université de BORDEAUX

Master Cryptologie et sécurité informatique

Travail d'étude et de recherche
Attaque par empoisonnement du cache DNS en exploitant le taux limite de ICMP

Auteurs :
Hanane Benarab
Alexandre Rose
Youssef Lachabi

Encadrant :
Abdou Guermouche

15 mai 2023

Remerciements

Tout d'abord, nous sommes reconnaissants envers notre encadrant Monsieur Abdou Guermouche pour nous avoir offert l'opportunité de choisir un sujet aussi passionnant. Sa confiance en notre capacité à explorer ce domaine des cyber-attaques a été un moteur de motivation pour nous. Ses conseils éclairés, ses remarques constructifs et sa capacité à répondre patiemment à nos interrogations ont grandement facilité notre cheminement.

Introuction

Dans le monde numérique d'aujourd'hui, les cyber-attaques sont devenues une menace majeure pour les individus, les entreprises, les organisations et les nations. Les récentes attaques spectaculaires contre des infrastructures, des banques ou des hopitaux ont mis en évidence les conséquences dramatiques que ces attaques peuvent avoir sur les finances, la réputation et la sécurité des organisations. Dans ce contexte, il est devenu essentiel de comprendre les enjeux de la sécurité informatique et de mettre en place des mesures de protection efficaces pour se protéger contre ces menaces.

Le présent rapport de travail d'études et de recherches se concentre sur l'analyse approfondie d'une cyber-attaque spécifique : empoisonnement du cache DNS en exploitant un taux limite de ICMP. Notre objectif est d'examiner les conséquences, les aspects techniques et les solutions possibles pour renforcer la résilience des systèmes d'information et prévenir de futures attaques similaires.

Ce rapport est divisé en plusieurs parties. Dans la première partie, nous présentons le rôle crucial de DNS dans internet ainsi que son fonctionnement. Dans la deuxième partie, nous examinons la fragilité de DNS en souligant sa fameuse faille trouvée par Dan Kaminsky. Dans la troisième partie, nous détaillons comment une utilisation malveillante par un attatquant du taux ICMP peut le ramener à l'attaque de base de Kaminsky.

Nous espérons que notre étude permettra de sensibiliser davantage à la nécessité de renforcer la sécurité de DNS et d'adopter des mesures de protection adéquates.

Table des matières

Chapitre 1

C'est quoi DNS

1.1 L'important rôle que joue DNS dans notre vie quotidienne :

En novembre 1983, la croissance rapide d'internet a entraîné d'énormes problèmes de comptabilité. Pour résoudre ce problème, un groupe composé de Jon Postel, Paul Mockapetris et Craig Partrige a publié la RFC [1]882 qui a créé le système de noms de domaine (**D**omaine **N**ame **S**ystem en anglais) pour faciliter la navigation sur Internet. Grâce à DNS, les utilisateurs peuvent saisir des noms d'hôtes tels que "google.com" au lieu de "8.8.8.8". Chaque adresse contiendrait des informations allant du plus spécifique au plus général. Ce RFC présente les noms de style domaine, leur utilisation pour le courrier Internet ARPANET [2] et la prise en charge des adresses hôtes, ainsi que les protocoles et les serveurs utilisés pour mettre en œuvre les installations de noms de domaine.
Avant DNS, la résolution devait se faire grâce au fichier texte /etc/hosts. Cette méthode présente plusieurs inconvenients dont le faite que ce fichier doit être local à chaque machine et que celle-ci n'est pratique que dans un réseau de taille modeste.

DNS est alors un protocole qui joue un rôle crucial dans la communication efficace et sécurisée d'informations sur internet. Sa mission principale est de traduire les noms de domaine en adresses IP, ce qui permet aux ordinateurs connectés de charger des pages webs, d'envoyer des e-mails et de communiquer entre eux.

1.2 Les différents types de serveurs DNS

1.2.1 Le résolveur DNS récursif :

Un résolveur récursif est le premier composant de DNS qui entre en action lors d'une demande de transalation. Ce n'est pas un programme particulier ; Il s'agit d'un ensemble de fonctions de bibliothèque liées à des programmes d'application tels que Telnet, FTP, les navigateurs web, etc. Par exemple, si Telnet doit traduire

1. **R**equest **F**or **C**omments (RFC) est une série de publications émanant des principaux organismes de développement technique et de normalisation de l'internet.
2. ARPANET est l'acronyme du premier réseau à transfert de paquets de données conçu aux États-Unis par la Defense Advanced Research Projects Agency.

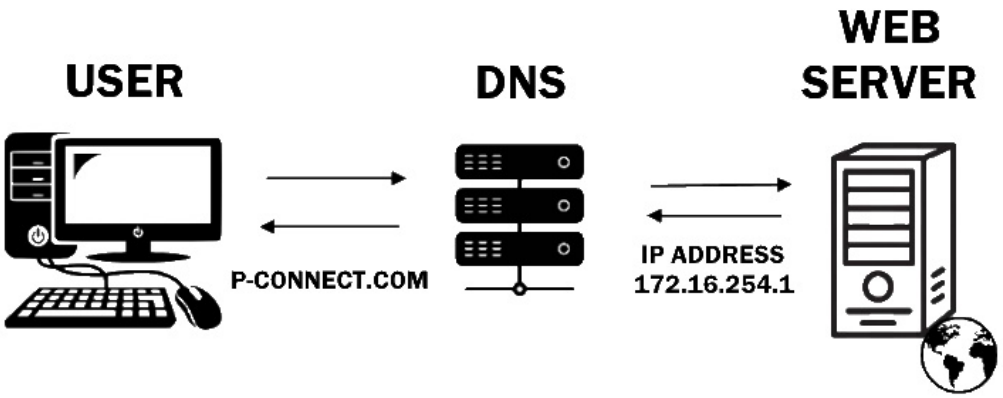


FIGURE 1.1 – Schéma descriptif des étapes d'une recherche DNS.

le nom d'un ordinateur en son adresse IP, il appelle les fonctions particulières de la bibliothèque liée. Dans ce cas, le résolveur appelle les fonctions de la bibliothèque (gethostbyname), qui formulent la requête et l'envoient récursivement aux serveurs racines.

1.2.2 Serveurs de noms racine :

Les 13 serveurs de noms racine DNS sont connus de tous les résolveurs récursifs. Ils constituent le premier arrêt dans la quête d'un résolveur récursif pour les enregistrements DNS. Un serveur racine accepte une requête de résolveur récursif qui inclut un nom de domaine, puis le serveur de noms racine répond en dirigeant le résolveur récursif vers un serveur de noms TLD, en fonction de l'extension de ce domaine (.com, .net, .org, etc.). Les serveurs de noms racine sont supervisés par un organisme sans but lucratif appelé ICANN (**I**nternet **C**orporation for **A**ssigned **N**ames and **N**umbers).
Le faite qu'il n'y ait que 13 serveurs de noms racine ne signifie pas qu'il n'y a que 13 machines dans le système. Il y a 13 types de serveurs de noms racine, mais de multiples copies de chaque type de serveur à travers le monde qui utilisent le routage Anycast pour fournir des réponses rapides. En additionnant tous les serveurs de noms racine, on obtient 1698 serveurs différents (décompte en Avril 2023).

1.2.3 Serveur DNS autoritaire :

Un serveur DNS autoritaire est un serveur DNS qui contient les enregistrements DNS originaux et à jour pour un domaine spécifique. Il est chargé de répondre aux requêtes DNS concernant les noms de domaine qu'il héberge. En d'autres termes, lorsqu'un serveur DNS récursif reçoit une requête DNS pour un nom de domaine

spécifique, il interroge le serveur DNS autoritaire pour ce domaine afin d'obtenir l'adresse IP correcte ou d'autres enregistrements DNS.

Par exemple, supposons que vous soyez propriétaire du domaine "exemple.com" et que vous hébergiez votre site web sur un serveur web dont l'adresse IP est 203.0.113.1. Vous devez configurer un serveur DNS autoritaire pour le domaine "exemple.com" afin qu'il fournisse des enregistrements DNS qui pointent vers l'adresse IP du serveur web.

Lorsqu'un utilisateur saisit "exemple.com" dans son navigateur web, son ordinateur envoie une requête DNS au serveur DNS récursif, qui envoie ensuite une requête au serveur DNS autoritaire pour "exemple.com". Le serveur DNS faisant autorité répond avec l'adresse IP correcte, que le serveur DNS récursif utilise ensuite pour diriger la requête de l'utilisateur vers le serveur web correct.

Il existe deux types de serveur DNS autoritaire :

Serveur DNS autoritaire primaire : Il s'agit du serveur DNS principal pour un domaine. Il stocke et gère les enregistrements DNS originaux et à jour pour le domaine.

Serveur DNS autoritaire secondaire : Il s'agit d'un serveur DNS de secours qui reçoit une copie des enregistrements DNS du serveur primaire. Il sert de mesure de redondance pour garantir que les requêtes DNS peuvent toujours être résolues en cas de défaillance du serveur primaire.

1.2.4 Serveur DNS de domaine de premier niveau(TLD) :

Un serveur DNS autoritaire et un serveur de domaine de premier niveau (TLD **T**op **L**evel **D**omain) sont tous les deux des types de serveurs DNS, mais ils remplissent des fonctions différentes dans la hiérarchie DNS.

Un serveur DNS autoritaire stocke et gère les enregistrements DNS originaux et à jour pour un domaine spécifique. En revanche, un serveur TLD est chargé de fournir des informations sur les domaines de premier niveau, tels que ".com", ".org", ".net", etc. Lorsqu'un serveur DNS récursif reçoit une requête DNS pour un nom de domaine appartenant à un TLD spécifique, il interroge le serveur TLD afin d'obtenir le serveur DNS autoritaire correspondant.

1.3 Les étapes de la translation d'une adresse web en une adresse IP :

Pour comprendre le fonctionnement de cette translation, il peut s'avérer utile de suivre le parcours d'une recherche DNS, depuis son émission dans le navigateur web jusqu'à son traitement par le processus de recherche DNS lui-même, avant de revenir à son origine dans le sens inverse. Tout d'abord, il est nécessaire de se rappeler que dans la plus-part du temps, les informations d'une recherche DNS sont souvent mises en cache localement, dans l'ordinateur à l'origine de la requête

ou à distance au sein de l'infrastructure DNS. Lorsque les informations DNS sont mises en cache, le processus de recherche DNS ignore certaines étapes, accélérant d'autant son traitement. Cependant, une recherche DNS se compose généralement de 8 étapes, l'exemple ci-dessous les décrits :

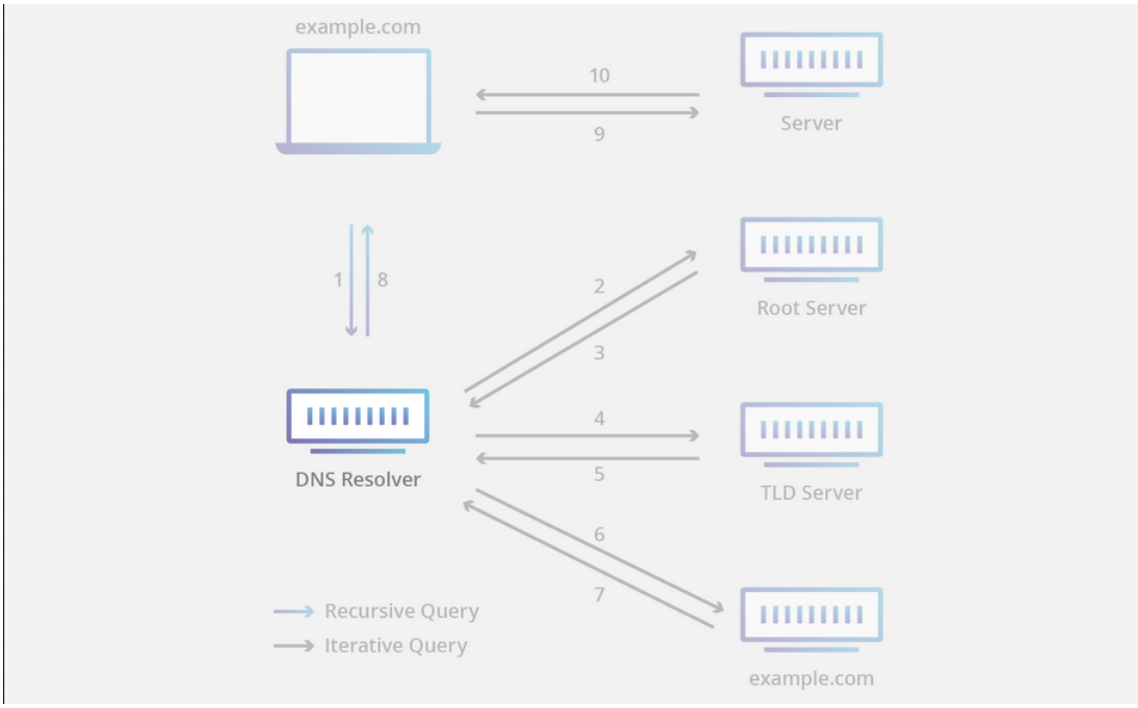


FIGURE 1.2 – Schéma descriptif des étapes d'une recherche DNS.

(1) La requête DNS est envoyée : Lorsqu'un utilisateur entre un nom de domaine dans son navigateur web par exemple "example.com", un processus de résolution de noms de domaine est déclenché. Le navigateur envoie une requête DNS. Cette dernière est acheminée via Internet et reçue par un résolveur DNS récursif.

(2) Le résolveur interroge alors un serveur de noms racine DNS.

(3) Ce dernier répond au résolveur avec l'adresse d'un serveur DNS de domaine de premier niveau (TLD) (comme « .com » ou .net) sur lequel sont conservées les informations relatives à ses domaines. Ainsi, lors d'une recherche portant sur « exemple.com », la requête est dirigée vers le TLD « .com ».

(4) Le résolveur émet ensuite une requête vers le TLD « .com ».

(5) Le serveur TLD répond alors avec l'adresse IP du serveur de noms de domaine : « exemple.com ».

(6) Enfin, le résolveur récursif envoie une requête au serveur de noms de domaine.

(7) Le serveur de noms de domaine renvoie ensuite l'adresse IP du site exemple.com vers le résolveur.

(8) Le résolveur DNS répond alors au navigateur web avec l'adresse IP du domaine initialement demandé.

Une fois que les 8 étapes de la recherche DNS ont renvoyé l'adresse IP d'exemple.com, le navigateur peut émettre la requête à la page web :

(9) Le navigateur envoie une requête HTTP à l'adresse IP.

(10) Le serveur situé à cette adresse IP renvoie la page web à afficher dans le navigateur.

Il existe une règle sur l'internet selon laquelle une base de données contenant les données nécessaires aux traductions est toujours sauvegardée sur au moins deux machines indépendantes (serveurs de noms indépendants). Si l'un d'eux n'est pas disponible, la traduction peut être effectuée par l'autre machine. En général, on ne peut pas s'attendre à ce que tous les serveurs de noms soient accessibles en permanence. Si le protocole TCP est utilisé pour une traduction, les tentatives de l'établissement d'une connexion avec un serveur de noms inaccessible entraîneraient de longs intervalles de temps pendant que le protocole TCP tente de se connecter. Ce n'est qu'une fois cet intervalle de temps écoulé qu'il est possible de se connecter au serveur de noms suivant. La solution du protocole UDP est plus

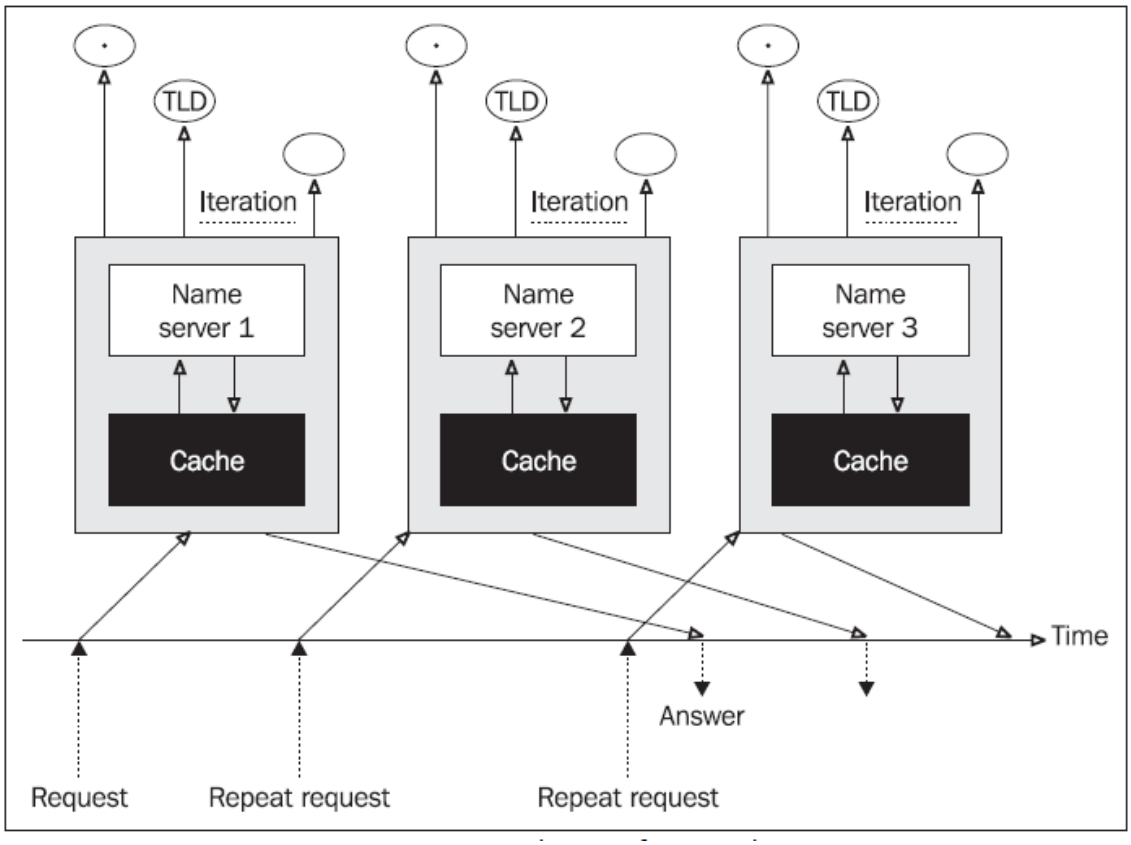


FIGURE 1.3 – Schéma descriptif des étapes d'une recherche DNS.

élégante : Un datagramme contenant une demande de traduction est envoyé au premier serveur. Si la réponse n'arrive pas dans un court intervalle de temps, un datagramme contenant une demande est envoyé à un autre serveur de noms, si la

réponse n'arrive pas, il est envoyé au suivant, et ainsi de suite. Si tous les serveurs de noms possibles sont utilisés, le cycle recommence à partir du premier, jusqu'à ce que la réponse revienne ou que l'intervalle fixé soit écoulé.

1.4 Le cache DNS :

1.4.1 L'utilité de stocker les réponses DNS :

Le cache DNS est utilisé pour stocker temporairement les réponses de requêtes DNS qui ont été précédemment demandées par un ordinateur ou un appareil. Lorsqu'un ordinateur effectue une requête DNS pour résoudre le nom de domaine en adresse IP, le serveur DNS consulté renvoie la réponse correspondante. Cette réponse est alors stockée dans le cache DNS local de l'ordinateur pour une période de temps déterminée (appelée "time-to-live" ou TTL).

Le stockage en cache permet de réduire le temps nécessaire pour résoudre les noms de domaine et améliore les performances globales du système en réduisant le trafic de requêtes DNS sur le réseau. En effet, lorsque la même requête DNS est effectuée ultérieurement, l'ordinateur peut récupérer la réponse à partir de son cache local plutôt que de devoir contacter à nouveau le serveur DNS. Cela permet de réduire la latence et d'améliorer la vitesse de navigation sur Internet.

En outre, la mise en cache des réponses DNS peut également contribuer à renforcer la sécurité en réduisant les risques d'attaques par empoisonnement de cache DNS. En effet, en évitant les requêtes DNS répétées, le cache limite les opportunités pour les pirates informatiques de falsifier les réponses DNS et de rediriger les utilisateurs vers des sites malveillants.

En résumé, la mise en cache des réponses DNS permet d'améliorer les performances, la rapidité et la sécurité du système de résolution des noms de domaine en réduisant les temps de latence et les risques d'attaques malveillantes.

1.4.2 Les deux types de cache DNS :

La mise en cache DNS dans le navigateur et dans le système d'exploitation sont deux méthodes différentes pour stocker temporairement les réponses de requêtes DNS.

La mise en cache DNS dans le navigateur est spécifique à chaque navigateur web. Lorsque le navigateur effectue une requête DNS pour résoudre un nom de domaine, il peut stocker la réponse dans son propre cache DNS local pour une durée déterminée. Si une autre requête est effectuée pour le même nom de domaine dans le même navigateur pendant la durée de validité de la réponse, le navigateur peut récupérer la réponse depuis son cache local sans avoir à effectuer une nouvelle requête DNS.

D'autre part, la mise en cache DNS dans le système d'exploitation est commune à tous les programmes et applications qui utilisent la résolution de noms de domaine. Le système d'exploitation stocke les réponses de requêtes DNS dans son propre cache DNS local, qui peut être utilisé par tous les programmes et applications du système. Cela peut améliorer les performances globales du système en réduisant le nombre de requêtes DNS envoyées à des serveurs DNS externes et en réduisant la latence de la résolution de noms de domaine pour l'ensemble du système.

En somme, la mise en cache DNS dans le navigateur est limitée à ce dernier, tandis que la mise en cache DNS dans le système d'exploitation est utilisée par tous les programmes et applications qui nécessitent la résolution de noms de domaine.

Chapitre 2

DNS comme vecteur d'attaques :

2.1 Fragilité de DNS :

DNS est un élément essentiel de l'infrastructure d'internet. Cependant, malgré son importance, DNS est exposé à une série d'attaques qui peuvent compromettre la sécurité et l'intégrité du système. Voici quelques exemples des cyber-attaques que DNS a subit :

(i) **Empoisonnement du cache DNS :** Dans cette attaque(que nous allons aborder en détails dans la partie qui suit), un pirate exploite une faiblesse dans le mécanisme de mise en cache du DNS pour insérer de fausses informations dans le cache d'un résolveur DNS. Cela peut amener le résolveur à fournir des informations DNS incorrectes aux clients. Un client peut ainsi être redirigé vers un faux site web ou voir ses informations sensibles compromises.

(ii) **Attaques DDoS :** DNS est également vulnérable aux attaques par déni de service distribué (DDoS), qui consistent à envoyer un grand nombre de requêtes à un serveur DNS afin de le submerger et de le rendre indisponible pour les utilisateurs légitimes.

(iii) **Attaques par transfert de zone :** Les serveurs DNS sont vulnérables aux attaques par transfert de zone, qui permettent à un pirate de demander une copie de l'ensemble du fichier de zone DNS d'un domaine. Cela peut lui permettre de recueillir des informations sensibles sur le domaine, telles que les adresses électroniques et les noms de serveurs.

2.2 Les requêtes DNS :

Pour bien comprendre l'attaque de l'empoisonnement du cache DNS, il faut avant tout voir en détails ce que c'est une demande d'information auprès de DNS (DNS Query en anglais) et quel est son format.

L'opération DNS se compose d'une requête et d'une réponse. Une requête contient une demande pour une ou plusieurs fiches de resources (RR Resource Records en anglais) de la base de données DNS. La réponse contient le RR en

question ou est un refus. Le RR contenu dans une réponse peut être la réponse finale ou aider le client à formuler une autre requête DNS pour atteindre l'objectif, c'est-à-dire formuler une autre itération. Une DNS Query utilise le même format de paquets pour les requêtes et les réponses. Un paquet se compose de cinq sections :

(i) **En-tête :** L'en-tête du paquet est obligatoire et est présente à la fois dans la requête et dans la réponse. Les deux premiers octets (2 x 8 = 16 bits) d'un en-tête contiennent un identifiant de requête (ID request). Un ID de requête est généré par un client et copié dans la réponse par un serveur. L'ID permet d'identifier de manière unique quelle requête correspond à quelle réponse. L'ID permet aussi à un client d'envoyer plusieurs requêtes à la fois sans attendre de réponse.

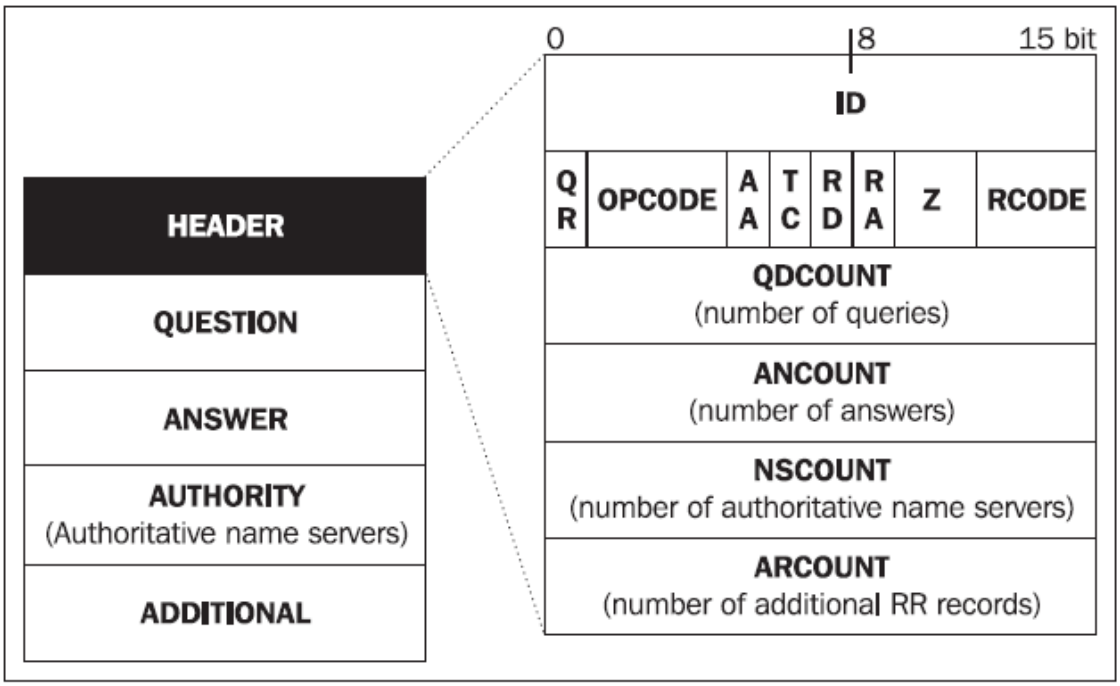


FIGURE 2.1 – Header en détails.

(ii) **Question :** Les paquets de requêtes DNS contiennent généralement une seule question(**QDCOUNT=1**) et par conséquent une seule section de question. Cette dernière se compose de trois champs : **QNAME :** contient le nom du domain, **QTYPE :** le type de cette question (A : IPv4, AAAA : IPv6, MX : Échange de mail, SIG : signature...) et **QCLASS :** la classe de la requête(IN : internet, * : Toutes les classes...).

```
Frame 2 (318 bytes on wire, 318 bytes captured)
Ethernet II, Src: Cisco_8e:1f:80 (00:15:63:8e:1f:80), Dst: Fujitsu_79:5d:0e
Internet Protocol, Src: 160.217.1.10 (160.217.1.10), Dst: 160.217.208.142
User Datagram Protocol, Src Port: domain (53), Dst Port: 1337 (1337)
Domain Name System (response)
    Transaction ID: 0x000c
    Flags: 0x8180 (Standard query response, No error)
    Questions: 1
    Answer RRs: 3
    Authority RRs: 6
    Additional RRs: 6
    Queries
        www.google.com: type A, class IN
            Name: www.google.com
            Type: A (Host address)
            Class: IN (0x0001)
    Answers
    Authoritative nameservers
    Additional records
```

FIGURE 2.2 – Un exemple de paquet DNS avec la section de questions indiquée en gras.

(iii) **Réponse :** Contient les addresses IP du nom de domaine recherché, il faut noter qu'on peut avoir plusieurs réponses à une seule requête DNS.

```
Frame 2 (318 bytes on wire, 318 bytes captured)
Ethernet II, Src: Cisco_8e:1f:80 (00:15:63:8e:1f:80), Dst: Fujitsu_79:5d:0e
Internet Protocol, Src: 160.217.1.10 (160.217.1.10), Dst: 160.217.208.142
User Datagram Protocol, Src Port: domain (53), Dst Port: 1337 (1337)
Domain Name System (response)
    Transaction ID: 0x000c
    Flags: 0x8180 (Standard query response, No error)
    Questions: 1
    Answer RRs: 3
    Authority RRs: 6
    Additional RRs: 6
    Queries
    Answers
        www.google.com: type CNAME, class IN, cname www.l.google.com
            Name: www.google.com
            Type: CNAME (Canonical name for an alias)
            Class: IN (0x0001)
            Time to live: 11 minutes, 32 seconds
            Data length: 8
            Primary name: www.l.google.com
        www.l.google.com: type A, class IN, addr 72.14.207.99
            Name: www.l.google.com
            Type: A (Host address)
            Class: IN (0x0001)
            Time to live: 4 minutes, 15 seconds
            Data length: 4
            Addr: 72.14.207.99
        www.l.google.com: type A, class IN, addr 72.14.207.104
            Name: www.l.google.com
            Type: A (Host address)
            Class: IN (0x0001)
            Time to live: 4 minutes, 15 seconds
            Data length: 4
            Addr: 72.14.207.104
    Authoritative nameservers
        l.google.com: type NS, class IN, ns d.l.google.com
            Name: l.google.com
```

FIGURE 2.3 – Un exemple de paquet DNS avec la section de réponse en gras.

(iv) **Serveur de noms autoritaires :** Contient le nom des serveur qui permis la translation et où le domaine est enregistré.

(v) **Partie supplémentaire :** Contient les addresses IP des serveurs autoritaires ayant contribuer à la translation.

2.3 Empoisonnement du cache DNS en exploitant la faille trouvée par Dan Kaminsky

"Parce qu'un homme de mauvaise foi peut provoquer ces courses de manière répétée sur le même serveur de noms, il finira par gagner" Daniel Kaminsky

2.3.1 Description de l'attaque :

Les implémentations précédentes des services DNS contenaient une vulnérabilité par laquelle des demandes multiples pour le même enregistrement de ressource (RR) généraient plusieurs requêtes en attente pour ce RR. Par conséquent, il est possible pour un attaquant d'appliquer une technique d'"attaque d'anniversaire" afin d'améliorer considérablement la probabilité de réussite d'une attaque d'usurpation d'identité DNS. Lorsqu'elle est effectuée contre un serveur de noms en cache, cette technique peut entraîner l'empoisonnement du cache.

En générant rapidement plusieurs requêtes pour un RR vers un résolveur vulnérable, l'attaquant peut induire une condition dans laquelle le résolveur vulnérable a plusieurs requêtes ouvertes pour ce RR. L'attaquant envoie ensuite un certain nombre de réponses usurpées au résolveur vulnérable. Avec la bonne combinaison de requêtes ouvertes et de réponses falsifiées, l'attaquant peut obtenir une forte probabilité de succès avec beaucoup moins de paquets (de plusieurs ordres de grandeur) que l'approche traditionnelle de force brute ne l'exigerait.

La seule différence entre cette attaque et l'approche traditionnelle par force brute (1 requête avec plusieurs réponses usurpées) est la génération de plusieurs requêtes simultanées. L'attaquant n'a pas besoin de renifler les paquets entre le résolveur victime et les serveurs de noms légitimes pour le RR usurpé. Le succès d'un attaquant contre une instance particulière de résolveur sera de nature probabiliste, un attaquant persistant étant toujours en mesure d'obtenir une probabilité raisonnable de succès après un nombre suffisant de tentatives.

2.3.2 Impacte de l'attaque :

Un attaquant capable de mener à bien une attaque par empoisonnement de cache peut amener les clients d'un serveur de noms à contacter des hôtes incorrects, voire malveillants, pour des services particuliers. Par conséquent, le trafic web, le courrier électronique et d'autres données importantes du réseau peuvent être redirigés vers des sites contrôlés par l'attaquant.

2.3.3 Randomisation du port source comme solution pour contourner la faille :

Sachant que les attaques contre ces vulnérabilités reposent toutes sur la capacité d'un attaquant à usurper le trafic de manière prévisible, la mise en œuvre d'une randomisation du port source par requête dans le serveur constitue une mesure pratique d'atténuation de ces attaques dans les limites de la spécification actuelle du protocole. Les ports sources randomisés peuvent être utilisés pour obtenir environ 16 bits supplémentaires d'aléa dans les données qu'un attaquant doit deviner. Bien qu'il y ait techniquement 65 535 ports, les utilisateurs ne peuvent pas tous les allouer (les numéros de port $<$1024 peuvent être réservés, d'autres ports peuvent déjà être alloués, etc.) Toutefois, le fait de rendre aléatoires les ports disponibles permet d'accroître considérablement la résistance aux attaques.

Il est important de noter qu'en l'absence de modifications du protocole DNS, telles que celles introduites par les extensions de sécurité DNS (DNSSEC), ces mesures d'atténuation ne peuvent pas empêcher complètement l'empoisonnement du cache. Toutefois, si elles sont correctement mises en œuvre, ces mesures réduisent les chances de réussite d'un attaquant de toute évidence et rendent les attaques impossibles à mettre en œuvre.

Le nombre spécifique de paquets requis pour exploiter un résolveur DNS particulier varie selon la mise en œuvre, mais le tableau suivant énumère quelques-uns des scénarios les plus courants que l'on trouve dans un certain nombre de mises en œuvre.

If the attacker has to guess...	...and is limited to the following number of open requests...	...it will take the following number of packets to achieve a 50% success rate (includes both requests and responses)
TID only (16bits)	1	32.7 k (2^{15})
TID only (16bits)	4	10.4 k
TID only (16bits)	200	427
TID only (16bits)	unlimited	426
TID and port (32 bits)	1	2.1 billion (2^{31})
TID and port (32 bits)	4	683 million
TID and port (32 bits)	200	15 million
TID and port (32 bits)	unlimited	109 k

FIGURE 2.4 – Nombre de paquets nécessaires pour atteindre une probabilité de succès de 50 pour 100 pour différents nombres de requêtes ouvertes

Chapitre 3

DNS reste toujours vulnérable

Comme expliqué dans la partie précédente, avec la randomisation du port source UDP, les attaques de type brute force ne sont plus possibles. En effet, l'univers devient de taille beaucoup trop importante. Cependant, nous présenterons dans cette partie qu'il existe des stratagèmes afin de se ramener à l'attaque de Kaminsky, avec pour seul inconnu à deviner le numéro de transaction qui est comme expliqué auparavant facilement devinable. Effectivement, des protections initialement déployées pour améliorer la sécurité des systèmes peuvent être détournées de leur usage premier afin de faciliter la tâche des attaquants.

3.1 Side channel attaque basée sur des taux limite

3.1.1 Définition brève du protocole TCP

Le protocole TCP fonctionne selon un modèle client-serveur, où un client initie une connexion à un serveur distant. La connexion TCP est établie à travers un processus en trois étapes appelé la "poignée de main TCP". Le client envoie un paquet spécial appelé paquet SYN au serveur, indiquant son intention d'établir une connexion. Le serveur répond ensuite avec un paquet SYN-ACK pour confirmer la réception et accepter la connexion. Enfin, le client envoie un dernier paquet ACK pour confirmer la réception de la réponse du serveur. Une fois que cette phase est terminée, la connexion TCP est établie et les données peuvent être échangées. De plus, TCP utilise un numéro de séquence pour identifier chaque octet de données envoyé et un accusé de réception (ACK) pour confirmer la réception réussie des données.

3.1.2 Exploit TCP

Considérons maintenant une attaque où une connexion TCP existe entre un client et un serveur. L'attaquant n'a pas accès à leurs échanges, mais il connaît le quadruplé (IP source,Port source, IP destination, Port destination).

Il est alors très facile pour lui de créer un paquet se faisant passer pour le client, ce paquet peut posséder le flag SYN et donc le serveur va comprendre que le

client a redémarré la connexion et donc que la connexion actuelle doit être fermée (reset). Et pour que cela arrive il suffit que le paquet spoofé possède un numéro de séquence compris dans une certaine fenêtre (compris entre le numéro de séquence que le serveur s'attend à recevoir plus la taille de la fenêtre de réception). Cela est donc relativement simple, il suffit d'envoyer un certain nombre de paquets parcourant l'espace des possibles pour le numéro de séquence, et fatalement après quelques secondes, la connexion se verra fermée alors que le client n'avait jamais initié ce redémarrage.

3.1.3 Solution : ACK Challenge

Pour se défendre contre de telles attaques, la RFC 5961 propose plusieurs modifications sur la façon dont TCP doit traiter les paquets entrants. Notamment, sur la gestion de la réception d'un paquet avec le flag SYN lors d'une connexion en cours, maintenant le serveur répond à ce paquet avec un ACK Challenge, possédant un numéro d'ACK tiré aléatoirement par le serveur. Ainsi la connexion n'est terminée que si le serveur reçoit une réponse a cet ACK possédant le bon numéro de séquence (issus du numéro d'ACK généré), donc notre attaquant qui n'a pas accès au trafic entre les deux entités ne pourra pas répondre correctement et la connexion restera ouverte. Mais l'envoi de paquet ACK Challenge peut consommer des ressources et si un attaquant fait exprès de les provoquer il peut même effectuer un déni de service du serveur qui passera son temps à envoyer ces paquets. C'est pour cela que le noyau Linux implémente un taux global limite, par défaut à 100 par seconde, qui stipule que l'on peut envoyer au maximum 100 paquets de type ACK Challenge par seconde.

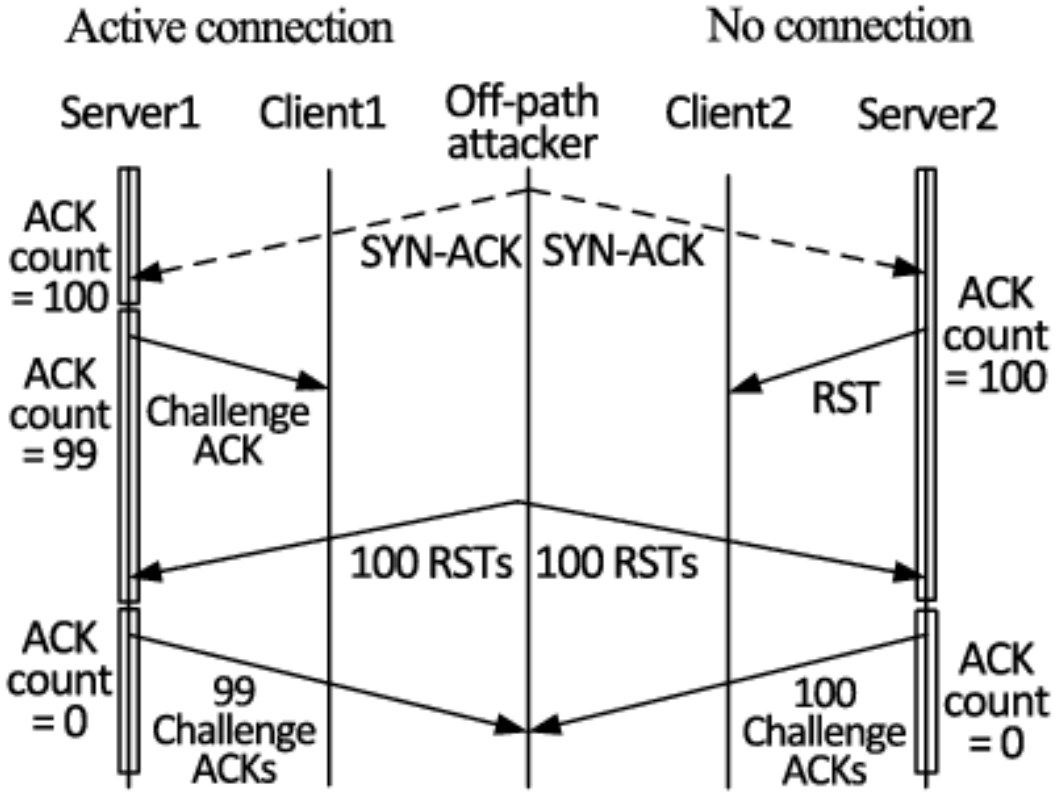
3.1.4 Utilisation du taux limite comme oracle

Ce compteur pourtant introduit afin de garantir la sécurité, va être détourné afin de deviner des informations sur des connexions actives. Nous présenterons seulement son utilisation afin de vérifier si un certain quadruplé correspond à une connexion active, mais il peut être utilisé aussi pour deviner les numéros de séquences et d'acquittement. L'attaque se passe comme précédemment à l'exception que cette fois ci l'attaquant ne sait pas si son quadruplé correspond a une

connexion active, c'est l'objectif de son attaque.

Il suffit à l'attaquant d'envoyer un paquet SYN-ACK en se faisant passé pour le client, avec le nouveau traitement des paquets de ce type, le serveur va générer un Challenge ACK si une connexion est vraiment ouverte sur ce quadruplé. L'attaquant va ensuite ouvrir une vraie connexion avec le serveur, et le forcer à lui envoyer 100 ACK Challenge, par exemple en envoyant 100 RESET (signifiant une fin de connexion).

À la fin, s'il n'a reçu que 99 ACK Challenge, alors il sait que le paquet avec le quadruplé spoofé a généré un ACK Challenge et donc cette connexion est active.Sinon il en aurait reçu 100 et donc ce quadruplé ne correspond à aucune connexion active.



Cette attaque est intéressante puisque nous savons que ce qui nous empêche d'exécuter l'attaque de Kaminsky est que nous ne savons pas deviner le port source randomisé en un temps limité. Mais peut-on s'inspirer de cette attaque et appliquer un protocole similaire sur UDP afin d'obtenir des informations. En effet, un taux global limite existe aussi dans de nombreux systèmes pour les messages utilisant le protocole ICMP.

3.1.5 Définition ICMP

Le protocole ICMP (Internet Control Message Protocol) est l'un des protocoles de base parmi la grande famille des protocoles Internet.ICMP permet la transmis-

sion de messages automatisés de contrôle et d'erreur afin que la machine émettrice sache que son paquet n'est pas arrivé à destination convenablement, par exemple lorsqu'un service ou un hôte devient inaccessible (Host Unreachable).

3.1.6 ICMP est un vecteur d'attaque

Comme vu en 3-1-3 avec l'ACK Challenge, des attaquants peuvent utiliser cette génération automatique de messages ICMP afin d'envoyer énormément de requêtes vers un serveur cible et donc causer un déni de service. De telles utilisations malveillantes de l'ICMP peuvent inclure un nombre élevé de paquets ping qui imitent une adresse IP source valide et une adresse IP de destination invalide (spoofed pings), ce qui a pour conséquence d'envoyer un nombre élevé de messages de réponse (tels que les messages d'erreur Destination Unreachable) générés par le réseau vers l'adresse IP source (que l'on a usurpé). De plus, un attaquant peut aussi forcer un serveur à générer des paquets ICMP en continu, ainsi ce dernier devient inaccessible car toutes ses ressources sont utilisées pour envoyer ces messages.

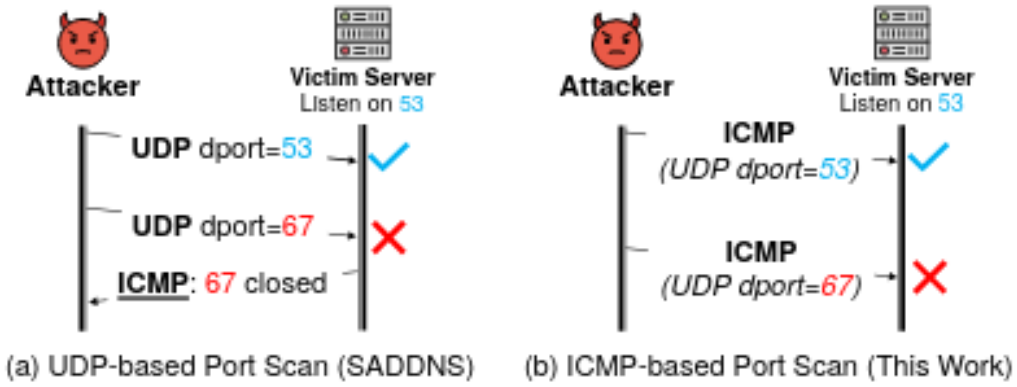
3.1.7 Solution : l'introduction d'un taux limite

Afin de résoudre ce problème, le noyau linux a mis en place une limitation de taux ICMP. Ces limites spécifient le nombre maximum de paquets ICMP autorisés à être envoyés par seconde ou par minute à partir d'une adresse IP. Si cette limite est dépassée, les paquets ICMP supplémentaires sont ignorés ou rejetés. Linux seulement est cité ici, car c'est le système d'exploitation le plus courant dans les serveurs et aussi le seul à utiliser une limite par IP. Mais il existe aussi une limite globale celle-ci présente dans la majorité des systèmes d'exploitation, ainsi en plus de limiter le nombre de réponse vers une seule IP, on limite aussi la charge en général. Ainsi, si n machines différentes communiquent en même temps sur le même serveur et qu'elles provoquent toutes la génération d'un message ICMP. Si n est supérieur à la limite globale du serveur, celui-ci n'émettra pas les n messages ICMP mais seulement jusqu'à sa limite (Usuellement 50 sous Linux). Ceci permet de protéger le serveur lui-même contre le déni de service.

3.1.8 Scan rapide de ports en utilisant le taux limite ICMP

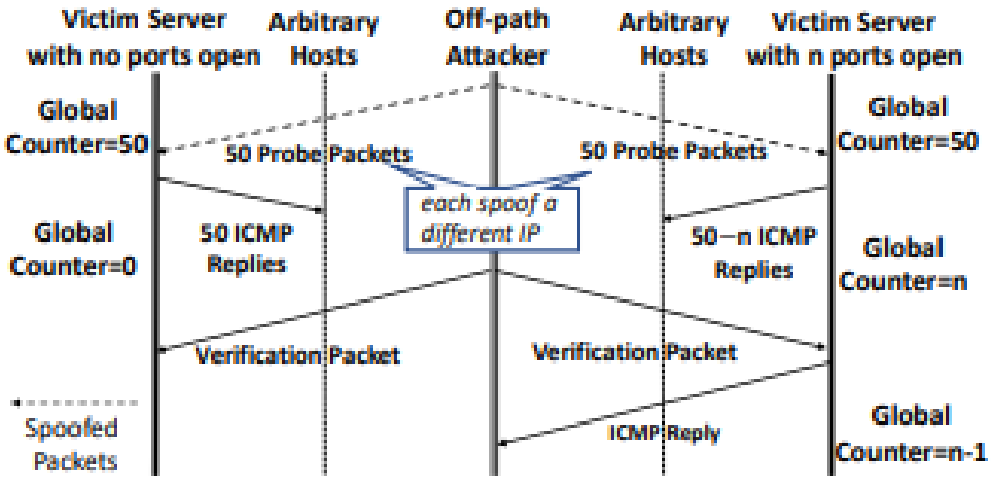
Cette fonctionnalité qui a premier lieu semble être une bonne mesure de sécurité, permet de scanner très rapidement les ports ouverts d'une machine. En effet,si un port est actuellement fermé un paquet avec comme port source ce dernier générera une réponse ICMP, «Port Unreachable».

Dans le noyau linux le taux limite globale ICMP accepte des «rafales» de 50 paquets. Ainsi si une entité envoie 50 requêtes et que ces dernières génèrent toutes



(a) UDP-based Port Scan (SADDNS) (b) ICMP-based Port Scan (This Work)

un message ICMP, le taux limite sera atteint et donc si cette même entité envoie une autre requête sur un port qu'elle sait fermé, elle devrait recevoir un ICMP message. Mais, elle ne le recevra pas si la limite est atteinte. Donc un attaquant peut forger des rafales de 50 paquets sur 50 ports différents, puis envoyer une requête de vérification, s'il reçoit une réponse, au moins un parmi les 50 ports testés est ouvert, sinon il sait qu'ils sont tous fermés. Cependant, dû à la limite par adresse IP de message ICMP présente dans le noyau linux, il faut que l'attaquant puisse envoyer chaque paquet présent dans la rafale avec une IP source différente, pour cela il peut soit usurper des adresses IP (side-channel), ou s'il possède plusieurs adresses IP. Il peut même en possédant une seule machine avec une adresse iPv6, obtenir 2^{64} adresses IP différentes, la même chose est possible en iPv4 en utilisant le protocole DHCP afin d'obtenir de multiple adresse iPv4 privée. En fin de compte, l'IP source des paquets de «sonde» n'est pas très importante, car la réponse ICMP n'intéresse pas forcément l'attaquant donc même s'il n'a pas accès à ce que reçoit cette IP, ce n'est pas grave. Seulement le paquet de vérification doit posséder une IP source où il est possible de voir si la réponse ICMP a été émise ou non en fonction de si le taux limite a été atteint.



En utilisant ce protocole d'attaque il est possible de scanner les 65536 ports possibles en un peu plus de 60 secondes, si la première rafale n'a pas trouvé de port ouvert l'attaque attend au moins 50ms afin que le taux soit remis à 0 avant d'en forger une autre. À ce rythme, une entité est capable de scanner 1000 ports par seconde.

3.2 Kaminsky à nouveau réalisable

3.2.1 Rappel sur l'attaque

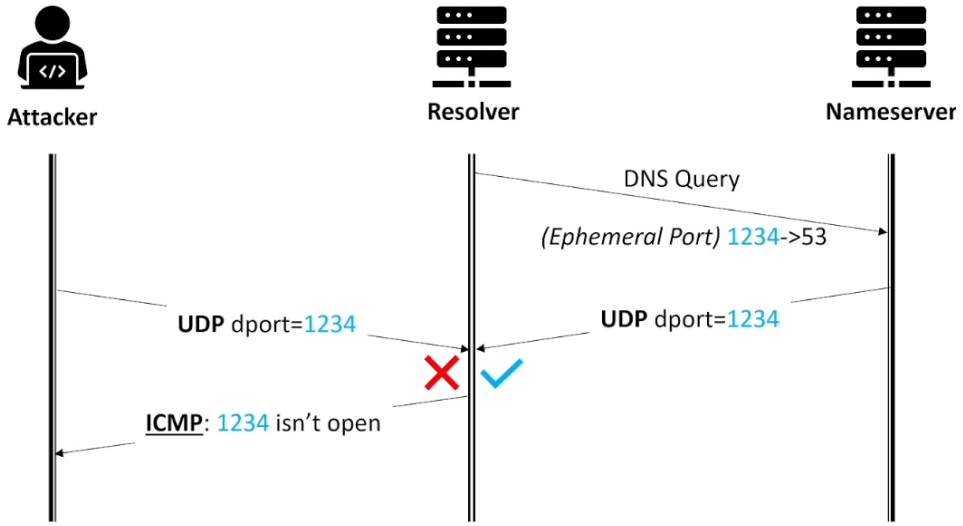
Dans les anciennes implémentations du système DNS, le port source utilisé pour les requêtes DNS sortantes était généralement fixé au port 53. L'attaquant exploitait cette prévisibilité pour effectuer une attaque par brute force du Transaction ID. Lors d'une attaque de Kaminsky, l'attaquant envoie une grande quantité de requêtes DNS falsifiées avec différents Transaction ID. L'attaquant tente de deviner le Transaction ID utilisé par la requête légitime du client. Une fois que l'attaquant a réussi à deviner le Transaction ID correct, il envoie une réponse DNS falsifiée avec le même Transaction ID pour tromper le serveur DNS cible. Le but de cette attaque est de faire en sorte que le serveur DNS mette en cache la réponse DNS falsifiée, ce qui peut entraîner une mauvaise résolution des noms de domaine et donc une redirection des utilisateurs vers des ressources malveillantes contrôlées potentiellement contrôlée par l'attaquant. Cependant, cette attaque ne fonctionnait plus, car il y a maintenant une randomisation du port source, de plus le brut force sur un port n'est pas possible puisque l'univers est de 2^{16} pour les ports et 2^{16} pour la transaction ID, ce qui fait 2^{32} possibilités différentes.

3.2.2 Exploitation du taux ICMP

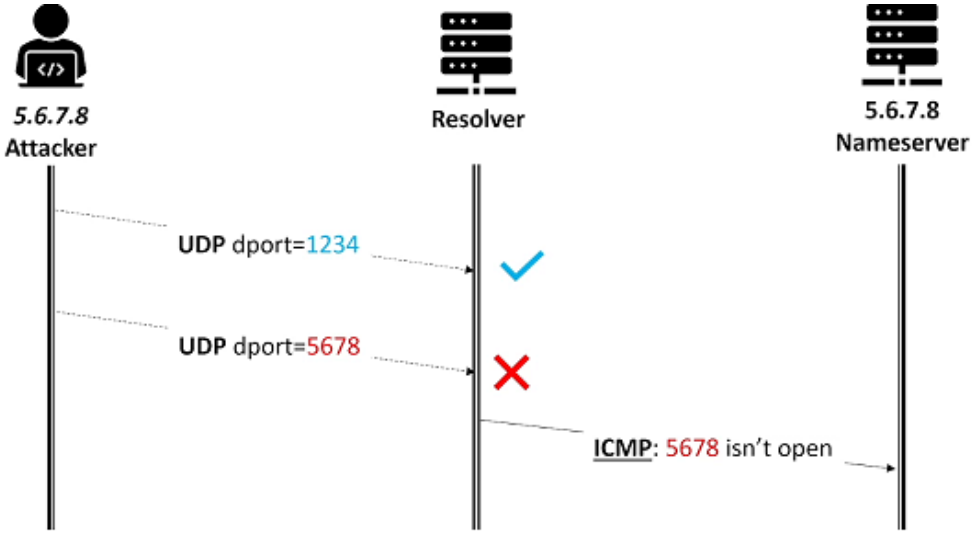
Mais avec l'exploitation du ICMP rate limite vu en 3-1-8, il est possible de se ramener à l'attaque de base sur DNS, Kaminsky. En devinant d'abord le port source puis le transaction ID, l'univers sera réduit à environ 2^{16}. En effet, lorsqu'un résolveur DNS résout un nom de domaine, il envoie une requête au serveur faisant autorité via UDP. Pendant que le résolveur attend la réponse, un attaquant hors chemin (c'est-à-dire qu'il n'a pas accès aux paquets envoyés entre les deux serveurs) peut exploiter cette vulnérabilité pour déduire rapidement le port éphémère de la requête sortante, et on se retrouve alors exactement dans le même contexte qu'auparavant.

3.2.3 Scan de port privé

Cependant, il se peut que le socket UDP entre le résolveur et le serveur de nom de domaine utilise connect(), ceci a pour effet de rendre le port privé rendant ce dernier ouvert uniquement pour la paire résolveur/serveur de noms.



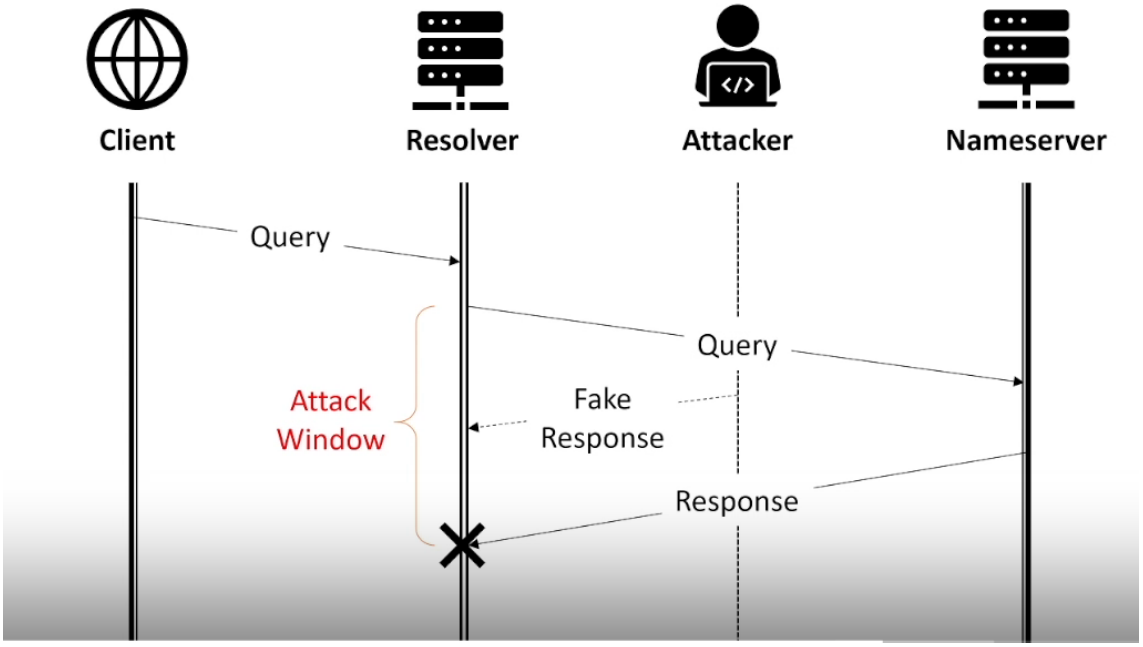
On pourrait penser que cela invalide complètement la méthode de recherche de port, mais ce n'est pas le cas. L'attaquant peut toujours forger des paquets en usurpant l'IP du serveur de nom de domaine, mais comme expliqué auparavant, il existe une limite ICMP par IP. Cela signifie que la vitesse de scan ne sera que d'un port par seconde, rendant l'attaque infaisable. Néanmoins, le taux global ICMP sous linux a été implémenté pour être vérifier avant la limite par IP, ainsi même si aucun paquet ICMP ne doit être envoyé car la limite a été atteinte pour cette IP, le compteur global sera quand même décrémenté. Ainsi il est toujours envisageable, de vérifier si un port est ouvert à une vitesse de 1000 par seconde même si ce dernier est privé. Une solution est de spécifier à la machine qu'elle ne doit plus émettre d'ICMP message : «UDP port unreachable». En détruisant les paquets sur des ports non ouverts et non plus en les rejetant.



3.3 Améliorations possibles de l'attaque

3.3.1 Agrandir la fenêtre d'attaque, pourquoi ?

Comment un attaquant peut-il augmenter sa fenêtre d'attaque, en effet pour l'attaque fonctionne, le port et le numéro de transaction ID doivent être identifié avant que le serveur de nom de domaine réponde, ainsi si l'attaquant répond avant ce dernier, l'attaque sera un succès. Il est alors intéressant de parvenir à retarder au plus possible cette réponse, l'attaquant aura plus de temps et donc pourra scanner plus de ports et au final augmenter les chances de réussite de l'attaque.

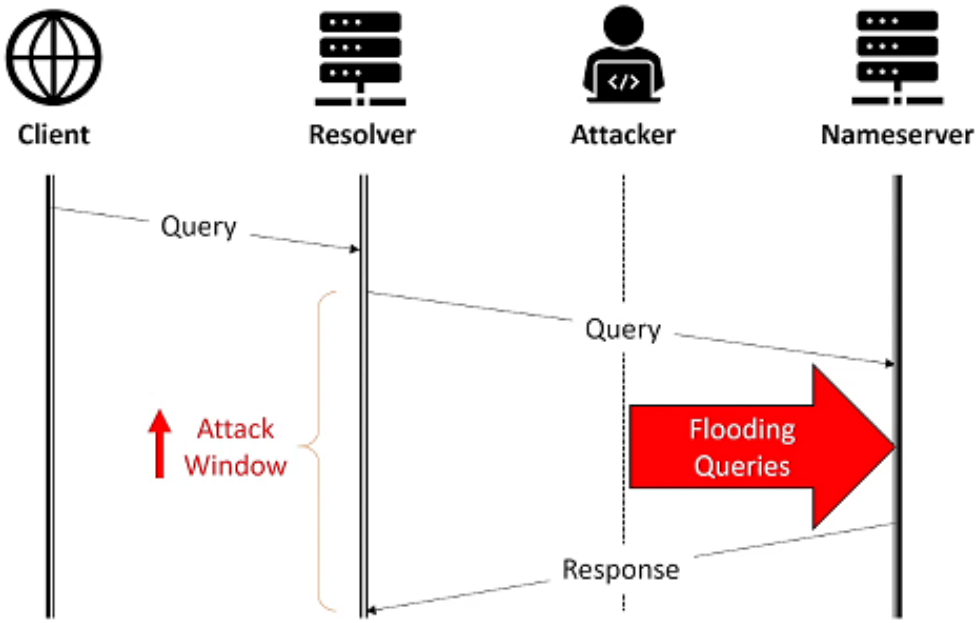


3.3.2 Comment agrandir la fenêtre d'attaque ?

Pour cela, il faut parvenir à forcer le serveur de nom d'autorité à répondre le plus lentement possible en utilisant le taux de réponse (RRL : Rate Reponse Limit). Initialement implémenté pour limiter les attaques par amplification DNS, où un grand nombre de requêtes de requêtes DNS malveillantes sont émises vers les serveurs de noms faisant autorité en usurpant l'adresse IP d'une victime afin que cette dernière se retrouve saturer de réponse. Concrètement, si la limite est atteinte, les réponses sont tronquées ou abandonnées.

Ironiquement, comme précédemment cette fonctionnalité peut être exploitée pour rendre muet un serveur de noms ou retarder sa réponse. L'attaquant usurpe l'IP du résolveur et envoie un grand nombre de requête au serveur de noms, la conséquence sera de ralentir la réponse de la vraie requête voir même que celle-ci ne soit jamais émise.

L'attaquant a donc maintenant beaucoup plus de temps pour scanner le port et ensuite forger sa fausse réponse



3.3.3 Attaque sur le serveur de redirection DNS

Une attaque similaire existe sur le serveur de redirection. Cette attaque DNS exploite le fait que les serveurs de redirection font confiance aux résolveurs en amont pour valider les réponses DNS. L'attaquant envoie une requête avec son propre domaine au serveur de redirection, qui la transmet au résolveur. L'attaquant contrôle un serveur de noms autoritaire qui est configuré pour ne pas répondre, ce qui entraîne un temps d'attente prolongé du serveur de redirection. Pendant ce temps, le port source reste ouvert, l'attaquant possède donc plus de temps pour fabriquer sa réponse frauduleuse. Et cette dernière est produite de telle sorte à répondre pour son propre domaine mais aussi pour celui d'une victime et cette réponse sera acceptée et les résolutions de domaine de l'attaquant et de la victime seront mis en cache.

3.3.4 Coutournement du TTL dans le cache

De même, si l'attaque échoue pour un certain nom de domaine alors la réponse authentique se voit mise en cache dans le résolveur durant une certaine période (TTL : time to live). Pour éviter d'attendre cette durée un attaquant peut contourner le TTL des enregistrements mis en cache. En injectant un enregistrement de serveur de noms fictif pour un domaine cible, tel que www.victim.com, en demandant par exemple la résolution de randomvalue.www.victim.com qui elle n'est pas dans le cache. Cela oblige maintenant le résolveur à interroger le serveur de noms autoritaire de l'attaquant pour obtenir de nouvelles réponses à propos de ce domaine.

3.4 Contre-mesures, des solutions viables ?

3.4.1 Solution contre le scannage de ports

La première mesure évidente qui peut être mise en place est le fait de désactiver la réponse ICMP «Port Unreachable» comme spécifié en 3-2. En plus de cela, il serait aussi intéressant de randomiser le taux global limite ICMP, si la valeur n'est plus toujours 50 ou un nombre facilement devinable le scan de ports se verra impacté négativement. DNS est un protocole pilier d'internet mais aussi un protocole peu sécurisé car il n'existe pas de moyen de vérifier l'authenticité d'une réponse.

3.4.2 Authentifier les réponses

Cependant, il existe plusieurs protocoles ayant comme objectif d'authentifier ces réponses.

DNSSEC (Domain Name System Security Extensions) est un ensemble de protocoles et de mécanismes de sécurité qui renforcent la sécurité des échanges de données DNS. Il vise à garantir l'intégrité, l'authenticité et l'origine des informations DNS en utilisant des signatures numériques. Avec DNSSEC, les enregistrements DNS sont signés numériquement par une signature cryptographique provenant des propriétaires du domaine concerné. Ainsi lors d'une requête DNS, le serveur DNS autoritaire fournit non seulement la réponse, mais également une signature. La personne ayant initié la requête DNS peut alors vérifier l'authenticité de la réponse en utilisant la clé publique du domaine, qui est obtenue à partir de la zone DNS parente. DNSSEC fournit également une chaîne de confiance où chaque zone DNS signe des enregistrements de zone subordonnés. Cela vérifie l'authenticité de toutes les parties impliquées dans la résolution DNS, du domaine cible jusqu'à la racine de l'arborescence DNS.

Il existe aussi le protocole DNS over HTTPS (DoH), ainsi au lieu de faire transiter les requêtes/réponses en clair, on les chiffre et l'envoi se fait via une connexion HTTPS. L'utilisation de HTTPS garantit l'intégrité et la confidentialité des requêtes DNS. En effet, en plus d'assurer l'authenticité d'une réponse les requêtes et réponses étant maintenant chiffrées, il n'est plus possible d'obtenir des informations sur les pages que consulte une entité. Il y a donc en plus une protection de la vie privé.

DNS est indispensable pour le bon fonctionnement d'internet mais pose des problèmes de sécurité important, il existe certain protocole capable d'authentifier les réponses mais dans les faits la majorité des résolveurs sont vulnérables comme on peut le voir sur ce tableau récapitulatif :

Name	Address	Example Backend Addr.	# of Backends	ICMP	Global Rate Limit	Using `connect()`	Vulnerable
Google	8.8.8.8	172.253.2.4	15	Y	Y	N	Y
CloudFlare	1.1.1.1	172.68.135.169	2	Y	Y	Y	Y
OpenDNS	208.67.222.222	208.67.219.11	107	Y	Y	Y	Y
Comodo	8.26.56.26	66.230.162.182	2	Y	Y	N	Y
Dyn	216.146.35.35	45.76.11.166	1	Y	Y	N	Y
Quad9	9.9.9.9	74.63.16.243	11	Y	Y	Y	Y
AdGuard	176.103.130.130	66.42.108.108	3	Y	Y	N	Y
CleanBrowsing	185.228.168.168	45.76.171.37	1	Y	Y	Y	Y
Neustar	156.154.70.1	2610:a1:300c:128::143	2	Y	Y	N	Y
Yandex	77.88.8.1	77.88.56.132	19	Y	Y	Y	Y
Baidu DNS	180.76.76.76	106.38.179.6	16	Y	Y	Y	Y
114 DNS	114.114.114.114	106.38.179.6	11	Y	N	N	Y
Tencent DNS	119.29.29.29	183.194.223.102	45	Y	N	N	N[1]
Ali DNS	223.5.5.5	210.69.48.38	160	N	N/A	N/A	N

Conclusion

Au cours de ce travail de recherche, nous avons pu comprendre le fonctionnement du protocole DNS qui a un rôle fondamental dans la résolutions des noms de domaine en adresse IP et donc qui est un pilier d'internet.

Nous nous sommes aussi rendu compte que ce protocole mis en place en 1985 possédait un énorme défaut, celui de ne pas assurer l'authenticité des réponses. Ce qui représente une faille de sécurité importante pouvant rediriger des utilisateurs vers un site frauduleux.

Dans un deuxieme temps, nous avons exploré les différentes attaques possibles visant à empoisonner le cache DNS. Ceci en partant de l'attaque la plus simple de Kaminsky et avons approfondies afin de voir que l'utilisation de side-channel et le détournement de certains protocoles de sécurités implémentés dans les sytèmes d'exploitations permettait toujours de compromettre DNS.

Finalement, nous nous sommes penchés sur les solutions possibles afin d'amener de la sécurité à ce protocole. Sachant qu'une refonte de ce dernier n'est pas envisageable car cela signifirait l'inaccessibilité à internet durant plusieurs heures. Nous avons vu que nous pouvions simplement chercher à rendre plus compliqué les attaques par side channel, en faisant des modifications sur la gestion des messages ICMP par les systèmes. Mais pour une sécurité pérenne, il faut authentifier les réponses. C'est ce que propose de faire DNSSEC à l'aide de signature cryptographique, permettant l'utilisateur de verifier la source de la réponse. L'utilisation de DNS over HTTPS est aussi faisable.

Ainsi, un protocole d'une haute importance tel que DNS reste hautement vulnérable en 2023 ce qui soulève de nombreuses questions, les moyens d'identifier les réponses devraient évoluer et devenir omniprésent dans les années à venir.