委任先の子サブゾーンに伝えること。また、サブゾーンの新しい KSK を取得する安全な手段が用意されている必要もある。

侵害された KSK が攻撃者に利用される可能性があるのは、親ゾーンがその KSK の安全性を保証しているあいだだけなので、露出の期間を最小限に抑える唯一の方法は、署名の有効期間（RRSIG レコード RDATA の署名有効期間開始フィールドから有効期限終了フィールドまでの時間）をできる限り短く保つことである。署名の有効期間を短くするデメリットは、再署名の頻度が高くなることである（下記参照）。ゾーン管理者は、委任 DS RRset RRSIG の署名の有効期間を決める際に、再署名に必要なリソースを考慮する必要がある。

チェックリスト項目 31：侵害された KSK が効力を持つ有効期間を縮小するために、親ゾーンの DS RRset に対する RRSIG の有効期間は可能な限り短く保つこと。推奨される有効期間は、2 日から 4 日、最長でも 7 日である。

11.3 ゾーンの再署名

次のような場合は、ゾーンファイルに再署名する必要がある（新しい RRSIG RR が生成される）。

+ 署名が期限切れになったか、または間もなく期限切れになる。
+ ゾーンファイルの内容が変更された（たとえば、動的更新の結果として）。
+ 署名鍵の 1 つが侵害されたか、または交換する予定がある。

ゾーンデータに再署名するには、2 つの方法がある。

+ **完全な再署名。**既存のすべての署名レコード（RRSIG RR）が削除され、ゾーンファイルが再ソートされ、すべての NSEC RR が再生成され、最後に新しい署名レコードが生成される。完全な再署名は、次のような状況で実行される。
  – ゾーンファイルがバックエンドのリレーショナルデータベースから生成された。
  – ゾーン管理者が、署名側で設定されている期限切れ時刻に基づいて、特定の日付/時刻に実行されるバッチジョブを準備した（UNIX では cron ジョブと呼ばれる）。
+ **増分再署名。**既存の RRSet が削除されたために NSEC RR が変更された場合や、新しい RRSet がゾーンファイルに追加されたために NSEC RR が追加された場合に、変更された RRset に対してのみ署名を生成する。増分再署名は、通常は動的更新後など、最後に行われた署名生成以降にゾーンファイルの内容に加えられた変更が小さいときに行われる。

チェックリスト項目 32：定期的な再署名は、ゾーンの RRSIG RR の期限切れフィールドにある期限の前までに実施するよう計画すること。これは、期限切れの署名が原因で署名付きゾーンが偽装されるリスクを軽減するためである。

チェックリスト項目 33：SOA RR のシリアル番号は、ゾーンファイルの再署名の前にインクリメントすること。セカンダリネームサーバは、SOA のシリアル番号の不一致に基づいてのみ更新されるため、この操作を行わないと、新しい署名が反映されない可能性がある。その結果、一部のセキュリティ対応リゾルバは署名を検証できる（そしてその結果、安全なレスポンスを得られる）が、それ以外のリゾルバは署名を検証できない可能性がある。

11.4 推奨事項のまとめ

以下に、このセクションで取り上げた主な推奨事項をまとめる。

+ **チェックリスト項目 26**：KSK のロールオーバは、ZSK よりも低い頻度で行う必要がある。KSK に対して推奨されるロールオーバの頻度は、1 年に 1 回（RSA/SHA1 を使用した 2048 ビットのサイズの場合）である。これに対して ZSK のロールオーバは、毎月行う必要がある（RSA/SHA1 を使用した 1024 ビットの鍵サイズ）。
+ 新しい公開鍵を事前公開するゾーンでは次に従う必要がある：
  – **チェックリスト項目 27**：公開鍵を事前公開するセキュアゾーンは、鍵のロールオーバの実行時期よりも、少なくとも TTL 期間 1 回分の期間だけ先立って行うこと。

– **チェックリスト項目 28**:古い公開鍵の削除後は、そのゾーンではゾーンファイル内のほかの鍵(DNSKEY RR)に基づいて新しい署名(RRSIG RR)を生成すること。

+ **チェックリスト項目 29**:DNS 管理者は、KSK の緊急ロールオーバを実行する必要が生じたときのために、直接の親ゾーンの緊急時の連絡先を用意しておくこと。

+ **チェックリスト項目 30**:親ゾーンは、子サブゾーンの緊急 KSK ロールオーバの場合の緊急時の連絡方法を、その委任先の子サブゾーンに伝えること。また、サブゾーンの新しい KSK を取得する安全な手段が用意されている必要もある。

+ **チェックリスト項目 31**:侵害された KSK が効力を持つ有効期間を縮小するために、親ゾーンの DS RRset に対する RRSIG の有効期間は可能な限り短く保つこと。推奨される有効期間は、2 日から 4 日、最長でも 7 日である。

+ **チェックリスト項目 32**:定期的な再署名は、ゾーンの RRSIG RR の期限切れフィールドにある期限の前までに実施するよう計画すること。これは、期限切れの署名が原因で署名付きゾーンが偽装されるリスクを軽減するためである。

+ **チェックリスト項目 33**:SOA RR のシリアル番号は、ゾーンファイルの再署名の前にインクリメントすること。セカンダリネームサーバは、SOA のシリアル番号の不一致に基づいてのみ更新されるため、この操作を行わないと、新しい署名が反映されない可能性がある。その結果、一部のセキュリティ対応リゾルバは署名を検証できる(そしてその結果、安全なレスポンスを得られる)が、それ以外のリゾルバは署名を検証できない可能性がある。

（本ページは意図的に白紙のままとする）

付録A—重要な用語の定義

『*セキュアなドメインネームシステム（DNS）の導入ガイド*』で使用している用語について、その一部の定義を以下に示す。以下の用語は、DNSSEC 仕様[RFC 4033、4034、4035、3833]で定義されている用語に、この文書で使われている特定の用語をいくつか加えたものである。

真正性確認の連鎖（Authentication Chain）：DNS 公開鍵（DNSKEY）RRset と Delegation Signer（DS）RRset の交互の並びで構成された署名付きデータの連鎖。連鎖内の各つながりが、次のつながりを保証する。DNSKEY RR を使用して、特定の DS RR を対象とする署名を確認することにより、その DS RR の真正性を確認することができる。その DS RR は、別の DNSKEY RR のハッシュを含んでおり、この DNSKEY RR は DS RR 内のハッシュと照合することによって真正性を確認される。この新しい DNSKEY RR は、今度は別の DNSKEY RRSet の真正性を確認し、さらにそのセットに含まれるいくつかの DNSKEY RR を使用して別の DS RR の真正性を確認する。この連鎖は、目的とする DNS データに署名する、対応する秘密鍵を持つ DNSKEY RR に到達した時点で終了する。たとえば、ルートの DNSKEY RRSet を使用して「example.」の DS RRSet の真正性確認を行うことができる。「example.」DS RRSet には、何らかの「example.」DNSKEY と一致するハッシュが含まれており、この DNSKEY の対応する秘密鍵が「example.」DNSKEY RRSet に署名する。「example.」DNSKEY RRSet の対応する秘密鍵は、「www.example.」のようなデータレコードや「subzone.example.」のような委任に関する DS RR に署名する。

認証鍵（Authentication Key）：DNSSEC 対応リゾルバによる検証が済んでおり、データの真正性確認に使用できる公開鍵。DNSSEC 対応リゾルバが認証鍵を取得する方法は 3 つある。1 つめの方法として、一般にリゾルバは少なくとも 1 つの公開鍵を認識するように設定される。この設定データは公開鍵そのものであるか、または DS RR 内にある公開鍵のハッシュである（「トラストアンカ（trust anchor）」参照）。2 つめの方法として、リゾルバは認証済みの公開鍵を使用して DS RR とそれが参照する DNSKEY RR を検証することにより取得する。3 つめの方法として、ある新しい公開鍵が、当該リゾルバによって検証済みの別の公開鍵に対応する秘密鍵によって署名されたと判断できる場合に、その新しい公開鍵を取得する。リゾルバは、新しい公開鍵を認証すべきか判断する際には、常にローカルポリシーの指針に従う必要がある。これは、ローカルポリシーが単に、リゾルバが署名を検証できたすべての新しい公開鍵を認証するというものであったとしても従う必要がある。

権威 RRSet（Authoritative RRSet）：あるゾーンの範囲内において、RRSet（名前、クラス、タイプが同じ RR）の所有者名が、ゾーン頂点と同じかまたは下位、かつそのゾーンと（もしあれば）子を分けるゾーンカットと同じかまたは上位にある名前空間のサブセット内にある場合に限り、その RRSet は権威を持つ。タイプが NSEC、RRSIG、および DS の RR は、委任先の子側ではなくゾーンカットの親側で権威を持つ RRSet の例である。

信頼の連鎖（Chain of Trust）：「真正性確認の連鎖（authentication chain）」を参照。

連鎖したセキュアゾーン（Chained Secure Zone）：ゾーンからトラストアンカまでの真正性確認の連鎖が存在する DNS ゾーン。

委任点（Delegation Point）：ゾーンカットの親側を指す名前。つまり、foo.example の場合、example ゾーンの foo.example ノードが委任点になる（「foo.example」ゾーンのゾーン頂点ではなく）。「ゾーン頂点（zone apex）」も参照のこと。

DNS 管理者（DNS Administrator）：この文書では、ゾーンデータの更新と組織の DNS サーバの運用を任されている人という意味で使われている。この用語は実際には、いくつかの正式な役割を対象とすることがあるが、そうした役割はこの文書では 1 つの用語で示している。

DNSSEC 対応ネームサーバ(DNSSEC-Aware Name Server):この文書で定義する DNS セキュリティ拡張を認識し、ネームサーバの役割を果たすもの。特に、DNSSEC 対応ネームサーバは DNS クエリの受信、DNS レスポンスの送信、EDNS0([RFC2671])メッセージサイズ拡張と DO ビット([RFC3225])のサポート、この文書で定義する RR タイプとメッセージヘッダビットをサポートするものである。

DNSSEC 対応再帰的ネームサーバ(DNSSEC-Aware Recursive Name Server):DNSSEC 対応ネームサーバと DNSSEC 対応リゾルバの両方の役割を果たすもの。使いにくいが同じ意味の言い回しとしては「再帰的なサービスを提供する DNSSEC 対応ネームサーバ」がある。「セキュリティ対応キャッシングネームサーバ」と呼ぶこともある。

DNSSEC 対応リゾルバ(DNSSEC-Aware Resolver):DNS セキュリティ拡張を認識し、リゾルバ([RFC1034]の 2.4 項に定義されている)の役割を果たすもの。特に、DNSSEC 対応リゾルバは、DNS クエリの送信、DNS レスポンスの受信、DNSSEC 仕様を認識できる。ただし、有効性確認を実施できない場合もある。

DNSSEC 対応スタブリゾルバ(DNSSEC-Aware Stub Resolver):DNS セキュリティ拡張を認識し、スタブリゾルバの役割を果たすもの。DNSSEC 対応スタブリゾルバは、スタブリゾルバ自身で DNSSEC 署名を検証しようと試みるか、その処理を行う友好的な DNSSEC 対応ネームサーバを信頼するかによって、「検証機能付き」と「検証機能なし」がある。「検証機能なしスタブリゾルバ(nonvalidating stub resolver)」も参照のこと。

セキュリティの孤島(Island of Security):委任され、署名付きであるが、委任元である親からの真正性確認の連鎖のないゾーン。つまり、委任元である親ゾーンに、当該孤島の DNSKEY RR のハッシュを持つ DS RR がないということである。セキュリティの孤島は、DNSSEC 対応ネームサーバのサービスを受け、委任されたすべての子ゾーンへの真正性確認の連鎖を提供することもできる。セキュリティの孤島とその子孫からのレスポンスは、その認証鍵を DNS プロトコル以外の何らかの信頼できる手段で真正性確認ができる場合に限り真正性確認できる。

鍵のロールオーバ(Key Rollover):使用中の既存の鍵に置き換えて、新しい鍵(対称鍵または非対称鍵のペア)を生成し、使用するプロセス。ロールオーバは、鍵が侵害された場合や、長期の使用により侵害されやすい状態になったなどの要因で実施される。

鍵署名鍵(Key Signing Key: KSK):あるゾーンに対する 1 つ以上の認証鍵に署名する秘密鍵に対応する認証鍵。通常は、鍵署名鍵に対応する秘密鍵が、ゾーン署名鍵に署名し、そのゾーン署名鍵に対応する秘密鍵がほかのゾーンデータに署名する。「ゾーン署名鍵(zone signing key)」も参照のこと。

検証機能なし DNSSEC 対応スタブリゾルバ(Nonvalidating DNSSEC-Aware Stub Resolver):この文書で述べた作業のほとんどを自身に代わって実行する 1 つまたは複数の DNSSEC 対応再帰的ネームサーバを信頼する DNSSEC 対応スタブリゾルバ。特に、検証機能なし DNSSEC 対応スタブリゾルバは、DNS クエリの送信と DNS レスポンスの受信を行い、DNSSEC 対応スタブリゾルバに代ってサービスを提供する DNSSEC 対応再帰的ネームサーバとの適切に保護されたチャネルを確立できる。「DNSSEC 対応スタブリゾルバ(DNSSEC-aware stub resolver)」も参照のこと。

検証機能なしスタブリゾルバ(Nonvalidating Stub Resolver):検証機能なし DNSSEC 対応スタブリゾルバを若干簡略化した呼称。

署名付きゾーン（Signed Zone）：RRset が署名されており、正しく設定された DNSKEY、Resource Record Signature（RRSIG）、Next Secure（NSEC）、および（任意で）DS レコードが含まれているゾーン。

トランザクション署名鍵（Transaction Signature（TSIG）Key）：TSIG RR に格納され、DNS メッセージ全体の認証に使われるメッセージ認証ハッシュを生成するための文字列。これは、暗号処理を伴うメッセージの署名と同じではない。

トラストアンカ（Trust Anchor）：あらかじめ設定された DNSKEY RR あるいは DNSKEY RR のハッシュ。検証機能付き DNSSEC 対応リゾルバは、署名付き DNS レスポンスへの真正性確認の連鎖を構築する際に、この公開鍵、またはハッシュを起点として使用する。一般に、検証機能付きリゾルバは、DNS プロトコル以外の安全かまたは信頼できる何らかの手段を通じて、そのトラストアンカの初期値を取得する必要がある。トラストアンカがあるということは、リゾルバが、トラストアンカが指すゾーンが署名付きであることを期待できることを意味する。トラストアンカは「セキュアエントリポイント」と呼ばれることもある。

署名なしゾーン（Unsigned Zone）：署名されていないゾーン。

ゾーン頂点（Zone Apex）：ゾーンカットの子の側を指す名前。「委任点（delegation point）」も参照のこと。

ゾーン署名鍵（ZSK）：ゾーンに署名するための秘密鍵に対応する認証鍵。通常ゾーン署名鍵は、鍵署名鍵（その対応する秘密鍵によって DNSKEY RRSet に署名する）と同じ DNSKEY RRSet に含まれるが、鍵署名鍵とはやや異なる目的で使われ、有効期間などいくつかの点で鍵署名鍵と異なる場合がある。認証鍵をゾーン署名鍵として指定するかは、純粋に運用上の問題である。DNSSEC の有効性確認では、ゾーン署名鍵とほかの DNSSEC 認証鍵は区別されない。また、1 つの鍵を鍵署名鍵とゾーン署名鍵の両方に使用することもできる。「鍵署名鍵（key signing key）」も参照のこと。

（本ページは意図的に白紙のままとする）

付録B—略語

『*セキュアなドメインネームシステム(DNS)の導入ガイド*』で使用している略語について、その一部の定義を以下に示す。

A	Address（アドレス）
ACL	Access Control List（アクセス制御リスト）
AD	Authenticated Data（完全性確認済みデータ）
ANSI	American National Standards Institute（米国国家規格協会）
ARP	Address Resolution Protocol（アドレス解決プロトコル）
CA	Certificate Authority（認証局）
ccTLD	Country-Code Top-Level Domain（国別トップレベルドメイン）
CD	Checking Disabled（チェック無効化）
DHCP	Dynamic Host Configuration Protocol（動的ホスト設定プロトコル）
DNS	Domain Name System（ドメインネームシステム）
DNSSEC	Domain Name System Security Extensions （ドメインネームシステムセキュリティ拡張）
DS	Delegation Signer（委任署名者）
DSA	Digital Signature Algorithm（ディジタル署名アルゴリズム）
DSS	Digital Signature Standard（ディジタル署名標準）
ECDSA	Elliptic Curve Digital Signature Algorithm（楕円曲線ディジタル署名アルゴリズム）
FIPS	Federal Information Processing Standards（連邦情報処理規格）
FISMA	Federal Information Security Management Act （連邦情報セキュリティマネジメント法）
FQDN	Fully Qualified Domain Name（完全修飾ドメイン名）
gTLD	Generic Top-Level Domain（汎用トップレベルドメイン）
HINFO	Host Information（ホスト情報）
HMAC	Hash-Based Message Authentication Code（ハッシュベースのメッセージ認証コード）
ICANN	Internet Corporation for Assigned Names and Numbers
IETF	Internet Engineering Task Force（インターネット技術特別調査委員会）
IN	Internet（インターネット）
IP	Internet Protocol（インターネットプロトコル）
ISP	Internet Service Provider（インターネットサービスプロバイダ）
IT	Information Technology（情報技術）
ITL	Information Technology Laboratory（情報技術ラボラトリ）
KSK	Key Signing Key（鍵署名鍵）
LAN	Local Area Network（ローカルエリアネットワーク）
LOC	Location（ロケーション）
MAC	Message Authentication Code（メッセージ認証コード）
MD	Message Digest（メッセージダイジェスト）

MX	Mail Exchanger（メールエクスチェンジャ）
NIST	National Institute of Standards and Technology（米国国立標準技術研究所）
NS	Name Server（ネームサーバ）
NSEC	Next Secure
NTP	Network Time Protocol（ネットワークタイムプロトコル）
OMB	Office of Management and Budget（行政管理予算局）
OS	Operating System（オペレーティングシステム）
PKI	Public Key Infrastructure（公開鍵基盤）
PIR	Public Internet Registry
RFC	Request for Comment（インターネット技術に関する IETF 発行文書）
RP	Responsible Person（責任者）
RR	Resource Record（リソースレコード）
RRSIG	Resource Record Signature（リソースレコード署名）
SEP	Secure Entry Point（安全なエントリポイント）
SHA	Secure Hash Algorithm（安全なハッシュアルゴリズム）
SHS	Secure Hash Standard（安全なハッシュ標準）
SOA	Start of Authority（権威の起点）
TCP	Transmission Control Protocol（伝送制御プロトコル）
TLD	Top-Level Domain（トップレベルドメイン）
TSIG	Transaction Signature（トランザクション署名）
TTL	Time to Live（存続時間）
TXT	Text（テキスト）
UDP	User Datagram Protocol（ユーザデータグラムプロトコル）
ZSK	Zone Signing Key（ゾーン署名鍵）

付録C—参考文献

I. DNS – リソースレコードと一般的なアーキテクチャ

[1] D. Atkins and R. Austein, “Threat Analysis of the Domain Name System (DNS)”, RFC 3833, August 2004. http://www.ietf.org/rfc/rfc3833.txt
[2] R. Elz and R. Bush, “Clarifications to the DNS Specification”, RFC 2181, July 1997. http://www.ietf.org/rfc/rfc2181.txt
[3] D. Barr, “Common DNS Operational and Configuration Errors”, RFC 1912, February 1996. http://www.ietf.org/rfc/rfc1912.txt
[4] P. Mockapetris, “Domain Names - Concepts and Facilities”, STD 13, RFC 1034, November 1987. http://www.ietf.org/rfc/rfc1034.txt
[5] P. Mockapetris, “Domain Names - Implementation and Specification”, STD 13, RFC 1035, November 1987. http://www.ietf.org/rfc/rfc1035.txt

II. DNSSEC

[6] R. Arends, et al, “DNS Security Introduction and Requirements”, RFC 4033, March 2005. http://www.ietf.org/rfc/rfc4033.txt
[7] R. Arends, et al, “Resource Records for DNS Security Extensions”, RFC 4034, March 2005. http://www.ietf.org/rfc/rfc4034.txt
[8] R. Arends, et al, “Protocol Modifications for the DNS Security Extensions”, RFC 4035, March 2005. http://www.ietf.org/rfc/rfc4035.txt
[9] O. Kolkman and R. Gieben, “DNSSEC Operational Practices”, Internet-Draft draft-ietf-dnsop-dnssec-operations-08.txt (Work in Progress), March 2006. http://www.ietf.org/internet-drafts/draft-ietf-dnsop-dnssec-operational-practices-08.txt

III.動的更新

[10] B. Wellington, “Secure Domain Name System (DNS) Dynamic Update”, RFC 3007, November 2000. http://www.ietf.org/rfc/rfc3007.txt
[11] P. Vixie, et al, “Dynamic Updates in the Domain Name System (DNS UPDATE)”, RFC 2136, April 1997. http://www.ietf.org/rfc/rfc2136.txt

IV. TSIG

[12] D. Eastlake 3rd, “DNS Request and Transaction Signatures (SIG(0)s)”, RFC 2931, September 2000. http://www.ietf.org/rfc/rfc2931.txt
[13] P. Vixie, et al, “Secret Key Transaction Authentication for DNS (TSIG)”, RFC 2845, May 2000. http://www.ietf.org/rfc/rfc2845.txt
[14] D. Eastlake 3rd, “HMAC SHA TSIG Algorithm Identifiers”, Internet-Draft draft-ietf-dnsext-tsig-sha-06 (Work in Progress), January 2006. http://www.ietf.org/internet-drafts/draft-ietf-dnsext-tsig-sha-06.txt

V. 暗号規格

[15] H. Krawczyk, et al, “HMAC: Keyed-Hashing for Message Authentication”, RFC 2104, February 1997. http://www.ietf.org/rfc/rfc2104.txt.
[16] NIST, “The Keyed-Hash Message Authentication Code (HMAC)”, FIPS PUB 198, March 2002. http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf.
[17] NIST, “Secure Hash Standard”, FIPS PUB 180-2, August 2002. http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf.

VI.セキュリティの脆弱性

[18] CERT®/CC's Vulnerability Notes Database at *http://www.kb.cert.org/vuls/*
[19] NIST NVD metabase at http://nvd.nist.gov/
[20] BIND vulnerabilities page at *http://www.isc.org/products/BIND/bind-security.html*

付録D—索引

き

く

け

こ

さ

し

す

せ

そ

つ

て

と

な

に

ね

は

ふ

ほ

ま

む

め

ゆ

り

る

れ