Statistische Analysen zur Erkennung von DNS- und DoH-Tunneling

Niklas Wilhelm
Fakultät für Informatik
Technische Hochschule Rosenheim
Rosenheim, Germany
niklas.wilhelm@stud.th-rosenheim.de

***Zusammenfassung*—Das Domain Name System (DNS) und darauf aufbauende Standards wie DNS over HTTPS (DoH) werden häufig von Angreifern verwendet, um Daten aus Unternehmensnetzwerken zu stehlen. Der Missbrauch dieser Protokolle zur Datenübertragung wird als DNS- bzw. DoH-Tunneling bezeichnet. Dieser Artikel analysiert insgesamt fünf Methoden, mit denen beide Tunneling-Techniken frühzeitig erkannt und somit die Kosten für Unternehmen reduziert werden können. Die Methoden basieren auf statistischen Analysen und lassen sich entweder dem Payload- oder dem Traffic-basierten Ansatz zuordnen. Bei dem Payload-basierten Ansatz liegt der Fokus auf den Nutzdaten eines einzelnen Datenpakets. Methoden dieser Kategorie können Tunneling u.a. anhand des Record-Typs, der Paketgröße oder des Domainnamens erkennen. Methoden des Traffic-basierten Ansatzes betrachten eine Menge von Datenpaketen über einen bestimmten Zeitraum. Sie erkennen Tunneling bspw. anhand des Traffic-Volumens oder des Zeitabstands zwischen Datenpaketen.**

Die Verschlüsselung der DNS-Kommunikation über DoH macht die Erkennung von Tunneling schwieriger. Dieser Artikel zeigt auf, dass trotz dieser Herausforderung geeignete statistische Analysen existieren. Mit drei der fünf untersuchten Methoden ist es möglich, DoH-Tunneling aufzudecken. Eine weitere Erkenntnis ist, dass Unternehmen eine Kombination von Erkennungsmethoden einsetzen sollten. Ein Angreifer kann bspw. auf häufig genutzte Record-Typen mit geringer Bandbreite zurückgreifen, um einige Payload-basierte Erkennungsmethoden zu umgehen. Die Übertragung einer Datei über einen Record-Typ mit geringer Bandbreite führt allerdings zu einer ungewöhnlich hohen Anzahl an gesendeten Datenpaketen. DNS- bzw. DoH-Tunneling kann somit weiterhin über die Analyse des Traffic-Volumens identifiziert werden.

I. Einführung

Ein Datenleck bezeichnet ein Ereignis, bei dem sensible Datensätze verloren gehen oder gestohlen werden. Zu diesen Datensätzen gehören personenbezogene Daten wie Gesundheits- und Finanzinformationen sowie Geschäftsgeheimnisse [1]. Bei dem von IBM veröffentlichten *Cost of a Data Breach Report 2023* [1] wurden 553 Organisationen analysiert, die zwischen März 2022 und März 2023 von Datenlecks betroffen waren. Die in Abbildung 1 dargestellten Analyseergebnisse verdeutlichen die enormen Kosten, die bei einem Datenleck entstehen. So verzeichneten Unternehmen mit weniger als 500 Mitarbeitern durchschnittliche Kosten in Höhe von 3,31 Millionen USD. Bei Unternehmen mit mehr als 25.000 Mitarbeitern beliefen sich die durchschnittlichen Kosten auf 5,42 Millionen USD. Laut dem Bericht sahen sich 57 Prozent der betroffenen Unternehmen gezwungen, die Kosten an den Kunden weiterzureichen und die Preise für ihre Produkte zu erhöhen. Gemäß dem Angebot-Nachfrage-Modell aus der Volkswirtschaftslehre bedeutet eine Preiserhöhung normalerweise eine geringere Nachfrage [2] und folglich auch eine Schwächung der Marktposition. Ein Datenleck kann daher zu Wettbewerbsnachteilen führen.

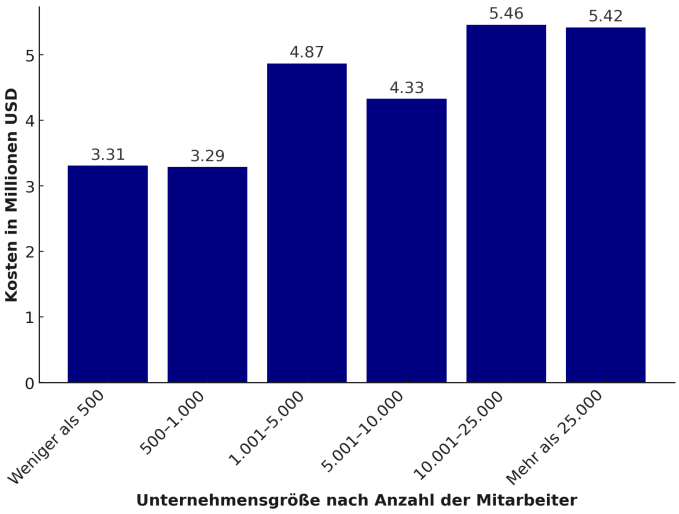

Abbildung 1. Kosten eines Datenlecks nach Unternehmensgröße [Eigene Darstellung, Daten aus [1] entnommen]

Angreifer missbrauchen oft die Namensauflösung des Domain Name Systems (DNS), um Daten unbemerkt aus einem Unternehmen zu schleusen [3]. Diese Methode ist besonders schwer zu erkennen, wenn Unternehmen Standards zur Verschlüsselung des DNS-Traffics verwenden [4]. Einer dieser Standards nutzt das HTTPS-Protokoll und ist als DNS over HTTPS (DoH) bekannt [5]. Die Übertragung von Daten über DNS und DoH wird als DNS- und DoH-Tunneling bezeichnet [4].

Ein Beispiel für DNS-Tunneling stellt der Supply-Chain-Angriff bei SolarWinds im Jahr 2020 dar [6]. SolarWinds ist ein US-amerikanisches Unternehmen, das eine Plattform namens Orion für die Überwachung und Verwaltung der IT-Infrastruktur anbietet. Angreifern ist es gelungen, den Build-Prozess zu infiltrieren und den Trojaner Sunburst in ein Update

der Orion-Plattform zu integrieren. Nachdem Kunden das Update installiert haben, hat Sunburst das Unternehmensnetzwerk infiltriert und unbemerkt Daten mittels DNS-Tunneling übertragen. Der Versicherungsschaden dieses Angriffs belief sich auf geschätzte 90 Millionen USD – mit ungefähr 18.000 betroffenen Unternehmen.

Aufgrund des häufigen Missbrauchs des Domain Name Systems sollten Unternehmen Programme für die Identifizierung von DNS- und DoH-Tunneling einsetzen. Eine frühzeitige Erkennung von Datenlecks kann die Kosten erheblich senken [1] und somit wettbewerbliche Nachteile minimieren. In diesem Artikel wird untersucht, wie sich DNS- und DoH-Tunneling mit statistischen Analysen erkennen lassen. Dabei wird zwischen Payload- und Traffic-basierten Erkennungsmethoden unterschieden [7]. Während sich die Payload-basierten Methoden auf die Nutzdaten eines einzelnen Datenpakets fokussieren, betrachten die Traffic-basierten Methoden eine Menge von Datenpaketen über einen bestimmten Zeitraum.

Die Methoden zur Erkennung von DNS- und DoH-Tunneling lassen sich in der Praxis auf unterschiedliche Weise einsetzen. Salat et al. [3] implementierten sie in der Bedrohungserkennungssoftware Suricata in Form von Regeln. Bei Überschreitung eines für die Regel festgelegten Grenzwertes wird ein Alarm ausgelöst. Die Methoden können alternativ als Merkmale (Features) dienen, um mittels Machine Learning einen Classifier zu trainieren. Ein Beispiel dafür liefern Moure-Garrido et al. [4] in einem Experiment zur Erkennung von DoH-Tunneling.

Dieser Artikel beginnt mit einer Einführung in die theoretischen Grundlagen des Domain Name Systems sowie des DNS- und DoH-Tunnelings. Anschließend werden mehrere Payload- und Traffic-basierte Methoden zur Erkennung dieser Tunneling-Techniken analysiert. Den Abschluss bildet ein Fazit, das die Analyseergebnisse zusammenfasst und einen Ausblick auf mögliche künftige Forschungsrichtungen gibt. Die Filterung von DoH-Paketen aus dem HTTPS-Traffic wird nicht behandelt. Dieses Thema ist Gegenstand anderer Forschungsarbeiten. Vekshin et al. [8] entwickelten zum Beispiel einen auf Machine Learning basierenden Classifier, der DoH-Pakete mit einer Genauigkeit von 99,9 Prozent identifizieren kann.

II. Domain Name System

Das Domain Name System erfüllt eine grundlegende Aufgabe im Internet: Es übersetzt für Menschen leicht zu merkende Domainnamen in IP-Adressen, die für das Routing von Datenpaketen im Netzwerk verwendet werden. Als dezentrales System setzt sich das DNS aus tausenden weltweit verteilten Nameservern zusammen [9]. Die Strukturierung dieser Nameserver folgt der in Abbildung 2 dargestellten Hierarchie.

An der Spitze der Hierarchie befinden sich Root-Nameserver. Diese kennen die IP-Adressen der Nameserver, welche für Top-Level-Domains (TLDs) wie *.com* oder *.de* zuständig sind. Jeder TLD-Nameserver verwaltet wiederum IP-Adressen, mit denen Nameserver der untergeordneten Second-Level-Domains (SLDs) erreicht werden können. Ein SLD-Nameserver gibt Auskunft darüber, über welche IP-Adresse eine Second-Level-Domain wie *elektronik-kompendium* aufrufbar ist. Es können optional noch nachgelagerte Nameserver existieren, die für die Namensauflösung von Subdomains auskunftsberechtigt sind [9].

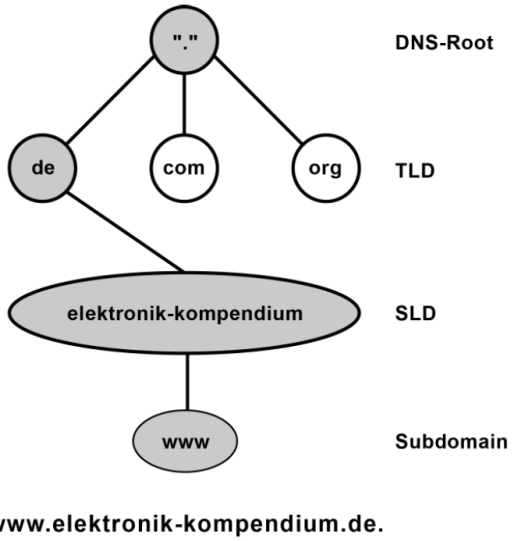

Abbildung 2. Nameserver-Hierarchie [9]

Die dezentrale Struktur des Domain Name Systems bietet einerseits eine erhöhte Sicherheit gegen Ausfälle und eine effiziente Lastenverteilung, während sie andererseits den Administrationsaufwand aufteilt. Wenn die Nameserver einer bestimmten Second-Level-Domain ausfallen sollten, betrifft das nur diese Domain und wirkt sich nicht auf das gesamte Domain Name System aus [10].

A. Namensauflösung

Dieser Abschnitt zeigt am Beispiel der Domain *www.elektronik-kompendium.de.* die Schritte auf, die nach Schnabel [9] bei der Namensauflösung durchlaufen werden. Es ist zu erwähnen, dass die Analyse der Domain von rechts nach links erfolgt. Die Domain endet außerdem formal mit einem Punkt auf der rechten Seite, der die Root-Ebene symbolisiert. Dieser Punkt wird jedoch meist weggelassen, da er keine praktische Funktion erfüllt [9].

Möchte ein Client die Domain *www.elektronik-kompendium.de* im Webbrowser aufrufen, ist nicht in jedem Fall eine Namensauflösung über das DNS notwendig. Moderne Browser sind in der Lage, die IP-Adresse für eine in der Vergangenheit aufgerufene Domain in ihrem Cache zu speichern [11]. Ist die IP-Adresse für die Domain bereits hinterlegt, kann diese für eine Verbindung mit dem Webserver verwendet werden. Befindet sich die IP-Adresse nicht im Cache des Browsers, stellt der Client eine Anfrage an den lokalen DNS-Resolver (siehe Abbildung 3).

Bei einem DNS-Resolver handelt es sich um einen nicht-autoritativen Nameserver. Das bedeutet, dass der Nameserver für eine Domain nicht selbst verantwortlich ist und die Informationen deshalb von nachgelagerten Instanzen einholen muss

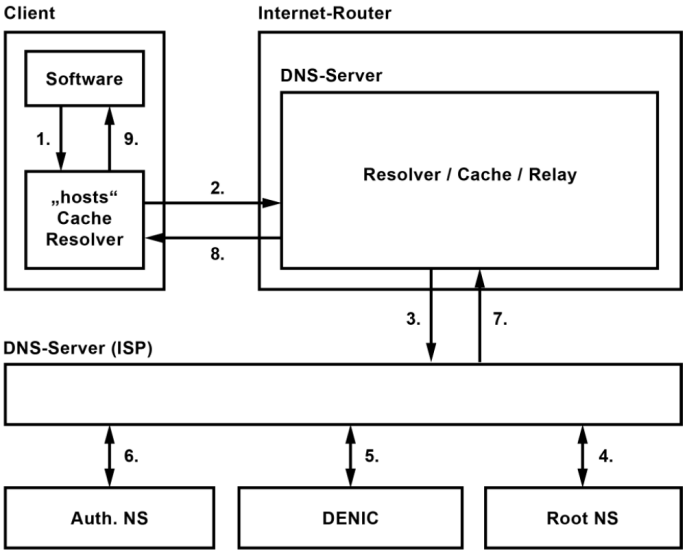

Abbildung 3. Prozess der Namensauflösung [9]

[9]. Der lokale DNS-Resolver ist Teil des Betriebssystems [11] und überprüft, ob in seinem Cache eine IP-Adresse für die Domain hinterlegt ist. Wird der lokale DNS-Resolver in seinem Cache nicht fündig, befragt er den DNS-Resolver im lokalen Netzwerk. In der Regel handelt es sich dabei um den Router [9]. Der Router überprüft zunächst ebenfalls seinen Cache, bevor er den DNS-Resolver beim Internet Service Provider (ISP) anfragt. Ist bei dem DNS-Resolver des ISP wie in den Fällen zuvor kein Eintrag im Cache vorhanden, schickt dieser eine Anfrage an den Root-Nameserver. Der Root-Nameserver weiß, dass für die *.de* Domain der Betreiber DENIC zuständig ist [12] und teilt dem DNS-Resolver dessen IP-Adresse mit. Der DNS-Resolver befragt DENIC, welcher als Antwort die IP-Adresse des autoritativen Nameservers liefert. Der autoritative Nameserver ist für die Domain *elektronik-kompendium.de* verantwortlich und verwaltet dessen Informationen. Wenn der DNS-Resolver den autoritativen Nameserver anfragt, bekommt er daher die IP-Adresse für den Webserver mitgeteilt.

Der DNS-Resolver des ISP übergibt die ermittelte IP-Adresse an den DNS-Resolver des Routers, welcher diese an den lokalen DNS-Resolver weitergibt. Am Ende landet die IP-Adresse bei der anfragenden Software. Wie bereits erwähnt, wird die ermittelte IP-Adresse in den Caches der DNS-Resolver gespeichert. Nach dem Ablauf einer vom autoritativen Nameserver vorgegebenen Time to Live (TTL) wird der Eintrag im Cache wieder gelöscht [10].

B. Nachrichtenformat

Bei der Kommunikation über das DNS-Protokoll werden Queries (Anfragen) und Responses (Antworten) in dem gleichen Nachrichtenformat übermittelt [13]. Dieses Format ist, wie in Abbildung 4 ersichtlich, in fünf Hauptabschnitte untergliedert.

Der *Header* existiert in jeder Art von Nachricht. Er beinhaltet Felder, die u.a. angeben, welche der nachfolgenden

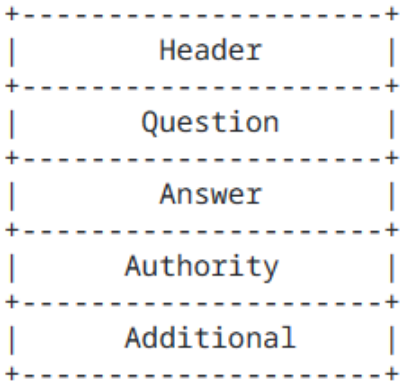

Abbildung 4. DNS-Nachrichtenformat [13]

Abschnitte vorhanden sind und ob die Nachricht eine Query oder eine Response darstellt. Der Abschnitt *Question* enthält eine Liste von Fragen, die dem Nameserver gestellt werden. Dies können beispielsweise Fragen nach der IP-Adresse einer Domain sein, wie etwa für *www.elektronik-kompendium.de*. Die letzten drei Abschnitte – *Answer*, *Authority* und *Additional* – sind ausschließlich in Responses enthalten und weisen das gleiche Format auf: eine Liste von Resource Records (RRs). Ein Resource Record ist ein vom Nameserver verwalteter Eintrag, der Informationen zu einer Domain enthält. Im Abschnitt *Answer* befinden sich RRs, welche die im Abschnitt *Question* gestellten Fragen beantworten. Der Abschnitt *Authority* enthält RRs, die auf einen autoritativen Nameserver verweisen. Sie sind daher typisch für Responses von TLD-Nameservern. Der Abschnitt *Additional* umfasst RRs, die in Bezug zur Query stehen, aber nicht direkt als Antworten auf die gestellten Fragen dienen [13].

Um den Aufbau einer Nachricht anhand eines Beispiels zu veranschaulichen, wurde im Rahmen dieses Artikels der DNS-Traffic für die Domain *www.elektronik-kompendium.de* mit dem Netzwerk-Analyseprogramm Wireshark aufgezeichnet. Die Abbildung 5 zeigt die Response, die der DNS-Resolver von einem Nameserver erhalten hat.

Die nachfolgende Interpretation der Felder basiert auf der Spezifikation RFC 1035 [13], die den Aufbau des DNS-Protokolls definiert. Zum Abschnitt *Header* gehören die Felder von *Transaction ID* bis *Additional RRs*. Die *Transaction ID* dient dazu, das eine Response einer Query zugeordnet werden kann. *Flags* spezifizieren, um was für eine Art von DNS-Nachricht es sich handelt. Der hexadezimale Wert *8180* signalisiert in diesem Fall, dass die Nachricht eine Response ohne Fehler ist. Die letzten vier Felder des Headers geben die Anzahl der Einträge in den Abschnitten *Question*, *Answer*, *Authority* und *Additional* an. Im dargestellten Beispiel beinhalten die Abschnitte *Question* und *Answer* eine Frage bzw. einen RR. Die Abschnitte *Authority* und *Additional* sind hingegen ohne Einträge.

Im Abschnitt *Question*, der in der Abbildung 5 unter *Queries* zu finden ist, befindet sich wie bereits erwähnt eine einzige Frage. Der in der Frage aufgeführte *Name* spezifiziert

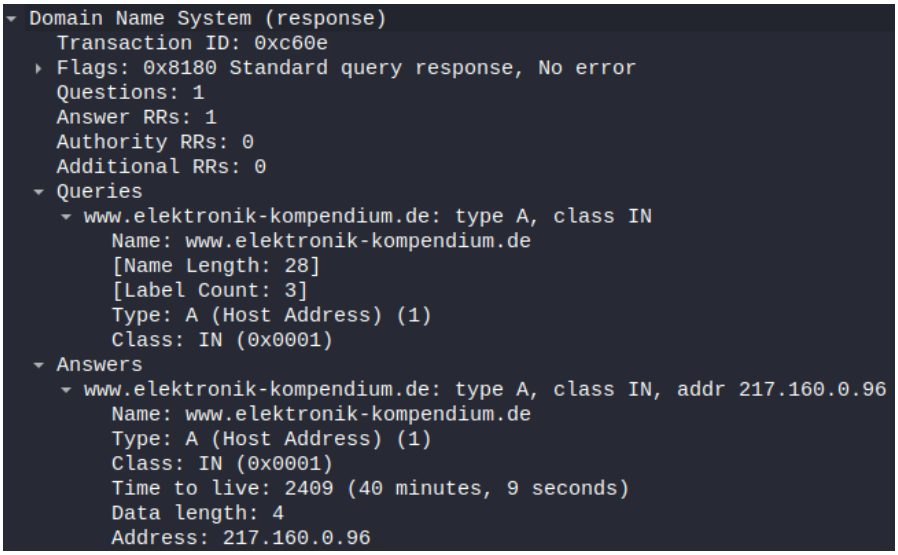

Abbildung 5. DNS-Response [Eigene Darstellung]

die Domain, für die Informationen eingeholt werden. Das Feld *Type* definiert die Art der gewünschten Information. Hierbei steht *A* für die IPv4-Adresse der Domain. Andere Typen sind beispielsweise *AAAA* oder *CNAME*. Während bei dem Typ *AAAA* die IPv6-Adresse der Domain abgefragt wird, lässt sich über *CNAME* der kanonische Name, also der echte Domainname, für einen Alias ermitteln. Das Feld *Class* klassifiziert die Frage, wobei *IN* für das Internet steht. Aus der in der Abbildung dargestellten Frage lässt sich folglich ableiten, dass der DNS-Resolver die IPv4-Adresse für die Domain *www.elektronik-kompendium.de* von einem Nameserver anfordert.

Im Answer-Abschnitt wird ein RR dargestellt, der sich aus sechs Feldern zusammensetzt. Die ersten drei Felder – *Name*, *Type* und *Class* – haben dieselbe Bedeutung wie die gleichnamigen Felder in einer Frage. Das Feld *Time to Live* definiert die Dauer in Sekunden, wie lange der RR in einem Cache gespeichert bleiben darf. *Address* zeigt die IPv4-Adresse an, die von dem DNS-Resolver angefragt wurde und über die auf *www.elektronik-kompendium.de* zugegriffen werden kann. *Data Length* gibt die Größe dieser Adresse in Bytes an. Da IPv4-Adressen eine Länge von 32 Bit aufweisen [14], sind in diesem Fall 4 Bytes angegeben.

Gemäß der Spezifikation RFC 1035 [13] erfolgt die Übermittlung von DNS-Nachrichten in der Regel über das UDP-Protokoll auf Port 53. Die Größe der Nachrichten, die über UDP gesendet werden können, ist allerdings auf 512 Bytes beschränkt. Sollte die Response eines Nameservers diese Größe überschreiten, werden deshalb nur die ersten 512 Bytes der Nachricht übertragen und im *Header* das Flag *Truncation (TC)* gesetzt. Eine Response mit einem solchen Flag weist auf eine abgeschnittene Nachricht hin und veranlasst den DNS-Resolver, eine weitere Query über das TCP-Protokoll auf Port 53 zu senden.

III. DNS-Tunneling

Wang et al. definieren in ihrer Arbeit „A comprehensive survey on DNS tunnel detection“ [7] einen DNS-Tunnel wie folgt:

> „The DNS tunnel is a type of tunnel technique, based on the DNS protocol, which utilises the DNS query process and encapsulates the data in the DNS query/response package to build a proprietary tunnel for transmission communication between the sender and receiver.“

Zusammenfassend ist unter einem DNS-Tunnel eine Methode zu verstehen, die sich das Domain Name System zunutze macht, um einen Datenübertragungskanal zwischen einem Sender und einem Empfänger zu etablieren.

DNS-Tunnel werden häufig von Angreifern für die Datenübertragung verwendet (siehe Abschnitt I). Dies liegt hauptsächlich daran, dass der Port 53 aufgrund der Wichtigkeit des Domain Name Systems meistens offen ist. Selbst wenn eine Firewall den Port blockiert, kann DNS-Tunneling funktionieren. Salat et al. [3] demonstrieren dies anhand eines kompromittierten Webservers in der Google Cloud. In diesem Fall ist DNS-Tunneling möglich, da die Google Cloud neben dem Webserver automatisch einen lokalen Metadatenserver startet. Dieser Server ist für essenzielle Dienste wie das DNS zuständig und wird durch die Firewall-Konfiguration nicht beeinflusst. Das DNS ist außerdem nicht für die Datenübertragung konzipiert. Administratoren schenken dem Protokoll deshalb weniger Aufmerksamkeit als bspw. dem File Transfer Protocol (FTP) [3]. Entscheiden sich Administratoren für eine Überwachung des DNS-Traffics, gestaltet sich die Erkennung von DNS-Tunneling aufgrund der enormen Datenmenge zudem als schwierig. So werden z.B. in einer von Ahmed et al. [15] untersuchten Forschungseinrichtung bis zu 400 DNS-Queries pro Sekunde gesendet.

A. Funktionsweise

Für die Realisierung eines DNS-Tunnels ist die Verwendung einer DNS-Tunneling-Software notwendig, die im Client/Server-Modus operiert [16]. Dabei befindet sich die Client-Komponente der Software auf einem kompromittierten System, während die Server-Komponente auf einem vom Angreifer kontrollierten autoritativen Nameserver installiert ist. Die Kommunikation zwischen dem kompromittierten System und dem autoritativen Nameserver entspricht grundsätzlich dem in der Abbildung 6 dargestellten Prinzip. In dem gezeigten Beispiel kontrolliert der Angreifer eine Domain mit dem Namen *tunnel.com*, die beim zuständigen TLD-Nameserver registriert wurde. Als autoritativer Nameserver, der Auskunft über die Domain geben kann, ist die IP-Adresse *A.B.C.D* angegeben. Der TLD-Nameserver wird auf diese IP-Adresse verweisen, falls er eine DNS-Query für die Domain *tunnel.com* erhält. Der Aufbau des DNS-Tunnels zwischen dem Client und dem Server gestaltet sich nach Wang et al. [7] wie folgt:

1) Die DNS-Tunneling-Software auf der Clientseite versteckt die zu übertragenden Daten in dem angefragten Domainnamen. Als Beispiel dient hier *updata.tunnel.com*, wobei *updata* die zu übertragenen Daten repräsentiert. Diese Daten werden anschließend an den lokalen DNS-Resolver übermittelt.
2) Aufgrund seiner Caching-Funktion kennt der lokale DNS-Resolver i.d.R. bereits den für *.com* zuständigen TLD-Nameserver. Er fordert von diesem Nameserver die Informationen für die Domain *tunnel.com* an.
3) Der TLD-Nameserver teilt dem DNS-Resolver mit, dass der für die Domain *tunnel.com* zuständige autoritative Nameserver über die IP-Adresse *A.B.C.D* erreicht werden kann.
4) Der lokale DNS-Resolver sendet eine DNS-Query für *updata.tunnel.com* an die IP-Adresse *A.B.C.D*. Die im autoritativen Nameserver installierte DNS-Tunneling-Software empfängt die Query und extrahiert die im Domainnamen versteckten Daten.
5) Die DNS-Tunneling-Software im autoritativen Nameserver versteckt Anweisungen in der DNS-Response und sendet diese an den DNS-Resolver zurück. Diese Anweisungen sind in der Abbildung als *downdata* gekennzeichnet.
6) Der DNS-Resolver leitet die Response, welche die *downdata* beinhaltet, weiter an den Client. Die DNS-Tunneling-Software im Client extrahiert die Anweisungen und führt darauf basierend die Übertragung neuer Daten durch.

In dem dargestellten Beispiel agiert der autoritative Nameserver als bösartiger Command-and-Control (C2) Server. Er empfängt die Daten, verarbeitet sie und sendet bei Bedarf Anweisungen an das kompromittierte System [7]. DNS-Tunneling kann jedoch auch für legitime Zwecke eingesetzt werden. McAfee bietet beispielsweise ein Reputationssystem an, das auf DNS-Tunneling basiert [17]. Wenn die McAfee-Software eine verdächtige Datei identifiziert, sendet sie Informationen wie den Hash der Datei über eine DNS-Query an einen File Reputation Server. Dieser Server ermittelt die Reputation, also die Vertrauenswürdigkeit der Datei, und hinterlegt das Ergebnis in Form einer Punktzahl in der DNS-Response. Die McAfee-Software empfängt die Response und bewertet auf Basis der Punktzahl, ob der Datei vertraut werden kann oder nicht.

B. Datenkodierung

Es existieren verschiedene Methoden, um Daten in einer DNS-Query unterzubringen. Eine Methode verwendet bspw. den ungenutzten Bereich eines Datenpakets, der normalerweise nicht für die Übertragung von Informationen vorgesehen ist [7]. Diese Methode ist leicht erkennbar, weshalb ihr in der Forschung weniger Beachtung geschenkt wird. Nach Wang et al. [7] liegt der Fokus der meisten Forscher auf dem DNS-Tunneling, das Domainnamen zur Datenübertragung nutzt (siehe Abschnitt III-A). Bei der Verwendung dieser Methode ist zu beachten, dass der in einer DNS-Query enthaltene Domainname bestimmte Vorgaben erfüllen muss [16]. Ein Domainname ist in sogenannte Labels unterteilt, die durch Punkte voneinander getrennt sind. Die einzelnen Labels dürfen jeweils höchstens 63 Zeichen aufweisen, wobei die Gesamtlänge des Domainnamens auf 255 Zeichen begrenzt ist. Die Zeichen in einem Label sind außerdem auf Buchstaben, Zahlen oder Bindestriche beschränkt. Es muss folglich ein Kodierungsverfahren verwendet werden, das die zu übertragenen Daten in dieses Format überführt.

Base32 ist das am häufigsten für DNS-Queries verwendete Kodierungsverfahren [16]. Laut der Spezifikation RFC 4648 [18] wird dabei eine Gruppe von 40 Eingabebits zuerst in acht 5-Bit-Blöcke aufgeteilt. Jeder dieser Blöcke wird dann in ein Zeichen des Base32-Alphabets umgewandelt, das aus Buchstaben (case-insensitive) und den Zahlen 2 bis 7 besteht. Da die Länge der kodierten Daten nicht immer ein Vielfaches von 40 Bits beträgt, wird das Gleichheitszeichen als Padding eingesetzt. Wie oben beschrieben, ist dieses Zeichen in Domainnamen allerdings nicht zulässig. DNS-Tunneling-Programme wie Iodine [19] implementieren daher ihre eine eigene Base32-Kodierung mit einer von der Spezifikation abweichenden Zeichentabelle.

In einer DNS-Response werden die zu übertragenden Daten in einem Resource Record platziert [7]. Der verwendete Typ des Resource Records variiert je nach eingesetztem DNS-Tunneling-Programm und kann teilweise vom Anwender selbst ausgewählt werden. Programme wie Iodine [19] und DNSCat2 [20] unterstützen unter anderem *NULL*, *TXT*, *MX*, *CNAME* und *A*, die hinsichtlich ihrer Bandbreite in absteigender Reihenfolge sortiert sind [13]. Für das DNS-Tunneling werden grundsätzlich Typen mit größerer Bandbreite bevorzugt, um die Daten in weniger Paketen zu übermitteln [7].

Bei dem für DNS-Responses eingesetzten Kodierungsverfahren handelt es sich in den meisten Fällen um Base64 [16]. Dieses Verfahren teilt 24 Eingabebits in vier 6-Bit-Blöcke auf, wobei jeder Block einem Zeichen aus einem 64-Zeichen-Alphabet zugeordnet wird. Dieses Alphabet umfasst Groß- und

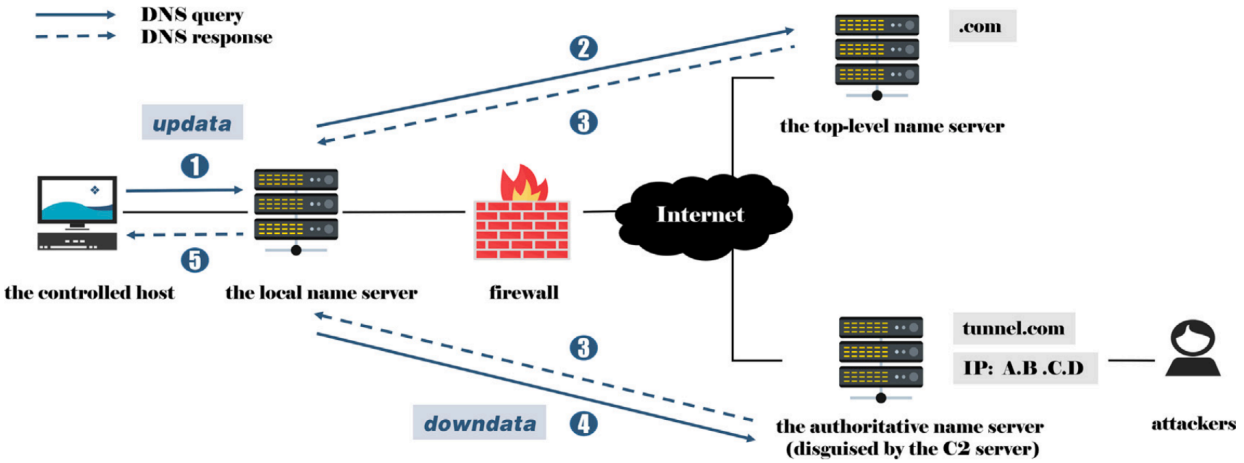

Abbildung 6. Funktionsweise eines DNS-Tunnels [7]

Kleinbuchstaben, Zahlen von 0 bis 9 sowie die Symbole + und /. Um auf ein Vielfaches von 24 Bits zu kommen, wird wie bei Base32 das Gleichheitszeichen als Padding verwendet [18].

Es ist zu beachten, dass für Typen wie CNAME und A die Base64-Kodierung nicht verwendet werden kann. Bei dem Typ *CNAME* liegt der Grund darin, dass das Base64-Alphabet 64 Zeichen umfasst, wohingegen für gültige Domainnamen nur 37 Zeichen erlaubt sind [16]. Folglich lässt sich für Base64 keine Zeichentabelle definieren, die ausschließlich aus für Domainnamen zulässigen Zeichen besteht. Bei dem Typ *A* sind die Kodierungsbeschränkungen besonders strikt. Es muss sich um eine IPv4-Adresse handeln [14], wodurch sowohl Base64- als auch Base32-Zeichensätze nicht anwendbar sind.

C. Beispiel

In diesem Artikel wurde die Packet Capture (PCAP) Datei eines von BSidesSF organisierten Capture-the-Flag Wettbewerbs [21] analysiert. Eine in der Datei enthaltene DNS-Response ist in Abbildung 7 dargestellt. Laut der Lösungsdokumentation wurde sie von dem Programm DNSCat2 erstellt und für die Übertragung von Daten verwendet. Der von dem DNS-Resolver angefragte Domainname wird aus Platzgründen nicht vollständig angezeigt. Er setzt sich aus folgenden Teilen zusammen: *encoded-data.encoded-data.encoded-data.encoded-data.c2.challenges.bsidessf.net*. Die ersten drei *encoded-data* Teile bestehen aus 60 und der letzte *encoded-data* Teil besteht aus 10 hexadezimal kodierten Zeichen. Der gesamte Domainname umfasst 220 Zeichen. Damit schöpft DNSCat2 die Formatgrenzen eines Domainnamens fast vollständig aus. Wie in Abschnitt III-B erläutert, darf ein einzelnes Label eines Domainnamens maximal 63 Zeichen und der gesamte Domainname maximal 255 Zeichen lang sein.

Für die Übertragung von Anweisungen vom autoritativen Nameserver zum kompromittierten System wurde in der DNS-Response ein *MX* Record verwendet. Dieser Typ gibt den Domainnamen des Mailservers an [22] und enthält die zwei Felder *Preference* und *Mail Exchange* [13]. Das Feld *Preference* spezifiziert die Priorität des Mailservers. Die Priorität ist wichtig, wenn der autoritative Nameserver mehrere *MX* Records mit verschiedenen Mailservern zurücksendet. Der DNS-Resolver wählt in diesem Fall den Mailserver mit der niedrigsten *Preference* für den E-Mail-Versand aus. Das Feld *Mail Exchange* beinhaltet den Domainnamen des Mailservers und unterliegt denselben Formatgrenzen wie der Domainname in einer DNS-Query [13]. In der untersuchten DNS-Response wird dieses Feld genutzt, um Anweisungen zu übertragen. Wie bei den DNS-Queries verwendet DNSCat2 auch bei den DNS-Responses eine hexadezimale Kodierung der Daten [20].

IV. DoH-Tunneling

DNS-Nachrichten werden normalerweise unverschlüsselt über Port 53 gesendet. Dies bietet einige Vorteile wie einen geringen Rechenaufwand und eine schnelle Auflösung der Domainnamen. Das Senden unverschlüsselter Daten stellt allerdings ein Sicherheitsrisiko dar, da DNS-Queries sensible Benutzerinformationen preisgeben können [23]. Die Internet Engineering Task Force (IETF) hat deshalb eine Reihe von Standards vorgeschlagen, die sichere Protokolle für die Namensauflösung verwenden. Einer dieser Standards nutzt HTTPS, das DNS-Traffic mittels des Transport Layer Security (TLS) Protokolls verschlüsselt. Hierbei erfolgt die Kommunikation über TCP auf Port 443. Dieser Ansatz ist als DNS over HTTPS [5] bekannt und stellt die beliebteste Methode zur Verschlüsselung von DNS-Traffic dar [23]. Obwohl DoH derzeit noch nicht so weit verbreitet ist wie das traditionelle DNS-Protokoll, nimmt seine Nutzung zu [24]. Cloud Service Provider wie Google und Alibaba haben bereits DNS-Resolver eingeführt, die DoH unterstützen [25], [26]. DoH wurde zudem im Jahr 2020 in Webbrowser wie Google Chrome, Microsoft Edge und Firefox integriert [23].

Obwohl die Verschlüsselung von DNS-Nachrichten ihre Vorteile hat, bringt sie nach Moure-Garrido et al. [4] auch Sicherheitsprobleme mit sich. Ein Problem ist das Stehlen von Daten über DoH-Tunneling. Informationen wie der Domainname oder der Record-Typ bleiben bei DoH verborgen (siehe Tabelle I). Die herkömmlichen Methoden zur Erkennung von

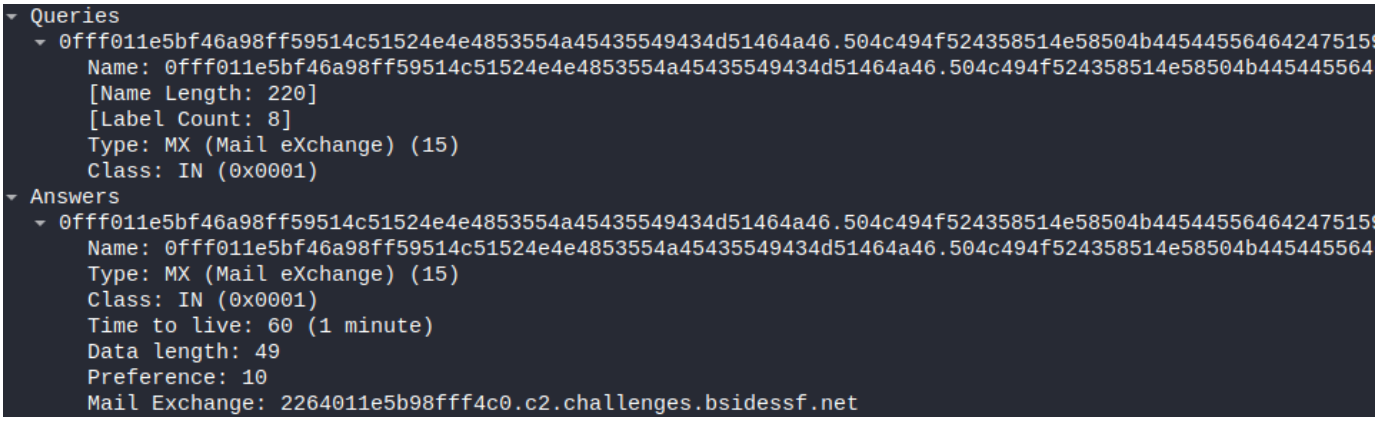

Abbildung 7. DNS-Tunneling-Response [Eigene Darstellung]

DNS-Tunneling sind jedoch auf diese Informationen angewiesen und deshalb nicht mehr effektiv. Um dieses Problem anzugehen, wurden neue Methoden zur Erkennung von DoH-Tunneling entwickelt. Sie nutzen einige der noch sichtbaren Eigenschaften wie die Paketgröße oder den Zeitstempel [4].

Tabelle I
SICHTBARE INFORMATIONEN IM DNS- UND DOH-TRAFFIC [4]

Information	DNS	DoH
Record-Typ	✓	×
Paketgröße	✓	✓
Domainname	✓	×
Zeitspanne zwischen DNS-Queries	✓	✓
Traffic-Volumen	✓	✓

V. PAYLOAD-BASIERTE ERKENNUNGSMETHODEN

Bei der Payload handelt es sich um die in einem Datenpaket enthaltenen Nutzdaten. Steuer- und Protokollinformationen, die sich im Header des Datenpakets befinden, gehören nicht dazu [9]. In diesem Abschnitt wird beschrieben, wie DNS- und DoH-Tunneling anhand von Eigenschaften der Payload erkannt werden können.

A. Ungewöhnliche Record-Typen

In der normalen DNS-Kommunikation treten verschiedene Typen von Resource Records mit unterschiedlichen Häufigkeiten auf. Tabelle 2 veranschaulicht die Verteilung der Record-Typen anhand von Daten, die von einem ISP aus Luxemburg stammen. Diese Daten umfassen zehn Millionen DNS-Responses, die in einem Zeitraum von einer Stunde aufgezeichnet wurden [27]. Die häufigsten Typen sind *A* und *PTR*, gefolgt von *MX*, *AAAA* und *TXT*.

Wie im Abschnitt III-B erläutert, werden für das DNS-Tunneling grundsätzlich Typen mit größerer Bandbreite wie *NULL*, *TXT*, *MX* oder *CNAME* bevorzugt. In den von dem ISP bereitgestellten DNS-Responses machten *TXT* Records 4,26 Prozent aus, während *CNAME* Records gänzlich fehlten. *NULL* Records wurden in der Häufigkeitsverteilung nicht aufgeführt, was auf einen sehr niedrigen oder nicht vorhandenen Anteil im DNS-Traffic hindeutet. Ein signifikanter Anteil an solchen Records ist aus diesem Grund ein starker Indikator für DNS-Tunneling [7]. Obwohl *MX* Records gelegentlich für DNS-Tunneling verwendet werden, ist die Identifizierung aufgrund einer Häufigkeit von 9,3 Prozent schwieriger. Das Tunneling mittels *A* Records ist zwar zeitaufwendiger, es wird jedoch kaum auffallen. Schließlich handelt es sich bei fast jeder zweiten DNS-Response um einen *A* Record. Die Erkennung von DoH-Tunneling ist mit dieser Methode nicht möglich, da der Record-Typ durch die Verschlüsselung verborgen wird (siehe Abschnitt IV).

Tabelle II
HÄUFIGKEITSVERTEILUNG DER RECORD-TYPEN [27]

Record-Typ	Häufigkeit in Prozent
A	49
PTR	27.28
MX	9.3
AAAA	8.5
TXT	4.26
NS	0.3
SOA	0.11
SRV	0.01
CNAME	0
Sonstige Typen	1.24

B. Ungewöhnlich große DNS-Queries

Die DNS-Queries, welche das DNS-Tunneling-Programm vom kompromittierten System sendet, enthalten typischerweise große Mengen an kodierten Daten [7]. Dies wird auch durch das im Abschnitt III-C dargestellte DNS-Tunneling-Beispiel deutlich: Der Domainname hat eine Größe von 220 Bytes bei einer möglichen Maximalgröße von 255 Bytes. Nach Wang et al. [7] ist eine ungewöhnlich große DNS-Query deshalb ein Indikator für DNS-Tunneling.

Moure-Garrido et al. [28] zeigen, dass sich diese Methode auch zur Identifikation von DoH-Tunneling anwenden lässt. Der von ihnen analysierte Datensatz umfasst sowohl gutartigen als auch bösartigen DoH-Traffic, wobei der gutartige Traffic durch das Surfen auf den Top 10.000 Alexa-Websites

generiert wurde. Der bösartige Traffic stammte von den DNS-Tunneling-Programmen Iodine, DNSCat2 und DNS2TCP. Wie die Analyseergebnisse in Abbildung 8 verdeutlichen, wurden größere Datenpakete überwiegend dem bösartigen DoH-Traffic zugeordnet. Vor allem bei Paketen über 220 Bytes trat kein gutartiger DoH-Traffic mehr auf.

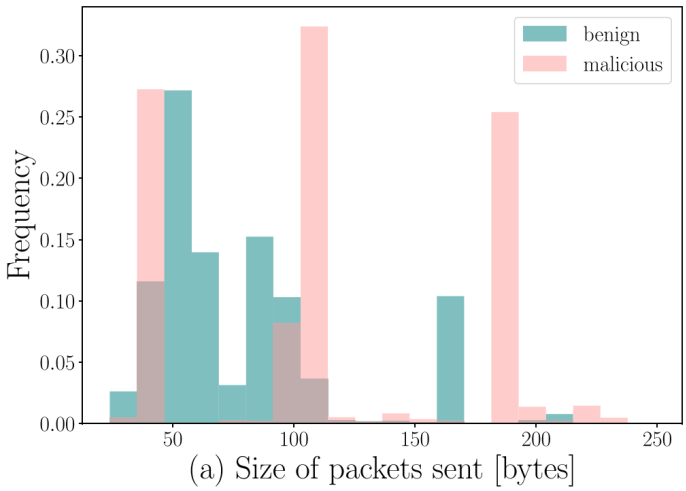

Abbildung 8. Häufigkeitsverteilung der DNS-Queries nach Größe, klassifiziert nach gut- und bösartig [28]

Der in Abschnitt III-C beispielhaft aufgezeigte DNS-Tunnel lässt sich durch diese Methode leicht identifizieren, da allein der Domainname bereits 220 Bytes umfasst. Eine nähere Untersuchung mittels Wireshark ergibt, dass die DNS-Query insgesamt 238 Bytes groß ist.

C. Domainnamen mit hoher Entropie

Legitime Domains haben normalerweise aussagekräftige und leicht zu merkende Namen, die dem Zipfschen Gesetz folgen [7]. Das Zipfsche Gesetz besagt, dass die Häufigkeit der Wörter in einem Text umgekehrt proportional zu ihrem Rang ist. Das zweithäufigste Wort wird demnach halb so oft verwendet wie das häufigste Wort, das dritthäufigste Wort nur ein Drittel so oft, und so weiter. Die für das DNS-Tunneling verwendeten Domainnamen sind aufgrund der enthaltenen kodierten Daten hingegen zufällig aufgebaut. Ein Domainname mit einer höheren „Zufälligkeit“ ist folglich ein Indikator für DNS-Tunneling [3]. Die Zufälligkeit von Informationen kann anhand der Entropie gemessen werden, die 1948 von Claude Shannon eingeführt wurde [29]. Die Shannon-Entropie H wird durch

$$H = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{1}$$

definiert, wobei p_i die Wahrscheinlichkeit eines Ereignisses i ist. Ein höherer Entropiewert weist auf eine höhere Zufälligkeit der Informationen hin.

Salat et al. [3] verwenden für die Berechnung der Entropie von Domainnamen die Suchmaschine Elasticsearch, welche ein Plugin für statistische Analysen bereitstellt. Bei Elasticsearch handelt es sich um ein auf GitHub verfügbares Open-Source-Projekt [30]. Im Rahmen dieses Artikels wurde der Quellcode analysiert, um die bei der Entropieberechnung angewandten mathematischen Methoden genauer zu verstehen. Die Analyse ergab, dass sich die Berechnung grundsätzlich in drei Schritte einteilen lässt. Im ersten Schritt wird gezählt, wie oft jedes Zeichen vorkommt. Bei der Domain *google.com* ergibt sich bspw. folgende Häufigkeitsverteilung: *{'g': 2, 'o': 3, 'l': 1, 'e': 1, '.': 1, 'c': 1, 'm': 1}*. Im darauffolgenden Schritt werden die Wahrscheinlichkeiten jedes Zeichens in Relation zur gesamten Zeichenanzahl bestimmt. Bei einem Domainnamen mit 10 Zeichen resultieren daraus die folgenden relativen Wahrscheinlichkeiten: *{'g': 0,2, 'o': 0,3, 'l': 0,1, 'e': 0,1, '.': 0,1, 'c': 0,1, 'm': 0,1}*. Abschließend wird im dritten Schritt die Entropie berechnet:

$$\begin{aligned} H = -\,(&0.2 \times \log_2(0.2) + 0.3 \times \log_2(0.3) + 0.1 \times \log_2(0.1) \\ &+ 0.1 \times \log_2(0.1) + \ 0.1 \times \log_2(0.1) + \ 0.1 \times \log_2(0.1) \\ &+ 0.1 \times \log_2(0.1)) \approx 2,64 \end{aligned} \tag{2}$$

Laut Salat et al. [3] weisen legitime Domainnamen i.d.R. einen Entropiewert von unter 4 auf. Im Gegensatz dazu überschreiten Domainnamen, die zum Transport kodierter Daten genutzt werden, häufig diese Grenze. Zur Überprüfung dieser Aussage wurde die Entropie für den 220 Zeichen langen Domainnamen berechnet, der im Abschnitt III-C als Beispiel für DNS-Tunneling dient. Mit der beschriebenen Berechnungsmethodik ergibt sich für diesen Domainnamen ein Entropiewert von $3,74$. Obwohl dieser Wert die von Salat et al. aufgestellte Grenze nicht überschreitet, liegt er trotzdem deutlich über dem Entropiewert der Domain *google.com*.

Die Bewertung eines Domainnamens anhand seiner Entropie kann zu falsch-positiven Ergebnissen führen. Dies tritt unter anderem bei Domainnamen-Generatoren auf, die von Cloud-Diensten genutzt werden [3]. Aus diesem Grund empfiehlt es sich, bei der Analyse von Domainnamen eine Kombination verschiedener Erkennungsmethoden zu verwenden. Es kann z.B. die Anzahl der Subdomains untersucht werden, da diese bei der Übertragung von Daten meist über dem Normalwert liegt. Ebenso kann das Verhältnis der verwendeten Zeichen ein wichtiger Indikator sein. Bei DNS-Tunneling existieren typischerweise mehr Ziffern und Großbuchstaben als bei einer legitimen DNS-Query [7].

Es ist zu beachten, dass bei DoH-Traffic der Domainname nicht sichtbar ist (siehe Abschnitt IV). Die beschriebenen Methoden eignen sich daher nur zur Erkennung von DNS-Tunneling.

VI. Traffic-basierte Erkennungsmethoden

Im Gegensatz zu Payload-basierten Erkennungsmethoden fokussieren sich die Traffic-basierten Methoden nicht auf ein einzelnes Datenpaket. Sie betrachten stattdessen eine Menge von Datenpaketen, die in einem bestimmten Zeitraum übertragen werden. Aufgrund dieser Charakteristik eignen sich die

Traffic-basierten Methoden nicht zur Erkennung von Tunneling in Echtzeit. Für solche Zwecke wird in der Praxis meistens auf die Payload-basierten Methoden zurückgegriffen [7].

A. Langer Zeitabstand zwischen Datenpaketen

Bei der Auflösung legitimer Domainnamen greifen DNS-Resolver häufig auf im Cache gespeicherte Resource Records zurück. In diesen Fällen ist der Zeitabstand zwischen einer DNS-Query und der dazugehörigen DNS-Response relativ kurz. Eine DNS-Query, die zu übertragende kodierte Daten enthält, wird hingegen nicht im Cache des DNS-Resolvers existieren. Eine solche Query muss jedes Mal an den autoritativen Nameserver gesendet werden, was zu einem längeren Zeitabstand zwischen Query und Response führt. Längere Zeitabstände können auch bei legitimen Domainnamen vorkommen. Allerdings tritt dies nur auf, wenn der entsprechende Resource Record nicht im Cache vorliegt. Ein DNS-Tunnel wird bei einer Messung über einen bestimmten Zeitraum deshalb im Durchschnitt einen längeren Zeitabstand zwischen den Queries und den Responses aufweisen [7].

Moure-Garrido et al. [28] haben durch die Analyse des im Abschnitt V-B beschriebenen Datensatzes herausgefunden, dass der Zeitabstand auch zur Erkennung von DoH-Tunneling eingesetzt werden kann. Es handelt sich dabei allerdings nicht um den Zeitabstand zwischen den Queries und den dazugehörigen Responses. Die *Transaction ID* in einer DNS-Nachricht ist im DoH-Traffic schließlich verschlüsselt und kann nicht für die Zuordnung der Queries und Responses verwendet werden. Die Autoren betrachten stattdessen den Zeitabstand zwischen zwei aufeinander folgenden Paketen, unabhängig von deren Richtung. Es kann sich also um zwei gesendete oder zwei empfangene Pakete handeln. Eine Messung des Zeitabstands ist möglich, da sich in dem DoH-Traffic die Zeitstempel der Pakete auslesen lassen (siehe Abschnitt IV).

Nach den Analyseergebnissen von Moure-Garrido et al. [28] weist ein längerer Zeitabstand zwischen zwei aufeinander folgenden Paketen auf DoH-Tunneling hin. Die Ursache dafür ist, dass Tunneling-Programme die DNS-Nachrichten in einer geringeren Frequenz senden als es bei legitimen DoH-Traffic üblich ist. Zur Vermeidung von falsch-positiven Ergebnissen ist es sinnvoll, den Zeitabstand in Kombination mit der Größe der Pakete zu betrachten (siehe Abbildung 9). Bei einem Zeitabstand von mehr als 0,5 Sekunden beträgt die Wahrscheinlichkeit für bösartigen DoH-Traffic 65 Prozent. Die Wahrscheinlichkeit, dass sich gutartiger Traffic über diesen Schwellenwert befindet, liegt allerdings noch bei 10 Prozent. Mource-Garrido et al. haben bei der Entwicklung ihres Tunneling-Erkennungssystems deshalb zusätzlich für die Paketgröße einen Schwellenwert von 175 Bytes festgelegt. Ab einem Zeitabstand von 0,5 Sekunden und einer Paketgröße von 175 Bytes sinkt die Wahrscheinlichkeit auf unter 1 Prozent, legitimen Traffic irrtümlich als bösartig zu klassifizieren [28].

B. Hohes Traffic-Volumen

Bei DNS-Tunneling werden häufig große Datenmengen in einer einzelnen DNS-Query untergebracht, um die Effizienz der Datenübertragung zu erhöhen [7]. Die Untersuchung des DNS-Traffics auf ungewöhnlich große DNS-Queries ist daher ein effektives Mittel zur Erkennung von DNS-Tunneling (siehe Abschnitt V-B). Ein Angreifer kann jedoch als Gegenmaßnahme die Datenmenge pro DNS-Query verringern und die Anzahl der DNS-Queries erhöhen [7]. In diesem Fall ist es mit dieser Methode nicht mehr möglich, bösartigen DNS-Traffic zu identifizieren. Die Methode, die DNS-Traffic auf ungewöhnliche Record-Typen untersucht (siehe Abschnitt V-A), wird ebenfalls umgangen. Der Angreifer muss schließlich nicht mehr auf unübliche Record-Typen mit größerer Bandbreite zurückgreifen und kann stattdessen häufig verwendete Typen wie den *A* Record verwenden.

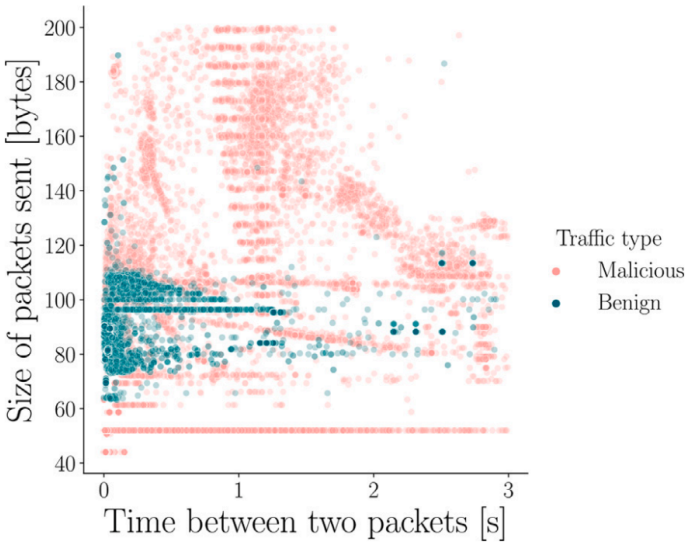

Abbildung 9. Korrelation zwischen der Paketgröße und dem Zeitabstand zwischen zwei aufeinander folgenden Paketen, klassifiziert nach gut- und bösartig [28]

Um dieses Szenario in der DNS-Tunneling-Erkennung abzudecken, sollte das Gesamtvolumen des DNS-Traffics beobachtet werden. Da ein Angreifer für die Übertragung von Daten sehr viele DNS-Queries sendet, deutet ein signifikanter Anstieg des Traffic-Gesamtvolumens auf DNS-Tunneling hin. Diese Annahme gilt allerdings nur für Netzwerke, die ein niedriges Aufkommen an DNS-Queries aufweisen. In Netzwerken, die ein hohes Aufkommen haben, kann diese Methode zu falsch-positiven Ergebnissen führen [3].

Wang et al. [7] stellen zwei alternative Methoden vor, die sich auf bestimmte Aspekte des Traffic-Volumens fokussieren. Bei der ersten Methode wird das Traffic-Volumen einzelner Quelladressen beobachtet. Der DNS-Traffic eines kompromittierten Systems übersteigt schließlich oft den von anderen Systemen im Netzwerk. Eine mögliche Gegenmaßnahme eines Angreifers ist IP-Spoofing, bei dem Datenpakete mit einer gefälschten Quelladresse gesendet werden [31]. Die zweite Methode erkennt DNS-Tunneling an einem hohen Traffic-Volumen, das mit einer bestimmten Domain in Verbindung steht. Auch hier kann der Angreifer durch die Verwendung

mehrerer Domains das Volumen verteilen und die Entdeckung erschweren [7].

Im Abschnitt IV wurde belegt, dass das Volumen zu den im DoH-Traffic sichtbaren Informationen gehört. Die in diesem Abschnitt geschilderten Methoden sind deshalb auch für die Erkennung von DoH-Tunneling geeignet.

VII. Fazit

Das Domain Name System hat die Aufgabe, Domainnamen in IP-Adressen aufzulösen. Die Kommunikation mit den Nameservern erfolgt in Klartext, was unter anderem Datenschutzprobleme mit sich bringt. Gleichzeitig ergibt sich daraus ein wesentlicher Vorteil: Die Datenübertragung über das DNS, auch DNS-Tunneling genannt, lässt sich auf vielfältige Weise erkennen. Es können nicht nur Methoden eingesetzt werden, die über einen bestimmten Zeitraum bspw. das Volumen oder den Zeitabstand zwischen Datenpaketen beobachten. Es ist auch möglich, die Felder eines einzelnen DNS-Pakets wie den Domainnamen oder den Record-Typen zu analysieren.

Ein neuer auf dem Domain Name System basierender Standard ist DNS over HTTPS. Dieser Standard verschlüsselt die Kommunikation mit den Nameservern und behebt dadurch die Datenschutzprobleme. Die Erkennung von DoH-Tunneling, also der Übertragung von Daten über DNS over HTTPS, wird durch die Verschlüsselung jedoch erschwert. Das liegt hauptsächlich daran, dass die Felder eines DNS-Pakets nicht mehr einsehbar sind und deshalb weniger Methoden zur Erkennung von Tunneling zur Verfügung stehen. Trotz dieses Problems ist es mit drei der fünf untersuchten Methoden möglich, DoH-Tunneling aufzudecken.

Sowohl bei DNS- als auch bei DoH-Tunneling können Angreifer Maßnahmen ergreifen, um die Erkennung zu vermeiden. Es ist bspw. möglich, auf Record-Typen mit hoher Bandbreite zu verzichten und stattdessen einen *A* Record zu nutzen. Obwohl dieser für die Datenübertragung ineffizient ist, wird die Analyse auf ungewöhnliche Record-Typen oder große Datenpakete keine Anzeichen für Tunneling aufzeigen. Die Übertragung einer Datei über den *A* Record führt allerdings dazu, dass das DNS-Tunneling-Programm eine sehr hohe Anzahl an Datenpaketen sendet. Durch die Analyse des Traffic-Volumens wird sich das Tunneling deshalb immer noch erkennen lassen. Für Unternehmen ist es aus diesem Grund sinnvoll, eine Kombination verschiedener Erkennungsmethoden einzusetzen.

Künftige Forschungsarbeiten können auf die in diesem Artikel gewonnenen Erkenntnisse aufbauen und zusätzliche Methoden zur Erkennung von DNS- und DoH-Tunneling untersuchen. Es wäre zudem interessant, ob sich die Methoden auch bei anderen sicheren DNS-Protokollen einsetzen lassen. Hierzu zählen Protokolle wie DNS over Transport Layer Security (DoT) und DNS over Quick UDP Internet Connections (DoQ).

Literatur

[1] IBM Corporation, *Cost of a Data Breach Report*, 2023.

[2] H. Edling, *Volkswirtschaftslehre - schnell erfasst*. Springer-Verlag, 2010.

[3] L. Salat, M. Davis, and N. Khan, "DNS Tunnelling, Exfiltration and Detection over Cloud Environments," *Sensors*, vol. 23, no. 5, p. 2760, 2023.

[4] Marta Moure-Garrido, Celeste Campo, and Carlos Garcia-Rubio, "Real time detection of malicious DoH traffic using statistical analysis," *Computer Networks*, vol. 23, 2023.

[5] Hoffman, Paul and McManus, Patrick, "RFC 8484: DNS Queries over HTTPS," 2018.

[6] ExtraHop Networks, Inc., *Taxonomy of SolarWinds Sunburst DNS Abuse Tactics*, 2021.

[7] Y. Wang, A. Zhou, S. Liao, R. Zheng, R. Hu, and L. Zhang, "A comprehensive survey on DNS tunnel detection," *Computer Networks*, vol. 197, p. 108322, 2021.

[8] D. Vekshin, K. Hynek, and T. Cejka, "DoH insight: Detecting DNS over HTTPS by machine learning," in *Proceedings of the 15th International Conference on Availability, Reliability and Security*, 2020, pp. 1–8.

[9] P. Schnabel, *Netzwerktechnik-Fibel: Grundlagen, Übertragungssysteme, TCP/IP, Dienste, Sicherheit*. Elektronik-Kompendium, 2022.

[10] W. Riggert and R. Lübben, *Rechnernetze: ein einführendes Lehrbuch*. Carl Hanser Verlag GmbH Co KG, 2022.

[11] A. Klein and B. Pinkas, "DNS Cache-Based User Tracking," in *NDSS*, 2019.

[12] DENIC eG, *Tätigkeitsbericht 2022: Zahlen, Daten, Fakten*, 2022.

[13] P. V. Mockapetris, "RFC 1035: Domain names: implementation and specification," 1987.

[14] J. Postel, "RFC 791: Internet protocol: DARPA Internet program protocol specification," 1981.

[15] J. Ahmed, H. H. Gharakheili, Q. Raza, C. Russell, and V. Sivaraman, "Monitoring enterprise DNS queries for detecting data exfiltration from internal hosts," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 265–279, 2019.

[16] S. Shafieian, D. Smith, and M. Zulkernine, "Detecting DNS tunneling using ensemble learning," in *Network and System Security: 11th International Conference, NSS 2017, Helsinki, Finland, August 21–23, 2017, Proceedings 11*, 2017, pp. 112–127.

[17] L. L. McAfee, *McAfee Global Threat Intelligence File Reputation Service*, 2012.

[18] S. Josefsson, "RFC 4648: The base16, base32, and base64 data encodings," 2006.

[19] E. Ekman, "Iodine." [Online]. Available: https://github.com/yarrick/iodine

[20] R. Bowes, "DNSCat2." [Online]. Available: https://github.com/iagox86/dnscat2

[21] BSides San Francisco, "CTF 2021 Release." [Online]. Available: https://github.com/BSidesSF/ctf-2021-release

[22] C. Partridge, "RFC 974: Mail Routing and the domain system," 1986.

[23] M. Lyu, H. H. Gharakheili, and V. Sivaraman, "A survey on DNS encryption: Current development, malware misuse, and inference techniques," *ACM Computing Surveys*, vol. 55, no. 8, pp. 1–28, 2022.

[24] S. Garcia, K. Hynek, D. Vekshin, T. Čejka, and A. Wasicek, "Large scale measurement on the adoption of encrypted DNS," *arXiv preprint arXiv:2107.04436*, 2021.

[25] Alibaba Cloud, "Alibaba Cloud Public DNS." [Online]. Available: https://www.alibabacloud.com/help/en/alibaba-cloud-public-dns

[26] Google LLC, "Google Public DNS." [Online]. Available: https://developers.google.com/speed/public-dns?hl=en

[27] S. Marchal, J. François, C. Wagner, R. State, A. Dulaunoy, T. Engel, and O. Festor, "DNSSM: A large scale passive DNS security monitoring framework," in *2012 IEEE Network Operations and Management Symposium*, 2012, pp. 988–993.

[28] M. Moure-Garrido, C. Campo, and C. Garcia-Rubio, "Real time detection of malicious DoH traffic using statistical analysis," *Computer Networks*, vol. 234, p. 109910, 2023.

[29] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE mobile computing and communications review*, vol. 5, no. 1, pp. 3–55, 2001.

[30] Elasticsearch B.V., "Elasticsearch." [Online]. Available: https://github.com/elastic/elasticsearch

[31] Kaspersky Labs Limited, "IP-Spoofing: Funktionsweise und Maßnahmen zur Prävention." [Online]. Available: https://www.kaspersky.de/resource-center/threats/ip-spoofing