Dossier pour le CAPET externe dans la section sciences industrielles de l'ingénieur en option ingénierie informatique

Nicola Spanti

Mai 2017

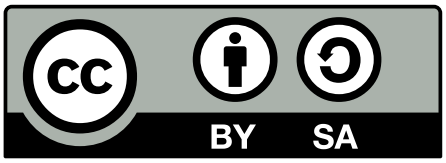

Table des matières

1 Introduction

En 2017, le réseau Internet est vastement utilisé dans les pays dits développés ou en voie de développement. Son usage est devenu courant pour de nombreux individus, y compris pour l'exercice de droits constitutionnels (comme la liberté d'expression). De plus, il est devenu indispensable à de nombreux collectifs de production, dont certains sont réputés producteurs de valeurs économiques (appelés couramment "entreprises"), auxquels la société francaise (et d'autres) a un attachement (systémique) particulier (puisqu'elles ont une place importante dans la vie de beaucoup de personnes). Les enjeux autour de ce réseau de télécommunications et des applications qui l'exploitent sont donc devenus majeurs.

Il faut qu'il y ait le moins de pannes possibles lors de son usage, et qu'il y ait une perception de rapidité. De plus, il est nécessaire de protéger les infrastructures et une partie de leurs contenus (les données personnelles et les secrets des organisations). On peut aussi considérer qu'il est souhaitable, dans certains cas, de privilégier certains flux au détriment d'autres (dont l'exemple le plus notable est probablement la notion de "services spécialisés" qui offusquent celles et ceux qui défendent la neutralité d'Internet). Ce sont des besoins techniques, leurs résolutions nécessitent des savoirs techniques et la capacité de les mettre en œuvre concrètement. Néanmoins ils ne peuvent pas être réduits à cela, en effet ces besoins techniques résultent d'une organisation politique de la société, dont certaines personnes (physiques ou morales) ont un ou des intérêts à attaquer. Des symptômes de ces intérêts sont par exemple des attaques par dénis de service ou modification de la page d'accueil d'un service (d'un concurrent, d'une organisation qui a une idéologie différente comme la Hadopi, etc) Pour répondre à ces besoins, il existe diverses réponses politiques et techniques. Besoins qui ne sont en réalité que les conséquences d'un seul besoin plus général : l'usage de l'infrastructure réseau doit sembler transparente pour celles et ceux qui l'utilisent (rapidité, absence de panne, sans effet indésirable, etc).

Une des réponses techniques est le proxy. Il consiste en un intermédiaire technique entre le demandeur d'une ressource et celui qui la possède supposément. Ce placement stratégique permet de nombreuses actions, dont certaines sont malveillantes (comme l'attaque dite *man in the middle*, ou attaque de l'humain du milieu en francais), mais également d'autres pouvant répondre aux besoins préalablement cités.

Un type très commun des proxys est le proxy web. Il permet bien souvent de garder une copie des ressources déjà demandées et reçues avec succès (ce qui évite au serveur web qui est derrière le proxy de les resservir) et de servir d'éponge s'il y avait un nombre inhabituellement élevé de requêtes (c'est ainsi le proxy qui doit avoir la capacité d'absorber, sans s'écrouler, un surplus non commun de requêtes). Dans les organisations qui gèrent une infrastructure, les proxys sont couramment utilisés. Ils permettent notamment de réduire les moyens potentiels de communiquer avec elle depuis l'extérieur, mais également l'inverse

(sa capacité à communiquer avec le dehors), tout en permettant d'enregistrer l'activité réseau ou une partie de celle-ci pour tenter de détecter une attaque en temps réel ou d'en identifier une a posteriori et d'en comprendre les mécanismes. Un usage moins courant mais en progression est l'enregistrement d'un parcours d'un individu dans une application (faisant usage du réseau Internet), pour vérifier qu'il marche toujours avec une version ultérieure de l'application.

Dans la première partie, nous analyserons plus en détails différents types de proxy. Dans la seconde partie, nous verrons des exploitations pédagogiques autour de cette thématique et en rapport avec ce qui est attendu par l'Éducation Nationale.

2 Partie scientifique et technique

2.1 Définition d'un proxy réseau

Un proxy permet l'accès à un type de ressource, mais indirectement. C'est un intermédiaire avec l'offreur réel de ressource(s). L'usage d'un proxy pourrait paraitre parasitaire (pourquoi rajouter un intermédiaire ?), mais ce n'est pas le cas, car un proxy offre au moins un service. Le ou les services qu'un proxy offre constitue sa plus-value, et donc la raison de son existence et son utilisation.

Un proxy réseau est un intermédiaire dans un réseau, comme Internet. Parmi les services qu'un proxy de ce type peut offrir, il y a :

- Filtrer des communications entrantes et/ou sortantes (pour un objectif de sécurité)
- Garder au plus près des données qui pourraient être redemandées, pour pouvoir les resservir rapidement et économiser de la bande passante, ce qui peut être décrit de façon plus concise comme un système de cache
- Anonymiser le demandeur réel d'une ressource vis-à-vis de l'offreur réel de cette dernière
- Absorber une demande inhabituellement excessive, ce qui permet à l'offreur réel d'une ressource de ne pas avoir à gérer cette éventualité, et donc de ne pas directement avoir à gérer techniquement et financièrement ce service s'il n'opère pas le proxy ayant cette fonction
- Enregistrer ce qui passe à travers lui ou au moins une partie pour :
  - l'analyser en temps réel ou a posteriori si le besoin s'en fait sentir
  - enregistrer un scénario de test qui pourra être rejoué pour vérifier que le fonctionnement réseau d'une application n'a pas changé

2.2 Proxy et chiffrement

2.2.1 Le chiffrement

Le chiffrement consiste à transformer des données via des principes mathématiques pour faire en sorte qu'uniquement le ou les personne(s) légitime(s) puisse(nt) lire les données originelles. Le déchiffrement consiste en l'opération inverse : transformer des données chiffrées en données claires (donc lisibles). Chacune de ses opérations nécessite une clé, qui est en réalité un "grand" nombre. Il y a donc une clé de chiffrement pour chiffrer et une clé de déchiffrement pour déchiffrer, du moins conceptuellement puisque dans certains systèmes (dits symétriques) une même clé a les 2 rôles.

Pour que ce genre de système soit efficace dans un système de communications, il faut qu'une personne qui intercepte un flux chiffré ne soit pas capable de retrouver le flux en clair, du moins pas dans un temps moyen raisonnable avec les capacités techniques de l'époque. Pour cela, il faut un secret partagé entre les 2 personnes qui communiquent, il s'agit de la clé de déchiffrement. La clé de chiffrement peut être publique si elle est distincte de la clé de déchiffrement, c'est-à-dire dans le cas d'un système dit asymétrique. Pour que le système soit efficace, il faut que la clé de déchiffrement ne puisse pas être retrouvée ni avec la clé de chiffrement, ni avec une donnée dans sa forme claire et chiffrée, ni avec la capacité de chiffrer n'importe quelle donnée. Bien entendu, il faut aussi que la clé de déchiffrement soit indispensable pour déchiffrer, sinon cela signifie que l'on peut décrypter, c'est-à-dire obtenir la donnée en clair à partir de sa version chiffrée sans la clé de déchiffrement.

Pour des raisons de confidentialité, le chiffrement est de plus en plus utilisé. Il a un intérêt fort dans le cadre des télécommunications, qui est d'ailleurs son principal champ d'application. Les révélations d'Edward Snowden sur la surveillance massive par des États ont fortement amplifié cette tendance. Les proxys reçoivent donc des données chiffrées, ce qui n'est pas nécessairement sans conséquence pour les opérations pour lesquelles ils ont été mis en place.

2.2.2 La problématique du chiffrement pour un proxy

Un proxy qui reçoit une donnée chiffrée sans la clé de déchiffrement ne peut réaliser aucune opération pertinente en rapport avec cette dernière. Or une donnée chiffrée pourrait contenir des informations nécessaires au(x) service(s) qu'il est censé réaliser. Pour les proxys, ainsi que celles et ceux qui les gèrent, il y a 2 cas de figure :

— Le ou les opération(s) du proxy empêché(s) par le chiffrement est/sont une plus value intéressante mais non nécessaire(s). Dans ce cas, il suffit de ne pas traiter les données chiffrées.
— Le ou les opérations du proxy impossible(s) à cause du chiffrement est/sont jugé(s) indispensable(s). Il va donc falloir d'une manière ou d'une autre contourner le chiffrement. Dans la suite de cette sous-sous-section, nous intéressons à ce cas.

Décrypter n'est pas possible, à moins de trouver au moins une faille dans le système de chiffrement. Mais pour cela, il faut avoir de fortes connaissances en mathématiques et du temps, ainsi que la "chance" qu'il y ait au moins une faille critique. C'est donc réservé aux chercheurs et chercheuses, et par extension à celles et ceux qui ont les moyens de s'offrir leurs services.

Une solution simple est de bannir ce qui est chiffré. Cela a le mérite d'être aisé techniquement et donc peu couteux. Néanmoins c'est mauvais pour la sécurité.

De plus, cette méthode n'est possible que si tous les services jugés indispensables auxquels les machines derrière le proxy veulent accéder acceptent de communiquer sans chiffrement. Pour ce qui est des services Internet publics, c'est de moins en moins le cas (puisque les fournisseurs de services ont intérêt à avoir une forte sécurité pour ne pas perdre la confiance du public), or leur usage est de plus en plus courant, et usuellement nommé sous l'appellation marketing de "*cloud*".

Une autre possibilité est de déchiffrer. Mais pour cela, il faut que le chiffrement ne soit plus du client au service final (puisque le proxy n'a pas la clé de déchiffrement du dit service), mais du client au proxy puis du proxy au service final. Pour faire cela, il faut que les clients aient une clé de chiffrement du proxy. C'est facile à mettre en place dans le cas d'ordinateurs contrôlés par l'organisation qui gère le proxy réseau. En effet, il suffit en effet de rajouter un certificat X.509, ce qui est aisé sur tous les systèmes d'exploitation grand public. Cependant, ce n'est pas suffisant, puisque le service final donne les informations pour chiffrer les données pour lui. Il faut donc que le proxy modifie ces informations pour les remplacer par les siennes. Ainsi, une fois que la configuration a été faite, c'est transparent pour les utilisatrices et utilisateurs.

Pour certains protocoles de communications, il est possible de faire un proxy non transparent. Avec ce genre de proxy, il est aisé de savoir qu'on se connecte à un service tiers, puisque c'est fait explicitement au niveau du protocole. Si une ressource fait une référence à une autre du même service, il est nécessaire de modifier la référence pour la faire pointer vers le proxy, ce qui nécessite de la puissance de calcul et une logique applicative dépendante du protocole. Pour contourner le chiffrement, il n'y a pas de problème puisque c'est au proxy de communiquer avec le service final et non plus au client. Ce genre de proxy est particulièrement utilisé pour le Web, par exemple pour contourner la censure avec un miroir (il en existe par exemple de nombreux pour *The Pirate Bay*) ou retrouver une page qui n'est plus en ligne (comme le propose la *Wayback Machine* de l'*Internet Archive*).

2.3 Proxy filtrant

2.3.1 Introduction sur les proxys filtrants

Internet est, comme sa dénomination l'indique, un réseau de réseaux. Concrètement, c'est le résultat de plusieurs réseaux inter-connectés, pas nécessairement directement les uns aux autres. Chaque réseau est administré par un opérateur, qui dans sa tâche n'a pas nécessairement les mêmes besoins que ces camarades. L'un peut par exemple souhaiter ne pas permettre l'accès à des applications qu'il juge inapproprié vis-à-vis de la raison de la mise en place du réseau. Par exemple, une école aura typiquement pour volonté d'interdire les sites web pornographiques. L'autre peut vouloir garder un ou des services en interne, car il(s) n'aurai(en)t (supposément) aucun intérêt au dehors ou plus communément pour

des raisons de sécurité (moins il y a de portes moins il y a de probabilité qu'au moins une soit mal fermée).

Certains de ses besoins peuvent se concrétiser en restriction(s) d'inter-connexion, que ce soit dans le sens des entrées et/ou dans celui des sorties, il s'agit là de filtrage. Pour le mettre en œuvre, ou plutôt le faire faire, l'opérateur d'un réseau utilise un ou des éléments techniques qui sont entre un réseau qu'il administre et un ou plusieurs autres. Le(s) "élément(s) technique(s)" sera/seront nommé(s) ici proxy(s) filtrant(s).

Tout filtre logique a besoin de facteur(s), pour déterminer s'il doit laisser passer ou non. Informatiquement cela se matérialise par un algorithme et un jeu de données si le filtre est paramétrable. Dans le cadre du réseau, un proxy filtrant écoute une interface (qui va lui fournir des éléments à filtrer) et émet sur une interface (ce qui a passé le filtre avec succès), l'interface de réception et d'émission pouvant être la même.

Pour filtrer un réseau informatique, on peut s'appuyer sur 2 grands types de données : les métadonnées et les données applicatives. Les métadonnées permettent techniquement de communiquer et sont bien souvent étrangères à un usager lambda, c'est une sorte de mal nécessaire. Les données applicatives sont inhérentes à la logique métier de l'application, donc techniquement les plus abstraites et qui peuvent informer plus précisément sur l'usage du réseau.

2.3.2 Proxy filtrant se basant sur les "métadonnées"

Ce que l'on a nommé métadonnées est très souvent codé de la même manière. Cela se traduit par l'omni-présent TCP/IP et UDP/IP dans une moindre mesure que le premier. Faire un filtre sur IP, UDP, et/ou TCP va toucher à peu près toutes les communications réseaux dans la vaste majorité des réseaux, et le tout simplement techniquement. Les filtres s'appuyant sur des propriétés de ces protocoles (notamment l'adresse source et destination pour IP, ainsi que le numéro de port pour UDP et TCP) sont donc légion. Puisque les données en question sont techniques, il y a peu de problèmes de vie privée à les manipuler et ces derniers ne sont pas compréhensibles intuitivement (pour beaucoup de personnes), il y a donc un consensus sur la proportionnalité de leur usage et par extension de la légalité de ce dernier.

Un filtre basé sur les métadonnées est efficace pour protéger les serveurs derrière le proxy et restreindre le béotien en informatique. En effet, un individu ayant des connaissances en informatique pourra aisément contourner ces restrictions, particulièrement s'il a un accès avec des droits illimités à une machine derrière le proxy. Il lui suffit d'installer un serveur dans un réseau non restreint par le proxy et de le configurer comme une passerelle réseau qui s'appuie sur un protocole accepté par le proxy et aura pour rôle de désencapsuler les requêtes et encapsuler les réponses. Cela se fait couramment en TCP/IP avec le port 443

(utilisé usuellement pour le HTTPS, c'est-à-dire le web chiffré), en transportant d'autres protocoles dans la charge utile au lieu de transporter des ressources web, charge utile que le serveur émettra et qui encapsulera la réponse dans la charge utile de TCP (au-dessus d'IP) en ayant indiqué le port 443. Face à ce genre d'individu et si on le juge utile, il faut donc utiliser au moins une autre méthode (en théorie sans enfreindre la légalité).

2.3.3 Proxy filtrant se basant sur des statistiques

Des méthodes de filtrage plus élaborées que celles précédemment présentées s'appuient sur des statistiques. En effet, les paquets qu'un protocole engendre ont souvent une taille similaire. De plus, l'intervalle de temps entre 2 paquets d'un même protocole peut être significativement différent (en moyenne) de celui d'autres protocoles. Il est donc possible de faire du filtrage avec des statistiques, à condition d'avoir un référenciel statistique sur les protocoles, ou au moins sur les plus courants que l'opérateur du réseau souhaite filtrer.

En informatique, ce référentiel est un jeu de données. Les statistiques sur les protocoles n'ont, dans la vaste majorité, pas été pensés par celles et ceux qui les ont conçus. Il faut donc bien souvent analyser l'usage des protocoles, ou utiliser des raisonnements mathématiques théoriques, pour obtenir des statistiques. Il y a des bases statistiques déjà faites que les fournisseurs de proxy peuvent fournir. On peut aussi souhaiter en créer une, par exemple par économie si celles qui nous conviendraient sont jugées trop couteuses ou pour en avoir une personnalisée vis-à-vis de l'usage des protocoles sur un réseau précis. Si on choisit cette dernière possibilité, on peut par exemple s'appuyer sur des algorithmes d'apprentissage, qui apprennent une fois pour toutes ou qui apprennent toute leur "vie", et sont un type d'algorithme d'intelligence artificielle qui est en ce moment à la mode (c'est par exemple ce genre de technique qui a été utilisé par AlphaGo qui a battu un célèbre joueur de go) (mais il y a des raisons objectives à son succès contemporain que nous ne développerons pas ici).

Cette technique de filtrage peut être contournée. Pour cela, il faut émettre plus de paquets que nécessaire (ce qui est très rarement prévu par le protocole), en essayant de se rapprocher du modèle statistique pour un ou des protocoles autorisés. Mais cela implique qu'il faut avoir la connaissance du modèle ou faire des hypothèses sur celui-ci, et avoir suffisamment de bande passante pour le surplus de paquets.

2.3.4 Proxy filtrant se basant sur le DPI applicatif

La méthode la plus élaborée et complexe, mais également la plus couteuse à implémenter est probablement le DPI au niveau applicatif. C'est l'abréviation *Deep Package Inspection*, qui signifie en français inspection en profondeur de

paquet. Concrètement le DPI applicatif fait usage des protocoles applicatifs, ce qui est particulièrement intrusif (en terme de vie privée).

Puisque le DPI applicatif s'appuie sur le protocole applicatif, il faut qu'il le comprenne, ce qui nécessite d'avoir un algorithme de filtrage pour chaque protocole que l'on veut supporter, or il y en a tellement qu'il est impossible *in concreto* de les supporter exhaustivement, il faut donc se limiter à une partie. Cela peut par exemple servir à discriminer seulement certaines ressources d'un site web (en se basant sur l'entête HTTP) ou des emails qui contiendraient un ou plusieurs mot-clés (comme l'aurait fait la NSA).

Néanmoins cette méthode est de moins en moins efficace en pratique. En effet, pour être efficiente, il faut pouvoir lire en clair les échanges dans le protocole applicatif. Or ils sont de plus en plus chiffrés. Le chiffrement peut être réalisé dans le protocole, ce qui est une recommandation de l'IETF (*Internet Engineering Task Force*), l'organisme qui s'occupe de faire la standardisation des protocoles Internet, par exemple à travers les RFC 7258 et 7624. Il peut également être fait par encapsulation dans un protocole prévu à cet effet. Cela se fait couramment via un VPN (*Virtual Private Network*) (qui est commun dans les collectifs de productions se souciant de leurs données confidentielles et chez les individus aspirant à ne pas être surveillables facilement) ou un réseau d'anonymisation (comme le célèbre projet Tor).

2.4 Proxy tampon

2.4.1 Proxy pour le cache

Il y a des ressources disponibles via un réseau qui ne changent pas ou peu souvent. C'est notamment le cas de certains types de ressources web, un très bon exemple est les images (bien que l'on pourrait objecter qu'il y en a qui sont générées automatiquement et potentiellement différemment selon le visiteur, mais c'est un cas fort peu commun actuellement).

Plutôt que de redemander une ressource déjà obtenue ou déja générée, il peut être pertinent de la garder en mémoire (persistante ou temporaire). On appelle cela un système de cache. Cela nécessite bien entendu de la place dans la mémoire. De plus, pour que cela ait un intérêt, il faut que la vitesse d'accès de la mémoire en lecture pour une ressource soit supérieure au temps d'accès de la même ressource par le réseau ou au temps de génération de la ressource.

Les applications, qui sont susceptibles de redemander par le réseau ou regénérer une ressource, ont souvent un système de cache pour être plus efficaces (en temps, en bande passante, en temps de calcul, etc). L'exemple le plus commun est le navigateur web. En effet, il n'aura échappé à aucun · e internaute que les pages web d'un même site web se ressemblent souvent (même thème graphique, même

logo, etc). Puisque les navigateurs web représentent une part notoire (en terme d'usage) des applications utilisant Internet, il y a un intérêt non négligeable à ce qu'ils aient un système de cache, ce qui est le cas pour les plus connus du grand public (comme Mozilla Firefox et Google Chrome).

Chaque application qui y a un intérêt peut donc avoir son système de cache en local. Mais si plusieurs applications ont besoin d'une même ressource qui pourrait avoir un intérêt à être dans un cache, il est préférable de mutualiser le système de cache plutôt que d'en avoir plusieurs indépendants. De même, si plusieurs internautes d'un même réseau demandent ou pourraient demander une même ressource récupérable par ledit réseau, il y a un intérêt à avoir un système de cache commun. C'est précisément le but d'un proxy réseau pour le cache.

Si celui-ci est transparent, c'est-à-dire que les applications n'ont pas d'indication qu'il y a un système de cache, elles vont aussi nécessairement utiliser leurs systèmes de cache pour celles qui en ont un. Cela peut être perçu comme redondant, et c'est vrai (partiellement), cependant cela a l'avantage de moins solliciter le proxy réseau et de moins en dépendre. De plus, les informations qui sont chiffrées de bout-en-bout ne peuvent être déchiffrés par le proxy (du moins pas trivialement) et sont différentes (dans leurs formes chiffrées) pour chaque usager d'un système cryptographique asymétrique (pour celles-ci il est donc inutile de les mettre en cache dans un proxy).

Ce type de proxy est couramment utilisé dans les organisations, notamment pour le Web. Parmi les implémentations de ce type de proxy, on peut par exemple citer Squid et Varnish, qui sont connus et libres.

2.4.2 Proxy anonymisant

Les réseaux transportent des données. On peut souhaiter que certaines d'entre elles ne soient pas ou difficilement rattachables à une ou plusieurs de nos identités. C'est d'ailleurs un sujet de préoccupation croissante pour la population, notamment depuis les révélations d'Edward Snowden que des médias de masse ont fortement relayé (au début du moins). Il y a donc un intérêt à ce qu'il y ait des moyens pour préserver au moins en partie sa confidentialité (qui est nécessaire pour les libertés fondamentales) lors de l'usage de réseau.

Un de ces moyens est le proxy réseau anonymisant. Son rôle est de transformer les données techniques des échanges à anonymiser par les siennes. Cela implique qu'il va recevoir les requêtes à anonymiser, ainsi que leurs potentielles réponses. Un corollaire de ce fait est qu'il faut avoir confiance pour le respect de la vie privée dans le proxy, ou en tout plus que dans le ou les services que l'utilisateur ou utilisatrice veut faire usage en étant anonyme. De plus, puisqu'un proxy anonymisant s'occupe de données techniques, il ne s'occupe pas des autres données, et n'est donc pas suffisant dans bien des cas pour maximiser son anonymat tout en continuant à utiliser le réseau (on pense notamment au JavaScript fortement

utilisé sur le Web).

Dans le cas d'Internet, un proxy anonymisant va a minima changer l'adresse source des paquets à anonymiser par une des siennes, tout en conservant au moins un moyen de faire la translation inverse. C'est une opération très basique et courante qui a d'autres intérêts. Elle est connue sous les abréviations de NAT et PAT.

Un NAT (*Network Address Translation*) change l'adresse source par une des siennes et garde la correspondance. Cette méthode implique qu'il faut autant d'adresses pour l'élément faisant du NAT que d'adresses sources à anonymiser. Cela peut être problématique en IPv4, puisqu'il n'y a presque plus d'adresses publiques libres de ce type, mais ne l'est pas en IPv6 puisqu'il y a 2^{16}, soit 65536, fois plus d'adresses qu'en version 4. De plus, ce n'est pas très efficace pour anonymiser puisque l'adresse IP source est fixe lors d'une session, ce qui fait une donnée constante et unique pour chaque communication et cela peu importe le protocole au-dessus d'IP.

Un PAT (*Port Address Translation*) change le numéro de port au niveau de la couche transport (en pratique celui de TCP ou UDP) ainsi que l'adresse IP source par une des siennes et retient la correspondance. Cette technique ne nécessite qu'une adresse IP, ce qui est particulièrement intéressant avec IPv4 (et explique son usage courant dans les routeurs Internet notamment ceux des opérateurs commerciaux souvent nommés *box*). De plus, puisque chaque application utilise un numéro de port différent des autres applications, le proxy modifie le numéro de port différemment pour chaque application. Pour un besoin de vie privée, il est fortement préférable que la translation (du numéro de port) soit faite avec un algorithme qui inclut un aléa non déterministe, sinon il serait possible de prédire la translation à condition de connaitre l'algorithme de translation et de génération d'aléa déterministe (s'il y en a un). Pour maximiser l'anonymat avec cette technique, il est souhaitable qu'un maximum d'internautes utilisent le proxy, puisqu'une même addresse IP sera utilisée (indirectement) par plusieurs ordinateurs.

Faire usage d'au moins le NAT ou le PAT est nécessaire, mais il y a des raisons qui peuvent pousser à ne pas s'en contenter. Tout d'abord, on peut craindre les intermédiaires techniques pour lesquels on a besoin de leurs éléments techniques pour échanger avec le proxy. Pour ne pas à avoir besoin d'avoir confiance en eux pour le respect de la confidentialité, il faut qu'il soit possible de chiffrer la communication avec le proxy de bout-en-bout (ce qui se fait souvent via le protocole TLS). De plus, on peut se méfier du proxy. Pour remédier à cela, on peut faire passer ses télécommunications par plusieurs proxys successivement, ainsi il n'y a que le premier qui a la connaissance de la réelle adresse IP et le dernier qui connait le contenu en "clair" de la charge utile (qui peut être chiffré de bout-en-bout avec le service final) s'il y a eu un chiffrement de bout-en-bout pour chaque proxy.

Pour éviter de devoir faire une installation complexe pour utiliser les 2 techniques complémentaires du paragraphe précédent, il existe des solutions de type "clé en main". La plus connue et utilisée s'appelle Tor (pour *The Onion Router* puisque le chiffrement successif de proxy en proxy peut faire penser aux couches d'un oignon) qui est parfois confondu avec *Tor Browser* (un navigateur web avec des réglages favorables à la confidentialité et qui utilise le réseau d'anonymisation Tor). Il faut néanmoins noter que Tor ne traite (actuellement) que les flux TCP/IP (à cause de son usage de TLS qui se base sur TCP), qui sont de loin les plus courants, mais ce qui peut tout de même être problématique pour certains usages (comme la voix sur IP qui utilise souvent UDP).

2.4.3 Proxy contre le déni de service

Une des convivialités qu'a permis les réseaux de télécommunications est l'accès à des services 24 heures sur 24 et 7 jours sur 7, bref tout le temps (et sans travail humain hormis la conception et la mise en œuvre). Ce genre de service est opéré par un serveur (qui peut être matériel mais qui n'est présentement presque plus qu'exclusivement logiciel) qui tente de répondre aux requêtes qu'il reçoit. Comme un serveur humain, un serveur informatique a une limite dans sa capacité de traitement quantitative. Si elle est dépassée, soit le serveur ne répondra qu'à une partie des requêtes, soit il s'arrêtera de fonctionner jusqu'au redémarrage. On appelle ce phénomène un déni de service. Pour éviter ou au moins limiter le risque que cela arrive, on peut mettre un proxy avant le serveur qui va s'occuper d'absorber le potentiel surplus de charge.

Un proxy contre le déni de service est souvent le seul service réseau (en ne prenant pas en compte ceux dédiés à l'administration) sur une machine physique. Il en est ainsi pour qu'il ait le maximum de capacité d'absorption de charge. De plus, il est souvent dimensionné d'une manière supérieure au(x) service(s) qu'il est censé protéger du déni de service.

S'il y a un nombre de requêtes supérieur à ce que peut traiter un service, un proxy contre le déni de service a 2 solutions pour faire barrage. La première est de jeter les paquets surnuméraires. Cela a le mérite d'être simple et ne pas demander de ressource matériel. La seconde solution est de garder en cache les requêtes et les envoyer au serveur quand la charge sera strictement inférieure au maximum. Le maximum peut être soit défini en dur dans le programme ou dans un fichier de configuration, soit déduit par le proxy par exemple via un algorithme d'apprentissage. Bien entendu, ce cache nécessite de la mémoire, et de la rapidité puisque les usagers pourraient fuir un service qu'ils jugeraient trop lent. Puisqu'un client s'attend souvent à avoir une réponse en un temps qui ne dépasse pas un seuil maximal, il est généralement pertinent de ne garder en cache que temporairement les requêtes de trop, du moins pour la gestion dans l'urgence, puisqu'à long terme il peut être pertinent de tenter de comprendre pourquoi il y a eu plus de requêtes qu'habituellement (mais on utilisera pour cela une mémoire

moins rapide).

Il est possible d'utiliser les 2 solutions avec un même proxy. Un proxy peut passer d'une solution à l'autre en fonction d'événements. Par exemple, si le cache venait à être presque plein, il pourrait être pertinent de passer en mode "tout jeter", jusqu'à ce que la charge devienne gérable par le service. Un proxy peut également utiliser une solution pour certains types de requêtes ou en fonction de la source ou de la destination, et l'autre solution pour le reste des requêtes. Cette technique peut par exemple servir pour favoriser certains internautes et/ou services, mais cela peut être perçu comme une violation du principe d'égalité (en théorie une valeur fondamentale en France) et donc de la neutralité d'Internet.

Les attaques de dénis de service peuvent être massives qualitativement (comme cela a été le cas pour OVH en 2016 avec une attaque de plus de 1.5Tbps). La majorité des organisations n'ont pas les moyens techniques (et leurs préalables financiers) pour résister à de grosses attaques. Si elles fournissent des services reposant sur un ou quelques ordinateurs (contrairement à des services massivement distribués dont l'extrême est les services pair-à-pair), elles ne peuvent donc pas résister toutes seules. Elles doivent donc accepter d'être vulnérables à des attaques de dénis de service (dont les conséquences sont souvent fâcheuses, comme une perte de ventes ou d'image, mais elles sont très rarement critiques, puisqu'il s'agit juste de rendre indisponible un service, ce qui ne peut pas entrainer la perte de données et un redémarrage est en principe facile) ou utiliser un service (de proxy) d'une organisation qui a plus de moyens pour contrer ce type d'attaque. Une entreprise célèbre qui est spécialisée dans ce domaine est CloudFlare.

2.5 Proxy enregistreur

2.5.1 Introduction sur les proxys enregistreurs

Puisqu'un proxy réseau est un intermédiaire technique pour un client ou un serveur, il peut techniquement enregistrer les communications qui passent par lui ou une partie (de celles-ci). En effet, un proxy enregistreur peut agir comme une caméra de surveillance : tout enregistrer pour une ou des exploitations en temps réel ou postérieure(s). Cela peut avoir plusieurs intérêts.

Il faut noter que ce n'est pas forcément légal. De plus, la loi définit des durées de conservation maximale pour certains types de données. Elle impose également une durée minimale et obligatoire d'enregistrement de certaines informations. C'est par exemple le cas des données de connexion en France, mais cela est remis en cause par l'arrêt *Digital Rights Ireland* de la CJUE (*Cours de Justice de l'Union Européenne*) qui estime que ce genre de pratique est attentatoire aux libertés fondamentales quand cela est pratiqué sur les communications d'individu a priori innocents.

2.5.2 Proxy pour analyser le trafic

Le trafic réseau peut être révélateur d'usages abstraits (à comprendre ici du point de vue d'un · e internaute et non de celui d'un · e technicien · ne). Des raisons peuvent pousser à essayer de de les trouver et de les comprendre.

Une de celles-ci est la compréhension d'attaques faites via le réseau, voire la réduction au moins partielle de celles-ci. L'exploitation peut se faire en temps réel, mais cela nécessite une certaine puissance de calcul, et un faible temps pour prendre une décision (ne rien faire, émettre un avertissement, bloquer une partie du trafic, etc), puisque dans le cas contraire cela ralentirait significativement le réseau et serait perçu comme une gène par certaines personnes l'utilisant. Les enregistrements peuvent aussi être analysés juste sur intervention humaine (par exemple pour une enquête), mais il faut pour cela les avoir préalablement stockés et donc avoir une infrastructure de stockage adéquate avec de préférence une autre pour la réplication des données. Dans ce cas, des outils d'analyse réseau et de filtrage réseau non automatiques sont utilisés, comme le célèbre Wireshark (qui dispose d'une interface graphique), et/ou des bibliothèques de programmation dédiées à l'analyse réseau, comme Scapy pour le langage Python.

Certains États pratiquent la surveillance de masse pour diverses raisons (lutte contre le terrorisme et le crime organisé, protection des intérêts économiques, maintien de la paix publique, concurrence économique, prévention de la prolifération des armes de destruction massive, etc). Pour cela, ils utilisent techniquement des proxys. On peut faire remarquer que la Loi Renseignement de 2015 a poussé à de vives "discussions" sur l'usage de proxys pour cela, qui ont couramment été nommés "boites noires". Dans ce genre d'usage et quand il s'agit d'individus, le proxy doit se baser sur des signaux faibles (donc peu fiables) (qui sont censés être révélateurs d'un potentiel danger vis-à-vis d'un ou plusieurs modèles de menaces) pour conserver des informations, puisqu'il serait bien trop couteux de tout sauvegarder et également trop long de faire une analyse plus poussée sur beaucoup de données.

Il faut noter qu'un proxy pour analyser le trafic peut servir pour plusieurs raisons et avoir divers rôles. Un exemple courant et libre de ce type de proxy est le logiciel Snort. Celui-ci peut entre autre enregistrer le trafic totalement ou partiellement (en obéissant à des règles programmables) (d'où le logo d'un cochon avec un gros nez renifleur) et détecter des attaques courantes (comme la recherche de ports ouverts) en prévenant via une alerte quand c'est le cas.

2.5.3 Proxy pour rejouer une session

En programmation, il est recommandé de faire des tests unitaires. Le but est d'avoir du code stable (ce qui revient à vouloir le moins de bogues possibles) et de ne pas avoir de régression (c'est-à-dire qu'une nouvelle version ne doit pas entrai-

ner le mauvais fonctionnement de ce qui fonctionnait bien avant). Concrètement chaque test unitaire appelle une partie du code avec potentiellement un ou des paramètres et vérifie que le résultat correspond à ce qui était attendu.

On peut appliquer ce principe pour vérifier le code d'une application réseau. Cependant cela ne vérifie pas le fonctionnement du logiciel tel qu'il va être utilisé *in concreto*. De plus, pour tester concrètement une application réseau, il faut lui envoyer un flux réseau et potentiellement répondre adéquatement à ce qu'elle enverra.

Il est possible de faire des tests unitaires réseaux, en démarrant l'application puis en comparant ce que l'on attendait d'elle et ce qu'elle a envoyé ou pas comme réponses à des stimuli (c'est-à-dire des requêtes par le réseau) qui lui ont été engendrés préalablement. On peut humainement créer les stimuli (via la programmation) et déterminer les réponses attendues. Dans le cadre de tests de non régression, on peut aussi utiliser l'application en enregistrant les informations réseaux qui lui sont liées (celles qui lui ont été envoyées et celles qu'elle a envoyé), pour rejouer ultérieurement la séquence des envois et comparer avec la séquence des réponses obtenues précédemment.

L'enregistrement des séquences réseaux, que l'on peut regrouper sous l'appellation de session réseau, est faite dans la majorité des cas par un proxy réseau. En effet, il est possible d'enregistrer la session au niveau du client réseau pour tester un serveur réseau et donc de se passer d'un proxy (comme le fait HPE TruClient dans ses versions *Light* et *Standalone*). Parmi les proxys réseaux enregistreurs, il y a par exemple *BlazeMeter Proxy Recorder*, *NeoLoad Recorder*, et celui de CLIF.

Le proxy s'occupe uniquement de l'enregistrement de la session et de l'export de celle-ci dans un format compréhensible (souvent en se basant sur XML, JSON, ou YAML). Ce format devra être interprétable par un logiciel capable de rejouer la session. Pour cela, on utilise souvent un logiciel de test de charge, comme Apache JMeter, Gatling, ou CLIF.

3 Partie pédagogique

3.1 Introduction

L'école a un rôle de transmission des savoirs sur l'environnement, au sens large, dans lequel baigne la société, pour que les individus puissent avoir un esprit critique et soient aptes à produire en son sein des valeurs d'usage. Les moyens d'accès au savoir et de communication sont des éléments cruciaux d'une société humaine, or ceux-ci sont pour une bonne part numériques dans le contexte actuel, notamment au travers du réseau Internet.

Nous allons présenter des moyens pour appréhender des technologies Internet, dans le cadre de l'école républicaine française, en prenant en compte le niveau des élèves et en s'appuyant sur des programmes de l'Éducation Nationale pour la classe terminale.

3.2 Références

— Baccalauréat en série STI2D (sciences et technologies de l'industrie et du développement durable)
  — Bulletin officiel spécial numéro 3 du 17 mars 2011
    "Enseignements technologiques (transversaux et spécifiques des spécialités architecture et construction, énergies et environnement, innovation technologique et éco-conception, systèmes d'information et numérique) du cycle terminal de la série STI2D"
    `http://www.education.gouv.fr/cid55415/mene1104262a.html`

— Baccalauréat en série scientifique de spécialité d'informatique et sciences du numérique
  — Bulletin officiel spécial numéro 8 du 13 octobre 2011
    "Enseignement de spécialité d'informatique et sciences du numérique de la série scientifique - classe terminale"
    `http://www.education.gouv.fr/pid25535/bulletin_officiel.html?cid_bo=57572`

3.3 Réalisation et compréhension d'un serveur web minimaliste

Situation professionnelle : Vous êtes en recherche de stage, et êtes acceptés pour passer un entretien technique. La personne en charge du recrutement vous

demande d'expliquer le fonctionnement d'un serveur web puis d'en créer un qui soit minimaliste, afin d'évaluer votre capacité à réaliser un proxy web (qui utilise les mêmes technologies) qu'elle aimerait faire réaliser via un stage.

Pré-requis :
— savoir interagir avec un ordinateur
— savoir programmer dans un langage de programmation

3.3.1 Séances

Sauf précision, les travaux sont effectués en binôme ou trinôme s'il y a un nombre impair d'éléves. Les séances sont prévues pour au maximum 20 élèves simultanés, donc probablement en demi-classe dans la majorité des écoles actuelles.

Durée totale : 14 heures

1. **Analyse du protocole IP**
   — Séance 1 (de 3h) : Codage de nombres entiers positifs en binaire, décimal et hexadécimal sur papier

     Découverte par la pratique des 3 bases les plus courantes en informatique pour représenter les nombres entiers positifs (binaire, décimal et hexadécimal). Pour cela, les élèves seront amenés à faire des conversions entre les 3 bases sur papier pour appréhender concrètement ces notions.

     Enseignement développé pour le baccalauréat STI2D :
     — Dans la partie 3.1.4 ("Traitement de l'information")
       — "Codage (binaire, hexadécimal, ASCII) et transcodage de l'information, compression, correction"

     Enseignements développés pour le baccalauréat de série scientifique de spécialité d'informatique et sciences du numérique :
     — Dans la partie 4.1 ("Représentation de l'information")
       — Savoir représenter un nombre binaire
       — Coder un nombre au travers d'un code standard

   — Séance 2 (de 2h) : Étude d'IP en lui-même sur ordinateur

     Dans un premier temps, visualisation de paquets IP avec un outil informatique (comme Wireshark) qui donne une représentation humaine. Le but est notamment d'identifier l'adresse source et l'adresse destination, et de comprendre leur utilité, voire les détournements possibles. Dans un second temps, décodage sur papier d'un ou plusieurs paquets IP en hexadécimale, avec l'aide d'une documentation papier en français.

Enseignement développé pour le baccalauréat STI2D :
— 3.2.4 “Transmission de l’information, réseaux et Internet”

Enseignements développés pour le baccalauréat de série scientifique de spécialité d’informatique et sciences du numérique :
— Dans la partie 4.4 (“Architectures matérielles”)
  — Adressage sur un réseau
  — Analyser le trafic (trames) sur un réseau

2. **Analyse du protocole TCP**
   — Séance 3 (de 2h) : Étude de TCP sur ordinateur

Dans un premier temps, visualisation d’échanges TCP avec un outil informatique (comme Wireshark) qui fournit une représentation humaine et graphique. Le but est de comprendre l’utilité de ce protocole, ainsi que son fonctionnement et donc les différents scénarios d’échanges. Dans un second temps, formalisation des scénarios possibles sous forme de diagrammes de flux, sur papier ou avec un logiciel adapté pour représenter des diagrammes (comme Dia).

Compétence développée pour le baccalauréat STI2D :
— CO7.sin3. “Exprimer le principe de fonctionnement d’un système à partir des diagrammes SysML pertinents Repérer les constituants de la chaîne d’énergie et d’information”

Enseignements développés pour le baccalauréat STI2D :
— Dans la partie 2.2.1 (“Représentation du réel”)
  — “Croquis (design produit, architecture)”

— Dans la partie 2.2.2 (“Représentations symboliques”)
  — “Représentations associées au codage de l’information : variables, encapsulation des données”

— Dans la partie 3.2.4 (“Transmission de l’information, réseaux et Internet”)
  — “Modèles en couche des réseaux, protocoles et encapsulation des données”

Enseignements développés pour le baccalauréat de série scientifique de spécialité d’informatique et sciences du numérique :
— Dans la partie 4.4 (“Architectures matérielles”)
  — Analyser le trafic (trames) sur un réseau
  — Mettre ainsi en évidence la notion de protocole

3. **Analyse du protocole HTTP**
   — Séance 4 (de 3h) : Compréhension du protocole HTTP 1

   Dans un premier temps, visualisation de messages HTTP avec un outil informatique (comme Wireshark) qui fournit une représentation humaine et graphique. Cela permettra aux élèves d'avoir une compréhension basique du protocole : demande d'une ressource, puis réponse avec un code de retour et contenu à afficher sur le client web. De plus, cela fournit un exemple de modèle client/serveur, ce qui permettra donc d'initier au concept. Ensuite, on propose de s'intéresser à la manière dont est encodé un en-tête HTTP. Ce sera une occasion d'introduire à l'ASCII (*American Standard Code for Information Interchange*). En s'appuyant sur les savoirs fraichement découverts, il sera demandé aux élèves de convertir en ASCII des en-têtes de requêtes HTTP 1 (courtes) en hexadécimal, ensuite d'y répondre en texte brut (encore une fois on se contentera d'un en-tête court), et faire la réponse en hexadécimal correspondante.
   S'il reste au moins 15 minutes, cela pourrait être l'occasion d'engager une réflexion collective sur les enjeux de vie privée de ce protocole. Une séance de 1h à 2h pourrait être ajoutée pour découvrir les éléments les plus importants de la RFC 1945 (qui définit le protocole HTTP 1.0), ce qui permettrait de mettre les élèves en contact avec un document technique et les faire pratiquer l'anglais.

   Compétence développée pour le baccalauréat STI2D :
   — CO7.sin1. "Décoder la notice technique d'un système, vérifier la conformité du fonctionnement"

   Enseignements développés pour le baccalauréat STI2D :
   — Dans la partie 3.1.4 ("Traitement de l'information")
     — "Codage (binaire, hexadécimal, ASCII)"

   Enseignements développés pour le baccalauréat de série scientifique de spécialité d'informatique et sciences du numérique :
   — Dans la partie 4.1 ("Représentation de l'information")
     — Coder un caractère au travers d'un code standard

   — Dans la partie 4.4 ("Architectures matérielles")
     — Analyser le trafic (trames) sur un réseau

4. **Réalisation d'un serveur HTTP très basique**
   — Séance 5 (de 3h) : Réalisation d'un serveur HTTP pour les requêtes GET et mono-utilisateur

Pré-requis : savoir programmer dans un langage de programmation (notion de variable, de fonction, etc) et manipuler les chaines de caractères dans celui-ci
L'objectif est de réaliser un serveur web minimaliste. Pour que ce soit possible en quelques heures et avec une compréhension basique de HTTP 1, il sera attendu uniquement que le serveur traite les requêtes GET, sans gestion des noms de domaine, et en ne traitant qu'une requête à la fois et ignorant toutes celles reçues pendant le traitement d'une requête.

Compétences développées pour le baccalauréat STI2D :
- Dans la partie 2.3.6 ("Comportements informationnels des systèmes")
  - "Modèles algorithmiques : structures algorithmiques élémentaires (boucles, conditions, transitions conditionnelles), variables"

- Dans la partie 3.1 ("Réalisation d'un prototype")
  - "Programmation de l'interface de communication"

Objectif et compétence de la spécialité systèmes d'information et numérique du baccalauréat STI2D :
- Dans la partie 2.1 ("Conception fonctionnelle d'un système local")
  - "Traitement programmé et composants programmables"

Enseignement développé pour le baccalauréat de série scientifique de spécialité d'informatique et sciences du numérique :
- Dans la partie 4.2 ("Algorithmique")
  - Concevoir et programmer un algorithme

5. **Réflexion sur l'accès au savoir et sur les privilèges d'exploitation**
   - Séance 6 (de 1h) : Débat sur l'ouverture et la fermeture des technologies

Les technologies vues dans les séances précédentes sont "ouvertes" : descriptif librement accessible, pas de redevance, pas de brevet, etc. Néanmoins un système de concurrence, comme le capitalisme, est enclin à mettre des barrières de "protection" (on pourrait développer sur la protection de quoi et de qui, mais ce n'est pas en phase avec l'objet de ce dossier). Cela explique qu'il y ait un nombre non négligeable de technologies "fermées", puisque dans le cadre actuel la fermeture peut être un avantage. Cette séance consiste en un débat sur cette thématique sans orientation particulière a priori. Cela pourra être une occasion d'aborder différents sujets non informatiques : le droit d'auteur et les licences, les brevets, les modèles économiques en fonction du modèle d'ouverture/fermeture que ce soit à une échelle micro ou

macro, etc.

Enseignement technologique commun du baccalauréat STI2D :
- Dans la partie 1.1.1 (“Paramètres de la compétitivité”)
  - “Recherche de solutions techniques (brevets) et créativité, stratégie de propriété industrielle (protection du nom, du design et de l’aspect”

Enseignement développé pour le baccalauréat de série scientifique :
- Dans la partie 4.1 (“Représentation de l’information”)
  - Droit sur l’immatériel

Compétence développée sur plusieurs séances pour le baccalauréat STI2D :
- Séances 2 à 4 : CO4.1. “Identifier et caractériser les fonctions et les constituants d’un système ainsi que ses entrées/sorties”

Enseignements développés sur plusieurs séances pour le baccalauréat STI2D :
- Séances 2 à 3 :
  - Dans la partie 2.2 (“Architecture fonctionnelle d’un système communicant”)
    - “Modèles en couche des réseaux, protocoles et encapsulation des données”

- Séances 3 à 5 :
  - Dans la partie 2.2 (“Architecture fonctionnelle d’un système communicant”)
    - “Architecture client/serveur”

- Séances 4 et 5 :
  - Dans la partie 3.2.4 (“Transmission de l’information, réseaux et Internet”)
    - “Architecture client/serveur : protocoles FTP et HTTP”

  - Dans la partie 2.2 (“Architecture fonctionnelle d’un système communicant”)
    - “Architecture client/serveur”

3.4 Analyse d'une attaque par déni de service et mise en place d'une contre-mesure

Situation professionnelle : Le service informatique a subi une attaque. Il vous est demandé d'enquêter sur son origine et sur ce qui a failli. Ensuite vous devrez proposer une solution afin de réduire le risque qu'une attaque de ce type ait un impact néfaste sur l'infrastructure informatique.

Pré-requis :
— savoir interagir avec un ordinateur
— savoir programmer dans un langage de programmation

3.4.1 Séances

1. **Capacité théorique du serveur**
   — Séance 1 (de 1h) : Identification des causes matérielles et logicielles d'un déni de service

   Tout d'abord, les élèves devront identifier les différents éléments nécessaires d'un serveur qui pourraient être saturés via la réception et le traitement de requêtes réseaux, ainsi que via l'envoi de réponses appropriées (processeur(s), RAM, bande passante, nombre de connexions simultanées autorisées par le système d'exploitation, etc). Ensuite, il y aura une phase de mise en commun, qui s'accompagnera d'une réflexion sur les hypothèses non pertinentes et leurs éliminations. Cette séance peut être faite sans ordinateur, mais il est aussi possible de faire avec, ce qui pourrait permettre d'introduire par la pratique le travail collaboratif et des mécanismes de modération qui lui sont liés (mais il faudrait pour cela rajouter une demi-heure à la séance).

   Enseignement développé pour le baccalauréat STI2D :
   — Dans la partie 3.2.4 ("Transmission de l'information, réseaux et internet")
      — "Organisations matérielle et logicielle d'un dispositif communicant : constituants et interfaçages"

   Enseignements développés pour le baccalauréat de série scientifique de spécialité d'informatique et sciences du numérique :
   — Dans la partie 4.4 ("Architectures matérielles")
      — Connaitre les éléments d'architecture
      — Expliquer le rôle des constituants d'un ordinateur

— Séance 2 (de 1h) : Calculs des différentes limites théoriques du serveur

L'enseignant · e choisit différentes causes plausibles de la non résistance du serveur face à l'attaque. Puis il ou elle choisit des valeurs réalistes moyennes des effets qu'une requête réseau provoque sur l'application (nombre d'unités de traitement du processeur pour traiter une requête, temps d'une unité de traitement dudit processeur, RAM nécessaire pour traiter une requête, etc). L'enseignant · e les soumet aux élèves, qui doivent calculer combien de requêtes chaque composant peut supporter théoriquement. Pour conclure, les élèves ont pour objectif de trouver le nombre maximal de requêtes que l'ensemble du système (c'est-à-dire le serveur) peut traiter (théoriquement), puis comparer l'écart avec des résultats empiriques qui leurs sont fournis.
Aucun matériel particulier n'est nécessaire. Les calculatrices peuvent être autorisées.

Compétence développée pour le baccalauréat STI2D :
— CO5.3. "Évaluer un écart entre le comportement du réel et le comportement du modèle en fonction des paramètres proposés"

Enseignement développé pour le baccalauréat de série scientifique de spécialité d'informatique et sciences du numérique :
— Dans la partie 4.4 ("Architectures matérielles")
  — Éléments d'architecture

2. **Analyse de l'attaque**
— Séance 3 (de 3h) : Recherche du moment d'arrêt du fonctionnement et ce qui l'a engendré

On suppose ici qu'un serveur a été victime d'une attaque de déni de service, et que l'on a pu récupérer les paquets réseaux que le serveur a reçu avant et pendant l'attaque. Ces derniers, créés pour l'occasion ou récupérés légalement sur Internet, sont donnés aux élèves sous la forme d'un fichier (pcap). Pour commencer, le contexte sera présenté. Puis, une brève introduction du logiciel graphique Wireshark sera faite aux élèves, puisqu'ils utiliseront cet outil pour le reste de cette séance. Ensuite, ils devront identifier le premier paquet qui n'a pas eu de réponse dans le but d'analyser les paquets qui l'ont précédé. En effet, c'est parmi ceux-ci qu'une ou des hypothèses sur le ou les causes du déni de service pourront être émises. Ils devront donc trouver au moins une hypothèse qui leur semble vraisemblable, et imaginer d'autres hypothèses qu'ils jugent non réalistes, en expliquant les faits et le raisonnement qui les ont amenés aux conclusions qu'ils auront proposées. Pour conclure, ils confronteront leurs analyses avec d'autres groupes, dans le but d'étayer

leurs arguments et la manière de les expliquer, ainsi que potentiellement remettre en cause certains arguments.

Compétence développée pour le baccalauréat STI2D :
— CO6.3. “Présenter et argumenter des démarches, des résultats, y compris dans une langue étrangère”

Enseignements développés pour le baccalauréat STI2D :
— Dans la partie 1.1.3 (“Compromis complexité-efficacité-coût”)
  — “Relation fonction/coût/besoin”
  — “Relation fonction/coût/réalisation”

Enseignement développé pour le baccalauréat de série scientifique de spécialité d’informatique et sciences du numérique :
— Dans la partie 4.4 (“Architectures matérielles”)
  — Analyser le trafic (trames) sur un réseau

3. **Proposition(s) de contre-mesure(s)**
— Séance 4 (de 1,5h) : Réflexion sur le ou les moyens qui auraient permis de limiter l’impact de l’attaque

La ou les causes de l’attaque sera/seront donné(s) aux élèves. À partir de celle(s)-ci, ils devront proposer des solutions qui auraient selon eux permis de limiter ou éliminer l’impact de l’attaque. Bien entendu, il faudra que des justifications soient mises en avant. Ils auront ensuite à évaluer les conditions de possibilité (facilité de mise en œuvre, cout financer, temps de travail humain, effets indésirables potentiels, etc) des différentes solutions proposées. Pour finir, l’organisation qui a subi l’attaque et ses moyens (financiers, techniques, humains, etc) sera indiqué aux élèves. Ils auront pour tâche de trier les solutions en fonction de ces nouvelles donnés, voire en éliminer certaines, le tout sans oublier de justifier.

Compétence développée pour le baccalauréat STI2D :
— CO3.2. “Évaluer la compétitivité d’un système d’un point de vue technique et économique”

Enseignement développé pour le baccalauréat de série scientifique de spécialité d’informatique et sciences du numérique :
— Dans la partie 4.2 (“Algorithmique”)
  — S’interroger sur l’efficacité d’un algorithme

4. **Mise en place d’une contre-mesure**

— Séance 5 (de 2,5h) : Implémentation d'un proxy contre les attaques de déni de service venant d'une seule IP

Pré-requis : savoir programmer dans un langage de programmation (notion de variable, de boucle, de fonction, etc) et manipuler une structure de type collection
On considérera que la cause retenue a été une attaque via une seule adresse IP qui aurait beaucoup de bande passante. C'est une hypothèse fort peu probable, mais c'est pour les besoins de l'exercice, qui ne peut pas consister à créer un système complexe, puisque cela nécessiterait trop de temps. On admettra que la solution retenue est un proxy réseau entre le serveur et Internet. Sa mission consiste à transmettre les paquets au serveur, à part ceux qui seraient dans une quantité supérieure à une valeur donnée dans un intervalle de temps donné et qui auraient la même adresse IP source. Concrètement le logiciel à implémenter écoute les paquets d'un port donné et en envoie sur le même port à une autre machine. Après l'écoute, un traitement devra être fait pour déterminer s'il faut envoyer le paquet ou pas, et il devra être expliqué selon ce que le binôme ou trinôme juge le plus adapté (texte, croquis, mélange, etc). Un algorithme simple pour cela est d'enregistrer les IP sources et l'horodatage auxquels ils ont été reçus, pour calculer à chaque fois combien de paquets ont été reçus avec la même IP source dans le laps de temps maximum défini, ce qui permet enfin de déterminer si on considère qu'il s'agit d'une attaque ou non, et donc de prendre la décision d'envoyer le paquet au serveur ou le supprimer. C'est bien entendu simpliste, gourmand pour le processeur et pour la RAM, mais ce n'était qu'un exemple qui a au moins le mérite d'être rapide à mettre en œuvre. Les solutions simples, performantes, paramétrables, et commentées seront valorisées. On est bien conscient que faire un programme (même petit) avec ces propriétés n'est pas chose aisée, si ce n'est impossible (si on les pousse à l'extrême), puisqu'en pratique ces propriétés peuvent être contradictoires.

Enseignement développé pour le baccalauréat STI2D :
— Dans la partie 2.3.6 ("Comportements informationnels des systèmes")
  — "Modèles algorithmiques : structures algorithmiques élémentaires (boucles, conditions, transitions conditionnelles), variables"

Objectifs et compétences de la spécialité systèmes d'information et numérique du baccalauréat STI2D :
— Dans la partie 1.2 ("Mise en œuvre d'un système")
  — "Compte rendu de la mise en œuvre d'un système, en utilisant un langage technique précis"

— Dans la partie 2.1 ("Conception fonctionnelle d'un système local")

— “Traitement programmé et composants programmables”

Enseignements développés pour le baccalauréat de série scientifique de spécialité d’informatique et sciences du numérique :
— Dans la partie 4.2 (“Algorithmique”)
  — Concevoir et programmer un algorithme
  — S’interroger sur l’efficacité d’un algorithme

5. **Interface graphique pour le proxy**
   — Séance 6 (de 3h) : Interface graphique pour visualiser et commander des actions du proxy

   Cette séance est optionnelle.
   Pour savoir ce que fait le proxy (réalisé lors de la précédente séance), il faut savoir programmer. Ce n’est pas un problème fonctionnel, mais cela peut être gênant. Nous proposons donc de le modifier pour qu’il affiche une interface graphique. Cela permettra aux élèves de pratiquer l’usage d’une bibliothèque et de comprendre des mécanismes d’une interface graphique. Pour que tous les élèves partent sur la même base, ils n’utiliseront pas le proxy qu’ils ont fait précédemment, mais un proxy qui sera fourni et adapté pour être facilement adaptable pour faire une interface graphique. Ils devront réaliser un afficheur des IP bloqués et un formulaire pour en bloquer une manuellement.

   Objectif et compétence de la spécialité systèmes d’information et numérique du baccalauréat STI2D :
   — Dans la partie 2.3 (“Modélisations et simulations”)
     — “Modèle de comportement : utilisation de librairies logicielles et paramétrage de caractéristiques”

6. **Réflexion sur la portée d’un programme**
   — Séance 7 (de 1h) : Réflexion sur le détournement de ce genre d’outil

   Cette séance est optionnelle et ne requiert pas d’avoir fait la précédente. Un outil est conçu dans un but précis par son ou ses concepteur(s) et conceptrice(s). Mais il se peut que l’outil puisse être adéquat pour réaliser une ou des tâche(s) non prévue(s). Les proxys réseaux ne font pas exception. Par exemple, un proxy pour bloquer les IP de machines attaquantes peut être utilisé pour faire de la censure politique en bloquant des services Internet que “le pouvoir” juge mauvais. D’une manière similaire, un proxy pour enregistrer dans le but de déceler des attaques informatiques peut servir à faire de la surveillance des individus ou des entreprises d’un autre pays via leurs communications électroniques.

De plus, puisqu'Internet est un réseau de réseaux inter-connectés, ce qui est fait sur l'un des réseaux peut impacter les autres. Le but de cette séance est dans un premier temps que les élèves proposent des détournements, puis qu'ils s'interrogent sous la forme d'un débat des conséquences potentielles du détournement d'un proxy réseau.

Enseignement développé pour le baccalauréat de série scientifique de spécialité d'informatique et sciences du numérique :
- Dans la partie 4.4 ("Architectures matérielles")
  - "Prendre conscience du caractère supranational des réseaux et des conséquences sociales, économiques et politiques qui en découlent"

Objectif et compétence sur plusieurs séances de la spécialité systèmes d'information et numérique du baccalauréat STI2D :
- Séances 3 et 4 :
  - Dans la partie 1.2 ("Mise en œuvre d'un système")
    - "Identification des dysfonctionnements et/ou description des solutions"

Objectif et compétence sur plusieurs séances de la spécialité systèmes d'information et numérique du baccalauréat STI2D :
- Séances 3 à 5 :
  - Dans la partie 2.1 ("Conception fonctionnelle d'un système local")
    - "Traitement d'une information numérique"

3.5 Journal de l'école via un site web

Les individus diffusent de plus en plus des contenus via Internet, notamment via le Web, en particulier les plus jeunes. Pour les faire réfléchir sur les enjeux liés, qui sont loin d'être exclusivement informatiques, il parait pertinent d'organiser un projet de site web public ou interne à l'école.

Par affinité, les élèves auraient à s'associer en groupes de 3 à 5. Chaque groupe devra choisir un des sujet(s) proposé(s) par l'enseignant · e lié(s) à l'école ou en proposer un que ce dernier ou cette dernière juge pertinent. Sur toute l'année, ou un semestre, les groupes auront pour mission de faire un site web lié à leur sujet, accompagné par un · e ou plusieurs enseignant · e · s.

Cette activité est pluridisciplinaire. Elle permettra par exemple d'aborder par la pratique les enseignements suivants :

— Les pages web en langage HTML (*HyperText Markup Language*)
— L'installation et la configuration d'un serveur web (comme Apache ou nginx)
— La gestion d'un projet
— L'organisation d'un groupe humain (mode d'organisation, gestion des conflits, etc)
— Le travail collaboratif
— La manière appropriée pour s'exprimer en publique via le Web
— La rédaction en français
— La rédaction en langue étrangère
— Le droit d'auteur (domaine public, licences propriétaires, licences libres, etc)

La durée de chaque séance et le nombre de séances seront à définir en équipe. Cela dépendra également des moyens alloués par l'école et du nombre d'enseignant · e · s de différentes disciplines impliqué · e · s dans cette démarche.

4 Conclusion

Les proxys réseaux sont un élément incontournable d'une large utilisation d'un réseau. C'est le cas d'Internet en 2017 et cela va a priori être amené à s'amplifier. Les proxys sont donc un élément important de l'informatique actuel et probablement du futur de l'informatique.

Nous avons pu étudier dans ce dossier comment fonctionnent certains éléments d'Internet et son application la plus connue (le Web), à partir de différentes problématiques traitées par les les proxys. Cela a été fait en montrant comment la diffusion de ces connaissances pouvait s'articuler avec l'enseignement, et en prenant en compte les directives de l'Éducation Nationale.

5 Licence(s)

Ce document est sous plusieurs licences (à l'exception des images et citations si elles sont sous droit d'auteur) :

— Licence Creative Commons BY-SA version 3.0 et 4.0
— Licence GNU de documentation libre (FDL) version 1.3
— Licence Art Libre version 1.3
— Licence publique générale GNU (GPL) version 2.0 et 3.0

Les licences citées autorisent l'utilisation pour tous les usages, la modification, et le partage. Cela s'applique pour tous moyens (Internet, clé USB, etc) et sa nature (centralisé, pair-à-pair, commercial, etc), ainsi qu'avec tous formats (code LaTeX, PDF, impression papier, etc), que cela soit une version originale ou modifiée. Les licences citées ne requièrent pas de demander ou dire que vous utilisez, modifiez et/ou partagez tout ou partie de ce document, il y a néanmoins une clause d'attribution et de partage à l'identique.

Vous pouvez choisir la licence qui vous convient le plus ou une licence compatible. Il en est de même pour les numéros de version. L'utilisation de plusieurs licences et plusieurs versions est faite pour maximiser la compatibilité juridique. En effet, certaines licences sont incompatibles (avec d'autres), c'est par exemple le cas de la GPL avec la FDL et la version 2.0 de la GPL avec sa version 3.0. Veuillez noter que ce sont toutes des licences avec *copyleft* (parfois traduit en "gauche d'auteur").

6 Divers

— Ce document a été rédigé en prenant en compte le caractère sexiste de l'usage actuellement majoritaire de la langue française. C'est par exemple pour cela que des points médians ont été utilisés. Voir "Guide pratique pour une communication publique sans stéréotype de sexe", édité en novembre 2015 par HCEfh (Haut Conseil à l'Egalité entre les femmes et les hommes), `http://www.haut-conseil-egalite.gouv.fr/IMG/pdf/hcefh__guide_pratique_com_sans_stereo-_vf-_2015_11_05.pdf`

— On a parfois utilisé des rectifications orthographiques du français de 1990 (que l'Éducation Nationale accepte), notamment pour les accents circonflexes.

— Les références jugées importantes sont écrites sous forme de texte, et sont donc présentes dans une version papier de ce document. D'autres non primordiales et nombreuses sont sous forme de liens hypertexte, ce qui implique qu'elles ne soient que dans des versions numériques du présent document.

— Ce document a été fait avec des logiciels libres et gratuits. Je tiens à remercier celles et ceux qui y ont contribué, notamment pour LaTeX (un système de composition de documents), GNU Emacs (un éditeur de texte), et Trisquel GNU/Linux (un système d'exploitation).