修士論文

DNS Exfiltration 抑止を目的とした分散ハッシュテーブルに基づく名前解決システム

高須賀 昌烈

2020 年 3 月 15 日

奈良先端科学技術大学院大学
先端科学技術研究科

本論文は奈良先端科学技術大学院大学先端科学技術研究科に
修士 (工学) 授与の要件として提出した修士論文である。

高須賀 昌烈

審査委員：

門林 雄基 教授　（主指導教員）
笠原 正治 教授　（副指導教員）
林 優一 教授　（副指導教員）
妙中 雄三 准教授　（副指導教員）

DNS Exfiltration抑止を目的とした分散ハッシュテーブルに基づく名前解決システム*

高須賀 昌烈

内容梗概

キーワード

ドメインネームシステム，情報流出，ネットワークセキュリティ，分散ハッシュテーブル

*奈良先端科学技術大学院大学 先端科学技術研究科 修士論文, 2020年3月15日.

A Name Resolution System to prevent DNS Exfiltration based Distributed Hash Table*

Shoretsu Takasuka

Abstract

Keywords:

Domain Name System(DNS), Information Leakage, Network Security, Distributed Hash Table(DHT)

*Master's Thesis, Graduate School of Information Science, Nara Institute of Science and Technology, March 15, 2020.

目 次

図 目 次

表 目 次

1. 序論

1.1 背景

ドメインネームシステム (Domain Name System, DNS) は，ドメイン名 (E.g. www.example.com) をインターネット上でのノードの住所を表す IP アドレス (E.g. 93.184.216.34) に変換する機能を担っており，DNS を通じて特定した宛先に問い合わせることで我々はサービスにアクセスできている．現在のインターネットの利活用において，名前解決の仕組みは極めて重要な技術の一つである．しかし，性善説的な当時の設計に伴い生じた脆弱性を利用した攻撃がいくつか報告されている．1987 年に RFC1034, RFC1035 [1, 2] として公開された DNS のコンセプトは，現在もなお本質的な仕組みは変更されることなく適用されている．その設計に起因する課題の内，DNS クエリのラベルおよびリソースレコード (Resource Record, RR) をデータ転送のメディアとする DNS Tunneling がある．

DNS Tunneling は，一般にフィルタリングされることが少ない DNS の特徴と DNS がデータ転送のメディアとして機能しているとは想像しない人の認知の隙間をついた手法であり，ファイヤー・ウォールや IDS/IPS といったセキュリティラインを突破するために使用される．このように本来の目的とは違う方法でデータを転送する手法は，一般に秘匿通信 (Covert Channel) と呼ばれる [3]．DNS Tunneling は，秘匿通信の代表例であり，マルウェアと C2(Command & Control) サーバとの通信の秘匿手法，または，ターゲットから取得したデータを外部に流出させるといった目的実行の手段として，実際のインシデントで広く利用されている [5, 6, 7, 8, 9, 10, 11, 12]．従来の DNS Tunnnling に対するアプローチには，検知による手法が採用されてきた．DNS Tunneling による DNS クエリは，以下 (1) に示すように，転送量に比例して長いラベルを持ち，ラベルとしての文字列制約を満たすためのエンコーディングによって高いエントロピーを示す特徴がある．

$$\begin{aligned} &obqxg43xmqytcmjr.exfil.com \\ &base32(password1111) = obqxg43xmqytcmjr \end{aligned} \tag{1}$$

また，インタラクティブなシェルなど双方向の通信を DNS Tunneling で実現し

ようとする場合，時間あたりに高頻度なトラフィックが発生するという特徴が現れる．このような特徴に基づき，パターンマッチングや機械学習，文字列分布などのメソッドを用いた検知手法が過去に多数考案されてきた [13, 14, 15, 16, 17, 18]．それら検知手法は，かなり高い精度で分類を実現しているものがあるが，DNS Tunneling として検知する対象としているパケットには一般に利用することができる DNS Tunneling ツールキット [19, 20, 21] が使用され，それらは特に過剰な特徴量を示し，明らかに正規の DNS クエリと異なる特徴がある．高い精度を示す従来の検知手法だが，しかし，それらを迂回する手法として，1 回あたりの転送データ量を少なくすることで特徴量を減らす Low Throughput なバイパス手法，また，パケット間のインターバルを数日・数ヶ月と長期化させることでファイル肥大から一定期間しか保存されることがないログ管理の隙間を突いた Slow な Tunenling 手法があり，従来の検知手法では対応することが困難である．悪意を持つユーザの視点として，1bit でも転送できることは秘匿通信として利用することができるため，転送量の少なさは軽視されるべきではない．

他方で，DNS は初めに述べたように，現在のインターネットの根幹技術として根ざしており，抜本的な改変は期待されない．すなわち，既存の DNS による名前解決のメカニズムに大幅な改変を加えないという制約下で，Tunneling に対処することが現実的な最適解であると考える．

1.2 目的

本研究では，既存の DNS の名前解決メカニズムの大部分を流用することが一部の改変に留めながら，DNS を用いたデータ転送としての機能の排除を実現する次世代の名前解決メカニズムを提案する．

1.3 本論構成

本稿の構成は以下の通りである．まず第 2 章で，準備として，DNS プロトコル・秘匿通信・Tunneling メカニズム・分散データベースの 4 点について説明する．第 3 章では，関連研究としてトラフィックおよびペイロード特徴に基づいた

検知手法を説明し，それら手法がLow Throughput 手法・Slow Tunneling 手法に対して検知が困難であることを説明する．第４章で提案手法とその実装について述べ，第５章で提案手法の性能評価と考察行い，第６章で残留する脅威モデルについて議論する．最後に，第７章で結論と今後の課題について述べていく構成になっている．

2. 準備

本章では，本論において核となる技術内容・特徴およびそのメカニズムについて説明する．

2.1 DNS プロトコル

2.1.1 概要

DNS(Domain Name System) は，インターネットに接続された無数のコンピュータを一意に識別するための IP アドレスを，人が認識しやすいドメイン名に変換するシステムである．元来，インターネット上でのホストの識別には IP アドレスが使用されてきた．しかし，32bit の名前空間で 10 進数表記の IPv4(E.g. “192.168.0.1”)，もしくは，128bit の名前空間で 16 進数表記の IPv6(E.g. “2001:200:16a:8::230”) は，人にとって認識しにくいものである．そのため，自然言語のようにアルファベットや数字で表記する方法が取られ，当初はその対応表である hosts.txt が中央集権的に管理されていたが，やがてホスト数の増大に伴い管理が困難になっていき，提案されたのが対応表を分散的に管理する DNS である．

DNS のシステムアーキテクチャは，クライアント・サーバ構成で成り立っている．一般に，クライアントがドメインを問い合わせた場合，サーバはドメインに対応づけている IP アドレスを応答することで，クライアントはドメインに対応づけられた IP アドレスを解決することができる．ドメインから IP アドレスの解決は正引きと呼ばれ，IP アドレスからドメインの解決を逆引きと呼ぶ．

2.1.2 ドメイン

ドメインは，数字とアルファベットおよびハイフン (“-”) の文字列で表記され，最大長は 63 オクテットと定義されている．また，ドメインはルートを頂点とする階層構造で構成され，各階層にはドメインを管理する主体として権威サーバが存在し，管理主体を委譲していくことによって分散的にデータベースを管理する仕組みで動作する．

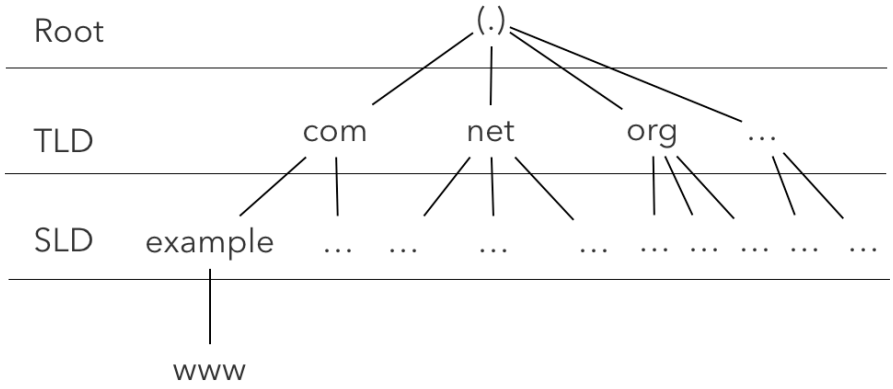

図 1　ドメインの名前空間

ドメイン名は，ドメインに相当するラベルをドット区切りで表され，最大長は 255 オクテットである．ドメイン名は右から順に階層序列が表現され，ドットで表現されるルートは一般には省略される．最も右に位置づくラベルが TLD(Top Level Domain) であり，その TLD から n 番目 ($n \mid n \in \mathbb{N}$) のラベルが第 n レベルドメインである．

TLD を大別すると，“.com” や “.net” をはじめとした特定分野別の gTLD(global Top Level Domain)，“.jp” や “.ch” のような国ごとに割り当てられている ccTLD(Country Code Top Level Domain) の二つに分けられる．

2.1.3　ノードの種類

DNS におけるノードは，機能に応じて 3 つに分けられる．

- Stab Resolver
- Full Service Resolver
- Authorization Server

スタブリゾルバは，名前解決の問い合わせるを依頼するクライアントノードである．フルサービスリゾルバ (キャッシュサーバ・リカーシブサーバとも呼称される) は，スタブリゾルバからの名前解決問い合わせをハンドリングするノードである．過去に問い合わせられた情報をキャッシュとして保持する機能と，レコー

ド情報を保持・提供する権威サーバに対する再帰問い合わせを行う機能を担う．一般に，フルサービスリゾルバは，“root.hints”というルート権威サーバとそのアドレスが対応づけられたファイルを保持しており，再帰問い合わせの際にはこのファイルに基づき，最初の宛先となる権威サーバのアドレスを解決する．権威サーバは，レコード情報を保持するサーバノードであり，フルリゾルバからの転送される問い合わせ依頼に応答する．

2.1.4 リソースレコード

ドメイン名に関連づけられる情報は，リソースレコード (Resorce Record, RR) と呼ばれる．リソースレコードは，タイプが定義されており，目的ごとに使用されるタイプが異なる．最も一般的なレコード，A レコードタイプは，ドメインに対して IPv4 アドレスを関連づけるために使用される．名前解決において，クライアントはドメイン名とそのドメインに関連づけられたリソースレコードを指定する．これによって，クライアントは，ドメインに関連づけられたリソースレコード情報を取得することができる．表 2.1.4 は，主要なリソースレコードである．

2.1.5 名前解決

いま，クライアントから “www.example.com” の IPv4 アドレスについて問い合わせられたとする．はじめに，クライアントであるスタブリゾルバは，スタブリゾルバと同一セグメント内のフルサービスリゾルバもしくは，ネットワークセグメントに依らずインターネット上のどのクライアントからもアクセスできるパブリックなフルサービスリゾルバ (オープンリゾルバ，パブリックリゾルバとも呼称される) に問い合わせる．フルサービスリゾルバははじめに，ドメイン名が “www.example.com” で，リソースレコードが “A” の応答レコードがキャッシュにあるかどうかを判別する．キャッシュにヒットした場合にはキャッシュの情報をクライアントに応答され，ヒットしなかった場合には，root.hints ファイルを参照しルート権威サーバにリクエストパケットを転送する．クエリ (問い合わせ) を受け取ったルート権威サーバは，ルートゾーン内の “com” ドメインが委譲された

表 1 主要リソースレコード一覧

タイプ	値	目的
A	1	ホストの IPv4 アドレス
NS	2	権威サーバ
MF	4	メール転送サーバ
CNAME	5	別名
SOA	6	権威ゾーンの開始
NULL	10	NULL(実験用)
PTR	12	ドメイン名のポインター (逆引き)
HINFO	13	ホスト情報
MINFO	14	メールボックスおよびメールリスト情報
MX	15	メール交換
TXT	16	任意文字列

権威サーバのアドレスを応答する．次に，フルサービスリゾルバは，“com”ドメインが委譲された権威サーバに対し同様のクエリを転送する．“com”ドメインを管理する権威サーバは，同様にして，“com”ゾーン内の “example.com”が委譲された権威サーバのアドレスを応答する．フルリゾルバは，“example.com”ドメインを委譲された権威サーバ宛に同様のクエリを転送する．“example.com”ゾーンを管理する権威サーバは，保持するゾーンファイルからクエリされたドメインのリソースレコードについて探索し，探索の結果としてレコード情報をフルサービスリゾルバに応答する．フルサービスリゾルバは，権威サーバからの応答されたレコード情報の TTL(Time To Live) の期間レコード情報をキャッシュしたのち，クライアントのスタブリゾルバに結果を応答する．このようにして，再帰的な問い合わせの仕組みに基づき名前解決がとり行われる．

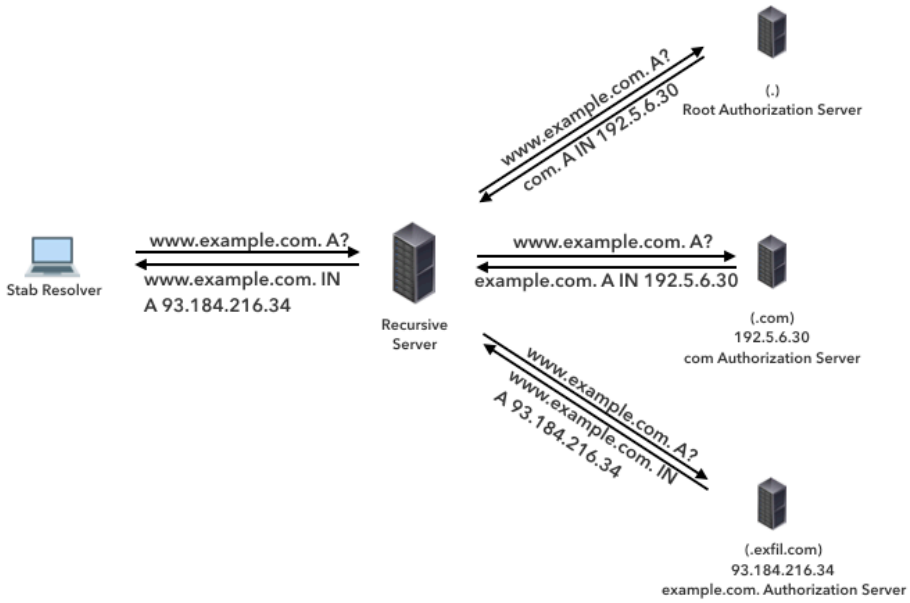

図 2 DNS による名前解決

2.2 DNS Tunneling

DNS は，2.1 節で示すように，ドメイン名とそのドメインに関連づけられたリソースレコードを解決するシステムである．現在のインターネットにおいては，極めて重要な機能を担うプロトコルスタックの一つである．他方で，本来他方，クライアントサーバのアーキテクチャに基づき，スタブリゾルバの問い合わせ情報が権威サーバに直接転送される仕組みは，データ転送の仕組みでもある．これによって，

2.2.1 流出メソッド：DNS Exfiltration

DNS を利用して情報を外部に転送するには，初めにデータの宛先となるドメイン (E.g. exfil.com) を作成することになる．転送する際のキャリアとなる DNS クエリのラベルには，使用できる文字列は数字・アルファベット・ハイフン (“-”) である必要があるため，一般に Base32・64 を用いて転送したい情報をエンコー

ディングすることでこの制約条件を満たす．用意できた QNAME(E.g. arbitrary-string.exfil.com) について，例えば A のリソースレコードをクエリすると，サブドメインの存在の有無に関わらず，宛先となるドメイン (exfil.com) に任意の情報を転送することができるという具合である．以下 3 に，DNS Exfiltration のメカニズムについて図解する．

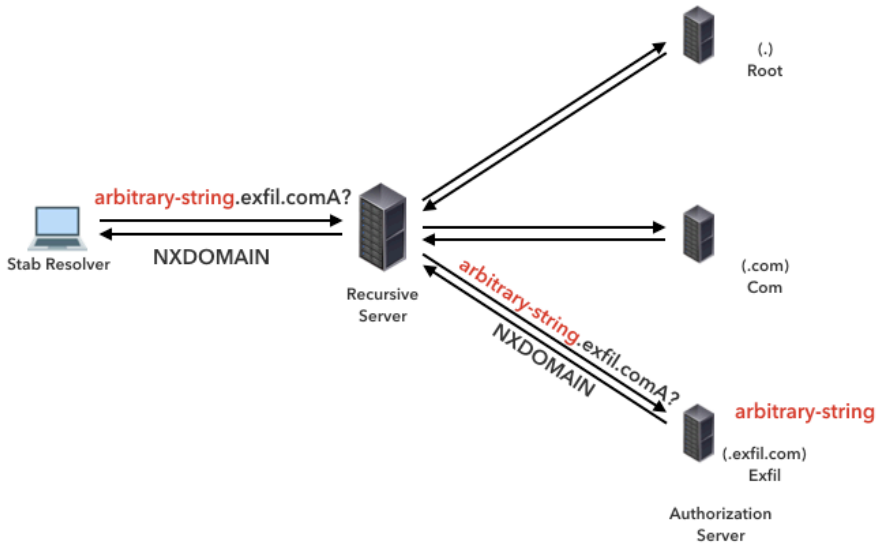

図 3　arbitrary-string という任意の文字列が，DNS クエリのラベル部を用いて，事前に用意した権威サーバ (exfil.com) に転送される様子．

2.2.2　流入メソッド：DNS Infiltration

また，管理する権威サーバのドメインに適当なホスト名 (E.g. www) を作成し，そのホスト名のリソースレコード (E.g. TXT) に情報を付与していた場合には，そのホストへの問い合わせを通じて逆方向，すなわち権威サーバから任意の情報を転送することができる．DNS のリソースレコードを転送キャリアとする流入通信のメカニズムを図解した様子が，4 である．

このような DNS を用いて双方向な通信手法が DNS Tunneling である．

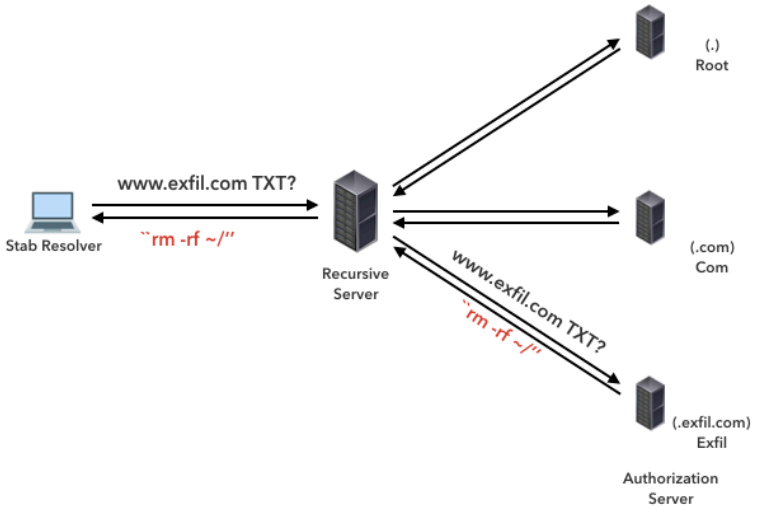

図 4 事前に TXT レコードに登録された情報を問い合わせることで，権威サーバからの命令情報を取得している様子.

3. 関連研究

本章では，はじめに既存の DNS Tunneling に対するアプローチとして提案されている検知アプローチを取り上げ，現在の検知に基づく対策の課題として，Low Throughput 手法と Slow な転送手法というバイパス手法に対処できないことを明らかにする．次に，これまでに提案されてきた P2P ベースの名前解決システムを説明し，提案手法との違いを示す．

3.1 特徴量に基づく検知手法

3.1.1 閾値推定

3.1.2 機械学習に基づくモデル

DNS Tunneling メソッドを使用した時の DNS クエリは，**??**で述べたような特徴が現れる性質がある．この性質に基づき，これまでに多数の検知手法が提案されてきた．

Born ら [13] は，

3.2 新しいアーキテクチャに基づく抑止手法

3.2.1 Peer-to-Peer ネットワークを利用した名前解決システム

これまでに，DNS における～の課題に対して，P2P に基づいた名前解決システムは数多く提案されてきた．

3.2.2 Blockchain を利用した名前解決システム

3.3 課題

4. 提案手法

本章では，はじめに提案手法の概要について説明する．続いて，システムの具体的なアーキテクチャ・プロトコル・動作について説明する．

4.1 分散ハッシュテーブルを利用した名前解決システム：REVRES

関連研究の課題 3.3 で示すように，これまでの対策手法では DNS Exfiltration 2.2.1 に対処するには限界がある．そこで，提案する新しい名前解決システムが，分散ハッシュテーブルをサーバ群で連携させ，クライアントからの名前解決問い合わせに対する応答サーバをランダムにさせることで，情報流出のメソッドとして利用される DNS Exfiltration の発生を抑止する名前解決システム，REVRES(REcord Value Resolution System) である．

4.2 アーキテクチャ

REVRES は，サーバ同士が相互連携するサーバ群とクライアントで構成されるクライアントサーバアーキテクチャで動作する．REVRES において，スタブリゾルバとフルサービスリゾルバとの間の手続きは，既存の DNS と変わらない．しかし，REVRES のサーバは，フルメッシュのネットワーク基盤のもとで各サーバが相互に連携することで，リソース情報を分散して管理する．REVRES におけるレコード情報は，QNAME とレコードタイプを引数として，ハッシュ関数によって算出されるコンテンツ ID がシンボルとして紐づけられる．

全てのフルサービスリゾルバは，シンボルとそのシンボルに紐づいたレコード情報を管理するサーバを対応づけた表を保持し，この対応表に基づきコンテンツ ID からサーバを一意に特定する．複数のサーバが，アルファベットおよび数字の順序 (a $\rightarrow$ z, 0 $\rightarrow$ 9) で並んだハッシュテーブルの連続した範囲を管理し合い，担当の範囲下にあるコンテンツ ID に紐づいたレコード情報を担当サーバが管理する．対応表 4.2 には，どこからのどこまでのハッシュテーブルの範囲をどのサーバが管理するのかについて記述されている．

表 2 ハッシュ値の範囲とその範囲を管理するサーバに関する対応表

管理範囲	ホストアドレス	ホスト名
(00…00, 4z…zz)	192.35.51.30	com.
(50…00, az…zz)	192.5.6.30	net.
(b0…00, gz…zz)	199.249.112.1	org.
(h0…00, mz…zz)	213.248.216.1	uk.
(n0…00, sz…zz)	199.254.31.1	info.
(t0…00, zz…zz)	194.0.0.53	de.

フルサービスリゾルバはQNAMEとリソースレコードタイプをキーとしてサーバに問い合わせるバリューが応答されるKVSモデルに基づく名前解決システムである．

QNAMEとリソースレコードタイプに基づき生成されるハッシュ値をキー，分散ハッシュテーブル上で保存されているレコード情報をバリューとするKVSモデルに基づく名前解決システムである．

REVRESでは，従来のDNSのエコシステムの内，スタブリゾルバからリカーシブサーバまでの手続きを継承することで，エッジノードにおけるシステムのマイグレーションに伴う課題を軽減する．全てのレコード情報は，レコードタイプとドメイン名もしくはIPアドレスの文字列和 (rtype+domain, rtype+ipaddress) をハッシュ関数に適用して算出されるハッシュ値をコンテンツIDとして，レコード情報に紐づっけr.

スタブリゾルバは既存のDNSと変更はなく，DNSクライアントとして，名前解決を依頼する主体として位置づくノードである．リカーシブサーバは，スタブリゾルバからの問い合わせに対してリソース情報を保持する主体に代理的に問い合わせ機能と，問い合わせた情報を一定期間キャッシュするキャッシュサーバとして機能するノードである．既存のDNSにおける権威サーバは，リカーシブサーバからの問い合わせに応答するマネージャと，リソース情報について作成・消去および更新などの操作をするプロバイダの二つに分けられる．REVRESにおいて，リソース情報は，オブジェクト (object) とリソースレコードタイプ (rtype)

を引数とするハッシュ関数から算出されるコンテンツ ID が紐づけられ，そのコンテンツ ID に基づきハッシュ空間上に対応づけられる．各マネージャは，ハッシュテーブル全体のうち連続した幾らかの管理範囲が割り当てられ，範囲下にあるコンテンツ ID に基づいたリソース情報を保存・管理する．このようにして，リソース情報は，特定の範囲ごとに分割されたハッシュテーブルにて分散的に管理される．マネージャ同士は，フルメッシュなネットワーク構造で接続し合い，各マネージャには地理的・意味的に類似なプロバイダが階層的な序列に基づき接続される．

プロバイダからリソース情報への操作リクエストがあった際には，リソース情報のコンテンツ ID を算出し，その ID が含まれるハッシュ空間を管理する担当マネージャに操作依頼を転送し，受け取った担当マネージャは直ちに，リソース情報への操作を実行する．

4.3 プロトコル

4.4 分散ハッシュテーブル

4.5 動作メカニズム

4.6 ハッシュ範囲に対する管理ノードの対応表

5. 評価

5.1 DNS Exfiltrationに対する定性評価

5.2 シミュレーション実験に基づく定量評価

5.2.1 シミュレーション実験構成

5.2.2 肥大化したリクエストペイロードサイズ

5.2.3 RTT(Round Trip Time)

5.2.4 トラフィック量

6. 考察

6.1 レプリケーションサーバごとのハッシュテーブル同期

6.2 クエリに対するハッシュ値計算の最適ノード

6.3 DNS Infiltration に対するリソースレコード

6.4 今後の課題

7. 結論

謝辞

ご指導ご鞭撻賜りありがとうございました.

参考文献

[1] P.V. Mockapetris. “Domain names - concepts and facilities. RFC 1034 (INTERNET STANDARD),” November 1987. Updated by RFCs 1101, 1183, 1348, 1876, 1982, 2065, 2181, 2308, 2535, 4033, 4034, 4035, 4343, 4035, 4592, 5936.

[2] P.V. Mockapetris. “Domain names - implementation and specification. RFC 1035 (INTERNET STANDARD),” November 1987. Updated by RFCs 1101, 1183, 1348, 1876, 1982, 1995, 1996, 2065, 2136, 2181, 2137, 2308, 2535, 2673, 2845, 3425, 3658, 4033, 4034, 4035, 4343, 5936, 5966, 6604.”

[3] ICANN, “What Is an Internet Covert Channel?,” August 2016. https://www.icann.org/news/blog/what-is-an-internet-covert-channel. (accessed 2019-12-12).

[4] S. Bortzmeyer. “DNS Privacy Considerations RFC 7626 (INTERNET STANDARD), ” August 2015.

[5] KrebsonSecurity. “Deconstructing the 2014 Sally Beauty Breach,” May 2015. https://krebsonsecurity.com/2015/05/deconstructing-the-2014-sally-beauty-breach/. (accessd 2019-11-30).

[6] IronNet. “Chirp of the PoisonFrog,” February 2019. https://ironnet.com/blog/chirp-of-the-poisonfrog/. (accessd 2019-11-30).

[7] Nick Hoffman. “BernhardPOS,” July 2015. https://securitykitten.github.io/2015/07/14/bernhardpos.html. (accessd 2019-11-30).

[8] Fireeye. “MULTIGRAIN – Point of Sale Attackers Make an Unhealthy Addition to the Pantry,” April 2016. https://www.fieeye.com/blog/threat-research/2016/04/multigrain_pointo.html. (accessd 2019-11-30).

[9] Palo alto Networks. “New Wekby Attacks Use DNS Requests As Command and Control Mechanism,” May 2016. https://unit42.paloaltonetworks.com/unit42-new-wekby-attacks-use-dns-requests-as-command-and-control-mechanism/. (accessd 2019-11-30).

[10] Kaspersky. “Use of DNS Tunneling for C&C Communications,” April 2017. https://securelist.com/use-of-dns-tunneling-for-cc-communications/78203/. (accessd 2019-11-30).

[11] CISCO Talos. "Spoofed SEC Emails Distribute Evolved DNSMessenger," October 2017. https://blog.talosintelligence.com/2017/10/dnsmessenger-sec-campaign.html. (accessd 2019-11-30).

[12] Cylance. "Threat Spotlight: Inside UDPoS Malware," Febrary 27 2018. https://threatvector.cylance.com/en_us/home/threat-spotlight-inside-udpos-malware.html. (accessd 2019-11-30).

[13] K. Born and D. Gustafson, "NgViz: detecting DNS tunnels through n-gram visualization and quantitative analysis," Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research, Oak Ridge, Tennessee, 2010, pp. 1-4.

[14] Cheng Qi, Xiaojun Chen, Cui Xu, Jinqiao Shi, Peipeng Liu, "A Bigram based Real Time DNS Tunnel Detection Approach," Procedia Computer Science, Volume 17, 2013, Pages 852-860.

[15] J. Liu, S. Li, Y. Zhang, J. Xiao, P. Chang and C. Peng, "Detecting DNS Tunnel through Binary-Classification Based on Behavior Features," 2017 IEEE Trustcom/BigDataSE/ICESS, Sydney, NSW, 2017, pp. 339-346.

[16] Asaf Nadler, Avi Aminov, Asaf Shabtai, "Detection of malicious and low throughput data exfiltration over the DNS protocol," Computers & Security, Volume 80, 2019, Pages 36-53.

[17] J. Steadman and S. Scott-Hayward, "DNSxD: Detecting Data Exfiltration Over DNS," 2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Verona, Italy, 2018, pp. 1-6.

[18] J. Ahmed, H. H. Gharakheili, Q. Raza, C. Russell and V. Sivaraman, "Monitoring Enterprise DNS Queries for Detecting Data Exfiltration from Internal Hosts," in IEEE Transactions on Network and Service Management.

[19] "OzymanDNS - Tunneling SSH over DNS," https://room362.com/post/2009/2009310ozymandns-tunneling-ssh-over-dns-html/, (accessed 2019-11-20).

[20] "iodine," http://code.kryo.se/iodine/, (accessd 2019-11-20).

[21] "DNScat2," https://github.com/iagox86/dnscat2, (accessed 2019-11-20).

付録

A. 発表リスト (国内研究会)

1. 高須賀 昌烈, 妙中 雄三, 門林 雄基, “非実在ドメインに対するネガティブキャッシュの拡張と再帰問い合わせハッシュ化の提案”, 電子情報通信学会情報ネットワーク研究会, 2019-10-ICTSSL-IN, 2019 年 10 月.