# .jprs DDoS 攻撃対策実証実験
# （DNS 運用組織間の連携・緩和策の検証）
# 実証実験報告書

第 1.0 版

2021 年 11 月

株式会社日本レジストリサービス

北海道総合通信網株式会社

北陸通信ネットワーク株式会社

ソフトバンク株式会社

フリービット株式会社

株式会社コミュニティネットワークセンター

株式会社オプテージ

株式会社エネルギア・コミュニケーションズ

株式会社 QTnet

沖縄通信ネットワーク株式会社

# 目次

# 概要

## 実証実験の背景・目的

オリンピック・パラリンピックは国際的な競技大会であり、注目度が高いことからサイバー攻撃の格好の対象となっている。特に、2012 年に開催されたロンドン大会以降、オリンピック・パラリンピックを標的としたサイバー攻撃が増加・大規模化しており、具体的な被害事例も報告されている。2020 年に予定され、新型コロナウイルス感染症の拡大により 2021 年に延期された東京大会においても、わが国の関連サイトを標的とした様々なサイバー攻撃の対象となるリスクが指摘されている。

DNS(Domain Name System)は、インターネットに必要不可欠な基盤技術の一つである。DNS を標的としたサイバー攻撃、特に分散サービス妨害攻撃(Distributed Denial of Service Attacks、以降 DDoS 攻撃)によって DNS サービスが停止した場合、インターネットの利用やサービスの提供に致命的な影響が及ぶ。そのため、そうしたサイバー攻撃による被害を未然に防ぎ、被害の拡大を抑制・緩和するための準備・対策を進める必要がある。

そのための手段の一つとして、最近の DNS ソフトウェアには、DDoS 攻撃を検知・防御・緩和するためのいくつかの機能が実装されている。これらの機能を活用することで、DNS を標的としたサイバー攻撃の影響を効果的に抑制・緩和することが期待できる。

2020 年時点において、主要なオープンソースの DNS ソフトウェアに実装されている DDoS 攻撃対策の機能として、以下の 4 つが挙げられる。これらの機能には攻撃発生時に攻撃トラフィックをフィルタまたは制限する、期限切れのキャッシュ情報を破棄せずに名前解決を継続するといった内容が含まれており、これらを有効活用することで、サイバー攻撃の効果を抑制・緩和することが可能になる。

1. fetch-limit, rate-limit
   - https://kb.isc.org/docs/aa-01304
   - フルリゾルバーから権威 DNS サーバーへの問い合わせ回数を制限する機能
2. serve-stale
   - https://www.rfc-editor.org/rfc/rfc8767.html
   - TTL が満了してもキャッシュを破棄せず、名前解決を継続する機能
3. NSEC/NSEC3 Aggressive Use
   - https://www.rfc-editor.org/rfc/rfc8198.html
   - DNSSEC の不存在証明を活用し、不要なクエリ送信を抑制する機能

4. DNS Cookies
    - https://www.rfc-editor.org/rfc/rfc7873.html
    - DNS パケットの偽造を検知する機能

DNS を標的としたサイバー攻撃を検知するためには、DNS サーバーソフトウェアにおいて統計情報を取得し、利用状況を把握することが有効である。従来、こうした統計情報は各ソフトウェアにおいて独自の形式で取得されており、それぞれに対応した統計処理・加工が必要であった。この問題を解決し、統計情報の取得やその処理・加工のための標準的な機能を提供する、dnstap の実装が主要な DNS ソフトウェアにおいて進められている。

5. dnstap
    - https://dnstap.info/
    - DNS のトラフィック情報を出力する機能
        - ✓ DDoS 攻撃を検知する仕組みに応用可能

また、DNS は、情報を提供する権威 DNS サーバーと情報を検索するフルリゾルバーが連携して動作している。そのため、権威 DNS サーバーのオペレーターとフルリゾルバーのオペレーターが連携して対処することが、DNS を標的としたサイバー攻撃の対策として効果的である。特に、多数のドメイン名が登録されている TLD の DNS オペレーターと、多数の利用者を有する ISP の DNS オペレーターが連携することで、DNS を標的としたサイバー攻撃に対し、より迅速な対処が可能になる。

以上の状況を踏まえ、DNS を標的としたサイバー攻撃への対処と攻撃発生時の連携・情報共有の方法について、国内の ISP9 社と JPRS による「.jprs DDoS 攻撃対策実証実験（DNS 運用組織間の連携・緩和策の検証）（以下、本研究）」を実施した。本研究では以下の 2 つのテーマにわけてそれぞれ実験を行った。

6. テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験
7. テーマ 2: DDoS 攻撃の防御・緩和実験

本研究のプラットフォームとしてインターネットに関する研究・開発を目的としたトップレベルドメイン(Top Level Domain、以降 TLD)である「.jprs」を用い、DNS を標的とした DDoS 攻撃に対処するための実証実験を実施した。実証実験を通じ、主要なオープンソースの DNS ソフトウェアに実装されている DDoS 攻撃対策のための各種機能を評価し、知見として得られた機能の限界や解決すべき課題について、各 DNS ソフトウェアベンダーへのフィードバックを実施した。

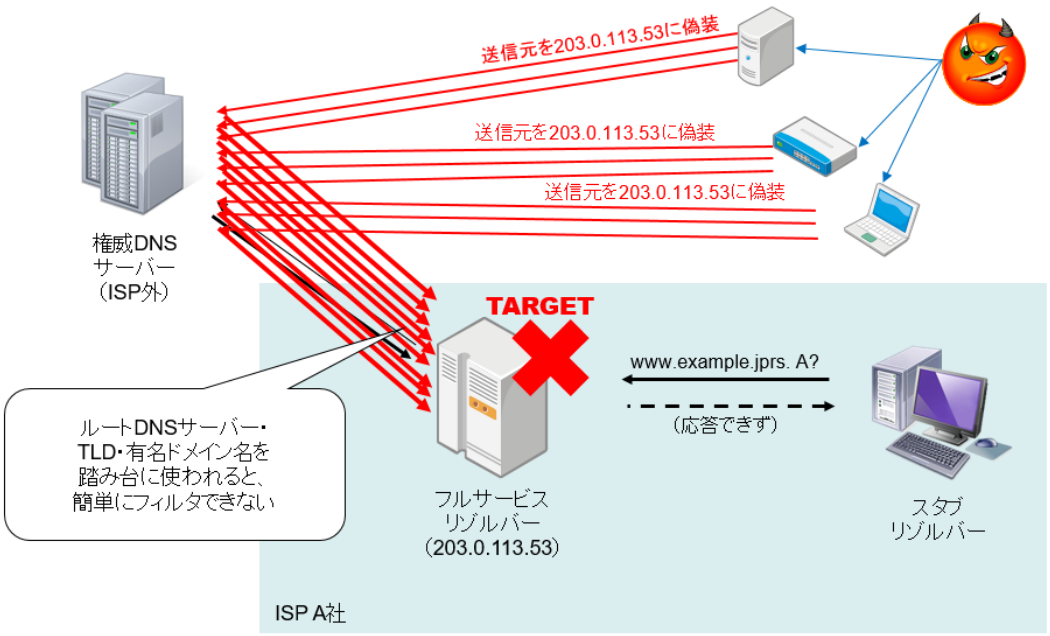
## 想定した攻撃手法

本研究では、サイバー攻撃の手法の 1 つである DDoS 攻撃を想定し、以下の 2 つの攻撃手法に対応するための実証実験（以下、本実証実験）を実施した。

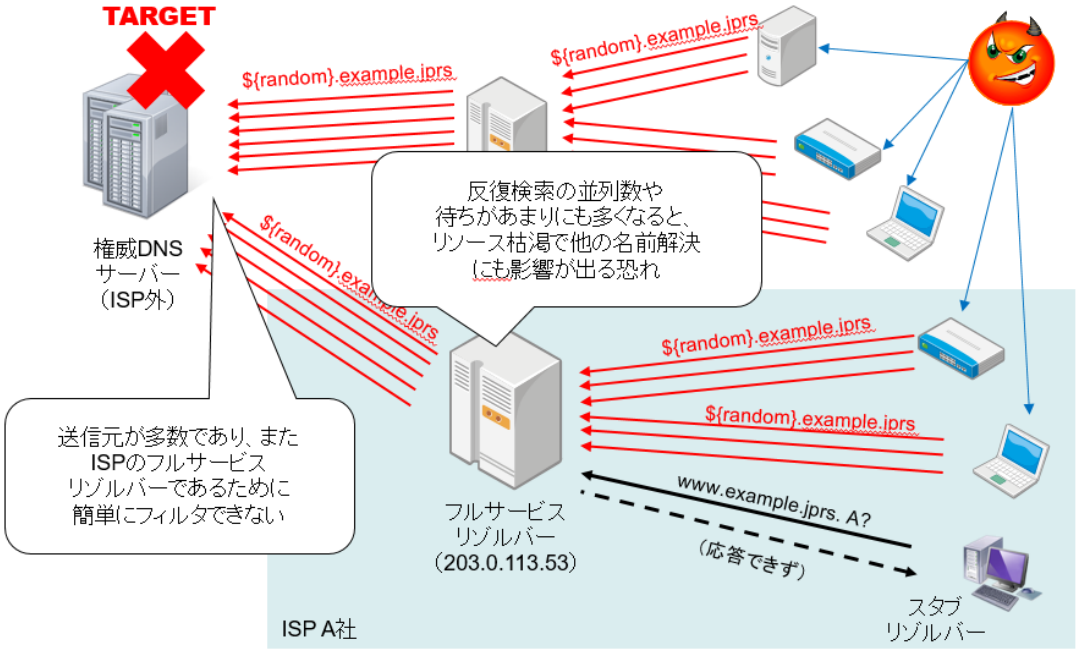
- ランダムサブドメイン攻撃
- DNS Amp 攻撃

これらの攻撃手法は以前から継続して報告されており、本実証実験の実施時点においても、一定頻度で発生し続けていることが確認されている。当該攻撃手法は DNS のプロトコル仕様を利用したものであることからさまざまな抑制・緩和策を継続して実施する必要があり、前述した対策機能の効果的な利用、攻撃発生時の早期発見・関係する組織間における、効果的な連携が求められている。

攻撃シナリオA: 権威DNSサーバーを踏み台にした攻撃（DNS Amp攻撃）




攻撃シナリオB: ランダムサブドメイン攻撃



## 検証内容に関する特記事項

- 本実証実験では攻撃トラフィックを実際に送信するため、運用中の権威 DNS サーバーやフルリゾルバーが接続されているネットワークをインターネットから実際に切断することは運用中のインターネットサービスに影響を及ぼす恐れがある。そこで本実証実験では機器の設定変更や擬似的な切断を発生させることで、検証を実施することとした。なお、その方法は当該組織の上位ネットワークへの接続形態に依存することから各 ISP と協議の上、適切な方法で検証することとした。

## 本実証実験における制約事項

- 本実証実験におけるプラットフォームとして使用した.jprs は、ICANN と JPRS の間で締結されたレジストリ契約により、レジストリオペレーターである JPRS が委任を受けている TLD である。レジストリ契約には.jprs TLD の権威 DNS サーバーのサービスレベルに関する遵守事項(SLA)が含まれるため、それに抵触する状況を故意に発生させる、例えば、.jprs TLD そのものを長時間にわたってインターネットから切断するといった実験は実施不可能である。
- 各 ISP に設置したフルリゾルバー、及び JPRS が運用する TLD DNS サーバーは共に、インターネットに接続されている。本実証実験ではこれらのサーバーに対し、インターネット経由で DDoS 攻撃をシミュレーションした実トラフィックを生成・送出するため、ISP や JPRS が運用中の既存のサービスに影響しないように、生成・送出する攻撃トラフィックのボリュームを制限するなどの配慮が必要になる。

参考：ICANN による新 gTLD DNS サーバーの SLA

| 機能 | 項目 | 月間のサービスレベル |
|---|---|---|
| DNS 名前解決機能 | TCP DNS 応答の RTT | の 95%以上が＜1500 ミリ秒 |
| | UDP DNS 応答の RTT | の 95%以上が＜500 ミリ秒 |
| | TCP,UDP DNS 応答の有無 | 月間停止時間の合計＜432 分(可用性 99%) |
| ゾーン転送機能 | 差分更新反映時間 | 観測値の 95%以上が＜15 分で転送完了 |
| | 全量更新反映時間 | 観測値の 95%以上が＜120 分で転送完了 |

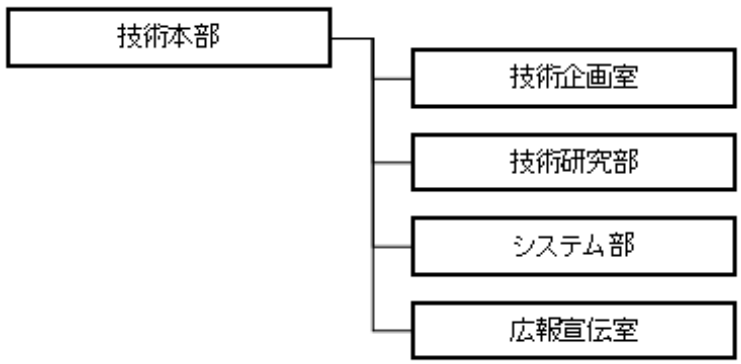
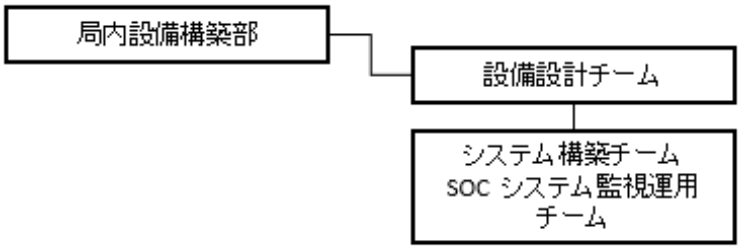
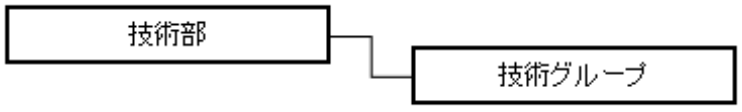
## 実証実験の体制

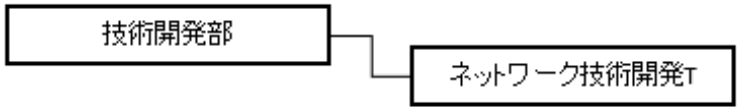
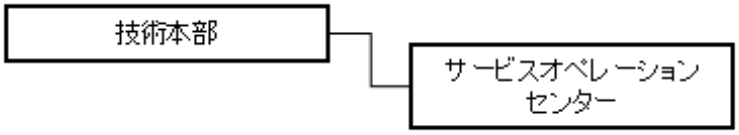
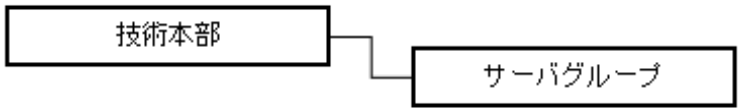
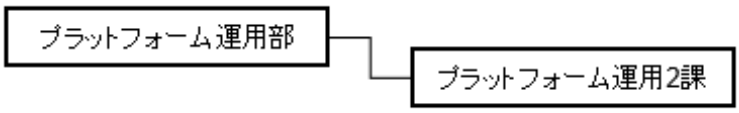
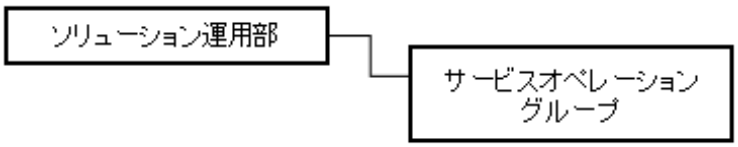
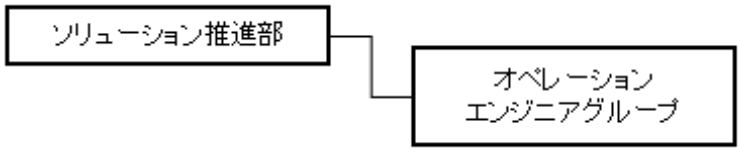
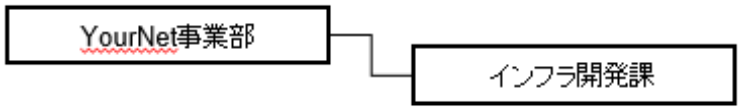
(1) 実証実験実施組織

本研究には、JPRS 及び国内の ISP9 社（五十音順）が参加した。

- 株式会社日本レジストリサービス(JPRS)
- 株式会社エネルギア・コミュニケーションズ(エネコム)
- 沖縄通信ネットワーク株式会社(OTNet)
- 株式会社オプテージ(OPTAGE)
- 株式会社 QTnet(QTnet)
- 株式会社コミュニティネットワークセンター(CNCI)
- ソフトバンク株式会社
- 北海道総合通信網株式会社(HOTnet)
- 北陸通信ネットワーク株式会社(HTNet)
- フリービット株式会社(FreeBit)

(2) 実証実験管理体制

| 組織名 | 体制図 |
|---|---|
| 株式会社日本レジストリサービス(JPRS) |  |
| 株式会社エネルギア・コミュニケーションズ(エネコム) |  |
| 沖縄通信ネットワーク株式会社(OTNet) |  |

| | |
|---|---|
| 株式会社オプテージ<br>(OPTAGE) |  |
| 株式会社 QTnet(QTnet) |  |
| 株式会社コミュニティネットワークセンター<br>(CNCI) |  |
| ソフトバンク株式会社 |  |
| 北海道総合通信網株式会社(HOTnet) |  |
| 北陸通信ネットワーク株式会社(HTNet) |  |
| フリービット株式会社<br>(FreeBit) |  |

# 実証実験報告（株式会社日本レジストリサービス）

## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### DDoS 攻撃の検知

DNS に対するサイバー攻撃は、大きく以下の 2 種類に分類される。

1. DNS そのものを対象とした攻撃
2. DNS の特性を利用し、第三者を対象とした攻撃

本テーマでは、前者の「DNS そのものを対象とした攻撃」のうち、ランダムサブドメイン攻撃の検知と、権威 DNS サーバーとフルリゾルバーのオペレーターの連携による適切な対処を目的としている。

ランダムサブドメイン攻撃は攻撃対象の権威 DNS サーバーにクエリを集中させることで、攻撃対象の権威 DNS サーバーやフルリゾルバーをサービス不能の状態に陥らせることを狙った攻撃手法である。2014 年頃から攻撃の発生が世界的に報告されており、複数の被害事例も確認されている。

ランダムサブドメイン攻撃では、攻撃対象の権威 DNS サーバーがホストするドメイン名（例：example.jp や example.co.jp)にランダムなサブドメインを付加した攻撃トラフィックを送信する。具体的には、ランダムな文字列を該当ドメイン名の 3LD や 4LD に付加したものになる（例：{ランダムな文字列}.example.jp）。攻撃者はあらかじめ用意した世界中にあるセキュリティの設定に問題がある端末を乗っ取り(ボット化)、それらの端末から攻撃トラフィックを送信する。乗っ取った大量の端末から攻撃トラフィックを送信することで、権威 DNS サーバーがサービス提供不能の状態になるように仕向ける。

ランダムサブドメイン攻撃では、乗っ取った端末(ボット)は ISP のフルリゾルバーへ攻撃トラフィックを送信する。通常、同一の QNAME（例：www.example.jp 等）についてはフルリゾルバーのキャッシュ機能により、キャッシュが有効である間は権威 DNS サーバーへの問い合わせは発生しない。しかし、ランダムサブドメイン攻撃は、ランダムな文字列を含む QNAME のクエリを攻撃トラフィックとして送信しているためフルリゾルバーのキャッシュ機能が働かず、権威 DNS サーバーへの問い合わせが毎回発生する。

このように、ランダムサブドメイン攻撃ではフルリゾルバーのキャッシュ機能を迂回させ、かつボットではなく正当なフルリゾルバーからターゲットとなる権威 DNS サーバーへ攻撃トラフィックが送信されるため、権威 DNS サーバー側では対処が難しい攻撃手法の一つである。

こうした、多数のフルリゾルバーを踏み台にし、権威 DNS サーバーを対象としたランダムサブドメイン攻撃を検知する箇所として、

1. 攻撃対象の権威 DNS サーバー
2. フルリゾルバー
3. 攻撃対象のドメイン名の親や祖先のゾーンの権威 DNS サーバー

の 3 つが考えられる。

1.と 2. については、攻撃クエリを直接受けることから検知が可能である。また、3. についてはフルリゾルバーに攻撃対象のドメイン名の委任情報がキャッシュされていない状態において、攻撃対象のドメイン名にランダムなサブドメインが付与されたドメイン名のクエリの到達を確認することで、検知が可能である。

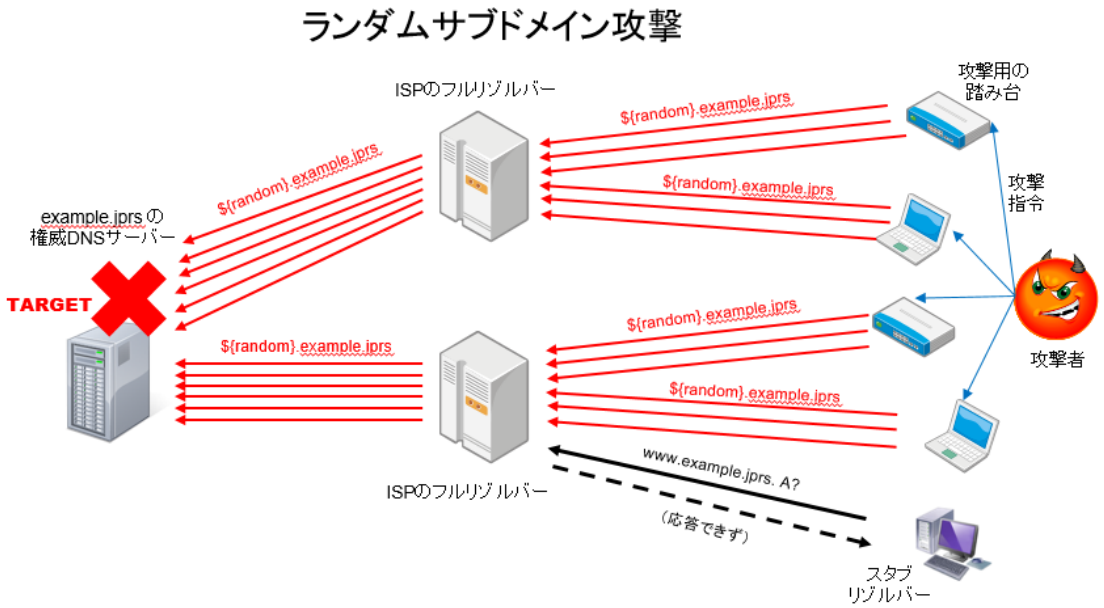



**図: ランダムサブドメイン攻撃の概要図**

以下、3.において攻撃の検知が可能になる理由について記述する。

ドメイン名空間は階層構造となっている。フルリゾルバーは、名前解決の要求元となるスタブリゾルバーから名前解決要求を受け取った際、必要な情報をキャッシュしていない場合、ドメイン名空間の起点となる情報(ルートヒント情報)から反復検索を開始する。

DNS の名前解決では反復検索の際、解決したい名前(QNAME)、及び解決したい資源レコードタイプ(QTYPE)はスタブリゾルバーから受け取ったものと同じ情報を、各階層の権威 DNS サーバーへ送信する。つまり、攻撃と同じクエリ、すなわちランダムなサブドメインを付加した QNAME のクエリは攻撃対象の権威 DNS サーバ

ーに加え、その親や祖先のゾーンの権威 DNS サーバーにも送信されることになる。そのため、当該クエリの到達を確認することで、攻撃の検知が可能になる。

なお、フルリゾルバーにおいて QNAME minimisation（RFC 7816）が有効にされている場合、親や祖先のゾーンの権威 DNS サーバーには解決したい名前に替えて子ゾーンの NS レコードを問い合わせるため、3.による検知が困難になる。

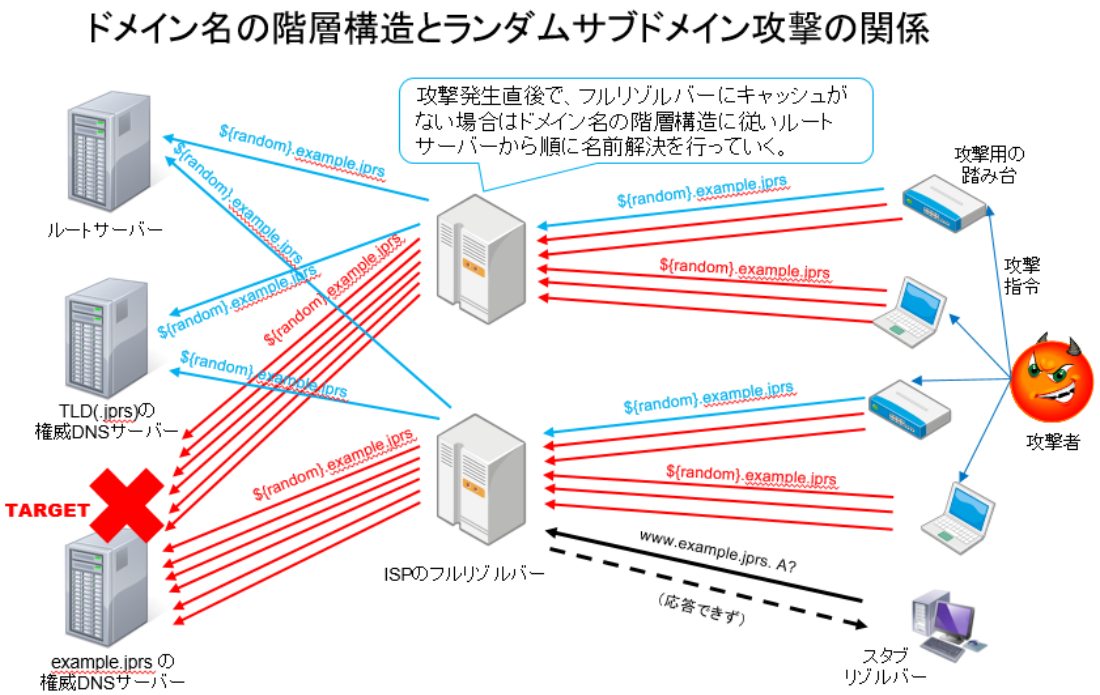



**図:ドメイン名の階層構造とランダムサブドメイン攻撃の関係**

フルリゾルバーは、親ゾーンの権威 DNS サーバーから当該ドメイン名の委任情報を受け取るとその情報の TTL が満了するまでキャッシュし、そのキャッシュが有効である間はその委任情報を使用するクエリが再度到達しても、親ゾーンの権威 DNS サーバーに問い合わせることはない。そのため、親ゾーンの権威 DNS サーバーは、当該フルリゾルバーに到達した最初の攻撃、または委任情報のキャッシュの TTL が満了して無効になった状態で到達した攻撃に由来するクエリのみを受けることとなる。

このことから、親ゾーンの権威 DNS サーバーに到達する攻撃に由来するクエリは相対的に少量であり、1.及び 2.と比べ、より効果的に攻撃を検知可能となることが期待できる。

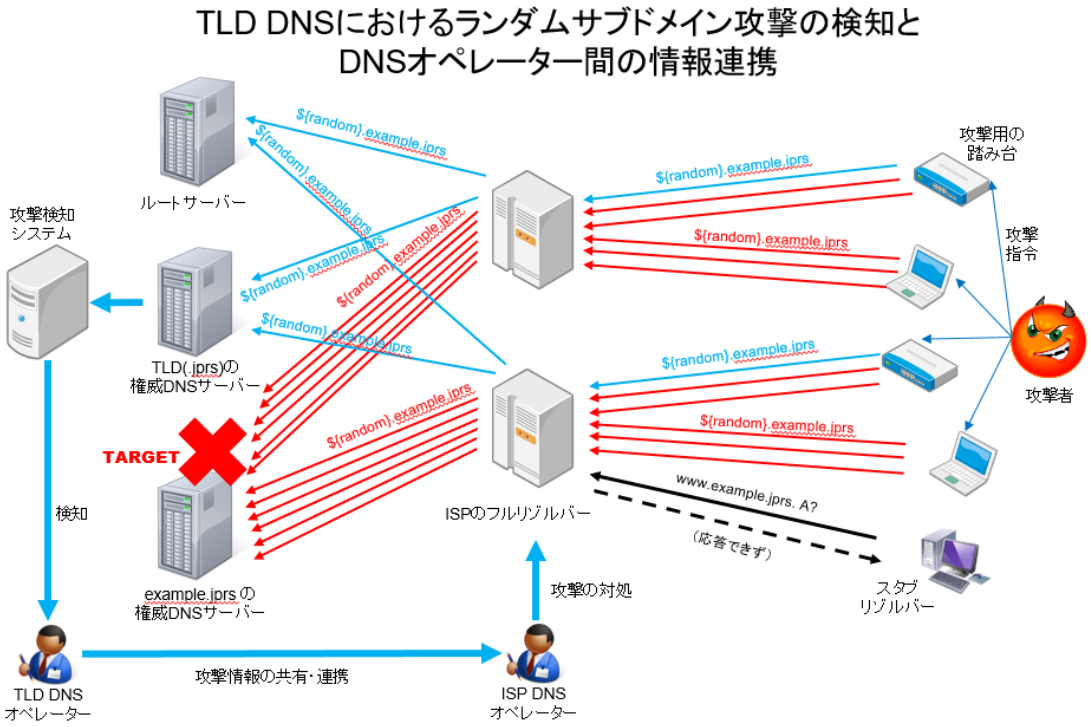



**図:TLD DNS におけるランダムサブドメイン攻撃の検知と DNS オペレーター間の情報連携**

そのため、本実証実験では、親ゾーンの権威 DNS サーバー側でランダムサブドメイン攻撃を検知・対処することを想定し、検証を実施することとした。検証には JPRS が研究用のトップレベルドメインとして管理している「.jprs」を使用した。

また、DNS は複数の構成要素が連携して動作しているため、DNS に対する攻撃を検知して適切に対処するためには、それらの構成要素を管理する関係者が円滑に連携し、速やかに対処を進める必要がある。そこで本実証実験では、攻撃検知時の情報共有手段・情報共有内容・対処のための連携フローについても、併せて検証することとした。具体的には、権威 DNS サーバーのオペレーターである JPRS と、フルリゾルバーのオペレーターである ISP 各社がどのような情報を連携し、どのような方法で連携することが適切であるかを、実証実験を通して考察することとした。

## 具体的な実装

本実証実験では、ランダムサブドメイン攻撃のクエリを捉えるために、親ゾーンの権威 DNS サーバーでクエリ情報を取得し、取得した情報を解析環境に送信して検知する構成とした。なお、権威 DNS サーバーが IP Anycast 技術やロードバランサーなどによって複数台のサーバーで構成されている場合、構成する各サーバーにクエリが分散され、ランダムサブドメイン攻撃の兆候を捉えることが難しくな

る場合がある。そのため、本実証実験ではクエリ情報を一か所に集約するための解析環境を別途用意した。

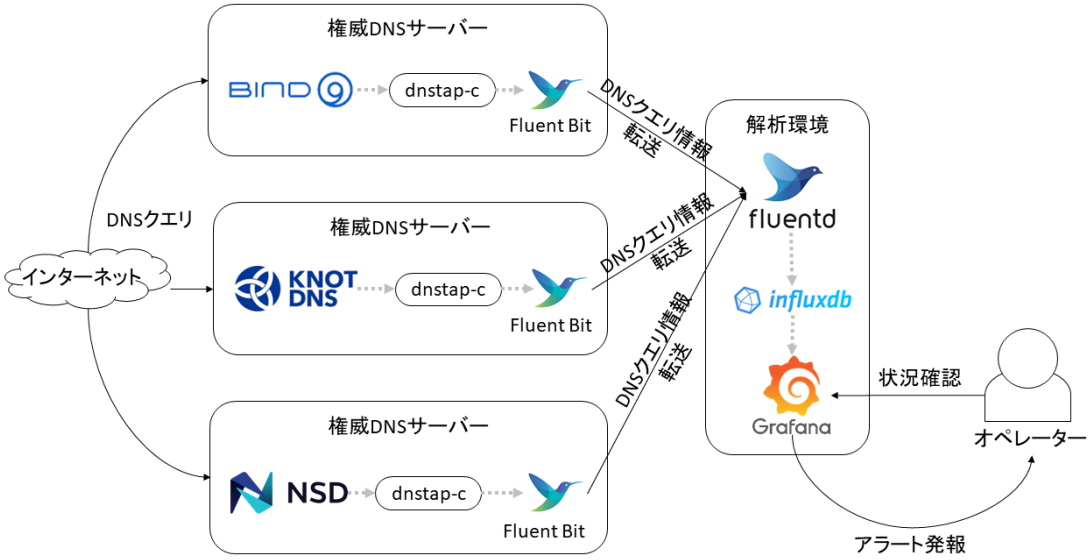



**図: システム構成図**

権威 DNS サーバーでクエリ情報を取得する技術として、dnstap を採用した。dnstap は BIND 9、NSD、Knot DNS など、複数のオープンソース DNS ソフトウェアに実装されているため、クエリ情報のフォーマットを異なる実装間で統一できる。

dnstap により取得されるデータは構造化されたバイナリフォーマットであり、そのままでは解析環境への転送には不向きである。この問題を解決するため、dnstap のデータフォーマットを解析し、プレーンテキスト・JSON・YAML 等の別のフォーマットに変換するためのツールが開発され、公開されている。代表的なツールを以下に示す。

表: dnstap の代表的なツール

| # | ツール名 | 公開サイト |
|---|---|---|
| 1 | golang-dnstap | https://github.com/dnstap/golang-dnstap |
| 2 | dnstap-read | https://github.com/isc-projects/bind9 (BIND 9 に付属) |
| 3 | dnstap-ldns | https://github.com/dnstap/dnstap-ldns |

本実証実験にあたり、これらのツールの機能を検証した結果、いずれのツールも別のフォーマットに変換する際、クエリに含まれるすべての内容が出力されないため、攻撃の検知に使う際に情報が不足する可能性があることが判明した。

また、本実証実験では後述するデータ収集ソフトウェアや時系列データベースソフトウェアにおいてデータを扱いやすいよう JSON フォーマットで出力することを想定していたが、上記のツールのうち、実験実施時点で JSON フォーマット出力に対応していたのは golang-dnstap のみであった。しかし、検証の結果、当該ツールは実測値で 5,000 クエリ/秒程度しかクエリを処理できず、パフォーマンス上の要件を満たしていないことが判明した。

そのため、本実証実験では dnstap を用いてクエリ情報を解析するツールを新たに開発することとした。本ツールは JPRS の阿波連良尚が開発し、dnstap-c と命名した。dnstap-c は JSON での出力とともに、クエリの送信元アドレスから AS 番号を割り出して出力情報に付加する機能も有している。

dnstap で取得したデータは、オープンソースのデータ収集ソフトウェアである Fluentd 及び Fluent Bit で各権威 DNS サーバーから解析環境に転送することとした。Fluentd 及び Fluent Bit はデータをほぼリアルタイムに転送できるため、迅速な解析及び攻撃の検知が可能になる。

解析環境では、時系列データベースソフトウェアである InfluxDB を使ってクエリ情報を格納し、解析・視覚化・攻撃検知はダッシュボードツールである Grafana を利用した。Grafana は閾値に基づいてアラートを発報する機能を有しており、その機能を利用し、オペレーターへの通知が可能である。

## ランダムサブドメイン攻撃検知の計算式と閾値

ここでは、本実証実験において使用した攻撃検知の計算式と閾値について記述する。

権威 DNS サーバーでランダムサブドメイン攻撃を検知する際に利用可能な手法として、以下の 3 つが挙げられる。

1. 受信したクエリパターンの分析
2. 送信元 IP アドレスをキーとした単位時間あたりのクエリ量の分析
3. QNAME をキーとした単位時間あたりのクエリ量の分析

1.は、ランダムサブドメイン攻撃のクエリパターンが攻撃対象のドメイン名にランダムなサブドメインを付与したドメイン名であることに注目し、受信したクエリのパターンを分析することで、攻撃の検知を図る手法である。2.と 3.はいずれも、

単位時間あたりのクエリ量を分析することで、攻撃の検知を図る手法である。2.ではクエリの送信元 IP アドレス、3.ではクエリの QNAME を分析のキーとしている。

これらはいずれも、有力なパラメーターである。しかし、権威 DNS サーバーのサービス対象範囲はインターネット全体であり、多種多様なクエリを受信する可能性があること、権威 DNS サーバーに到達するクエリ量はインターネットの利用状況により随時変化すること、大規模 ISP のフルリゾルバーやパブリック DNS サービスなど、平常時から大量の問い合わせを権威 DNS サーバーに送信するインターネットサービスも存在することなどから、権威 DNS サーバー側でサイバー攻撃を効果的に検知するためには、これらをはじめとする複数のパラメーターや複数のアルゴリズムを組み合わせた、多面的な分析が必要になる。

そこで、本実証実験では複数のパラメーター・アルゴリズムを組み合わせた分析のための前段階として、クエリの QNAME のユニーク数と送信元 IP アドレスのユニーク数を組み合わせた簡易な数式を用いて分析を模擬し、攻撃を検知することとした。具体的には、下記の計算式を適用した。

**(単位時間あたりの) QNAME の 3LD 以降のユニーク数 / 送信元 IP アドレスのユニーク数**

ランダムサブドメイン攻撃のクエリでは、QNAME の右から 2 ラベル目は攻撃対象となるドメイン名で固定、右から 3 ラベル目以降はランダム文字列となるはずである。攻撃が発生した場合、到着するクエリの 3LD 以降は大量のランダム文字列となるため、上記の計算式の分子が急速に増加することが予測される。一方、TLD の権威 DNS サーバーに到達するクエリの送信元アドレスは基本的にフルリゾルバーの IP アドレスとなるため、上記の計算式の分母は、攻撃に利用されるフルリゾルバーの数となることが予測される。

本計算式を用いた場合の閾値の設定では、その権威 DNS サーバーに登録されているドメイン名の数を TTL で割った値が目安となりうる。例えば、JP ドメイン名の場合 2020 年 10 月現在の登録数はおよそ 160 万件であり、委任情報の TTL は 86400 であることから、

**1,600,000 / 86,400 = 18.51**

となる。この値は、1 台のフルリゾルバーがすべての JP ドメイン名を問い合わせた場合における、1 秒当たりのクエリ数の平均値とみなすことができる。もちろん、存在しない JP ドメイン名に対するクエリや、子ゾーン側の NS レコードによるフルリゾルバーのキャッシュの上書きの影響も考慮する必要があるため、実際のフルリゾルバーではこの数値とは異なる閾値を採用する必要がありうる。しかし、設定すべき初期値の目安として、この値は有効であると考えられる。

## ランダムサブドメイン攻撃の検知実験

本実証実験のシナリオとして、連携フローが異なる 2 つのパターンを準備した。

以下、それらのシナリオについて記述する。いずれも ISP の顧客の PC がボット化して攻撃トラフィックを発生させることを想定し、その検知から対処までを模したシナリオとなっている。

シナリオ 1

Step. 1 ISP 側で疑似攻撃生成ツールを用い、tld4.nic.jprs に 50 クエリ/秒の攻撃を発生させる

Step. 2 解析環境で攻撃を検知し、TLD の権威 DNS サーバーのオペレーター(JPRS)にメール/Slack で通知が送信される

Step. 3 JPRS は解析環境でトラフィック状況を確認する

Step. 4 ランダムサブドメイン攻撃が発生していることを確認し、攻撃発生元となっている ISP へメール/Slack でその旨を通知する

Step. 5 通知を受けた ISP は解析環境へアクセスし、攻撃の詳細情報を調査する

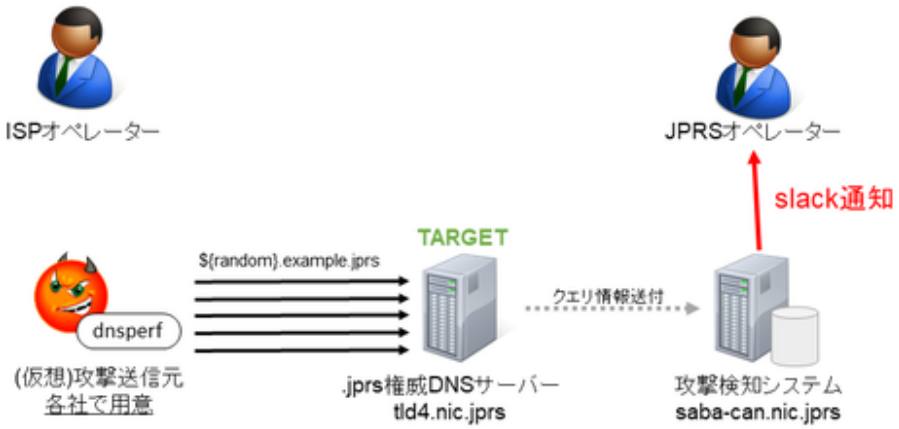
Step. 6 攻撃の特定及び対処ができたと想定し、疑似攻撃生成ツールを停止する

Step. 7 ISP 及び JPRS は解析環境へアクセスし、攻撃が停止していることを確認する

# 実験シナリオ1（1）

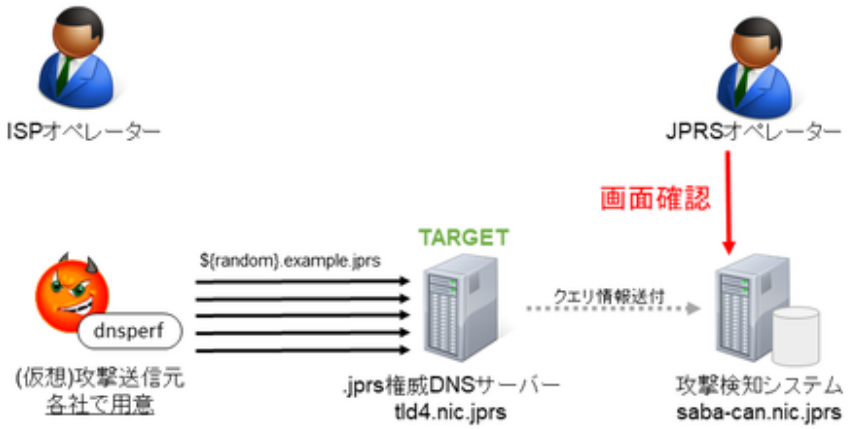

各社の(仮想)攻撃送信元からdnsperfでtld4.nic.jprsに向かって50qps程度のランダムサブドメイン攻撃を発生させる

# 実験シナリオ1（2）



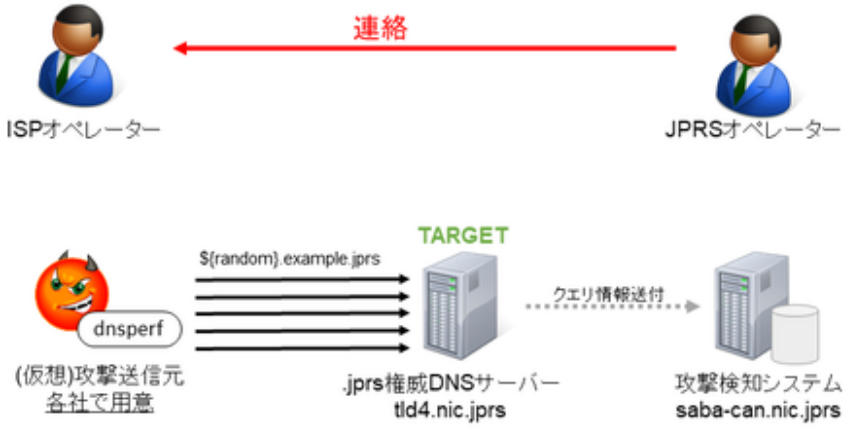

攻撃検知システムからJPRSオペレーターへアラート通知が飛ぶ

# 実験シナリオ1（3）



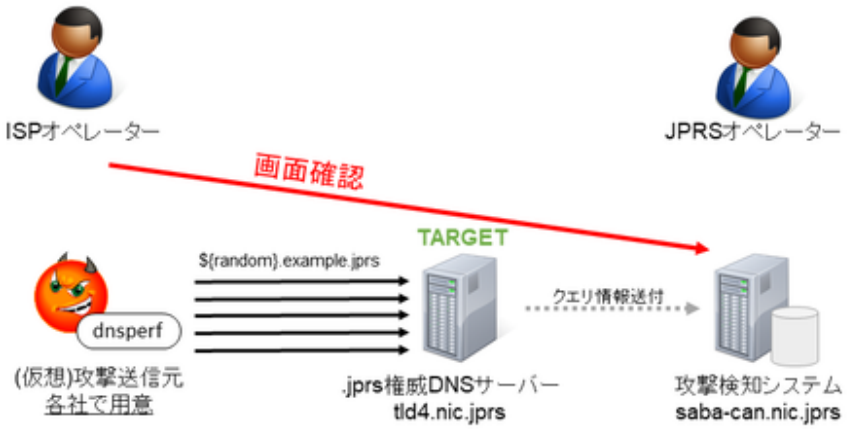

JPRSオペレーターは攻撃検知システムの画面を開き、
どのISPから攻撃がきているかを確認する

# 実験シナリオ1（4）



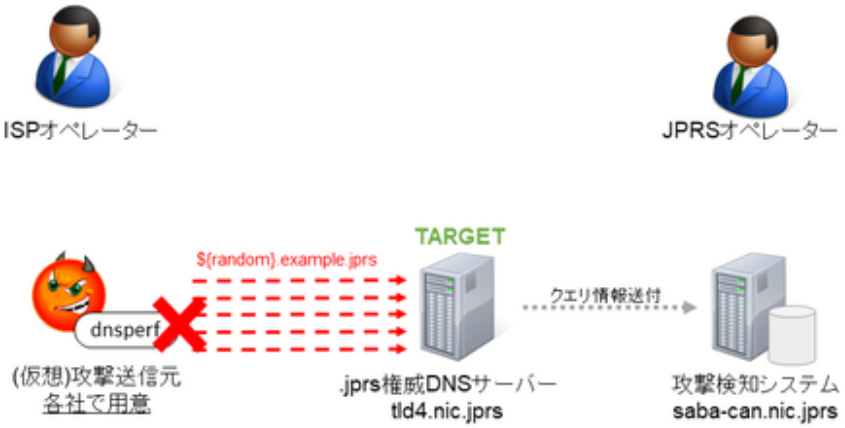

JPRSオペレーターから攻撃が発生しているISPへメールで連絡する
（今回は saba-theme1-ml@tldlabs.jprs MLを用いる）

## 実験シナリオ1（5）

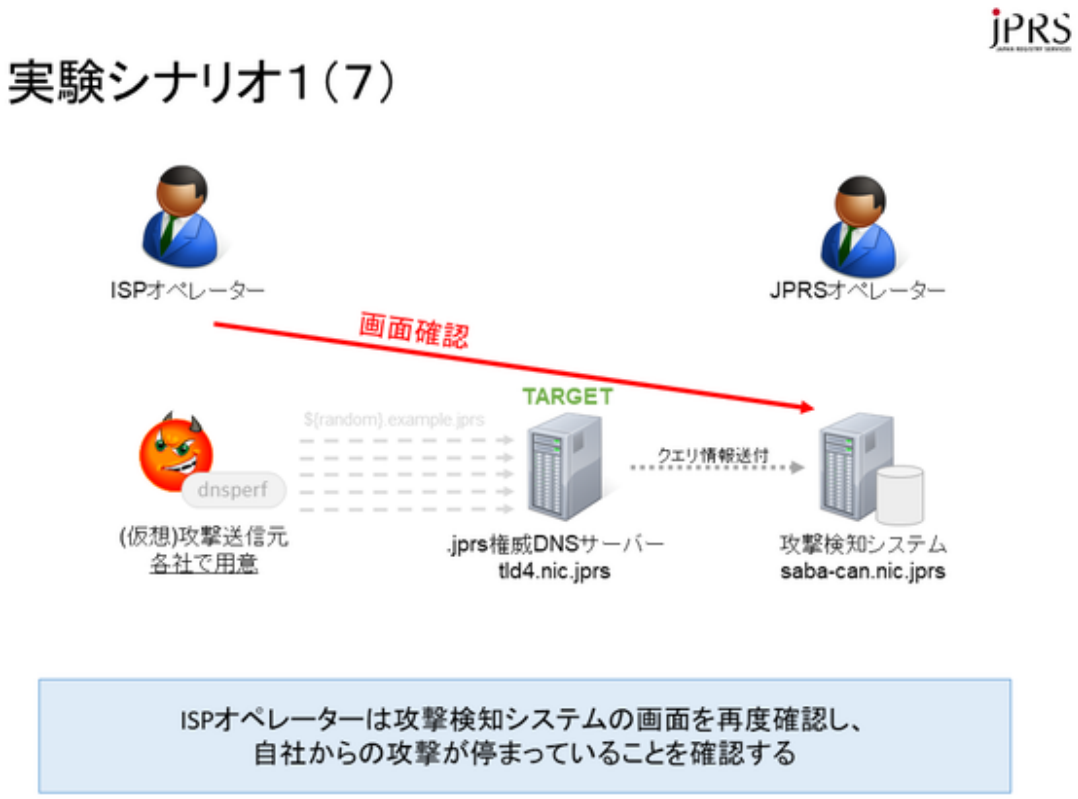



ISPオペレーターは攻撃検知システムの画面を開き、
自社から攻撃が発生していることを確認する

## 実験シナリオ1（6）




各社でdnsperfを停止し、攻撃トラフィックを止める




シナリオ 2

Step.1 ISP 側において疑似攻撃生成ツールを用い、tld4.nic.jprs に 50 クエリ/秒の攻撃を発生させる

Step.2 解析環境で攻撃を検知し、TLD の権威 DNS サーバー(JPRS)及び ISP のオペレーターの双方にメール/Slack で通知が送信される

Step.3 JPRS 及び ISP は解析環境でトラフィック状況を確認し、攻撃の詳細情報を調査する

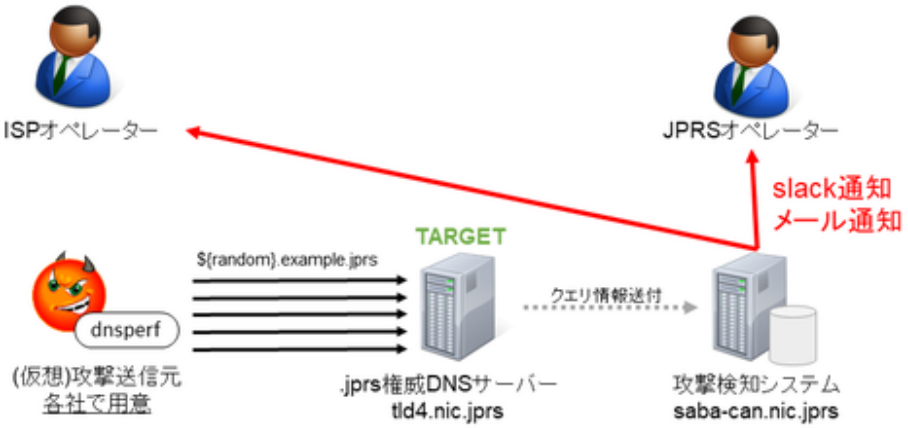
Step.4 攻撃の特定及び対処ができたと想定し、疑似攻撃生成ツールを停止する

Step.5 ISP 及び JPRS は解析環境へアクセスし、攻撃が停止していることを確認する

# 実験シナリオ2（1）

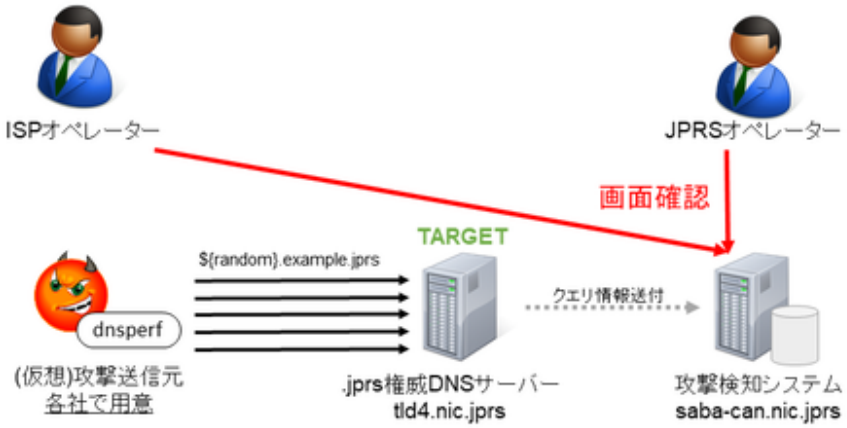

各社の(仮想)攻撃送信元からdnsperfでtld4.nic.jprsに向かって50qps程度の
ランダムサブドメイン攻撃を発生させる

# 実験シナリオ2（2）



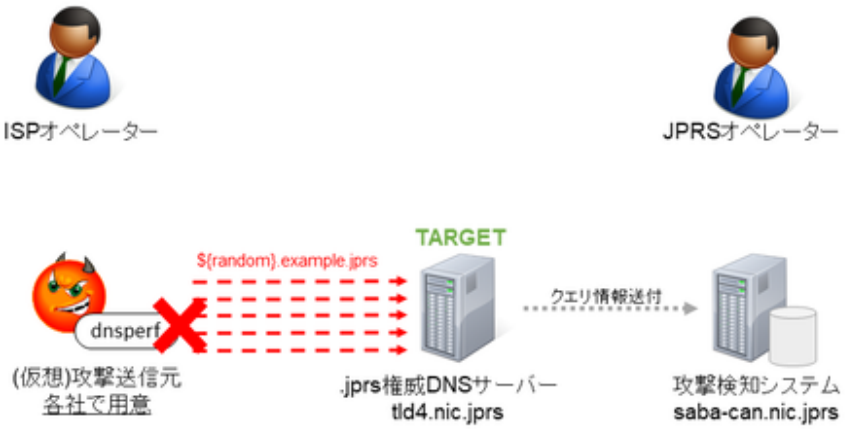

攻撃検知システムからJPRSオペレーター、ISPオペレーターへ
slackおよびメールでアラート通知が飛ぶ
（メールは saba-theme1-ml@tldlabs.jprs MLを用いる）
（slackは dotjprs.slack.com #alerting チャンネルを用いる）

# 実験シナリオ2（3）

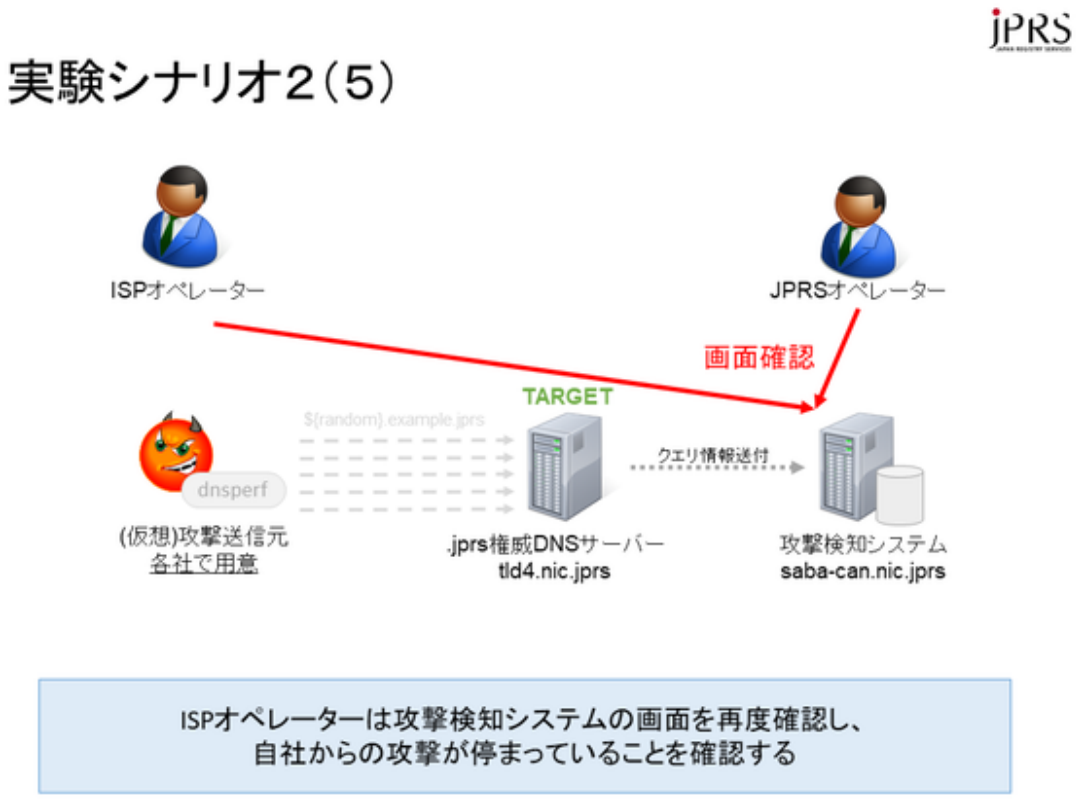



JPRSオペレーターおよびISPオペレーターは攻撃検知システムの画面を開き、
どこから攻撃がきているかを確認する

# 実験シナリオ2（4）




各社でdnsperfを停止し、攻撃トラフィックを止める




疑似攻撃生成ツールには権威 DNS サーバーやフルリゾルバーのベンチマークテストに使われる dnsperf [1] を使用し、以下のコマンドで疑似攻撃を生成した。

```
$ for I in {1..1000000}; do echo "dns$I.example1.jprs. A" >> /tmp/attack
_query_2ld.txt ; done
$ dnsperf -s 103.198.210.1 -S 1 -d /tmp/attack_query_2ld.txt -Q 50
```

攻撃検知の閾値として、前述した「1 秒間あたりの QNAME の 3LD 以上のユニーク数 ÷ 送信元 IP のユニーク数」の結果が 5 分間連続で 10 を超えていた場合に、アラートが上がるように設定した。

### 攻撃検知の際の情報共有に関する課題

本実証実験では攻撃を検知した際に各 ISP に攻撃に関する情報を共有することで、JPRS と ISP における連携を図っている。そのため、JPRS と ISP の間でどのような情報をどのようなタイミングでどのように共有するか、いくつかの視点で整理が必要になる。

[1] https://www.dns-oarc.net/tools/dnsperf

技術的な側面からは、攻撃に関する詳細な情報を共有できれば、それに対処する方法を検討しやすくなるため、できるだけ詳細な情報を速やかに共有することが理想的である。一方、法律的な側面からは、共有された情報により通信相手・通信内容などを相手先に特定されるおそれがあり、結果として電気通信事業における通信の秘密を侵す可能性がある。

本実証実験では、どのような情報の共有が望ましいかを確認するため、できる限り多くの情報を共有することとした。実運用の際には関連する法律や事例を照会・考慮しながら、実データの取得や情報共有を進めていく必要がある。

## 実験結果

本実証実験は、スケジュール上の都合で 2 日間に分けて実施した。ここでは、実験開始時間を基点とし、1 つにそろえた時系列として、実験結果をまとめている。

表: シナリオ 1 (時間単位:分)

| Step | HOTnet | QTnet | Softbank | OTNet | HTNet | Optage | FreeBit | CNCI | エネコム | 平均 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | - | - | - | - | - | - | - | - | - | - |
| 2 | - | 0 | 1 | 6 | 3 | 13 | - | - | 3 | 4 |
| 3 | - | 0 | 1 | 6 | 3 | 13 | - | - | 3 | 4 |
| 4 | 9 | 4 | 4 | 8 | 5 | 15 | 9 | 11 | 5 | 8 |
| 5 | 9 | 4 | 4 | 8 | 5 | 15 | 9 | 11 | 5 | 8 |
| 6 | 12 | 11 | 6 | 14 | 7 | 16 | 10 | 14 | 8 | 11 |
| 7 | - | 16 | 6 | 15 | 13 | 22 | - | - | 8 | 13 |

表: シナリオ 2 (時間単位:分)

| Step | HOTnet | QTnet | Softbank | OTNet | HTNet | Optage | FreeBit | CNCI | エネコム | 平均 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | - | - | - | - | - | - | - | - | - | - |
| 2 | 5 | 0 | 0 | 1 | 1 | 4 | 10 | 3 | 0 | 2 |
| 3 | 5 | 0 | 0 | 1 | 1 | 4 | 10 | 3 | 0 | 2 |
| 4 | 9 | 4 | 6 | 7 | 5 | 6 | 13 | 8 | 7 | 7 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 5 | 9 | 9 | 11 | 10 | 5 | 11 | 18 | 8 | 12 | 10 |

シナリオ 1 の HOTnet・FreeBit・CNCI では、攻撃のボリュームが閾値をわずかに下回っており、解析環境が攻撃として検出しなかったため、Step2 の通知を送信する時間が取得できず、それ以降の Step3 や回復と判断するための Step7 についても、時間の記録ができていなかった。実験の途中で JPRS が本件を検知し、Step4 の ISP への連絡を実施したため、Step4 以降の時間のみ記録が存在している。また、Step7 は攻撃が停止していることの確認であり、当該の ISP ではそもそも攻撃として検出していなかったため、記録としては「なし」となっている。

この、シナリオ 1 において一部 ISP で閾値を下回ってしまった理由として、疑似攻撃の規模を 50 クエリ/秒としていたものの、実験環境では秒ごとに攻撃頻度にばらつきが存在し、5 分間単位の計測においてユニークなクエリ数が 10 を下回る瞬間があったためと考えられる。例えば、ある 1 秒間では 45 クエリがサーバーへ到着し、次の 1 秒間では 55 クエリがサーバーへ到着する、といった状況である。このことから、今回の実験では閾値の設定値が大き過ぎたと考えられる。

全体的な傾向として、シナリオ 1 とシナリオ 2 を比較した場合、シナリオ 2 が時間的に優れている旨の結果が得られた。権威 DNS サーバーのオペレーターを介さない分、その後のレスポンスが早くなっているものと考えられる。平均的には、攻撃検知から停止(シナリオ 1 では Step6、シナリオ 2 では Step4)までの時間が、シナリオ 2 のほうが 4 分ほど早かった。

Softbank・エネコムでは、シナリオ 1 のほうが時間的に優れている結果が出ているが、実際に攻撃が停止した時点、すなわちシナリオ 1 の Step6 とシナリオ 2 の Step4 の時間を比較した場合、ほぼ同時間、もしくはシナリオ 2 のほうがやや優れた結果となっている。シナリオ 1 の Step7 やシナリオ 2 の Step5 は、どちらも攻撃が停止したと権威 DNS サーバーのオペレーターが認知するまでの時間であり、システム的・自動的ではない要素が含まれているため、このような結果になったと考えられる。

## 得られた知見

本実証実験では、シナリオ 2 として解析環境が攻撃を検知した際、解析環境からフルリゾルバーのオペレーターに直接アラートを発報することも実験したが、これは実環境では即座に実現することは難しいと判断した。その理由として、現状における ISP 各社の運用として、人から人へのメールや電話による連絡をきっかけとして調査や対処などのアクションが始まる流れになっており、社外のシステムから機械的に送信されたアラートのみでは、アクションを起こしにくいことが挙げられる。

また、解析環境は、閾値に基づいて攻撃を機械的に検知しているだけであり、False Positive や False Negative の発生があり得る。現状の DNS 運用においては、解析環境が検知した後、権威 DNS サーバーのオペレーターが目視により確認し、フルリゾルバーのオペレーターとの連携が必要か否かの判断を実施した上で、連絡することが最適であると考えられる。

ある検知システムにおける最適な閾値を見出すことはそれ自体が研究テーマの一つであり、異常検知の分野においてさまざまな研究が進められている。本実証実験では、現状の DNS 運用において現実的に構築可能な解析環境と閾値をきっかけとして検知する仕組みと、その対処における関係者との情報連携を対象としており、閾値の精度については対象外としている。

## 今後の課題

本実証実験では、簡易な計算式による閾値をきっかけとした、異常検知とその対処を実装・検証した。今後の実験対象として、異常検知の分野で見出されている見を取り入れることで検知の速度や精度を向上させ、オペレーター間における具体的な連携に反映させることが考えられる。

本実証実験で実装した解析環境は、ランダムサブドメイン攻撃の検知以外にも応用できる可能性がある。例えば、設備の増強計画を策定するためのトラフィック量の計測/記録や、サービスの安定性向上を図る際の権威 DNS サーバー・フルリゾルバーにおけるきめ細かなトラフィックバランスを確認する手段としても応用可能であると考えられる。こうした項目を考慮しつつ、平常時及び非常時における DNS のよりよい運用について、今後も継続的に検討を進めていきたい。

## テーマ 2: DDoS 攻撃の防御・緩和実験

### DDoS 攻撃の防御・緩和

本テーマでは、主要なオープンソースの DNS ソフトウェアに実装されている DDoS 攻撃の防御・緩和策について、実証実験を通じその効果を評価することを目的としている。

前述した通り、DNS に対するサイバー攻撃は大きく以下の 2 種類に分類される。

1. DNS そのものを対象とした攻撃
2. DNS の特性を利用し、第三者を対象とした攻撃

ここでは、前者の「DNS そのものを対象とした攻撃」としてランダムサブドメイン攻撃を、後者の「DNS の特性を利用し、第三者を対象とした攻撃」として DNS Amp 攻撃を取り上げ、それぞれの攻撃の防御・緩和策として実装されている以下の 4 種類の機能について、実運用に準じた環境で構築した実験環境を用い、その効果を評価した。

1. fetch-limit, rate-limit
2. serve-stale
3. NSEC/NSEC3 Aggressive Use
4. DNS Cookies

評価の結果、一部の DNS ソフトウェアにおいて期待した動作と異なる動作をするものがあることが判明したことから、当該 DNS ソフトウェア開発元へのフィードバックを併せて実施した。

### DNS Amp 攻撃の概要

本テーマにおいて取り上げた二つの DDoS 攻撃手法のうち、ランダムサブドメイン攻撃の概要は既に解説しているため、ここでは、DNS Amp 攻撃の概要について解説する。

DNS Amp 攻撃は DNS の特性を利用し、権威 DNS サーバーやフルリゾルバーの運用者以外の第三者を攻撃する DDoS 攻撃の 1 つである。

DNS には、問い合わせよりも応答の方がデータのサイズが大きく、小さな問い合わせパケットで大きな攻撃パケットを生成（攻撃を増幅）できるという特性がある。DNS では通信プロトコルに UDP と TCP の両方を使っているが、UDP では比較的容易に送信元の偽装が可能であるため、攻撃対象の IP アドレスを送信元として設定したクエリをフルリゾルバーや権威 DNS サーバーに送り、増幅された応答

を攻撃対象に大量に送りつけることで、攻撃対象のネットワーク回線等のリソースを逼迫させることが可能になる。

また、現時点において十分な対策が実施されておらず、本攻撃に利用可能なフルリゾルバーや権威 DNS サーバーがインターネット上に多数存在する。そのため、それらを踏み台として利用することで、送信元 IP アドレスによるフィルタリングを活用した防御が難しくなる。本件の対策として、送信元を偽装したパケットが組織外のネットワークに出て行くことを防ぐフィルタを導入することが BCP 38[2]として以前から推奨されているが、現時点における普及率は十分ではない。

かつ、BCP38 による対策は攻撃の送信元のネットワークで導入されている必要があり、攻撃を受信するネットワークにおける導入は、攻撃そのものの防御には効果を発揮しない。このような状況から、インターネットを構成するネットワーク全体に十分に普及するまでは、BCP38 は DNS Amp 攻撃をはじめとする、送信元 IP アドレスの偽装への対策とはなり得ない状況となっている。

## 実証実験の概要

本実証実験では主要なオープンソースの DNS ソフトウェアに実装されているランダムサブドメイン攻撃、DNS Amp 攻撃の防御・緩和策である、以下の 4 種類の機能について評価した。

- ランダムサブドメイン攻撃に対する防御・緩和機能
  - fetch-limit, rate-limit
    - https://kb.isc.org/docs/aa-01304
    - フルリゾルバーから権威 DNS サーバーへの問合せ量を制限する機能
  - serve-stale
    - https://www.rfc-editor.org/rfc/rfc8767.html
    - TTL の満了後もキャッシュを破棄せず応答を返し名前解決を継続する機能
  - NSEC/NSEC3 Aggressive Use
    - https://www.rfc-editor.org/rfc/rfc8198.html
    - 不存在証明を活用した不要なクエリ送信を抑制する機能
- 送信元を偽装した DNS Amp 攻撃に対する防御・緩和機能
  - DNS Cookies
    - https://www.rfc-editor.org/rfc/rfc7873.html
    - DNS パケットの偽装を検知する機能

[2] https://www.rfc-editor.org/bcp/bcp38.html

まず、前述した 4 種類の機能について、その概要や各サーバーソフトウェア実装における特徴を把握するための事前評価のための実験を実施した。実験では、各実装においてそれぞれの機能を無効から有効・有効から無効にした際にどのような挙動を示すか、また、ログにどのような出力がされるかについて、実験に参加したすべての組織において評価を実施した。

続いて、実運用に準じた環境、かつ複数の機能を有効にした状態でどのような挙動を示すかを評価する本評価を実施した。本評価にあたり、ランダムサブドメイン攻撃が発生したことを想定したシナリオを作成し、それに沿った形で手順を進めた。

この際、serve-stale が有効に機能してキャッシュの TTL を無視した応答がされている間に、権威 DNS サーバー側が保持する情報が変化したケースについて、シミュレーションを実施した。

実験では、TLD の権威 DNS サーバーオペレーターから共有された DDoS 攻撃情報を契機とし、ISP が対処する形で手順を進めた。

### 事前評価

各サーバーソフトウェアにおける機能の実装についての把握、及び実験に参加した ISP 各社における各機能への習熟を目的として、事前評価を実施した。

- BIND 9
  - fetch-limit, rate-limit: 対応（9.9.8～）
  - serve-stale: 対応（9.12～）
  - NSEC/NSEC3 Aggressive Use: NSEC 対応（9.12～）
  - DNS Cookies: 対応（9.11.26～）
- Unbound
  - fetch-limit, rate-limit: 対応（1.5.4～）
  - serve-stale: 対応（1.6.0～）
  - NSEC/NSEC3 Aggressive Use: 対応（1.7.0～）
  - DNS Cookies: 当時未対応

## 実験環境

事前評価は、評価用のシステム構成を用意し評価を実施した。評価にあたり、実験のためのクエリをインターネット上で実運用されているサーバーに送信しないよう、隔離された環境を構築し、その環境に、実験専用のドメイン名空間を構築した。

実験専用のドメイン名空間を構築するため、隔離された環境にルートゾーン・TLD（.jprs）・2LD の権威 DNS サーバーを準備した。それらの IP アドレスはすべて実験用のものに置き換え、実験環境内のループバックアドレスを用いて設定した。

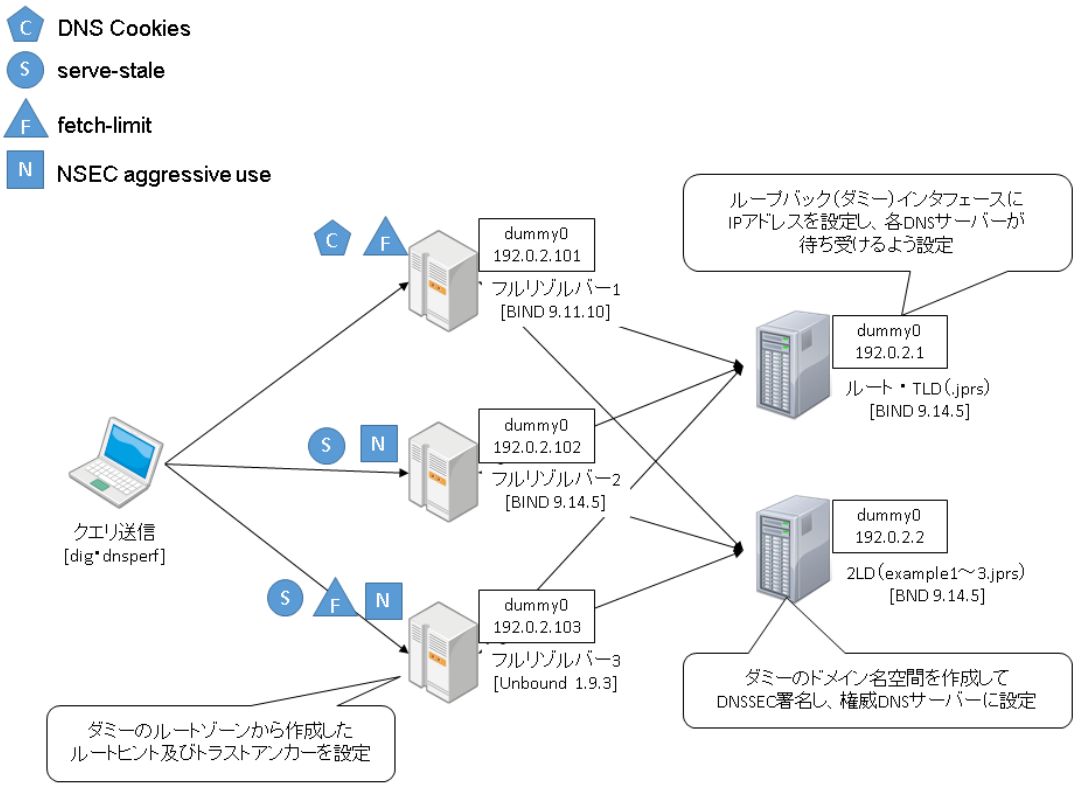



**図: 事前評価のシステム構成図**

それぞれの実装の評価は、実験に参加した各組織で以下のように分担した。

| 参加社名 | ソフトウェア |
|---|---|
| HOTnet | BIND 9 |
| HTNet | BIND 9 |
| Enecom | BIND 9 |
| OPTAGE | Unbound |
| QTnet | BIND 9 |
| OTNet | Unbound |
| Softbank | Unbound |
| CNCI | Unbound |
| FreeBit | BIND 9 |

各サーバーの IP アドレス割り当ては次の通りとした。

| 用途 | IPv4 アドレス | IPv6 アドレス |
|---|---|---|
| ルート・TLD 権威 DNS サーバー | 192.0.2.1 | 2001:db8:53::1 |
| 2LD 権威 DNS サーバー | 192.0.2.2 | 2001:db8:53::2 |
| フルリゾルバー（BIND 9.11） | 192.0.2.101 | 2001:db8:53::101 |
| フルリゾルバー（BIND 9.14） | 192.0.2.102 | 2001:db8:53::102 |
| フルリゾルバー（Unbound） | 192.0.2.103 | 2001:db8:53::103 |

実験専用のドメイン名空間は次の通りとした。

| ゾーン | DNSSEC 署名 | 不在証明方式 |
|---|---|---|
| ルート | あり | NSEC |
| TLD（.jprs） | あり | NSEC3+Opt-Out |
| example1.jprs | あり | NSEC3+Opt-Out |
| example2.jprs | あり | NSEC3+Opt-Out |
| example3.jprs | あり | NSEC3+Opt-Out |

ゾーンの内容は次の通りとした。

- ルート
  - シリアル 2019091000 をもとに作成
    - ルート DNS サーバーの情報を実験環境のものに書き換え
    - ルートゾーン自身の情報と TLD への委任、DNSSEC 関連のレコードのみ
  - .jprs 含む現実の TLD 全ての委任情報を記載
    - .jprs の委任情報だけ、実験環境のものに書き換え
    - .jprs 以外は現実の委任情報を返してしまうが、実験環境は隔離されているので問題ない
  - DNSSEC 署名は実験用のものに差し替え
    - オリジナルの DNSSEC 署名の情報（RRSIG RR・DNSKEY RR）を削除
    - KSK・ZSK を新たに作成し、2020 年末まで有効な DNSSEC 署名を作成
    - 不在証明は実際のルートゾーンと同じ NSEC 方式
- TLD（.jprs）

- 実験環境用に新規作成
  - TLD DNS サーバーの情報を実験環境のものに書き換え
  - ルートゾーン自身の情報と TLD への委任、DNSSEC 関連のレコードのみ
- 実験用セカンドレベルドメインの委任情報だけを記載
  - example1.jprs, example2.jprs, example3.jprs
  - 全て同じ IP アドレスのサーバーに委任
- DNSSEC 署名は実験用のものに差し替え
  - KSK・ZSK を新たに作成し、2020 年末まで有効な DNSSEC 署名を作成
  - 不在証明は実際の.jprs ゾーンと同じ NSEC3 方式（Opt-Out あり）

- example1.jprs
  - 実験環境用に新規作成
  - DNSSEC 署名は新規作成
    - KSK・ZSK を新たに作成し、2020 年末まで有効な DNSSEC 署名を作成
    - 不在証明は NSEC3 方式（Opt-Out あり）

表: example1.jprs ゾーンに登録したリソースレコード

| ドメイン名 | レコードタイプ | レコード内容 |
|---|---|---|
| example1.jprs | SOA・DNSSEC 関連 | （省略） |
| example1.jprs | NS | ns.example1.jprs |
| example1.jprs | MX | 10 mail.example1.jprs |
| ns.example1.jprs | A | 192.0.2.2 |
| ns.example1.jprs | AAAA | 2001:db8:53::2 |
| mail.example1.jprs | A | 198.51.100.1 |
| mail.example1.jprs | AAAA | 2001:db8:198:51:100::1 |
| www.example1.jprs | A | 198.51.100.1 |
| www.example1.jprs | AAAA | 2001:db8:198:51:100::1 |
| mail.sub.example1.jprs | A | 198.51.100.101 |
| mail.sub.example1.jprs | AAAA | 2001:db8:198:51:100::101 |
| www.sub.example1.jprs | A | 198.51.100.101 |
| www.sub.example1.jprs | AAAA | 2001:db8:198:51:100::101 |

- example2.jprs

- 実験環境用に新規作成
- DNSSEC 署名は新規作成
  - KSK・ZSK を新たに作成し、2020 年末まで有効な DNSSEC 署名を作成
  - 不在証明は NSEC3 方式（Opt-Out あり）

表: example2.jprs ゾーンに登録したリソースレコード

| ドメイン名 | レコードタイプ | レコード内容 |
|---|---|---|
| example2.jprs | SOA・DNSSEC 関連 | （省略） |
| example2.jprs | NS | ns.example2.jprs |
| example2.jprs | MX | 10 mail.example2.jprs |
| ns.example2.jprs | A | 192.0.2.2 |
| ns.example2.jprs | AAAA | 2001:db8:53::2 |
| mail.example2.jprs | A | 198.51.100.2 |
| mail.example2.jprs | AAAA | 2001:db8:198:51:100::2 |
| www.example2.jprs | A | 198.51.100.2 |
| www.example2.jprs | AAAA | 2001:db8:198:51:100::2 |

- example3.jprs
  - 実験環境用に新規作成
  - DNSSEC 署名は新規作成
    - KSK・ZSK を新たに作成し、2020 年末まで有効な DNSSEC 署名を作成
    - 不在証明は NSEC3 方式（Opt-Out あり）

表: example3.jprs ゾーンに登録したリソースレコード

| ドメイン名 | レコードタイプ | レコード内容 |
|---|---|---|
| example3.jprs | SOA・DNSSEC 関連 | （省略） |
| example3.jprs | NS | ns.example3.jprs |
| example3.jprs | MX | 10 mail.example3.jprs |
| ns.example3.jprs | A | 192.0.2.2 |
| ns.example3.jprs | AAAA | 2001:db8:53::2 |
| mail.example3.jprs | A | 198.51.100.3 |
| mail.example3.jprs | AAAA | 2001:db8:198:51:100::3 |

| | | |
|---|---|---|
| www.example3.jprs | A | 198.51.100.3 |
| www.example3.jprs | AAAA | 2001:db8:198:51:100::3 |

## 評価の内容

評価の対象とした 4 つの機能について、BIND 9 と Unbound での設定内容や挙動を確認した。

- fetch-limit
  - BIND 9（フルリゾルバーとして動作）と Unbound において、ドメイン名や権威 DNS サーバー単位での反復検索の並列度を制限する機能について評価を実施した。
  - それぞれの実装について、設定パラメーターとその意味を確認した。
  - 実際にクエリを送信し、当該機能による制限が働くことを確認した。
- serve-stale
  - BIND 9（フルリゾルバーとして動作）と Unbound において、TTL が満了しているがキャッシュに存在するレコードを利用した応答について評価を実施した。
  - 実験環境内の権威 DNS サーバーを停止させることで応答を得られない状態を作り出し、TTL が満了したレコードを使い、継続して応答することを確認した。
- NSEC Aggressive Use
  - BIND 9（フルリゾルバーとして動作）と Unbound において、NSEC レコードを利用した不在応答の生成について評価を実施した。
  - 存在しない TLD やその下にあるドメイン名について、カバーする NSEC レコードがキャッシュされた状態であれば、ルート DNS サーバーへの反復検索を実施することなく、不在応答を返せることを確認した。
  - 当該機能の無効・有効を切り替え、同じクエリを送った場合、機能が有効な場合にのみ、期待した挙動を示すことを確認した。
- DNS Cookies
  - BIND 9（権威 DNS サーバーとして動作）において、DNS Cookies の挙動の評価を実施した。
  - クッキーを付加した問い合わせを送信することで、サーバークッキーとクライアントクッキーの性質の違いを確認した。
  - クッキーの一部を書き換えたり、違うクライアント IP アドレス宛のクッキーを使ったりすると不正なリクエストとして扱われることを確認した。

### 事前評価を受けた検討の状況

以上の事前評価により各サーバー実装における設定ファイルの記述や挙動の観測を通じて得られた知見を活用し、本評価を実施するにあたっての検討を実施した。

その結果、特に DNS Cookies と NSEC/NSEC3 Aggressive Use について参加組織の担当者から、現時点のインターネットにおける導入・普及状況・サーバーソフトウェアにおける実装状況などから、実際のインターネットへの導入を前提とした本評価を実施するには時期尚早ではないかという懸念・課題が指摘された。

以下、それぞれの機能について指摘された懸念・課題について記述する。

## DNS Cookies

DNS Cookies については、実験時点では実環境における権威 DNS サーバー側の対応率が低いこと、運用を支援する仕組み・機能が不足していたこと、プロトコルの修正を実施する動きがあったことから、今回の実験では評価が難しいと判断した。

事前評価を実施した当時、実環境における権威 DNS サーバー側の対応率が低いというデータがあった。このため、DDoS 攻撃対策として DNS Cookies を必須にすると、対応していない権威 DNS サーバーへの問い合わせができなくなることから現実的でないという懸念が指摘された。その対応として、フルリゾルバーからの反復検索においてクエリを送信する権威 DNS サーバーごとに DNS Cookies を有効・無効に設定できるとよいのではないか、という提案があり、当該手法の実現性について検討した。

その検討の中で、権威 DNS サーバーごとに DNS Cookies の有効・無効を切り替えた場合、DNS Cookies に対応している権威 DNS サーバーとの間では DNS メッセージの偽装を防ぐことができる一方、対応しているサーバーのリストをフルリゾルバーにて保守することは、リストの正しさに関する検証が必要になることもあり、実環境における適用は現時点では難しいのではないかという懸念が指摘された。

また、不正な DNS Cookies が含まれる DNS メッセージを受け取った旨のログからスプーフィングや DNS Amp 攻撃を検知する仕組み、BIND 9 に付属の dig コマンド等の DNS 問い合わせを送るソフトウェアで簡単に DNS Cookies を送る方法など、運用を支援する仕組み・機能の追加が必要になるのではないかという意見も出された。

更に、DNS Cookies では、事前評価を実施した当時、権威 DNS サーバー側でクッキーを生成するための secret を円滑に更新（ロールオーバー）する仕組みがなかったこと、また、IETF の dnsop ワーキンググループにおいて Cookie を生成する

アルゴリズムを統一するためのプロトコルの修正を実施する動きがあり、今後大きく実装が変わる可能性があることも、本評価を進める上での障害となり得ることが課題として挙げられた。

### NSEC/NSEC3 Aggressive Use

NSEC/NSEC3 Aggressive Use については、ゾーンの不在証明方式によっては効果が得られない場合があることから、今回の実験では評価が難しいと判断した。

事前評価を実施した当時、主要なフルリゾルバー実装における NSEC/NSEC3 Aggressive Use への対応は、不在証明として NSEC 方式を採用している場合に限定されていた。そのため、NSEC 方式を採用しているルートゾーンでは導入による効果が期待できるが、一部の TLD では外部からの DNS 検索で登録済みのドメイン名の一覧をリスト化されることを防ぐなどの目的で不在証明として NSEC3 方式を採用しており、そうした TLD に対しては対応したフルリゾルバー実装がないので実験ができないという懸念が指摘された。

また、NSEC3 方式では DNSSEC に対応していない委任情報を不在証明の対象外とする Opt-Out が有効にされている場合、本機能による効果が得られなくなる。一部の TLD では DNSSEC 署名のコストやゾーンファイルの容量削減のために Opt-Out が有効にされており、そうした TLD では導入による効果が期待できないことも、懸念として指摘された。

## 検討状況を受けた評価対象の絞り込み

以上の検討結果を受け、事前評価を実施した 4 つの機能のうち DNS Cookies と NSEC/NSEC3 Aggressive Use については、本評価の対象外とすることとした。

一方、serve-stale と fetch-limit については複数のフルリゾルバー実装が対応しており、かつ、現実の TLD への導入を前提とした、本評価の実施が可能であると判断した。

以上により、本実証実験では serve-stale と fetch-limit の 2 種類の機能について、本評価を実施することとした。

### 本評価

事前評価で得られた知見に基づいて実証実験のシナリオを作成し、本評価を実施した。

## 実験環境

本評価を実施した環境を以下 に示す。

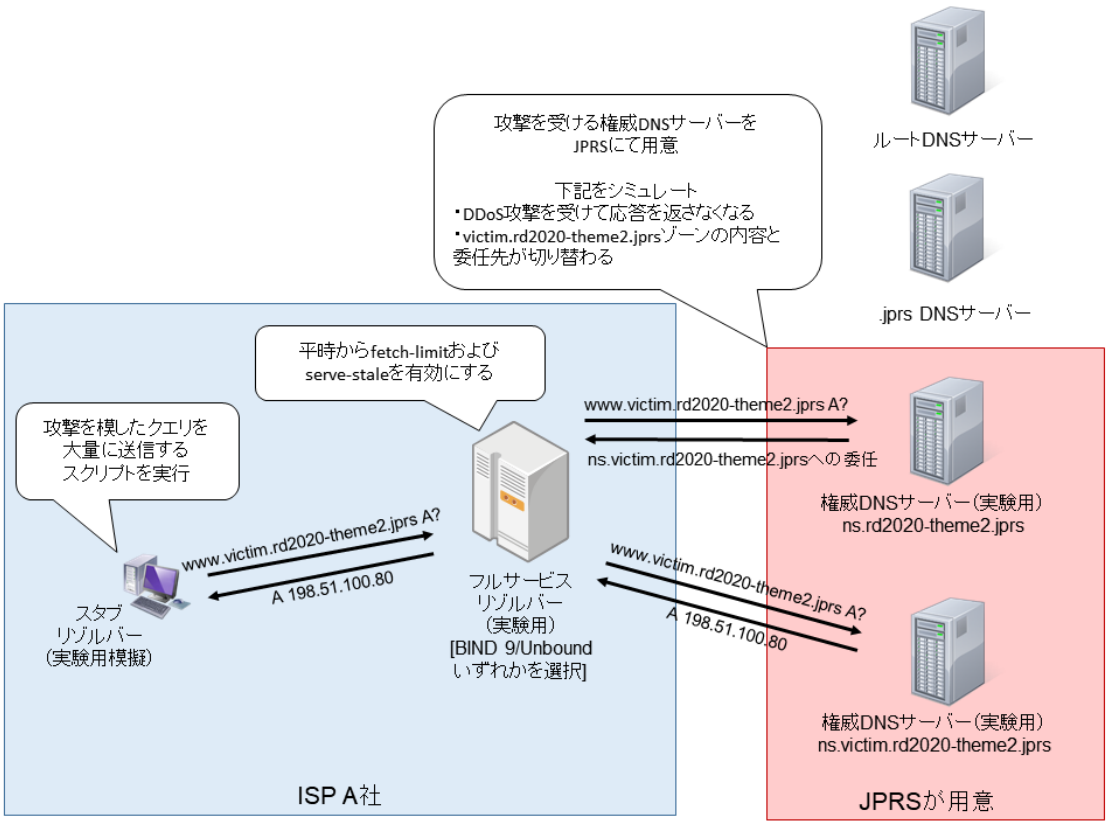



**図: 本評価のシステム構成図**

本評価に使用するドメイン名として、 **rd2020-theme2.jprs** を登録した。ゾーンの構成は次の通りとした。

- rd2020-theme2.jprs
  - 実験環境用に新規作成
  - DNSSEC 署名なし

表: rd2020-theme2.jprs ゾーンに登録したリソースレコード

| ドメイン名 | レコードタイプ | レコード内容 |
|---|---|---|
| rd2020-theme2.jprs | SOA・DNSSEC 関連 | （省略） |
| rd2020-theme2.jprs | NS | ns1.rd2020-theme2.jprs |
| rd2020-theme2.jprs | NS | ns2.rd2020-theme2.jprs |
| ns1.rd2020-theme2.jprs | A | (IPv4 アドレス) |
| ns2.rd2020-theme2.jprs | A | (IPv4 アドレス) |

| | | |
|---|---|---|
| victim.rd2020-theme2.jprs | NS | ns.victim.rd2020-theme2.jprs |
| ns.victim.rd2020-theme2.jprs | A | (IPv4 アドレス) |

- victim.rd2020-theme2.jprs （別サーバーに委任）
  - 実験環境用に新規作成
  - DNSSEC 署名なし
  - 委任元（rd2020-theme2.jprs）と別サーバーで運用

表: victim.rd2020-theme2.jprs ゾーンに登録したリソースレコード

| ドメイン名 | レコードタイプ | レコード内容 |
|---|---|---|
| victim.rd2020-theme2.jprs | SOA・DNSSEC 関連 | （省略） |
| victim.rd2020-theme2.jprs | NS | ns.victim.rd2020-theme2.jprs |
| ns.victim.rd2020-theme2.jprs | A | (IPv4 アドレス) |
| www.victim.rd2020-theme2.jprs | A | 198.51.100.80 |

- victim.rd2020-theme2.jprs （同じサーバーに委任）
  - 実験環境用に新規作成
  - DNSSEC 署名なし
  - 委任元（rd2020-theme2.jprs）と同じサーバーで運用

表: victim.rd2020-theme2.jprs ゾーンに登録したリソースレコード

| ドメイン名 | レコードタイプ | レコード内容 |
|---|---|---|
| victim.rd2020-theme2.jprs | SOA・DNSSEC 関連 | （省略） |
| victim.rd2020-theme2.jprs | NS | ns.victim.rd2020-theme2.jprs |
| ns.victim.rd2020-theme2.jprs | A | (IPv4 アドレス) |
| www.victim.rd2020-theme2.jprs | A | 203.0.113.80 |

「攻撃への対策として、権威 DNS サーバーが別サーバーに変更された」という状況をシミュレーションするため、 **victim.rd2020-theme2.jprs** を **rd2020-theme2.jprs** とは別サーバーに委任している構成（性能が低く、委任先が DDoS 攻撃でサービスダウンすることを想定）と、同じサーバーに委任している構成（性能が高く、DDoS 攻撃に耐えることを想定）を切り替えることとした。なお、本実験では実験環境のサーバー台数を減らすことを目的とし、親ゾーンと子ゾーンを同じサーバーに配置した。

実験にあたり、各参加社が都合の良い時間に実施できるよう、ゾーン構成の切り替えや応答・無応答状態の切り替えを、時間帯によって自動的に切り替わるよう

に設定した。具体的には、2 時間を 1 セットとし、1 時 0 分～2 時 59 分のセット、3 時～4 時 59 分のセット、……という形でスケジュールした。

| 時間帯 | victim ゾーンの委任先 | www.victim の A レコード |
|---|---|---|
| 1 時 0 分～1 時 29 分 | 別サーバーに委任 | 198.51.100.80 |
| 1 時 30 分～1 時 59 分 | 別サーバーに委任 | （応答せずタイムアウト） |
| 2 時 0 分～2 時 59 分 | 同じサーバーに委任 | 203.0.113.80 |

それぞれの実装の評価は、実験に参加した各組織で以下のように分担した。

| 参加社名 | ソフトウェア |
|---|---|
| HOTnet | BIND 9 |
| HTNet | BIND 9 |
| Enecom | BIND 9 |
| OPTAGE | Unbound |
| QTnet | BIND 9 |
| OTNet | Unbound |
| Softbank | Unbound |
| CNCI | Unbound |
| FreeBit | BIND 9 |

## 評価の内容

現実に起こりうる攻撃シナリオを想定し、serve-stale と fetch-limit を平時から有効にしておくことが、攻撃効果の軽減にどの程度有効か検証することを、本評価の目的とした。

そのため、評価の前提として、serve-stale・fetch-limit を共に平時から有効としている状態を想定した、この状態においてランダムサブドメイン攻撃が発生し、権威 DNS サーバーがサービスダウンしてから復旧するまでの一連の流れを、シナリオとして準備した。

以下、準備したシナリオの内容について記述する。

1. 特定ドメイン名に対するランダムサブドメイン攻撃が発生
   - 実験用に準備したドメイン名に対し、そのサブドメインに対するランダムサブドメイン攻撃を発生させる

2. fetch-limit の効果で、自組織からの問い合わせ量が絞り込まれる（攻撃への加担を軽減）
    - 1.への対応として、平時から有効していた防御機能が有効に機能していることを確認する
3. ランダムサブドメイン攻撃の攻撃先ドメイン名の権威 DNS サーバーがサービスダウン
    - 大規模な攻撃によって権威 DNS サーバーが応答できない状態を作り出す
4. serve-stale の効果で、ドメイン名の名前解決は継続
    - 3.の状態であっても、フルリゾルバーにキャッシュが残っていればその TTL が満了していても応答することを確認する
5. ランダムサブドメイン攻撃の攻撃先ドメイン名が権威 DNS サーバーごと移転し復旧
    - 委任情報を変更して、権威 DNS サーバーの引っ越しをシミュレーションする
6. serve-stale の副作用で、引き続き古い（引っ越し前の）情報が応答されてしまう
    - 5.で委任情報が変わり権威 DNS サーバーが復旧した後も、4. の効果で古い情報を使い続けるかどうか確認する
7. 対象ドメイン名のキャッシュをクリアし、改めて反復検索して新しい情報で応答するようになる
    - 6.でキャッシュの古い情報を使い続ける状態が続く場合、キャッシュクリアで対応できることを確認する

ランダムサブドメイン攻撃のシミュレーションには次のスクリプトを利用した、実験用フルリゾルバーにて実行した。

```
#!/bin/sh

set -u

TARGET_QPS=100

if [ ! -f /tmp/random_query.txt ]; then
        for prefix in `seq -w 1 1000000`; do
                echo "random${prefix}.www.victim.rd2020-theme2.jprs. A"
        done > /tmp/random_query.txt
fi
```

```
/var/saba-theme2/local/dnsperf/bin/dnsperf -s 127.0.0.1 -S 1 -d /tmp/ran
dom_query.txt -Q "${TARGET_QPS}"
```

BIND 9 については、named.conf の **options** ステートメントに以下の内容を記載した。

```
// disable DNS Cookies
answer-cookie no;
send-cookie no;

// disable NSEC Aggressive Use
synth-from-dnssec no;

// serve-stale
stale-answer-enable yes;
max-stale-ttl 86400;

// fetch-limit
fetches-per-zone 10 fail;
```

Unbound については、unbound.conf に以下の内容を記載した。

```
# serve-stale
serve-expired: yes
serve-expired-ttl: 86400
serve-expired-ttl-reset: no

# fetch-limit
ratelimit: 10
ratelimit-size: 4m
ratelimit-slabs: 1
ratelimit-factor: 10
```

## 本評価の結果

現実に起こりうる状況を想定したシナリオを実行した結果、各機能の効果と運用において注意すべき点に関し、以下の知見が得られた。

各実装の fetch-limit・serve-stale の挙動について、想定と一部異なる点が見受けられた。fetch-limit については、想定通り自組織からの問い合わせ量が絞り込まれたことを確認できた一方、BIND 9 について、後述の serve-stale を fetch-limit と組み合わせて使った際、応答を得られない場合があることが判明した。

その理由について実装内容を調査した結果、権威 DNS サーバーが応答しないドメイン名への問い合わせに関し、serve-stale で応答を返す処理よりも先に fetch-limit の制限が働いてしまうためであると判明した。すなわち、実証実験時点における BIND 9 実装では、fetch-limit と serve-stale を組み合わせた場合、権威 DNS サーバーがサービスダウンしていても名前解決できる状態にするという、serve-stale の恩恵を得られないことが判明した。なお、本件については後述の通り、開発元へのフィードバックを実施し、将来のリリースにおいて修正予定である旨の回答を得た。

serve-stale については、権威 DNS サーバーが応答を返せなくなった後、キャッシュに残っている TTL が満了したレコードを応答していることから、権威 DNS サーバーがサービスダウンした場合でも名前解決を継続できることが確認できた。

その一方、攻撃への対策等を目的とした権威 DNS サーバーの切り替えによって権威 DNS サーバーのサービスが復旧した後も引き続きキャッシュに残っている古いレコードを応答してしまう場合があることが確認できた。この対策として、古いレコードを使ってよい期間を制限する、または手動で当該キャッシュをクリアするといった対応が必要になると考えられる。

### DDoS 攻撃の防御・緩和実施結果

## 評価結果の総評と今後の課題

事前評価では、DDoS 攻撃の防御・緩和を目的とした 4 つの機能について、隔離されたネットワークの中で実際の挙動を試すことができた。その結果、それぞれの機能やサーバーソフトウェアにおける実装について、理解を深めることができた。

本評価では、fetch-limit と serve-stale に関して、インターネット上に用意した権威 DNS サーバーに対して攻撃を模擬したクエリを実際に送信し、権威 DNS サーバーが応答できない状態を作り出して、より実運用に近い環境で各機能の挙動についての実験を実施できた。その結果、二つの機能を組み合わせた際に想定と異なる挙動をすることを見つけることができ、その旨を開発元にフィードバックすることができた。

今後の課題として、各機能を平時から有効にしていた場合の、正常なクエリへの影響調査（各機能が消費するサーバーリソースは大きくないか、正常なクエリの処理に対して各機能が悪い影響を与えることがないか）、開発元にフィードバックした内容が修正された後の再試験の実施が挙げられる。

## DNS ソフトウェア開発元へのフィードバックとその結果

BIND 9 の開発元である ISC に対し、以下の問い合わせを行った。

- **`stale-answer-enable yes;`**にした状態で **`fetches-per-server`**・**`fetches-per-zone`** を設定すると、serve-stale より前に fetch-limit が作用して ServFail 応答が返ってくるように見受けられた。この挙動は仕様通りか。

ISC からは、以下の回答が得られた。

- ISC でも認識しており、修正される予定となっている。
  https://gitlab.isc.org/isc-projects/bind9/-/issues/1712
  https://gitlab.isc.org/isc-projects/bind9/-/issues/2066

Unbound の開発元である NLnet Labs に対し、以下の問い合わせを行った。

- **`serve-expired yes`** にした状態で、権威サーバーから応答が得られないことをしばらく覚えているように見受けられた。クライアントには stale な応答を返す際に、権威サーバーが復活しているかどうかたまにしかチェックしないのか？

NLnet Labs からは、以下の回答が得られた。

- Unbound は到達できない権威 DNS サーバーの情報を覚えており、その TTL は「infra-host-ttl」オプションで設定できる。デフォルト値は 900 秒である。

この回答に対し、以下の意見を送った。

- **`serve-expired-ttl`** のデフォルト値は **`0`** だが、RFC 8767（2020/3/31 発行）では stale timer として 1 日～3 日が妥当であるとされている。デフォルト値を変えたほうがいいのではないか？

その意見に対し、以下の回答が得られた。

- オペレーターの混乱を引き起こすことを考えると、すぐにデフォルト値を変えるべきかどうか判断が難しい。

# 実証実験報告（北海道総合通信網株式会社）

## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

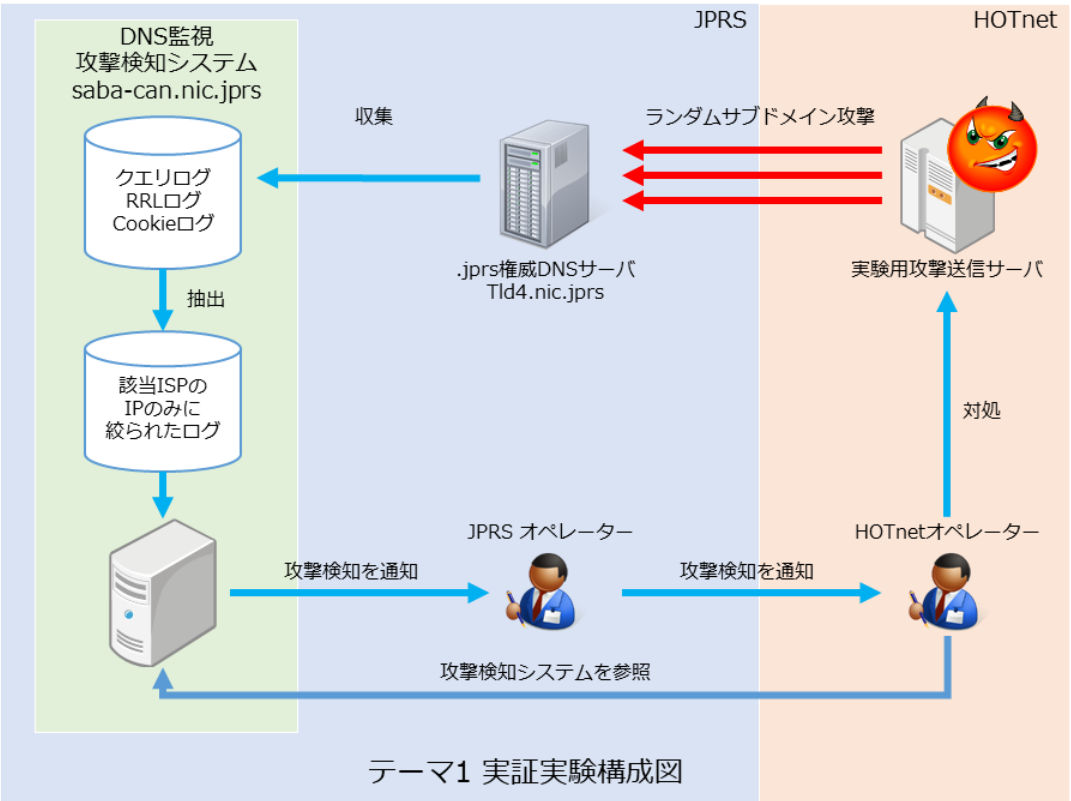
### 目的

近年、DNS サービスを標的としたサイバー攻撃が増加している。大規模なイベントを標的とした DDoS 攻撃が発生することで DNS サービスが停止すると、インターネットの利用やサービス提供に致命的な影響が発生することとなる。

本実験では、DDoS 攻撃手法としてのランダムサブドメイン攻撃を TLD DNS サーバーで検知し、攻撃を媒介しているフルリゾルバーを管理している ISP へ攻撃情報を連携・共有できるか確認を行った。

また、攻撃検知システムより提供される情報から、攻撃元の特定および対処の可否について検証を行った。

### 構成




テーマ1 実証実験構成図

図 1. テーマ 1　実証実験構成図

## 情報連携・共有手段

JPRS(TLD DNS サーバー側)から HOTnet(フルリゾルバー管理者)への情報連携ツールとして、既存の業務ツールであるメールおよび、リアルタイムで双方向の情報共有することに長けた Slack を導入した。

## 実験結果

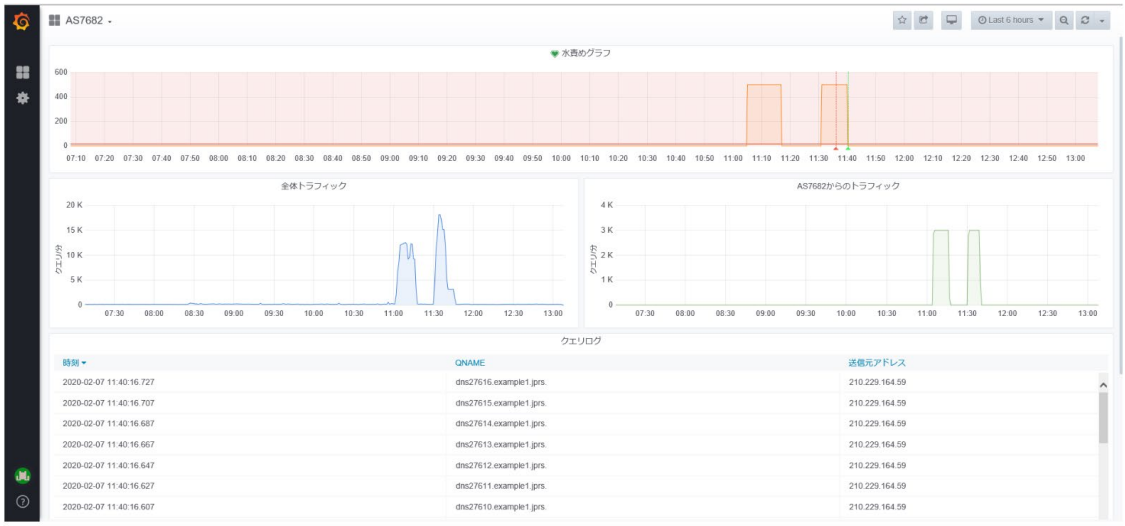



図 2. 攻撃検知システム

- 攻撃検知システムの表示サンプルを 図 2. に示した。
- ランダムサブドメイン攻撃を受けている時間帯、および送信元 IP アドレスを識別することが可能であった。
- また、ランダムサブドメイン攻撃が停止した後、本システム上でも攻撃が止まったことが確認できた。
- 検知した攻撃トラフィックを速やかに通知する仕組みとして、メールおよび slack ツールは想定した通り機能した。
- スムーズな情報共有ツールとして、メールだけでなく Slack の利用が効果的であると体感できた。

## 考察

- 攻撃検知システムを参照することで、攻撃元 IP アドレスの識別、および攻撃を受けている時間帯が想定通り可視化されていることを確認できた。

- クエリログにより送信元 IP アドレスを確認し、それを可視化することで、攻撃の状況を短時間で理解できた。
- TLD DNS サーバーの応答確認と合わせ、トラフィック量をグラフで確認できたことで、状況把握が容易になった。
- DNS オペレーターの習熟度によらず、状況の把握が容易になった。
- メールと Slack は速やかな情報共有をする上で十分に機能することが確認できた。
- メールは既存のフローで利用可能であるため、業務実装の実現性が高い。
- Slack を使用すると双方向の情報共有が容易である点に、利便性の高さを感じられた。

## 課題

- 自社顧客のサービス通信を停止してしまうなどの影響を考慮すると、攻撃トラフィックを即時停止することは難しい。
- オープンリゾルバーとなっているために攻撃を媒介している機器を検知した場合、攻撃主体であるとは限らないことが想定される。
- 実際の攻撃元を一つに絞ることは難しいことが想定される。送信元がランダムな IP アドレスから攻撃された場合の対応として、広範囲にトラフィックを停止することは自社顧客への通信影響が懸念される。
- メールおよび Slack によって情報共有が容易になる分、共有すべき情報や共有のフローなどについては、TLD 管理者と ISP の間における取り決め（ルール）が必要である。
- 24 時間 365 日の運用に落とし込むには厳格なルールが必要になる。
- DNS のクエリログの内容を取り扱うことから、通信の秘密について十分に考慮したルールが必要になると考える。
- クエリログの表示だけでは攻撃元を特定することが難しい。
- 実環境では正規のクエリと攻撃クエリが混合した状態で表示されるため、それらを区別できない可能性がある。
- 複数の IP アドレスから攻撃が発生した際、クエリログの集計機能があれば、クエリの傾向が攻撃元を特定する手助けになると考えられる。
- 攻撃の検知から調査および対処完了までの時間が短縮されることで、顧客のインターネット利用サービスへの影響時間を短縮できるため、以下の可能性を示したい。
- 攻撃検知システムにより攻撃元として判定された IP アドレスのリストが作成できれば、攻撃クエリの区別をすることなく、速やかな対応が可能となると考えられる。

- 攻撃検知後の連絡を自動化するなど、よりスムーズに攻撃情報を把握することができれば、より対応時間を短縮ことが可能になると考えられる。

## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

近年、DNS ソフトウェアには DDoS 攻撃を検知・防御・緩和するためのいくつかの機能が実装されている。これらの機能を活用することで DDoS 攻撃による DNS への影響を抑制・緩和することが期待できる。

本実験ではランダムサブドメイン攻撃に対する対策として、fetch-limit と serve-stale の有効性を検証する。

## 構成

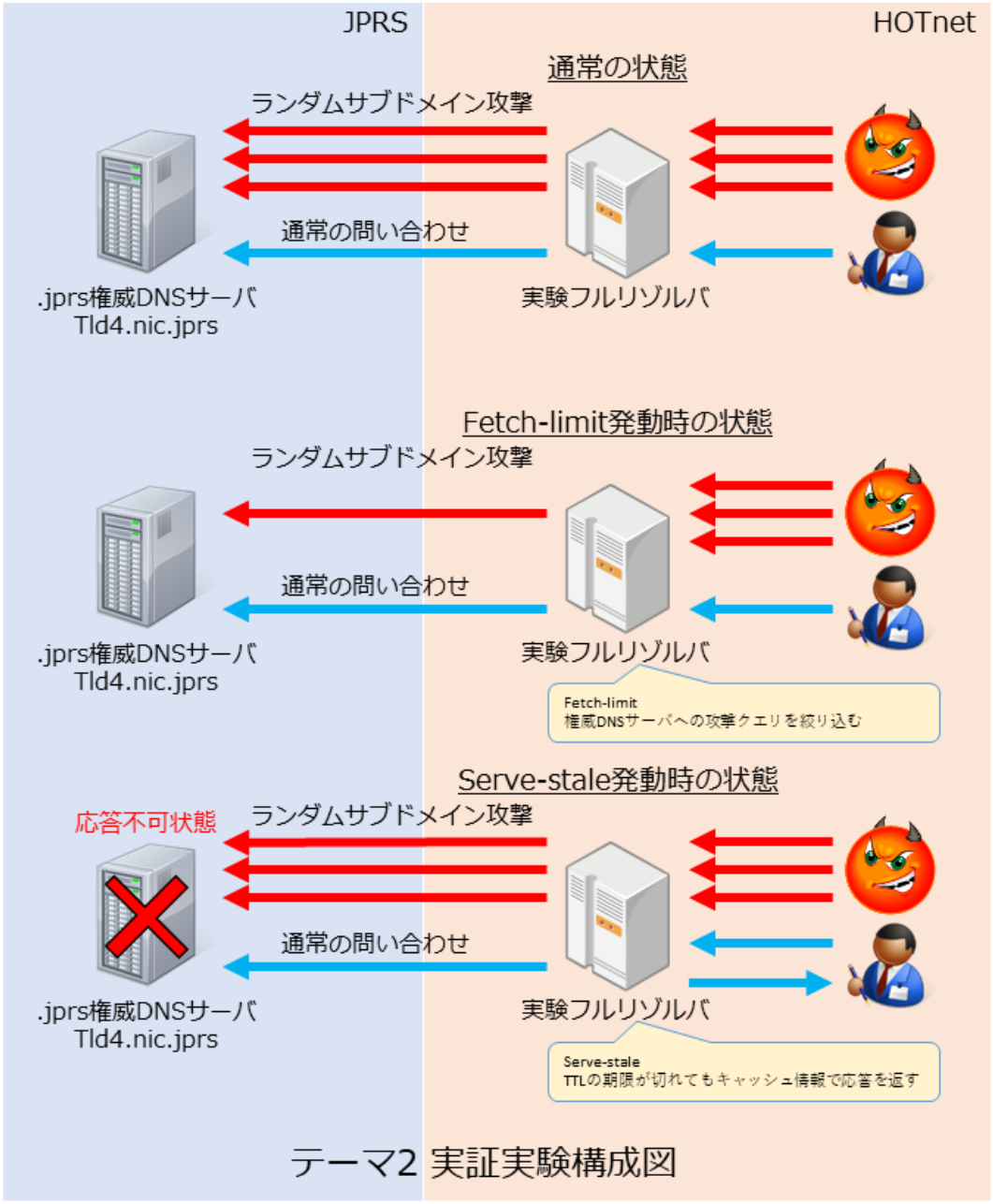



図 3. テーマ 2 実証実験構成図

- HOTnet フルリゾルバー
  - OS：CentOS 7.3

- DNS：BIND 9.14.5

- named.conf へ以下の設定を行った。
  - fetch-limit 設定値：**`fetches-per-zone 10 fail`**
  - serve-stale 設定値：**`stale-answer-enable yes`**
    **`max-stale-ttl 86400`**

図 3. に示す通り、TLD へランダムサブドメイン攻撃を発生させた。

## 実験結果

1. ランダムサブドメイン攻撃が開始されると TLD DNS サーバーが無応答になることを確認した。
2. fetch-limit 発動後、攻撃クエリを絞り込むことができていることを確認した。
   - 以下のログから、fetch-limit が攻撃対象のドメイン名に対してのみクエリを抑制する設定が機能し、通常のクエリが抑制されないことを確認できた。
   - 03-May-2020 21:30:17.137 spill: info: too many simultaneous fetches for XXXXXXXX.XXXXXXXX.jprs (allowed 10 spilled 1)
3. fetch-limit 発動後、権威 DNS サーバーの応答再開を確認した。
4. serve-stale によって TTL が満了してもキャッシュを破棄せず名前解決の応答を返す想定だったが、名前解決は継続されず serve-stale は機能しなかった。
   - キャッシュが満了したタイミングから status: SERVFAIL となり、期待した結果が得られなかったことから判断した。
5. 攻撃発生中、TLD の復旧作業として TLD DNS サーバーの IP アドレスの変更を実施した。
   - TLD DNS サーバーの IP アドレス変更後、フルリゾルバーにてキャッシュのクリアを実施した。
   - TLD DNS サーバーの IP アドレスが変更された後、TLD DNS サーバーからフルリゾルバーへの名前解決の応答が返るようになったことを確認できた。

## 考察

- fetch limit の動作を確認し、ランダムサブドメイン攻撃に対し攻撃クエリを絞り込むという期待した効果を今回の設定で得られることが確認できた。
- fetch limit の効果が発動した際の出力ログについての知見が得られた。
- fetch-limit の設定値について

- 本番運用を想定した場合、設定値を最適化することでより有効な効果が得られることを期待している。
- 適切な設定値については今回採用した設定値での運用から知見をためる必要があると考えている。
- 本実験において、攻撃によって無応答となった TLD DNS サーバーが復旧のために IP アドレス変更で対応した場合は、フルリゾルバー側の迅速なキャッシュクリアが求められた。
- 迅速なサービス復旧のためには、事業者間の密な情報連携が求められるということを本実証実験で体感することができた。

**今後の課題**

- fetch-limit については想定通り、特定のドメインについて自組織からの問い合わせ量の絞り込みが可能となっていることが確認できた。
- しかし、その動作によって自社顧客の通信が制限されてはならないため、導入の際は慎重な検証が必要であると考える。
- fetch-limit 機能によってどの程度サーバーリソースが守られたかについては、本実験では確認できていない。 -当該機能を本番環境に実装・導入するには、サーバーリソースと fetch-limit の効果の関連性の更なるデータの収集・分析が必要である。
- serve-stale 機能については想定外の動作となったため、動作について理解を深める追試が必要である。
- serve-stale 機能のみ利用する場合の動作について実験することで、機能の単体としての動作を確認することができると考えている。

# 実証実験報告（北陸通信ネットワーク株式会社）

## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### 目的

ランダムサブドメイン攻撃検知と ISP 間の情報共有を兼ねた攻撃検知システムの動作評価

### 構成

CentOS Linux release 7.7.1908(Core) -BIND 9.11.10 & BIND 9.14.5

### 実施日時

- シナリオ 1
  - ランダムサブドメイン攻撃時間:2020/02/05 11:05:45 - 11:13:36
  - 攻撃アラート発生メール受信:2020/02/05 11:11:32
- シナリオ 2
  - ランダムサブドメイン攻撃時間:2020/02/05 11:36:45 - 11:42:37
  - 攻撃アラート発生メール受信:2020/02/05 11:38:33

### 実施日時

各 ISP より疑似ランダムサブドメイン攻撃(dnsperf を使用)を行い、攻撃検知システムで検知した内容を JPRS オペレーターもしくは、システムからの自動メールにより、各 ISP に通知を行った。

シナリオ 1

1. tld4.nic.jprs に対して 50 クエリ /秒の攻撃開始
2. 攻撃検知システムで検知した内容を JPRS(オペレーター)が Slack もしくはメールで各 ISP に連絡
3. 攻撃検知システムにてランダムサブドメイン攻撃グラフ上昇、及びクエリログで QNAME を確認
4. tld4.nic.jprs に対しての攻撃停止

5. 攻撃検知システムで検知した内容を JPRS(オペレーター)が Slack もしくはメールで各 ISP に連絡
6. 攻撃検知システムにてランダムサブドメイン攻撃グラフ下降を確認

シナリオ 2

1. tld4.nic.jprs に対して 50 クエリ/秒の攻撃開始
2. 攻撃検知システムが検知した内容を Slack もしくはメールで各 ISP に連絡
3. 攻撃検知システムにてランダムサブドメイン攻撃グラフ上昇、及びクエリログで QNAME を確認
4. tld4.nic.jprs に対しての攻撃停止
5. 攻撃検知システムが検知した内容を Slack もしくはメールで各 ISP に連絡
6. 攻撃検知システムにてランダムサブドメイン攻撃グラフ下降を確認

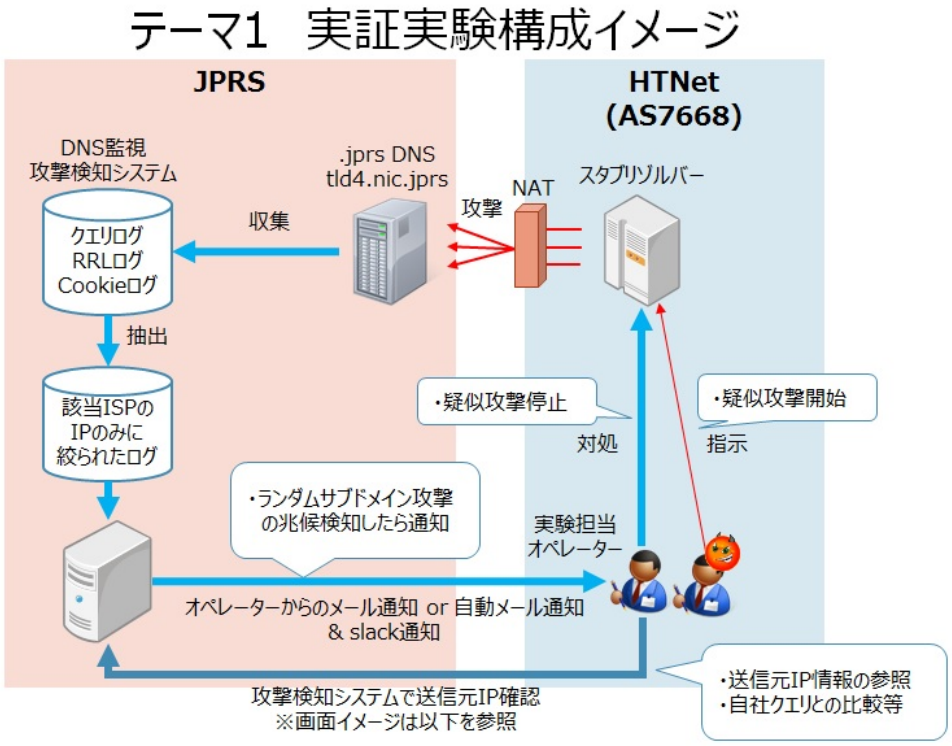



図: テーマ 1 実証実験構成イメージ


図: 攻撃検知システム UI イメージ

## 結果

シナリオ 1,2 ともに、疑似ランダムサブドメイン攻撃時におけるメール受信、及び攻撃対象の IP アドレスを攻撃検知システムで確認することができた。

## 知見

- 一度オペレーターが介在するほうが、運用者として対処はしやすい。
- 攻撃検知システムからの直接通知であれば迅速に通知を行えることができるが、誤検知の恐れや機械的なメール文面から攻撃と判断するのは難しい。

## 課題

- 今回は攻撃の通知までの検証であるが、攻撃を早く止めるための対策(テーマ 2)や対応など、実験のポイントを事前に整理しておく必要がある。
  - 対象への通信遮断処置を行うのは難しく、実際には注意喚起程度になり、情報元開示有無などどこまで情報を開示するか整理が必要と考えら考えられる。
- 攻撃検知の閾値を定める必要がある。
  - 実験における各社で設定した攻撃への閾値が統一されていなかったため、ある程度統一する必要があると考えられる。
- ISP 同士の接続点におけるフィルタ実施などの対応策も有効と考えられ、そのための ISP 間の情報連携も必要と考えられる。

## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

DNS(BIND/Unbound)に実装されている技術(fetch-limit/serve-stale)を使った DDoS 対策・緩和策評価

### 構成

CentOS Linux release 7.7.1908(Core) -BIND 9.11.10 & BIND 9.14.5

### 実施日時

実施日時 -2020/03/31 13:10 - 14:10

### 実施内容

1. tld4.nic.jprs に対して攻撃開始
2. ISP フルリゾルバーにて fetch-limit 設定
3. ISP フルリゾルバーにて serve-stale 設定

4. tld4.nic.jprs にて A リソースレコード変更 5.ISP フルリゾルバーにてキャッシュ情報クリア実施

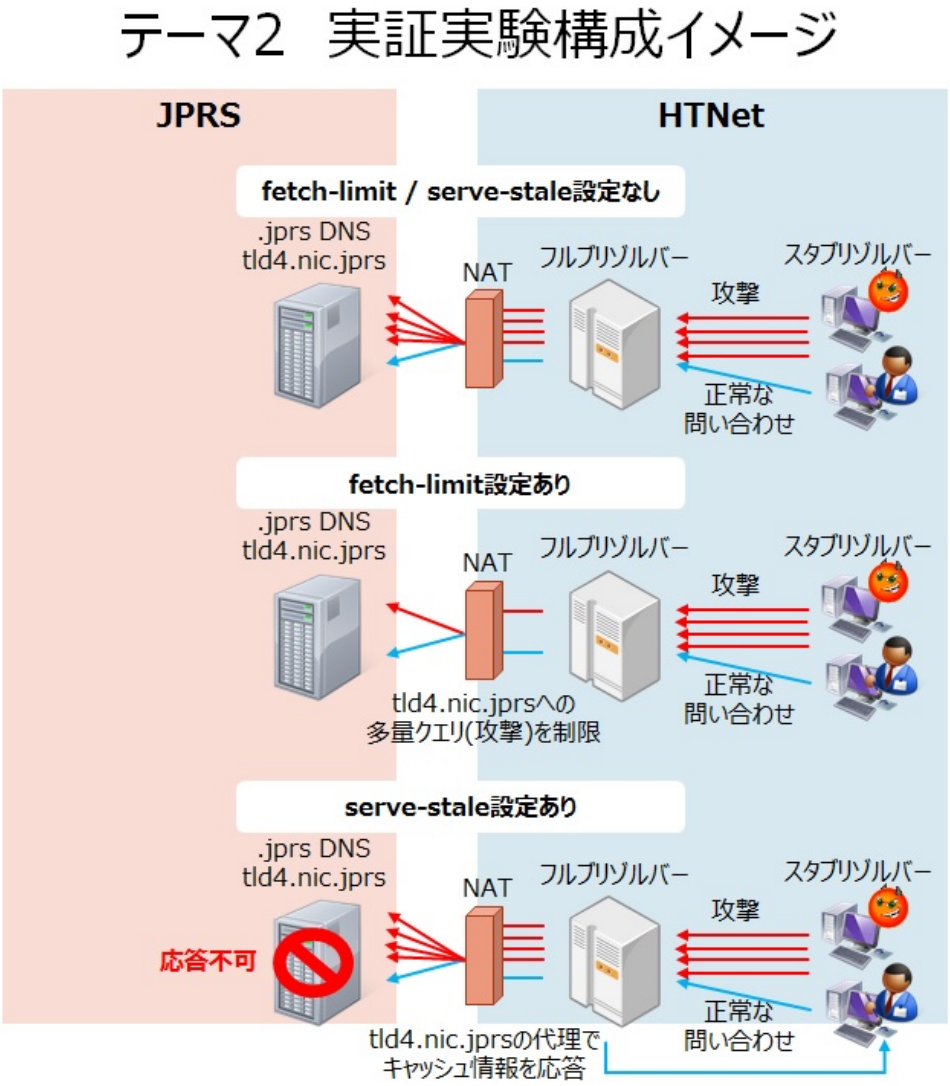



図: テーマ 2 実証実験構成イメージ

## 結果

- fetch-limit
  - fetch-limit 発動時に以下のログを確認できた。

```
31-Mar-2020 13:17:14.603 spill: info: too many simultaneous fetches for
```

```
victim.RD2020-THEME2.jprs (allowed 10 spilled 1)
```

- serve-stale

TTL が切れた後、応答がなくなることを確認できた。 キャッシュクリア後に別 IP アドレスでの応答を確認できた。

-応答あり

```
---
2020-03-31 13:34:34

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> +retry=0 +timeout=1 +ig
nore +notcp @127.0.0.1 -t A www.victim.rd2020-theme2.jprs
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 58212
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN  A

;; ANSWER SECTION:
www.victim.RD2020-THEME2.jprs. 1 IN A   198.51.100.80

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: 火  3月 31 13:34:34 JST 2020
;; MSG SIZE  rcvd: 103
```

-TTL 切れ、応答なし

```
---
2020-03-31 13:34:35 - 14:02:51

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> +retry=0 +timeout=1 +ig
nore +notcp @127.0.0.1 -t A www.victim.rd2020-theme2.jprs
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 55812
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1
```

```
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN  A

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: 火  3月 31 13:34:35 JST 2020
;; MSG SIZE  rcvd: 58
```

-キャッシュクリア実施

```
---
2020-03-31 14:02:52

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> +retry=0 +timeout=1 +ig
nore +notcp @127.0.0.1 -t A www.victim.rd2020-theme2.jprs
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60535
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN  A

;; ANSWER SECTION:
www.victim.rd2020-theme2.jprs. 30 IN    A   203.0.113.80

;; Query time: 81 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: 火  3月 31 14:02:52 JST 2020
;; MSG SIZE  rcvd: 74
```

## 知見

- fetch-limit/serve-stale の動作を知ることが出来た。
- serve-stale でフルリゾルバーからのキャッシュ情報を確認することが出来なかった。

- テーマ 1 の攻撃通知と fetch-limit/serve-stale 機能を実装できれば早急な対応が可能であり、運用における対応フローを確立できる感触を得た。

## 課題

- キャッシュクリアをどのタイミングで行うかについて整理しておく必要があると考えられる。
- fetch-limit/serve-stale ともに知見が少ない機能であるため、運用者がトラブルシュートを円滑に行うために詳細な挙動や設定値について、今後確認していく必要があると考えられる。
- “stale-answer-enable yes”かつ“max-stale-ttl 86400”が設定しているにも関わらず、キャッシュ情報を確認できなかった原因について、特定が必要であると考えられる。

# 実証実験報告（ソフトバンク株式会社）

## 参加の目的

- サイバー攻撃対策の有効性の確認
- 他社担当者との情報交換

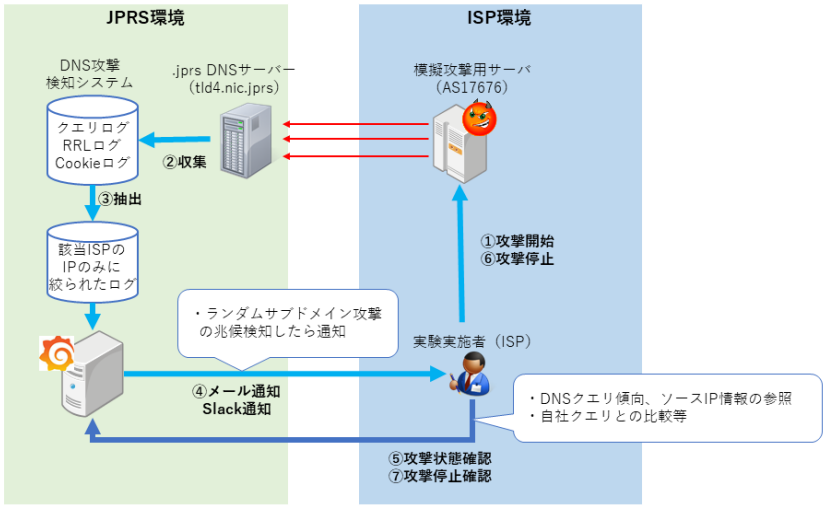
## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### 目的

-攻撃受信の確認　-受信時のアクションの確認

### 構成

テーマ1　実証実験全体構成



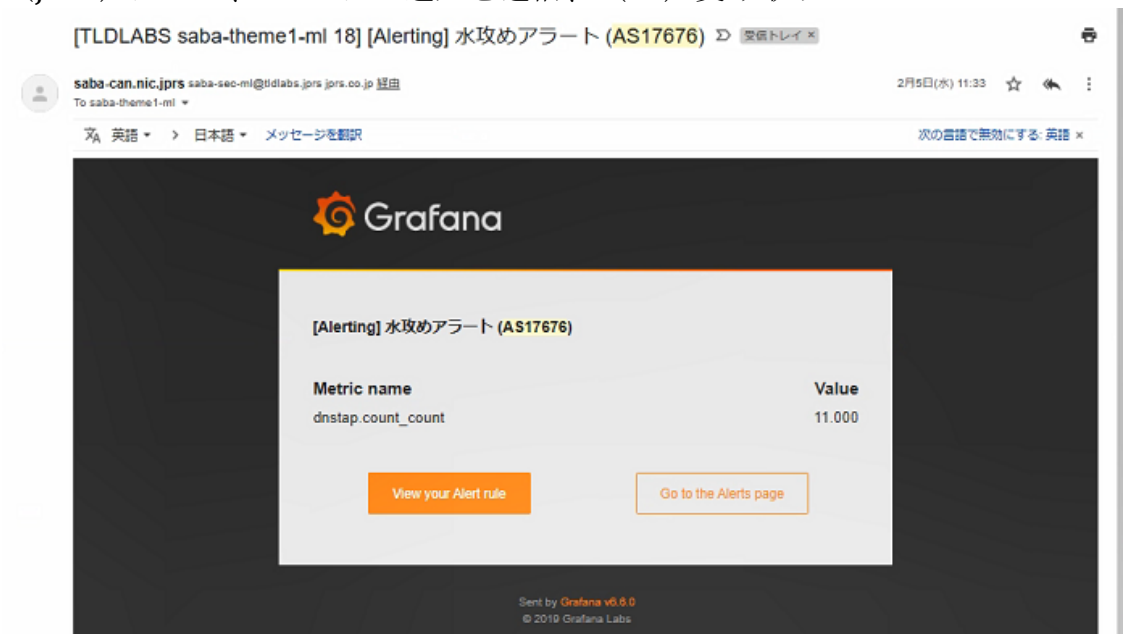

### 実施内容・結果

①　（SB）dnsperf コマンドで tld4.nic.jprs に 50 クエリ/秒程度の疑似攻撃を発生

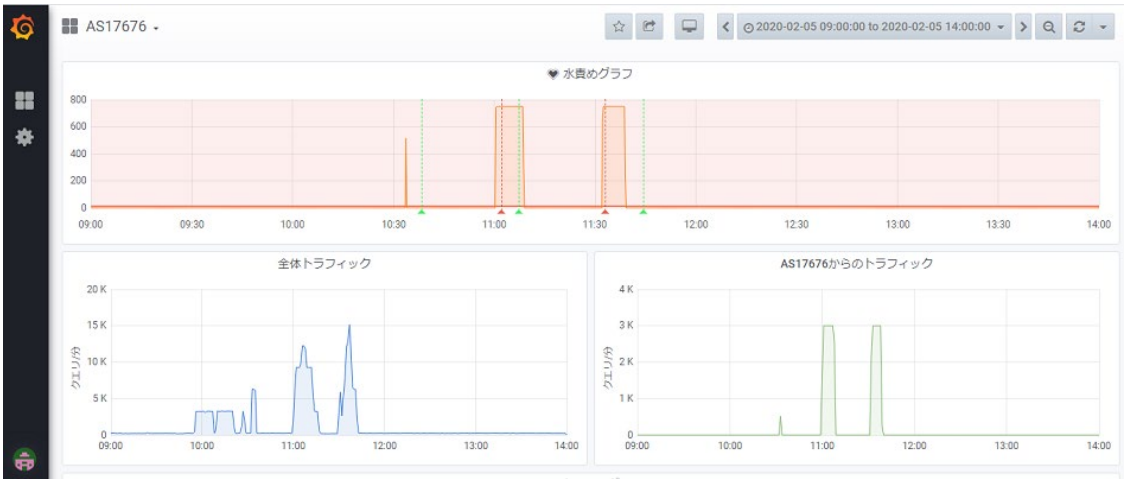
②　（JPRS）TLD DNS サーバーより攻撃情報収集

③　（JPRS）攻撃検知システムにて攻撃情報を抽出

④　（JPRS）メール、Slack にて通知を送信、（SB）受け取り




⑤　(SB) 攻撃検知システムのダッシュボードにて攻撃送信状況の確認、攻撃状況、送信元 IP アドレスを認識




クエリログ

| 時刻 | QNAME | 送信元アドレス |
|---|---|---|
| 2020-02-05 11:39:05.377 | dns20407.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.357 | dns20406.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.337 | dns20405.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.317 | dns20404.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.297 | dns20403.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.277 | dns20402.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.257 | dns20401.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.237 | dns20400.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.217 | dns20399.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.197 | dns20398.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.177 | dns20397.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.157 | dns20396.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.137 | dns20395.example1.jprs. | 221.111.69.22 |
| 2020-02-05 11:39:05.117 | dns20394.example1.jprs. | 221.111.69.22 |

⑥　(SB)模擬攻撃を停止

⑦　(SB) 攻撃検知システムのダッシュボードにて攻撃送信状況の確認、攻撃が停止していることを認識

## 考察

一連の流れとして、想定通りのフローを確認出来た。今回の実験では、攻撃停止は攻撃元で行ったが、送信元 IP アドレスが多種である等、攻撃元で停止が容易に行えない状況であった場合を踏まえ、TLD 側へ攻撃対処を依頼し停止してもらえるようなフローについても検討したい。　今後はこの仕組みを JPRS⇔各 ISP だけでなく、権威 DNS サーバーを保有する ISP⇔ISP 間においても情報共有されることが望ましい。そのため、当システムのような監視システムを容易に導入できるようアプリケーション化し、可能な限り共通な監視、自動で行われる仕組みを検討することを今後の課題としたい。

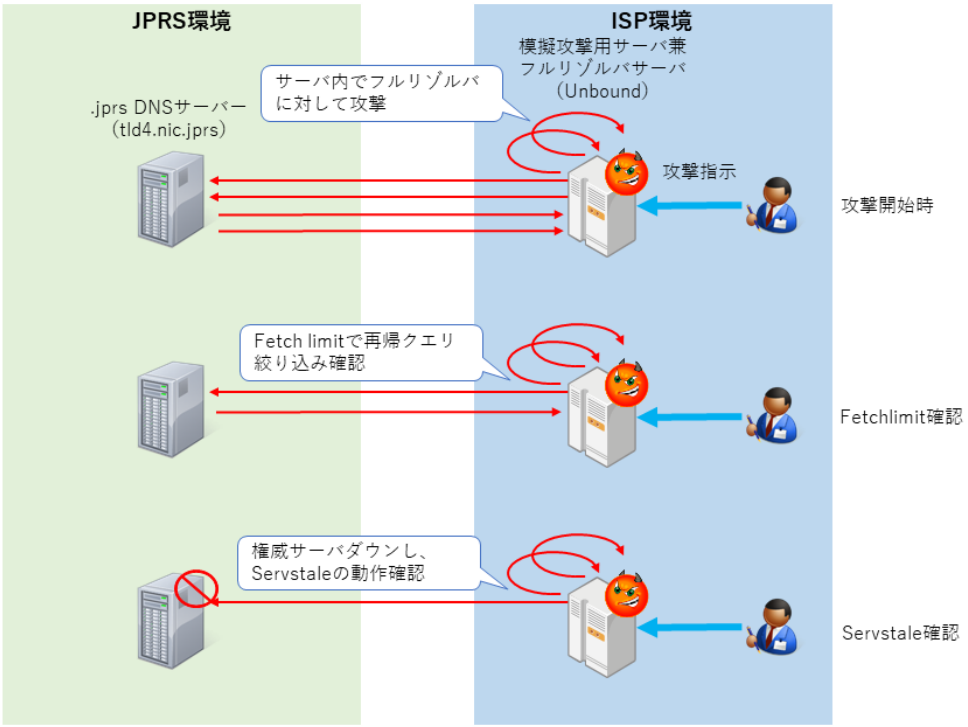
## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

- 攻撃緩和の対策として有効性の確認
- 機能有効時の振る舞いの確認

### 構成

テーマ2　実証実験全体構成




- OS：CentOS7.4
- CPU：2Core（VM）
- Memory：4GB
- DNS：unbound1.9.3

## 実施内容

1. 疑似攻撃送信
2. 権威サーバー停止
3. fetch-limit 発動、動作確認
    - ログより、notice: ratelimit exceeded の文言を確認し、fetchlimit が動作していることを確認。
4. TTL が満了し、serve-stale が発動、動作確認
    - Unbound ではログにて serve-stale 中のクエリ応答状況を確認できなかった。
    - 後日 JPRS にて調査頂いた結果、total.num.expired の値が該当することが判明。ただし、どのドメインが serve-stale の対象となっているかは確認できなかった。
5. 権威サーバーにてレコード変更
6. キャッシュクリアにて、レコードが変更されることを確認
    - キャッシュクリアしたタイミングより、レコードが更新されていることを確認。

## 考察

DDos 攻撃に対し、権威サーバー側の停止により特定ドメイン名のサービス継続が行えることを確認できた。この機能を活用することで、サービスの継続性が向上する。

今回の実証で fetch-limit が動作したことは確認出来たが、大量のクエリに対する DNS サーバー自体の負荷軽減の効果までは確認できておらず、サーバー負荷対策としての効果確認は今後の課題となる。

serve-stale については、権威サーバーの IP アドレスを変更した際に能動的にキャッシュを消す必要があり、復旧に時間がかかるという点はデメリットであると感じた。

本実験はあくまで、インターネット上の権威サーバーに問題があった際の自社内における名前解決サービスの継続性が目的であり、その品質向上では無いため、積極的に導入を進めていくかについては、更なる検討が必要である。

# 実証実験報告（フリービット株式会社）

## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### 目的

DNS サーバーへの攻撃のひとつとしてランダムサブドメイン攻撃が知られている。この攻撃の特徴として、攻撃の影響が攻撃対象のドメイン名の TLD DNS サーバーにも及ぶことが挙げられる。そのため、この特徴を利用した TLD DNS サーバーにおける攻撃の検知が可能になると考えられる。

本テーマでは、TLD DNS オペレーターと ISP オペレーターの連携による以下の対応が可能であるかを、疑似攻撃を用いて検証することを目的としている。

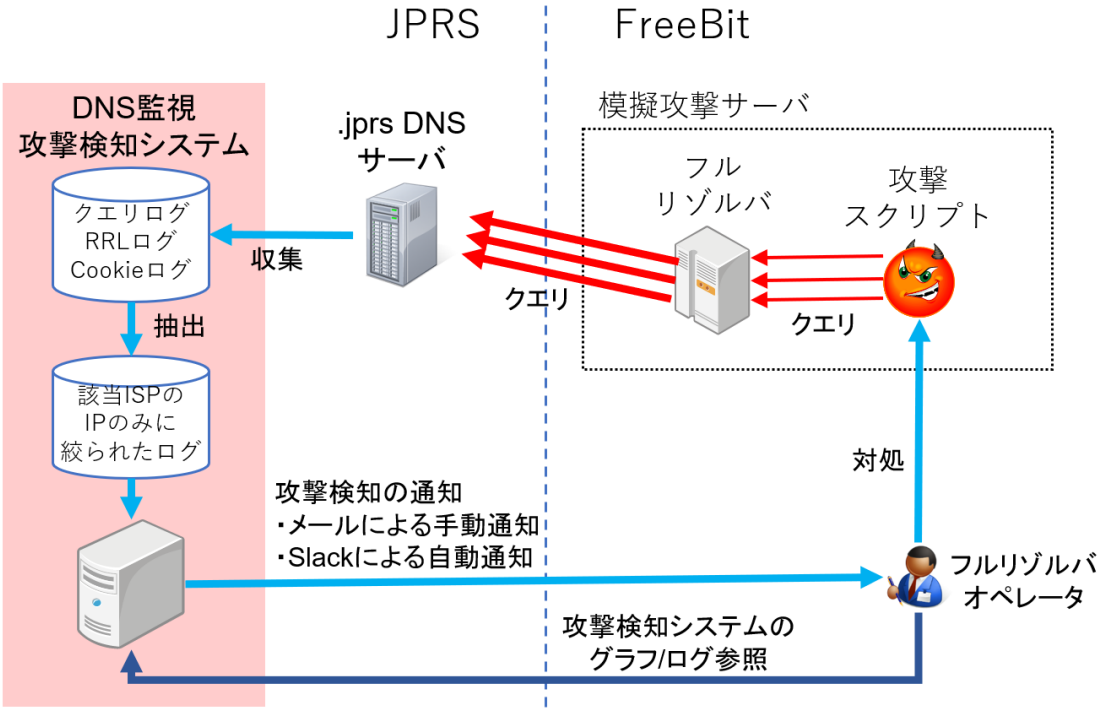
- TLD DNS サーバーにおける攻撃の早期検知
- 組織間での情報共有
- 攻撃への対処

### 実験環境

表: 議事攻撃発生環境

| 項目 | 仕様 |
|---|---|
| 機種 | VMware 上の x64 仮想マシン |
| CPU 数 | 1 |
| メモリサイズ | 1GB |
| ディスクサイズ | 20GB |
| OS | CentOS7 |
| DNS サーバーソフトウェア | BIND 9.14.5 |
| DNS クエリコマンド | DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 |
| DNS 負荷発生ソフトウェア | dnsperf 2.3.2 |
| 攻撃模擬スクリプト | JPRS より提供されたスクリプト |

## システム構成




*テーマ 1 システム構成図*

## 連絡手段

JPRS より提示された連絡手段の選択肢は、メールと Slack であった。Slack を利用するには環境の準備が必要であったため、即座に利用可能なメールを採用した。

## 実験日時

| シナリオ | 日付 | 開始時刻 | 終了時刻 |
|---|---|---|---|
| 1 | 2020/2/7 | 11:04:14 | 11:15:00 |
| 2 | 2020/2/7 | 11:33:23 | 11:46:27 |

## 実験結果

### シナリオ 1 の実験進行

| 時刻 | 作業内容 | 想定進行との一致 |
|---|---|---|
| 11:01 | 実験着手 | ○ |
| 11:06 | 攻撃スクリプト起動 | ○ |
| 11:13 | メールで攻撃検知通知を受信 | ○ |

| | | |
|---|---|---|
| 11:15 | トラフィック停止 | ○ |
| 11:19 | メールで停止報告 | ○ |
| 11:21 | メールで攻撃終息通知を受信 | ○ |
| 11:25 | 実験終了 | ○ |

### シナリオ 2 の実験進行

| 時刻 | 作業内容 | 想定進行との一致 |
|---|---|---|
| 11:30 | 実験着手 | ○ |
| 11:33 | 攻撃スクリプト起動 | ○ |
| 11:44 | メールで攻撃検知通知を受信 | ○ |
| 11:46 | トラフィック停止 | ○ |
| 11:53 | メールで攻撃終息通知を受信 | ○ |
| 11:53 | 実験終了 | ○ |

## 得られた知見

### JPRS から得られた情報の内容

攻撃検知システムのダッシュボードで状態の検知とクエリ量の時系列グラフを確認でき、グラフのピークから、攻撃の状況を把握できた。

### JPRS から得られた情報を使って調査できたこと

- テストシナリオと自社業務形態との整合性
- 以下の理由により、シナリオ 2 が当社業務との親和性が高いと判断した。
  - 定型文面により、自動処理のトリガーとなるメッセージを作成しやすいこと
  - 担当者が手動で対応した場合、文面に誤りが混入する可能性があること

## 今後の課題

- 実際に攻撃検知情報を受け取った際に実現可能と考えられる課題
  - 監視台からサービス担当にエスカレーションし、攻撃元を特定
  - 攻撃元の性質により制御可能な箇所がある場合におけるフィルタの設定
- 実際に攻撃検知情報を受け取った際に現時点では実現が困難と考えられる課題
  - 現実に発生するランダムサブドメイン攻撃では、権威 DNS サーバー及び付随する検知 システムには、フルリゾルバーの IP アドレスだけが見える

形になる。そのため、連絡を受けた ISP ではフルリゾルバーのクエリログから本来の攻撃元を探索しなければならない。実環境においてそれが可能かどうか。

- 攻撃によりアクセス量が増加した場合、生のクエリログでは大量かつ高速に流れてしまう状況となるため、ログを目で追えない(そもそも表示が追い付かない)のではないかという懸念がある。

- 攻撃検知システムによる調査・対処にあたり、あると有用であったと考えられる情報
  - クエリログの度数分布の集計
- 攻撃検知システムによる調査・対処にあたり、より詳細に見られたほうがよかった情報
  - ISP のフルリゾルバーの背後に存在する真の攻撃元を探しやすくするため、ランダムサブドメイン攻撃の対象となっているドメイン名を判別可能にするための統計処理(ホスト名部分のバリエーション数、及びクエリ数でソートした攻撃対象ドメイン名ランキング)が見られるようになっているとよい。

## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

DNS サーバーへの攻撃のひとつとしてランダムサブドメイン攻撃が知られている。この攻撃は、ボットネットを構成するクライアントから最寄りのフルリゾルバーを経由して攻撃対象ドメインの権威 DNS サーバーに大量のクエリを送り付けてサービス不能状態にすることを目的としていると考えられている。かつ、攻撃の踏み台にされるフルリゾルバーも未完了の再帰問い合わせによって大量の状態を抱えさせられ、サービス不能状態に至るという問題がある。

本テーマでは、DNS サーバーソフトウェアに備わった以下の攻撃対策機能について、ランダムサブドメイン攻撃に対する有効性を検証する。

| 機能名 | 機能の内容 |
|---|---|
| fetch-limit | 同一問い合わせ元からの時間窓当たりのクエリ数に上限を設け、これを超える問い合わせを破棄する |
| serve-stale | 再帰問い合わせを要求されたレコードについて、すべての権威 DNS サーバーから応答が得られない場合、TTL が満了したキャッシュの内容をクライアントに応答する |

### 実験環境

#### サーバー

| 項目 | 仕様 |
|---|---|
| 機種 | VMware 上の x64 仮想マシン |
| CPU 数 | 1 |
| メモリサイズ | 1GB |
| ディスクサイズ | 20GB |
| OS | CentOS7 |
| DNS サーバーソフトウェア | BIND 9.14.5 |
| DNS クエリコマンド | DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 |
| DNS 負荷発生ソフトウェア | dnsperf 2.3.2 |
| 攻撃模擬スクリプト | JPRS より提供されたスクリプト |

ネットワーク

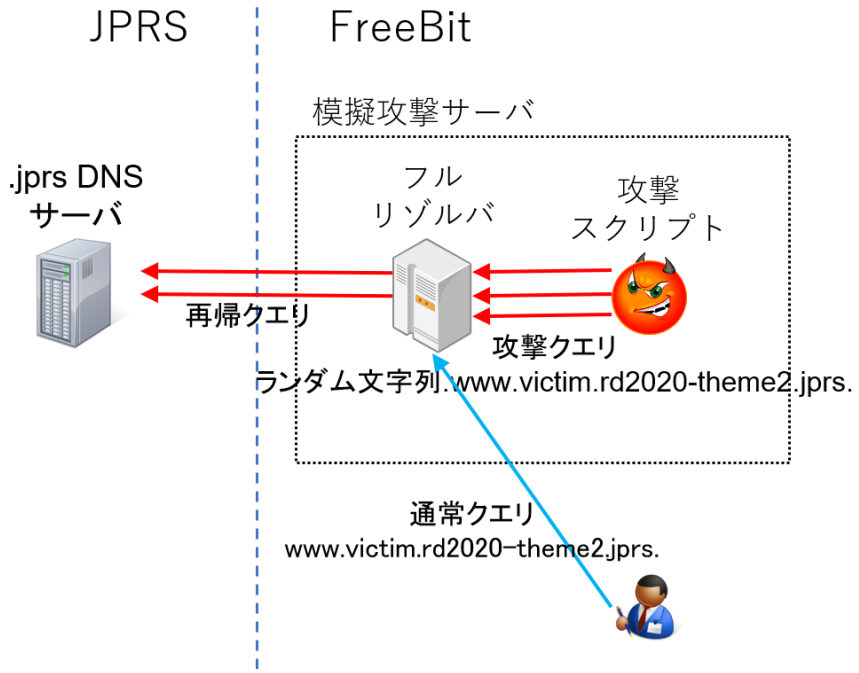



*図: テーマ2 システム構成図*

**実験日時**

| 日時 | 内容 |
|---|---|
| 2020-03-17 14:00 | 開始 |
| 2020-03-17 15:30 | 終了 |

**実験結果**

- step 1: フルリゾルバーと疑似クライアント起動
  - コンソール出力(疑似クライアント)

```
2020-03-17 14:13:40

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> +retry=0 +timeout=1
+ignore +notcp @127.0.0.1 -t A www.victim.rd2020-theme2.jprs
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 65137
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
```

```
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN      A

;; ANSWER SECTION:
www.victim.RD2020-THEME2.jprs. 240 IN   A       198.51.100.80

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Mar 17 14:13:40 JST 2020
;; MSG SIZE  rcvd: 103
```

- コンソール出力(動作確認)

```
[tmiura@cbctest-monit1 ~]$ dig @::1 www.victim.rd2020-theme2.jprs.

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> @::1 www.victim.rd20
20-theme2.jprs.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4328
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN      A

;; ANSWER SECTION:
www.victim.RD2020-THEME2.jprs. 217 IN   A       198.51.100.80

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Tue Mar 17 14:19:01 JST 2020
;; MSG SIZE  rcvd: 103
```

- 想定との差異
  - なし
- step 2: 疑似攻撃ツール起動・水責め攻撃疑似発生
  - コンソール出力(attacker.log)

```
DNS Performance Testing Tool
Version 2.3.2

[Status] Command line: dnsperf -s 127.0.0.1 -S 1 -d /tmp/random
```

```
_query.txt -Q 100
[Status] Sending queries (to 127.0.0.1)
[Status] Started at: Tue Mar 17 14:34:34 2020
[Status] Stopping after 1 run through file
1584423275.767656: 91.859730
1584423276.768651: 99.900599
    :
```

- コンソール出力(named.log の変わり目)

```
17-Mar-2020 14:12:39.786 lame-servers: info: network unreachabl
e resolving '_.rd2020-theme2.jprs/A/IN': 2a01:8840:1ba::1#53
17-Mar-2020 14:12:39.866 lame-servers: info: network unreachabl
e resolving 'ns.victim.rd2020-theme2.jprs/AAAA/IN': 2001:dc3::3
5#53
17-Mar-2020 14:34:34.857 spill: info: too many simultaneous fet
ches for victim.RD2020-THEME2.jprs (allowed 10 spilled 1)
17-Mar-2020 14:34:44.768 serve-stale: info: random0000001.www.v
ictim.rd2020-theme2.jprs resolver failure, stale answer unavail
able
```

- コンソール出力(問合せ結果)

```
[tmiura@cbctest-monit1 attacker]$ date ; dig @::1 www.victim.rd
2020-theme2.jprs.
Tue Mar 17 14:38:33 JST 2020

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> @::1 www.victi
m.rd2020-theme2.jprs.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 51564
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITION
AL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN      A

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Tue Mar 17 14:38:33 JST 2020
;; MSG SIZE  rcvd: 58
```

- 想定との差異

- SERVFAIL となった。これは、serve-stale より先に fetch-limit が効いてしまっていると解釈される。
- step 3: キャッシュ TTL の満了

- コンソール出力(問合せ結果)

```
[tmiura@cbctest-monit1 attacker]$ date ; dig @::1 www.victim.rd2020-t
heme2.jprs.
Tue Mar 17 14:46:50 JST 2020

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> @::1 www.victim.rd20
20-theme2.jprs.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 58631
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN      A

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Tue Mar 17 14:46:50 JST 2020
;; MSG SIZE  rcvd: 58
```

- コンソール出力(/var/saba-theme2/fullresolver2/named.recursing 抜粋)

```
victim.RD2020-THEME2.jprs.: 10 active (93145 spilled, 1910 allowed)
```

8. コンソール出力(/var/saba-theme2/fullresolver2/named_dump.db 抜粋)

```
; authanswer
; stale (will be retained for 85594 more seconds)
www.victim.RD2020-THEME2.jprs. 0 A      198.51.100.80
```

9. コンソール出力(/var/saba-theme2/fullresolver2/named.stats 抜粋)

```
1094 attempts to use stale cache data after lookup failure
```

10. コンソール出力(named.log 抜粋)

```
17-Mar-2020 14:54:59.879 serve-stale: info: www.victim.rd2020-theme2.
jprs resolver failure, stale answer used
```

11. コンソール出力(まれに応答が返っている例)

```
2020-03-17 14:37:32

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> +retry=0 +timeout=1
+ignore +notcp @127.0.0.1 -t
A www.victim.rd2020-theme2.jprs
; (1 server found)
```

```
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28757
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN      A

;; ANSWER SECTION:
www.victim.RD2020-THEME2.jprs. 1 IN     A       198.51.100.80

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Mar 17 14:37:32 JST 2020
;; MSG SIZE  rcvd: 103
```

12. 想定との差異
    - SERVFAIL となるような情報をキャッシュしているように見える。
    - しかし、named_dump.db の SERVFAIL cache セクションは空であり、少なくともダンプされる情報としてキャッシュされている情報は異なるように見える。
    - step 4: オリジン IP アドレス変更
13. コンソール出力(問合せ結果)

```
[tmiura@cbctest-monit1 attacker]$ date ; dig +timeout=10 @::1 www.vic
tim.rd2020-theme2.jprs.
Tue Mar 17 15:02:30 JST 2020

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> +timeout=10 @::1 ww
w.victim.rd2020-theme2.jprs.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 28866
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN      A

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Tue Mar 17 15:02:30 JST 2020
```

```
;; MSG SIZE  rcvd: 58
```

14. 想定との差異
    - オリジン側の変更だけでは問合せ結果は自動的には変化しない。
    - step 5: キャッシュクリア
15. コンソール出力(問合せ結果)

```
[tmiura@cbctest-monit1 attacker]$ date ; dig +timeout=10 @::1 www.vic
tim.rd2020-theme2.jprs.      Tue Mar 17 15:04:05 JST 2020

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> +timeout=10 @::1 ww
w.victim.rd2020-theme2.jprs.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54195
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.victim.rd2020-theme2.jprs. IN      A

;; ANSWER SECTION:
www.victim.RD2020-THEME2.jprs. 24 IN    A       203.0.113.80

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Tue Mar 17 15:04:06 JST 2020
;; MSG SIZE  rcvd: 103
```

16. 想定との差異
    - フルリゾルバーの再起動により、応答が想定内容に変わった。
    - client.query.log における 15:04 の問合せが、応答内容が変わった最初の問合せであった。
    - step 6: 実験終了
    - fetch-limit の効果
17. 確認できた。
18. 但し、ほとんどの問い合わせが破棄されており、ほぼ SERVFAIL しか返らなくなった。
    - serve-stale の効果
19. 確認できなかった。終始 SERVFAIL しか返らなかった。
20. しかし、フルリゾルバーのキャッシュダンプを取得すると、stale キャッシュエントリは存在した。

- 想定と異なる点

21. fetch-limit と serve-stale の双方が成立する条件下では、fetch-limit の効果のみが確認出来、serve-stale の効果は確認できなかった。
22. この結果の理由として、BIND における応答のロジックがそのような構造になっているためであると推測した。

## 得られた知見

23. 機能に対する理解
    - BIND の fetch-limit オプションの機能がランダムサブドメイン攻撃に対し、どのような挙動をするかを理解できた。
24. 想定通りに動作したこと
    - BIND の fetch-limit オプションの効果が想定通りであることを確認できた。これにより、fetch-limit オプションがランダムサブドメイン攻撃の対策として有効に機能することが理解できた。
25. 想定とは異なる動きをしたこと
    - BIND の serve-stale オプションの効果は今回の実験を通じて確認することができなかった。これについては、BIND における応答のロジックがその理由であるという仮説を立てることができた。

## 今後の課題

26. 機能的に不足している点
    - serve-stale を fetch-limit と併用した場合、現時点ではランダムサブドメイン攻撃の状況下において、有効な対策として機能しないことが判明した。
27. その解決方法
    - fetch-limit が機能している状況下においても serve-stale が機能するように、BIND の応答のロジックが改修されることを期待したい。

# 実証実験報告（株式会社コミュニティネットワークセンター）

## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### 目的

JPRS が運用する TLD DNS サーバーにて攻撃を検知した場合の通知と対処について検証する。

- 攻撃クエリに連動して通知がされるか
- 対処に必要な情報が通知されるか
- 通知されてから対処する運用が可能か

### 実験環境

#### 疑似攻撃発生環境

28. OS/ディストリビューション: CentOS7
29. ソフトウェア: dnsperf 2.3.2

## 実験概要

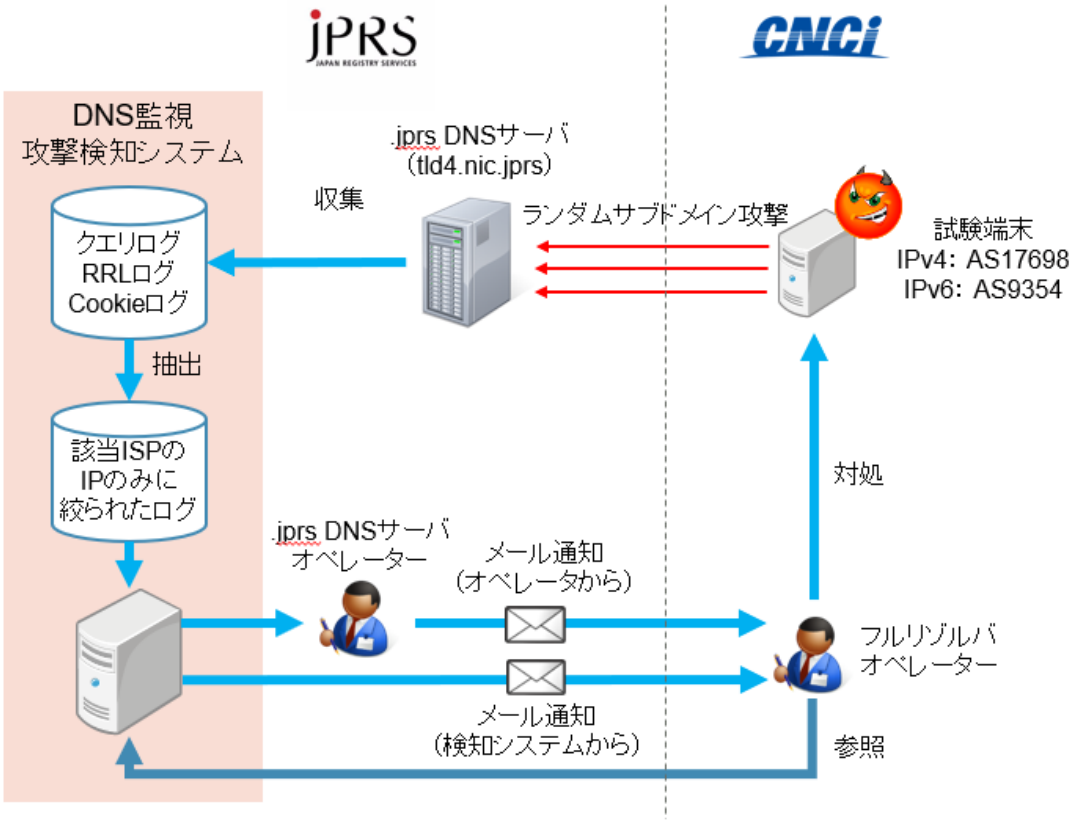



*図: 概要図*

30. デュアルスタックの疑似攻撃発生サーバーより実施。
31. IPv4: AS17698
32. IPv6: AS9354

## 連絡手段

33. メール
34. Slack

## 実験日時

35. シナリオ 1(TLD DNS サーバーのオペレーターから通知): 2020/02/07 11:02:53～11:17:26
36. シナリオ 2(システムから自動通知): 2020/02/07 11:31:27～11:39:56

## 実験結果

37. できたこと

- 異常と復帰の通知を受けることができた。
- 攻撃検知システムにより平常時との違いを認識できた。
- 攻撃検知システムにおいて、IPv4 及び IPv6 で検知していることを確認できた。
- 攻撃検知システムにおいて、具体的な情報(IP アドレス、ドメイン名)を確認できた。

38. できなかったこと
    - 自社以外(他事業者でも発生している問題なのか)の状況把握ができなかった。

## 得られた知見

39. シナリオ 1(TLD DNS サーバーのオペレーターから通知)
    - TLD DNS サーバーのオペレーターが対応するため、通知されるまでに時間がかかる。
    - TLD DNS サーバーのオペレーターが対応するため、通知内容に誤りが発生する可能性がある。
    - オペレーターを経由するため、通知内容に疑問があった場合の連絡が取りやすい。
40. シナリオ 2(システムから自動通知)
    - 自動的に通知されるため、運用における手順書の作成や自動化を図りやすい。

## 今後の課題

41. 実際に通知をトリガーとして対処するためには、運用体制やフローを別途整備する必要がある。
42. 提供を受ける情報について、通信の秘密に関する整理が必要である。
43. 実運用に向け、攻撃検知システムで検知する事象の範囲や精度を向上させる必要である。

## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

ランダムサブドメイン攻撃に対し、fetch-limit と serve-stale が有効であるかについて検証する。

### 実験環境

#### サーバー

44. OS/ディストリビューション: CentOS7
45. DNS サーバーソフトウェア:Unbound 1.9.3
46. dig 9.11.4
47. dnsperf 2.3.2
48. 攻撃模擬スクリプト: JPRS が提供したスクリプトを利用した。

## 実験概要

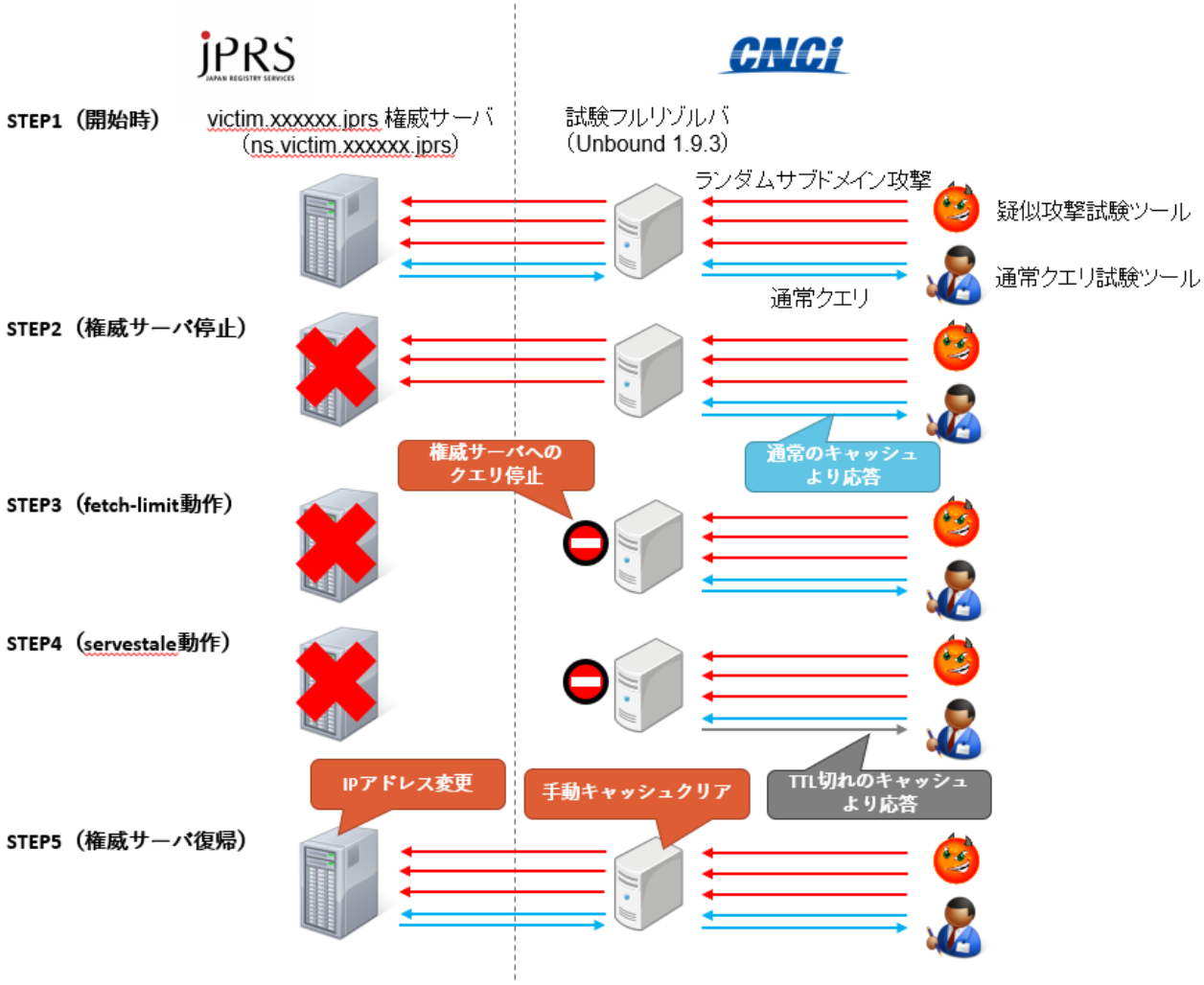



*図: 概要図*

## 実験日時

49. 2020/03/05 13:00 - 14:02

## 実験結果

50. fetch-limit の効果
    - 権威 DNS サーバーへのクエリが停止することが確認できた。
51. serve-stale の効果
    - 保持期間を超過したキャッシュを利用した応答が確認できた。
52. 機能は想定通りに動作した。

### 得られた知見

53. fetch-limit 及び serve-stale の機能について理解できた。
54. 動作状況や顧客からの問い合わせに応じ、手動でキャッシュをクリアする運用は現実的ではない。そのため、権威 DNS サーバーの状況を確認し、キャッシュを自動的にクリアする等の機能が必要になると考えられる。

### 今後の課題

55. 実環境への導入を前提として場合、想定外の動作や運用負荷が発生しないか、引き続き検証が必要である。

# 実証実験報告（株式会社オプテージ）

## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### 目的

TLD DNS サーバーで検知が可能な DNS 攻撃を代表するランダムサブドメイン攻撃に対して攻撃情報の早期検知を行うための仕組み作り、および DNS オペレーター間での情報共有方法を確立するため、ISP 内部からランダムサブドメイン攻撃を模擬した DNS トラフィックを発生させ、早期検知の仕組みの効果や関係組織の連携における課題の確認を行う。

### 疑似攻撃発生環境の構成

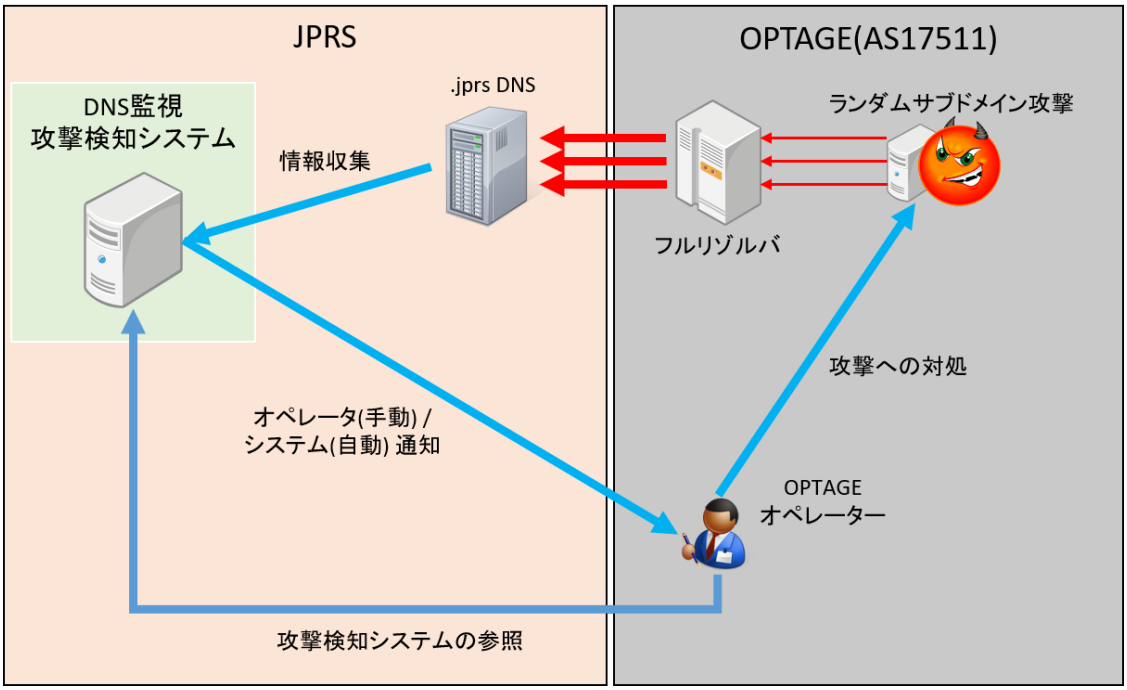
- OS
  - CentOS7.5.1804
- Tools
  - dig 9.11.4 P2
  - dnsperf 2.3.2
  - 攻撃模擬スクリプト: JPRS が作成したスクリプト
- 


図: テーマ 1 実証実験構成図

## 連絡手段

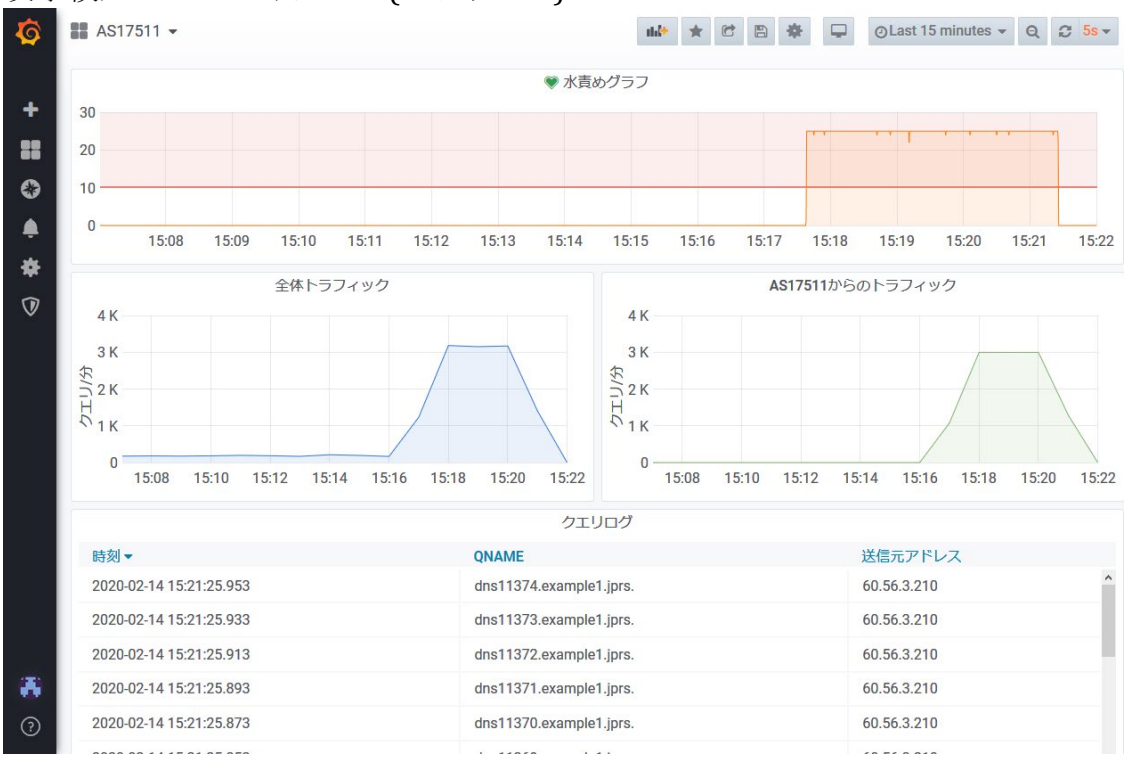
- メール(オペレーターによる手動通知)
- Slack(システムによる自動通知)

## 実験日時

- 実験日：2020/02/07(シナリオ 2),2020/02/14(シナリオ 1※事前設定不備のため別日実施)
- 実験開始・終了時刻
  - シナリオ 1:発生時刻(15:17:00)～停止時刻(15:33:30)
  - シナリオ 2:発生時刻(11:38:00)～停止時刻(11:45:00)

## 実験結果

- 攻撃検知システムイメージ(シナリオ 1)




- 攻撃検知システムで、攻撃を受けている時間帯および送信元 IP アドレスを確認することができた。
- メールと Slack による JPRS との連絡により、攻撃に対処できた。

## 考察

- 攻撃検知システムによりランダムサブドメイン攻撃の発生時刻・収束時刻や送信元 IP アドレスなどの詳細な情報が可視化され、瞬時に把握することができるため、攻撃への迅速な対処が可能であった。
- 攻撃検知システムがなかった場合、状況を把握するまでの時間が DNS オペレーターの経験やスキルに依存するため、当該システムのグラフ表示によってそうした経験やスキルの差を埋めることができ、攻撃の対処に必要な時間が削減できると考えられる。
- 連絡手段として、情報伝達の速さ・手軽さの面で、Slack が有用であると感じた。

## 課題

- 今回の実験では、Slack による情報連携が理想的であると考えられる。しかし、会社として情報を受け取ることを想定した場合、窓口が監視部門となり一次連絡手段がメールあるいは電話になり、メールで通知を受け取るシナリオ 1 が現時点における最適解となる。
- TLD オペレーターと ISP 間、あるいは ISP 同士で恒久的な NDA を締結するなど、共有する情報の範囲を明確にするためのルール作りが必要である。
- 攻撃への対処方法について、攻撃の送信元 IP アドレスからの通信を即時に遮断することが理想ではあるものの、同時に当該顧客へのサービス停止につながるため、実運用における即座な適用は困難であると考えられる。
- 実際に攻撃検知システムを運用した場合、通常のトラフィックと攻撃トラフィックの両方に関する情報が表示されることが想定される。そのため、今回の実験環境ほど容易に状況を把握することは困難であると考えられる。
- 攻撃の検知から攻撃元 IP アドレスの抽出、オペレーターが状況を把握し対処するまでの一連のフローを整理する必要がある。

## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

DNS 攻撃を代表するランダムサブドメイン攻撃に対して、フルリゾルバーから権威 DNS サーバーへの非再帰問い合わせを制限することで攻撃対象ドメイン名の名前解決への影響を軽減する技術(fetch-limit)や、権威 DNS サーバーが攻撃により応答を返せない状況下においてもフルリゾルバーが期限切れのキャッシュを保持し続けることで名前解決が可能となる技術(serve-stale)について、その有効性を確認する。

### テーマ 2 事前検証

事前検証項目

- テーマ 2a: DNS Cookies による DNS amp 攻撃のフィルタ、RRL の適用
- テーマ 2b: fetch-limit, serve-stale によるランダムサブドメイン攻撃の緩和
- テーマ 2c: Aggressive Use of NSEC/NSEC3 によるランダムサブドメイン攻撃の緩和(Unbound では未実施)

### テーマ 2a

- DNS Cookies

```
・Stub resolver console log
# Server Cookie を得る
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +norec +nobadcookie +coo
kie @192.0.2.1 -t SOA .
~~
; COOKIE: 58b3a8a496c9219289dacbfb5da7d22ec3b32849673c32e3 (good)
~~
# Client Cookie を指定
# 1 回目
~~
; COOKIE: 00ffffffffff0000669ca7f85da7d2c2b518d911da3a5805 (good)
~~
# 2 回目
~~
; COOKIE: 00ffffffffff00007bea757e5da7d35e98e58f445ee737f0 (good)
~~

# Cookie なしの挙動
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +norec +nobadcookie @19
```

```
2.0.2.1 -t SOA .
~~
;; ->>HEADER<<- opcode: QUERY, status: BADCOOKIE, id: 49909
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 28e8aff1e70f256a4925af0b5da7d4d7d9bab26df2a61ef4 (good)
~~

# 有効な Cookie ありの挙動
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +norec +nobadcookie +coo
kie=28e8aff1e70f256a4925af0b5da7d4d7d9bab26df2a61ef4 @192.0.2.1 -t SO
A .
~~
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42590
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 28e8aff1e70f256a491a8a025da7d5eef26eb1e99dd351f8 (good)
~~

# 無効な Cookie の挙動: 別のクライアント
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +norec +nobadcookie +coo
kie=28e8aff1e70f256a491a8a025da7d5eef26eb1e99dd351f8 @2001:db8:53::1
-t SOA .
~~
;; ->>HEADER<<- opcode: QUERY, status: BADCOOKIE, id: 29041
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

# 無効な Cookie の挙動: 誤った値
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +norec +nobadcookie +coo
kie=38e8aff1e70f256a491a8a025da7d5eef26eb1e99dd351f8 @192.0.2.1 -t SO
A .
~~
;; ->>HEADER<<- opcode: QUERY, status: BADCOOKIE, id: 58942
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 38e8aff1e70f256a2fe290a85da7d716dbaa53d227501068 (good)
```

- こういう機能や設定があると良いと思った内容
  - 得られた Cookie の値をクライアントが意識することなく dig コマンドを実行できるような機能

## テーマ 2b

- BIND 9 fetches-per-server

```
# 3.1. fetches-per-server 無効(権威 DNS 応答有)
----------------------------------------
Statistics:

  Queries sent:         159970
  Queries completed:    157556 (98.49%)
  Queries lost:         2314 (1.45%)
  Queries interrupted:  100 (0.06%)

  Response codes:       NXDOMAIN 157556 (100.00%)
  Average packet size:  request 40, response 99
  Run time (s):         243.190920
  Queries per second:   647.869583

  Average Latency (s):  0.079368 (min 0.001602, max 4.997461)
  Latency StdDev (s):   0.290409
----------------------------------------


# 3.2. fetches-per-server 無効(権威 DNS 応答有->応答なし)
----------------------------------------
  Statistics:

  Queries sent:         2227
  Queries completed:    0 (0.00%)
  Queries lost:         2127 (95.51%)
  Queries interrupted:  100 (4.49%)

  Response codes:
  Average packet size:  request 38, response 0
  Run time (s):         111.021042
  Queries per second:   0.000000

  Average Latency (s):  0.000000 (min 0.000000, max 0.000000)
  ----------------------------------------


# 3.3. fetches-per-server 有効(権威 DNS 応答なし)
----------------------------------------
Statistics:

  Queries sent:         102274
  Queries completed:    98534 (96.34%)
```

```
  Queries lost:         3640 (3.56%)
  Queries interrupted:  100 (0.10%)

  Response codes:       SERVFAIL 98534 (100.00%)
  Average packet size:  request 39, response 39
  Run time (s):         250.136342
  Queries per second:   393.921168

  Average Latency (s):  0.065305 (min 0.000140, max 4.595752)
  Latency StdDev (s):   0.419372
-----------------------------------------
·     BIND 9 fetches-per-zone
# 3.1. fetches-per-zone 無効(権威 DNS 応答有)
-----------------------------------------
Statistics:

  Queries sent:         59606
  Queries completed:    58852 (98.74%)
  Queries lost:         654 (1.10%)
  Queries interrupted:  100 (0.17%)

  Response codes:       NXDOMAIN 58852 (100.00%)
  Average packet size:  request 39, response 98
  Run time (s):         77.358324
  Queries per second:   760.771394

  Average Latency (s):  0.070467 (min 0.001607, max 2.913739)
  Latency StdDev (s):   0.200640
-----------------------------------------

# 3.2. fetches-per-zone 無効(権威 DNS 応答有->応答なし)
-----------------------------------------
Statistics:

  Queries sent:         1000
  Queries completed:    0 (0.00%)
  Queries lost:         900 (90.00%)
  Queries interrupted:  100 (10.00%)

  Response codes:
  Average packet size:  request 37, response 0
  Run time (s):         48.084181
  Queries per second:   0.000000

  Average Latency (s):  0.000000 (min 0.000000, max 0.000000)
-----------------------------------------
```

```
# fetches-per-zone 有効(権威 DNS 応答有なし)
----------------------------------------
Statistics:

  Queries sent:         22663
  Queries completed:    21912 (96.69%)
  Queries lost:         651 (2.87%)
  Queries interrupted:  100 (0.44%)

  Response codes:       SERVFAIL 21912 (100.00%)
  Average packet size:  request 39, response 39
  Run time (s):         41.585535
  Queries per second:   526.913986

  Average Latency (s):  0.027965 (min 0.000738, max 4.049589)
  Latency StdDev (s):   0.243314
----------------------------------------
```

- こういう機能や設定があると良いと思った内容
  - Unbound ratelimit のようにドメイン名単位で応答を制限する機能
- BIND 9 serve-stale

```
---log---
18-Oct-2019 20:46:50.763 mail.example1.jprs resolver failure, stale a
nswer used
18-Oct-2019 20:46:51.780 connection refused resolving 'mail.example1.
jprs/A/IN': 192.0.2.2#53
18-Oct-2019 20:46:51.780 connection refused resolving 'mail.example1.
jprs/A/IN': 2001:db8:53::2#53

18-Oct-2019 20:47:36.550 mail.example1.jprs resolver failure, stale a
nswer unavailable
18-Oct-2019 20:47:37.567 connection refused resolving 'mail.example1.
jprs/A/IN': 192.0.2.2#53
18-Oct-2019 20:47:37.568 connection refused resolving 'mail.example1.
jprs/A/IN': 2001:db8:53::2#53
```

- こういう機能や設定があると良いと思った内容
  - stale-answer-ttl を 1 日など長い期間に設定した場合に、当該 TTL の持続中にリソースレコード内容が更新されて権威 DNS サーバーが復旧した段階で TTL 切れを待たずにキャッシュしている旧リソースレコードを更新する機能
- Unbound ratelimit

```
# 3.1. 全ドメイン名を制限
----------------------------------------------------------------------
Statistics:
```

```
  Queries sent:         300
  Queries completed:    300 (100.00%)
  Queries lost:         0 (0.00%)

  Response codes:       SERVFAIL 92 (30.67%), NXDOMAIN 208 (69.33%)
  Average packet size:  request 37, response 78
  Run time (s):         30.000342
  Queries per second:   9.999886

  Average Latency (s):  0.000333 (min 0.000103, max 0.001719)
  Latency StdDev (s):   0.000236

# 3.2. 特定ドメイン名を制限
----------------------------------------------------------------------
$ /tmp/attack_query1.txt
  Statistics:

  Queries sent:         100
  Queries completed:    100 (100.00%)
  Queries lost:         0 (0.00%)

  Response codes:       SERVFAIL 71 (71.00%), NXDOMAIN 29 (29.00%)
  Average packet size:  request 36, response 54
  Run time (s):         10.000339
  Queries per second:   9.999661

  Average Latency (s):  0.000314 (min 0.000115, max 0.002681)
  Latency StdDev (s):   0.000301


$ /tmp/attack_query2.txt
  Statistics:

  Queries sent:         100
  Queries completed:    100 (100.00%)
  Queries lost:         0 (0.00%)

  Response codes:       NXDOMAIN 100 (100.00%)
  Average packet size:  request 36, response 95
  Run time (s):         10.000353
  Queries per second:   9.999647

  Average Latency (s):  0.000627 (min 0.000380, max 0.002139)
  Latency StdDev (s):   0.000232
```

```
# 3.3. 特定サブドメインの制限
--------------------------------------------------------------------
動作確認できず
```

- 試せなかった範囲とその理由や状況
  - 3.3. 特定サブドメインの制限の動作を確認できず。**ratelimit-below-domain** を有効にしたが効果が出ていない状況であった。
- こういう機能や設定があると良いと思った内容
  - クエリのオプション毎に応答を制限できる機能（+any を指定したクエリのみ制限するなど）

## テーマ 2c

- NSEC3 agggressive use

```
・TLD console log
15-Oct-2019 10:53:53.756 client @0x7ffa680855a0 192.0.2.102#33324 (jp
a): query: jpa IN AAAA -E(0)D (192.0.2.1)
15-Oct-2019 10:53:53.756 client @0x7ffa68094000 192.0.2.102#39145
(.): query: . IN NS -E(0)D (192.0.2.1)
15-Oct-2019 10:53:53.757 client @0x7ffa680a2a60 192.0.2.102#58272 (jp
a): query: jpa IN AAAA -E(0)TD (192.0.2.1)
15-Oct-2019 10:53:53.757 client @0x7ffa68094000 192.0.2.102#52857
(.): query: . IN DNSKEY -E(0)D (192.0.2.1)
-----synth-from-dnssec yes;を no に-----
15-Oct-2019 13:25:57.166 client @0x7ffa680855a0 192.0.2.102#39898 (jp
a): query: jpa IN AAAA -E(0)D (192.0.2.1)
15-Oct-2019 13:25:57.166 client @0x7ffa680855a0 192.0.2.102#36611
(.): query: . IN NS -E(0)D (192.0.2.1)
15-Oct-2019 13:25:57.167 client @0x7ffa68094000 192.0.2.102#39479
(.): query: . IN DNSKEY -E(0)D (192.0.2.1)
15-Oct-2019 13:25:57.168 client @0x7ffa680a2a60 192.0.2.102#43982 (jp
a): query: jpa IN AAAA -E(0)TD (192.0.2.1)
15-Oct-2019 13:29:13.058 client @0x7ffa680855a0 192.0.2.102#55918 (jp
c): query: jpc IN AAAA -E(0)D (192.0.2.1)
15-Oct-2019 13:29:13.058 client @0x7ffa680855a0 192.0.2.102#41127
(.): query: . IN NS -E(0)D (192.0.2.1)
15-Oct-2019 13:29:56.595 client @0x7ffa68094000 192.0.2.102#54048
(.): query: . IN NS -E(0)D (192.0.2.1)
15-Oct-2019 13:29:56.595 client @0x7ffa68094000 192.0.2.102#43924 (_.
jpc): query: _.jpc IN A -E(0)D (192.0.2.1)
15-Oct-2019 13:29:56.597 client @0x7ffa68094000 192.0.2.102#53299 (a-
root-server): query: a-root-server IN AAAA -E(0)DC (192.0.2.1)
15-Oct-2019 13:29:56.597 client @0x7ffa680855a0 192.0.2.102#58236 (_.
nic.jpc): query: _.nic.jpc IN A -E(0)D (192.0.2.1)
15-Oct-2019 13:29:56.598 client @0x7ffa680c0b90 2001:db8:53::102#3765
```

```
6 (www.nic.jpc): query: www.nic.jpc IN AAAA -E(0)D (2001:db8:53::1)
15-Oct-2019 13:29:56.598 client @0x7ffa680cf750 2001:db8:53::102#4212
6 (www.nic.jpc): query: www.nic.jpc IN AAAA -E(0)TD (2001:db8:53::1)

・2LD console log
15-Oct-2019 13:31:49.570 network unreachable resolving './DNSKEY/IN':
 2001:503:c27::2:30#53
15-Oct-2019 13:31:49.570 network unreachable resolving './NS/IN': 200
1:503:c27::2:30#53
15-Oct-2019 13:31:49.570 network unreachable resolving './DNSKEY/IN':
 2001:503:ba3e::2:30#53
15-Oct-2019 13:31:49.570 network unreachable resolving './NS/IN': 200
1:503:ba3e::2:30#53
15-Oct-2019 13:31:49.570 network unreachable resolving './DNSKEY/IN':
 2001:500:a8::e#53
15-Oct-2019 13:31:49.570 network unreachable resolving './NS/IN': 200
1:500:a8::e#53
15-Oct-2019 13:31:49.570 network unreachable resolving './DNSKEY/IN':
 2001:500:2f::f#53
15-Oct-2019 13:31:49.570 socket.c:4858: unexpected error:
15-Oct-2019 13:31:49.570 connect(192.33.4.12#53) 22/Invalid argument
15-Oct-2019 13:31:49.570 network unreachable resolving './NS/IN': 200
1:500:2f::f#53
15-Oct-2019 13:31:49.570 managed-keys-zone: Unable to fetch DNSKEY se
t '.': unexpected error
15-Oct-2019 13:31:49.571 socket.c:4858: unexpected error:
15-Oct-2019 13:31:49.571 connect(192.33.4.12#53) 22/Invalid argument
15-Oct-2019 13:31:49.571 resolver priming query complete

・Full resolver console log
15-Oct-2019 10:53:53.755 client @0x7fd200094000 192.0.2.102#37480 (jp
a): query: jpa IN AAAA +E(0) (192.0.2.102)
15-Oct-2019 10:53:53.758 resolver priming query complete
15-Oct-2019 11:23:28.654 client @0x7fd2000855a0 192.0.2.102#41933 (jp
c): query: jpc IN AAAA +E(0) (192.0.2.102)
15-Oct-2019 13:15:42.080 client @0x7fd2000855a0 192.0.2.102#33288 (ww
w.nic.jpc): query: www.nic.jpc IN AAAA +E(0) (192.0.2.102)
-----synth-from-dnssec yes;を no に-----
15-Oct-2019 13:25:57.165 client @0x7f56140855a0 192.0.2.102#45059 (jp
a): query: jpa IN AAAA +E(0) (192.0.2.102)
15-Oct-2019 13:25:57.168 resolver priming query complete
15-Oct-2019 13:29:13.057 client @0x7f5614094000 192.0.2.102#44416 (jp
c): query: jpc IN AAAA +E(0) (192.0.2.102)
15-Oct-2019 13:29:13.058 resolver priming query complete
15-Oct-2019 13:29:56.595 client @0x7f5614044020 192.0.2.102#36927 (ww
w.nic.jpc): query: www.nic.jpc IN AAAA +E(0) (192.0.2.102)
```

```
15-Oct-2019 13:29:56.596 resolver priming query complete

・Stub resolver console log
~~
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +ad @192.0.2.102 -t AAAA
 jpa.
; <<>> DiG 9.14.5 <<>> +ad @192.0.2.102 -t AAAA jpa.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 565
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL:
 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;jpa.                           IN      AAAA

;; AUTHORITY SECTION:
.                       10800   IN      SOA     a-root-server. saba-s
ec-ml.tldlabs.jprs. 2019091000 1800 900 604800 86400

;; Query time: 4 msec
;; SERVER: 192.0.2.102#53(192.0.2.102)
;; WHEN: 火 10月 15 10:53:53 JST 2019
;; MSG SIZE  rcvd: 104
~~
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +ad @192.0.2.102 -t AAAA
 jpc.
; <<>> DiG 9.14.5 <<>> +ad @192.0.2.102 -t AAAA jpc.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 35487
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL:
 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;jpc.                           IN      AAAA

;; AUTHORITY SECTION:
.                       84625   IN      SOA     a-root-server. saba-s
ec-ml.tldlabs.jprs. 2019091000 1800 900 604800 86400
```

```
;; Query time: 0 msec
;; SERVER: 192.0.2.102#53(192.0.2.102)
;; WHEN: 火 10月 15 11:23:28 JST 2019
;; MSG SIZE  rcvd: 104
~~
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +ad @192.0.2.102 -t AAAA
 www.nic.jpc.

; <<>> DiG 9.14.5 <<>> +ad @192.0.2.102 -t AAAA www.nic.jpc.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 62557
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL:
 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.nic.jpc.                   IN      AAAA

;; AUTHORITY SECTION:
.                       77891   IN      SOA     a-root-server. saba-s
ec-ml.tldlabs.jprs. 2019091000 1800 900 604800 86400

;; Query time: 0 msec
;; SERVER: 192.0.2.102#53(192.0.2.102)
;; WHEN: 火 10月 15 13:15:42 JST 2019
;; MSG SIZE  rcvd: 112

-----synth-from-dnssec yes;を no に-----

~~
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +ad @192.0.2.102 -t AAAA
 jpa.

; <<>> DiG 9.14.5 <<>> +ad @192.0.2.102 -t AAAA jpa.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 59097
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL:
 1

;; OPT PSEUDOSECTION:
```

```
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;jpa.                           IN      AAAA

;; AUTHORITY SECTION:
.                       10800   IN      SOA     a-root-server. saba-s
ec-ml.tldlabs.jprs. 2019091000 1800 900 604800 86400

;; Query time: 3 msec
;; SERVER: 192.0.2.102#53(192.0.2.102)
;; WHEN: 火 10月 15 13:25:57 JST 2019
;; MSG SIZE  rcvd: 104
~~
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +ad @192.0.2.102 -t AAAA
 jpc.

; <<>> DiG 9.14.5 <<>> +ad @192.0.2.102 -t AAAA jpc.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 37101
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL:
 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;jpc.                           IN      AAAA

;; AUTHORITY SECTION:
.                       10800   IN      SOA     a-root-server. saba-s
ec-ml.tldlabs.jprs. 2019091000 1800 900 604800 86400

;; Query time: 1 msec
;; SERVER: 192.0.2.102#53(192.0.2.102)
;; WHEN: 火 10月 15 13:29:13 JST 2019
;; MSG SIZE  rcvd: 104
~~
$ /var/saba-theme2/local/bind-9.14.5/bin/dig +ad @192.0.2.102 -t AAAA
 www.nic.jpc.

; <<>> DiG 9.14.5 <<>> +ad @192.0.2.102 -t AAAA www.nic.jpc.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 20339
```

```
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL:
 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.nic.jpc.                   IN      AAAA

;; AUTHORITY SECTION:
.                       10800   IN      SOA     a-root-server. saba-s
ec-ml.tldlabs.jprs. 2019091000 1800 900 604800 86400

;; Query time: 4 msec
;; SERVER: 192.0.2.102#53(192.0.2.102)
;; WHEN: 火 10月 15 13:29:56 JST 2019
;; MSG SIZE  rcvd: 112
```

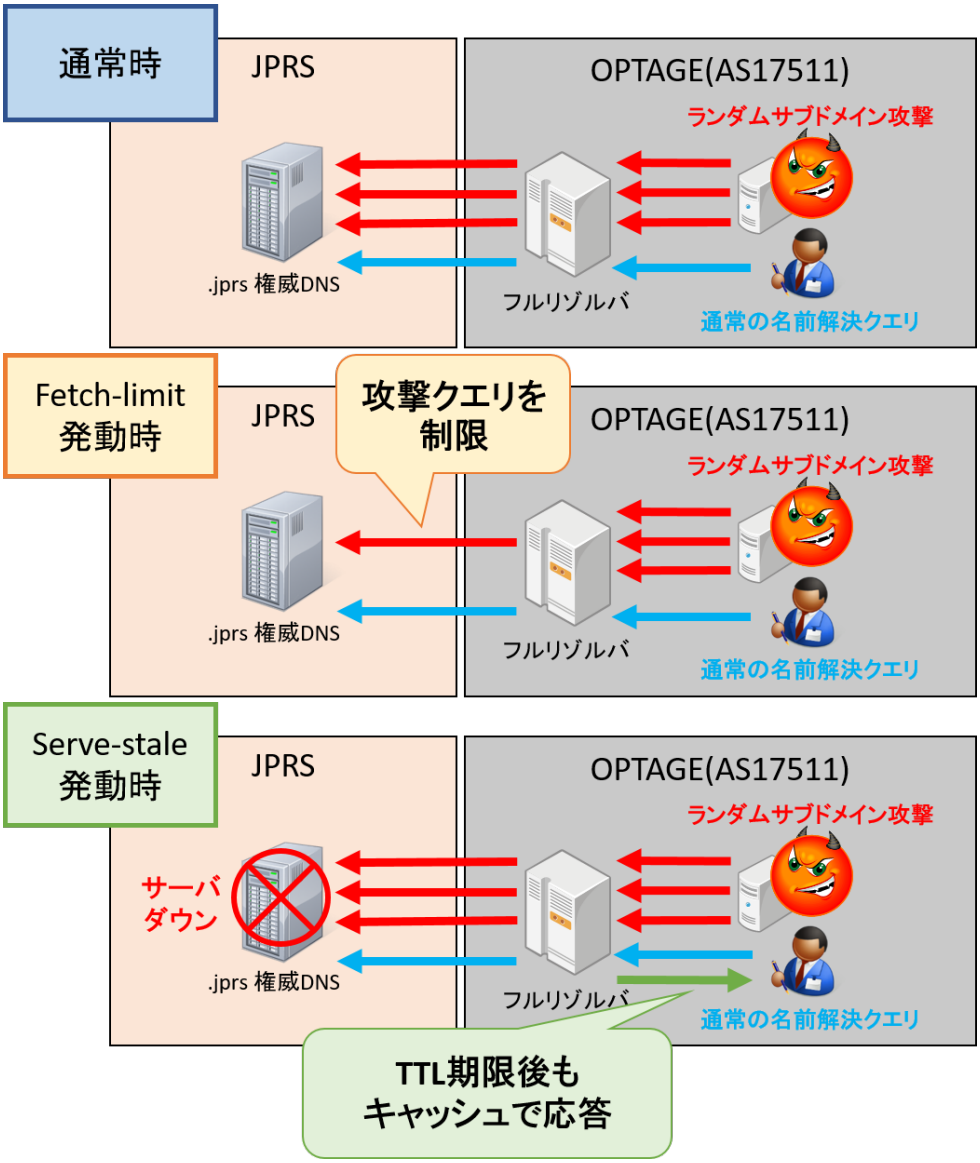
- こういう機能や設定があると良いと思った内容
  - zone enumeration 対策として、網羅的な問い合わせに対する応答を制限する機能 （NSEC3 ではゾーン名をハッシュ値とすることで対策しているが、応答自体は制限していないため）

## テーマ 2 実験

### 構成

- OS: CentOS7.5.1804
- DNS software: Unbound 1.9.3
- Tools
  - dig 9.11.4 P2
  - dnsperf 2.3.2
  - 攻撃模擬スクリプト: JPRS が作成したスクリプト

- テーマ 2 実証実験構成図




## 実験日時

- 2020 年 3 月 30 日 15:15～16:01

## 実験結果

- ランダムサブドメイン攻撃発生（権威 DNS サーバー無応答の状態）
  - dig コマンドを実行し、NO ERROR で ANSWER を返すことを確認した。
  - total.num.cachehits が増えていることを確認した。
  - SERVFAIL が増加し、NXDOMAIN は 0 のままであることを確認した。

- **notice : ratelimit exceeded** のメッセージが表示されることを確認した。

- キャッシュの TTL が満了し、serve-stale が発動
  - total.num.cachehits が増えていることを確認した。
  - SERVFAIL が増加し、NXDOMAIN は 0 のままであることを確認した。
- 権威 DNS の IP アドレス変更
  - 返す IP アドレスが変わっていないことを確認した。
- キャッシュクリア
  - 返す IP アドレスが変わったことを確認した。
  - total.num.cachehits が増えていることを確認した。
  - SERVFAIL は増加せず、NXDOMAIN が増加していることを確認した。

## 得られた知見

- 攻撃クエリ数が閾値を超えると制限する機能の動作を理解できた。
- 権威 DNS サーバーのキャッシュの TTL が満了した後も、フルリゾルバーが継続してキャッシュし応答を返す機能の動作を理解できた。
- IP アドレス変更後に権威 DNS サーバーが復旧した場合、フルリゾルバー側でキャッシュをクリアする操作が必要であることを理解できた。

## 課題

- serve-stale 機能については、権威 DNS サーバーの IP アドレスが変更されるかどうかに関わらず、復旧後にフルリゾルバー側でキャッシュをクリアする操作が必要であり、復旧からキャッシュクリアまでのタイムラグが発生すると考えられる。
- 上記の権威 DNS サーバー復旧の通知の連絡手段は、テーマ 1 の課題に記載した通り、メールあるいは電話になる。
- rate-limit 機能については、QNAME のみの判定に加えてリソースレコードタイプも判定の要素になっていれば、正規の名前解決クエリに影響しないよう、より柔軟な設定ができると考えられる。

# 実証実験報告（株式会社エネルギア・コミュニケーションズ）

## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### 目的

ISP のネットワークから外部への攻撃が発生した状況を想定した以下のシナリオによる実証実験を通じて、ISP 自社網から他社への攻撃や自社ドメイン名への攻撃の迅速な検知・的確な対応を可能にするための知見を取得し、それぞれの DNS 運用に役立てる。

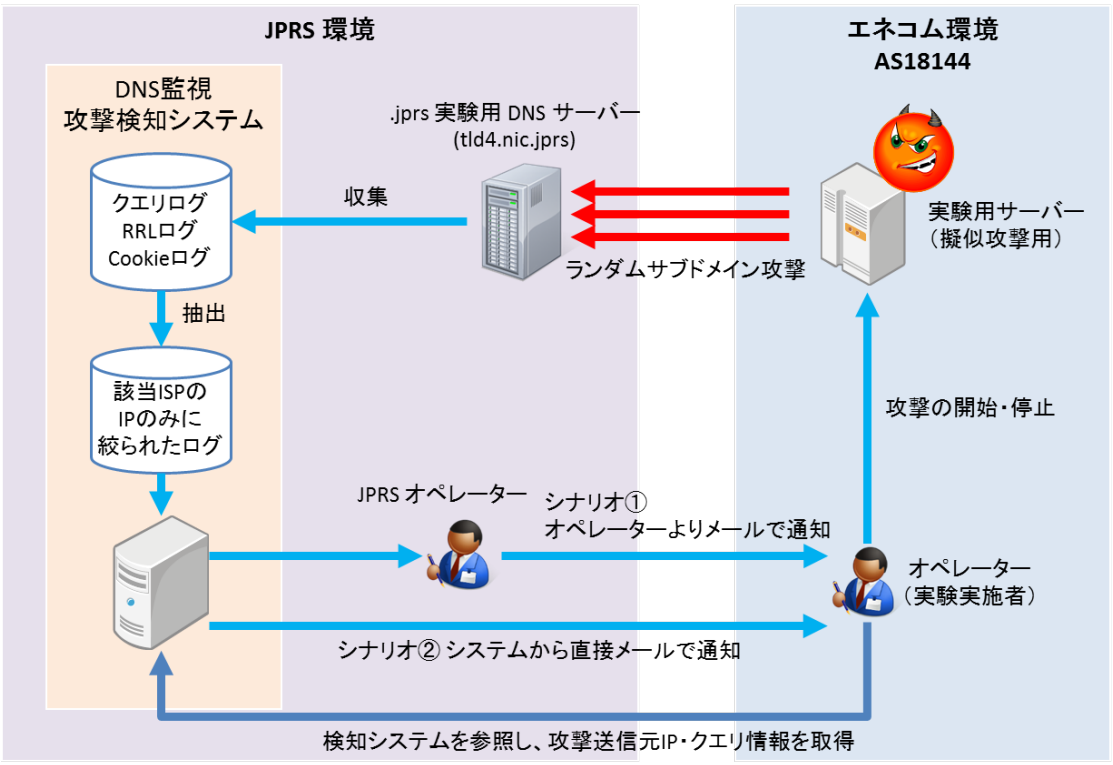
- TLD DNS サーバーを運用する JPRS にて各 ISP から流入するトラフィック量やクエリ情報をもとに攻撃発生を検知し、2 通り（TLD DNS サーバーオペレーターによるメール連絡、システムによるメール通知）の方法で各 ISP へ通知する。
- 各 ISP は、攻撃検知システムのダッシュボードにて攻撃元の情報（IP アドレス等）を確認し、攻撃を止める。

### 実験環境

- 攻撃検知システムは JPRS にて構築した。
- 当社は疑似攻撃発生環境を用意した。

### 疑似攻撃発生環境の構成

- サーバー
  - CentOS 7.6
  - 攻撃模擬スクリプト: JPRS から提供されたスクリプトを利用(dnsperf コマンドを使用)




## 連絡手段

- メールによる連絡

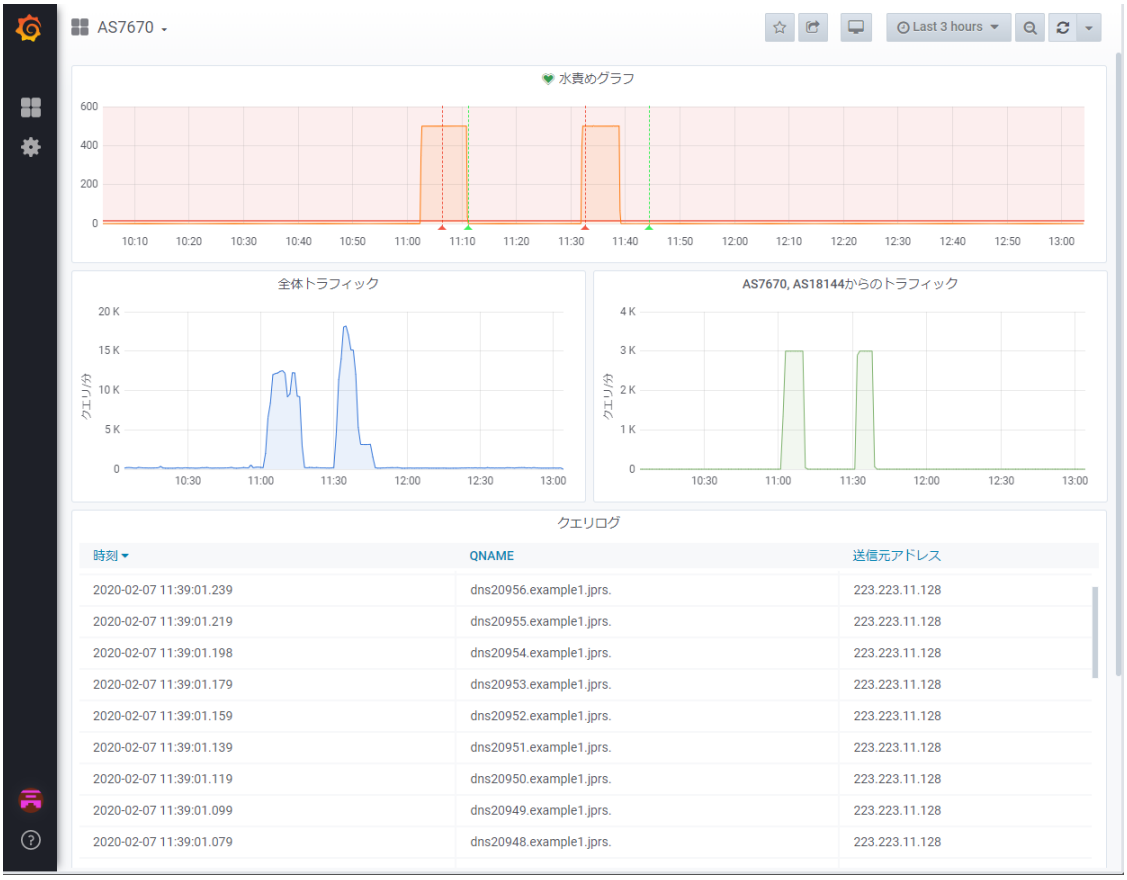
## 実験日時

- 実験日：2020/02/07
- 実験開始・終了時刻
  - シナリオ 1（TLD DNS サーバーのオペレーター介在通知）:発生時刻(11:02:35)～停止時刻(11:11:00)
  - シナリオ 2（攻撃検知システムの自動通知）:発生時刻(11:32:00)～停止時刻(11:39:00)

## 実験結果

- 実験の想定ステップ通りに攻撃検知の通知が行われた。
- 通知を受け、攻撃検知システムのダッシュボードにより自社攻撃元の IP アドレスを確認できた。

- 確認した IP アドレスをもとに攻撃元のサーバーからの攻撃（今回は実験のためスクリプト）を停止した。




## 得られた知見

- シナリオ 1 とシナリオ 2 の違い
  - 複数の宛先への送信が可能である場合、実効性の観点ではシナリオ 1 とシナリオ 2 には、運用に影響を及ぼす大きな差異はなかった。
  - 被害組織名や文面は見た目上、シナリオ 1 の方が行動の根拠としやすかった（シナリオ 2 では、システムによる通知文面のカスタマイズが望ましい）。
- 実際に攻撃を受けた際にできうる対応
  - 攻撃検知システムによりクエリ内容と送信元の確認を確認し、通信を遮断する。
  - ISP のフルリゾルバーの状態を確認する。

## 今後の課題

- 即座に当該送信元からの通信を止める判断は難しいと感じた。

- ISP の顧客通信の遮断にあたっては、約款上の問題がないか等の確認が必要そうである。
- 今回の検知・対応フローを踏まえた、監視・運用部隊との運用上のすり合わせが必要である。
- 検知時に通知する情報の内容が通信の秘密に該当しないか（違法性阻却事由に該当するか）について、より詳細な検討が必要と思われる。
- 自社以外への攻撃発生状況も二次情報として知ることができれば、より連携しやすくなるのではと感じた。
- ただし、その際に開示される情報の取り扱いについては注意が必要と思われる。

## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

ランダムサブドメイン攻撃への対処として、fetch-limit 機能、serve-stale 機能によるサーバー保護機能の動作、運用上の課題等を確認する。

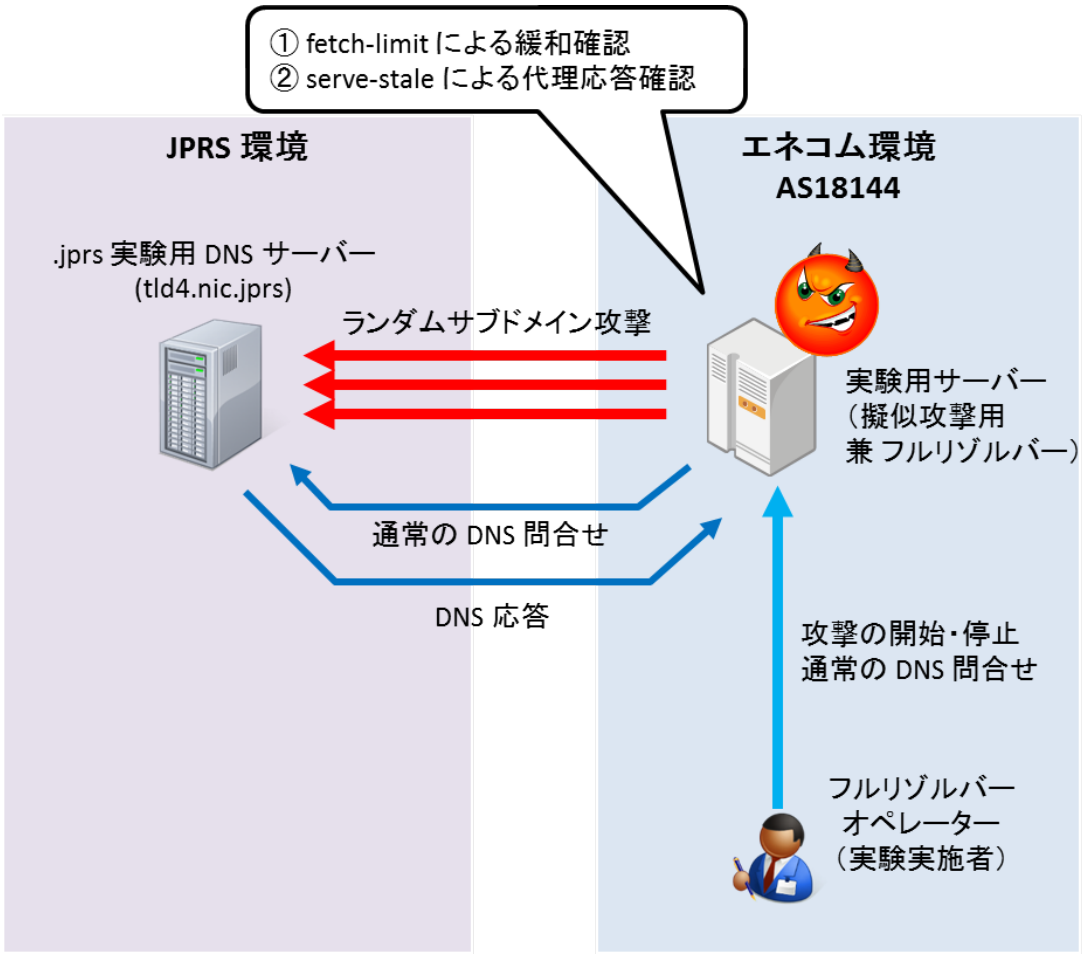
- シナリオとして、JPRS の権威 DNS サーバーがランダムサブドメイン攻撃を受けた場合に、ISP 各社のフルリゾルバーで本機能を用いてエンドユーザーへの影響を最小限に抑えることを想定した。

### 実験環境

- 権威 DNS サーバーは JPRS で構築した。
- 当社はフルリゾルバーを用意した。

### 構成

- サーバー
  - CentOS 7.6
  - DNS サーバーソフトウェア（BIND 9.14.5）
  - dig（9.14.5）
  - dnsperf（2.3.2）
  - 攻撃模擬スクリプト: JPRS から提供されたスクリプトを利用(dnsperf コマンドを使用)




## 実験日時

- 実験日：2020/03/17
- 実験開始・終了時刻(14:00:02～15:32:17)

## 実験結果

1. 疑似攻撃の送信
2. ランダムサブドメイン攻撃による権威 DNS サーバー無応答を確認
3. fetch-limit の確認
   - ログより、当該ドメイン名に対し fetch-limit が動作していることを確認できた。
   - 17-Mar-2020 14:31:49.487 spill: info: too many simultaneous fetches for ホスト名 (allowed 10 spilled 1)
4. serve-stale の確認

- SERVFAIL 応答となる場合と、想定通りに TTL が 1 となる応答が返るケースがあり、判然としなかった。
- SERVFAIL：17-Mar-2020 14:48:53.876 serve-stale: info: ホスト名 resolver failure, stale answer unavailable
- serve-stale：17-Mar-2020 14:48:54.668 serve-stale: info: ホスト名 resolver failure, stale answer used

5. キャッシュクリア
   - 想定通りに応答が返るようになったことを確認した。
6. 実験終了

## 得られた知見

- 機能への理解
  - fetch-limit, serve-stale 機能のそれぞれの挙動に関し、基礎的な理解が得られた。
- 想定通りに動いたこと
  - fetch-limit については、想定通りに機能した。
  - 実運用に近い環境での想定として、権威 DNS サーバーが攻撃への対処として IP アドレスを変更した場合、フルリゾルバー側でキャッシュをクリアする必要があることが分かった。
- 想定とは違う動きをしたこと
  - serve-stale に関する動作に疑問が残った。fetch-limit が有効になっており、かつその制限が発動する状況であった場合、serve-stale よりも fetch-limit による制限が優先され、SERVFAIL となっているように見受けられた。

## 今後の課題

- 実際の導入にあたっては、機能を有効活用するための想定ケースを踏まえた検証・検討、そのためのオプションの設定とそのチューニングが必要である。

# 実証実験報告（株式会社 QTnet）

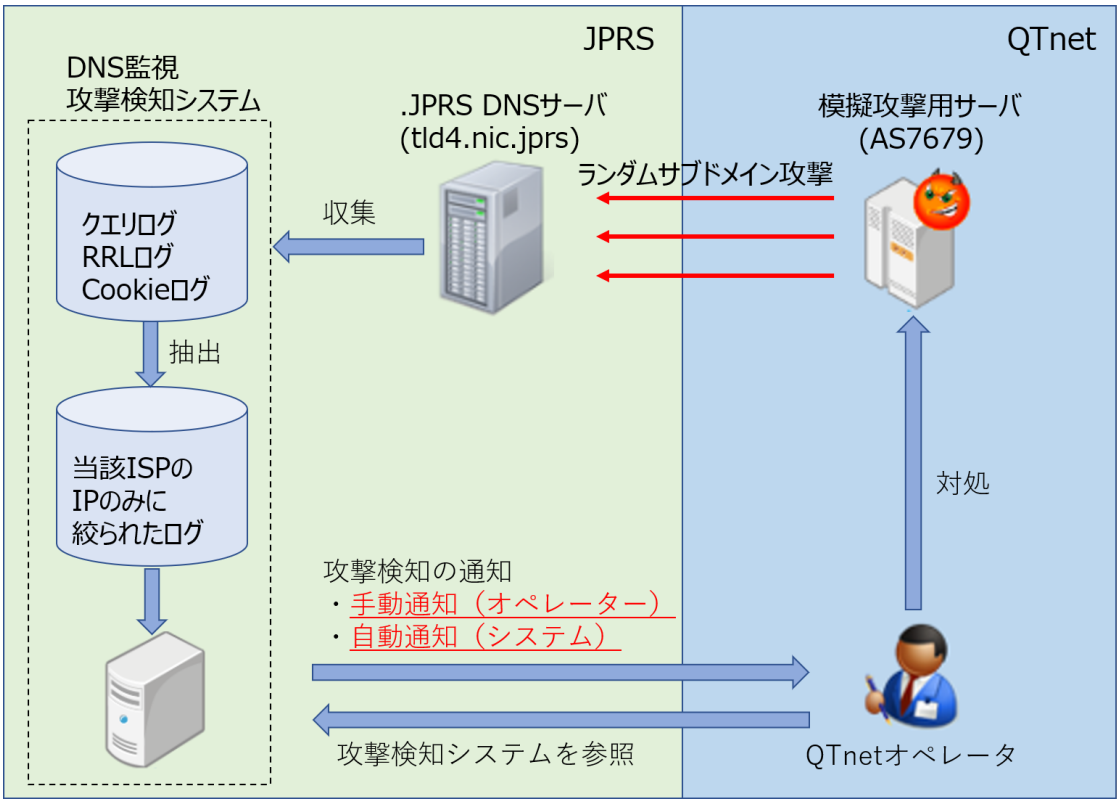
## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### 目的

DNS サーバーに対する DDoS 攻撃手法の一つにランダムサブドメイン攻撃（DNS 水責め攻撃）がある。本テーマでは、ISP 内からランダムサブドメイン攻撃を模擬した DNS トラフィックを発生させ、攻撃検知システムにて、攻撃の検知と攻撃発生を ISP に対し通知可能かの検証の実施を目的とする。

### システム構成

テーマ 1 実施時のシステム構成を記す。




### 実験内容

「ランダムサブドメイン攻撃の検知実験」のシナリオに基づいて実施した。

本実験では、ISP 内の模擬攻撃用サーバーからランダムサブドメイン攻撃を模擬した DNS トラフィックを一定の流量で発生させ、JPRS で開発された攻撃検知システ

ムにより、攻撃検知の可否を検証する。また、攻撃の発生を ISP へ通知する際に、手動通知と自動通知の２つの実験シナリオを実施し、通知までに要する時間を計測する。

## 実験シナリオ

・シナリオ 1(JPRS オペレーターからの手動の通知)

① ISP 内のクライアントから tld4.nic.jprs に対しての疑似攻撃開始

② 攻撃検知システムで攻撃を検知し、JPRS(TLD オペレーター)に通知

③ JPRS(TLD オペレーター)から攻撃の発生を Slack とメールへ 手動通知

④ 攻撃発信元の ISP が攻撃検知システムにて、攻撃の発生状況を確認

⑤ ISP 内のクライアントから tld4.nic.jprs に対しての疑似攻撃を停止

⑥ 攻撃発信元の ISP が攻撃検知システムにて、攻撃の停止を確認

・シナリオ 2(攻撃検知システムからの自動の通知)

① ISP 内のクライアントから tld4.nic.jprs に対しての疑似攻撃開始

② 攻撃検知システムで攻撃を検知し、攻撃の発生を Slack とメールで ISP へ 自動通知

③ 攻撃発信元の ISP が攻撃検知システムにて、攻撃の発生状況を確認

④ ISP 内のクライアントから tld4.nic.jprs に対しての疑似攻撃を停止

⑤ 攻撃発信元の ISP が攻撃検知システムにて、攻撃の停止を確認

## 実験結果

ランダムサブドメイン攻撃について、攻撃検知システムによる検知が可能であった。 攻撃検知システムにて検知した、ランダムサブドメイン攻撃の全参加組織分の攻撃トラフィックの推移を以下に示す。

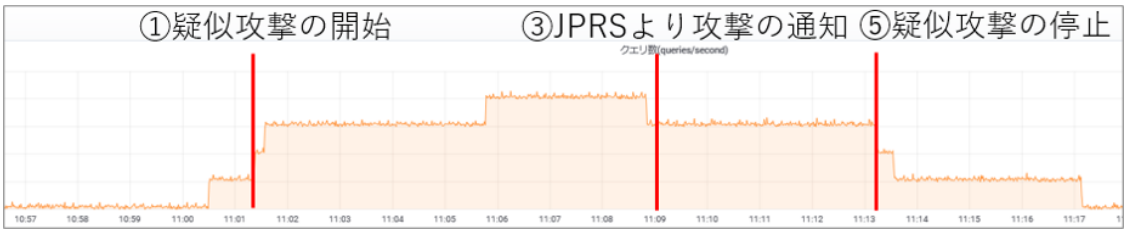



図 1. 手動通知(シナリオ１)攻撃トラフィックの推移


図 2. 自動通知(シナリオ２)攻撃トラフィックの推移

攻撃の発生を Slack で手動通知した際の様子を図 3 に示す。

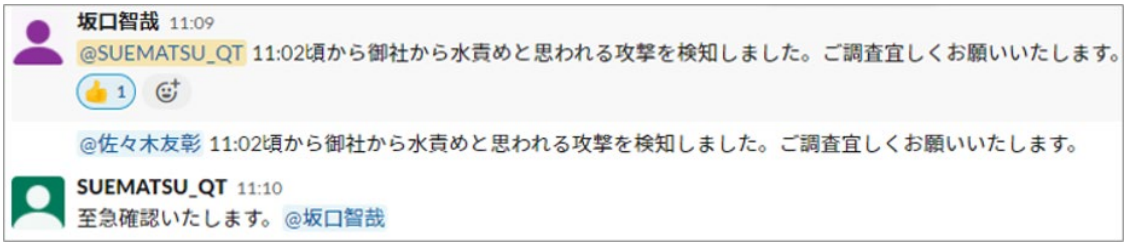



図 3. JPRS のオペレーターからの手動通知の様子

攻撃の発生を Slack で自動通知した際の様子を図 4 に示す。


図 4. 攻撃検知システムからの自動通知の様子

実験開始時間を起点に各 STEP での経過時間をシナリオごとにまとめたものを表 1 に示す。

表１.各シナリオにおける経過時

時間単位:分

| STEP | シナリオ１(手動通知) | シナリオ２(自動通知) |
|---|---|---|
| 攻撃発生 | 0 | 0 |
| 攻撃検知 | 0 | 0 |
| 攻撃通知 | 4 | 0 |
| 攻撃停止 | 11 | 4 |

## 結果の評価

本実験の目的である、TLD DNS 側でログを解析することによるランダムサブドメイン攻撃の検知については、ISP 内から模擬攻撃を発生させた直後に攻撃の検知ができた(図 2)。

攻撃発生から ISP への攻撃通知までに手動で通知(シナリオ１)した場合、通知に 4 分要したのに対し、自動で通知(シナリオ 2)した場合、攻撃発生直後に ISP へ通知を行うことができた(表 1)。

## 副次的な効果

今回用いた検知システムは、専用ツールは使用せず Web ブラウザから攻撃の発生状況を確認することができるため、対応するオペレーターのスキルレベルに依存せず、状況を容易に把握できるメリットがあることが確認できた。

## 課題

本実験での攻撃検知では任意の閾値を定め、それを超過した場合に攻撃として検知するように実装されており、攻撃が発生と収束を繰り返した際に過剰な検知が行われることが懸念される。そのため、実運用する際には過剰な通知を抑制する仕組みを検討する必要がある。。

## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

DNS サーバーに対する DDoS 攻撃手法の一つにランダムサブドメイン攻撃（DNS 水責め攻撃）がある。この攻撃には、正常な名前解決要求と攻撃の区別がつけにくいことから、攻撃の対策が難しいという特徴がある。

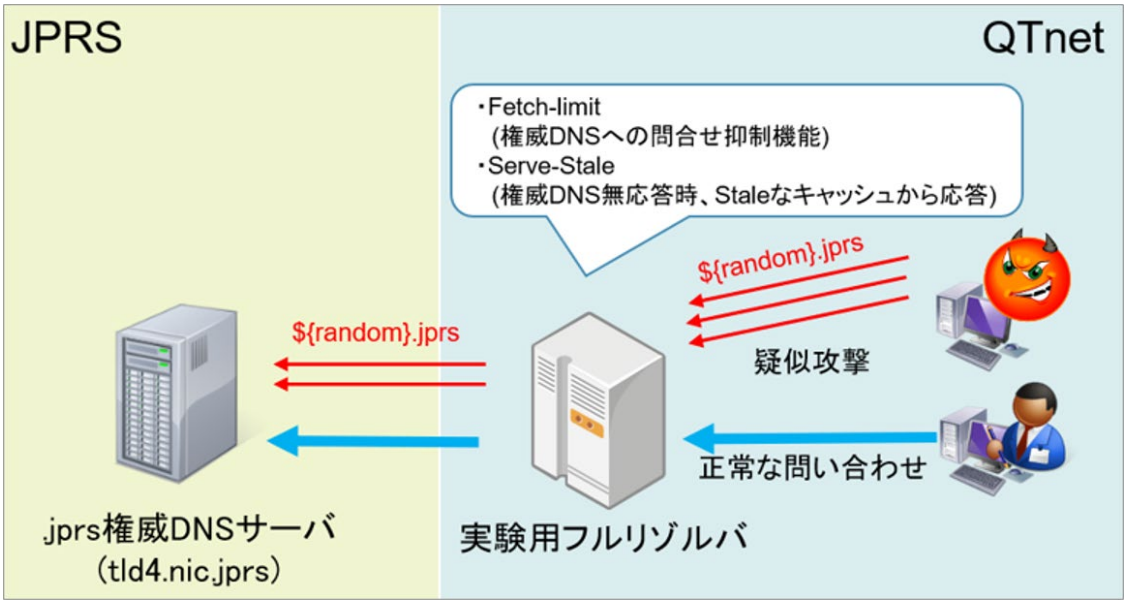
本テーマでは、下記に示す攻撃対策の有効性を検証する。

攻撃対策 I フルリゾルバーから権威 DNS サーバーへの反復問い合わせを抑制することによる攻撃対策(fetch-limit)

攻撃対策 II 権威 DNS サーバーが攻撃により応答を返せない状況下においても名前解決を可能とする攻撃対策(serve-stale)

### システム構成

テーマ 2 実施時のシステム構成を記す。




- フルリゾルバーの OS・ハードウェア構成
  - OS:CentOS 7.6
  - CPU:2Core(VM)
  - Memory:2GB
- フルリゾルバーのソフトウェア構成
  - BIND 9.14.5
  - dig 9.11.4-P2

- dnsperf 2.3.2

## 実験 1 の内容

「DDoS 攻撃の防御・緩和実験」のシナリオに基づいて実施した。

本実験では、フルリゾルバーに対して以下の機能を実装し、ISP 内の端末からランダムサブドメイン攻撃を模擬した DNS トラフィックを発生させ、名前解決の可否と攻撃対策の有効性を検証した。

### フルリゾルバーの設定内容

攻撃対策 I フルリゾルバーから権威 DNS サーバーへの反復問い合わせを抑制することによる攻撃対策（fetch-limit） 設定値は下記の通り

- 反復問い合わせする際の最大並列数：10 並列
- 最大並列数に到達時：SERVFAIL で応答

攻撃対策 II 権威 DNS サーバーが攻撃により応答を返せない状況下においても名前解決を可能とする攻撃対策（serve-stale） 設定値は下記の通り

- 有効期限(TTL)が過ぎているが応答を返す最大時間：1 日

### 実験シナリオ

① フルリゾルバー起動（攻撃対策 I:fetch-limit、攻撃対策 II:serve-stale）

② 名前解決の正常性を確認

③ ISP 内のクライアントからランダムサブドメイン攻撃開始

④ 権威 DNS サーバーの無応答を確認

⑤ フルリゾルバーにて、serve-stale が発動

⑥ 権威 DNS サーバーの IP アドレスを変更

⑦ フルリゾルバーにおいてキャッシュクリアし、名前解決の正常性を確認

## 実験結果

表１.各項番における名前解決の可否

| | ① | ② | ③ | ④ | ⑤ | ⑥ | ⑦ |
|---|---|---|---|---|---|---|---|
| 想定結果 | - | ○ | ○ | ○ | ○ | ○ | ○ |
| 実験結果 | - | ○ | ○ | ○ | × | × | ○ |

表２.各項番における攻撃対策の動作

| | ① | ② | ③ | ④ | ⑤ | ⑥ | ⑦ |
|---|---|---|---|---|---|---|---|
| 攻撃対策 I | - | ○ | ○ | ○ | ○ | ○ | ○ |
| 攻撃対策 II | - | - | - | - | × | × | ○ |

## 結果の評価

２つの攻撃対策を有効にした状態で名前解決可否を確認したが、正常に名前解決が行えない結果となった。また、下記の二つの攻撃対策については、_攻撃対策 I_ に関しては有効に機能したが、_攻撃対策 II_に関しては正常に機能しないことが判明した。

攻撃対策 I フルリゾルバーから権威 DNS サーバーへの反復問い合わせを抑制することによる攻撃対策

攻撃対策 II 権威 DNS サーバーが攻撃により応答を返せない状況下においても名前解決を可能とする攻撃対策

## 実験２の内容 (追試)

実験 1 の際に想定外に名前解決が継続できない事象が発生した。 そのため、攻撃対策 II の設定を変更し、名前解決が継続可能か再度確認を実施した。

## フルリゾルバーの設定内容

攻撃対策 I フルリゾルバーから権威 DNS サーバーへの反復問い合わせを抑制することによる攻撃対策（fetch-limit） 設定値は下記の通り

- 反復問い合わせする際の最大並列数：10 並列
- 最大並列数に到達時：無応答 #SERVFAIL から無応答へ見直し

攻撃対策 II 権威 DNS サーバーが攻撃により応答を返せない状況下においても名前解決を可能とする攻撃対策（serve-stale） 設定値は下記の通り

- 有効期限(TTL)が過ぎているが応答を返す最大時間：1 日

## 実験シナリオ(追試)

各社合同での実施時と同様のシナリオを項番 1 から項番 5 まで実施した。

### 実験結果(追試)

表 3. 各項番における名前解決の可否

| | ① | ② | ③ | ④ | ⑤ |
|---|---|---|---|---|---|
| 想定結果 | - | ○ | ○ | ○ | ○ |
| 実験結果 | - | ○ | ○ | ○ | ○ |

表 4. 各項番における攻撃対策の動作

| | ① | ② | ③ | ④ | ⑤ |
|---|---|---|---|---|---|
| 攻撃対策 I | - | ○ | ○ | ○ | ○ |
| 攻撃対策 II | - | - | - | - | ○ |

### 結果の評価

２つの攻撃対策を有効にした状態で、正常に名前解決が行えることが確認できた。また、下記の二つの攻撃対策については、どちらも有効に機能することを確認できた。

攻撃対策 I フルリゾルバーから権威 DNS サーバーへの反復問い合わせを抑制することによる攻撃対策

攻撃対策 II 権威 DNS サーバーが攻撃により応答を返せない状況下においても名前解決を可能とする攻撃対策

### 副次的な効果

２つの_攻撃対策_を有効にした状態で、特定の条件下では、期待(図 2)とは異なり正常に名前解決が行えない事象(図 3)が発生することが判明した。また、２つの攻撃対策を有効にした状態においても設定を変更することで、正常に名前解決が行えるようになることが判明した。

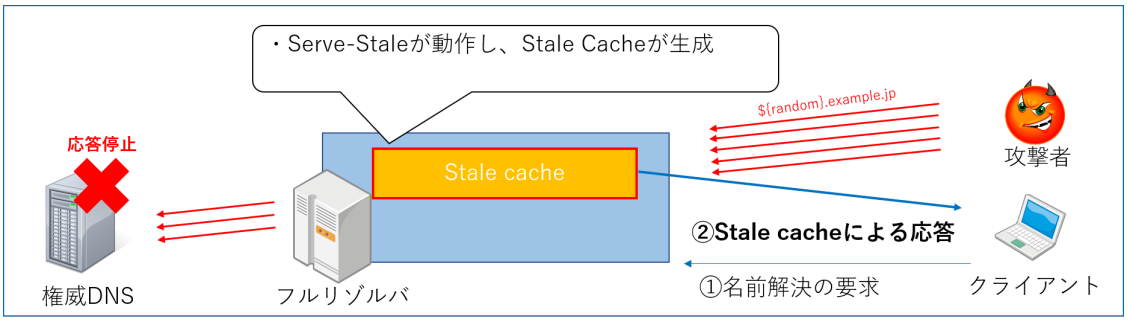



図 2. Stale キャッシュによる応答(期待)

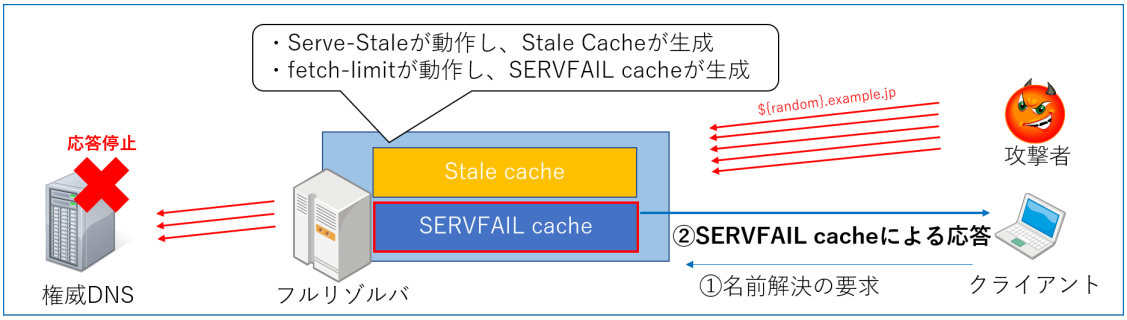



図 3. Servfail キャッシュによる応答(事象)

# 実証実験報告（沖縄通信ネットワーク株式会社）

## テーマ 1: DDoS 攻撃検知と DNS 運用組織間連携の実験

### 目的

自社ネットワーク配下から JPRS が運用する.jprs 権威 DNS サーバーに対して異常な負荷を発生させている端末が発生した場合、それを容易に特定できる仕組みを検証する。 具体的には次の通り。

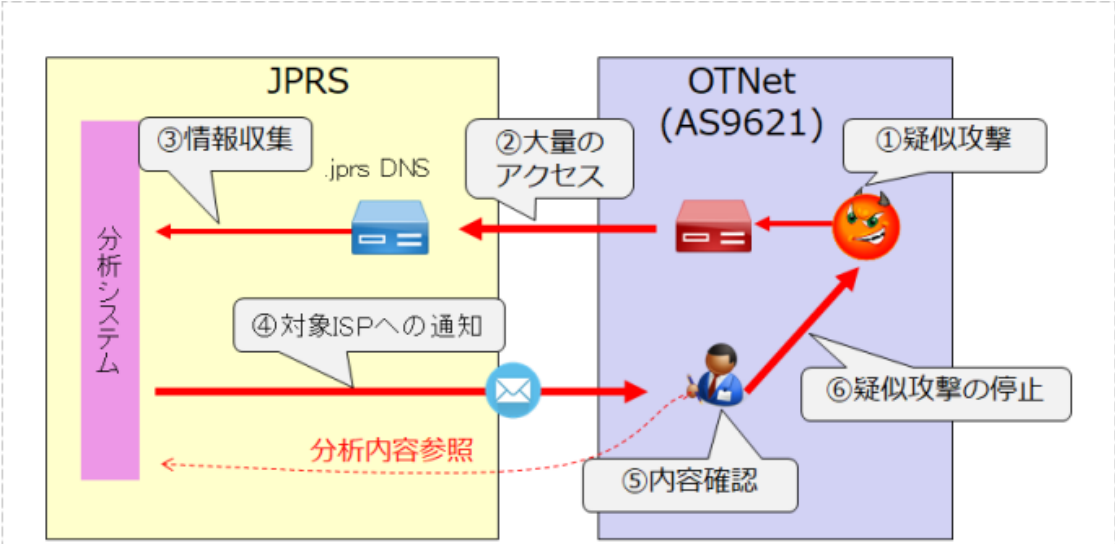
- 疑似端末で異常な問い合わせを発生させる
- JPRS 側で準備された攻撃検知システムで異常トラフィックを検知する
- JPRS より、異常発生のメール通知を行う

### 実験環境

#### 疑似端末構成概要

- OS CentOS 7.7.1908
- 攻撃模擬スクリプト: JPRS から提供されたスクリプトを利用

#### ネットワーク構成図と実験概要




インシデント発生時、上記のフローで十分な対応ができるかを検証した

## 連絡手段

メールでの連絡をおこなった(以下 2 パターンで確認を実施)

- シナリオ 1：JPRS 側オペレーターから送信されたメールの確認
- シナリオ 2：攻撃検知システムから自動送信されたメールの確認

## 実験日時

- シナリオ 1: 2019/02/05 11:00～11:30
  - 発生時刻(11:01:35)
  - 停止時刻(11:17:40)
- シナリオ 2: 2019/02/05 11:30～12:00
  - 発生時刻(11:35:00)
  - 停止時刻(11:42:58)

### 実験結果

- 計画された日程通りに実験が完了した
- 当社配下の IP アドレスを割り当てた疑似端末からランダムサブドメイン攻撃を実行したのち、計画通り、メールで通知が届いた
- 準備された攻撃検知システムにより、対象の IP アドレスと問い合わせ総数を確認できた

### 得られた知見

通知メールの様式について

## シナリオ 1 (人手でのメール連絡)

- JPRS のオペレーターと詳細内容についてのコミュニケーションが可能であるため、運用窓口における対応との親和性が高い

## シナリオ 2 (機械的なメール通知)

- 本実験による機械的なメールでは文面が英語であったことから、運用窓口に充分に周知していない場合、メールが無視され対応されない可能性が高い
  - 日本語、かつ内容についてある程度の説明が記載された文面であれば、対応は可能になる
- 詳細内容を確認するエスカレーション先が必須である
- 事象発生の頻度が少ない場合、対応方法が忘れられる可能性が高いので、継続的な教育が必要になる

## 障害検知システム上の情報について

- 送信元 IP アドレスが表示されているので、自社で運用している設備や BtoC 顧客であれば、何らかの対応が即時に実施可能であると考えられる
- BtoB 顧客の場合、一定規模以上の法人顧客はネットワーク管理や構築を外注しているケースが多く、どのように内容を伝えて対応を依頼するのかの検討が必要になる
- レアケースと考えられるが、同じタイミングで複数の IP アドレスから攻撃が発生した場合、対応の優先度の確認が困難になると考えられる。ランダムサブドメイン攻撃のグラフを TOP5 送信元 IP アドレスで区別・表示できるようにすることで、より状況を把握できると考えられる

## 今後の課題

- 機械的な通知で対応を行う場合、IP(インターネットプロトコル)の知識の少ない運用スタッフでも内容を想像できる文面の検討・作成が必要である
- BtoB 顧客配下ネットワークで問題を検知した場合、具体的にどんな内容を伝えるのかの検討が必要である
- 問題発生時のチケット管理方法の確認方法について、検討・決定が必要である(例えば、何をもって Close とするか・単に負荷が下さがれば Close でいいのか、など)

## テーマ 2: DDoS 攻撃の防御・緩和実験

### 目的

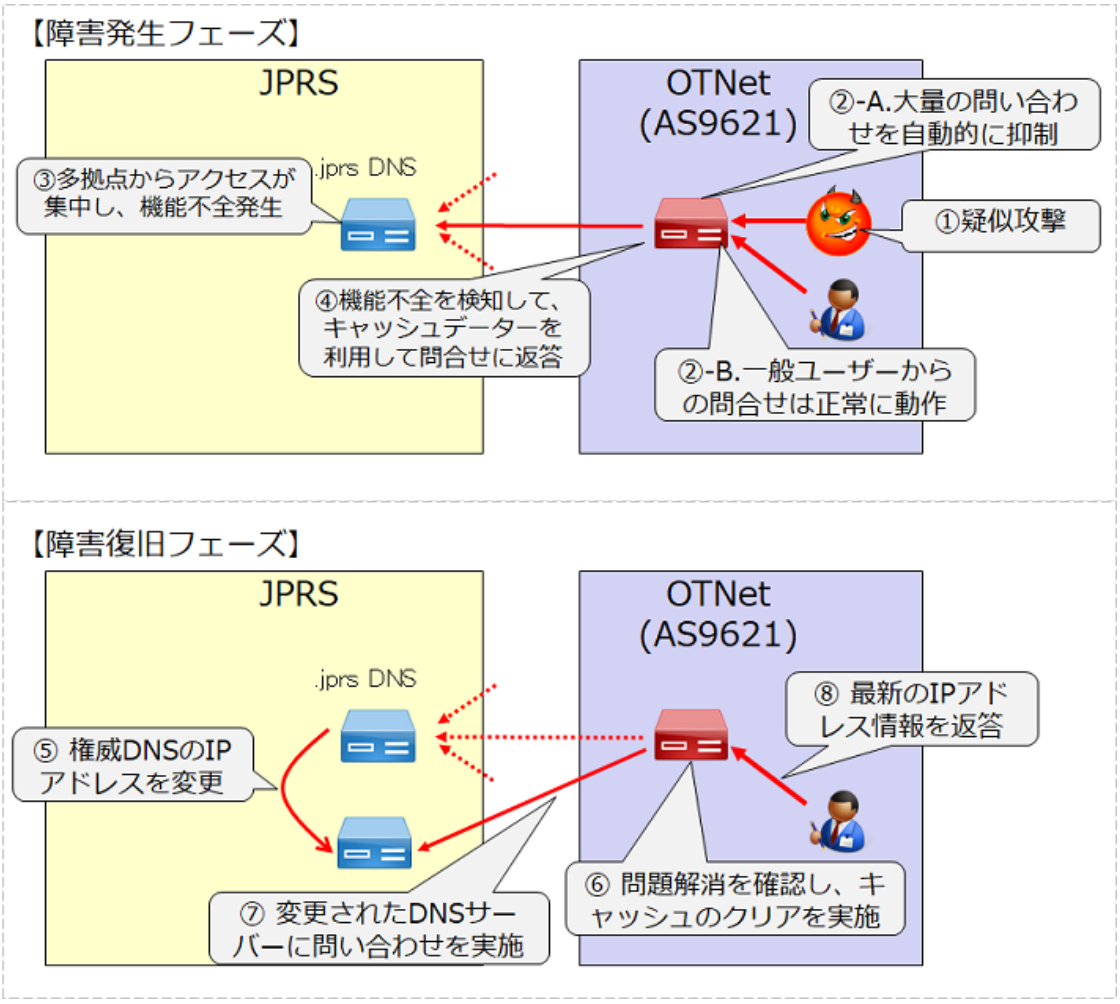
- 自社運用のフルリゾルバーに意図しない負荷がかかった場合、自動的に問題を検知してアクセス数の制限を行う仕組みの検証を行う
- 外部の権威 DNS サーバーへの通信が何らかの理由で不通となった場合、キャッシュの有効期限を超過しても、一定期間情報を保持してユーザーに返答する仕組みの検証を行う

### 実験環境

#### サーバー

- OS CentOS7.7.1908
- DNS サーバーソフトウェア（Unbound 1.9.3)
- dig（9.14.5）
- dnsperf（2.3.2）
- 攻撃模擬スクリプト: JPRS から提供されたスクリプトを利用

## ネットワーク構成図と実験概要




## 実験日時

- 2020 年 3 月 30 日 15:15～16:01

## 実験結果

- fetch-limit の効果(異常な問い合わせへのアクセス数制限)
  - attack を開始してすぐに rate-limit(アクセス数制限)が発動した事が Unbound の log と attacker のログで確認できた(抑制機能発動を確認) –
  - 正常端末(別 IP アドレス)からの問い合わせについては、問題なく返答できている事を確認できた(サーバーとしての正常動作を維持)
- serve-stale の効果
  - serve stale が発動したことを確認できた(意図した機能が動作した)

- 発動のタイミング(キャッシュの TTL 満了)については、想定よりも早かった(15:45～16:00 発動の予定が、15:34 で発動)

## 得られた知見

本実験により、以下の項目に関する知見が得られた。

- 権威 DNS サーバーから受け取った応答の TTL が満了してもフルリゾルバー側で情報をキャッシュする機能の理解
- 権威 DNS サーバーが IP アドレス変更で復活しても、迅速に復旧を検知するわけではないことの確認
- 権威 DNS サーバーの IP アドレスが変更になった場合、キャッシュのクリア操作が必要になることの確認
- 権威 DNS サーバーの IP アドレスを変更することなく状況が改善した場合も、仕組み的にはキャッシュのクリアが必要になることを想定した運用の必要性

また、実運用を前提とした場合の要検討事項として、以下が得られた。

- 権威 DNS サーバーが DNS 情報の有効期限(TTL)を無視する事になる。serve-stale を導入した組織ごとに挙動が変化することで、混乱が発生しないか？
- どの組織が serve-stale を運用しているか、確認が必要になるのではないか？
- 権威 DNS サーバーが攻撃された場合の問題回避策については、権威 DNS サーバー側で事前に検討を実施した上で対応することが望ましいのではないか？運用の敷居が高いと思うが、IP Anycast の導入など、もっと有効な技術があるのではないか？

## 今後の課題

serve-stale を実環境に適用するためには、今回確認できなかった以下の挙動についての確認が必要になる。

- 権威 DNS サーバーが別の IP アドレスで復活したことを定期的に確認する機能
- 機能の復活が一定時間(指定時間)継続した後、安定と判断できた際に、期限切れとなったキャッシュを自動的にクリアし、新しい情報を学習する機能

# 総括

## 本研究に関するまとめ

本研究では、DNS を標的とした DDoS 攻撃への対処と攻撃発生時の連携・情報共有の方法について、国内の ISP9 社と JPRS が研究用の TLD である「.jprs」を活用し、インターネット上に構築した実環境を用いた実証実験を実施した。

実証実験では DDoS 攻撃手法の一つであるランダムサブドメイン攻撃を対象とし、TLD の権威 DNS サーバーにおいて攻撃を検知するシステムを構築したうえで、攻撃の影響を受けている ISP と連携し、攻撃の情報を共有する実験を実施した。また、受け取った情報を活用しつつ、ランダムサブドメイン攻撃に対処するため、インターネット上に設置した実験用のフルリゾルバーに攻撃をシミュレーションするトラフィックを送信し、BIND 9 と Unbound に実装された攻撃対策機能を有効にして、その効果を確認・評価した。

本実験により、ランダムサブドメイン攻撃の発生時には攻撃トラフィックの一部が TLD の権威 DNS サーバーに到達すること、及びそのトラフィックを解析することで、攻撃の発生を検知可能であることが確認できた。本実験では、ランダムサブドメイン攻撃の特徴を利用する形で攻撃を検知するシステムを構築したが、同様の手法はランダムサブドメイン攻撃以外の攻撃手法にも活用可能であり、今後の応用が期待できる。

また、攻撃の発生を TLD の権威 DNS サーバーのオペレーターと ISP の DNS オペレーターの間で共有することで、各 ISP の設備において攻撃による問題が発生していないかの調査に活用可能であることが確認できた。運用体制や連絡窓口・共有された情報の社内展開の方法は ISP ごとに差異があることから、自動的な通知や効果的な情報共有のための共通のプラットフォームの決定と活用（本実験では slack を使用）、各社の運用体制に合わせた情報共有の方式の策定といった事前対策が必要である旨の知見が得られた。

BIND 9 と Unbound の攻撃対策機能については、本実験に参加したすべての ISP との共同実験の結果、事前に想定した挙動を示すものと、事前の想定とは異なる挙動を示すものがあることが判明した。前者については、攻撃トラフィックを検知して制限を発動させる機能(fetch-limit)が設定通りに動作することと、当該機能が動作した際に DNS サーバーソフトウェア側のログ・統計情報にその旨が出力されることを確認し、実環境で運用する際に必須となる確認点を把握できた。後者については、fetch-limit と TTL が満了した際にもキャッシュを破棄せず、名前解決を継続する機能(serve-stale)を併用した場合、そのキャッシュ情報を応答するのではなく SERVFAIL を応答することが判明した。本事象を当該 DNS ソフトウェアの

ベンダーにフィードバックした結果、ベンダー側が意図していなかった挙動であることが判明し、後日修正された。

更に、攻撃対策機能である DNS Cookies についても、実験環境を用いた動作確認・評価を実施した。その結果、前者の DNS Cookies については個々のサーバーにおいて、事前に想定した挙動を示すことが確認できた。しかし、本機能が有効に機能するためにはフルリゾルバーと権威 DNS サーバーの双方のシステムが DNS Cookies に対応し、それぞれのクッキーを使って相手を相互に確認できる必要があるが、現状はまだ対応していない DNS サーバーが多く、今後の課題となった。

後者の NSEC Aggressive Use についてはランダムサブドメイン攻撃が発生した際のフルリゾルバーから権威 DNS サーバーへの不要なトラフィックを抑制でき、効果が非常に大きいことが判明した。しかし、本機能はフルリゾルバーと権威 DNS サーバーの双方が DNSSEC に対応していることが前提となっていることから、DNSSEC が十分に普及していない現在の状況においては、その効果は限定的であることが判明した。また、.jp や.com/.net をはじめとするいくつかの TLD では署名コストの低減・ゾーンウォーキングの回避などの理由により、DNSSEC 署名方式として NSEC3 Opt-Out が採用されており、そうした TLD では本機能による恩恵を受けることができないという課題も顕在化した。

## 本研究の成果公開

本実証実験の成果公開の状況を以下に示す(()内は公開件数)。

| 種別 | 2020 年<br>4 月～6<br>月 | 2021 年<br>1 月～3 月 | 2021 年<br>4 月～10 月 |
|---|---|---|---|
| 特許出願 | (0) | (0) | (0) |
| 研究論文 | (0) | (0) | (0) |
| 外国発表予稿等 | (0) | (0) | (0) |
| 収録論文 | (0) | (0) | (0) |
| 学術解説等 | (0) | (0) | (0) |
| 外部機関誌論文 | (0) | (0) | (0) |
| 著書等 | (0) | (0) | (0) |
| 一般口頭発表 | (0) | (1) APRICOT (2021/02) | (2) JANOG (2021/07) |
| 報道発表 | (0) | (0) | (0) |
| その他資料 | (0) | (0) | (1)APNIC Blog(2021/10) |
| 展示会(社外開催) | (0) | (0) | (0) |
| 展示会(社内開催) | (0) | (0) | (0) |
| 標準化提案 | (0) | (0) | (0) |

## 注記事項

株式会社コミュニティネットワークセンター

株式会社オプテージ

株式会社エネルギア・コミュニケーションズ

株式会社 QTnet

沖縄通信ネットワーク株式会社