Engineer

IPv6

128ビットのアドレスを使うInternet Protocolのバージョン

どのようなものか

IPv6は、Internet Protocol version 6の略で、RFC 791が記述するIPv4の後継として設計されたInternet Protocolのバージョンです。現在の仕様は、RFC 2460を置き換えた2017年7月のRFC 8200で、インターネット標準（Internet Standard、STD 86）です。最も目立つ違いは、アドレスの大きさです。IPv4の32ビットに対して128ビットあります。IPv4全体には4,294,967,296個のアドレスがありますが、標準的な大きさのIPv6サブネット一つ、/64は、その数を二乗した個数を収容します。

パケットのほかの部分も変わりました。RFC 8200は、処理の負担を減らすため、一部のIPv4ヘッダーフィールドを除いたり任意にしたりしています。オプション情報は、主ヘッダーの後に置く別の拡張ヘッダーに格納し、生存時間（Time to Live）をホップ制限（Hop Limit）に改名しました。転送する各ノードが、その値を一つ減らします。ヘッダーのチェックサムはありません。ルーターはパケットを分割せず、IPv6の断片化は送信元だけが行います。各リンクは、少なくとも1280オクテットのパケットを運べる必要があり、送信側には、経路MTU探索（Path MTU Discovery）で経路が運べる最大のパケットサイズを調べることを、強く勧めています。

アドレスは、16ビットずつの十六進数を八組、コロンで区切って書きます。RFC 4291は、各組の先頭のゼロを省き、すべてゼロの組が続く一か所を二重コロンに置き換えることを認めています。二重コロンは一回だけ使えます。そのため、2001:db8:0:0:8:800:200c:417aは、2001:db8::8:800:200c:417aとも書けます。同じアドレスを何通りにも書けるため、2010年8月のRFC 5952が、一つの形を選んでいます。小文字を使い、先頭のゼロを除き、最も長くゼロの組が続く箇所に二重コロンを使います。同じ長さなら最初の箇所を選び、一組だけには使いません。ループバックやリンクローカルなどの特別なアドレスを除き、この記事で例に使うアドレスとプレフィックスはすべて、文書用範囲から取っています。RFC 3849が確保した2001:db8::/32と、IPv4ではRFC 5737が予約したブロックです。

RFC 4291は、アドレスの割り当て先も変えています。アドレスはマシンではなくインターフェースに属し、一つのインターフェースが同時にいくつも持てます。各インターフェースには、そのリンク内だけで有効な、fe80::/10配下のリンクローカルアドレスが必要です。ループバックアドレスは::1です。ブロードキャストはなく、その役割はマルチキャストが担います。ff02::1は、リンク上の全ノードに届きます。RFC 4193が定義するfc00::/7配下のユニークローカルアドレスは、サイト内部で使います。グローバルユニキャストアドレスは、ローカルなサイトの外との通信に使います。ほとんどのアドレスでは、末尾64ビットがインターフェース識別子です。そのため、通常のネットワーク区間は/64になります。

ホストは、通常、自分でアドレスを設定します。RFC 4862のステートレスアドレス自動設定では、ルーターが、定期的なルーター広告（Router Advertisement）でリンクのプレフィックスを知らせます。各ホストは、プレフィックスと自分のインターフェース識別子を組み合わせ、使う前に重複アドレス検出を行います。近隣要請（Neighbor Solicitation）メッセージを送り、同じアドレスを持つ別のノードがないか確認します。厳密な割り当てをより細かく管理したいサイトでは、DHCPv6が代替手段になり、両方を併用することもできます。

概念を判断に生かす

「IPv6とは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 Wi-Fi、ファイアウォール、リモートアクセスを整え、どこからでも働け、権限のある人だけが接続できるようにします。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 Ubuntu Linux VPSインスタンス30台を管理 — 30 復旧計画とともに管理したLinuxサーバー

Ubuntu Linux VPSインスタンス30台を管理 お問い合わせ →

なぜ重要なのか

多くのチームにとって、IPv6は、自分で開始を決めるプロジェクトではありません。RFC 4862の要点は、人がホストにアドレスを入力する必要がないことです。ルーターがプレフィックスを広告するリンクにサーバーを置けば、グローバルIPv6アドレスを自分で得られます。IPv4だけを想定した許可リスト、ファイアウォール規則、ログのパーサー、ヘルスチェックには、新しいアドレス空間の分だけ、見えない範囲ができます。

この死角は、見落としやすいものです。二つのバージョンは、通信上の形式を共有しない別々のプロトコルだからです。パケットフィルターは、ヘッダーを別々に照合します。IPv4アドレスをキーにするレート制限には、IPv6のクライアント

が見えず、127.0.0.1にだけ接続するテストからは、::1について何も分かりません。一方のバージョンを厳重に制限しても、もう一方が開いたままのホストには、実質的にファイアウォールがありません。

IPv6は、IPv4以上に制御メッセージに依存します。近隣探索、ルーター広告、経路MTU探索は、すべてICMPv6で運びます。そのため、境界でICMPを捨てるというIPv4からの習慣は、IPv6を壊します。接続がデータを運び始めてから、初めて症状が出ることもあります。また、一つのアドレスに多くの有効な表記があるため、文字列で比較するコードは、一貫しない判断をします。

実際の活用例

Python標準の `ipaddress` モジュールは、有効な表記をすべて解析し、先頭のゼロを省き、最長のゼロの連続を縮めた短い形を出力します。プレフィックスを計画し、二つの表記が同じアドレスか確認する、手軽な方法です。以下のセッションは、文書用プレフィックスのアドレスを使い、Python 3.15で実行した、この記事の説明用の例です。

```
>>> import ipaddress
>>> site = ipaddress.ip_network("2001:db8:42::/48")
>>> site.num_addresses == 2**80
True
>>> lans = site.subnets(new_prefix=64)
>>> next(lans), next(lans)
(IPv6Network('2001:db8:42::/64'), IPv6Network('2001:db8:42:1::/64'))
>>> len(list(site.subnets(new_prefix=64)))
65536
>>> a = ipaddress.ip_address("2001:0DB8:0000:0000:0001:0000:0000:0001")
>>> a
IPv6Address('2001:db8::1:0:0:1')
>>> a.exploded
'2001:0db8:0000:0000:0001:0000:0000:0001'
>>> a == ipaddress.ip_address("2001:db8:0:0:1::1")
True
>>> ipaddress.ip_address("fe80::1%eth0").scope_id
'eth0'
```

/48では80ビット分を割り当てに使え、/64の区間へ分割すると、その区間を65,536個作れます。区間の番号になるのは第四の組です。大文字ですべてのゼロを明記したアドレスは、短縮形の2001:db8::1:0:0:1で返ります。二重コロンは、同じ長さの二つのゼロの連続のうち、最初を省略します。別の有効な表記である

2001:db8:0:0:1::1と比較しても、同じ128ビットなので等しくなります。最後の行は、ゾーンを伴うリンクローカルアドレスを示します。`%eth0`が、どのリンクに属するかを表します。

例の後半は、nftablesを動かすLinuxホストで、オフィスからのSSHを両バージョンで許可する規則です。nft(8)マニュアルは、`inet`を、IPv4とIPv6のパケットをともに扱う混合ファミリーと説明しています。`ip saddr`はIPv4ヘッダーを読み、`ip6 saddr`はIPv6ヘッダーを読みます。

```
add table inet filter
add chain inet filter input { type filter hook input priority filter; }
add rule inet filter input ip saddr 198.51.100.0/24 tcp dport 22 accept
add rule inet filter input ip6 saddr 2001:db8:42::/48 tcp dport 22 accept
add rule inet filter input tcp dport 22 drop
```

第四行がなければ、オフィスのIPv6プレフィックスから接続する管理者は、どちらの許可規則にも一致せず、遮断されます。この場合は、少なくとも失敗がはっきり分かります。気付きにくいのは、その逆です。IPv6アドレスも持つホストで、`ip`ファミリーだけに規則を作ると、IPv6パケットはまったく検査しません。

2026年9月の実績ページ、第二のサーバーをコードから構築は、同じような外部公開による危険を、構築の順序から扱います。最初のマシンは手で作成し、後からファイアウォールを付けたため、新しいホストに既定のパスワードポリシーが適用されたまま、インターネットから接続できる時間がありました。第二のマシンは、ファイアウォールを作ってから作成します。ファイアウォールのplayをサーバーのplayより先に実行し、両プロバイダーのファイアウォールを、それぞれのREST APIに直接アクセスして設定します。同ページはIPv6に触れていません。したがって、上の規則は、この記事の説明用で、そのホストの設定ではありません。この記事が付け加えるのは、ホストが存在する前に、そのホストが持つすべてのアドレスを、IPv6も含めてファイアウォールで保護する必要があるという点です。

問題が起きるところ

ファイアウォールがIPv4しか扱いません。 `ip`ファミリーの規則や、IPv4プレフィックスだけの許可リストでは、ホストの各IPv6アドレスへの通信の扱いが、その先の仕組み任せになります。`inet`ファミリーで規則を書くか、`ip`と`ip6`の表を同期させます。すべての`ip saddr`規則に対応する`ip6 saddr`規則を用意し、各サービスを、IPv4だけでなくIPv6アドレスでも接続して試します。

ICMPv6を一律に捨てます。 ファイアウォールでのフィルタリングを扱うRFC 4890は、「ICMPv6は、IPv6の動作に不可欠である」と明言しています。次のリンクには大きすぎて転送できないパケットを、ルーターは捨て、パケット過大（Packet Too Big）メッセージを返します。RFC 8201は、フィルタリングがそのメッセージを止めると、送信元が経路MTUを知ることができないと指摘します。その結果が、同文書のいうブラックホール接続です。TCPハンドシェイクは成功するのに、その後の転送が止まります。RFC 4890は、IPv6ネットワークに参加するノードが捨ててはならないメッセージとして、近隣要請（Neighbor Solicitation）、近隣広告（Neighbor Advertisement）、ルーター要請（Router Solicitation）、ルーター広告（Router Advertisement）も挙げています。より広い範囲を遮断する前に、nftが `packet-too-big` 、 `nd-neighbor-solicit` 、 `nd-neighbor-advert` 、 `nd-router-solicit` 、 `nd-router-advert` と呼ぶ種別を受け入れます。

アドレスをテキストとして比較します。 RFC 5952は、表計算の検索で2001:db8:0:0:1::1を探し、2001:db8::1:0:0:1と書いた項目を見落とす例を示します。新しいアドレスを割り当てる前に、既に使われていないか一覧を検索する場合、問題になります。比較前にアドレスを解析し、Pythonの `ipaddress` で解析したアドレスを `str()` に渡して得られる短縮形など、一つの正規形で保存、表示します。

アドレスとポートを続けて書きます。 2001:db8::1:80は、一つのアドレスとも、アドレスにポート80を続けたものとも読めます。RFC 5952は、この形を曖昧とし、URLと同様、[2001:db8::1]:80という角括弧を勧めています。必ず括弧を使い、コードでは、最後のコロンではなく、閉じる括弧を基準にホストとポートを分けます。

リンクのプレフィックスが/64ではありません。 自動設定では、プレフィックスとインターフェース識別子を合わせて128ビットになる必要があります。64ビットの識別子では、RFC 4862は、それ以外の長さで広告されたプレフィックスをホストに無視させます。そのため、ルーター広告が/72や/56を伝えるリンクでは、ホストはそのプレフィックスからアドレスを作りません。各リンクを/64として計画し、より大きな割り当ては、上のセッションと同様に/64へ分割します。

リンクローカルアドレスをゾーンなしで使います。 各インターフェースはfe80::/10内のアドレスを持ち、RFC 4291は、その有効範囲を一つのリンクに限っています。そのため、複数のインターフェースを持つホストでは、アドレスだけでは、どのリンクか分かりません。Pythonの文書は、ゾーンだけが異なる二

つのアドレスも等しくないと述べています。アドレスとともにゾーンを保つか（fe80::1%eth0）、サービスにグローバルアドレスやユニークローカルアドレスを与え、リンクローカルは、そのリンク内の用途に使います。

参考資料

- RFC 8200：Internet Protocolバージョン6（IPv6）の仕様 — 現在のIPv6標準です。ヘッダー形式、拡張ヘッダー、断片化とパケットサイズの規則を扱います。
- RFC 4291：IPバージョン6のアドレス体系 — アドレス種別、テキスト表記、リンクローカルとマルチキャストのアドレス、64ビットのインターフェース識別子です。
- RFC 5952：IPv6アドレスのテキスト表記に関する推奨事項 — アドレスの正規表記と、多様な書き方が起こす問題です。
- RFC 4862：IPv6のステートレスアドレス自動設定 — 広告されたプレフィックスから、ホストがアドレスを作り、重複を調べる方法です。
- RFC 4890：ファイアウォールでのICMPv6メッセージのフィルタリングに関する推奨事項 — ファイアウォールが通す必要のあるICMPv6メッセージと、その理由です。
- ipaddress：IPv4/IPv6操作ライブラリー — IPv6のアドレスとネットワークの解析、比較、分割に関する、Pythonのリファレンスです。

用語集

実際の制約に合わせて選ぶ

デプロイの流れ、運用上のリスク、緊急対応の担当者をお知らせください。検査、復旧手段、明確な責任の所在を伴う変更の範囲を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「IPv6とは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/ipv6/