10'00"00 インターネット10分講座

MIN SEC 1/100S INTERNET

DNSキャッシュ

今回の10分講座では、DNS（Domain Name System）の仕組みを理解するのに必要な、DNSのキャッシュとそれに起因する脆弱性についてお話しします。

◆ DNSのおさらい

まずはじめに、DNSの仕組みについておさらいします。

DNSは、ルートゾーンを起点としたツリー構造を持つ、世界中に存在する多数のサーバが協調しあって動作する分散データベースです。これらのサーバ群にアクセスすることで、ホスト名からIPアドレスを検索したり、メールアドレスから送信先メールサーバを特定したりします。

DNSでは、ある特定のサーバ1台がドメイン名情報をすべて持っているわけではなく、「委任」と呼ばれる仕組みでデータを階層ごとに分散化し、併せてサーバの冗長化を実現しています。

DNSクライアントがデータを得るときは、この委任をルートゾーンから順次たどっていくことで、最終的に必要な情報を得ます。

DNSでは、ドメイン名に関する情報を検索することを「名前解決」と呼んでおり、名前解決で起点となるルートゾーンを管理するサーバは「ルートサーバ」と言います。

ルートゾーンには、COMドメインやJPドメインといった各TLD（トップレベルドメイン）の情報を保持するサーバが、それぞれどのような名前で、どういったIPアドレスが付けられているかなどの情報が保存されています（**図1**）。

そして、例えばCOMドメインの情報を保持するサーバには、同様にexample.comといったドメインの情報を保持するサーバの名前や、IPアドレスの情報が保存されています。なお、これらドメインの情報を保持するDNSサーバを権威サーバと呼びます。

名前解決は、まずルートサーバに知りたいドメイン名を問い合わせることでTLDサーバの情報を得て、次にTLDサーバに問い合わせてさらに次のサーバの情報を得て……と繰り返していき、最終的に目的のデータを持つサーバを特定し問い合わせることで、ドメイン名の情報（リソースレコードと言います）を得ることができます。

図1：ルートゾーンに含まれるCOMに関する情報の例

```
com.                  172800  IN  NS    a.gtld-servers.net.
com.                  172800  IN  NS    b.gtld-servers.net.
com.                  172800  IN  NS    c.gtld-servers.net.
 ⋮                      ⋮     ⋮  ⋮          ⋮
com.                  172800  IN  NS    k.gtld-servers.net.
com.                  172800  IN  NS    l.gtld-servers.net.
com.                  172800  IN  NS    m.gtld-servers.net.
a.gtld-servers.net.   172800  IN  A     192.5.6.30
a.gtld-servers.net.   172800  IN  AAAA  2001:503:a83e:0:0:0:2:30
b.gtld-servers.net.   172800  IN  A     192.33.14.30
b.gtld-servers.net.   172800  IN  AAAA  2001:503:231d:0:0:0:2:30
c.gtld-servers.net.   172800  IN  A     192.26.92.30
d.gtld-servers.net.   172800  IN  A     192.31.80.30
 ⋮                      ⋮     ⋮  ⋮          ⋮
k.gtld-servers.net.   172800  IN  A     192.52.178.30
l.gtld-servers.net.   172800  IN  A     192.41.162.30
m.gtld-servers.net.   172800  IN  A     192.55.83.30
```

◆ 名前解決の流れ

この名前解決について例を用いて紹介します。

ここでは、www.example.jpのIPアドレスを得たいとします（**図2**）。

図2：Webブラウザでwww.example.jpを入力

通常、クライアントPCが直接ルートサーバへ問い合わせることはなく、キャッシュサーバと呼ばれるDNSサーバ（フルリゾルバ）が、クライアントPCの代わりに名前解決を行います（**図3**）。キャッシュについては、この後説明する「DNSのキャッシュ」で詳しく説明します。

a. まずはじめに、クライアントPCからあらかじめ設定されたキャッシュサーバに対して、問い合わせを行います（**図3-a**）。

b. 問い合わせを受けたキャッシュサーバは、ルートサーバへwww.example.jpのIPアドレスについて問い合わせます（**図3-b**）。

c. ルートサーバは、a.dns.jp、b.dns.jp、...、g.dns.jpのいずれかのサーバへ問い合わせるよう回答します（**図3-c**）。

d. キャッシュサーバは、ルートサーバからの回答の中からjpのDNSサーバを一つ選び、そのサーバへ再び問い合わせます（**図3-d**）。

e. jpのDNSサーバはexample.jpのDNSサーバの情報を回答します（**図3-e**）。

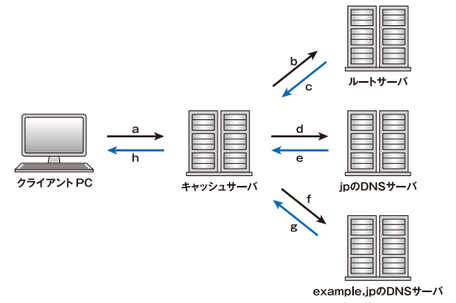
f. キャッシュサーバは同様にして回答の中からexample.jpのDNSサーバを一つ選び、そのサーバへ問い合わせます（**図3-f**）。

g. example.jpのDNSサーバは、キャッシュサーバにIPアドレスの情報を回答します（**図3-g**）。

h. キャッシュサーバは、クライアントPCにIPアドレスの情報を回答します（**図3-h**）。

クライアントPCは、www.example.jpのIPアドレスを、このようにして得ることができます。

図3：キャッシュサーバによる名前解決例



◆ DNSのキャッシュ

DNSの名前解決は以上のようにして行われますが、毎回この動作が繰り返されるわけではありません。

普通に考えると、検索の起点となるルートサーバには、検索のたびに問い合わせがされることになりますが、そうするとDNSの名前解決を行う世界中のクライアントPCから、ルートサーバへ大量のクエリ（問い合わせ）がきます。また、クライアントPCから見た場合、名前解決のたびに多数のサーバに対して問い合わせを行い、その回答を待つことになり時間がかかってしまいます。

そうしたことを軽減するため、DNSではキャッシュと呼ばれる仕組みによって、手順を簡略化することができるようになっています。

例えば、クライアントPCがすでにwww.example.jpのIPアドレス情報を得ている場合、それを再利用することで名前解決をせず、すぐにIPアドレスを利用することができます（**図4**）。

また、キャッシュサーバもその名前の通り、名前解決で行った問い合わせの結果を保存しておき、同様の問い合わせの際に再利用することができます（**図5**）。

前述の名前解決の流れで示した、各問い合わせの結果をローカルに保存しておき、後で同じ問い合わせがあった場合にその内容を再利用することで、再度問い合わせを行わずに済むようにすることができるのです。

図4：クライアントPCからキャッシュサーバへ問い合わせずに、キャッシュを利用

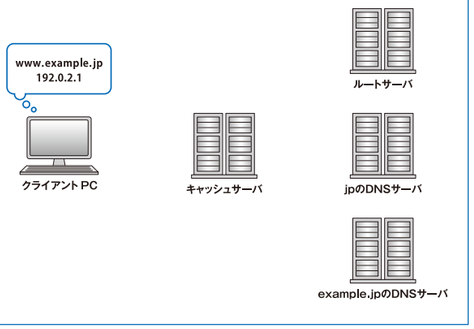
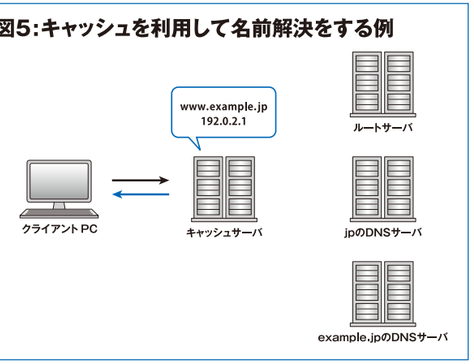


図5：キャッシュを利用して名前解決をする例

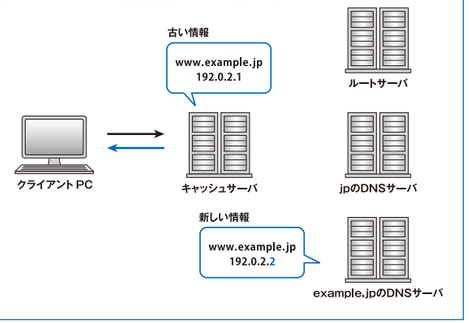


◆ キャッシュと生存期間（TTL）

キャッシュとして保存されたデータは、そのままずっと使われるわけではありません。なぜなら、DNSは最初に述べた通り分散データベースであり、そのデータは任意のタイミングで変更される可能性があるためです。

データが書き換わったにもかかわらず、クライアントPCやキャッシュサーバが手元に残したキャッシュをずっと使用していると、実際の状態と不整合が起きることになります（図6）。

図6：データが書き換わった様子



そのため、DNSでは「どのくらいの期間までキャッシュとして利用してよいか」という、TTL（Time To Live）と呼ばれるパラメータが、それぞれのデータ（レコード）に設定されています。**図7**を例に挙げると、図中の「172800」という数字がそれであり、これは「取得してから172,800秒（48時間）の間、キャッシュとして利用してよい」という意味になります。この期間を過ぎた場合は、このデータ（レコード）をキャッシュから破棄することが求められます。

図7：NSレコードのTTLが172,800秒に設定されている例

com.	172800	IN	NS	a.gtld-servers.net.

◆ レコードの書き換えとキャッシュ

TTLの仕組みにより、定期的にキャッシュが更新されることになっていますが、タイミングによっては古い情報を参照してしまう場合があるため、レコードの書き換えの際に戸惑ってしまう人が見受けられます。

例えば、あるWebサーバのIPアドレスを変更するため、DNSに設定されているWebサーバを示すIPアドレス設定を書き換えたいとします。ここでは、Webサーバには IPv4アドレスが使われているとします。その場合には、IPv4アドレスを示すAレコードを変更することになります（**図8**）。

あるタイミングでこのAレコードを書き換えたとしましょう。この瞬間からWebサーバへのアクセスは変更後のIPv4アドレスへ行われることが期待されますが、実際には上で述べたキャッシュの仕組みが働くため、キャッシュとして保存されたレコードのTTLで指定された時間が経過するまで、古いIPv4アドレスへアクセスされ続けることになります。そのため、「アドレスを変更したのになぜか古いサーバへアクセスがある」といった事象が起こります（**図9**）。

こうした状況は、レコードを変更する前に、TTLを十分短くしておくことで軽減することができます。TTLが短ければ、それだけ早くキャッシュが破棄されることになり、その結果、新しいデータが参照されやすくなるためです（**図10**）。

図8：新旧WebサーバとDNSレコード

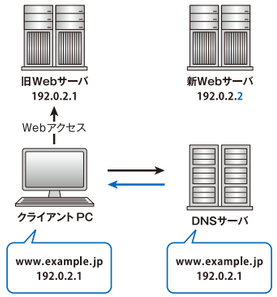


図9：新旧Webサーバとアクセスの状況

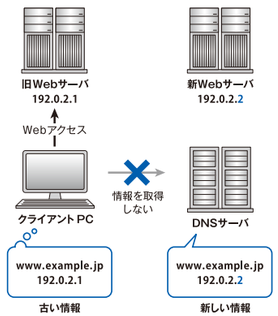
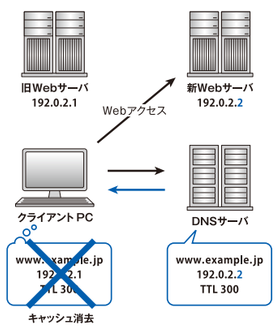


図10：TTLを短くしたときの新旧Webサーバとアクセスの状況



◆ DNSキャッシュの脆弱性

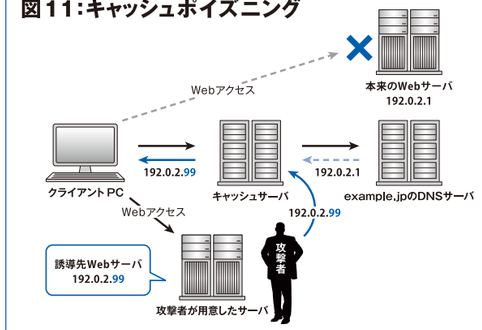
ここまで、DNSの名前解決およびキャッシュの仕組みについて紹介してきました。DNSは、インターネットにおいて重要なサービスの一つですが、それだけに悪用された場合、被害が大きくなる可能性があります。

ここからはキャッシュに関連する、二つのDNSの脆弱性についてご紹介します。

⑴キャッシュポイズニング

前述したように、DNSの問い合わせ結果などは、キャッシュサーバやクライアントPCなどにキャッシュとして一時的に保存されます。このキャッシュを何らかの方法で意図的に変更することで、名前解決をできないようにしたり、本来のデータとは違う内容のデータを回答させ、悪意のあるサイトへ誘導するなどといった手法があり、これをキャッシュポイズニングといいます（**図11**）。

図11：キャッシュポイズニング



通常、DNSの問い合わせと回答は、通信に使われるポート番号やIDと呼ばれる番号が適切であるかどうかによって、その内容が正しいかどうかを判断していますが、この番号を詐称することで、偽の内容をキャッシュさせる手法があります。従来、偽装データをキャッシュサーバに保存させるのは比較的困難だと思われていましたが、2008年8月、セキュリティ研究者のDan Kaminsky氏によって容易に行える攻撃方法が発表され、その対応が求められることになりました。現在では、いくつかの対策によって攻撃のリスクは軽減されていますが、十分ではないと考えられています。そのため、より強固な対応策として、DNSSEC（Domain Name System Security Extensions）と呼ばれる公開鍵暗号方式を用いた技術の導入が進んでいます。

JPNICニュースレター No.43
インターネット10分講座「DNSSEC」
http://www.nic.ad.jp/ja/newsletter/No43/0800.html

⑵ghost domain names

また、キャッシュの仕組みに対する脆弱性として、“ghost domain names（幽霊ドメイン名）”と呼ばれるものがあります。これは、2012年2月に清華大学のHaixin Duan氏らによって発表されたもので、ある条件では、保持するキャッシュのレコードを上書きするかどうかの判定が実装依存になるため、強制的にレコードを保持させ続けることができてしまうといった脆弱性です。

例えば、ghost-domain.example.comというドメインと、その権威サーバdns.example.comがあったとします。脆弱性を持つキャッシュサーバに対して、この権威サーバのIPアドレスを問い合わせます。そうすると、キャッシュサーバは以下のような情報をキャッシュします。

ghost-domain.example.com.	86400	NS	dns.example.com.
dns.example.com.	86400	A	192.0.2.1

時間が経過するにつれ、キャッシュのTTLは減少していきます。

ghost-domain.example.com.	43200	NS	dns.example.com.
dns.example.com.	43200	A	192.0.2.1

ここで、ghost-domain.example.comの権威サーバの名前を、IPアドレスは同じままで名前だけ違うもの（例えばdns-new.example.com）に変更し、その後で先ほど使用したキャッシュサーバに対して、変更後のサーバ名のIPアドレスについて問い合わせます。この脆弱性を持つキャッシュサーバは、レコードが変更されたものとしてキャッシュを上書きします。そして、同時にキャッシュの生存期間を上書きしてしまい、その結果、元のTTLの値を越えてレコードをキャッシュし続けることになります。

ghost-domain.example.com.	86400	NS	dns-new.example.com.
dns-new.example.com.	86400	A	192.0.2.1

この脆弱性は、レジストリが何らかの理由でドメイン名を登録抹消もしくは変更したとしても、そのドメイン名を利用し続けられるといったことができるものです。この現象は、キャッシュを更新する際、特定の条件下ではどのように処理するべきかが明確に規定されておらず※1実装依存となるため、こうした動作を行うキャッシュサーバが実際にあることから、脆弱性として発表されました。

この脆弱性に対しては、キャッシュサーバで利用しているサーバソフトウェアの更新などで回避することが可能です。

◆ 最後に

DNSは、ドメイン名とIPアドレスの対応付けや、メールの配送先の指定など、インターネットの重要な基盤サービスの一つであり、その動作を理解することは、インターネットの安定利用に役立ちます。今回は基本的な仕組みについてご紹介しましたが、DNSSECなどDNSに対する新しい技術は日々導入されています。そのため、JPNICでは今後もDNSに関する情報を皆さんにご紹介していく予定です。

（JPNIC 技術部 小山祐司）

※1 RFC2181“Clarifications to the DNS Specification”
http://www.ietf.org/rfc/rfc2181.txt

参考:

DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION
http://www.ietf.org/rfc/rfc1035.txt

Clarifications to the DNS Specification
http://www.ietf.org/rfc/rfc2181.txt

US-CERT Vulnerability Note VU#800113
- Multiple DNS implementations vulnerable to cache poisoning
http://www.kb.cert.org/vuls/id/800113

Ghost Domain Names:Revoked Yet Still Resolvable | Internet Society
http://www.internetsociety.org/ghost-domain-names-revoked-yet-still-resolvable

CVE - CVE-2012-1033
https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-1033