キャッシュポイズニング攻撃対策：
キャッシュDNSサーバー運用者向け―基本対策編

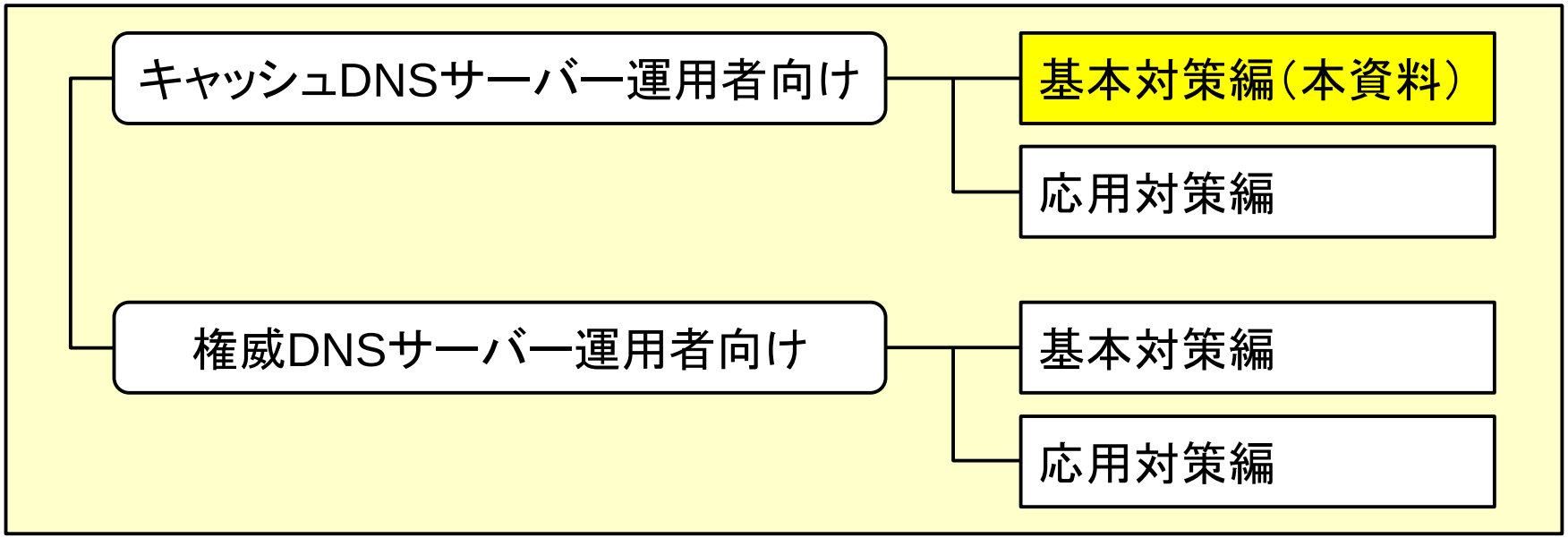
初版作成：2014年4月30日

最終更新：2014年4月30日

株式会社日本レジストリサービス（JPRS）

本資料の位置づけ

- 本資料は以下の四部構成の資料の一部
  - 対象者ごとに、キャッシュDNSサーバー運用者向けと権威DNSサーバー運用者向けに大別
  - それぞれを、基本対策編と応用対策編の二部で構成



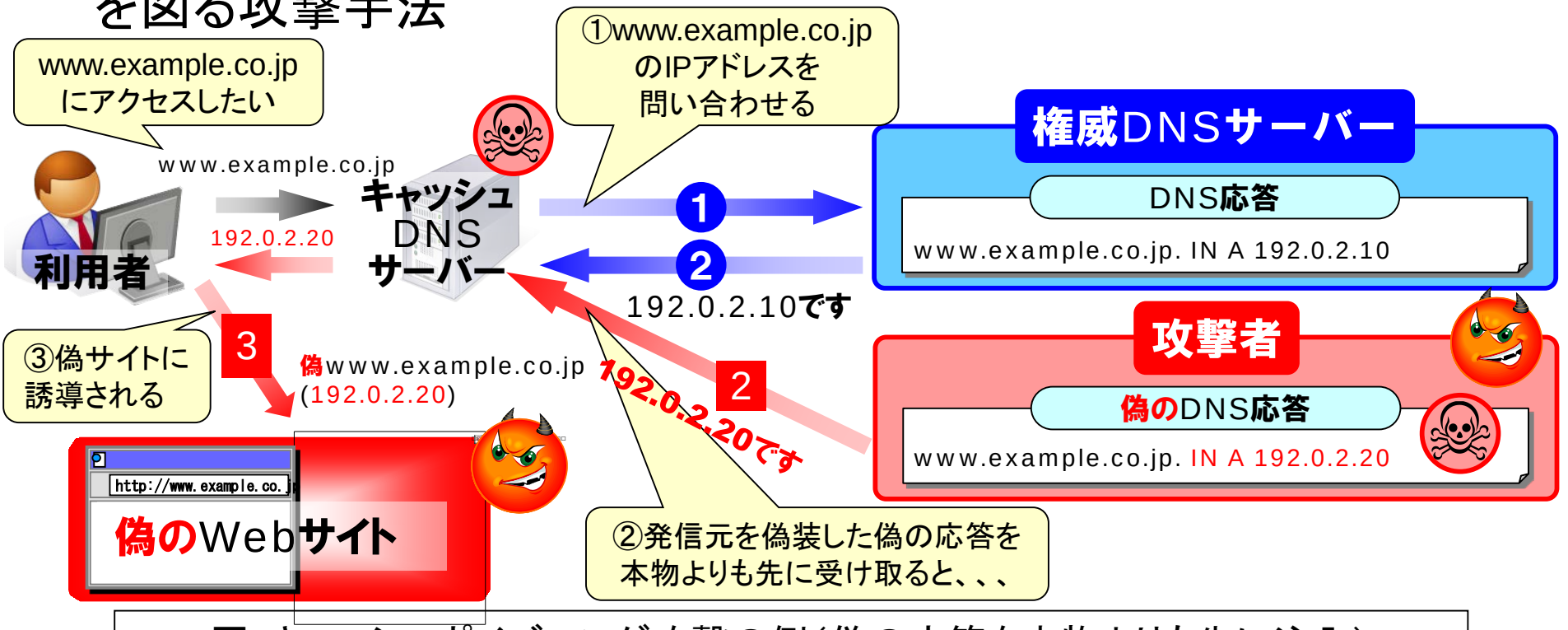
本資料の内容（基本対策編）

- 本資料ではキャッシュDNSサーバー運用者向けの基本対策編として、以下の項目について解説
  - おさらい（1）：キャッシュポイズニング攻撃の概要
  - おさらい（2）：攻撃対策の基本
  - UDP問い合わせポートのランダム化（ソースポートランダマイゼーション）の実施
  - オープンリゾルバー対策
  - 攻撃の検知と対応

おさらい（1）：
キャッシュポイズニング攻撃の概要

キャッシュポイズニング攻撃とは？

- 偽のDNS応答をキャッシュDNSサーバーにキャッシュさせることでアクセスを偽サイトに誘導し、フィッシングや電子メールの盗難などを図る攻撃手法



図：キャッシュポイズニング攻撃の例（偽の応答を本物よりも先に注入）
（注：上記は例であり、偽のDNS応答をキャッシュさせる方法は他にも存在する）

キャッシュポイズニングの基本

- コンセプト：偽の応答をキャッシュさせる
  - これまでにいくつかの方法が発表・実行されている
- 対策：TTL（キャッシュの有効期限）による保護
  - キャッシュ済のデータは問い合わせない
    - 問い合わせない ⇒ 攻撃の機会がない
  - TTLを長く設定することで、攻撃の機会を低減可能
    - 攻撃実行後、TTLが満了するまで同じ名前を再攻撃不可
  - この特性により、同じ名前に対する連続攻撃を防止

しかし、2008年にこの対策を突破する攻撃手法が公開された

カミンスキー型攻撃手法の出現

- 2008年7月に、Dan Kaminsky氏が発表
- 存在しない名前への問い合わせを攻撃に用い、その応答に付随する情報により毒の注入を図る
- 存在しない名前への問い合わせを用いることで、TTLによる保護を無力化、連続攻撃を可能にした
  - つまり、カミンスキーは攻撃者に「効率の良い武器」を提示したといえる

実社会における例え：火縄銃の時代（一度攻撃した後、しばらくの間再攻撃できない）に、カミンスキーは機関銃の作り方を提示した。

おさらい(2):攻撃対策の基本

キャッシュポイズニングが成功するまで

1. 攻撃者が攻撃を仕掛ける
   – 攻撃が成立（偽の応答が先に到達）した場合・・・

   ⇒ 偽の応答の注入に成功する（第一関門突破）
2. キャッシュDNSサーバーが応答をチェックする
   – チェックをパスしてしまった場合・・・

   ⇒ 注入した偽の応答がキャッシュされる（第二関門突破）
3. キャッシュDNSサーバーが名前解決をする
   – キャッシュされた応答が有効であった場合・・・

   ⇒ キャッシュされた偽の応答が使われる（第三関門突破）
4. キャッシュポイズニング攻撃が成功する

攻撃対策の基本(三つの対策)

- 対策の基本:それぞれの関門を突破されない(されにくい)ようにする
- それぞれの関門(再掲)
  1. 偽の応答の注入に成功する(第一関門)
  2. 注入した偽の応答がキャッシュされる(第二関門)
  3. キャッシュされた偽の応答が使われる(第三関門)
- 対応する三つの対策
  ① 偽の応答を注入されない(されにくくなる)ようにする
  ② 受け取った応答のチェックを厳重にする
  ③ 攻撃を検知して対応する

攻撃対策の基本(三つの対策)

- 三つの対策(再掲):

  ① 偽の応答を注入されない(されにくくなる)ようにする
  ② 受け取った応答のチェックを厳重にする
  ③ 攻撃を検知して対応する

- 各対策が上記のどの対策であり、どのような効果があるのかをきちんと理解することが重要
- 具体例:
  - ソースポートランダマイゼーションの実施…①の一つ
  - Unboundにおけるharden-referral-pathオプションの設定(応用対策編で説明)…②の一つ
  - 問い合わせ／応答パケット数の照合…③の一つ

ソースポートランダマイゼーションの実施

① 偽の応答を注入されない（されにくくなる）ようにする対策の一つ

おさらい：偽の応答の注入成功の条件

① 偽の応答を本物よりも先に到達させられること

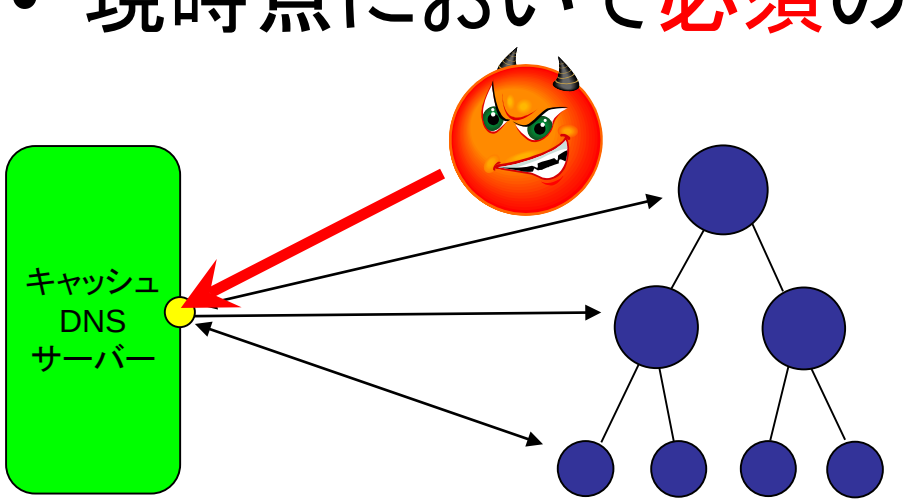
② 以下のすべてが本物と一致していること

- IPアドレス（送信元・送信先）
  …偽装可能（あらかじめわかっている）
- 送信先ポート番号（送信元ポート番号は常に53）
- 問い合わせた名前（QNAME）・型（QTYPE）
  …偽装可能（あらかじめわかっている）
- 問い合わせID（TXID）…総当たり可能

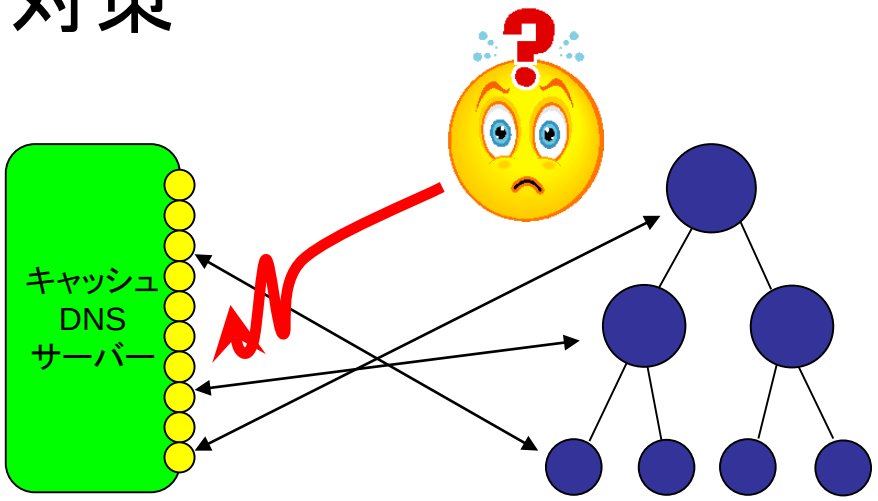
TXIDは16ビットであり、TXIDのみを変化させた多数の応答を同時に送りつける、総当たり攻撃が可能（カミンスキー型攻撃手法で採用）

ソースポートランダマイゼーション

- 問い合わせ送信時のポート番号をランダムに変化させ、応答が到達するポート番号を一致させにくくする
  - 一致しなければ、偽の応答の注入は成立しない
- 現時点において必須の対策



同じポートを毎回使用（偽の応答を注入しやすい）



多数のポートをランダムに使用（偽の応答を注入しにくい）

ソースポートランダマイゼーションの効果

- 偽の応答の注入が成功する確率を下げる

あるキャッシュDNSサーバーに対し、
偽の応答の注入が成功する確率

$$P_S = \frac{R \times W}{N \times Port \times ID}$$

R: 攻撃対象1台当たりに送るパケット量（pps）

W: 攻撃可能な時間（問い合わせ⇒応答のRTT）

N: 攻撃対象レコードを保持する権威DNSサーバーの数

$Port$: 問い合わせに使うソースポートの数

ID: 問い合わせID（TXID：16ビット = 65,536）

ソースポートランダマイゼーションにより、上記数式の$Port$の値が増える

ソースポートランダマイゼーションの効果

- 偽の応答の注入の確率を下げる対策であり、攻撃を不可能にする対策ではないことに注意
  - 最大で1/(2 ^ 16) = 1/65,536
  - 攻撃成功までの時間を対策前の65,536倍に延ばす
- 攻撃の検知・対応と併せて実施するのがより効果的

BIND 9/Unboundにおけるサポート状況

- 最新版のBIND 9/Unboundでは、
  ソースポートランダマイゼーションを標準サポート
- 設定ファイルの不適切な内容に注意
  - BIND 9：以下の設定が有効になっていないことを確認

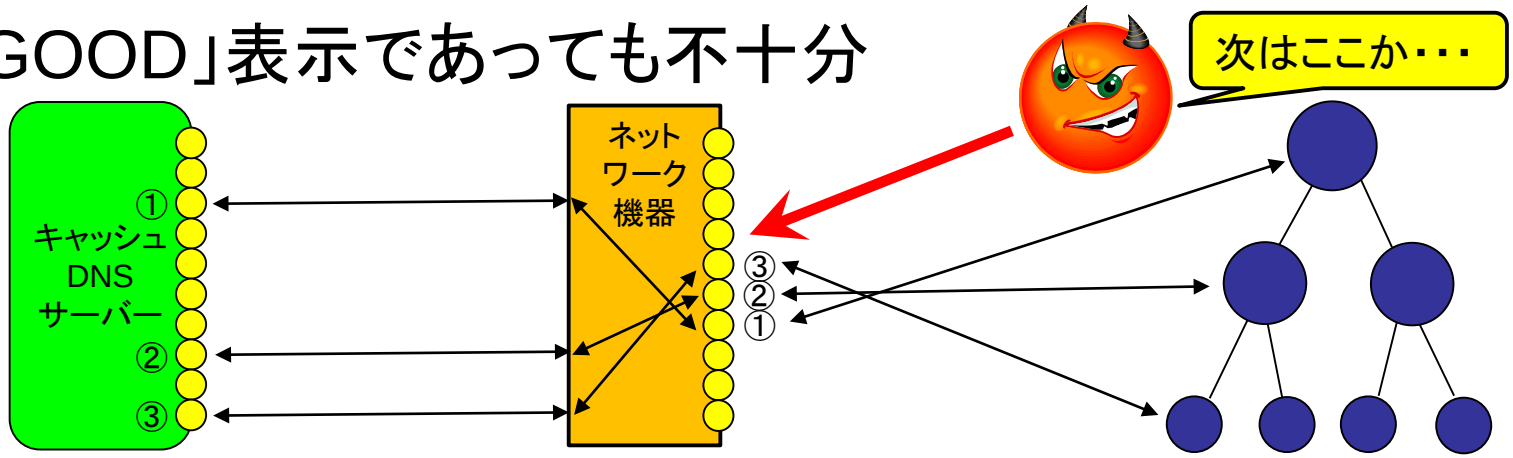
```
query-source port 53;
query-source-v6 port 53;
```

ポートを意図的に固定する不適切な設定

    - 有効であった場合、削除/コメントアウトしてnamedを再起動
  - Unbound：ポートを意図的に固定できるが、設定方法がやや複雑であり通常は出回っていない
    - 実運用には不適切な設定であるため方法の紹介は省略

注意事項：不適切なポートの再変換

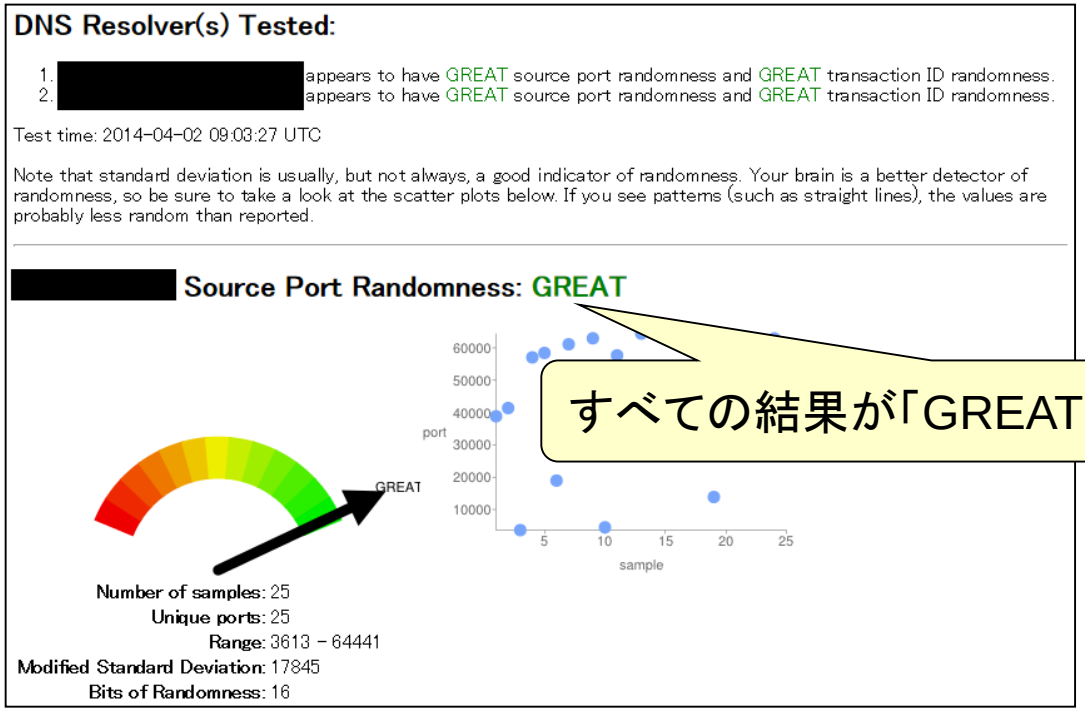
- ネットワーク機器（ファイアーウォールなど）における、不適切なポートの再変換に注意
  - 予測できる形に再変換されてしまう場合がある（図）
  - この場合、機器の設定変更や更新などが必要になる
- 該当する場合、次ページで紹介するDNS-OARCの確認サイトで「POOR」または「GOOD」と表示される
  - 「GOOD」表示であっても不十分



ネットワーク機器における不適切なポートの再変換の例（順番に並べ替え）

設定の確認方法（Webページ）

- DNS-OARCが確認のためのWebページを提供
  - Web-based DNS Randomness Test
    <http://entropy.dns-oarc.net/>
- 実行例：



設定の確認方法（digコマンド）

- digコマンドでも確認可能

  ```
  $ dig +short porttest.dns-oarc.net TXT
  ```

- 実行例：

```
$ dig +short porttest.dns-oarc.net TXT
porttest.y.x.w.v.u.t.s.r.q.p.o.n.m.l.k.j.i.h.g.f.e.d.c.b.a.pt.dns-oarc.net.
“xxx.xxx.xxx.xxx is GREAT: 26 queries in 2.9 seconds from 26 ports with std dev 17554"
```

結果が「GREAT」であることを確認

オープンリゾルバー対策

① 偽の応答を注入されない（されにくくなる）ようにする
対策の一つ

オープンリゾルバーの危険性

- 攻撃対象がオープンリゾルバーである場合、攻撃者が外部からそのサーバーに問い合わせを送ることで、名前検索を始めさせることができる
- 名前検索に合わせてキャッシュポイズニング攻撃を仕掛けることにより、成功確率を上げられる
  - 攻撃者が問い合わせと偽の応答を同時に送信

偽の応答を本物よりも先に到達させられる確率が上がる

オープンリゾルバーの危険性

- キャッシュポイズニング攻撃のリスク軽減の観点からも、オープンリゾルバーの修正は必須

キャッシュポイズニングでは、
自組織の利用者や顧客が直接危険にさらされる

オープンリゾルバーの設定修正

- オープンリゾルバーでなくすことにより、外部からの攻撃のリスクを下げられる
  - 組織/ISP内からの攻撃抑制の効果はない
  - オープンリゾルバーでない場合も、Webページへのアクセス誘導や電子メールの送りつけなどにより、内部からの名前検索を誘発させる攻撃手法もある
- ソースポートランダマイゼーションと同様、攻撃の検知・対応と併せて実施することがより効果的

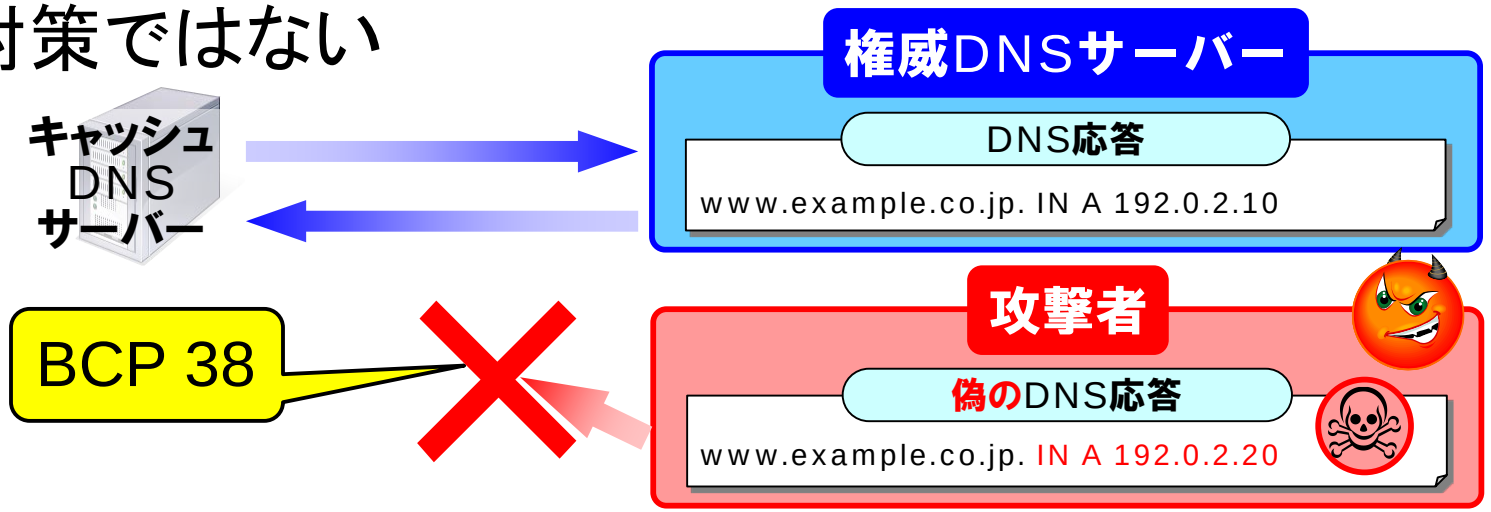
BIND 9における設定確認

- 最新版のBIND 9では、オープンリゾルバーとならないようにデフォルト値が設定されている
  - 古いBIND 9を使っている場合、バージョンアップすることでオープンリゾルバーでなくすことができる
- アクセスコントロールの設定が適切であることを確認
- 設定例:

```
// 組織内ネットワーク一覧をACLで定義
acl MYNET {
        192.0.2.0/24;
        2001:db8:2::/64;
};
options {
        // 内部からの問い合わせのみ受け付ける
        allow-query { MYNET; };
        allow-recursion { MYNET; };
        allow-query-cache { MYNET; };
};
```

送信元検証（BCP 38）の導入

- BCP 38の導入により、送信元を偽装したDNS応答をインターネットに送信できなくすることも大切
  - そのネットワークからのキャッシュポイズニング攻撃が不可能になる
  - 自分が偽の応答を注入されないようにするための対策ではない

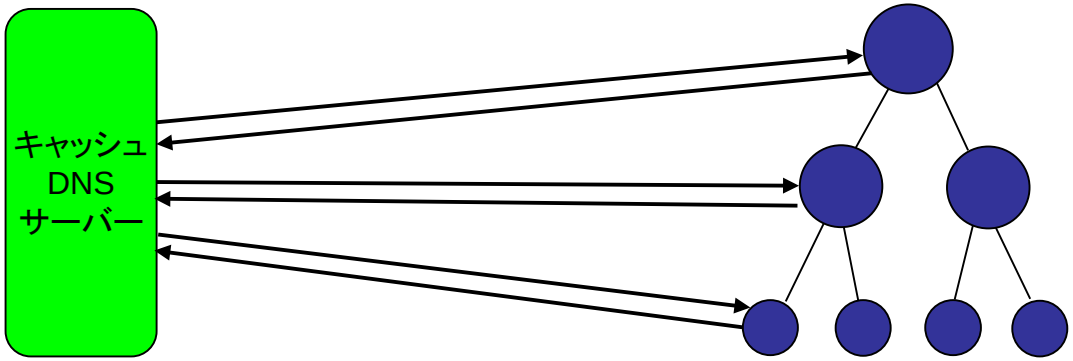


攻撃の検知と対応

③ 攻撃を検知して対応する
対策の一つ

攻撃の検知（DNSパケットの数）

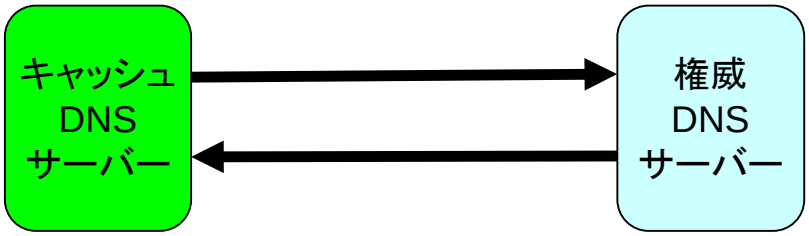
- DNSでは通常、UDPの問い合わせパケットと応答パケットの数は一致（一対一対応）している
  - IPフラグメンテーションが発生した場合や、応答がなかった場合などを除く



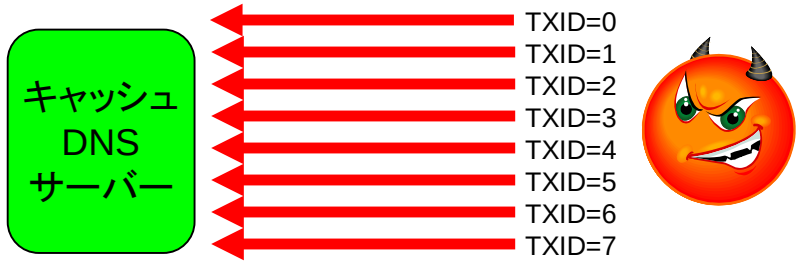
DNSの問い合わせパケットと応答パケットは一対一対応

攻撃の検知（DNSパケットの数）

- キャッシュポイズニング攻撃では、攻撃時に問い合わせID（TXID）のみを変化させたDNS応答が到達
  - 総当たり型の攻撃では、多数のDNS応答が連続して到達
- この特性を利用し、キャッシュDNSサーバー側でDNSパケットの状況をチェックすることにより、攻撃の検知が可能



通常：問い合わせと応答が一対一対応



攻撃：TXIDのみを変化させた多数のDNS応答が連続して到達

攻撃の検知（DNSパケットの数）

- 典型的な攻撃パターン
  - 問い合わせパケットの数に比べ、応答パケットの数が多い状態が続いている
  - 単位時間あたりの応答パケットの数が急増している
- 検出方法の例
  - MRTGなどによる継続的な観測や、異常（攻撃パターン）検出時の管理者への通知

攻撃の検知
（キャッシュDNSサーバーの負荷）

- 攻撃による大量のDNS問い合わせ・応答により、キャッシュDNSサーバーの負荷が上昇する
  - 特に、カミンスキー型攻撃手法では存在しない（キャッシュされていない）名前の問い合わせが高頻度で到達するため、通常の攻撃よりも負荷上昇が大きい
- キャッシュDNSサーバーの負荷を監視することで攻撃の検知が可能
  - キャッシュポイズニング攻撃に限らず、DoS攻撃など他の攻撃の検知にも有効

攻撃の検知（DNS応答の内容）

- カミンスキー型攻撃手法の場合、攻撃パケットの内容が特徴的
  - $(random).攻撃対象ドメイン名に対する大量の応答
    - 攻撃対象ドメイン名にランダム文字列を加えたサブドメイン
- これにより、攻撃対象のドメイン名をある程度判定可能

対応例（キャッシュの照合・クリア）

- 攻撃対象ドメイン名が判明した場合、他のキャッシュDNSサーバーとの応答内容照合や、必要に応じたキャッシュのクリアなどが有効
- キャッシュクリアにより、キャッシュポイズニング攻撃が成立しやすくなる場合もあることに注意
  - キャッシュクリアの直後に名前検索を実行して、正当な応答をキャッシュしておくとよい

Unboundの unwanted-reply-thresholdオプション

- Unboundではunwanted-reply-thresholdオプションを有効にすることで不審なDNS応答を検知、警告ログの出力とキャッシュクリアが行われる
  - 設定値：検知の閾（しきい）値
    - デフォルトは0（機能無効）
  - Unboundのマニュアルでは10 million（1000万）を推奨
- 不審なDNS応答：
  送信元IPアドレスやTXIDの不一致
- 警告ログの例：

```
unwanted reply total reached threshold (閾値) you may be under attack.
defensive action: clearing the cache
```

対応例(攻撃流入元の調査)

- 攻撃パケットの流入元を調べることで、どのネットワーク(方面)から到達しているかを調査可能
  - 対応:相手先や上流ネットワーク管理者への連絡など
- 発信元IPアドレスは偽装されていることに注意

参考リンク

- （緊急）キャッシュポイズニング攻撃の危険性増加に伴う
  DNSサーバーの設定再確認について（2014年4月15日公開）
  <http://jprs.jp/tech/security/2014-04-15-portrandomization.html>
- JPRS トピックス&コラム No.020
  DNSの安全性・安定性向上のためのキホン
  ～お使いのDNSサーバーは大丈夫ですか？～
  <http://jprs.jp/related-info/guide/020.pdf>

更新履歴

- 2014年4月30日　　初版作成