一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
IA2021-24(2021-09)

RPZ を用いた不審な DNS 外部クエリーの検知・遮断システムの一検討

一瀬 光† 金 勇‡ 飯田 勝吉††

†東京工業大学 オープンファシリティセンター
‡東京工業大学 学術国際情報センター
††北海道大学 情報基盤センター

E-mail: †hichise@nap.gsic.titech.ac.jp, ‡yongj@ gsic.titech.ac.jp, ††iida@iic.hokudai.ac.jp

あらまし ボットの中でも組織内 PC が DNS フルリゾルバを経由せずに直接外部に DNS クエリー(直接外部 DNS クエリー)を送信するボットが問題となっている。本研究ではその直接外部に DNS クエリーを送信するボットを検知・遮断することを目的とする。本稿では、この課題を解決するために DNSRPZ（Response Policy Zone）を活用したシステムを提案・実装し、初期的な評価結果を報告する。

キーワード ボットネット，不正通信，DNS，RPZ，直接外部クエリー，SDN.

A Detection and blocking system for suspicious DNS outbound query using RPZ

Hikaru ICHISE† Yong JIN‡ Katsuyoshi IIDA††

†Open Facility Center, Tokyo Institute of Technology
‡Global Scientific Information and Computing Center, Tokyo Institute of Technology
††Information Initiative Center, Hokkaido University

E-mail: †hichise@nap.gsic.titech.ac.jp, ‡yongj@ gsic.titech.ac.jp, ††iida@iic.hokudai.ac.jp

Abstract Bot programs that send DNS query without using the DNS full resolvers in an organization network (direct outbound DNS query) have become a critical problem. In this paper, we purpose to detect and block such malicious direct outbound DNS queries. In order to solve the problem, we proposed a detection and blocking system using DNS RPZ (Response Policy Zone) and implemented a prototype system. Based on the evaluation results we confirmed the features.

Keywords Botnet , Malicious traffic, DNS, RPZ, Direct outbound query and SDN.

1. はじめに

昨今でもボットネットは新たな技術を取り入れ、激化している。文献[1]によると、2021 年 6 月に DirtyMoe マルウェアと呼ばれる Internet Explorer の脆弱性を悪用したボットネットが用いられ、仮想通貨の不正なマイニングを目的として利用されているが、DDoS 攻撃の発生も報告されており、2021 年度だけでも 10 万の被害が報告されている。

このような状況の下、企業や大学などの組織のネットワーク管理者は、組織内部がボットネットによる DoS 攻撃、コインマイナー、ランサムウェア等のターゲットにならないようにすることが重要である。また、同時に、組織内部に踏み台、つまりボットに感染した PC 等が存在しないことの定期的確認が必要となる。このように、ネットワーク管理者によるボットネット通信の発見方法あるいは遮断方法の開発が急務と言える。

ボットネットのプロセスとして、図 1 に示すように、初めにメールや Web 閲覧により、PC はボットに感染する(以下、ボット感染 PC 群と呼ぶ)。

次にボット感染 PC 群は C＆C (Command and Control) サーバを検知し、当該サーバと通信をする。この通信をボットネット通信と呼ぶ。そして、ボットネット通信を行ったボット感染 PC は C&C サーバの命令に従って、感染していない PC やサーバに対し、自動的にスパムメールを送信、不正アクセス、第 3 者への攻撃パケット等を送信する。このようなボットネット通信を利用した多くの攻撃を防御するために本研究ではボットネット通信に着目する。

最近の研究ではボットネット通信が DNS プロトコルを利用していることが報告されている[2]。本来の DNS の使用方法はドメイン名から IP アドレスへの変換(＝名前解決）を行うことである。しかしながら、インターネットサービスの増加により DNS も進歩し、TXT レ

コード[3]、OPT レコード[4]など新たなリソースレコードの追加や利用方法がされてきている。

これまで我々の研究[5,7]では、組織内部の DNS を分析することによって、TXT レコードを用いているボットネット通信の検出を行った。次に、文献[6,7]では組織内の PC が DNS による名前解決を行う場合、通常は組織内の DNS リゾルバに問合せを行う。しかしながら、図 2 に示すように PC が組織内の DNS リゾルバを経由せずに、さらに DNS リゾルバを経由した NS レコード情報の入手を行わずに直接外部に通信を行う場合（直接外部クエリー）に着目し、DNS 通信を分析し、直接外部クエリーのボットネット通信を検出した。さらに、その直接外部クエリーを自動的に検知・遮断するようなシステムを検討した。さらに、文献[8]では検討したシステムについて設計、実装をした。その後、そのシステムについて不正な直接外部クエリーは遮断し、正当な直接外部クエリーは通過できているかの機能評価を行い、最終的に本システムと実際に運用されているネットワークとの比較する性能評価を行った。その結果、本システムが実際に運用できる有効性を示した。

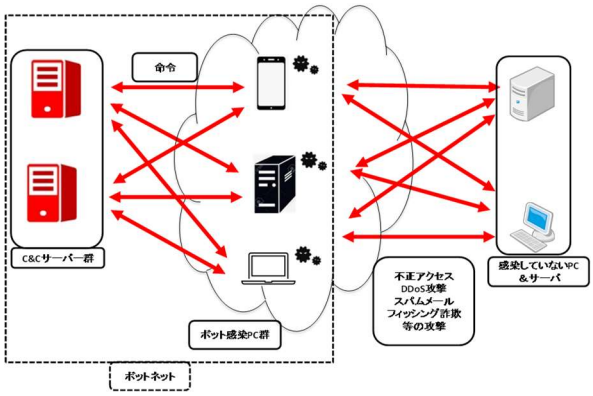

図 1:ボットネットの概要

しかしながら、これまでの研究課題は DNS 通信の性能評価においてネットワーク遅延が発生していた。そのため、本研究ではデータベースを利用するよりももっと効率的かつ、遅延を少なくするための方法を検討する。具体的には、ネットワーク管理者が自由に設定可能な機能として RPZ（Response Policy Zone）があり、この機能を利用することで従来方式(NS_DB)と同様の不正な直接外部クエリーを検知・遮断することができる。さらに性能評価により DNS 通信の遅延削減効果をあきらかにする。

本稿の構成は以下の通りである。2 節で関連研究について述べ、3 節で具体的な本システムの提案手法と実装を報告する。4 節ではシステム基づいた機能評価と性能評価を行う。5 節ではまとめと今後の課題を記載する。

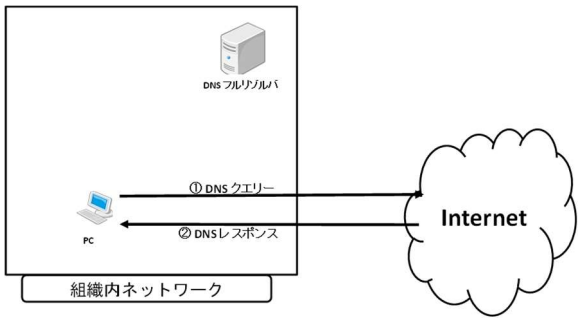

図 2：直接外部クエリー

2. 関連研究

これまで多くのボットネット通信に関する研究がなされている。本節では本研究と関連の深い研究を紹介する。

ボットネット通信の一つの技術として DNS トンネリングがある。DNS トンネリングでは DNS プロトコルとそのレコードの両方を悪用する。そのレコードの中でも TXT レコードを用いた方法が存在する。具体的には TXT レコードに BASE64 等でエンコードしたコンテンツを入れて返信することによってトンネリング通信を行う。その手法を使用しているものに Feederbot、Moto という不正プログラムが存在する[9]。また、Domain Flux と呼ばれる通信方法では、ボットプログラムが事前に定められたアルゴリズムに従って多数のドメイン名を作成し、そのドメイン名に対する DNS の名前解決を行うことで、C&C サーバを検知する。C&C サーバ検知した後は DNS リゾルバを経由せずに何らかの通信プロトコルを使って C&C サーバと通信すると考えられる。文献[10]ではボットプログラムは C&C サーバを探索するために多くの NXdomain のレスポンスを受信する。その挙動に着目して、複数の PC から同じドメイン名の名前解決を行った際に NXdomain のレスポンスを受信した場合、ボットに感染していると判断し、感染端末を検知し、遮断する研究もある。

以上のように、DNS を用いたボットネット通信については、これまで多数の分析がなされている。しかし、本研究で対象とする直接外部クエリーによる検知・遮断システムの研究についての研究は少ないことがあげられる。

3. RPZ を用いた検知・遮断システムの提案手法と実装

前節では DNS を用いたボットネット通信の検知・遮断システム関する関連研究を述べた。1.節で述べたとおり、直接外部クエリーのボットネット通信には、組織の DNS フルリゾルバを経由しないものが存在する。これはボットプログラムが直接外部クエリーを利用することにより、ネットワーク監視をすり抜けることを意図している。そのため、本研究では DNS フルリゾルバを経由しない不正な DNS クエリーに着目し、検知・遮断するシステムについて設計と実装する。

3.1. 概要

組織内の PC が DNS で名前解決を行う場合、通常フルリゾルバを経由する。正当な DNS 通信はこの時、フルリゾルバは名前解決に必要な NS (Name Server) レコードとそれに対応する glueA レコードを取得しながら、最終的に完全修飾ドメイン (FQDN) の IP アドレスを入手する。一方で、不審な直接外部クエリーを送信する PC についてはこのようなプロセスを取らず、事前に NS レコードを取得しない。PC に感染したボットプログラムに予め C&C サーバの IP アドレスがハードコーディングされており、感染した PC は DNS フルリゾルバを経由することなく直接その C&C サーバに DNS クエリーを送信し、通信をする。

そこで、以下の 2 つのステップを用いた、検知・遮断方式を提案する。

- NS レコードとそれに対応する glueA レコードを取得し、RPZ に登録
- リアルタイムに直接外部クエリーを監視し、その正当性を RPZ に確認と適切な通信制御

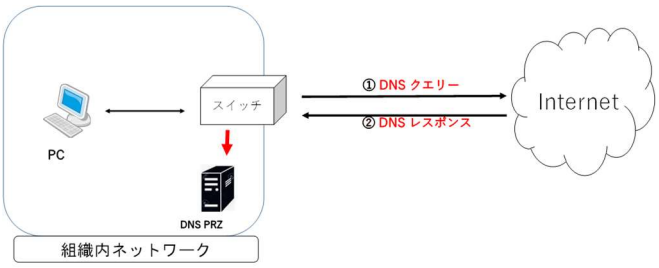

図 3:データ取得方法

まず初めに図 3 に示すように DNS クエリーとそれに対応する DNS レスポンスを取得する。次にその DNS レスポンスの中から NS レコードとそれに対応する NS レコードの IP アドレス (glueA レコード) を入手する。それを DNS フルリゾルバの機能である RPZ に保存する。

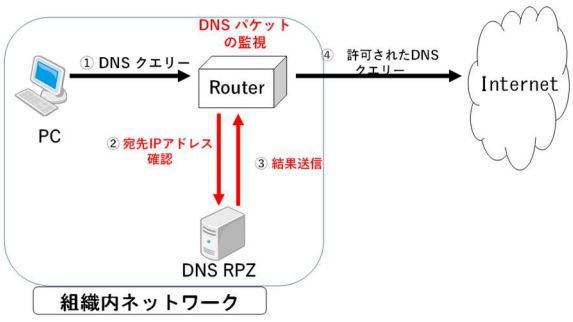

図 4: 提案システムの概要

次に、図 4 に示すように、PC が直接外部クエリーを送る際に宛先 IP アドレスを常に監視する。その宛先 IP アドレスが DNS フルリゾルバの RPZ に登録されている場合、DNS クエリーの通過を許可する。一方で、登録されていない場合、DNS クエリーをルータで遮断するようなシステムを構築することで直接外部クエリーを利用したボットネット通信を検知・遮断することが可能となる。しかしながら、直接外部クエリーにも正当な利用方法もある。その例として Google Public DNS がある[11]。この Google Public DNS の使用目的は直接外部クエリーを利用して、高速な名前解決をすることである。この他にも Cloudflare DNS[12]等があるため、このような正当な利用方法は DNS 通信を許可するために、IP アドレスを RPZ に登録しておく必要がある。

3.2. RPZ と設計

RPZ(Response Policy Zone)は DNS の機能の 1 つである特別なゾーンであり、特定のドメイン名に対して本来のドメイン名ではなく修正されたドメイン名の結果を DNS クライアントに返す機能である[13]。この機能は BIND9 から実装された。この機能の目的はクライアントから送信された潜在的に悪意のある DNS 名前解決に対して故意に失敗させたり、ローカルデータを返したりすることである。

この機能を利用することによって、DNS クエリーとそれに対応するレスポンスから RPZ 構築する。単純な方法の一つとして組織内ネットワークから全ての DNS トラフィックを取得する。さらに、その DNS トラフィックから正当な DNS の NS レコードとそれに対応する glueA レコードを分析し、RPZ に登録する。tcpdump コマンドを使って pcap フォーマットで全ての DNS トラフックを取得し、RPZ を構築するために分析した。具体的には、pcap ファイルにおいて分析プログラムは DNS クエリーなら、DNS クエリーの処理を行い。そうでないならレスポンスの処理を行う。DNS クエリーの処理についてはリストとして保存し、一方で DNS レスポンスでは DNS クエリーに対応するレスポンスを照合し、そのレスポンスから NS レコードとそれに対応する glueA レコードを取得し、RPZ に登録する。一方で、NS レコードのみの場合には一時的に NS レコードを RPZ に登録し、分析プログラムで継続的に DNS トラフィックを監視して、対応する glueA レコードがあった場合に RPZ に再度登録する。このようにして RPZ による不正 DNS 通信の遮断システムを実現する。

3.3. システムアーキテクチャ

本節では不正な直接外部クエリーを検知・遮断するためのシステムアーキテクチャを述べる。以下のような用語を定義する。

- クエリーの宛先 IP アドレス：PC が送信する DNS クエリーの宛先 IP アドレス
- RPZ：Public DNS と同様に正当な NS レコードとそれに対応する glueA レコードを登録した DNS フルリソルバの機能

本提案システムを構築するために SDN を利用する[14]。SDN はデータプレーンとコントロールプレーンからなる制御システムである。コントロールプレーンでデータプレーンであるスイッチやルータなどのネットワーク機器を制御することができる。これはネットワーク管理者が SDN を利用し、プログラムを開発することによって、ネットワーク制御、変更、管理を容易

にすることができる。この性質を利用し、図 5 に示すようなシステムアーキテクチャを提案する。具体的には SDN である OpenFlow スイッチと OpenFlow コントローラを利用して、クライアント PC から送信する直接外部 DNS クエリーを以下の手順に従って制御するシステムを提案する。

(1) PC はインターネットに対し直接 DNS クエリーを送る。その時、そのパケットは OpenFlow スイッチに転送する。
(2) OpenFlow スイッチは DNS クエリーを受信し、その DNS クエリーの宛先 IP アドレスのフローテーブルがない場合に OpenFlow コントローラに転送する。
(3) OpenFlow コントローラが DNS クエリーを受信した場合にそのコントローラは DNS フルリゾルバの RPZ に DNS クエリーの宛先 IP アドレスをチェックする。
(4) その宛先 IP アドレスがその RPZ に存在している場合、その結果を OpenFlow コントローラに返信する。
(5) その OpenFlow コントローラはその結果に基づいて DNS クエリーを通すかどうかを判断する。もしその DNS クエリーの宛先 IP アドレスが RPZ に存在する場合、OpenFlow コントローラはその DNS クエリーは正当な DNS クエリーを判定し、通信許可の制御を行う。そうでない場合、遮断する制御を行う。
(6) DNS クエリーの宛先 IP アドレスが RPZ に存在する場合にのみ、OpenFlow スイッチは外部のインターネットに通信を許可する。

上記の手順に従って、正当な DNS クエリーを検知し、正当な DNS クエリーのみを通信許可し、不審な DNS クエリーについては遮断するシステムを提案する。

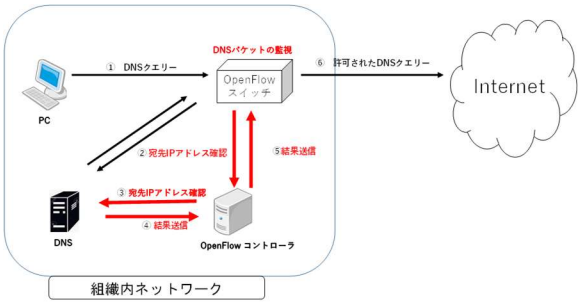

図 5: システムアーキテクチャ

3.4. 実装

3.1 節と 3.2 節で述べたような提案手法に基づいて、本節では 2 つのプログラムを作成し、実装をする。1 つは pcap ファイルから RPZ に登録するプログラム、もう一つは OpenFlow スイッチを制御するための OpenFlow コントローラのプログラムである。

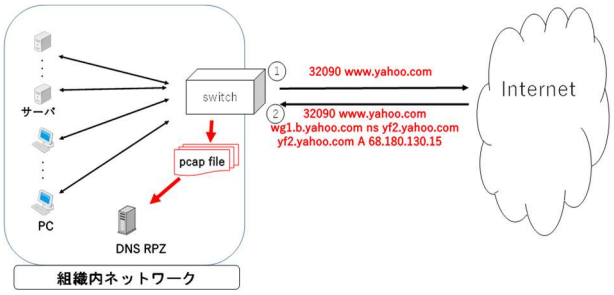

図 6: RPZ の登録

初めに、pcap ファイルから RPZ に登録するためのプログラムを説明する。実装プログラムでは python を利用し、pcap ファイルの処理では dpkt モジュールを使用する。さらにその処理した結果を perl のプログラムで RPZ に登録する処理を行っている。図 6 は RPZ にデータを登録するための実際のネットワーク構成を示す。pcap ファイルはスイッチから全てのクエリー（図 6 矢印①）とレスポンス（図 6 矢印②）を取得する。初め、dpkt を利用し、スイッチから取得した pcap ファイルを解読する。解読した pcap ファイルからクエリーの ID“32090”とクエリー名を一時的にリストに保存する。次に、そのクエリーの ID“32090”と照合したレスポンスを解読した pcap ファイルから取得し、RPZ に“wg1.b.yahoo.com NS yf2.yahoo.com”と“yf2.yahoo.com A 68.180.130.15”と“68.180.130.15 A 127.0.0.1”を perl の DNS dynamic update packet である Net::DNS モジュール[15]を利用して登録する。

次に SDN 技術を利用して、検知、遮断機能を実装した。OpenFlow スイッチは Open vSwitch[16]を利用し、OpenFlow コントローラでは Ryu プログラムを使用した。2 つのサーバ（ホストサーバ Server1 と Server2）と KVM（Kernel Virtual Machine）を使ってネットワーク環境を構築した。

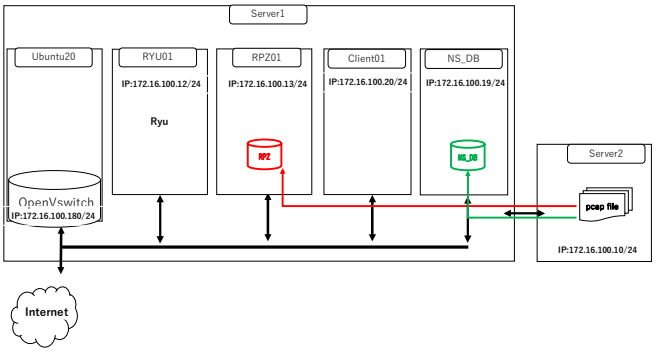

図 7:ネットワーク環境

4. 評価

提案システムの機能と遅延測定を確認するためにローカルのネットワーク環境を構築し、実装したシス

テムを評価する。3.4 の実装はプロトタイプであるため、ストレス評価等の性能評価は行っていない。主に本システムの機能評価として怪しい DNS トラフィックを遮断できるかの確認をする。次に DNS クエリーの名前解決をするためにこれまでのシステムとの差異を示す。

4.1. ネットワーク環境

本提案システムの評価をするために図 7 に示すようにクライアント、Open vSwitch、 Ryu コントローラ、NS 履歴データベースからなる実験ネットワーク環境を構築した。ホスト PC(Server1) 上に Open vSwitch、Ryu コントローラ、クライアント、NS 履歴データベースを KVM 上にインストールした。Ryu コントローラ、クライアントは Open vSwitch に繋がれ、Open vSwitch を経由してインターネットと通信できる。NS 履歴データベースと RPZ を Ryu と通信できるように KVM 上に設定した。つまり、全てのトラフィックは Open vSwitch を通過する。それゆえ、Ryu コントローラは全てのトラフィックの宛先 IP アドレスをチェックすることができ、さらにそのトラフィックを通過させるか、遮断させるかを決定することが可能となる。また、別のホスト PC (Server2) には pcap ファイル (200MB) と実装した RPZ の処理プログラムと既存の NS 履歴データベースの処理プログラムを設定した。以前の我々の研究[6]において、東工大のボーダルータから多くの DNS トラフィックの pcap ファイルを入手した。約 2 時間の DNS トラフィックの 1 塊の pcap ファイルを利用し、本システムを評価した。1 つの pcap ファイルを処理し、RPZ に登録する時間は 43 分かかった。これは 1 日の東工大の DNS トラフィックは 5GB であるため、処理するための時間としては充分と言える。

4.2. 機能評価

実験用のネットワーク環境において RPZ を構築したあと、提案システムにおいて怪しい DNS トラフィックの検知と遮断機能を検証した。具体的には RPZ に登録されている宛先 IP アドレスをもつ DNS クエリーをクライアントから送信し、Open vSwitch で Ryu コントローラによって適切に通過できるか確認する。さらに RPZ に登録されていない場合、遮断できるかを確認した。

Ryu プログラムを検証するために、事前に RPZ に Google Public DNS の宛先 IP アドレス「8.8.4.4」のみを登録した。表 1 に示すようにクライアントからインターネットに対し DNS クエリーを送信した。Open vSwitch は初め DNS クエリーを受信し、フローテーブルがないために、パケットインを経て、Ryu コントローラに転送される。この時に Ryu プログラムは DNS クエリーをチェックして、RPZ の登録された宛先 IP アドレスかをチェックする。もし宛先 IP アドレスが RPZ に登録されている場合はその DNS クエリーは Open vSwitch を通過され、Ryu プログラムで「<宛先 IP アドレス> is registered and pass」と表示される。一方で、RPZ に登録されていない場合は「<宛先 IP アドレス> block」と表示される。その結果は表 2 になり、想定通りの結果であった。

表 1: 機能評価結果

クライアントからのコマンド	OpenVswitch の挙動
dig @8.8.4.4 www.google.com	通過
dig @8.8.8.8 www.google.com	遮断
dig @1.1.1.1 www.google.com	遮断

4.3. 遅延評価

本節では以前の我々の研究[8]で行った NS 履歴データベースで発生していた DNS 通信のネットワーク遅延と今回の RPZ のネットワーク遅延について比較した。そのために、事前に gooogle.com の NS レコードと glueA レコードを取得し、PRZ と NS 履歴データベースにその glueA レコードを 50 個及び機能評価で遮断した IP アドレスを通信許可するために「8.8.8.8」と「1.1.1.1」を登録した。

表 2 はその結果である。フィルタリング制御なしとはスイッチとして全てのトラフィックを通過させる手法の設定である。一方で、前回の研究の NS 履歴データベースによるフィルタリング手法の結果はフィルタリング制御なしの設定よりも約 2 倍遅延が発生している。今回、提案した手法、つまり RPZ によるフィルタリング手法では NS 履歴データベースによるフィルタリング手法に比べて約 0.7 倍遅延が抑えられている。これは前回の NS 履歴データベースを利用した手法よりも本提案手法が効果的、かつ迅速に DNS クエリーを遮断及び通過できるシステムであることを示している。

5. おわりに

直接外部クエリーを利用したボットネット通信が報告されている。これまでに我々はそのような直接外部クエリーを利用したボットネット通信の検知・遮断方法について研究してきた。既存の研究の課題としてネットワーク遅延が大きかったため、それを改善させる方法を検討してきた。本研究ではその課題解決するために DNS の一部の機能である RPZ を利用することを提案し、実装、機能評価、及び基本的な遅延測定を行った。その結果、既存の NS 履歴データベース方式より

表 2: digによるクエリー遅延の評価結果

コマンド	DNSリゾルバの種別	フィルタリング制御なし		NS_DBによるフィルタリング手法		RPZによるフィルタリング手法（提案手法）	
		平均値［ms］	標準偏差［ms］	平均値［ms］	標準偏差［ms］	平均値［ms］	標準偏差［ms］
dig @172.16.100.13 www.google.com （キャッシュ無し）	ローカル設置	1,626.9	121.6	3,409.6	381.6	2,406.4	419.2
dig @172.16.100.13 www.google.com （キャッシュあり）	ローカル設置	0.0	0.0	0.0	0.0	0.0	0.0
dig @8.8.8.8 www.google.com	Public DNS (Google)	6.2	0.4	8.5	2.4	8.2	2.6
dig @1.1.1.1 www.google.com	Public DNS (Cloudflare)	6.5	0.5	8.1	2.5	7.3	1.5

もクエリー遅延を約 70%に低減できることを明らかにした。

今後は、詳細な性能評価を行うことで、より現実的なネットワークにおける本システムの有効性を明らかにする必要がある。

謝辞

本研究の一部は、日本学術振興会・科学研究費補助金・基盤研究 B (19H04103) の支援を受けている。ここに記し謝意を表す。

文　献

[1] AVAST 社 (online), avail from https://decoded.avast.io/martinchlumecky/dirtymoe-1/ (accessed 2021-07-22).

[2] N.M. Hands, B. Yang, and R.A. Hansen, “A Study on Botnets Utilizing DNS,” *Proc. ACM Conference on Research in Information Technology* (RIIT'15), pp.~23--28, ACM (2015).

[3] P.Mockapetris, “Domain names: Concepts and facilities,” *IETF RFC1034*, Nov. 1987.

[4] P. Vixie, “Extension Mechanisms for DNS (EDNS(0)),” *IETF RFC6891*, April 2013.

[5] H. Ichise, Y. Jin, and K. Iida, “Analysis of via-resolver DNS TXT queries and detection possibility of botnet communications,” *Proc. IEEE Pacific Rim Conference on Communications, Computersand Signal Processing (PACRIM2015)*, pp. 216–221, Aug. 2015. DOI:10.1109/PACRIM.2015.7334837

[6] Y. Jin, H. Ichise, and K. Iida, “Design of detecting botnet communication by monitoring direct outbound DNS queries,” *Proc. IEEE Int'l Confrerence on Cyber Security and Cloud Computing (CSCloud2015)*, New York, NY, pp.37-41, Nov.2015.

[7] H. Ichise, Y. Jin, and K. Iida, “Analysis of DNS TXT record usage and consideration of botnet communication detection,” *IEICE Trans. Commun.*, vol. E101-B, no. 1, pp. 70–79, Jan. 2018. DOI: 10.1587/transcom.2017ITP0009

[8] H. Ichise, Y. Jin, K. Iida, and Y. Takai, “NS record history based abnormal DNS traffic detection considering adaptive botnet communication blocking,” *IPSJ J. Information Processing*, vol. 28, pp. 112-122, Feb. 2020. DOI: 10.2197/ipsjjip.28.112

[9] C.J. Dietrich, C. Rossow, F.C. Freiling, H. Bos, M. Steen, and N. Pohlmann, “On botnets that use DNS for command and control,” in *Proc. IEEE European Conference on Computer Network Defence (EC2ND'11),* Gothenburg, Sweden, pp.9-16, Sept. 2011.

[10] Y. Iuchi, Y. Jin, H. Ichise, K. Iida, and Y. Takai, “Detection and Blocking of DGA-based Bot Infected Computers by Monitoring NXDOMAIN Responses,” *Proc. IEEE Int'l Confrerence on Cyber Security and Cloud Computing (CSCloud2020)*, New York, NY, Aug.2020.

[11] Google Public DNS, https://developers.google.com/speed/public-dns/ (accessed 2021-08-12).

[12] Cloudflare, https://1.1.1.1/ (accessed 2021-08-12).

[13] Internet Systems Consortium, https://www.isc.org/rpz/ (accessed 2021-08-12).

[14] Open Networking Foundation: SDN Architecture (online), https://www.opennetworking.org/ (accessed 2021-08-12).

[15] Net::DNS, https://www.net-dns.org/docs/README.html (Accessed 2021-08-11)

[16] Open vSwitch: Open vSwitch, https://www.openvswitch.org/ (Accessed 2021-08-11)