DNSの仕組みの基本と
現場のトラブルシューティングにおける二つのポイント

2024年1月17日

JANOG53 DNSチュートリアル

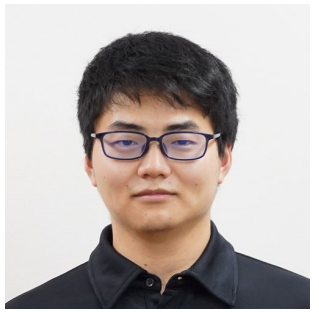
株式会社日本レジストリサービス（JPRS）

森下 泰宏・月東 健人

講師自己紹介

- **森下 泰宏（もりした やすひろ）**
  - 所属：JPRS 技術広報担当・技術研修センター
  - 2001年にJPNICから転籍
  - 主な業務内容：技術広報活動全般・社内外の人材育成

- **月東 健人（がっとう けんと）**
  - 所属：JPRS システム部
  - 2021年に新卒入社
  - 主な業務内容：JP DNSのインフラ構築・サーバー運用全般



会社概要（JPRS）

株式会社 日本レジストリサービス

JaPan Registry Services

■主な役割

- JPドメイン名（.jp）の登録管理
- JP DNSサーバーの運用、Mルートサーバーの共同運用
- インターネットのポリシー策定や技術の標準化など、国際活動・研究開発への貢献

このチュートリアルの構成

- 以下の4パートで構成
  1. **DNSの構成要素と分散管理の仕組み**
  2. **構成要素の役割と名前解決の仕組み**
  3. **現状を踏まえた名前解決の流れ**
  4. **現場のトラブルシューティングにおける二つのポイント**

前半2パートを月東が、後半2パートを森下が担当します

想定する参加者

- 特に、こんな方に聞いてほしい！
  - 新入社員や新しい担当者に、**DNSの仕組みを教えたい**
  - 担当者になったので、**DNSの仕組みを改めて勉強したい**
  - 現在のインターネットにおける、**名前解決の流れを知りたい**
  - 現場のトラブルシューティングに役立つ、**二つのポイントを知りたい**

宣伝：DNSがよくわかる教科書

- 2018年の発売後、継続してご好評をいただいており、2023年4月に**7刷**となりました！[*1]
  - 増刷時に細部の修正・コラムの追加などを実施しています[*2]
- 当社の**新人研修**でも使用しています！
  - 森下が毎年、技術系新入社員に講義しています
  - 月東も2021年に講義を受けました

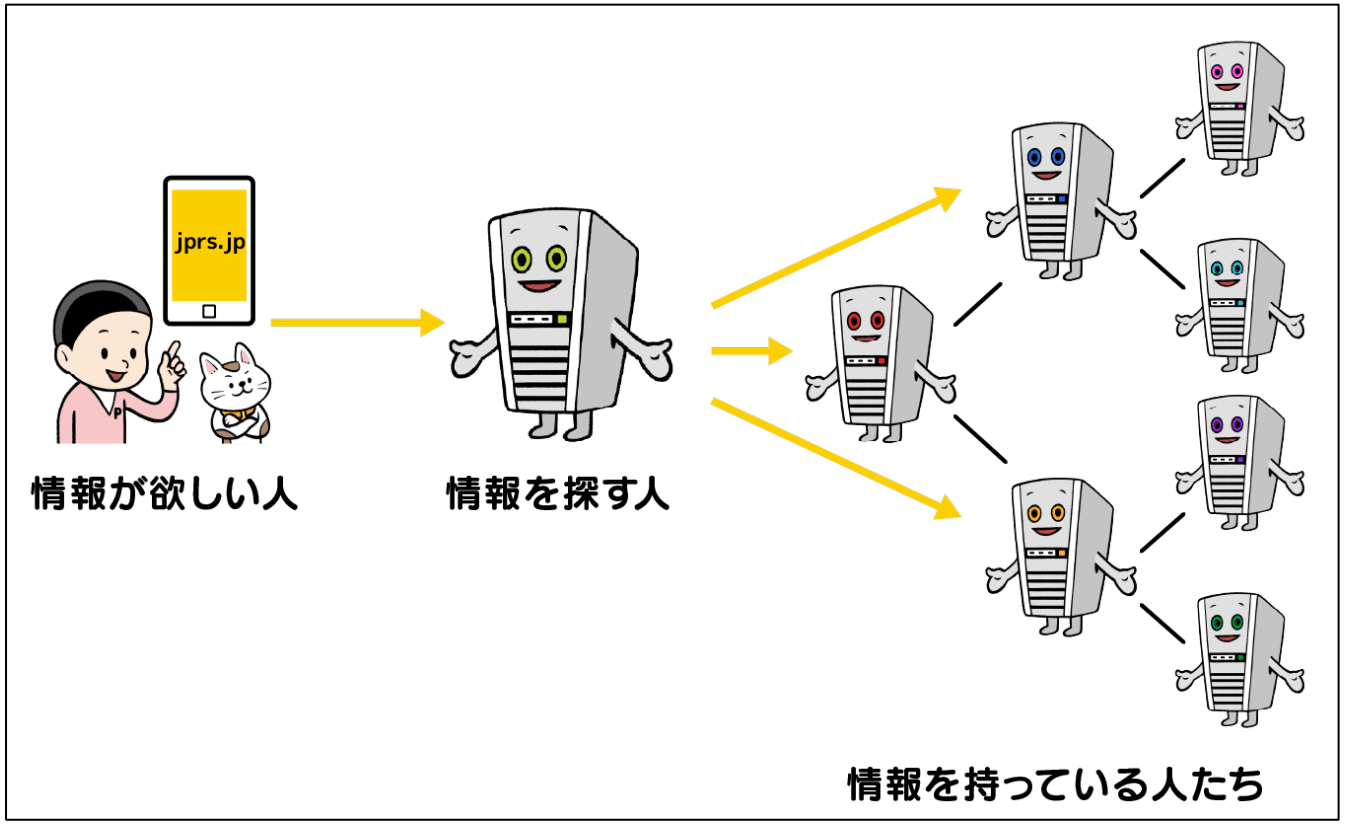
[*1] DNSがよくわかる教科書 | SBクリエイティブ <https://www.sbcr.jp/product/4797394481/>

[*2] DNS関連のRFC - JPRS DNS 関連技術情報 <https://jprs.jp/tech/index.html#dns-rfc-info>

1. DNSの構成要素と分散管理の仕組み

3種類の構成要素

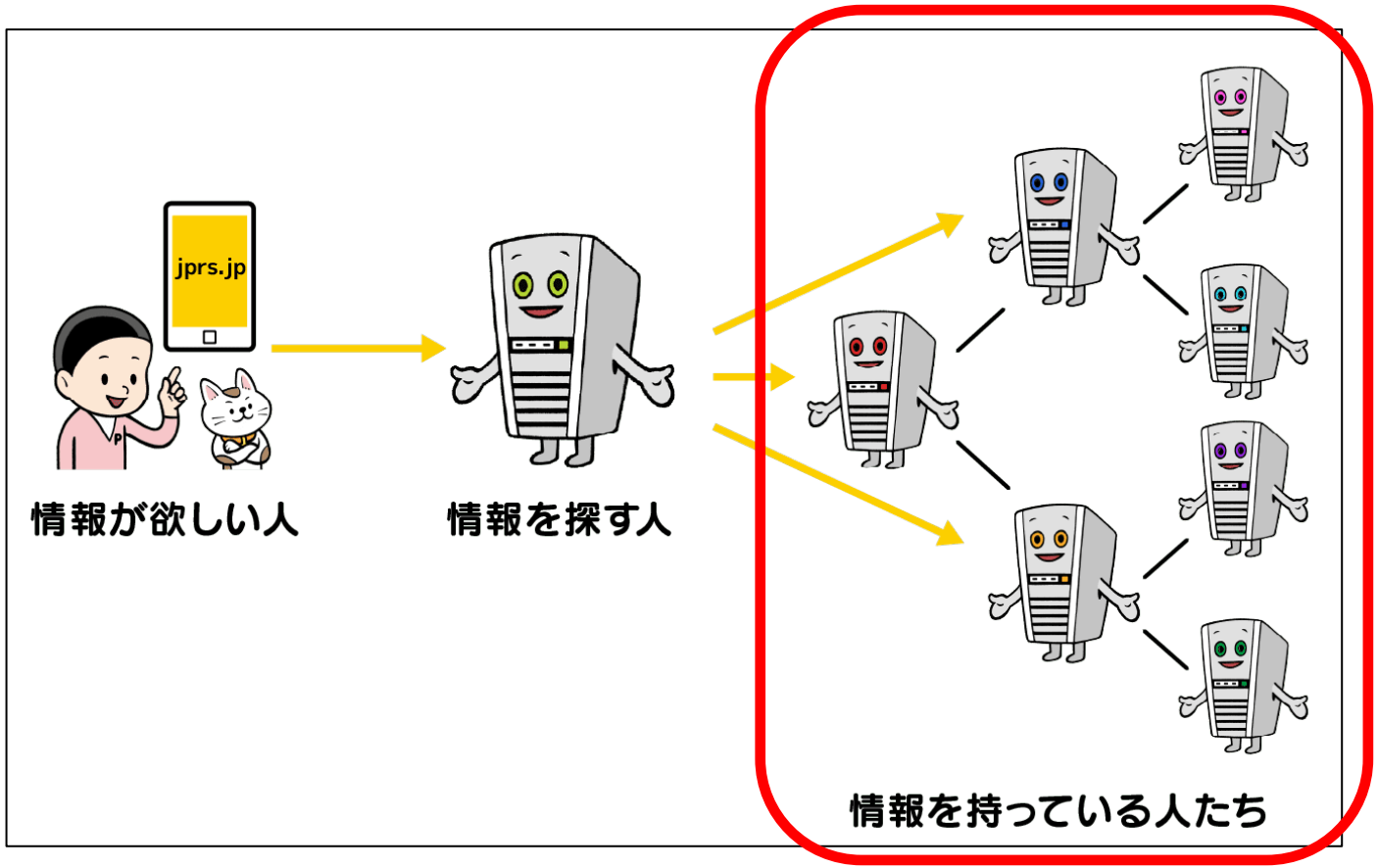
- DNSには、**3種類の登場人物（構成要素）**が存在する
  - **① 情報が欲しい人**
  - **② 情報を探す人**
  - **③ 情報を持っている人たち**
- DNSは、これらの構成要素が互いに連携し合って動作している



画像引用元：<https://withponta.jp/>

情報を持っている「人たち」

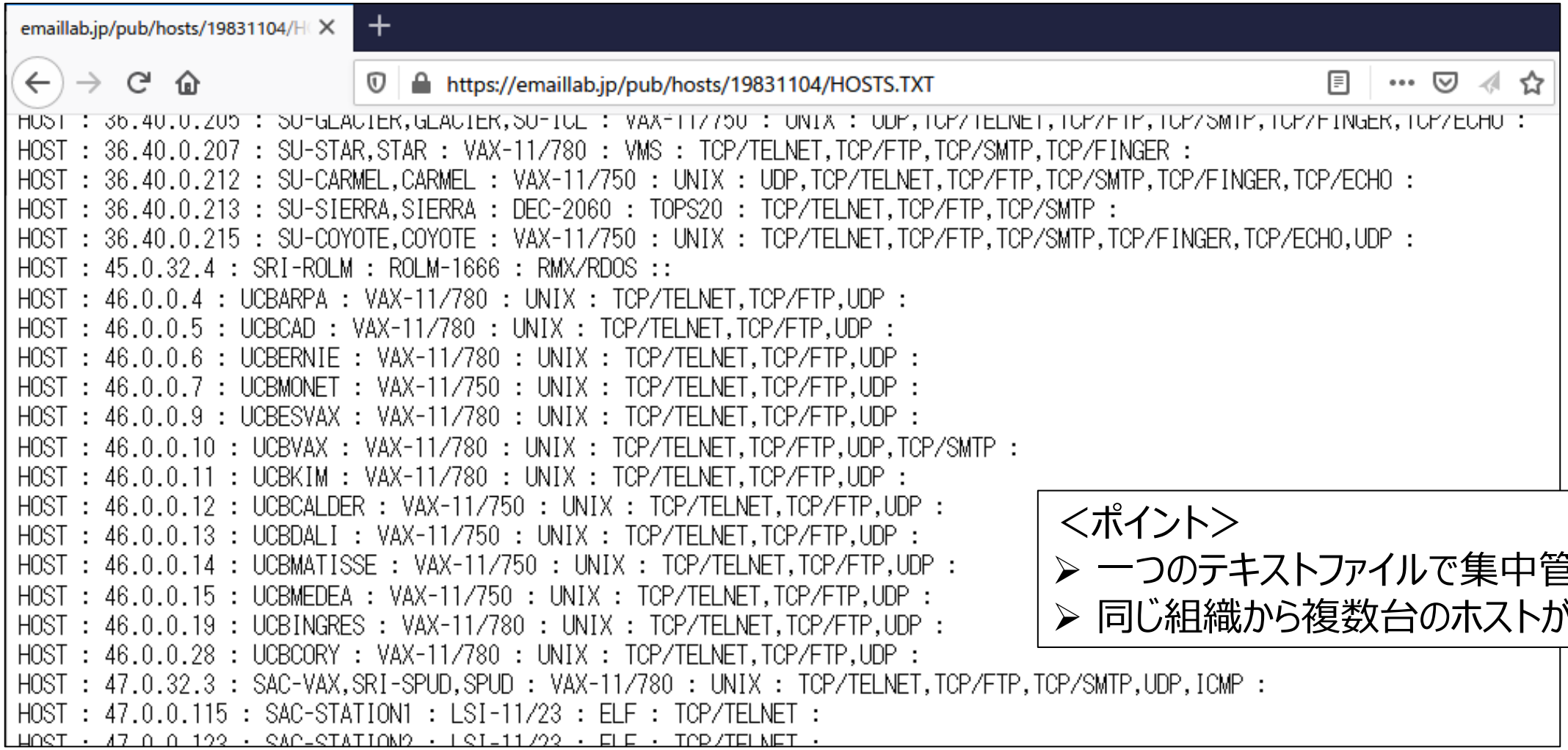
- なぜ、情報を持っている「**人たち**」なのか？



DNS以前は「情報を持っている人」だった

- DNSが導入されるまで、対応付けは**集中管理**されていた
  - 米国のSRI-NICという組織が「持っている**人**」を担当していた
  - SRI-NICはすべての対応付けが書かれた表（**HOSTS.TXT**）を、オンラインで公開していた

1983年11月4日のHOSTS.TXT（抜粋）



＜ポイント＞
- 一つのテキストファイルで集中管理
- 同じ組織から複数台のホストが登録

引用元：<https://emaillab.jp/dns/hosts/>

HOSTS.TXTによる名前の一元管理

- みんながこのHOSTS.TXTを使うようにすることで、**名前の一元管理**が可能になった
  - インターネットの**すべての利用者が同じ名前を使える**
  - すべての名前が**インターネット全体で同じ意味を持つ**

HOSTS.TXTによる集中管理の限界

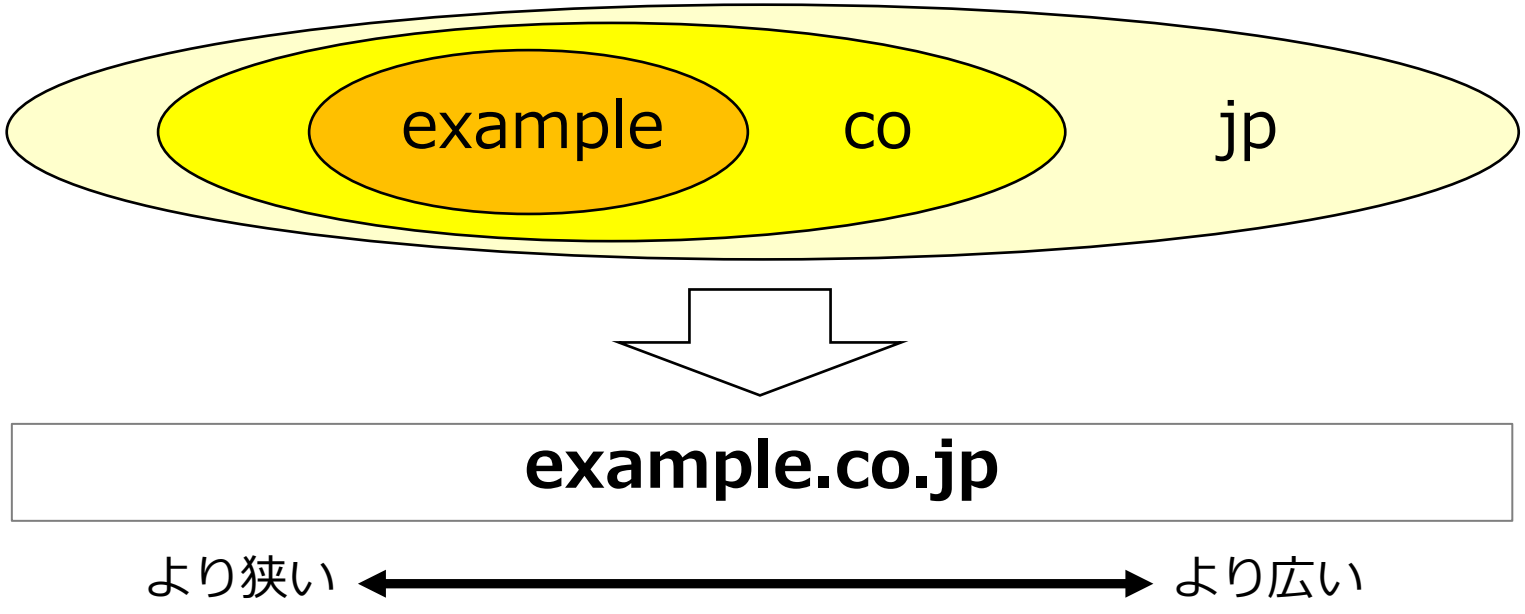
- 当時、HOSTS.TXTは以下のように管理されていた
  ① 各組織がホストを接続する時、SRI-NICに**登録申請**する
  ② SRI-NICは各組織からの申請内容を、HOSTS.TXTに**反映**する
  ③ 各組織のホストは、HOSTS.TXTをオンラインで**入手・利用**する
- インターネット自身の成長により、この管理方法ではHOSTS.TXTのファイルサイズと更新頻度が、**管理の限界を迎える**ことは明らかであった

限界を迎えないようにするための仕組み

- そのため、DNSには**情報を手分けして持ち**、かつ**HOSTS.TXTと同様の一元管理を実現する**ための、**分散管理の仕組み**が導入されることになった
- 以降では、そのために導入された以下の仕組みについて解説する
  - **ドメイン名**
  - **階層化**
  - **委任**
  - **ゾーン**

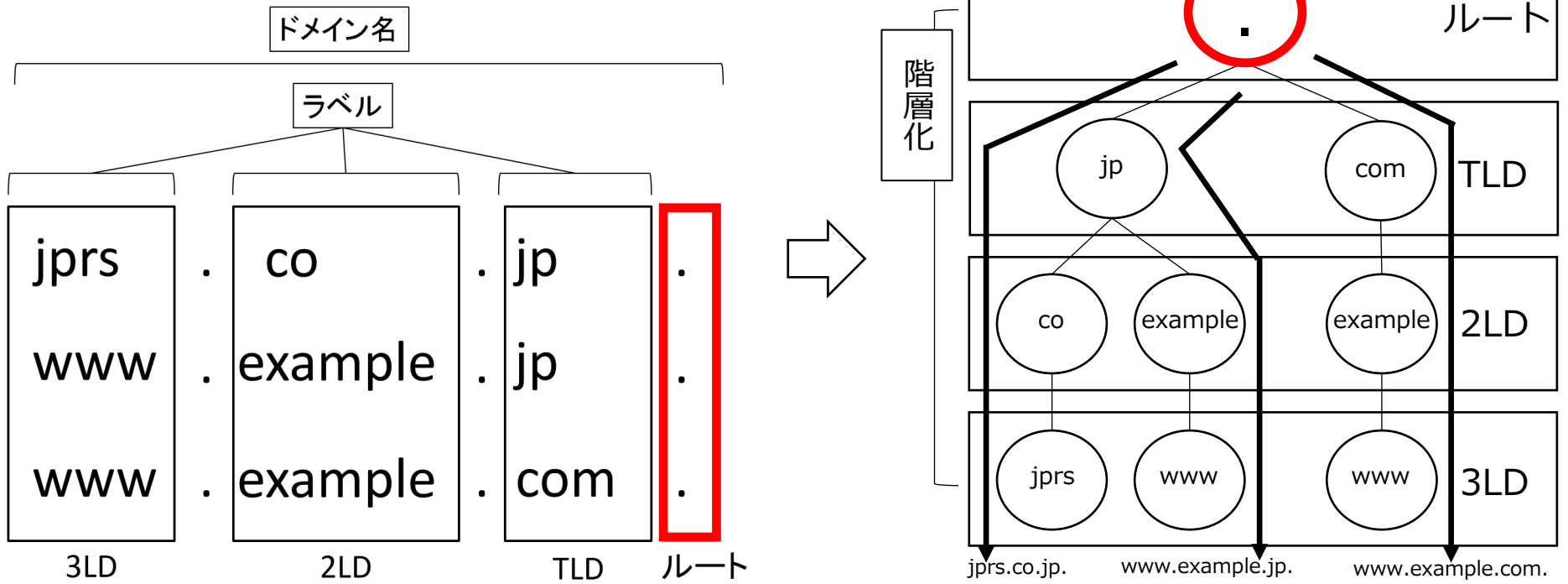
ドメイン名

- 管理する範囲（**ドメイン**）を、名前で表せるようにした
  - 部分的な名前（**ラベル**）を、**ドット（.）でつないだ形**で表す
  - **左側のラベルほど、範囲が狭くなる**ように設計された



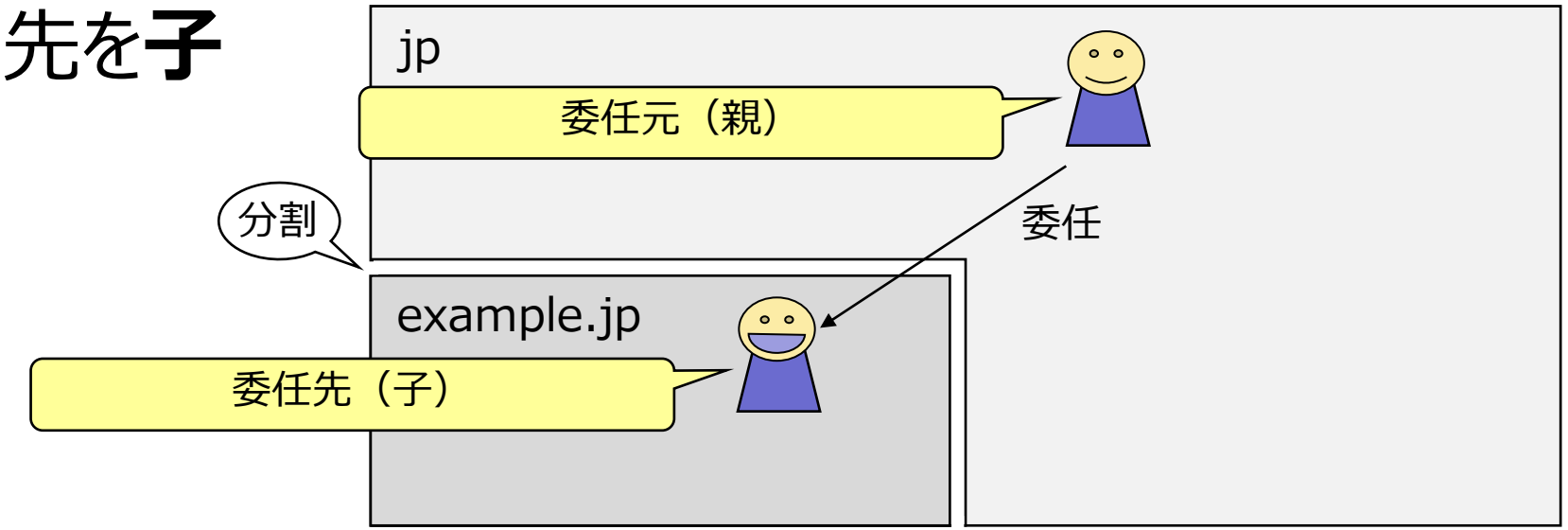
階層化

- **一つのルートを起点とする形**で、ドメイン名を**階層化**した
  - HOSTS.TXTと同様の、一つの**名前空間**を使う**一元管理**が可能になった



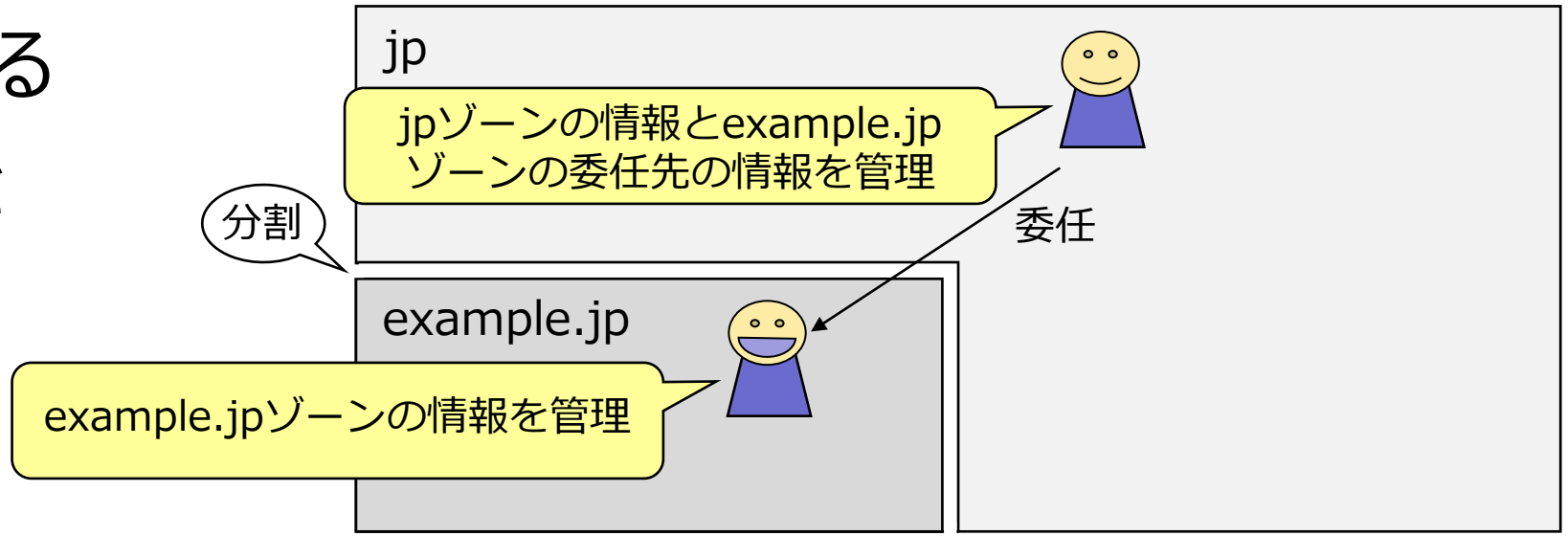
委任

- **階層を分割**して、**管理を任せられる**ようにした
  - 下記の例では、委任元（jp）は委任先（example.jp）の情報のみを管理し、example.jpの実際の管理は委任先の管理者が担当する
- 委任元を**親**、委任先を**子**と呼ぶ



ゾーン

- 「情報を持っている人たち」は、以下の情報**のみ**を管理する
  - **自分が管理するドメイン名**の情報
  - **自分が委任したドメイン名の委任先**の情報
- 委任によって作られる管理の単位のことを「**ゾーン**」と呼ぶ



DNSの分散管理と一元管理

- **ドメイン名**を導入して名前を**階層化**し、**委任**によって管理の単位を**ゾーン**に**分割**することで、**分散管理**を実現できる
- **一つのルート**を**階層構造の起点**とすることで、HOSTS.TXTと同様の**一元管理**が可能になる

ドメイン名とDNSで、限界を超えないようにするための**分散管理**と、
すべての名前がインターネット全体で同じ意味を持つ
「**名前空間の一意性**」の**双方を実現できる**

まとめ：DNSの構成要素と分散管理の仕組み

- DNSには**3種類の構成要素**があり、連携し合って動いている

情報が欲しい人・情報を探す人・情報を持っている人たち

- 限界を超えないように**情報を手分けして持ちつつ、一元管理を実現する**ための**分散管理の仕組み**が導入されている

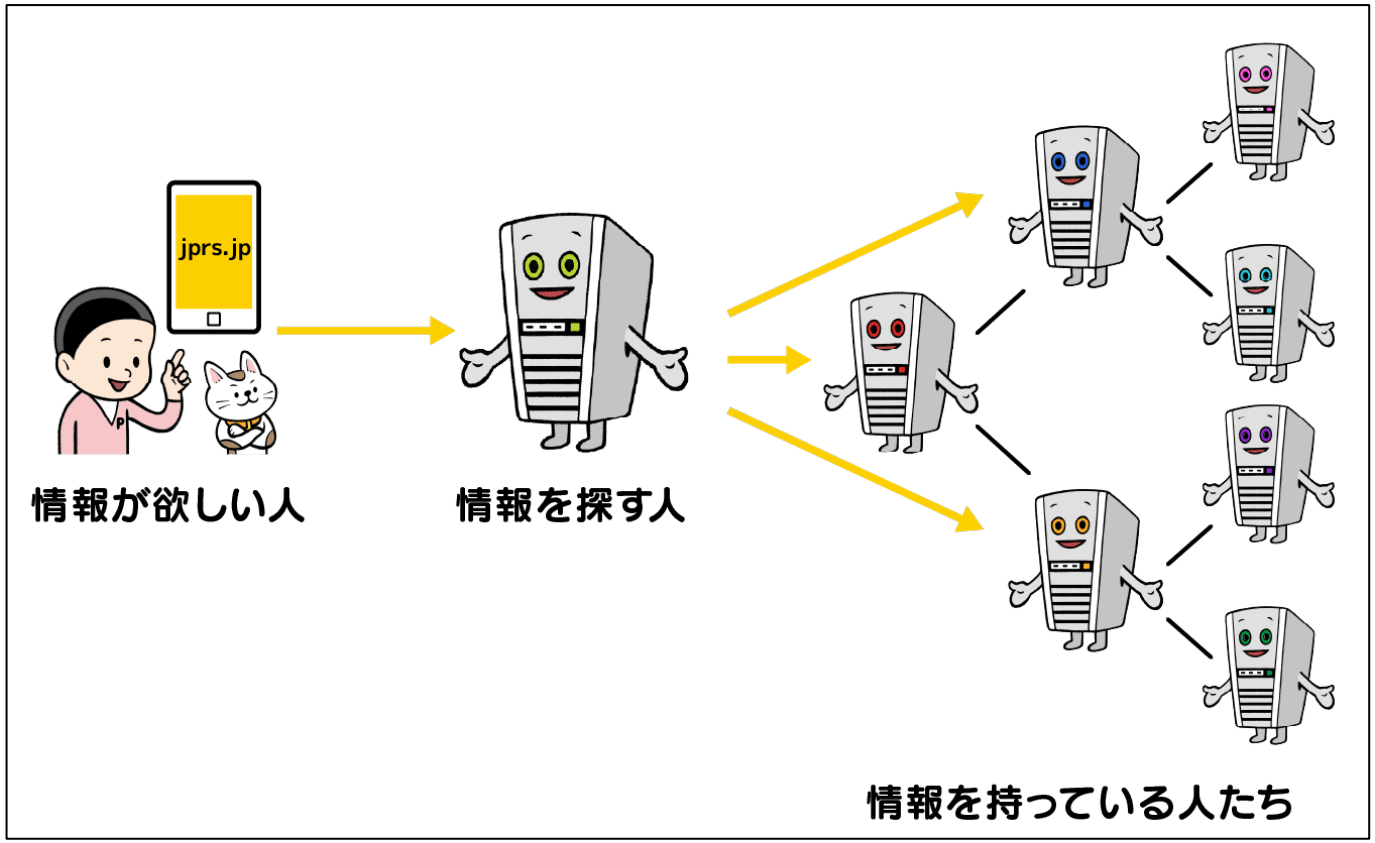
ドメイン名・階層化・委任・ゾーン

- 次のパートでは、**DNSの3種類の構成要素の役割と名前解決の仕組み**について解説する

2. 構成要素の役割と名前解決の仕組み

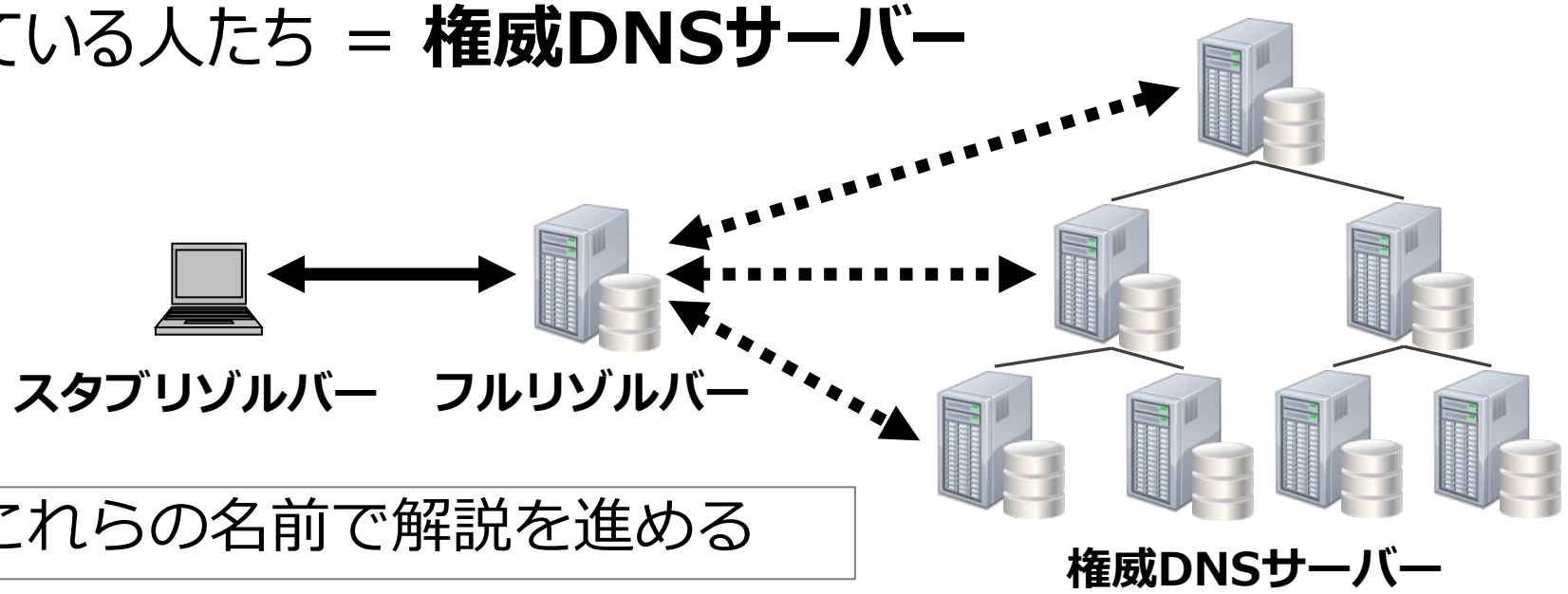
おさらい：DNSの構成要素

- 情報が欲しい人・情報を探す人・情報を持っている人たち



構成要素の名前

- それぞれの構成要素には、名前が付けられている
  - 情報が欲しい人 ＝ **スタブリゾルバー**
  - 情報を探す人 ＝ **フルリゾルバー**
  - 情報を持っている人たち ＝ **権威DNSサーバー**



以降では、これらの名前で解説を進める

統一されていない用語に注意

- 構成要素を示す用語は**統一されておらず**、文献・著者の違い・注目する機能などにより、**さまざまな名前で呼ばれている**
  - 混乱を招きやすく、**DNSの理解の妨げになりやすい**ため、注意が必要
- 用語の不統一は、**DNS全般の特徴**となっている
  - 用語を定義するためのRFC（**RFC 8499**）が発行されている

このチュートリアルで使う用語	現在使われている別の用語（主なもの）
スタブリゾルバー	DNSクライアントなど
フルリゾルバー	キャッシュDNSサーバー、参照サーバー、 ネームサーバー、DNSサーバーなど
権威DNSサーバー	権威サーバー、ゾーンサーバー、 ネームサーバー、DNSサーバーなど

「DNSサーバー」には特に注意

- 構成要素の「**フルリゾルバー**」と「**権威DNSサーバー**」はいずれも、「**DNSサーバー**」と呼ばれている
  - かつ、「フルリゾルバー」「権威DNSサーバー」「フルリゾルバーと権威DNSサーバーの双方」の、**いずれの意味でも使われる**
- 書籍や資料、会話などでDNSサーバーという用語が出て来た場合、**どの構成要素を指しているかの把握**が重要
  - 誤解を避けるため、**自分からはできるだけ使わない**ことを推奨

名前解決とは？

- 利用者の要求に応じ、**名前に対応する情報を取り出す**こと
  - 対応するIPアドレスや、メールサーバーホスト名の検索など

example.jpのIPv4アドレス → 192.0.2.1

example.jpのメールサーバー → mail.example.jp

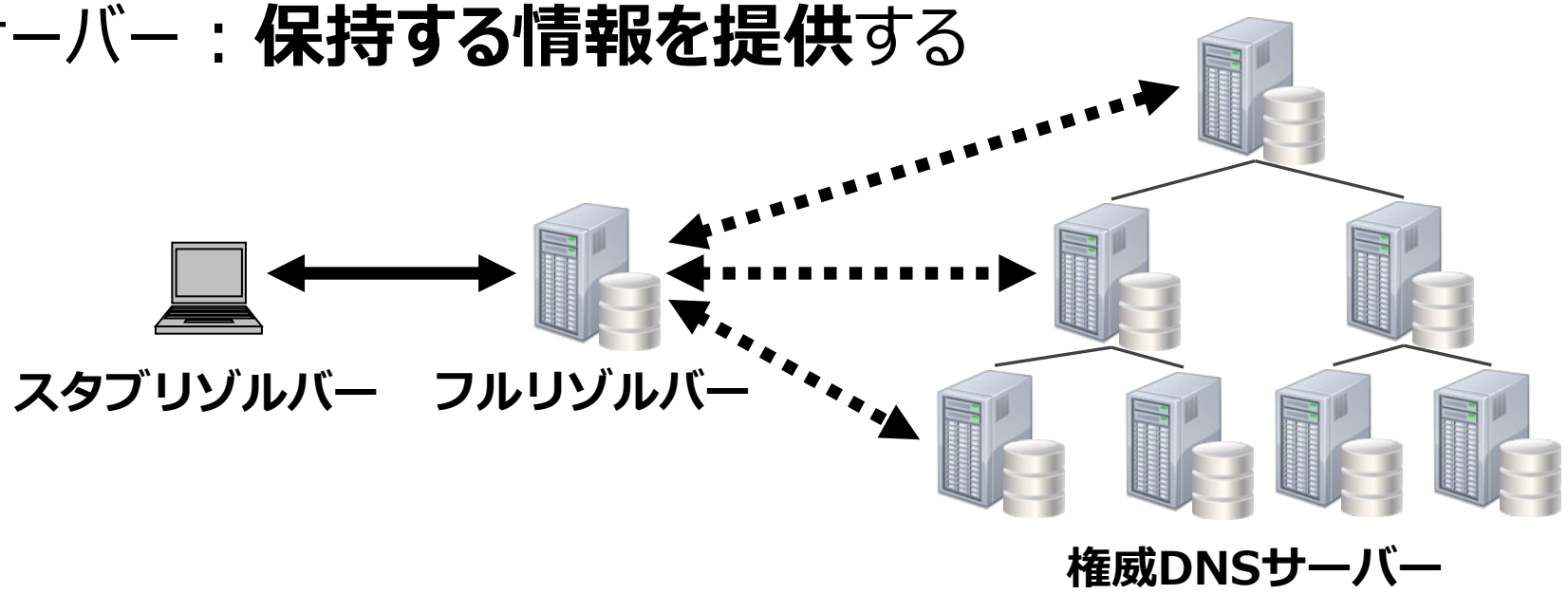
- DNSの名前解決では、**ドメイン名と種類（タイプ）の双方を指定**して問い合わせる

example.jpのIPv4アドレスは？

example.jpのメールサーバーホスト名は？

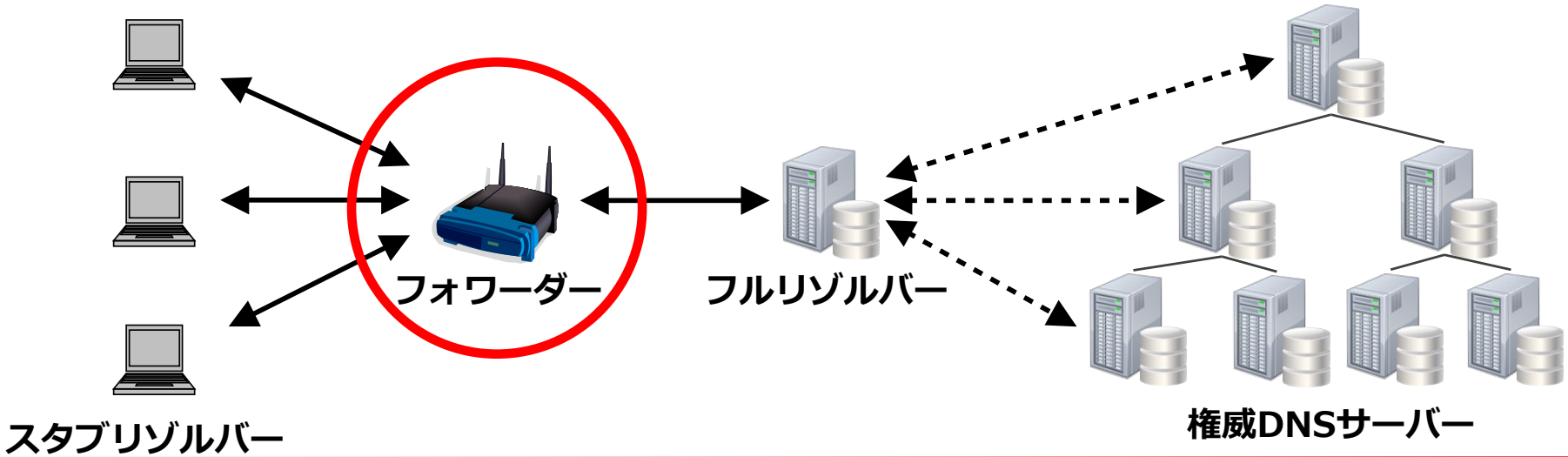
名前解決における構成要素の役割

- 以降では、名前解決における構成要素の役割について解説する
  - スタブリゾルバー：**名前解決を依頼する**
  - フルリゾルバー：**名前解決を実行**し、得られた**応答を蓄える**
  - 権威DNSサーバー：**保持する情報を提供**する



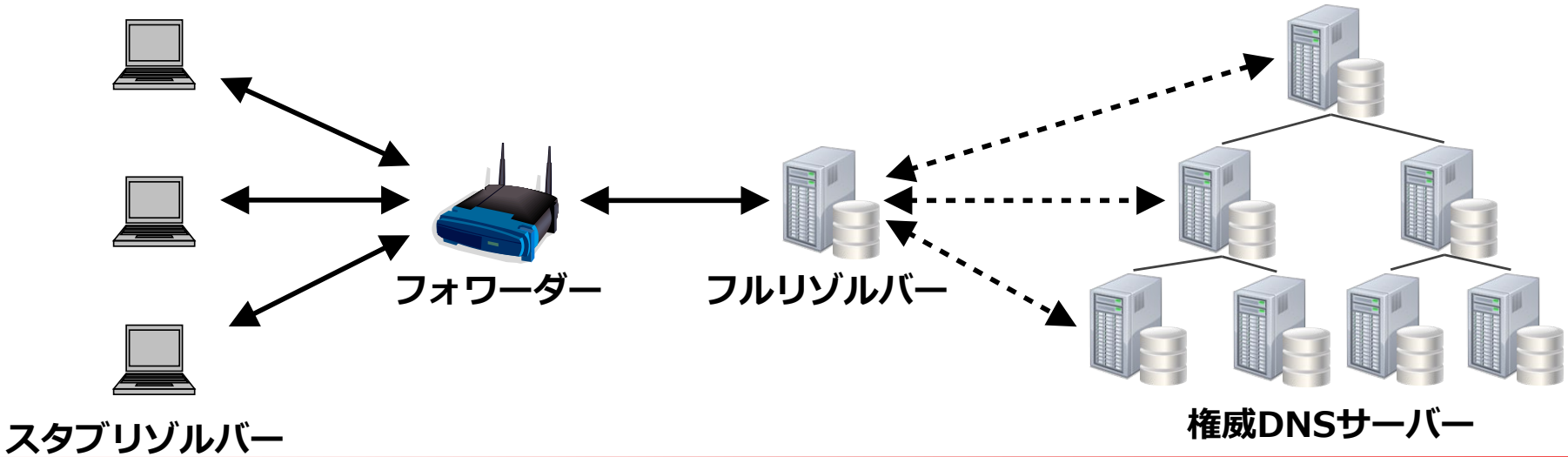
第4の構成要素：フォワーダー

- スタブリゾルバーとフルリゾルバーの間に**フォワーダー**が配置される場合がある
  - 役割：**名前解決を転送する**
- 身近なフォワーダーの例：**ホームルーター**
  - スタブリゾルバーからはフルリゾルバーに、フルリゾルバーからはスタブリゾルバーに見える



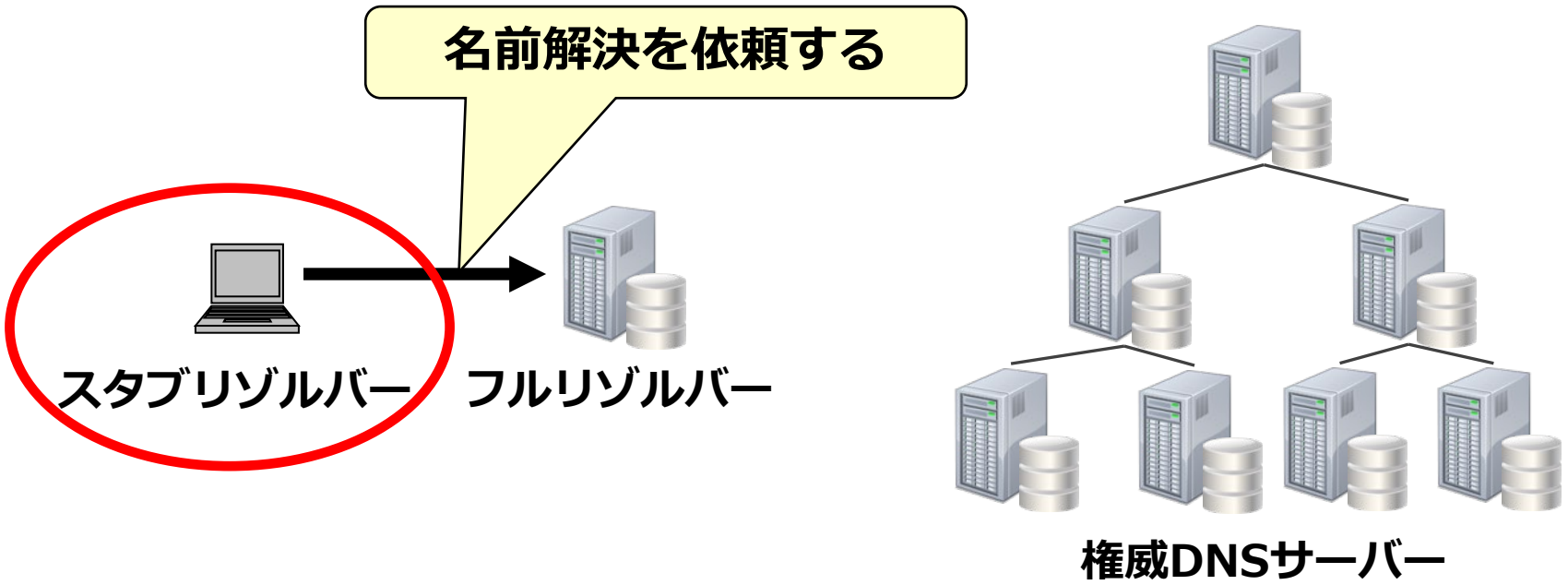
フォワーダーのメリット

- 機能が限られている機器で、フルリゾルバーの機能を提供できる
- 接続ISPを変更したり、複数ISPを使い分けたりする際、利用者のDNS設定を変更する必要がなくなる
  - IPv4のプライベートネットワークや、NATとの相性が良い



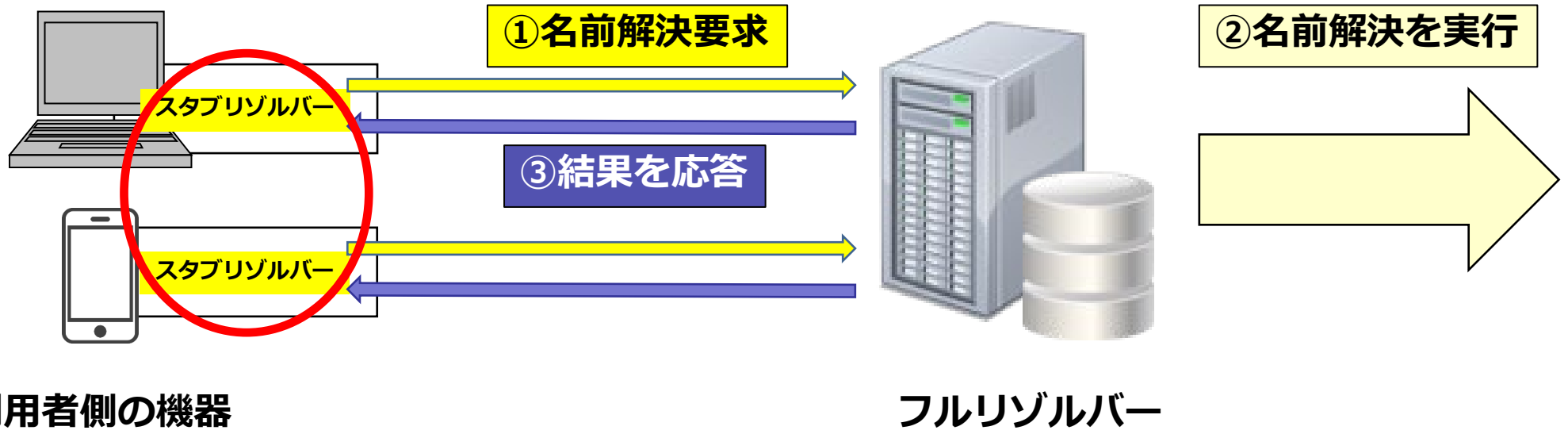
スタブリゾルバーの役割

- Webブラウザーなどのアプリケーションから呼び出され、フルリゾルバーに**名前解決を依頼する**
  - この動作を「**名前解決要求**」と呼ぶ



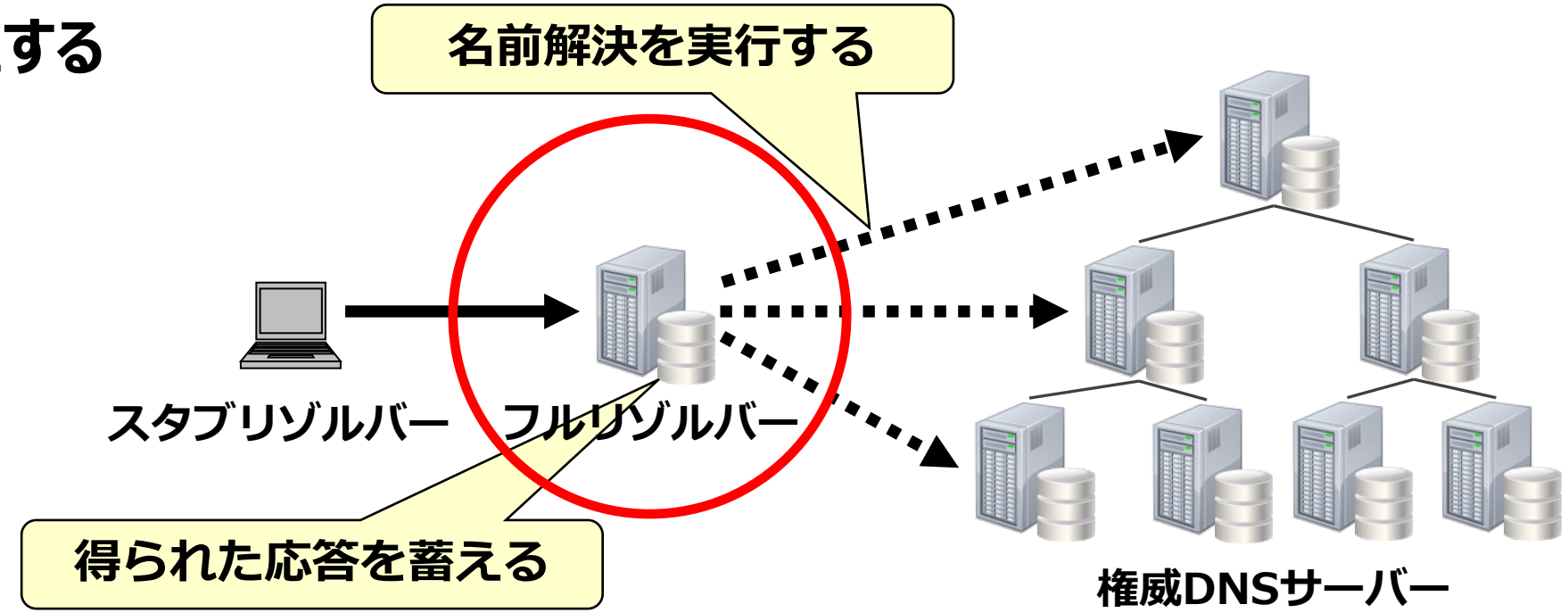
名前解決要求

- 「**私の代わりに名前解決を実行して、結果を教えてください**」という問い合わせを送る
  - 名前解決要求を受け取ったフルリゾルバーが名前解決を実行し、結果をスタブリゾルバーに応答する



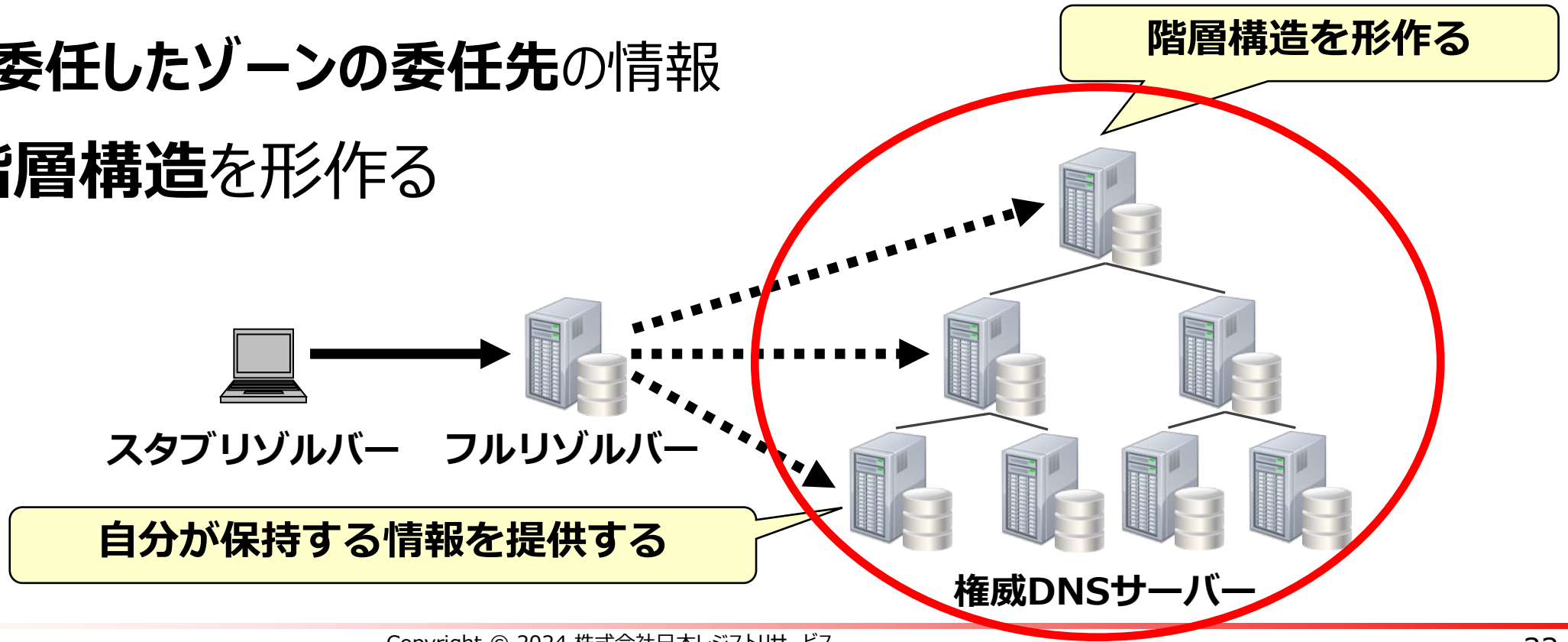
フルリゾルバーの役割

- 名前解決要求を受け取って**名前解決を実行**し、結果を返す
  - ルートから始めて、権威DNSサーバーへの問い合わせを繰り返す
- 次回以降の名前解決に使うため、**得られた応答を蓄える**
  - **キャッシュする**



権威DNSサーバーの役割

- 問い合わせに応じ、**自分が保持する情報を提供**する
  - **自分が管理するゾーン**の情報
  - **自分が委任したゾーンの委任先**の情報
- DNSの**階層構造**を形作る



まとめ：構成要素の役割と名前解決の仕組み

- DNSの名前解決では、**ドメイン名と種類（タイプ）の双方を指定**して問い合わせる
- DNSの名前解決は、**三つの構成要素**が連携して実行される
  - **スタブリゾルバー：名前解決を依頼する**
  - **フルリゾルバー：名前解決を実行し、得られた応答を蓄える**
  - **権威DNSサーバー：保持する情報を提供する**
- 次のパートでは**DNSの名前解決における具体的な動作**について、**現在の状況を踏まえた形で解説**する

3. 現状を踏まえた名前解決の流れ

このパートの内容

- このパートでは、DNSの名前解決の流れを説明する

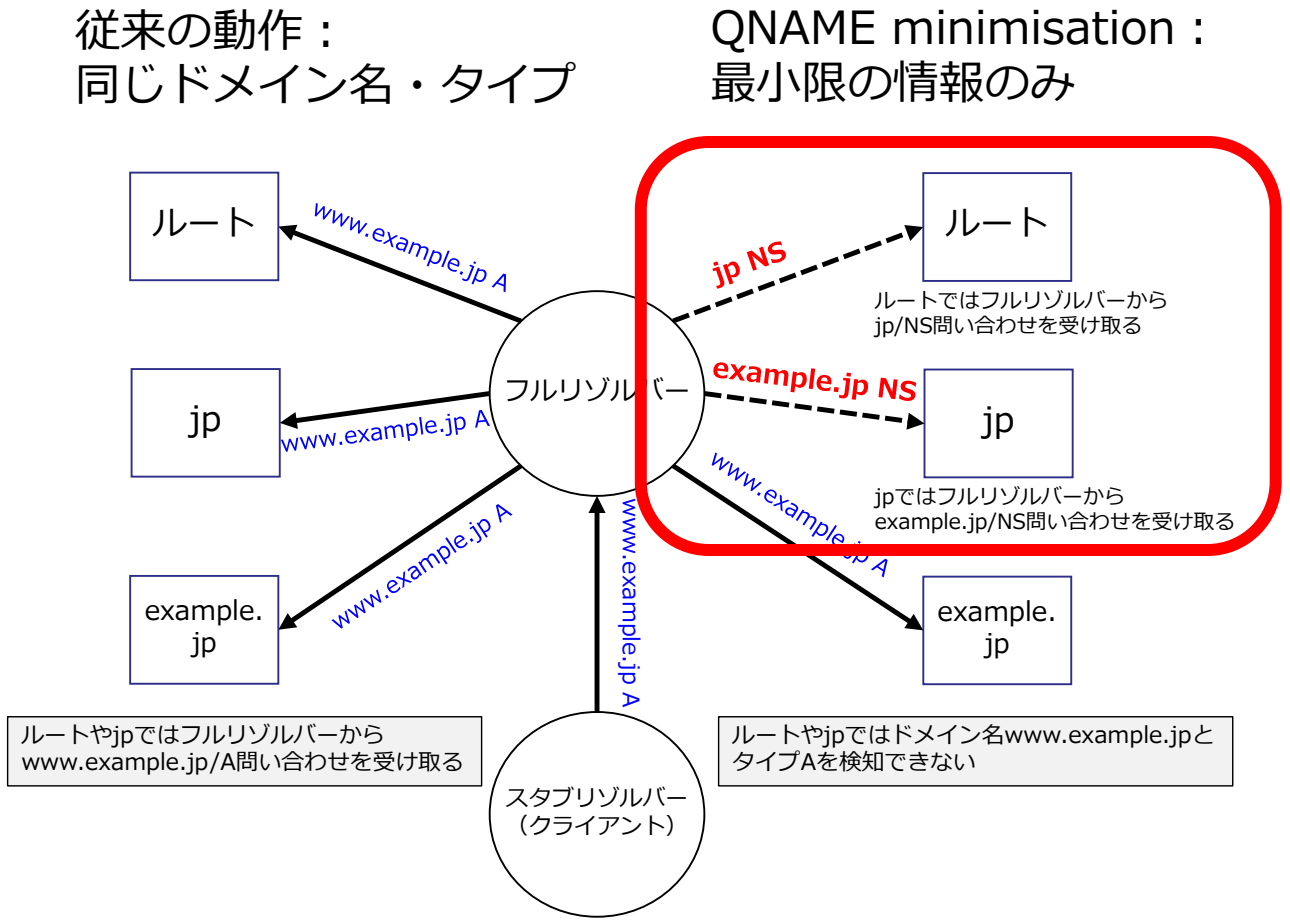
その前に…

- 現在のインターネットではDNSにおける**プライバシー上の懸念点**を解決するため、名前解決のアルゴリズムが**従来のものから変更**されている
  - 従来：名前解決要求された**問い合わせ内容をそのまま使う**
  - 現在：権威DNSサーバーに問い合わせる際、**情報を小出し**にする

この仕組みを「QNAME minimisation」と呼ぶ

QNAME minimisation

- フルリゾルバーの**名前解決アルゴリズムを変更**
  - 委任元の権威DNSサーバーには、**委任先のみを問い合わせる**
- ルートサーバーやTLDの権威DNSサーバーには、**委任先のドメイン名を利用するつもりであることのみが伝わる**
  - 利用者が問い合わせたドメイン名・タイプを**知ることができなくなる**



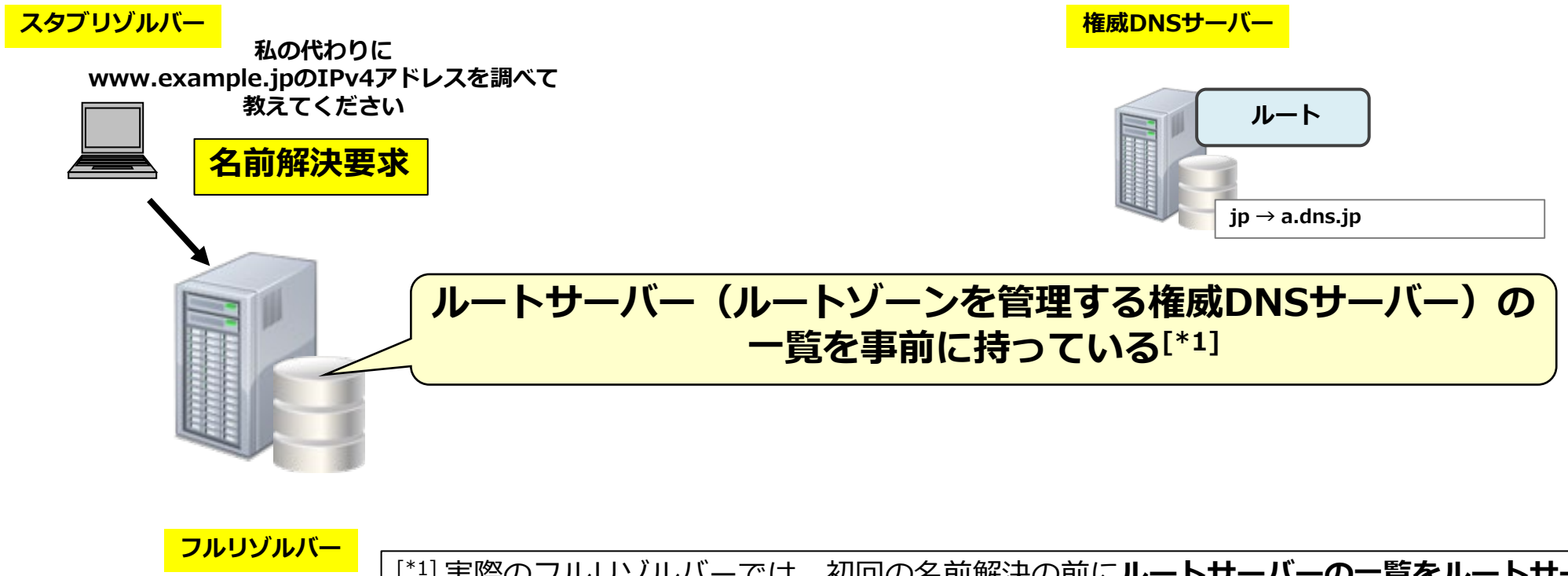
このチュートリアルでは、QNAME minimisationに対応した名前解決の流れを解説する

QNAME minimisationで使われる問い合わせのタイプ

- 当初の仕様（RFC 7816：2016年）では、**NSレコード**を使用
  - 委任先の情報（NS+グルーレコード）を得るための問い合わせ
- しかし、**NSレコードを適切にサポートしない一部のサーバーにQNAME minimisationで問い合わせた場合、名前解決に失敗してしまう場合がある**ことが報告された
- この問題を避けるため、現在の仕様（RFC 9156：2021年）では、**A/AAAAレコード**の使用を推奨
  - 現在の多くのフルリゾルバーは、**Aレコード**を使用している
    - QNAME minimisationを使っていることのカモフラージュにも役立つ

QNAME minimisationに対応した名前解決の流れ

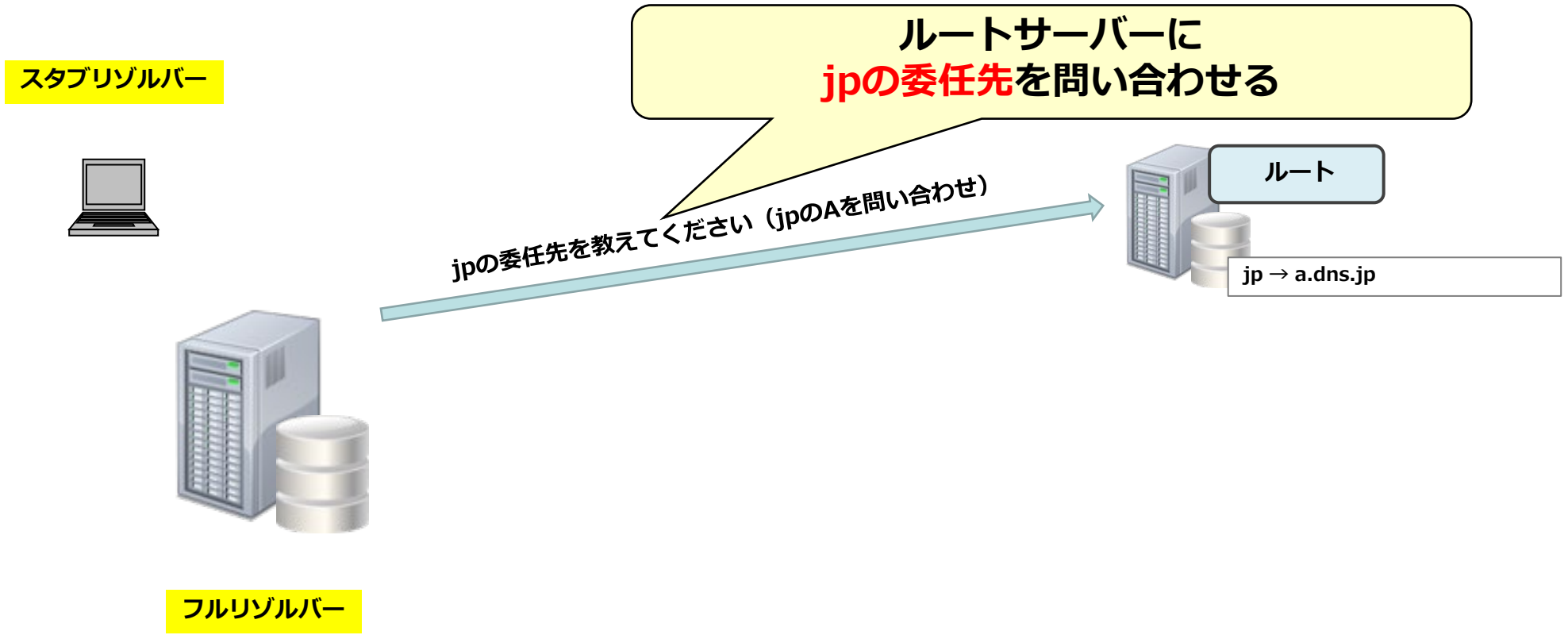
www.example.jpのIPv4アドレスを名前解決する場合の例



[*1] 実際のフルリゾルバーでは、初回の名前解決の前に**ルートサーバーの一覧をルートサーバー自身に問い合わせて応答をキャッシュし、以降はそれを使う（プライミング**：このチュートリアルでは説明を省略）

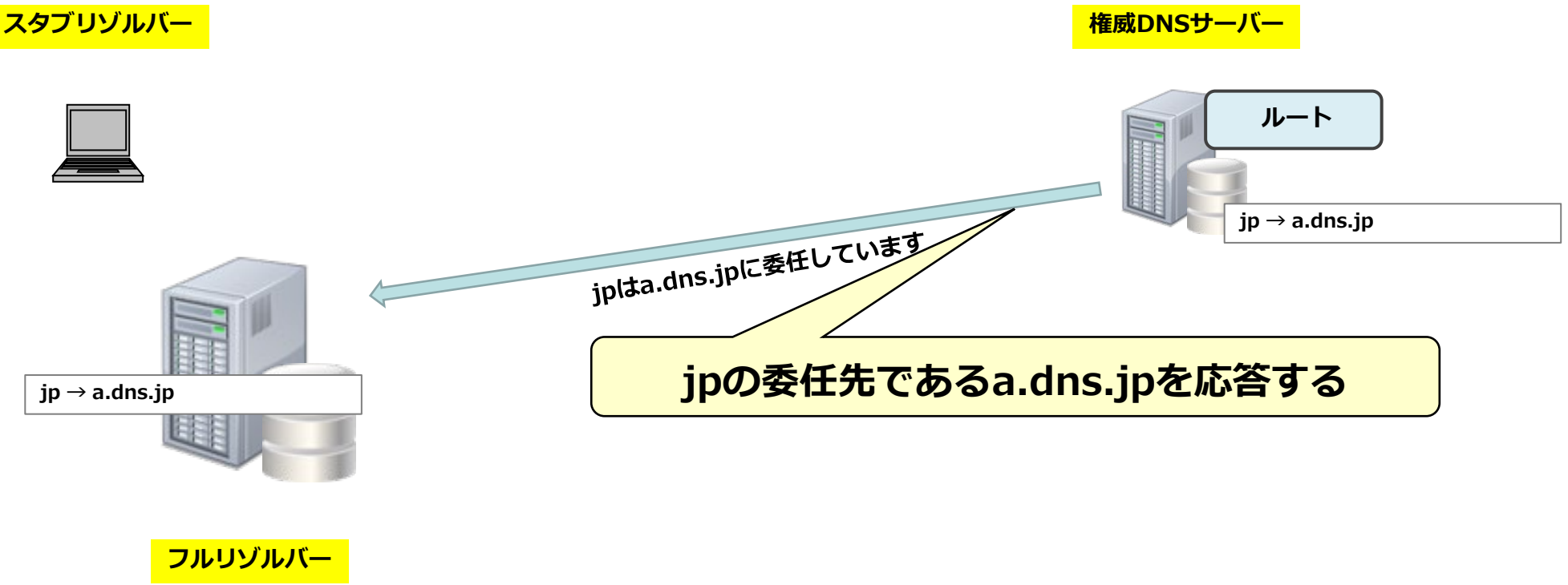
QNAME minimisationに対応した名前解決の流れ

www.example.jpのIPv4アドレスを名前解決する場合の例



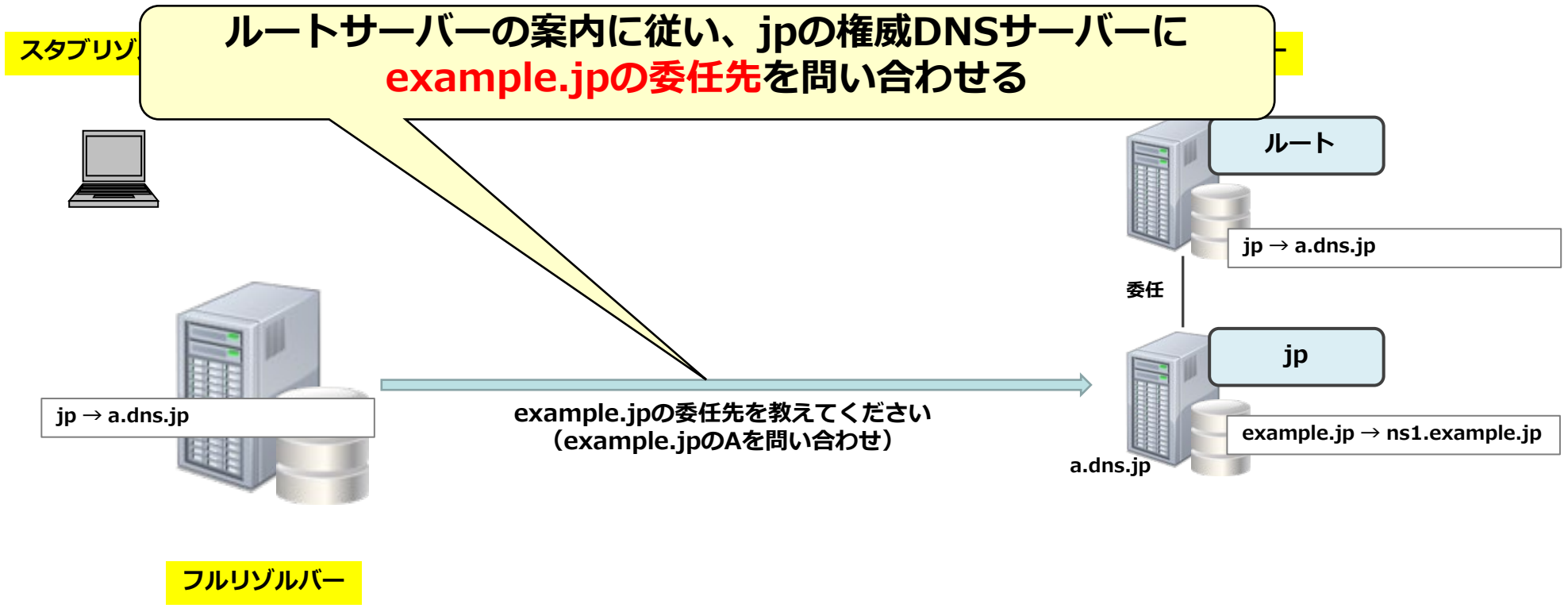
QNAME minimisationに対応した名前解決の流れ

www.example.jpのIPv4アドレスを名前解決する場合の例



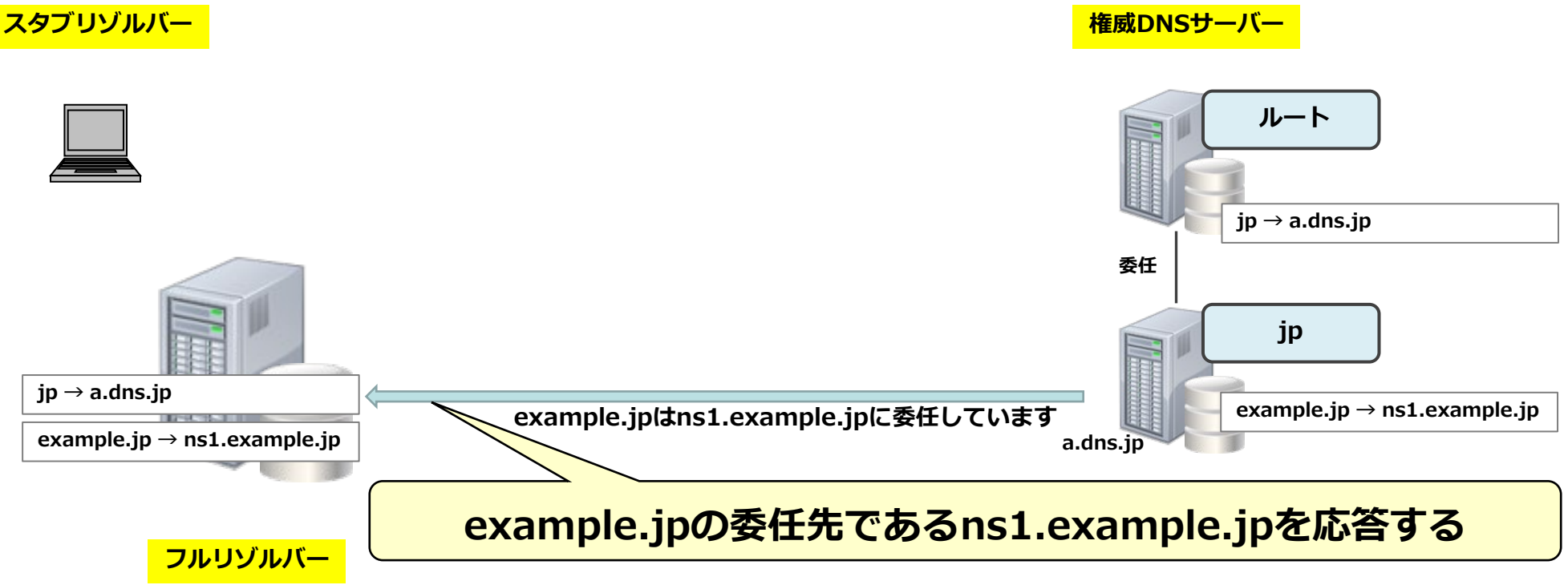
QNAME minimisationに対応した名前解決の流れ

www.example.jpのIPv4アドレスを名前解決する場合の例



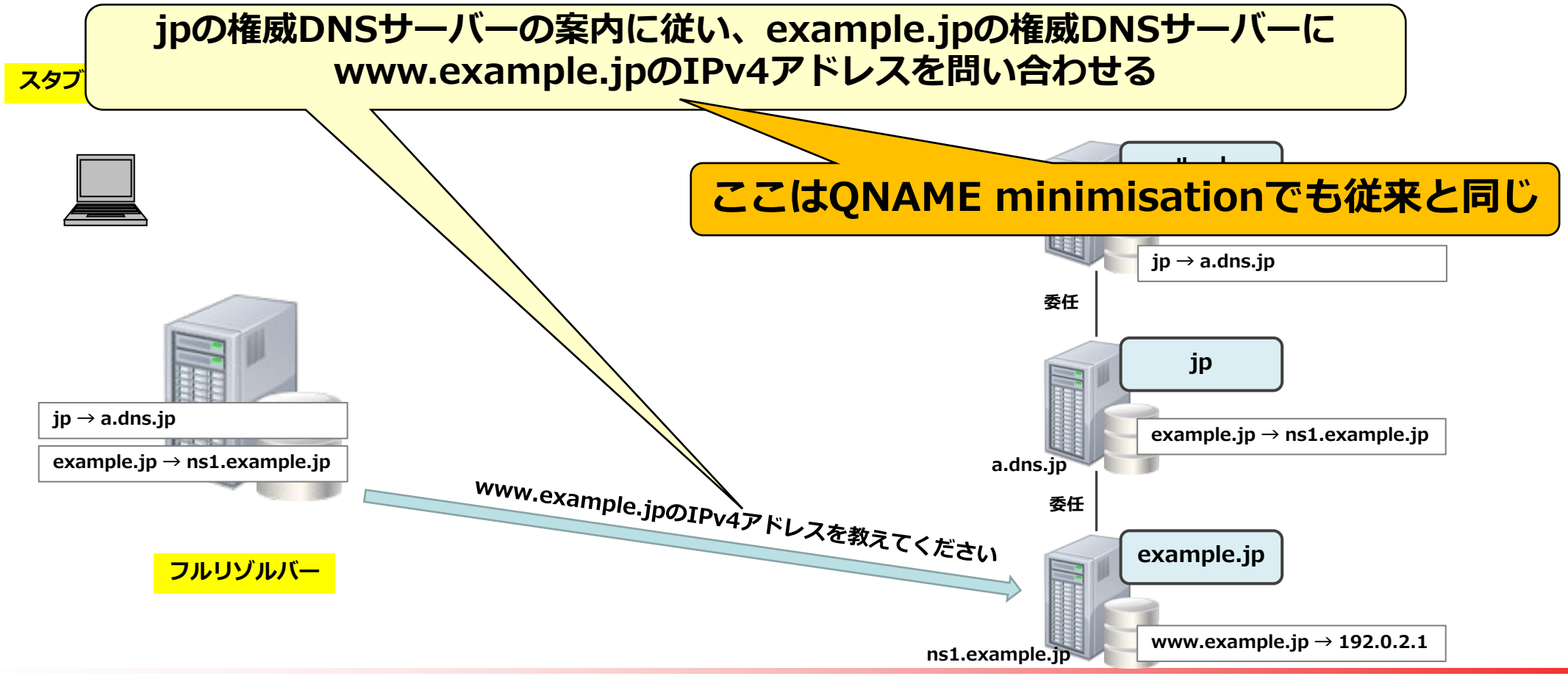
QNAME minimisationに対応した名前解決の流れ

www.example.jpのIPv4アドレスを名前解決する場合の例



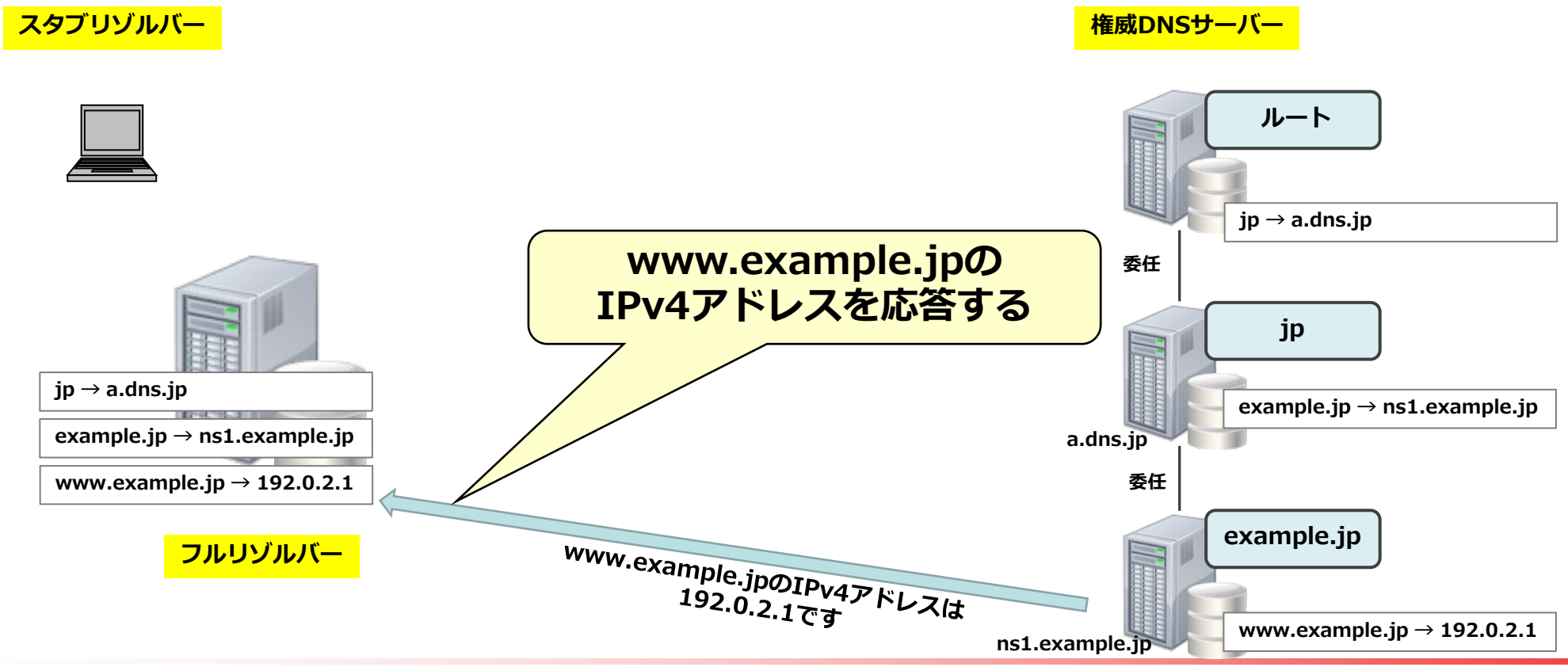
QNAME minimisationに対応した名前解決の流れ

www.example.jpのIPv4アドレスを名前解決する場合の例



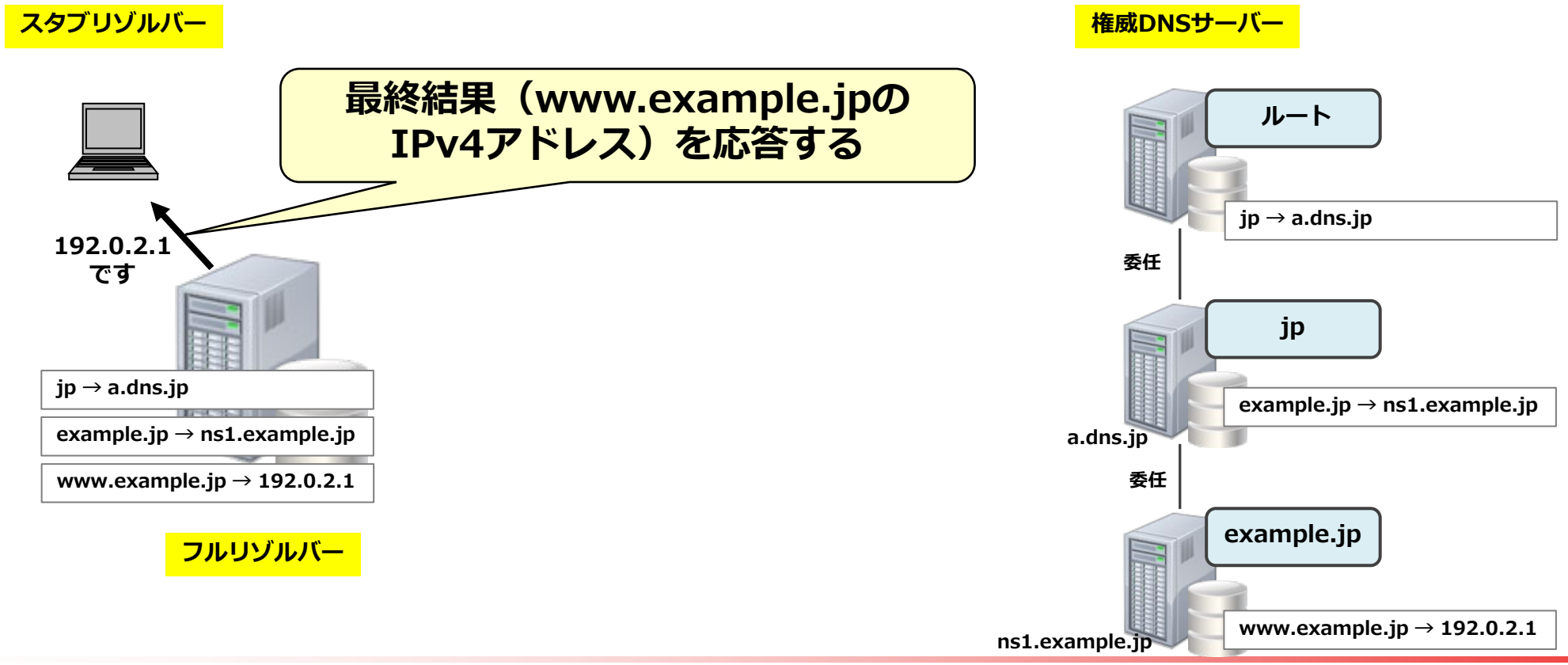
QNAME minimisationに対応した名前解決の流れ

www.example.jpのIPv4アドレスを名前解決する場合の例



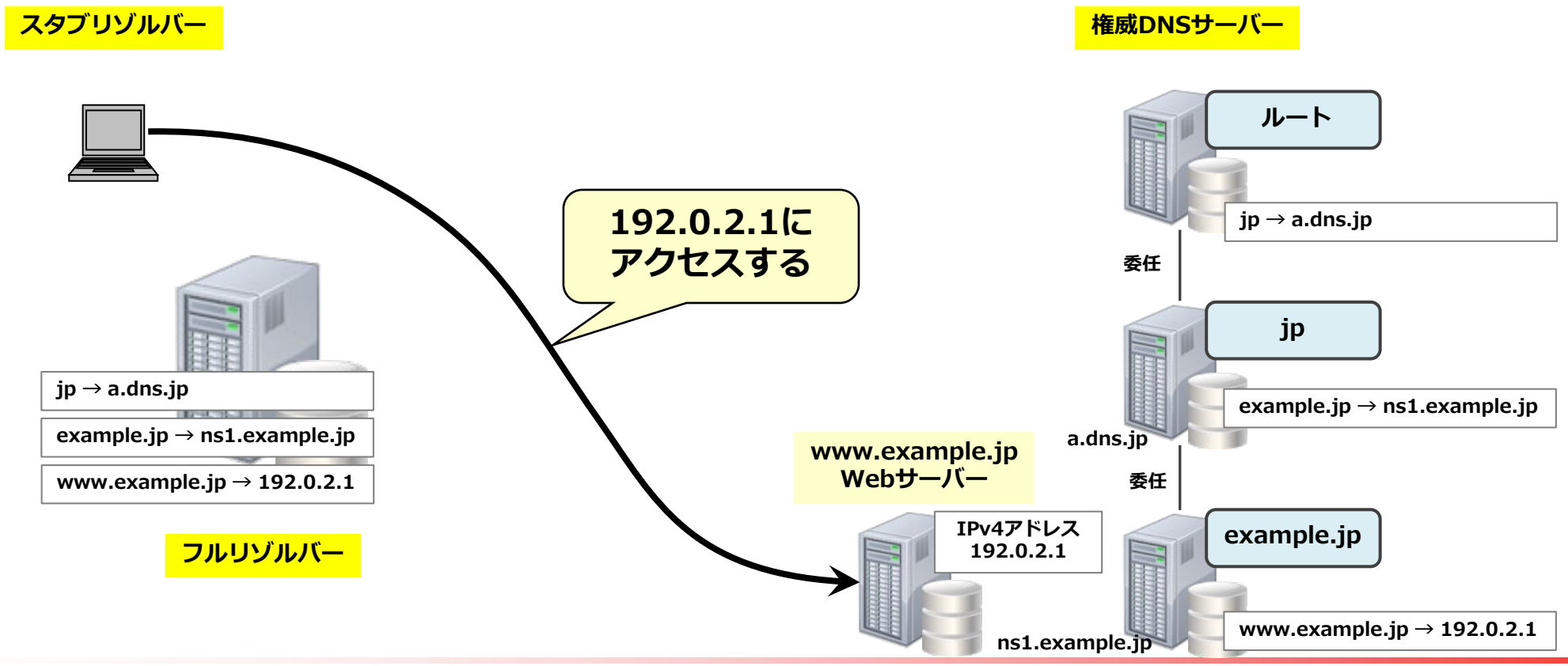
QNAME minimisationに対応した名前解決の流れ

www.example.jpのIPv4アドレスを名前解決する場合の例



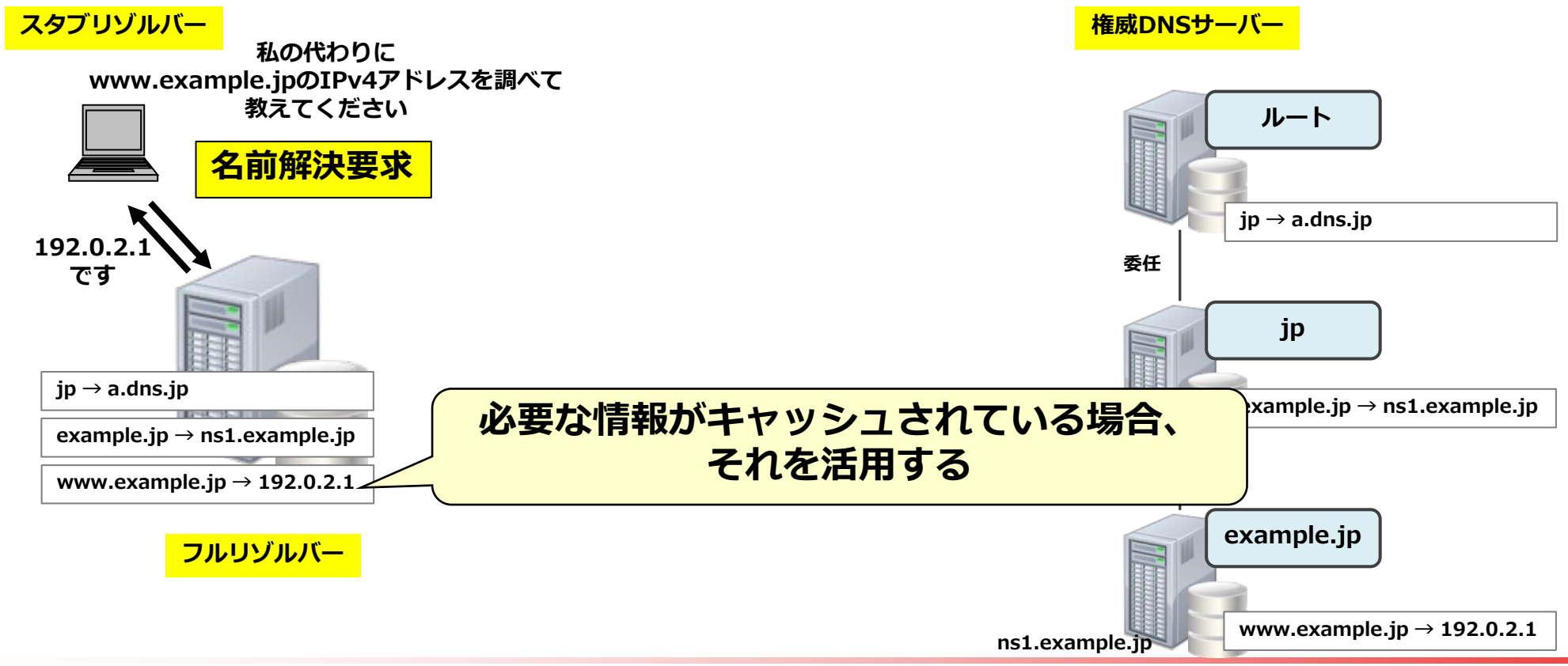
QNAME minimisationに対応した名前解決の流れ

www.example.jpのIPv4アドレスを名前解決する場合の例



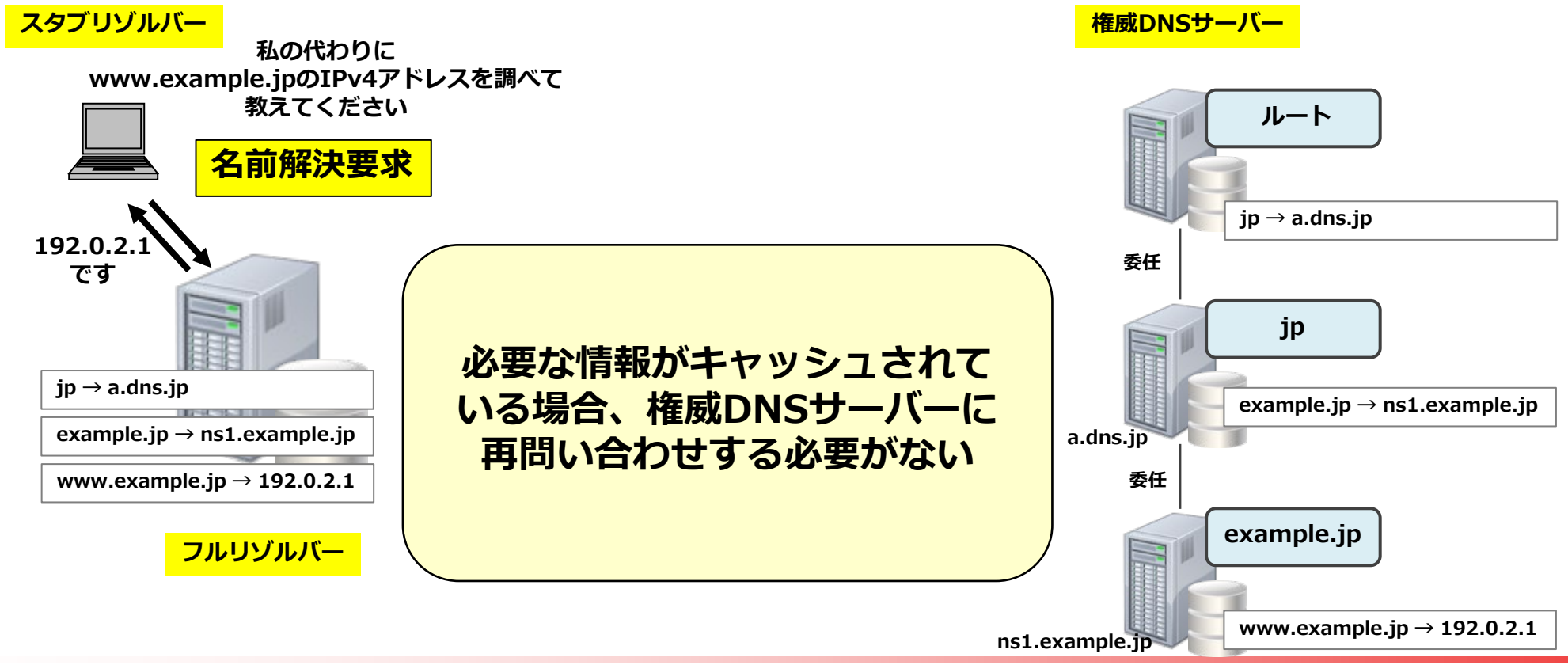
名前解決の流れ（2回目以降）

www.example.jpのIPv4アドレスを名前解決する場合の例



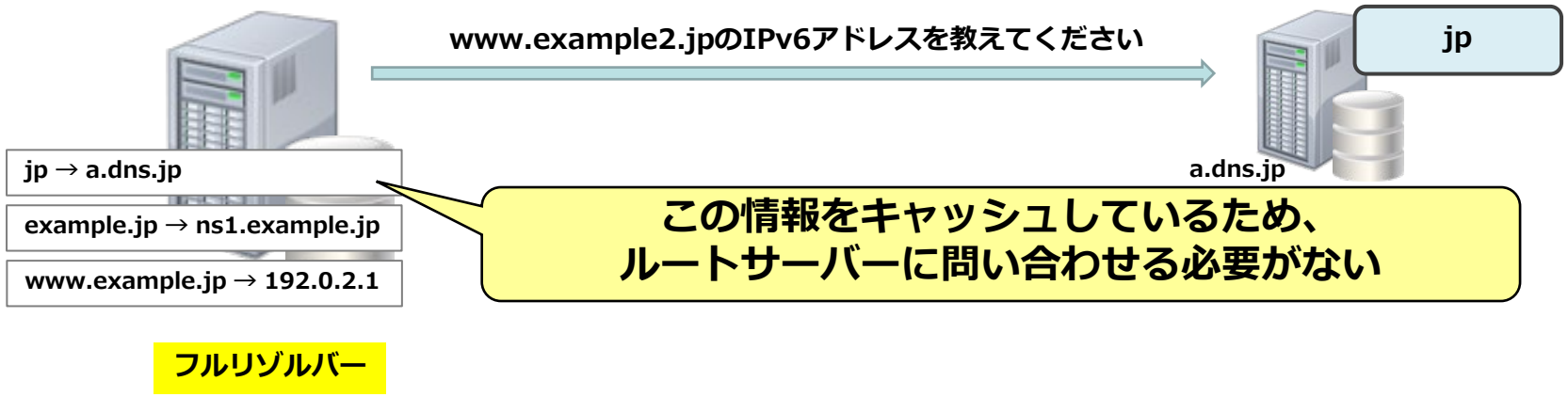
キャッシュの活用

- キャッシュを活用して、名前解決の**負荷と時間を軽減**する



途中の情報もキャッシュ・活用

- 名前解決の**途中で得られた情報もキャッシュ・活用**される
  - 例えば、**www.example.jp**のIPv4アドレスを名前解決した直後に**www.example2.jp**のIPv6アドレスが問い合わされた場合、**ルートサーバーへの問い合わせを省略して、jpのサーバーから問い合わせを始める**ことができる

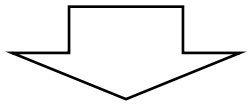


キャッシュの有効時間：TTL

- フルリゾルバーが**データをキャッシュしてよい時間**は、管理者が**自分のゾーンの権威DNSサーバーで設定**する
- この時間を**TTL**（Time To Live）と呼ぶ
  - TTLには**キャッシュの生存時間**を**秒単位**で設定する
    - 時間の経過と共に**TTLが減算**され、**0になるとキャッシュが満了**

www.example.jp → 192.0.2.1

このデータを最大**3600秒（1時間）**キャッシュさせたい



```
www.example.jp. 3600 IN A 192.0.2.1
```

ゾーンの管理者がTTLを**3600**に設定

「存在しない」こともキャッシュ・活用

- 名前解決では「**問い合わされたデータは存在しない**」という応答（**不在応答**）もキャッシュされ、次回以降の名前解決で使われる
  - **ネガティブキャッシュ**と呼ぶ
- ネガティブキャッシュのTTL[*1]も、管理者が自分のゾーンの権威DNSサーバーで設定する

[*1] ネガティブキャッシュのTTLは、そのゾーンのSOAレコード自身のTTLとSOAレコードのMINIMUMのうち、小さい方の値に設定される（このチュートリアルでは説明を省略）

まとめ：現状を踏まえた名前解決の流れ（QNAME minimisationによる名前解決）

- DNSにおける**プライバシー上の懸念点**を解決するため、名前解決のアルゴリズムが**従来のものから変更**されている
  - 従来：名前解決要求された名前・タイプをそのまま使用
  - 現在：権威DNSサーバーに問い合わせる際、**情報を小出し**にする
- この仕組みを、**QNAME minimisation**と呼んでいる
- 現在の主なフルリゾルバーはQNAME minimisationにおいて、委任先ゾーンの**Aレコード**を問い合わせている

まとめ：現状を踏まえた名前解決の流れ（キャッシュとネガティブキャッシュ）

- 名前解決の**負荷と時間を軽減**するために情報を蓄える、**キャッシュ**という仕組みが存在する
  - **名前解決の途中で得られた情報もキャッシュ・活用**される
  - キャッシュの有効期限（**TTL**）は、**各ゾーンの権威DNSサーバーで設定**する
- **存在しない旨の応答（不在応答）もキャッシュ・活用**される
  - **ネガティブキャッシュ**と呼ばれる
- 実は、不在応答の取り扱いには**ちょっとした落とし穴**がある
  - 次のパートで解説する

4. 現場のトラブルシューティングにおける二つのポイント

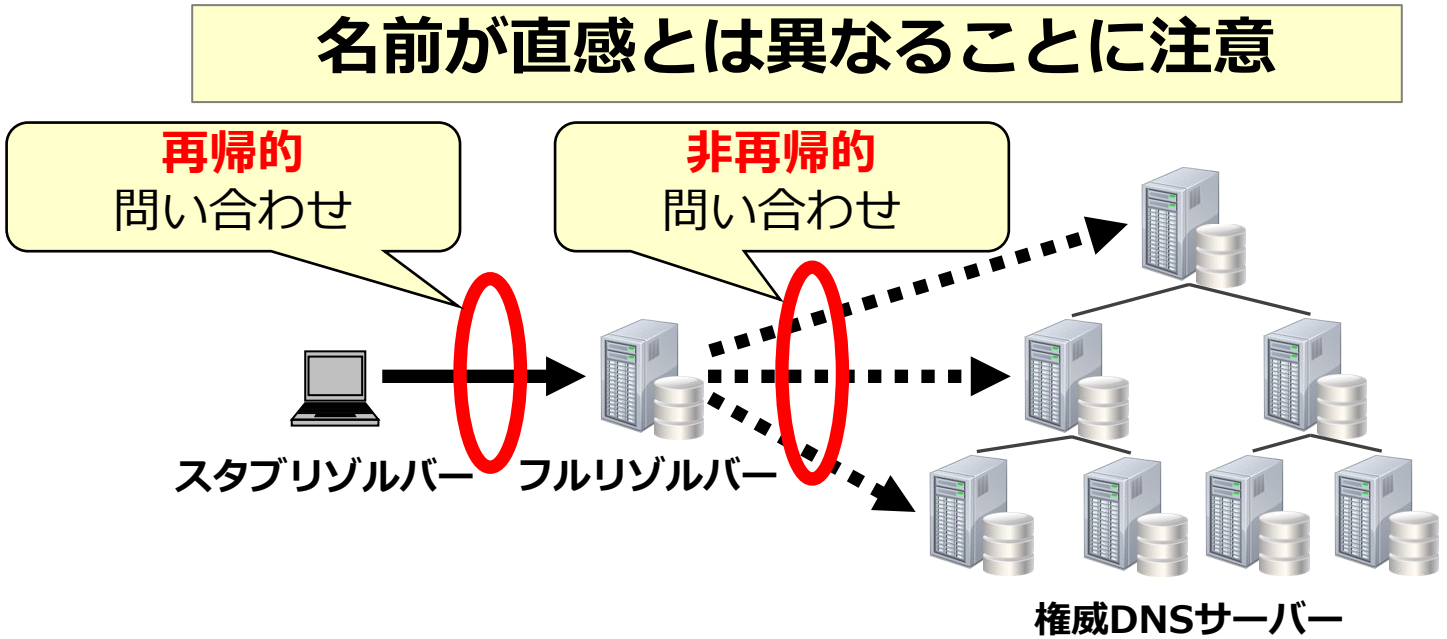
本パートで取り上げる内容

- このパートでは運用の現場で役立つ、二つのポイントについて解説する
  - その1：**役割が異なる2種類の問い合わせ**
  - その2：**意味が異なる2種類の不在応答**
- DNSのトラブルシューティングのポイントには他にもいろいろあるが、この二つは以下の点において重要
  - その1：2種類の問い合わせがあることを理解し、**調査対象により使い分けることで、トラブルの原因切り分け・対応に役立つ**
  - その2：2種類の不在応答があることを理解し、**機器ベンダーやサービスプロバイダーの実装を調査することで、トラブルの原因切り分け・対応に役立つ**

その１：役割が異なる2種類の問い合わせ

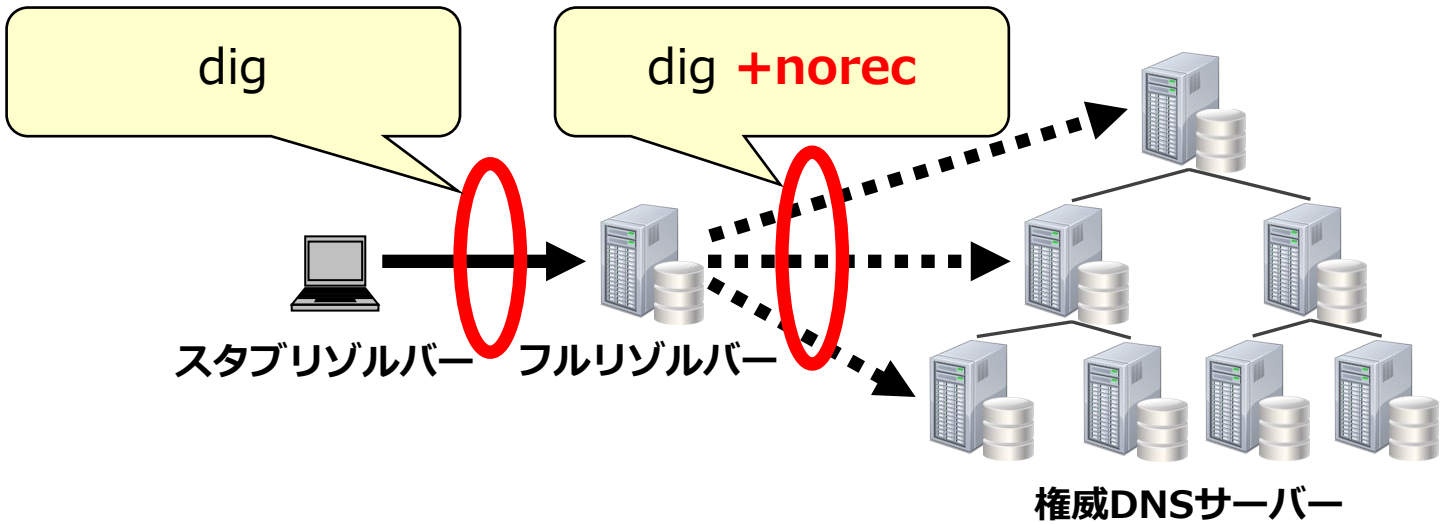
役割が異なる2種類の問い合わせ

- DNSには**再帰的問い合わせ**と**非再帰的問い合わせ**という、**役割が異なる2種類の問い合わせ**が存在する
  - **再帰的問い合わせ：スタブリゾルバー**から**フルリゾルバー**への問い合わせ
  - **非再帰的問い合わせ：フルリゾルバー**から**権威DNSサーバー**への問い合わせ



digコマンドでの使い分け

- digコマンドは、再帰的問い合わせ・非再帰的問い合わせの双方をサポートしている
  - **+rec（デフォルト）：再帰的問い合わせを送信**
  - **+norec：非再帰的問い合わせを送信**
- 権威DNSサーバーをdigコマンドで調べる場合は、**+norecオプションを付ける**
  - **フルリゾルバーの気持ちになることが重要**



使い分けのメリット

- 今どちらのサーバーを調べているか、**明確に意識付けできる**ようになる
  - 調査対象が権威DNSサーバーかフルリゾルバーか、常に意識することが重要
    - どちらかわからないと、**うまくトラブルシューティングできない**
- BINDで運用されているサーバーなど、**過去の経緯などにより双方の機能を兼用しているサーバーを調べる**際に、特に役立つ
  - もちろん、権威DNSサーバーとフルリゾルバーは**そもそも兼用しない方がよい**
    - 同じサーバーに入れる場合、権威DNSサーバーとフルリゾルバーのプロセス・IPアドレスを分ける

その2：意味が異なる2種類の不在応答

意味が異なる2種類の不在応答

- 権威DNSサーバーが返す応答は、以下の**6種類**に分類される
  - ① 問い合わされた**名前・タイプに合致するレコードが存在する**
  - ② 問い合わされた名前とそのサブドメインには**レコードが一つも存在しない**
  - ③ 問い合わされた**名前・タイプに合致するレコードは存在しない**
    （他のタイプのレコードや、サブドメインは存在するかもしれない）
  - ④ 問い合わされた名前は**別名である**
  - ⑤ 問い合わされた名前は**他のサーバーに委任している**
  - ⑥ 問い合わされた名前を**管理も委任もしていない**（管理範囲外）

意味が異なる2種類の不在応答

DNSには、意味が異なる不在応答が2種類存在する

② 問い合わされた名前とそのサブドメインには**レコードが一つも存在しない**

③ 問い合わされた**名前・タイプに合致するレコードは存在しない**
（他のタイプのレコードや、サブドメインは存在するかもしれない）

**これら2種類の不在応答は意味が異なるため、
区別して取り扱う必要がある**

②の応答の例（NXDOMAIN応答）

- statusが**NXDOMAIN**
- flagsにaaがセット
- ANSWERが0
- AUTHORITYが1
  - 内容はそのゾーンのSOAレコード

```
% dig +norec +multi noexist.jprs.jp a @ns1.jprs.jp
…
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 46467
;; flags: qr aa; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1
…
;; AUTHORITY SECTION:
jprs.jp.   86400 IN SOA ns1.jprs.co.jp. postmaster.jprs.co.jp. (
                                        1701747003 ; serial
                                        3600       ; refresh (1 hour)
                                        900        ; retry (15 minutes)
                                        1814400    ; expire (3 weeks)
                                        86400      ; minimum (1 day)
                                        )
```

jprs.jpゾーンの権威DNSサーバーに
noexist.jprs.jpのAレコードを問い合わせた際の応答

意味：noexist.jprs.jpという名前とそのサブドメインには**レコードが一つも存在しない**

③の応答の例（NODATA応答）

- statusが**NOERROR**
- flagsにaaがセット
- ANSWERが0
- AUTHORITYが1
  - 内容はそのゾーンのSOAレコード

```
% dig +norec +multi www.jprs.jp txt @ns1.jprs.jp
…
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33310
;; flags: qr aa; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1
…
;; AUTHORITY SECTION:
jprs.jp.   86400 IN SOA ns1.jprs.co.jp. postmaster.jprs.co.jp. (
                                        1701747003 ; serial
                                        3600       ; refresh (1 hour)
                                        900        ; retry (15 minutes)
                                        1814400    ; expire (3 weeks)
                                        86400      ; minimum (1 day)
                                        )
```

jprs.jpゾーンの権威DNSサーバーに
www.jprs.jpのTXTレコードを問い合わせた際の応答

意味：www.jprs.jpという名前には**TXTレコードが存在しない**
（他のタイプのレコードや、サブドメインは存在するかもしれない）

NODATAを返すべきケースで、誤って NXDOMAINを返す実装・サービスが存在する

- **NODATA応答を返すべきケースにおいて、誤ってNXDOMAIN応答を返してしまう権威DNSサーバーやネットワーク機器、サービス**が存在する
- **ロードバランサー**や**アプライアンスサーバー**の一部で見受けられる
  - 2005年発行のRFC 4074（著者：森下・神明達哉さん）で、**よくある実装ミスの一つ**として紹介したが、**現在に至るまで不定期に発生**している
- **QNAME minimisationとの食い合わせが悪いため、問題が顕在化**
  - Spamhaus**DNSBL**でも発生し、問題になった[*1]

[*1] QNAME Minimization and Spamhaus
<https://kb.isc.org/docs/qname-minimization-and-spamhaus>

まとめ：運用の現場における注意（2種類の問い合わせ）

- DNSの**不安定な動作**や**不可解な名前解決エラー**の原因の特定には、**権威DNSサーバーの個別調査**が欠かせない
  - 例：あるゾーンの権威DNSサーバーのうち、1台の動作がおかしい
- **+norecオプション**を覚えて、**2種類の問い合わせを使い分ける**
  - digの代わりにdrillが付いている場合は、**-o rdオプション**を使う
    - +multiオプションも併用するとよい（出力が読みやすくなる）
- 権威DNSサーバーを調べる時は、**フルリゾルバーの気持ちになる**
  - dig +traceでの調査だけでは不十分なケースも多い

まとめ：運用の現場における注意（2種類の不在応答）

- **接続先の権威DNSサーバーの不具合により、自分のフルリゾルバーの名前解決がうまくいかなくなる**場合がある
  - 例：Aレコードが設定されているにも関わらず、名前解決エラーになる
    - 接続先の権威DNSサーバーの実装者が、**2種類の不在応答を認識できていない**
- 接続先のネットワーク機器やサービスの不具合が修正されない場合、**自分のフルリゾルバーの設定を一時的に変更し、運用でカバーする**ことも考えられる
  - 例：問い合わせるドメイン名を見て、QNAME minisationをオフにする
  - **お勧めできない**（これが繰り返された結果、**DNS flag day**が必要になった）

おわりに

- このチュートリアルでは**DNSの構成要素と分散管理の仕組みと役割**、**QNAME minimisationによる名前解決の動作**に加え、**運用の現場で役立つ二つのポイント**についても解説しました。
- 今回のチュートリアルが、**インターネットを支える重要な基盤技術の一つ**である、**DNSについて学び直すきっかけ**となれば幸いです。

ご清聴ありがとうございました！

<https://jprs.jp/tech/>

@JPRS_official　JPRSofficial　JPRSpress