DNS キャッシュサーバ上のキャッシュ特性に注目した悪性トラヒック解析

代表研究者	石倉　直武	大阪公立大学　情報学研究科　博士後期課程
共同研究者	近藤　大嗣	東京大学　情報基盤センター　准教授
共同研究者	戸出　英樹	大阪公立大学　情報学研究科　教授

1 研究背景

Domain Name System (DNS) [1, 2]は，今日のネットワークにおいて重要なインフラストラクチャとしての役割を担っているが，それゆえに DNS の脆弱性を狙った攻撃は重大な影響を及ぼす．DNS プロトコルは 1980 年代に設計されたプロトコルでセキュリティ面の考慮が十分にされておらず，現在においても様々なセキュリティ脅威にさらされている [3]．企業ネットワークにおいても DNS に関する脅威は看過できないもので，マルウェアの侵入を許してしまった場合，DNS クエリの Fully Qualified Domain Name (FQDN)や DNS レスポンスに含まれるリソースレコードに本来の用途と異なる情報を埋め込むことで双方向通信を確立する DNS トンネリングを使用して対象ネットワークのファイアウォールやセキュリティを回避し，Command & Control (C&C)通信の確立や情報漏洩を達成することができる.企業ネットワークでは不要な通信ポートやプロトコルを制限することが一般的だが，DNS で利用されるポートやプロトコルを制限することは難しく，DNS トンネリングによる攻撃が検知されることなく長期間にわたって継続する可能性がある．

DNS トンネリングは,DNS プロトコルを悪用して通常の DNS の用途とは異なるデータ通信を行う手法である．通常のネットワークフィルタリングやファイアウォールでは DNS で用いられるポートやプロトコルに対して制御されておらず，DNS トンネリングは制約の厳しいネットワークと外部ネットワーク間でのデータ通信を実現できる．DNS トンネリングの秘匿性の高さとネットワーク制限の回避は攻撃者にとって強力な手段である．例えネットワーク制限やセキュリティの厳しい企業ネットワークであっても，マルウェアの侵入が成功すれば DNS トンネリングを悪用して C&C 通信やデータの窃取を実行しつつも攻撃の隠蔽が可能になる．DNS トンネリングはクライアントサーバモデルで，攻撃者が DNS トンネリングを実行するには DNS トンネリングサーバを用意する必要がある．攻撃者は DNS トンネリングに使用する登録可能なドメインを取得し，この取得したドメインを管理する権威 DNS サーバが DNS トンネリングサーバとして機能する．

図 1 に DNS トンネリングクライアントと DNS トンネリングサーバ間の通信フローを示す．(1) 攻撃対象の企業ネットワークに侵入したマルウェアは DNS トンネリングクライアントとして送信したいデータ *outbound.data* を DNS クエリに含まれる FQDN のプレフィックスドメインに埋め込み，通常の DNS クエリと同じ手順で DNS トンネリングサーバが管理するドメイン *attacker.net* に送信する．(2)(3)(4)(5) この DNS クエリは通常の企業ネットワークに設置されている正規の DNS キャッシュサーバを経由し，企業ネットワークのゲートウェイを通過して権威DNSサーバに対して目的のドメインである *attacker.net* を管理するサーバを特定するまで繰り返し問い合わせを行う．(6)(7) 最終的に DNS トンネリングサーバに DNS クエリが到達し，攻撃者は DNS クエリに埋め込まれたデータである *outbound.data* を取得できる．(8)(9)(10) DNS トンネリングサーバは DNS レスポンスの中にマルウェアに対する新たな命令を埋め込むことで相互通信を実現し，攻撃者は継続して C&C 通信や情報漏洩が可能である.DNS トンネリングを悪用して情報漏洩を達成するためには，ユニークな情報を含む DNS クエリが必ず DNS キャッシュサーバ上でキャッシュミスし，図 1 中の(1)，(6)，(7)，(8)，(9)，(10)の通信フローを発生させなければならない.DNS キャッシュサーバは通信遅延を軽減し，システム全体の負荷を低減するために DNS キャッシュと呼ばれるキャッシング機構を備えており,過去の DNS クエリとそれに対応する DNS レスポンスを，その DNS レスポンスに含まれる Time to live (TTL)の期間だけ記憶する．DNS キャッシュが有効な間は，新規の問い合わせに対して DNS キャッシュを参照し，対応する応答が記憶されていればキャッシュヒットとなり，DNS キャッシュから直接応答するため，DNS キャッシュサーバよりも外部に DNS クエリが送信されず，図 1 中の(1)及び(10)の通信フローのみが発生する．そのため，DNS トンネリングにおいてはユニークな FQDN の使用または TTL 等のキャッシュ消去ポリシーによって DNS キャッシュが削除されるのを待つことで DNS クエリがキャッシュミスするように送信される．

また，DNS トンネリングにおいての外部サーバとの直接通信は一元して DNS キャッシュサーバが担っているため，DNS キャッシュサーバに注目した対策は DNS トンネリングに対して有効である．

公益財団法人電気通信普及財団
研究調査助成報告書　第40号　2025年度

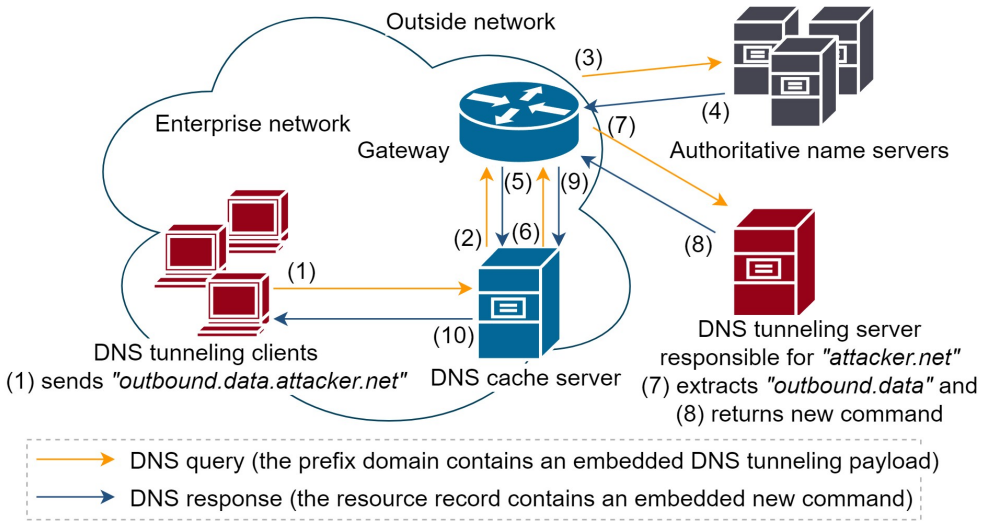

図1: DNS トンネリングによる情報漏洩フロー

DNS トンネリングの用途は様々あり,DNS トンネリングでは使用される DNS パケットに関する様々なチューニングが可能である. DNS クエリ及び DNS レスポンスに埋め込む送受信データ（つまり, DNS トンネリングのペイロード部)サイズを拡張させるために DNS クエリには非常に長い FQDN を使用し,DNS レスポンスには *TXT* や *CNAME* のような文字列を柔軟に扱えるレコードタイプを使用することでより多くの情報を埋め込み, DNS クエリの送信頻度を高く設定することで DNS トンネリングのスループット向上を図ることができる. これらは通常の DNS パケットとは大きく異なるが, 課金型ネットワークの料金を支払わずに利用する手段や, セキュリティが厳格なネットワークと外部ネットワーク間の Virtual Private Network 接続で合理的な選択肢として使用される [4]. 一方で, 情報漏洩のような攻撃のために DNS トンネリングを使用する場合, 攻撃を悟られないように通常の DNS トラヒックで頻繁に観測される *A* や *AAAA* レコード [5]を使用し, FQDN 長やラベル長を一般的な正常な DNS クエリに近づけ, DNS クエリの送信頻度を下げることで, DNS トラヒックに紛れ込ませ,ペイロード解析に基づく特徴量及び時間制約を持つトラヒック解析に基づく特徴量を難読化できるが, DNS トンネリングの実行時間が長期間必要となる. 実際に, 通常の DNS トラヒックで一般的に見られる *A* や *AAAA* レコードタイプを使用し, DNS クエリに含める送信データの文字列長が短く設定された DNS トンネリングも観測されている [6].

なお, 本研究では, Mozilla Foundation により維持管理されている Public suffix list [7]に登録されたサフィックスと, その直前の 1 ラベルから構成される部分（つまり, effective Top-Level Domain + 1 ラベル)を単にドメイン, FQDN からドメインを除いた残りの部分をプレフィックスドメインと定義する. この定義において, ドメインは各組織または個人が取得し運用する管理単位, プレフィックスドメインはドメインの管理者と DNS クライアント間で通信される情報として扱うことができる. 例えば, *login.mail.example.co.jp* という FQDN においては, *example.co.jp* がドメイン, 残りの *login.mail* がプレフィックスドメインに, *www.example.com* という FQDN においては, *example.com* がドメイン, *www* がプレフィックスドメインに相当する.

2 関連研究

2-1 DNS トンネリングの送信スループットに基づくチューニング

DNS トンネリングは送受信データを小さく分割し,DNS クエリの送信間隔を大きくするほど低スループットになり, 単一のクエリやレスポンスに含まれる DNS トンネリングの送受信データは小さく, 観測される DNS トンネリングトラヒックが一定の時間ウィンドウ内で減少する. そのような低スループット DNS トンネリングはペイロード解析のみでの異常検知は非常に困難である. そのため, 低スループット DNS トンネリング検知にはトラヒック解析に基づいた設計をする必要がある. ただし, トラヒック解析においても DNS トンネリ

ングのスループットが低下するほど特徴が難読化され，異常検知の精度低下につながることに留意する．

表1: 2021 年以降の関連研究における
DNS トンネリングツールのクエリ送信スループットに基づくチューニング状況

論文（年）	DNS トンネリングツール	パラメータ変更	最大クエリ送信間隔（秒）	最小 FQDN 長
Steadman ら [8]（2021）	Iodine [15], dnscat [16], DNSExfiltrator [17], DET [18]	なし	指定なし	指定なし
Bai ら [9]（2021）	Iodine, DNS2TCP [19]	なし	指定なし	指定なし
Ishikura ら [10]（2021）	dnscat2 [20]	クエリ送信間隔	**100**	指定なし
Zhan ら [11]（2022）	DNSExfiltrator	クエリ送信間隔, FQDN 長	20	**27**
Ziza ら [12]（2023）	DNSExfiltrator, Iodine	クエリ送信間隔, FQDN 長	10	40
Ozery ら [13]（2024）	Iodine, FrameworkPOS [21], Backdoor.Win-32.Denis [22]	クエリ送信間隔, FQDN 長	1.5	27 + ドメイン長
Guangyuan ら[14]（2024）	Iodine, dnscat2, DNS2TCP, dnspot [23], DNS-Shell [24], TUNS [25], Cobalt Strike [26], OzymanDNS [27], tcp-over-dns [28], AndIodine [29]	なし	指定なし	指定なし

トラヒック解析を採用している 2021 年以降の関連研究で使用された DNS トンネリングツールの DNS クエリに関する送信スループットを明らかにするため，表 1 にトンネリングクエリ送信間隔及びトンネリングクエリ長に基づいて整理した．なお，我々の調査した限りでは 2021 年以前のトラヒック解析に基づく DNS トンネリング検知手法の評価において，送信スループットのチューニングがなされている関連研究は存在しない．いくつかの関連研究では DNS トンネリングがそのまま実行されており，容易に検出できる DNS トンネリングに対する性能評価に留まっている [8,9,14]．Ishikura ら[10]は dnscat2 [20]の DNS クエリ送信時間間隔のみチューニングしており，最も大きい条件で 100 秒である．FQDN に関する特徴量は使用しないことから FQDN 長のチューニングはされていない．Zhan ら [11]は DNSExfiltrator の DNS クエリの平均送信時間間隔を最大で 20 秒，最小の FQDN 長を 27 バイトとチューニングしている．Ziza ら[12]のデータセットは公開されているため，3-1 節で DNS トンネリングトラヒックの特徴を分析する．Ozery ら[13]は FrameworkPOS [21]及び Backdoor.Win32.Dennis [22]で実験を行っており，DNS クエリ送信時間間隔がチューニングされている．FrameworkPOS は 6 カ月間で 5,600 万件のクレジットカード情報を流出するようにシミュレートされている．このことから，FrameworkPOS が生成する FQDN 長は少なくとも平均 27 以上のペイロードとドメインから構成される必要がある．また，Backdoor.Win32.Dennis は FQDN 長に関する詳細な生成ルールについて言及されていないが，Cobalt Kitty's operation security report analysis [30]に基づいたキープアライブを実行し，クエリ送信間隔が 1.5 秒という短い時間で実行される．このように，関連研究における DNS トンネリングのスループットに関するチューニングは，DNS クエリによる送信スループットに影響を及ぼす FQDN 長は著しく長く，DNS クエリの送信間隔は攻撃者がその送信間隔をさらに拡大する余地が残されている．

2-1 キャッシュミス及びユニーク情報に注目した特徴量

企業ネットワークに侵入したマルウェアが DNS トンネリングを実行して情報漏洩を達成する際，漏洩情報

を含む DNS クエリが外部に設置された DNS トンネリングサーバまで到着しなければならず，これは隠蔽できない決定的な痕跡として DNS キャッシュサーバ上に残る．それゆえに，キャッシュミスやユニークな FQDN 情報に基づいた特徴量は DNS トンネリングを明確に特徴付け，攻撃者にとっても難読化することが困難である．ただし，それらの優れた特徴量を使用している関連研究においても攻撃者が検出を回避して DNS トンネリングを実行できる余地が残っている．

Ishikura らが提案したフィルタはキャッシュミスからインスピレーションを受けて定義されたアクセスミスを使用しており，DNS トンネリングツールの種類を問わず検出できるが，時間ウィンドウを採用していることから，その時間ウィンドウを超える送信間隔で DNS トンネリングクエリを送信することで検出を回避できる．Ozery らが提案したフィルタではユニークなプレフィックスドメイン情報をドメイン単位で蓄積し，長期間にわたって実行される DNS トンネリングにも対応できるが，複数ドメインを利用した DNS トンネリングを検出することは極めて難しい．

3 DNS トラヒック解析

本研究では，DNS トンネリングトラヒックについて公開されているデータセット及び企業の社内 DNS トラヒックについて解析し，DNS トンネリング検知における課題や悪性トラヒックを特徴づけるような統計的指標を調査する．

3-1 DNS 公開データセット解析

本節では，正常 DNS トラヒックと異常 DNS トラヒックの両方を提供し，最先端の DNS トンネリング検知手法である Ozery らの研究において評価されている Ziza らのデータセットに注目して調査する．Ziza らのデータセットはセルビア国内の主要なインターネットサービスプロバイダ（ISP）が運用するプライマリ DNS サーバにおいて，24 時間にわたり観測された約 5,200 万件の DNS クエリログを基に構築されている．このデータセットは ISP の一般ユーザからの正常な DNS クエリと意図的に生成された 89,843 件の DNS トンネリングクエリから構成される．DNS トンネリングクエリは Iodine [15]および DNSExfiltrator [17]を用い，DNS クエリの FQDN 長，エンコーディング方式，クエリ送信間隔などのパラメータを多様に変化させて生成している．さらに，検知回避を目的として文字エントロピーを意図的に低減させる独自のエンコーディング方式である Base32map を DNSExfiltrator に実装し，より巧妙な DNS トンネリングトラヒックを生成している．これらに加え，ログ内で発見されたアンチマルウェアツールによるデータ転送など，技術的に情報窃取に該当する DNS クエリも異常として分類され，異常トラヒックには合計 3 種類のドメインが含まれている．

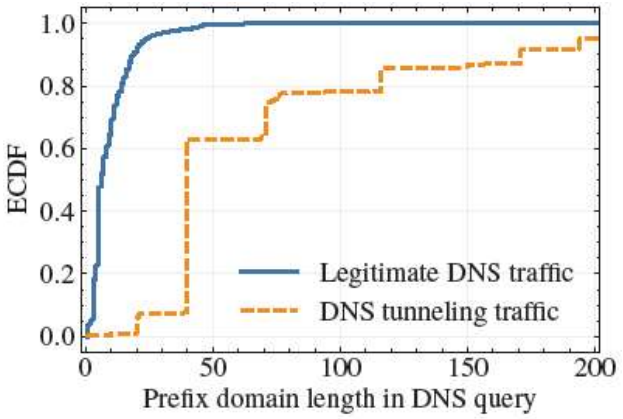

図 2: プレフィックスドメイン長の ECDF

正常 DNS クエリ及び DNS トンネリングクエリのプレフィックスドメイン長の Empirical cumulative distribution function (ECDF) を図 2 に示す．Ziza らのデータセットでは，正常 DNS トラヒックのプレフィックスドメイン長の中央値は 6 であるのに対して DNS トンネリングトラヒックは 40 と，正常 DNS トラヒックから乖離している．平均値は正常 DNS トラヒックの 8.08 に対して DNS トンネリングトラヒックは 71.08 と 8 倍よりも大きく，プレフィックスドメイン長に基づく特徴量または統計的指標に注目することで容易に検知可能であることがわかる．正常

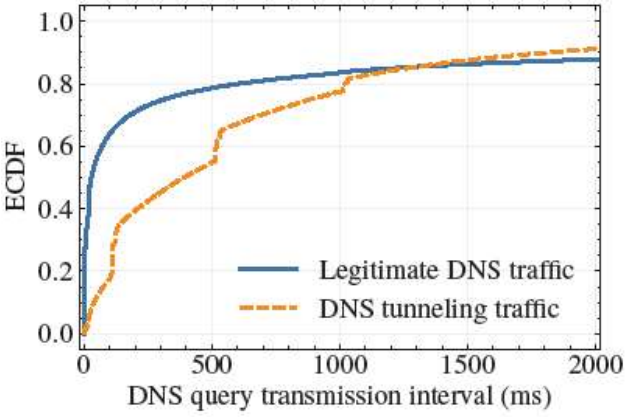

図 3: ドメイン毎のクエリ送信間隔に関する ECDF

DNS トラヒック及び DNS トンネリングトラヒックにおけるドメイン毎の DNS クエリ送信間隔に関する ECDF を図 3 に示す．Ziza らのデータセットでは 90%以上のクエリ送信間隔が正常 DNS トラヒックでは 3,096 ミリ秒であるのに対して，DNS トンネリングトラヒックでは 1,818 ミリ秒であり，DNS トンネリングは短い間隔でクエリを送信する傾向にあることがわかる．攻撃者は異常検知されることを回避するため，これらの公開データセットよりも DNS トンネリングクエリの送信間隔を引き延ばし，正常 DNS トラヒックに埋もれるように通信を分散させることも可能である．

3-2 企業 DNS トラヒック解析

本研究では企業ネットワーク内の DNS キャッシュサーバ上で観測された DNS トラヒックを用いて，正常 DNS トラヒックに関する特徴をキャッシュミスやユニークな情報に基づく特徴について解析する．解析には株式会社オプテージの社内ネットワークの DNS キャッシュサーバで 18 日間収集した DNS トラヒックを使用する．収集期間中に観測された DNS パケットは約 2,300,000,000 である．

ドメインに属するラベル数に着目し，ドメイン毎に観測されたラベル数に基づいたドメインランキングを図 4 に示す．FQDN のラベル部分は一般的にサービス単位で使い分けられるため，人気のあるサービスを多く提供するドメインでは，より多くのラベル数が観測され，ジップの法則に従っていることがわかった．一方で，全体で観測されたドメイン数が 28,098 であるのに対してラベル数が高々1 つの（つまり，FQDN が 1 種類しか観測されなかった）ドメインが 25,333 ドメインと 90%以上を占め，テール部分が非常に長いことがわかった．これは，攻撃者が意図的に多数のドメインを使用してドメイン単位の異常を示す特徴を分散，難読化させた場合，そのような多数の悪性ドメインが正常なドメインのテール部分に紛れる可能性を示している．

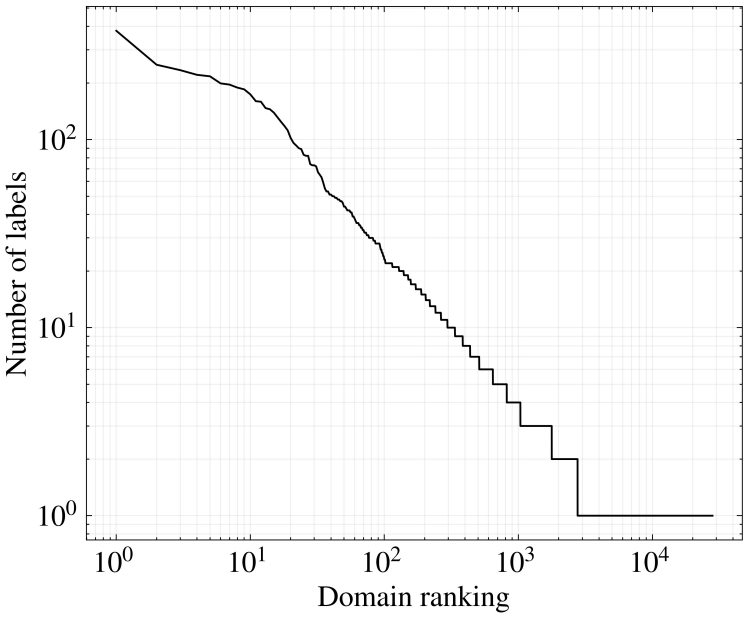

図 4: ドメイン毎のラベル数に基づいたドメインランキング

FQDN のラベル長に関する ECDF を図 5 に描く．ラベル長の 90%以上が 10 文字以内で，ラベル長の平均値は 7.0 である．これらは，表 1 に示したようなチューニングされた DNS トンネリングツールの FQDN 長が，実際の DNS トラヒックとは大きく異なり，既存研究において評価のために使用された DNS トンネリングトラヒックがラベル長に基づく特徴量または統計的指標に注目することで容易に検知可能であることを示す．

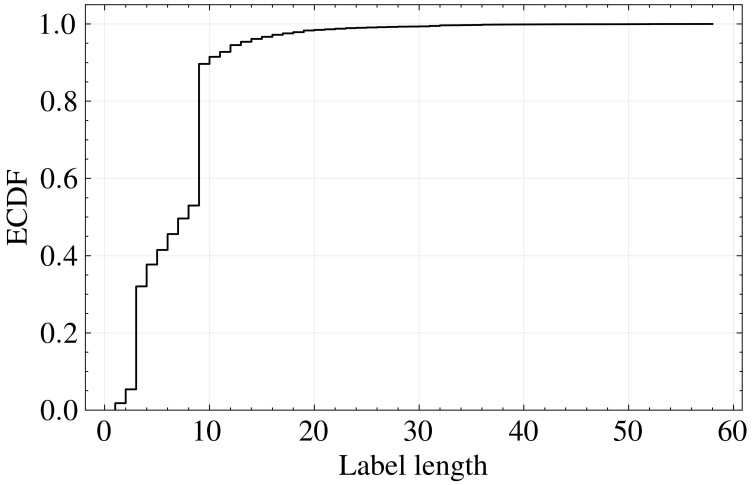

図 5: ラベル長に関する ECDF

4 まとめと今後の展望

本調査において，低スループットかつ複数のドメインを使用した DNS トンネリング検知に関する研究が十分になされておらず，容易に検知できるような DNS トンネリングに対する評価に留まっていることが分かっ

た．また，ドメイン単位で抽出した統計的指標だけでは，多数の正常なドメインに紛れ，そのような指標に基づく特徴が難読化される可能性を示した．本研究ではさらに，低スループットかつ複数ドメインを使用した DNS トンネリング検知に向け，企業のような管理ネットワークにおいて有用なユーザ単位での統計的指標に注目して調査を進め，有効な対抗策の確立を目指す．

本研究のさらなる展望として，複数ドメインを併用した DNS トンネリングと多量のドメインが観測されるという点で近い特徴を示す可能性が高い Domain Generation Algorithm (DGA)型ボットネットの検知を目指す．DGA 型ボットネットは，ボットが定期的にアルゴリズムに基づいて多数のドメイン名を生成し，その中から事前に攻撃者が登録した C&C サーバと通信することで，攻撃者とボット間で送受信を行う仕組みである．従来の固定ドメイン型ボットネットに比べ，DGA によって生成されるドメインは，一見ランダムかつ多様であり，ブラックリストによるドメインベースのアクセス制御では効果的に防御することが困難である．また，C&C サーバの所在が頻繁に変化するため，一時的な遮断では通信を完全に断つことができず，ボットによる持続的なコマンドの受信が可能となる．さらに，DGA はしばしば通常の DNS トラフィックに偽装されるため，企業の監視システムによる検知が難しい．特に，企業ネットワークにおいては日常的に大量の DNS 問い合わせが発生することから，DGA による不審なドメイン名のクエリが平常時のノイズに紛れて可視化されにくいという問題がある．実際に，初期感染後に DGA を用いて外部と通信しながら，情報窃取や追加マルウェアのダウンロード，さらにはランサムウェアの実行へと至る例も報告されている．このような持続的かつ秘匿性の高い攻撃は，企業の機密情報流出や業務停止といった深刻な被害を引き起こす可能性があり，対抗策の導入が強く求められる．

【参考文献】

[1] RFC 1034, Network Working Group, dOI: 10.17487/RFC1034 (Nov. 1987).
URL https://www.rfc-editor.org/info/rfc1034

[2] RFC 1035 Domain Names- Implementation and Specification. URL https://tools.ietf.org/html/rfc1035

[3] R. Li, B. Liu, C. Lu, H. Duan, J. Shao, A worldwide view on the reachability of encrypted dns services, in: Proceedings of the ACM Web Conference 2024, WWW '24, Association for Com puting Machinery, New York, NY, USA, 2024, p. 1193–1202. doi:10.1145/3589334.3645539.
URL https://doi.org/10.1145/3589334.3645539

[4] Understanding dns tunneling traffic in the wild.
URL https://unit42.paloaltonetworks.com/dns-tunneling in-the-wild/

[5] K. Hasegawa, D. Kondo, M. Osumi, H. Tode, Collaborative defense framework using fqdn-based allowlist filter against dns water torture attack, IEEE Transactions on Network and Ser vice Management 20 (4) (2023) 3968–3983. doi:10.1109/TNSM .2023.3277880.

[6] Dns tunneling in the wild: Overview of oilrig 's dns tunneling.
URL https://unit42.paloaltonetworks.com/dns-tunneling in-the-wild-overview-of-oilrigs-dns-tunneling/

[7] Public suffix list. URL https://publicsuffix.org/

[8] J. Steadman, S. Scott-Hayward, Dnsxp: Enhancing data exfil tration protection through data plane programmability, Com puter Networks 195 (2021) 108174.
doi:https://doi.org/10 .1016/j.comnet.2021.108174.
URL https://www.sciencedirect.com/science/article/pii/ S1389128621002310

[9] B. Huiwen, L. Weiwei, L. Guangjie, D. Yuewei, H. Shuhua, Application behavior identification in dns tunnels based on spatial-temporal information, IEEE Access 9 (2021) 80639 80653.
doi:10.1109/ACCESS.2021.3085500.

[10] I. Naotake, K. Daishi, V. Vassilis, I. Iordan, T. Hideki, Dns tun neling detection by cache-property-aware features, IEEE Trans actions on Network and Service Management 18 (2) (2021) 1203–1217.
doi:10.1109/TNSM.2021.3078428.

[11] M. Zhan, Y.Li, G. Yu, B.Li, W.Wang, Detecting dns over https based data exfiltration, Computer Networks 209 (2022) 108919. doi:https://doi.org/10.1016/j.comnet.2022.108919.
URL https://www.sciencedirect.com/science/article/pii/ S1389128622001104

[12] K. ˇ Ziˇ za, P. Tadi´c, P. Vuleti´ c, Dns exfiltration detection in the presence of adversarial attacks and modified exfiltrator be haviour, International Journal of Information Security 22 (6) (2023) 1865–1880. doi:10.1007/s10207-023-00723-w. URL https://doi.org/10.1007/s10207-023-00723-w
[13] Y. Ozery, A. Nadler, A. Shabtai, Information-based heavy hit ters for real-time dns data exfiltration detection, Network and Distributed System Security Symposium (2024). doi:https: //dx.doi.org/10.14722/ndss.2024.24388.
[14] G. Guangyuan, N. Weina, G. Jiacheng, G. Dujuan, L. Song, Z. Mingxue, Z. Xiaosong, Graphtunnel: Robust dns tunnel de tection based on dns recursive resolution graph, IEEE Trans actions on Information Forensics and Security 19 (2024) 7705 7719. doi:10.1109/TIFS.2024.3443596.
[15] iodine. URL https://github.com/yarrick/iodine
[16] dnscat. URL http://tadek.pietraszek.org/projects/DNScat/
[17] DNSExfiltrator. URL https://github.com/Arno0x/DNSExfiltrator
[18] DET. URL https://github.com/sensepost/DET
[19] dns2tcp. URL https://github.com/alex-sector/dns2tcp
[20] dnscat2. URL https://github.com/iagox86/dnscat2
[21] Three Month FrameworkPOS Malware Campaign Nabs ˜43,000 Credit Cards from Point of Sale Systems. URL https://www.anomali.com/blog/three-month-framewor kpos-malware-campaign-nabs-43000-credits-cards-from poi
[22] Backdoor.Win32.Denis. URL https://otx.alienvault.com/pulse/590314fb6575a0374 6de87a8
[23] dnspot. URL https://github.com/mosajjal/dnspot
[24] DNS-Shell. URL https://github.com/sensepost/DNS-Shell
[25] tuns. URL https://github.com/lnussbaum/tuns
[26] Cobalt Strike. URL https://www.cobaltstrike.com/
[27] OzymanDNS- Tunneling SSH over DNS. URL https://malicious.link/post/2009/2009310ozymandns tunneling-ssh-over-dns-html
[28] TCP-over-DNS. URL https://analogbit.com/software/tcp-over-dns/
[29] andiodine. URL https://gitlab.com/andiodine/andiodine
[30] A. Dahan, Operation cobalt kitty. cybereason labs analysis. URL https://cdn2.hubspot.net/hubfs/3354902/Cybereason% 20Labs%20Analysis%20Operation%20Cobalt%20Kitty.pdf

（注書き）本研究におけるデータセットの解析は，株式会社オプテージと大阪公立大学が秘密保持契約を締結し，大阪公立大学 情報学研究科倫理委員会の承認を受けて実施している．

〈発　表　資　料〉

題　　名	掲載誌・学会名等	発表年月