DNS完全解説 の仕組み

最終回 DNSが担う新たな役割

馬場達也

前回は、これまでの復習として、ネームサーバを適切に運用するための設定例を紹介した。最終回となる今回は、近い将来にDNSが担うことになると思われる新たな役割について紹介する。

さらに重要性を増すDNSの役割

インターネットがARPANETと呼ばれていた時代には、SRI-NIC（SRI International Network Information Center）が管理していた「HOSTS.TXT」というファイルによって、ホスト名とIPアドレスを対応付けていた。このファイルは、現在のUnixシステムで使用されている「/etc/hosts」ファイルの基となったものであり、ホストの追加などがあった場合は、各サイトの管理者がSRI-NICにメールで通知してファイルを更新してもらい、ユーザーがその最新版を定期的にダウンロードすることで運用していた。しかし、ホストの数が増加するにつれて、この方式では対応することが難しくなったため、1984年に現在のような分散システムによるDNSが開発された。

HOSTS.TXTファイルでは、ホスト名とIPアドレスの対応や、OSの種類などのホスト情報を管理していたが、DNSでは、これらに加え、メール配送先の検索や、特定のサービスを提供しているホストの検索など、ホスト名とIPアドレスとの間の変換以外の機能が追加されてきた。現在も、DNSに新しい機能を追加するための議論がIETF（Internet Engineering Task Force）において行われており、今後もさらにDNSの役割が重要になってくると思われる。そこで、今回は、まだ議論されている段階であるが、近い将来にDNSに追加されると思われる新しい機能をいくつか紹介したい。なお、ここで紹介する機能はまだ議論段階のものであり、内容は変更される可能性があることをご了承いただきたい。

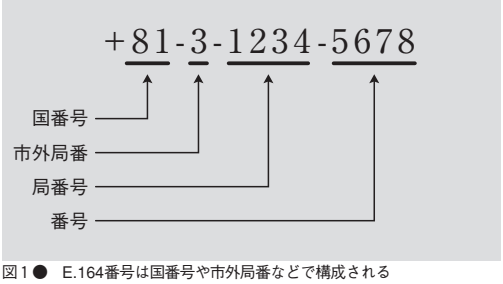


図1● E.164番号は国番号や市外局番などで構成される

電話番号をドメイン名に対応付けるENUM

最初に紹介するのは、ENUM（「イーナム」と読む）である。ENUMは、電話やFAXなどで使用されている番号を、DNSを使用してドメイン名に対応付けるための技術である。従来の電話やFAXは、回線交換である電話網を経由して接続されていたが、現在は、インターネットなどのIPネットワークを使用するIP電話が注目されてきている。IP電話は、従来の電話と同様の番号を持つが、インターネットを経由する場合は、電話番号のみでは接続することができないため、IETFのTelephone Number Mapping（ENUM）ワーキンググループでは、DNSを使用して、電話番号をURI（Uniform Resource Identifiers）に変換するENUMについて議論している。ENUMによって返却されるURIは、アクセス手段となるプロトコルとアクセス先のユーザー名およびホスト名からなっており、これを用いることで、インターネット経由でIP電話などの端末にアクセスすることができるようになる。

電話やFAXで使用されている番号の体系としては、ITU-TのE.164勧告で規定されている「E.164番号」がある。E.164番号は、国番号を含めた15けた以内の国際公衆電気通信番号であり、例えば、東京であれば日本の国番号である「81」を最初に付加し、市外局番の最初の「0」を除いた、図1のような番号となる。

```
$ORIGIN 8.7.6.5.4.3.2.1.3.1.8.e164.arpa.
     IN NAPTR 10 100 "u" "E2U+sip"         "!^.*$!sip:baba@example.com!"      .
     IN NAPTR 10 101 "u" "E2U+msg:mailto" "!^.*$!mailto:baba@example.com!"   .
```

リスト1● ENUMで使用するNAPTRリソースレコードの例

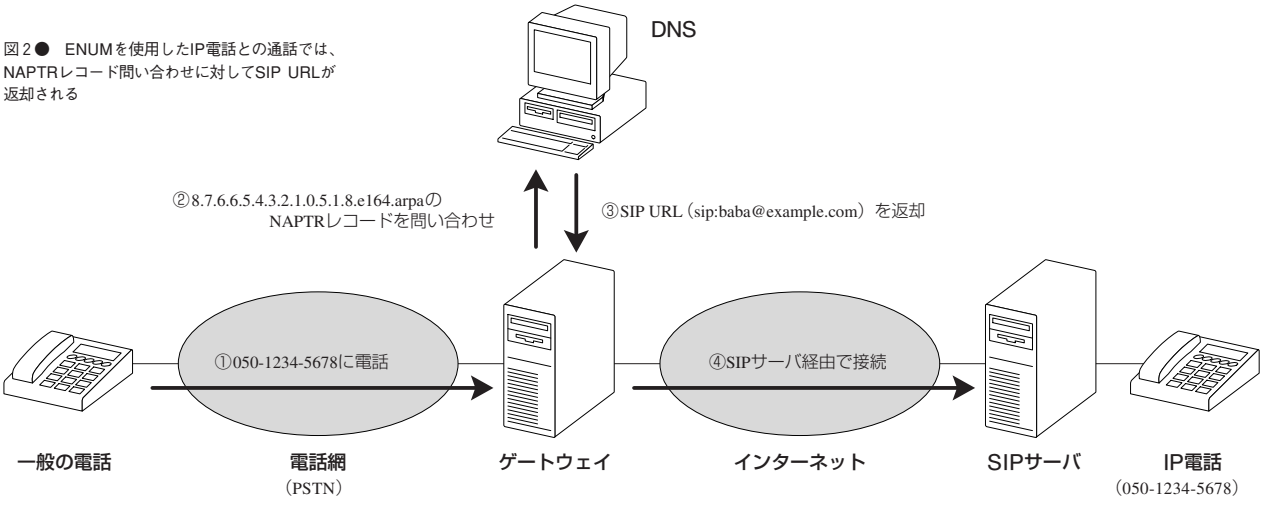


図2● ENUMを使用したIP電話との通話では、NAPTRレコード問い合わせに対してSIP URLが返却される

このE.164番号をDNSで扱えるように、ENUMでは次のような処理を行うことによって、E.164番号をドメイン名形式で表現する。

①E.164番号から「+」記号と数字以外の文字を除く。この結果をAUS（Application Unique String）と呼ぶ。
+81-3-1234-5678 → +81312345678

②AUSから最初の「+」記号を除く。
+81312345678 → 81312345678

③それぞれの数字の間に「.」を挿入する。
81312345678 → 8.1.3.1.2.3.4.5.6.7.8

④得られた文字列の順番を逆に並べ替える
8.1.3.1.2.3.4.5.6.7.8 → 8.7.6.5.4.3.2.1.3.1.8

⑤得られた文字列の最後に「.e164.arpa」を付加する。
8.7.6.5.4.3.2.1.3.1.8→8.7.6.5.4.3.2.1.3.1.8.e164.arpa

そして、ドメイン名形式に変換されたE.164番号とURIを対応付けるために、NAPTR（Naming Authority Pointer）リソースレコードを使用してリスト1のように記述する。

各フィールドの詳細な説明は省略するが、リスト1では、「+81-3-1234-5678」というE.164番号を、「sip:baba@example.com」および「mailto:baba@example.com」の2つのURIに対応付けている。これは、最初にSIP（Session Initiation Protocol）を使用して、「sip:baba@example.com」というSIP URLにアクセスし、もし、SIPによるアクセスに失敗した場合は、次に「baba@example.com」というアドレスに電子メールを送信することを意味している。このように、ENUMを使用すると、ある電話番号に対して、複数のアクセス手段を対応付けることができ、さらに、その優先度を相手に伝えることも可能となる。

それでは、ここで実際に通常の電話からIP電話に電話をかける場合の例を図2を用いて説明する。通常の電話から、IP電話用の電話番号に電話をかけると、最初にインターネットとの間に設置されているゲートウェイに接続される（①）。接続要求を受けたゲートウェイは、接続先の電話番号（E.164番号）をドメイン名に変換し、DNSに対してNAPTRリソースレコードを問い合わせる（②）。接続先のIP電話がSIPを使用している場合は、NAPTRリソースレコードから「sip:baba@example.com」のようなSIP URLが取得できるので（③）、SIPを使用して、インターネット経由で目的のIP電話に接続する（④）。

DNSによる公開鍵の配布

次に、DNSによる公開鍵の配布に関する取り組みについて紹介する。IPsec（IP Security Protocol）やSSH（Secure Shell）などのセキュリティプロトコルでは、主に公開鍵暗号を使用して相手の認証を行う。公開鍵を使用して相手の認証を行うためには、相手の正しい公開鍵を入手することが前提となるが、この役目を担っているPKI（Public Key Infrastructure）が十分に整備されていないという問題があった。そこで、このPKIと同等の機能を、すでに広く構築されているDNSで実現しようとする動きが出始めてきた。

PKIでは、公開鍵を認証局（CA）の秘密鍵で署名した公開鍵証明書を使用して、公開鍵の配布を行っている。DNSでは、DNSSEC（DNS Security Extensions）の仕組みによって、リソースレコードに署名を付加することができるようになったため、公開鍵をリソースレコード中に格納し、そのリソースレコードを上位のネームサーバによって署名されたゾーン鍵で署名することによって、PKIで使用する公開鍵証明書と同等の機能をDNSで実現することが可能となった。

現在、IETFでは、IPsecで使用する公開鍵と、SSHで使用する公開鍵のフィンガープリントをDNSで配布するための議論が行われている。ここでは、それぞれの取り組みの内容について紹介する。

IPSECKEYリソースレコードでIPsec用公開鍵を配布

IPsecでは、最初に、IKE（Internet Key Exchange）を使用して、IPsecで使用する暗号鍵などのセットアップを行う。IKEでは、正当な相手であることを相互に確認するために、事前共有秘密鍵やデジタル署名を使用して認証を行うが、事前共有秘密鍵による認証では、認証用の鍵をあらかじめ秘密に共有しておかなければならないという問題がある。これに対して、デジタル署名による認証の場合は、相手の公開鍵をオンラインで入手することで認証が行えるため、IPsecの相手認証方式の本命とされている。しかし、PKIが十分に整備されていない現在、正しい公開鍵をどのように入手するのかという問題がある。また、IPsecでは、セキュリティゲートウェイ（IPsecを実装したルータなどの機器）との間にIPsecトンネル（VPN）を構築することによって、そのトンネル経由でほかのホストにアクセスすることが可能となるが、目的のホストにアクセスするためには、どのセキュリティゲートウェイとの間にIPsecトンネルを構築すればよいのかがわからないという問題がある。

そこで、IETFのIPSECKEY WG（IPSEC KEYing information resource record Working Group）では、IPsec機器が、相手認証のために使用する公開鍵をDNSから取得できるようにするための新たなリソースレコードとして、IPSECKEY（IPsec Keying Information）リソースレコードを検討している。IPSECKEYリソースレコードは、あるホストがほかのホストとIPsecで通信する場合に使用するセキュリティゲートウェイのIPアドレスや公開鍵を格納する。本稿執筆時点で検討されているIPSECKEYリソースレコードのフォーマットは以下のとおりである。

<owner> <ttl> IN IPSECKEY <precedence> <algorithm> <gateway> <public key>

<owner>には、ホストのIPアドレスが「10.200.135.163.in-addr.arpa.」のような逆引き形式のドメイン名で記述される。<precedence>には優先度が入り、同じ<owner>に対してIPSECKEYリソースレコードが複数存在した場合は、この値が小さいものから評価される。<algorithm>には、公開鍵が使用する署名アルゴリズム（RSAやDSSなど）の番号が入る。そして、そのホスト自身が直接IPsecを使用した通信をせず、別のセキュリティゲートウェイが代わりにIPsecトンネルを構築して通信する場合には、<gateway>にセキュリティゲートウェイのホスト名またはIPアドレスがドメイン名形式で入る。セキュリティゲートウェイを使用しない場合には、<gateway>には「.」と記述される。また、<public key>には、セキュリティゲートウェイを指定した場合はセキュリティゲートウェイの公開鍵が記述され、セキュリティゲートウェイを指定していない場合は、<owner>の公開鍵が記述される。

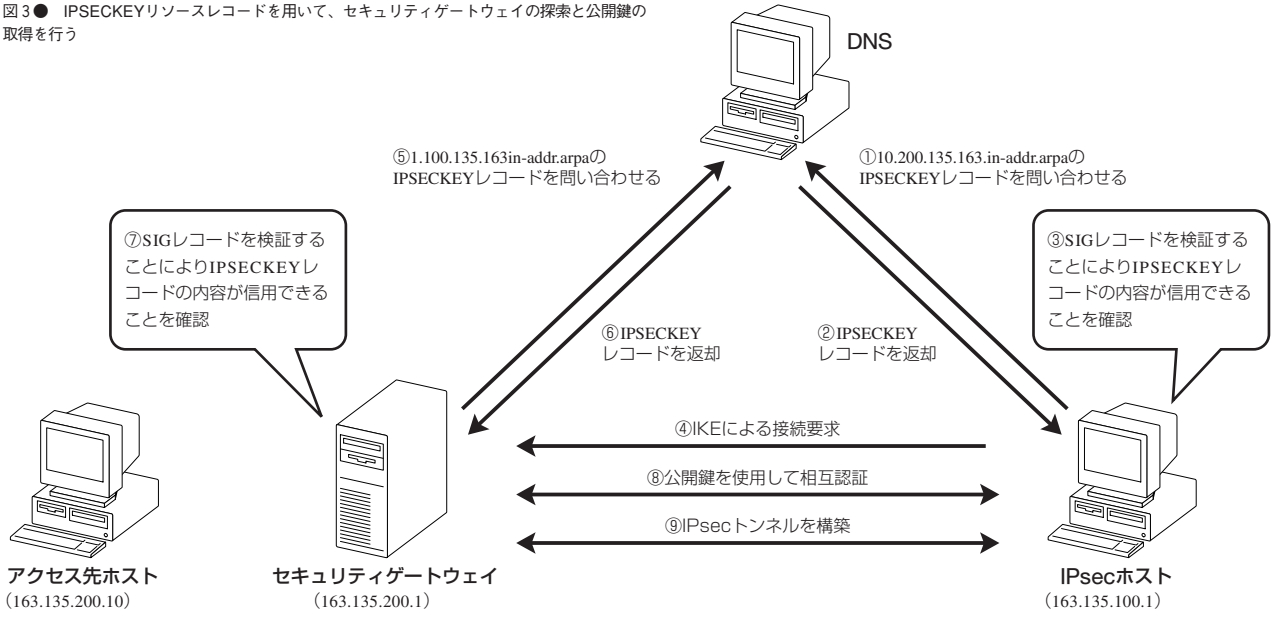
例えば、163.135.200.10というIPアドレスを持つホストにIPsecでアクセスする場合に、163.135.200.1というIPアドレスを持つセキュリティゲートウェイにIPsecトンネルを構築してからアクセスするようにするためには、リスト2のようにIPSECKEYリソースレコードを記述する。

それではIPSECKEYリソースレコードを使用した

リスト2● IPSECKEYリソースレコードの例

```
10.200.135.163.in-addr.arpa. IN IPSECKEY ( 10 1 1.200.135.163.in-addr.arpa.
                    AQOrXJxB56Q28iOO43Va36elIFFKc/QB2orIeL94BdC5X4idFQZjSpsZ
                    Th48wKVXUE9xjwUkwR4R4/+1vjNN7KFp9fcqa2OxgjsoGqCn+3OPR8La
                    9uyvZg0OBuSTj3qkbh/2HacAUJ7vqvjQ3W8Wj6sMXtTueR8NNcdSzJh1
                    49ch3zqfiXrxxna8+8UEDQaRR9KOPiSvXb2KjnuDan6hDKOT4qTZRRRC
                    MWwnNQ9zPIMNbLBp0rNcZ+ZGFg2ckWtWh5yhv1iXYLV2vmd9DB6d4Dv8
                    cW7scc3rPmDXpYR6APqPBRHlcbenfHCt+oCkEWse8OQhMM56KODIVQq3
                    fejrfi1H )
```

図3● IPSECKEYリソースレコードを用いて、セキュリティゲートウェイの探索と公開鍵の取得を行う



IPsec処理の流れを、図3を用いて説明する。あるホストに対して、IPsecを使用してアクセスする場合は、最初に接続先のIPアドレスに対するIPSECKEYリソースレコードをDNSに対して問い合わせる(①)。DNSからは、問い合わせたIPSECKEYリソースレコードと、それに対するSIGリソースレコードが返却されるので(②)、SIGリソースレコードに含まれる署名を検証して、入手したIPSECKEYリソースレコードの内容が正しいことを確認する(③)。そして、返却されたIPSECKEYリソースレコードに、セキュリティゲートウェイのIPアドレスとその公開鍵が含まれていれば、指定されたセキュリティゲートウェイに対して、IPsecトンネルを確立するためのIKEパケットを送信する(④)。IKEパケットを受信したセキュリティゲートウェイは、IPsecによる通信を確立しようとしてきたホストのIPアドレスに対するIPSECKEYリソースレコードをDNSに対して問い合わせる(⑤)。DNSからは、問い合わせたIPSECKEYリソースレコードと、それに対するSIGリソースレコードが返却されるので(⑥)、SIGリソースレコードに含まれる署名を検証して、入手したIPSECKEYリソースレコードの内容が正しいことを確認する(⑦)。これでお互いの公開鍵が取得できたので、取得した公開鍵を使用して相手の認証を行う(⑧)。お互いの認証が成功し、IKEによるセットアップが完了すると、セキュリティゲートウェイとの間にIPsecトンネルが構築されるので(⑨)、目的のホストに対してIPsecトンネル経由でアクセスする。

SSHFPリソースレコードによるSSH用フィンガープリントの配布

SSHは、ほかのホストに安全にリモートからログインしたり、安全にファイルを転送したりするためのプロトコルである。SSHでは、クライアントから接続し

```
$ ssh server.example.com
The authenticity of host 'server.example.com (192.168.0.3)' can't be established.

RSA key fingerprint is 15:d7:93:d8:73:f9:d0:34:11:31:6e:ad:4d:21:79:98.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'server.example.com' (RSA) to the list of known hosts.

baba@server.example.com's password:
```

サーバの公開鍵のフィンガープリント

示されたフィンガープリントがほかの方法で安全に取得したフィンガープリントと一致すれば公開鍵を信用して受け入れる

リスト3● OpenSSHを使用してアクセスした場合の例

```
server.example.com. IN SSHFP 1 1 123456789abcdef67890123456789abcdef67890
```

リスト4● SSHFPリソースレコードの例

ようとしているサーバが本当に意図する相手であるのかどうかを、サーバから送信された公開鍵のフィンガープリントをクライアントのユーザーが確認することで検証する。ユーザーは、サーバから送信された公開鍵のフィンガープリントと、ほかの信用できる方法で入手したサーバの公開鍵のフィンガープリントを比較する。両者が一致すれば、クライアントはサーバの正しい公開鍵を取得したことになる(リスト3)。しかし、このフィンガープリントの比較は、現在、リスト1のように、画面上に示されたフィンガープリントをユーザーが目で見て確認することによって行われている。セキュリティをよほど気にしているユーザーなら、このフィンガープリントとほかの信頼できる方法で取得したフィンガープリントとを比較してからその公開鍵を受け入れるかもしれないが、ほとんどのユーザーは、フィンガープリントの内容をまったく確認せずに公開鍵を受け入れてしまっているのが現状である。

そこで、IETFのSECSH WG(Secure Shell Working Group)では、SSHで使用する公開鍵のフィンガープリントをクライアントがDNSから自動的に取得できるようにするための新たなリソースレコードとしてSSHFP(Secure Shell Fingerprint)リソースレコードを検討している。本稿執筆時点で検討されているSSHFPリソースレコードのフォーマットは次のとおりである。

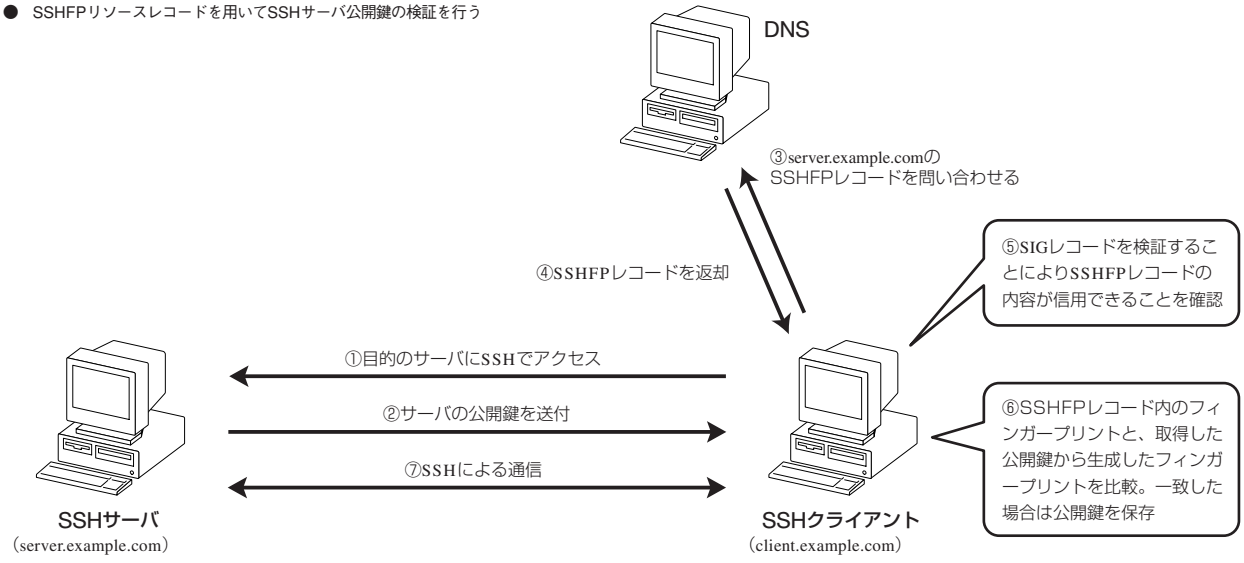
<owner> <ttl> IN SSHFP <algorithm> <fingerprint type> <fingerprint>

<owner>には、サーバのホスト名が入り、<algorithm>には、サーバの公開鍵が使用する署名アルゴリズム(RSAやDSSなど)の番号が入る。そして、<fingerprint type>には、公開鍵からフィンガープリントを生成する際に使用したハッシュアルゴリズム(SHA-1やMD5など)の番号が入り、<fingerprint>には、サーバの公開鍵のフィンガープリントが16進数の文字列で入る。

例えば、「server.example.com」のRSAの公開鍵(アルゴリズム番号:1)からSHA-1により生成したフィンガープリント(フィンガープリントタイプ:1)を含むSSHFPリソースレコードはリスト4のようになる。

それでは、SSHFPリソースレコードを使用したSSH接続の処理の流れを、図4を用いて説明しよう。クライアントがサーバにSSHでアクセスすると(①)、サーバは自身の公開鍵をクライアントに送信する(②)。クライアントは、DNSに対して、サーバの公開鍵のフィンガープリントが格納されたSSHFPリソースレコードを問い合わせる(③)。DNSからは、問い合わせたSSHFPリソースレコードと、それに対するSIGリソースレコードが返却されるので(④)、SIGリソースレコードに含まれる署名を検証して、入手したSSHFPリソースレコードの内容が正しいことを確認する(⑤)。そして、DNSから取得したSSHFPリソースレコードに格納されているフィンガープリントと、サーバが送信してきた公開鍵から生成したフィンガープリントとを比較し、両者が一致すれば、サーバが送信してきた公開鍵は正当なものであるとして、その公開鍵をクライアントの鍵ファイルに保存し(⑥)、SSHによる通信を行う(⑦)。もし、両者が一致しなければ、サーバは

図3● SSHFPリソースレコードを用いてSSHサーバ公開鍵の検証を行う



偽の公開鍵を送信してきたことになり、サーバが何者かに成り済まされている可能性があるため、接続を拒否する。

DNSの最新の動向を知るための情報源

以上、今回は、DNSが担う新たな役割として、ENUMとDNSによるIPsecの公開鍵の配布、そしてDNSによるSSHのフィンガープリントの配布について紹介してきた。DNSの最新の動向をさらに知りたい場合は、IETFでの議論を調査するのが最もよい方法である。以下に、DNSに関する議論を行っているIETFのワーキンググループの一覧をリストアップしたので、DNSに関する最新の情報を知りたい場合は、これらのURLにアクセスしてみてほしい。

●DNSSECやIPv6対応の動向など

DNSEXT WG(DNS Extensions Working Group)
http://www.ietf.org/html.charters/dnsext-charter.html

●DNSの運用に関する議論など

DNSOP WG（Domain Name System Operations Working Group）
http://www.ietf.org/html.charters/dnsop-charter.html

●ENUMに関する動向

ENUM WG（Telephone Number Mapping Working Group）
http://www.ietf.org/html.charters/enum-charter.html

●IPsecで使用する公開鍵の配布に関する動向

IPSECKEY WG（IPSEC KEYing information resource record Working Group）
http://www.ietf.org/html.charters/ipseckey-charter.html

●SSHで使用する公開鍵のフィンガープリントの配布に関する動向

SECSH WG（Secure Shell Working Group）
http://www.ietf.org/html.charters/secsh-charter.html

さて、12回続いたこの連載も今回で終了であるが、次号からは、DNSを構築する際に直面するトラブルについてのQ&Aの企画を予定しているので、今後はそちらもぜひご覧いただければと思う。

NTTデータ 馬場達也

●今回の内容に関連するRFC

RFC 2916 "E.164 number and DNS"