DNS(コンテンツ)サーバの各実装による動作の相違とその考察
- BIND 9, djbdns, NSD -

Internet Week 2003
DNS DAY
2003年12月2日 (2004年1月29日加筆訂正)
株式会社日本レジストリサービス(JPRS)
森下 泰宏 <yasuhiro@jprs.co.jp>

このコマの目的

- フリーソフト/オープンソースのDNSサーバ実装であるBIND 9, djbdns, NSDについて、それぞれの機能・設計・設定方法・いくつかの場合における実際の動作の違い等について比較・考察することにより、それぞれのサーバの設計ポリシーを理解し、DNSに対する理解を深める一助とする

注: このコマではDNSサーバの各実装について、中立の立場で解説することを心がけます。

できるだけ

注意

- 設計に伴う特徴等、定性的な話について言及します
  - この実装ではデータ保持形式が～である
  - これは、～という効果を期待していると考えられる
  - この実装では他の実装と比較して、～という点に配慮している、等
- 各DNSサーバのパフォーマンス等の定量的な性能の相違については、今回は取り扱いません
  - 負荷テストで何queries/secを叩き出したか
  - ゾーンのリロードにどのくらいの時間がかかったか
  - どのくらいの大きさのゾーンファイルを保持・管理できたか
  - どのくらいのメモリ容量を消費するのか、等

注意 (cont.)

- BIND 8については、DNSサーバとしてのいくつかの動作について、比較対照としてのみ言及します
- 今回の考察対象は、DNSコンテンツサーバ機能のみとします
  - DNSキャッシュサーバ機能も非常に重要ですが、今回は取り上げません

おさらい
- BIND 9, djbdns, NSD -

Japan Registry Service

BIND 9

- **開発元**: Internet Software Consortium (ISC)
  - http://www.isc.org/products/BIND/bind9.html
- **代表的なDNSの実装の一つ**
- **DNSコンテンツサーバ、キャッシュサーバの双方の機能を備える**
- **従来のBIND 8までのBINDとは全く別の、スクラッチから作成されたコードにより動作**
- **「リファレンス実装」として、各種インターネット標準を広くサポート**

djbdns

- **開発者**: D. J. Bernstein**氏**
  - http://cr.yp.to/djbdns.html
- DNS**コンテンツサーバ、キャッシュサーバの双方の機能を備える**
- DNS**コンテンツサーバとキャッシュサーバは、それぞれ別のプログラムにより提供される**
- **開発者の哲学を色濃く反映**
  - **「思想的」とか、「使い方の押し付け」であるという声もある**
- **セキュリティに対する配慮**
- **機能ごとに分割された小さなプログラム**
- **簡潔な機能、設定ファイル**

NSD

- **開発元**: NLnet Labs
  - http://www.nlnetlabs.nl/nsd/
  - **オランダ**NLnet(**先進ネットワーク技術に関する研究開発を行う非営利団体**)**の下部組織として、**2000**年に設立**
  - DNS、DNSSEC、IPv6**に関する研究開発が主目的**
  - NSD**は** “Name Server Daemon” **の頭文字により命名**
  - **現在は、**NLnet Labs**と**RIPE NCC**により共同開発**
    - 2003**年**2**月より**K.root-servers.net (RIPE NCC**が管理)は、**NSD**により**DNS**サービスを提供**
- DNS**コンテンツサーバの機能のみに特化**
- **高パフォーマンス、シンプル**
  - BIND**の数倍と公称**

比較

- 機能･設計･設定方法
- いくつかの場合における動作の違い
- 「設計ポリシー」にかかわる部分

- 比較対象とするバージョン
  - 原稿執筆時における「正式版」の「最新版」
    - BIND 9: BIND 9.2.3
    - djbdns: djbdns-1.05
    - NSD: NSD 1.2.3

比較する項目

1. DNSコンテンツサーバの起動
   - DNSサーバ起動時の動作の違い
2. 複数のDNSサーバ間におけるデータの同期
   - 複数のDNSコンテンツサーバ間におけるデータ同期の戦略の違い
3. glueの取り扱いの違い
   - glueをサーバ内部でどのように取り扱うかの違い
4. 権限を持たないゾーンに対する問い合わせ
   - 自らが管理権限を持たないゾーンに対する問い合わせを受けた場合の動作の違い

DNSコンテンツサーバの起動

- BIND 9
  - ゾーンファイルはテキストで記述され、起動時にメモリ上に読み込む
  - named.confにゾーンの設定を記述しているため、多くのゾーンを設定すると、起動が遅くなる
- djbdns
  - データファイルは、テキスト形式のデータベースをmakeコマンドでバイナリデータベースに変換する
  - バイナリデータベースは常にディスク上のものが参照され、メモリ上には読み込まれない
  - ゾーンファイルにsyntax errorがある場合、makeがエラーとなり、バイナリデータベースは書き換わらない
- NSD
  - BIND互換のゾーンファイルを、nsdc rebuildコマンドでバイナリデータベースに変換しておき、起動時にはバーナリデータベースをメモリ上に読み込む
  - 複数のゾーンファイルが存在する場合でも、一つのデータベースを生成
  - バイナリデータベースの更新時にはnsdc rebuildしてnsdc reloadする必要がある
  - ゾーンファイルにsyntax errorがある場合、nsdc rebuildがエラーとなり、バイナリデータベースは書き換わらない

結果: DNSコンテンツサーバの起動

- BIND 9: テキストベース、オンメモリ
- djbdns: バイナリデータベース、ディスク上
- NSD: バイナリデータベース、オンメモリ

- NSDは、BINDとdjbdnsを見てからデータベース構造を決めたのではないか、と強く推測される

複数のDNSサーバ間におけるデータの同期 (各ソフトウェアにおける標準的な方法)

- BIND 9
  - NOTIFY+ゾーン転送
  - rndc reloadにより、プライマリの更新とセカンダリの更新が行われる
  - 必要に応じてIXFRを使用
- djbdns
  - バイナリデータベースを直接ssh+rsync等で転送
  - Makefileを編集して設定することにより、makeコマンドで更新するように設定可
  - ゾーン転送機能そのものはサポートされており利用することも可能であるが、推奨されていない(サポートはAXFRのみ)
- NSD
  - NOTIFY+ゾーン転送
  - ただし、ゾーン転送にはBIND 8のnamed-xferコマンドを使用
  - nsdc updateコマンドで行う
    - nsdc rebuild, nsdc reload, nsdc updateの3段階でも可
    - nsdc updateはこれらを必要に応じて適宜呼び出す
  - IXFRはサポートされておらず、今後もしない予定

結果:複数のDNSサーバ間における データの同期

- BIND 9: テキスト読み込み-NOTIFY-IXFR
- djbdns: バイナリデータベースを直接同期
- NSD: コンパイル-バイナリ読み込み-NOTIFY-AXFR

- djbdns: ゾーン転送はデータの同期であるから、外部のよりよいプログラムに任せればよい
  - DJB's Webに記述
- NSD: IXFRのサポートは当面必要ない
  - 付属ドキュメントに記述

glueの取り扱いの違い

- 下記のような委譲があった場合の、glueの取り扱いの違いについて
- (下記は、jpゾーンのオーソリティにおける設定例)

```
example.jp.       IN      NS      ns1.example.jp.
ns1.example.jp.   IN      A       192.168.0.1
```

- この場合に、ns1.example.jpのAを牽かれたら、jpゾーンのネームサーバは何を答える(べき)か?

glueの取り扱いの違い (cont.)

- BIND 9, NSD
  - answer sectionなし(ANCOUNT=0)
  - example.jpのNSをauthority sectionで答え、glue Aとして ns1.example.jpのIPアドレスをadditional sectionで答える
  - つまり、glueでないホストのAを牽かれた場合と同じ
  - glueを持っていても、単独で牽かれた場合には答えない
- djbdns, (BIND 8)
  - ns1.example.jpのAをanswer secionで答える(ANCOUNT=1)
  - そのうえでexample.jpのNSをauthority sectionで答え、glue Aとして ns1.example.jpのIPアドレスをadditional sectionで(も)答える
  - すなわち、glueのみを単独で牽かれた場合、answer sectionで応答を返す

結果: glueの取り扱いの違い

- BIND 9, NSD: glueはglueとしてのみ取り扱い、該当ゾーンがdelegationされている場合、glueを単独で牽かれても答えない
- djbdns, BIND 8: 自分が管理するゾーンの下位ゾーンに属するglueであれば、delegationがあってもglue単独での問い合わせに答える

- データベースの設計の違い
- glueに対する考え方の違い

権限を持たないゾーンに対する問い合わせ

- 例えば、jpのオーソリティに対してexample.comのAを牽いた場合
- BIND 9
  - root zoneのreferralを返す
  - ただし、[a-m].root-servers.netのAは返さない
- djbdns
  - 何も返さない(到着をログに残した上で意識的に捨てる)
- NSD
  - SERVFAILを返す
- (BIND 8)
  - (2004/1/29更新)デフォルト設定時
    - root zoneのreferralを返す
    - additional sectionで[a-m].root-servers.netのAを返す
  - (2004/1/29更新)設定変更により、SERVFAILを返すように設定できる

結果:権限を持たないゾーンに対する問い合わせ

- BIND 9: クライアントに(とりあえず)ヒントとなるreferralを返してみる
  - この動作はRFC1035の下記の記述に基づいていると考えられるが、いまやあまり有益ではない
  - When the resolver processes a user query it asks a known name server for the information; in return, the resolver either receives the desired information or a referral to another name server. Using these referrals, resolvers learn the identities and contents of other name servers.
- BIND 8: ヒントを出した上で、Aも返してみる
  - このAは、ほぼ間違いなく使われない…
- djbdns: 余計なことは一切しない
- NSD: 自分が管理していないものは返せないので、SERVFAILにする
- 参考: Nominum ANSではQUERY REFUSEDを返す模様

- 通常の運用では、このようなパケットが到達することはありえない
- 各実装の設計ポリシーに強く依存

まとめ

「設計ポリシーの違い」が、それぞれの動作の違いとなって現れている

＜まとめ＞

- DNSコンテンツサーバの起動
  - BIND 9: テキストベース、オンメモリ
  - djbdns: バイナリデータベース、ディスク上
  - NSD: バイナリデータベース、オンメモリ
- 複数のDNSサーバ間におけるデータの同期
  - BIND 9: テキスト読み込み-NOTIFY-IXFR
  - djbdns: バイナリデータベースを直接同期
  - NSD: コンパイル-バイナリ読み込み-NOTIFY-AXFR
- glueの取り扱いの違い
  - BIND 9, NSD: glueはglueとしてのみ取り扱う
  - djbdns, BIND 8: 下位ゾーンのAなら、単独でも取り扱う
- 権限を持たないゾーンに対する問い合わせ
  - BIND 9: クライアントに(とりあえず)ヒントを出してみる
  - BIND 8: ヒントを出した上で、Aも教えてみる
  - djbdns: 余計なことは一切しない
  - NSD: 自分が管理していないものは返せないので、SERVFAILにする

参考: DNSキャッシュサーバの場合

- DNSキャッシュサーバについても、このようなトピックスはたくさんある
  - サーバ選択アルゴリズムの違いによる動作の相違
  - ルートサーバの一覧を初期値として指定されたルートサーバから取得しなおすか、しなおさないか
  - 取得する場合、どのルートサーバから取得するか
  - 複数台あるDNSサーバのうちの一部が到達不可になった場合の動作の相違
  - 上位ゾーンにあたるDNSコンテンツサーバから得た応答(例えば、jpゾーンのDNSサーバから得たns1.example.jpのA)を、どのように取り扱うか
  - ネガティブキャッシュのデフォルト値の違いによる動作の相違
  - glueが必要な場合に、AのTTL値だけが0になった場合の動作
- 機会があれば、今後これらの事項についても取り上げたい

参考: その他のDNSサーバの実装 (フリーソフト・オープンソース系)

- DNSコンテンツサーバ+DNSキャッシュサーバ
  - MaraDNS
    - http://www.maradns.org/
- DNSコンテンツサーバのみ
  - PowerDNS
    - http://www.powerdns.com/
  - Posadis
    - http://www.posadis.org/
  - MyDNS
    - http://mydns.bboy.net/
  - Pliant DNS client and server
    - http://pliant.cx/pliant/protocol/dns/
  - Eddie
    - http://eddie.sourceforge.net/lbdns.html
- DNSキャッシュサーバのみ
  - Oak DNS Server
    - http://www.digitallumber.com/oak/

参考: その他のDNSサーバの実装 (プロダクト系)

- Windows Serverにバンドルしているサーバ
  - Windows 2000 Server, Windows 2003 Server
- 大規模用
  - DNSサービス系
    - ANS/CNS (Nominum Inc.)
    - CNR (Cisco Systems Inc.)
  - 広域ロードバランサー系
    - 3-DNS (f5 Networks)
    - ServerIron (Foundry Networks)
  - サービス統合系
    - UltraDNS
- 小規模・中規模用
  - いくつかの家庭用ルータ
    - 「DNSサーバ機能」や「DNSキャッシュ機能」を持っているもの
  - Windowsで動作する商用DNSサーバ
    - 数千円程度のものから、サーバ用の高級品に至るまでいろいろある模様