



L'anonymat dans le réseau Tor (The
Second-Generation Onion)

Brieuc Barthélemy

Année académique 2015-2016

Table des matières

1	Introduction	3
2	Présentation de Tor	3
2.1	Aperçu	3
2.2	Fonctionnement général	4
2.3	But du réseau	4
2.4	Ce que le réseau ne couvre pas	4
2.5	Installation du navigateur internet Tor	5
2.5.1	Installation par E-Mail	6
3	Fonctionnement	7
3.1	Serveurs d'annuaires	7
3.2	Les cellules de taille fixe	8
3.2.1	Les cellules de contrôle	8
3.2.2	Les cellules de relais	8
3.3	Initialisation du chemin	10
3.3.1	Les différentes clés utilisées	10
3.4	Création du circuit pas à pas	11
3.4.1	Échange de clé Diffie-Hellman	12
3.4.2	TLS - Transport Layer Security	13
3.5	Intégrité des flux de données dans Tor	13
3.6	Noeuds de sortie	14
4	Les points de Rendez-vous & Services cachés	15
5	Attaques et défenses	16
5.1	Les différents types d'attaques	16
5.1.1	Attaques passives	16
5.1.2	Attaques actives	16
5.1.3	Attaques des serveurs d'annuaires	17
5.1.4	Attaques des points de rendez vous	17
5.2	Attaques répertoriées	17
6	La gestion du réseau	18
6.1	Congestion	18
6.2	Limite de cadence de transmission	18
7	Le futur de Tor	19
8	Conclusion	19
A	Annexes	22

1 Introduction

L'anonymat sur le réseau Internet est une problématique importante pour la vie privée des utilisateurs. Mais il faut aussi prendre en compte les régimes dictatoriaux qui filtrent l'accès à certaines ressources sur le réseau mondial et qui empêchent leurs concitoyens d'atteindre ces sources d'informations. Tor est une des réponses à ce genre de problématique.

Ce document traitera des différentes réponses que le réseau Tor apporte pour rendre les utilisateurs anonymes. Il est basé en grande partie sur l'article de Roger Dingledine, Nick Mathewson et Paul Syverson : "Tor : The Second-Generation Onion Router" paru en 2004[1].

Nous aborderons l'architecture de communication, la façon dont il transmet les données mais aussi les serveurs d'annuaires et "les points de rendez-vous & les services cachés". Nous expliquerons enfin comment un client vérifie qu'il parle au bon interlocuteur, mais aussi les attaques qui peuvent être entreprises afin de casser l'anonymat des utilisateurs.

2 Présentation de Tor

2.1 Aperçu

"The Second-Generation Onion Router" (Tor) est un service de communication réseau informatique à faible latence sous licence libre [2].

Il permet de rendre anonyme une application réseau basée sur TCP comme par exemple un navigateur web. Pour ce faire, Tor utilise l'interface proxy Socket Secure (SOCKS) afin de permettre aux applications tierces de s'y connecter et donc de l'utiliser. Le but est d'éviter, en écoutant un seul point de passage comme un des routeurs, qu'on ne puisse relier le destinataire et sa source. Le projet Tor met à disposition des applications "prêtes à l'emploi" : le navigateur internet "Tor Browser" [3], un système de messagerie instantanée "Tor Messenger Beta" [4] ou encore une connexion "Secure Shell" (SSH) [5].

SOCKS est un protocole internet qui permet l'échange de paquets réseaux entre un client et un serveur à travers un serveur proxy. [6] Il est utilisé pour permettre à une connexion protégée derrière un firewall de communiquer de façon transparente et sécurisée sur le réseau internet. [7]



FIGURE 1 – Le logo du projet Tor.

2.2 Fonctionnement général

Pour bénéficier du réseau Tor et pour devenir anonyme, le client - Onion Proxy (OP) - doit suivre un processus. Prenons une personne qui désire se rendre anonymement sur une page web, cette dernière peut alors utiliser le navigateur Tor. L'application choisit un chemin composé de nœuds (des routeurs oignons ou "OR" en anglais) par lesquels il voyagera à travers le réseau. Le choix se fait via une liste reçue par les "répertoires de serveurs" -voir 3.1-. Chaque nœud ne connaît que son prédécesseur et le nœud suivant. Le contenu à destination des "OR" est crypté, chaque donnée à destination des "hops" (un hop est une portion entre une source et une destination, comme un routeur, une passerelle, un pont) est encapsulée dans une couche propre, comme un oignon, d'où le nom de "routage en oignon". Le client négocie ses clés de chiffrement avec chacun des routeurs séparément -voir 3.3-. La communication dans le réseau se fait via des cellules de taille fixe de deux types -voir 3.2-.

2.3 But du réseau

Tor cherche avant tout à frustrer l'attaquant qui tenterait de récupérer des informations sur une voie de communication venant ou allant vers un utilisateur. Mais pour y parvenir, il est important de prendre en compte certaines considérations. Tor cherche à être facile à installer et à être utilisé pour une raison d'anonymat. Il est moins simple de repérer un utilisateur parmi une foule que dans un petit groupe. Tor doit être déployé sur le réseau internet mondial et ne pas demander d'effort "extraordinaire". Il ne doit pas impliquer un surcout aux bénévoles qui sont prêts à aider, il ne doit pas demander des compétences techniques spécialisées (comme appliquer une modification au noyau du système d'exploitation (OS)). Il ne peut exiger des fournisseurs d'accès internet qu'ils endossent une responsabilité supplémentaire afin d'éviter que ceux-ci ne bloquent le réseau. Tor doit être aussi installable sur la plupart des plateformes ; on peut d'ailleurs l'utiliser sous Windows, Linux, Solaris, BSD-style Unix, MacOS X. [1]. Il doit être simple pour une bonne compréhension du design en cas d'ajout d'implémentation.

2.4 Ce que le réseau ne couvre pas

Pour une mise en place facile et une structure simple, les créateurs ont pris le choix de différer plusieurs solutions possibles car certains problèmes sont résolus dans d'autres systèmes ou parce qu'il n'y a pas encore de solution [1, 3. Non-goals].

Le peer-to-peer (P2P) n'est pas supporté. Alors que Tarzan [8] et MorphMix [9] ont pour but de décentraliser l'environnement P2P sur des milliers de serveurs

éphémères, certains pourraient être contrôlés par des personnes de l'autre camp.

Tor ne prétend pas être complètement sécurisé contre les attaques de bout à bout. Autrement dit : Si un personne a accès au point de départ et au point d'arrivée il est possible de faire correspondre un utilisateur visitant un site grâce à la fréquence d'envoi et de réception de données.

Le réseau ne peut résoudre tous les problèmes d'anonymat, il se focalise sur le transport des données. Si une autre application que le browser Tor est utilisé, il n'est pas négligeable d'utiliser un proxy - comme Privoxy [10] ou Anonymizer [11] - qui filtre les méta données entre le client et le service utilisé sur le réseau général (et donc pas un "rendez-vous" point de Tor). Ces derniers permettent de filtrer des données qui pourraient remonter la piste de l'utilisateur comme des cookies par exemple.

La stéganographie ne fait pas partie non plus du but du réseau. Cette technique permet de cacher un message dans un autre. Cependant un outil comme le proxy Stégotorus [12] peut être couplé avec Tor afin de cacher un message pour la destination finale. Ce message peut être du texte, de la video ou encore de l'audio et peut se cacher aussi dans ces mêmes médias. Un article détaillé datant de 2012 le présente [13].

2.5 Installation du navigateur internet Tor

L'application la plus connue de Tor est son navigateur web même si, le projet ne s'arrête pas à cela. D'ailleurs une liste sur le site officiel (<https://www.torproject.org/projects/projects.html.en>) définit les différents projets. Comme expliqué dans la partie "but du réseau" -voir 2.3-, Tor tente de simplifier l'effort fourni pour son utilisation et donc, il suit cette même logique pour son navigateur web. Basé sur le navigateur Firefox, Tor Browser ne demande pas de droit particulier pour son utilisation [1, Point 3.]. Il existait une extension pour le navigateur Firefox mais elle a été abandonnée en 2011 et retirée de addons.mozilla.org; les développeurs de l'extension ne pouvaient pas suivre la vitesse de sortie des nouvelles versions de Firefox [14] [15, Can I install other Firefox extensions?].

Bien que le navigateur Tor soit basé sous Firefox, et donc qu'il accepte les extensions de ce dernier, le projet Tor ne recommande pas d'installer de plugins qui pourraient casser l'anonymat, comme un contournement via un serveur proxy par exemple. Le projet Tor ne compte pas implémenter un bloqueur de publicité, il recommande même de ne pas le faire afin d'éviter l'altération des requêtes du navigateur. *Users are free to install these addons if they wish, but doing so is not recommended, as it will alter the browser request fingerprint* [16, Philosophy]. De plus cela pourrait nuire à des utilisateurs de Tor qui rémunèrent leur propre site avec de la publicité.

2.5.1 Installation par E-Mail

S'il est impossible pour un utilisateur de se rendre sur le site afin de télécharger Tor, l'équipe du projet a mis en place un système par e-mail afin de recevoir les informations dans la boîte de courrier électronique. [15, Your website is blocked in my country. How do I download Tor ?].

Lorsqu'un e-mail est envoyé à l'adresse "gettor@torproject.org" sans que la structure ne soit respectée, nous recevons une réponse qui explique la marche à suivre (Voir Figure 7).

Un envoi avec le mot "windows" dans le corps du message (Figure 7) fait réagir le robot Tor différemment, puisqu'il va envoyer une liste de trois services "cloud" (voir Figure 8). Cela diffère de la FAQ qui stipule qu'un seul service cloud est supporté : *Currently, the only cloud service supported is Dropbox.* [15, Your website is blocked in my country. How do I download Tor ?]. Cette liste de services permet de télécharger le navigateur d'une façon contournée. Il existe deux autres façons proposées par Tor pour télécharger le navigateur. La première est de la demander à un ami via clé USB (ou un autre espace de stockage portatif) et la seconde consiste à chercher dans le cache de Google afin de vérifier si une des copies fonctionne pour l'utilisateur [15, Your website is blocked in my country. How do I download Tor ?].

3 Fonctionnement

3.1 Serveurs d'annuaires

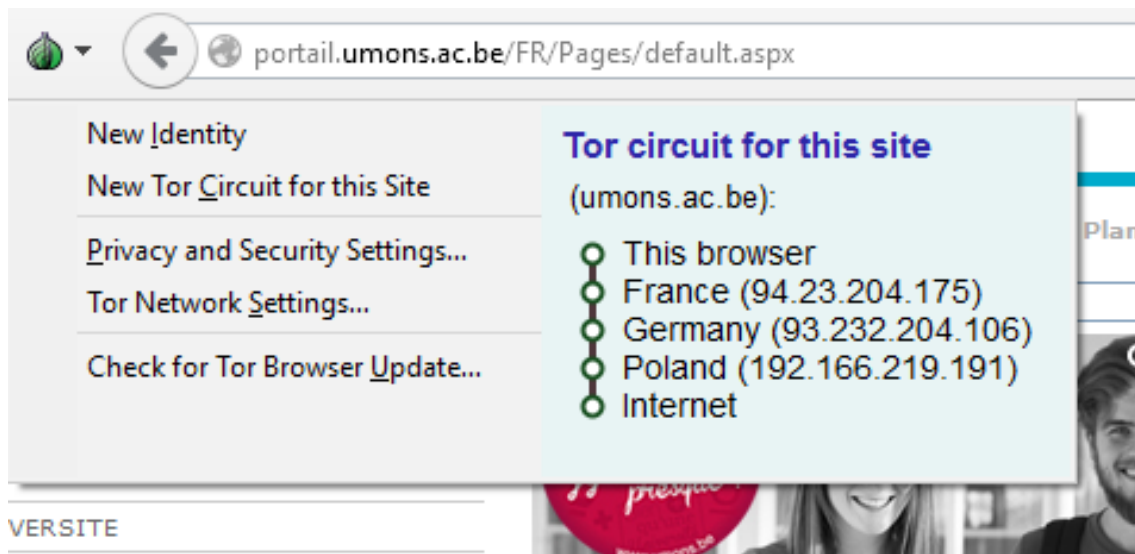


FIGURE 2 – Chemin jusqu’au site de l’UMons

Pour voyager dans le réseau, l’interlocuteur “A” doit connaître les différents ORs afin de déterminer préalablement son chemin jusqu’au nœud de sortie (voir Figure 2). Pour ce faire, il a besoin d’un annuaire de références, un “Directory Server” (DS), afin de recevoir la liste des ORs sûrs. Cet annuaire fournit une liste signée de tous les relais connus et dans cette liste se trouvent les certificats émis par les routeurs spécifiant leur clé, leur localisation et leurs politiques de sortie. Il existe plusieurs de ces annuaires qui ont la même information et donc la même liste d’ORs.

Il existe 10 serveurs d’annuaires, (pour le moment) qui sont encodés sur le navigateur Tor : <https://atlas.torproject.org/#search/flag:authority>. Ils sont au nombre de quatre aux États-Unis, et de six en Europe. La Figure 9 montre la page d’accueil web (sur le protocole HTTP, port 80) du serveur d’annuaire Faravahar qui se trouve aux États-Unis.

Pour qu’une personne malveillante puisse spécifier au client Tor un OR, elle devrait contrôler la majorité des serveurs d’annuaire.

Afin qu’un “Oignon router” soit dans un serveur d’annuaire, il doit être approuvé par un administrateur. Les ORs envoient périodiquement leur état aux serveurs d’annuaires et pour s’identifier, ils utilisent une clé fournie par un “Directory Server” -voir 3.3.1-. Les serveurs d’annuaires ne mettent pas dans leur liste les ORs non reconnus afin d’éviter qu’un adversaire crée plusieurs serveurs. Dans la Figure 6, le relais Tor créé dans le cadre de ce projet, mais toutefois permanent, a été listé dans

les serveurs d'annuaires deux heures après sa création.

3.2 Les cellules de taille fixe

Les routeurs à oignon communiquent entre-eux et les utilisateurs OP communiquent via une connexion TLS. Pour ce faire, ils se servent de cellules de données de 512 octets. Il en existe deux types qui seront décrits dans les points suivants. Cependant, leur structure générale est commune : elles sont composées d'un en-tête et d'une "payload". Les premiers 2 octets représentent le "CircID" qui définit de quel circuit se rapporte cette cellule. Il est nécessaire parce que plusieurs circuits peuvent être multiplexés dans une seule connexion TLS. Le "CircID" est spécifique entre chaque nœud OP vers OR et OR vers OR qu'il traverse.

3.2.1 Les cellules de contrôle

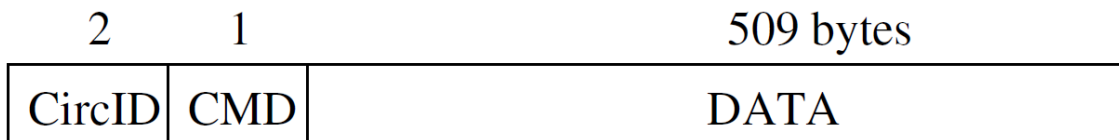


FIGURE 3 – Une cellule de contrôle

Les cellules de contrôle (Figure 3) sont toujours interprétées par le nœud qui les reçoit. Les commandes de ces cellules sont :

- Padding** Utilisé pour garder la connexion valide il s'agit de ce que l'on appelle le "keepalive".
- Create or Created** "Create" permet la création d'une liaison entre deux nœuds, alors que "Created" confirme cette connexion.
- Destroy** Cette commande est utilisée pour détruire le circuit.

3.2.2 Les cellules de relais

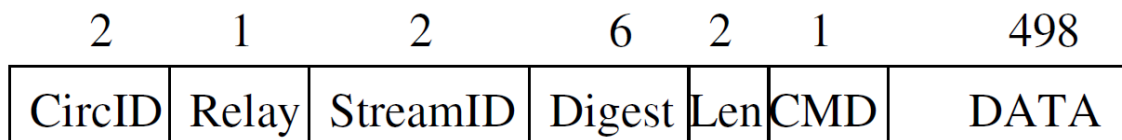


FIGURE 4 – Une cellule de relais

Lorsque que les connexions sont établies dans le réseau, l'OP peut envoyer des cellules relai. Afin de vérifier que la cellule soit valide, l'OR va décrypter cette

dernière et vérifier la correspondance avec la partie “Digest” de 6 octets. (Pour l’optimisation, les 2 premiers octets de la vérification d’intégrité sont à zéro. Donc, dans la plupart du temps, l’OR peut éviter de calculer le “hash”). Si c’est valide, il accepte la cellule et traite les informations qui s’y trouvent. Si ça ne l’est pas, l’OR fera suivre la cellule au prochain nœud. Si la fin de la chaîne reçoit une cellule de relais non reconnue, le circuit est démoli.

La cellule de relais est divisée en plusieurs parties :

- CircID** Comme la cellule de contrôle, elle détermine l’Id du circuit.
- Relay** Défini que c’est une cellule de relai.
- StreamID** Donne la référence du flux à laquelle cette cellule appartient.
- Digest** Comme expliqué plus haut, sert à contrôler l’intégrité.
- Len** Donne la longueur de la donnée contenue dans le champs DATA.
- CMD** Détermine la commande à utiliser pour cette cellule, nous verrons les différentes commandes un plus bas dans ce point.
- DATA** Ce sont les données envoyées dans la cellule.

Les différentes commandes des cellules de relais sont :

- Relay Data** Quand des données sont envoyées dans le flux du réseau.
- Relay Begin/Connected** “Begin” demande une connexion avec la destination ; “Connected” prévient la source que la connexion est établie.
- Relay Teardown** Pour fermer un flux qui ne répond plus.
- Relay End** Ferme une connexion proprement.
- Relay Truncate/Truncated** Pour détruire une partie du circuit ainsi que ses acquittements.
- Relay sendme** Utilisé pour les problèmes de congestion du réseau.
- Relay drop** Utilisé pour implémenter des faux profils. [1, Point 3 “Flexibility”]

3.3 Initialisation du chemin

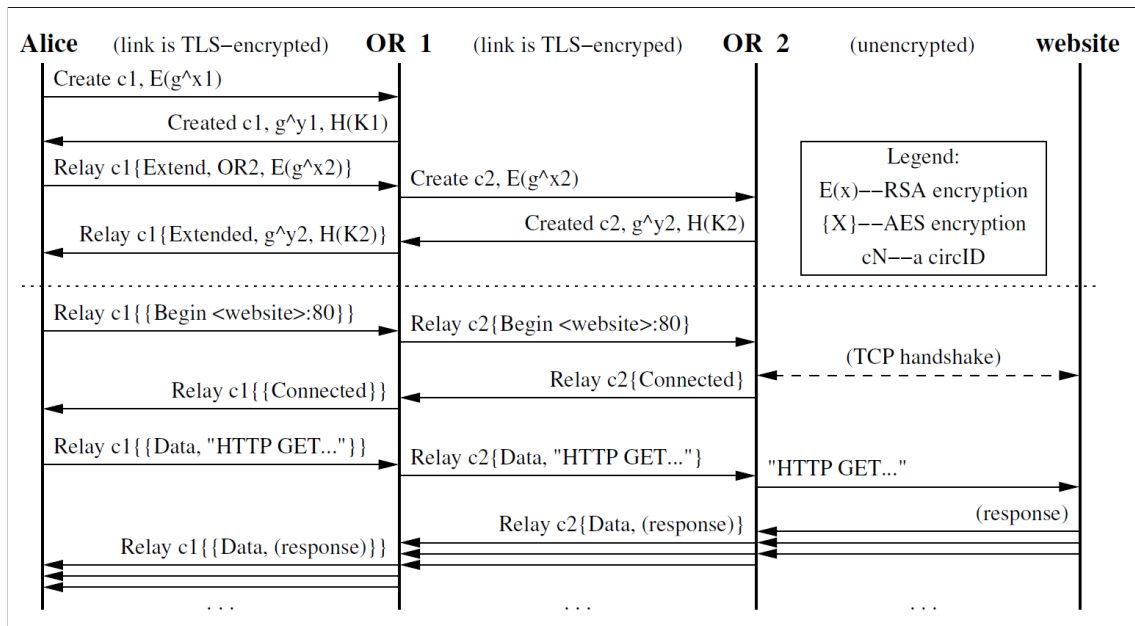


FIGURE 5 – Création d’un chemin par Alice pour lire une page web

Pour que l’utilisateur puisse se rendre sur une page web, il doit construire son chemin à travers le réseau Tor via des relais qu’il a reçu d’un serveur d’annuaires précédemment -voir 3.1-. Lorsque l’application a fait son choix, celle-ci va contacter chacun des “hops” de façon incrémentale pour créer le canal sécurisé. C’est à dire qu’elle va passer, si on se réfère à la Figure 5, par le “OR1” pour atteindre l’ “OR2” et passer par l’“OR2” pour interroger le site web. L’OP ne contactera pas directement les OR indépendamment, sauf le premier d’entre eux ; c’est lui son point d’entrée dans le réseau.

3.3.1 Les différentes clés utilisées

Afin de garantir l’authentification sur le réseau, la coordination - en vérifiant si le routeur auquel le système parle est bien celui qu’il prétend - et l’encryptage des données, le réseau utilise un jeu de clés avec leur particularité. [15, Tell me about all the keys Tor uses.]

Notons que toutes les connexions entre ORs ainsi qu’entre l’OP et le 1^{er} OR utilisent le protocole TLS -voir 3.4.2-.

La clé d’identité de longue durée permet la coordination. Celle-ci est utilisée pour signer les certificats TLS, son “routeur descriptor” [17] et pour vérifier via les serveurs d’annuaires que l’OR est bien celui qu’il prétend.

Une clé oignon de courte durée décrypte les requêtes du client. Ce dernier connaît toutes les clés oignons du chemin et chacun des relais connaissent évidemment leur propre clé. Chaque relais ne sait pas quelles sont les directives des autres. Cette clé permet à la création du circuit de vérifier si le routeur que le client a choisi est bien celui prévu. La cellule de commande “create” est cryptée avec la clé d’oignon du routeur. La clé oignon est modifiée une fois par semaine.

Des clés de lien de courtes durées permettent un calque supplémentaire de chiffrement, utilisant l’“Advanced Encryption Standard” (AES) avec une clé d’une longueur de 128bits, qui permet que seul le relais de sortie ne puisse lire les cellules.

Dans le document de référence [1, 4.The Tor Design], on peut lire un passage sur les clés éphémères. En posant la question directement à Robert Dingledine, ce dernier parle bien des clés utilisées pour l’encryptage (voir Figure 10 dans les annexes).

3.4 Création du circuit pas à pas

Voici une description pour la création du circuit : L’OP, Alice sur la Figure 5, envoie une cellule de contrôle de commande “CREATE” à l’“OR1” qui contient la première partie de l’échange de clés Diffie-Hellman -voir 3.4.1- chiffrée grâce à la “clé oignon”, qu’ils utilisent dans leur communication -voir 3.4.2-. Ensuite “OR1” répond à Alice par une cellule de contrôle “CREATED” avec l’autre partie de l’échange. Alice et “OR1” peuvent maintenant calculer la clé secrète chacun de leur côté et s’échanger des message sécurisés.

Pour continuer dans le réseau, Alice envoie maintenant une cellule de relais de commande “EXTEND” à “OR1”, afin de lui demander d’étendre le réseau, avec l’adresse de “OR2” et comme donnée, la première partie de l’échange de clés Diffie-Hellman entre elle et l’“OR2”. “OR1” va alors envoyer une cellule de contrôle de commande “CREATE” à l’ “OR2”. Ce dernier va alors répondre par une cellule de contrôle “CREATED” avec la deuxième partie de la clé à destination d’Alice, puisque c’est elle qui a initié l’échange de clé. C’est “OR1” qui recevra la réponse et qui la transmettra à Alice. Cette dernière peut maintenant aussi communiquer avec “OR2” de manière sécurisée. Dans cet exemple, “OR2” est le dernier OR du circuit et donc aussi celui de sortie.

Pour interroger le site web, Alice va envoyer une cellule relais de commande “BEGIN” pour initialiser la connexion avec la cible. Une fois que le nœud de sortie a créé la connexion avec le site web, il répond à l’OP avec une cellule relais de commande “RELAY CONNECTED”. L’“OP” est ainsi averti que la connexion est définie jusqu’au site web. Une cellule relais de commande “DATA” est envoyée jusqu’au routeur de sortie et ce dernier peut alors créer la connexion, ici HTTP, pour récupérer les informations et les envoyer à Alice d’une manière sécurisée.

Il y a un risque qu'une application qui utilise le client Tor demande une connexion DNS afin de connaître l'adresse IP de la destination avant d'entrer dans le réseau Tor. Elle va donc révéler sa destination au serveur DNS. Mozilla utilise le proxy "Provoxy" [10] et donc l'ordinateur de l'OP ne fait pas de résolution DNS. Le projet Tor préconise d'utiliser "Provoxy" partout où c'est possible.

La fermeture d'un flux Tor est analogue à celui d'une fermeture TCP. La fermeture se fait en deux étapes et via une cellule relais de commande "END". Puisque les messages sont cryptés, seulement le dernier nœud sait qu'il s'agit d'une commande de fermeture de flux. Si le flux est interrompu de façon anormale, c'est une cellule relais de commande "TEARDOWN" qui est alors envoyée. Les 2 étapes de fermeture permettent à Tor de supporter les applications basées TCP.

3.4.1 Échange de clé Diffie-Hellman

L'échange de clés "Diffie-Hellman", du nom de ses auteurs *Whitfield Diffie et Martin Hellman*, a pour but de créer une clé symétrique afin de communiquer de manière chiffrée entre deux acteurs (i.e. client/routeur).

Le principe est le suivant : Les deux acteurs Alice et Bob décident de crypter leur communication et d'échanger la clé de cryptage via cette méthode. Ils définissent ensemble un même nombre premier "p" et un nombre aléatoire "g" inférieur à "p". Chacun de leur côté, Alice et Bob choisissent en plus un nombre aléatoire "Ax" (resp. "Bx") qui sera utilisé dans le calcul d'échange.

Alice calcule la clé qu'elle enverra à son partenaire "Ay" : $Ay = g^{Ax} \pmod p$

Bob, de son côté fait la même chose avec By : $By = g^{Bx} \pmod p$

Ils échangent les résultats de Ay et By. Ensuite ils peuvent calculer la clé secrète commune :

Du côté d'Alice : $CleSecrete = By^{Ax} \pmod p$

Du côté de Bob : $CleSecrete = Ay^{Bx} \pmod p$

Faisons un exemple avec de petits nombres :

$$p = 17 \ \& \ g = 4$$

Alice :

nombre aléatoire : $Ax = 2$

$$Ay = 4^2 \pmod{17} = 16$$

Bob :

nombre aléatoire : $Bx = 3$

$$By = 4^3 \pmod{17} = 13$$

Ils échangent les résultats :

Alice calcule la clé secrète : $CleSecrete = 13^2 \bmod 17 = 16$

Bob calcule la clé secrète : $CleSecrete = 16^3 \bmod 17 = 16$

Le résultat, et donc la clé secrète est bien “16” des deux cotés, ils ont les informations adéquates pour s’échanger des informations grâce à une clé symétrique.

3.4.2 TLS - Transport Layer Security

La communication entre 2 noeuds dans le réseau Tor se fait par le protocole “Transport Layer Security” (TLS), une évolution d’SSL. La première version a été définie en janvier 1999 [18] et l’actuelle version 1.3 n’est pas encore définie puisque toujours en version “draft” mise encore à jour le 15 mai 2016[19]. TLS permet l’authentification du serveur consulté, la confidentialité des données échangées, l’intégrité de celles-ci et de manière optionnelle l’authentification du client. TLS est largement utilisé dans la navigation sur le web grâce au protocole HTTPS.

L’authentification est la première étape, elle permet de vérifier que le serveur à qui nous parlons est bien celui qu’il prétend être. Pour réaliser cela, le client va recevoir la clé publique du serveur, ses informations et sa signature numérique. Celle-ci doit être déchiffrée directement par le navigateur et s’il y arrive, il pourra alors envoyé une requête “Online Certificate Status Protocol” (OCSP) à l’autorité pour vérifier si le certificat du serveur est reconnu et toujours valide. Pour cette partie, Tor utilise l’algorithme de chiffrement RSA (Les initiales des auteurs : *Ronald Rivest, Adi Shamir et Leonard Adleman.*).

La confidentialité des données échangées est obtenue grâce à un chiffrement de clés symétriques. Dans le réseau Tor, l’algorithme utilisé est l’ “Advanced Encryption Standard” (AES) d’une longueur de 128 bits.

L’intégrité des données permet de vérifier si celles-ci n’ont pas été altérées pendant le transport grâce à une fonction de hachage. Tor utilise la version 1 de “Secure Hash Algorithm” (SHA).

3.5 Intégrité des flux de données dans Tor

Parce que Tor utilise TLS, un adversaire externe ne peut modifier les données. Le projet pourrait faire une vérification de l’intégrité à chaque hop, en incluant un hash ou une authentification chiffrée. Mais cela comporte des problèmes. D’abord, cela demanderait plus de données à chaque nœud. Ensuite la solution ne pourrait alors que vérifier le trafic qui viendrait d’Alice, les ORs ne seraient pas capable de reproduire des “hash” adaptés pour les hops intermédiaires. Troisièmement, Tor accepte qu’il ne couvre pas une attaque qui surveille le réseau de bout à bout, donc

utiliser une attaque qui marquerait les messages - en le modifiant - n'aiderait pas plus un attaquant.

3.6 Noeuds de sortie

L'abus de sortie - autrement dit quand la connexion sort du réseau Tor - est une vraie problématique. En effet, les personnes malveillantes peuvent commettre leurs méfaits sans que l'on puisse remonter facilement leur piste. Pour contrer cela, une politique de sortie a été mise en place, elle permet à l'administrateur d'un OR de définir les règles de ce dernier.

Open exit nodes Ceux ci sont les points de sortie du réseau. C'est à dire ceux qui vont permettre d'atteindre la destination. Il est important d'en avoir beaucoup, sinon il est plus simple pour un attaquant de surveiller le réseau de sortie. Le contrôle du trafic "End-to-end" est une faiblesse reconnue du réseau. Comme beaucoup d'OR le font déjà, il est possible de restreindre certains ports de sortie afin de limiter les services utilisés sur le réseau, et ainsi d'éviter les abus comme par exemple le Simple Message Transport Protocol (SMTP) permettant d'envoyer des e-mails qui est par défaut désactivé.

Middleman nodes Ils sont utilisés uniquement pour relayer le trafic dans le réseau. Cependant, il est important d'en avoir un nombre conséquent, afin de permettre un réseau robuste.

Private exit nodes Permet de se connecter à un hôte ou à un réseau privé.

Les auteurs préconisent d'utiliser un proxy pour nettoyer le trafic qui quitte le réseau, bien qu'à l'écriture du document ils n'aient pas encore été confrontés à un abus dans le réseau déployé. Un abus serait par exemple d'exploiter une faille bien connue dans un script.

4 Les points de Rendez-vous & Services cachés

Le réseau Tor permet aux utilisateurs de fournir des services TCP cachés, comme un serveur web par exemple, sans que le “surfeur” ne connaissent l’adresse IP du service. Ces services cachés ont pour but :

Le contrôle d’accès : le service a besoin d’une façon de filtrer le contenu qui arrive afin que les attaquants ne pourront pas inonder de requêtes le service.

La robustesse : le service doit garder un anonymat même si une défaillance d’un routeur oignon se produit, le système doit être capable de se répercuter sur un autre OR.

La résistance au marquage : un service malveillant, ne doit pas être capable de faire croire que le routeur “rendez-vous” est à l’origine du service illégal ou d’une mauvaise réputation.

Transparence : même s’il faut qu’un utilisateur emploie un programme spécial pour bénéficier des services cachés, il ne doit pas modifier son application.

L’idée est que si Alice doit se rendre sur le service de Bob - qu’elle connaîtrait parce qu’on lui a donné l’adresse - elle ne puisse pas le contacter directement. Elle va devoir passer par un point de rendez-vous (qui n’est autre qu’un OR) et lui donner le “rendez-vous cookie” qui permet à l’OR de reconnaître Bob. Alice va construire un flux anonyme et lui envoyer un message chiffré avec la clé publique de Bob et la première partie de l’échange de clé Diffie-Hellman. Si ce dernier veut parler à Alice il construira un circuit avec la deuxième partie de la clé DH et un hash de la clé de session qu’ils partagent.

Alice et Bob sont connectés et peuvent échanger des informations normalement.

Cela permet aussi de protéger le serveur caché des attaques par déni - Denial of Service (DoS) - puisque l’adversaire ne connaît pas l’adresse IP du serveur mis en place.

5 Attaques et défenses

5.1 Les différents types d'attaques

Il existe 4 catégories d'attaques : les attaques dites passives, les actives, les attaques ciblant les serveurs d'annuaires et contre les points de rendez-vous. L'article en détaille plusieurs ainsi que ce que l'adversaire peut en tirer comme informations utiles.

5.1.1 Attaques passives

les attaques passives sont les attaques "d'écoute". Celles-ci n'essaient pas de modifier les données mais font des correspondances dans le trafic. En voici quelques unes :

Observation du trafic de l'utilisateur : Observer le flux de connexion d'entrée et sortie d'un utilisateur ne révèle pas la destination ou les données. De plus, plusieurs applications peuvent envoyer et recevoir des informations en même temps, il faudrait un traitement ultérieur des données.

Observation du contenu : Le contenu coté utilisateur est crypté, cependant, celui du coté du destinataire peut ne pas l'être. Tor peut utiliser Privoxy ou un logiciel apparenté pour filtrer et anonymiser le contenu non crypté.

La distinction des options : Tor permet de gérer plusieurs options par l'utilisateur. Le problème étant que si les options sont spécialisées, l'utilisateur ne pourrait plus être dans la masse et sortira alors du lot. Ce qui pourrait nuire à son anonymat.

Emprunte du site web : Plutôt que de confirmer un timing ou un volume de données, l'attaquant crée une base de données des emprunts d'un site web, comme la taille ou un modèle d'accès d'un site web cible. Il peut alors ensuite confirmer une connexion de l'utilisateur en lisant la base de données qu'il aura construite.

5.1.2 Attaques actives

Les attaques actives sont plus agressives, elle essaient de modifier l'environnement du réseau. En voici quelques unes :

Clés compromises : Un attaquant qui apprend la session TLS pourrait voir les cellules de contrôle et les relais encryptés. Connaître une clé de session lui permettrait de déchiffrer un calque. Le changement de clé périodique permet d'éviter ce genre d'attaque.

Attaques "bavures" : Un utilisateur pourrait utiliser un serveur de sortie pour des actes répréhensibles. Son fournisseur d'accès au réseau internet pourrait le fermer pour des raisons légales. Il faut donc que chaque OR fasse attention à sa politique de sortie.

Faire fonctionner un (ou plusieurs) OR hostile : En plus d'être un observateur, un adversaire pourrait ajouter plusieurs ORs et persuader les serveurs d'annuaires qu'ils sont sûrs. Plus il en ajoute, plus il pourrait contrôler le passage des utilisateurs. Une autre façon pourrait être de donner une police de sortie alléchante et donc il pourrait récolter un maximum de trafic.

5.1.3 Attaques des serveurs d'annuaires

Les attaques contre les serveurs d'annuaires ont pour but de donner des informations hostiles aux clients du réseau. En voici quelques unes :

Destruction de serveurs d'annuaires : Si quelques serveurs disparaissent, les autres pourront toujours déterminer un répertoire valide. Cependant, si la moitié ne fonctionne plus, une intervention humaine sera nécessaire pour le client afin de décider s'il faut faire confiance au résultat de l'annuaire.

Subvertir une majorité des serveurs d'annuaires : Un adversaire qui contrôle plus d'une moitié des serveurs d'annuaires pourrait ajouter des OR non reconnus officiellement. Pour éviter ce genre d'ennui, les serveurs d'annuaires doivent chacun résister aux attaques.

Convaincre les annuaires qu'un OR qui ne fonctionne pas bien, fonctionne bien : Dans l'implémentation, les serveurs d'annuaires supposent qu'un OR fonctionne bien s'il peuvent démarrer une connexion TLS avec lui. L'OR d'un attaquant pourrait flouter ce test en ignorant les cellules. Les serveurs d'annuaires doivent activement tester les ORs en créant des circuits et des flux comme c'est demandé.

5.1.4 Attaques des points de rendez vous

Les attaques contre les points de rendez vous sont aussi une des façons d'attaquer le réseau :

Créer beaucoup de requêtes d'introduction : Un attaquant pourrait essayer de créer une attaque DoS sur le service de Bob en l'inondant de requêtes. Cependant, les points de "rendez-vous" peuvent ne pas répondre à un acquittement (Acknowledgment, "ACK") non reçu. Mais en plus le propriétaire du service caché peut filtrer le volume de demande ou demander des calculs pour chaque demande.

Compromettre un point d'introduction : Un attaquant qui contrôle le point d'introduction de Bob pourrait inonder de requêtes le service de Bob directement. Ce dernier pourrait toutefois détecter une submersion dans le circuit et alors le fermer.

5.2 Attaques répertoriées

Il existe une multitude d'articles, parlant d'attaques sur le réseau Tor. Qu'elles soient commises pas le "Federal Bureau of Investigation" des Etats Unis d'Amérique

(le FBI), la “National Security Agency” (NSA), ou encore par Éric Filiol [20], nous ne décrivons pas ces différentes méthodes dans ce document. Retenons qu’Éric Filiol répond à une interview sur le sujet dans le “journal dunet” [20]. Le réseau Tor quant à lui écrit une version qui diffère de cette dernière attaque : <https://blog.torproject.org/blog/rumors-tors-compromise-are-greatly-exaggerated>

Nous n’entrerons pas plus dans les détails de toutes les attaques citées sur le net, la liste est vaste et demande un travail d’investigation plus en profondeur pour filtrer les “on dit”.

6 La gestion du réseau

6.1 Congestion

La congestion est une problématique connue sur le réseau internet. Le protocole TCP se charge d’ailleurs de régler cela à son niveau. Dans Tor, le problème est qu’un nœud pourrait, s’il est trop sollicité, comme par exemple pour le transit d’un gros fichier, créer un goulot d’étranglement.

Tor met en place un principe pour contrôler le flux des cellules *DATA* qui sont les plus nombreuses sur le réseau. Deux niveaux de congestion sont pris en compte :

Au niveau du circuit, deux fenêtres sont utilisées pour vérifier l’utilisation de la bande passante : la fenêtre de “packaging”, qui est le nombre de cellules que l’OR est prêt à délivrer à l’OP, et la fenêtre de livraison (*delivery windows*), qui piste le nombre de cellule de relais prêtent à être délivrée au flux TCP (autrement dit en dehors du réseau). Les deux fenêtres ont un nombre défini lors de l’initialisation. Quand un paquet est emballé ou délivré, la fenêtre appropriée se décrémente ; quand il reçoit un *relay sendme* il incrémente ce même nombre. Une fois le numéro de la fenêtre à zéro, l’OR arrête de lire le flux et attend un *relay sendme* pour continuer.

Au niveau du flux, un mécanisme similaire se trouve au *niveau du circuit* (entre les ORs et les OPs) et utilise le *relay sendme* pour implémenter un contrôle bout à bout.

6.2 Limite de cadence de transmission

Les personnes volontaires (i.e. celles qui font fonctionner un OR) préfèrent que le réseau ne soit pas trop gourmand en bande passante. Dans la technique du “Seau à jetons” [21](utilisée par Tor), le flux s’écoule à une allure définie, cependant le réseau accepte un débordement ponctuel de la part d’une requête. C’est un paramètre que l’on retrouve dans la configuration d’un relais Tor (voir Figure 11).

7 Le futur de Tor

L'article énumère plusieurs points sur le futur de Tor. En voici une liste non exhaustive datant de l'article de référence [1] :

La classification des bandes passantes des nœuds dans le réseau. Tor suppose que toutes les bandes passantes sont équivalentes. Une gestion de cette partie pourrait éviter les goulots d'étranglement des flux.

Les récompenses pour les volontaires. Afin d'augmenter le nombre d'ORs et donc augmenter l'anonymat, le projet s'interroge sur la façon d'attirer plus de volontaires. Un piste serait d'utiliser des récompenses autres que le fait d'être plus anonyme - Puisque un OP qui est aussi un OR permet d'ajouter une couche d'anonymat - mais aussi comprendre pourquoi la plupart des utilisateurs n'utilisent pas un système anonyme.

Le déploiement d'un réseau plus large : En 2004, le projet avait assez d'utilisateurs pour évaluer les décisions de la structure de données choisies comme dans la robustesse du réseau. Depuis, le réseau s'est agrandi, le nombre d'ORs disponibles a atteint plus de 7000 au 22 Mai 2016 [22] alors que les serveurs d'annuaires, eux, sont maintenant au nombre de 10 -voir 3.1-.

8 Conclusion

Tor est un réseau informatique qui fournit aux utilisateurs d'application TCP via le protocole SOCKS un anonymat sur internet. C'est un réseau mature et largement utilisé depuis plusieurs années. Il ne cesse d'évoluer et les gestionnaires du projet communiquent fréquemment via le blog <https://blog.torproject.org/blog/>.

Tor utilise le réseau internet, ce qui permet aux utilisateurs et aux volontaires qui aident le réseau en devenant un routeur oignon de ne pas modifier le comportement ni le noyau de leur ordinateur. Pour naviguer sur le web, Tor Browser est la meilleure option et en utiliser un autre est une mauvaise idée [15]. Son installation est simple et il existe des solutions si le téléchargement n'est pas disponible directement via le site web.

Il apporte, grâce aux différents points de connexion, les OPs et ORs, une réponse concrète à l'anonymat sur internet. Il est conçu pour rendre impossible une identification d'utilisateur en n'écoutant qu'un point de passage. Tor utilise des concepts qui sont intégrés sur le réseau internet, comme le protocole TCP, TLS, mais aussi le chiffrement AES.

Toutefois, il est important de suivre les règles définies pour l'utilisation du projet Tor [23, Want Tor to really work?] afin d'être sûr de ne pas commettre un impair et casser son anonymat sur le réseau. Des attaques sur le réseaux sont toujours

possibles à différents niveaux et des solutions existent.

Références

- [1] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor : The second-generation onion router. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.
- [2] Tor Project. Tor project license, 2001-2016. <https://gitweb.torproject.org/tor.git/plain/LICENSE>.
- [3] Tor Project. Tor browser, 2001-2016. <https://www.torproject.org/projects/torbrowser.html.en>.
- [4] Tor Project. Tor messenger beta : Chat over tor, easily, 2015. <https://blog.torproject.org/blog/tor-messenger-beta-chat-over-tor-easily>.
- [5] vwochnik on howtoforge.com. Anonymous ssh sessions with tor, ? <https://www.howtoforge.com/anonymous-ssh-sessions-with-tor>.
- [6] Wikipedia Community. Socks. <https://en.wikipedia.org/wiki/SOCKS>.
- [7] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones. Socks protocol version 5, 1996. <https://tools.ietf.org/html/rfc1928>.
- [8] Michael J. Freedman and Robert Morris. Tarzan : A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS '02*, pages 193–206, New York, NY, USA, 2002. ACM.
- [9] Marc Rennhard and Bernhard Plattner. Practical anonymity for the masses with morphmix. In Ari Juels, editor, *Proceedings of Financial Cryptography (FC '04)*, pages 233–250. Springer-Verlag, LNCS 3110, February 2004.
- [10] List on https://www.privoxy.org/user_manual/copyright.html. Privoxy - home page, 2001-2016. <https://www.privoxy.org/>.
- [11] List on https://www.privoxy.org/user_manual/copyright.html. Anonymizer - home page, 1995 -2016. <https://www.anonymizer.com/>.
- [12] Nick Mathewson and George Kadianakis. Stegotorus - github, 1995 -2016. <https://sri-csl.github.io/stegotorus>.
- [13] Zachary Weinberg, Jeffrey Wang, Vinod Yegneswaran, Linda Briesemeister, Steven Cheung, Frank Wang, and Dan Boneh. StegoTorus : A camouflage proxy for the Tor anonymity system. In *Proceedings of the 19th ACM conference on Computer and Communications Security (CCS 2012)*, October 2012.

- [14] Tor Project. To toggle, or not to toggle : The end of tor-button, May 2011. <https://blog.torproject.org/blog/toggle-or-not-to-toggle-end-torbutton>.
- [15] Tor Project. Tor project faq. <https://www.torproject.org/docs/faq.html.en>.
- [16] Tor Project. The design and implementation of the tor browser. <https://www.torproject.org/projects/torbrowser/design>.
- [17] Tor Project. Stem docs, mirror mirror on the wall.
- [18] T. Dierks and C. Allen. *RFC 2246 : The TLS Protocol Version 1.0*. IETF RFC 2246, January 1999.
- [19] E. Rescorla. The transport layer security (tls) protocol version 1.3 - draft, 2016.
- [20] Éric Filiol. https://fr.wikipedia.org/wiki/%C3%89ric_Filiol.
- [21] Token bucket. https://en.wikipedia.org/wiki/Token_bucket.
- [22] Blutmagie. Tor network status. <https://torstatus.blutmagie.de/>.
- [23] Tor Project. Tor network download. <https://www.torproject.org/download/download-easy.html.en>.

A Annexes

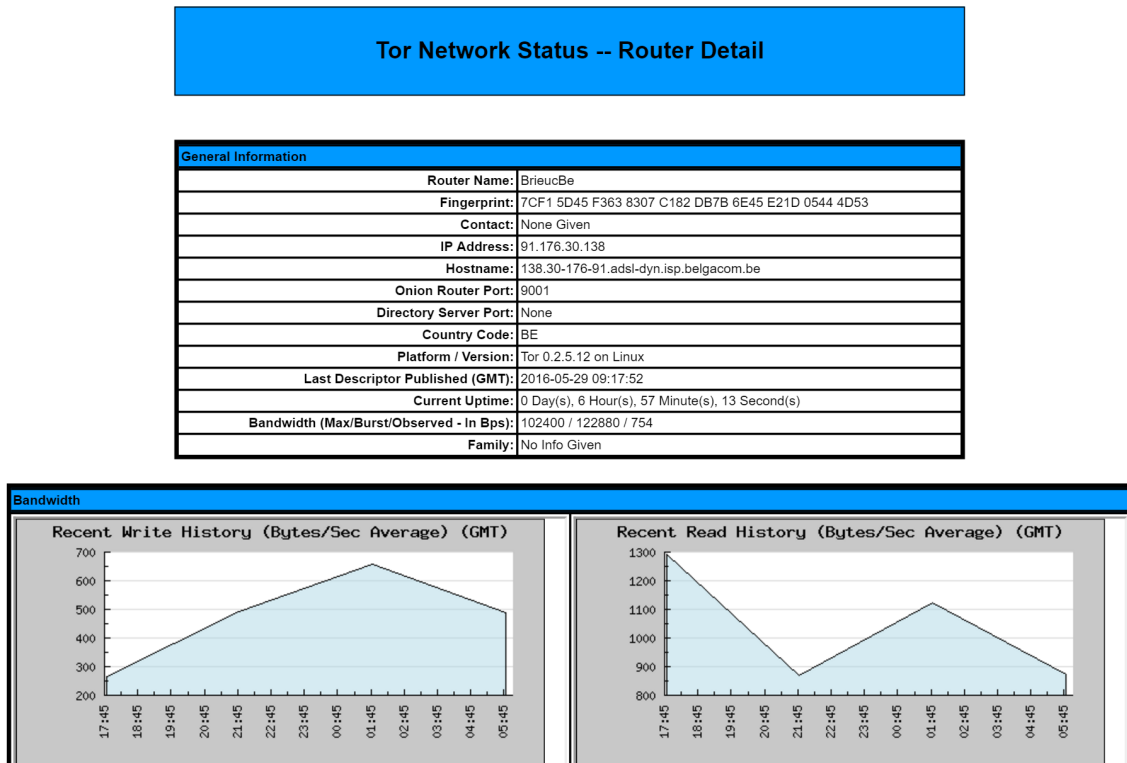


FIGURE 6 – Capture d'écran du site Blutmagie.de : Le relais Tor personnel pour ce travail est listé dans le serveur d'annuaire.

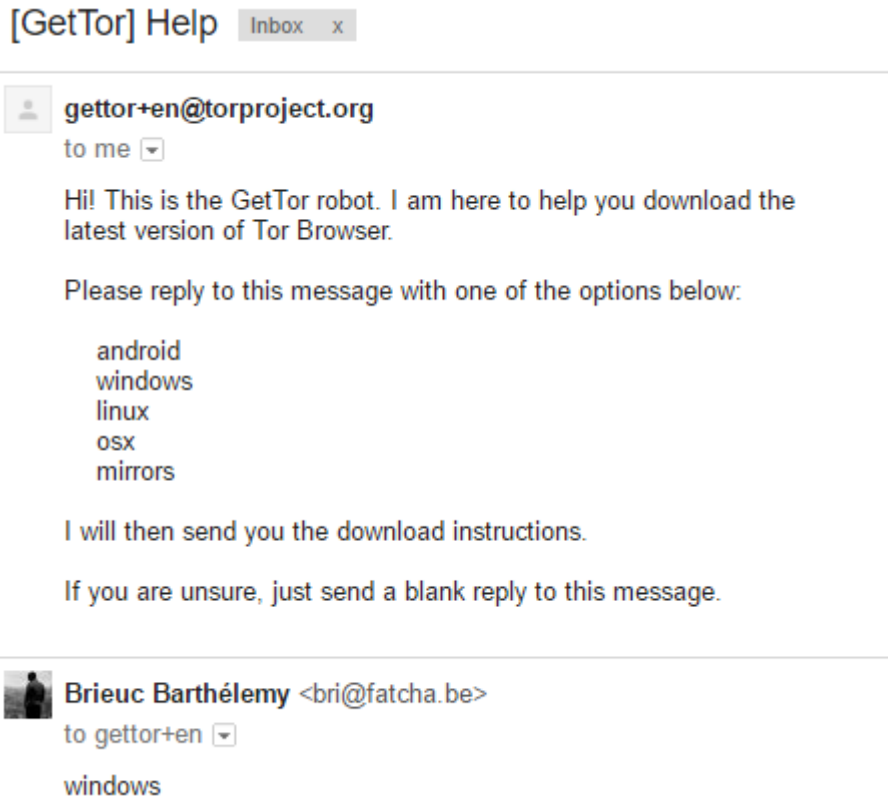


FIGURE 7 – Capture d’écran : Envoi des instructions sur l’adresse e-mail

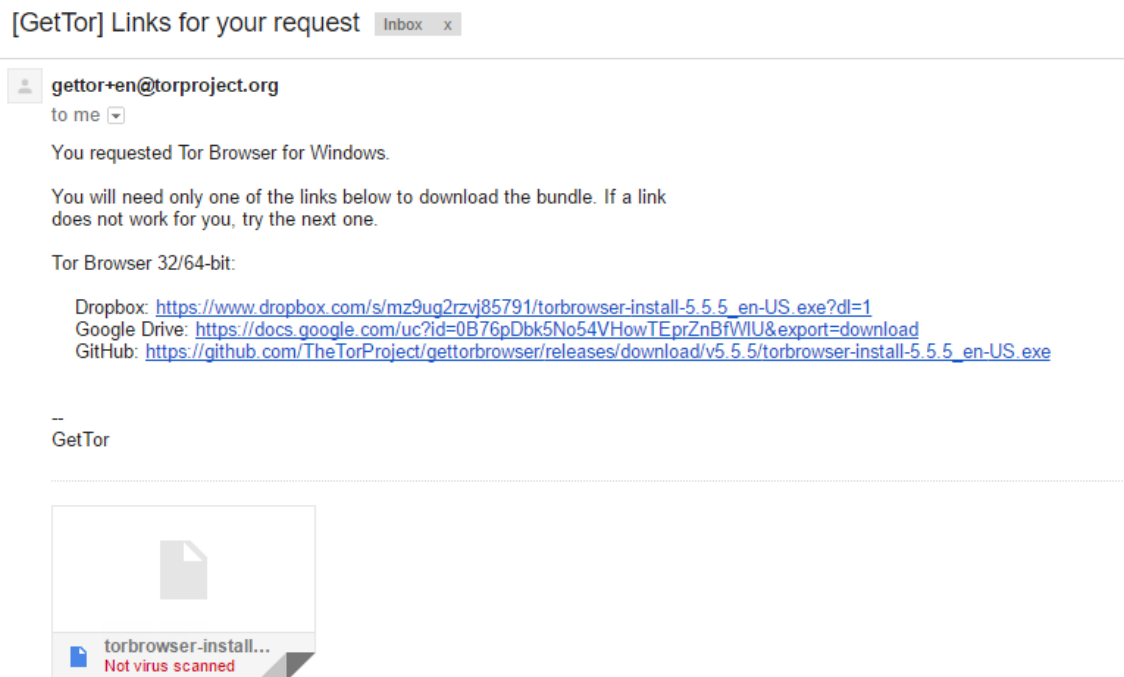



FIGURE 8 – Capture d’écran : Réceptions des informations



FIGURE 9 – Official directory server “Faravahar” - page d’accueil.

 **Roger Dingledine** arma@mit.edu
to Brieuc Barthélemy ▾

On Thu, May 19, 2016 at 10:13:54PM +0200, Brieuc Barthélemy wrote:

- > In point 4 (Page 4) the article talks about different keys:
- > - A long term identity key: to sign TLS certificate, OR route descriptor
- > and to sign directories.
- > - A short-term onion key: used to decrypt requests from users.
- > - Ephemerals keys (first words on page 5)
- > - And "link keys" for TLS communication.
- >
- > My question is: do you think the last two of them (Ephemerals and link
- > eyes) are the same one ?
- > Because I don't find any clear informations about those "ephemerals"

You might enjoy
<https://www.torproject.org/docs/faq#KeyManagement>

It sounds like the 'ephemeral keys' you mention could be the circuit-level keys, that is, the symmetric keys negotiated by the circuit handshake and used for the lifetime of that circuit.

...

--Roger

FIGURE 10 – Réponse de Robert Dingledine (Arma) -Président, Directeur, et cofondateur du Projet Tor à la question qu’est ce que la clé éphémère.


```
## Define these to limit how much relayed traffic you will allow. Your
## own traffic is still unthrottled. Note that RelayBandwidthRate must
## be at least 20 KB.
## Note that units for these config options are bytes per second, not bits
## per second, and that prefixes are binary prefixes, i.e. 2^10, 2^20, etc.
RelayBandwidthRate 100 KB # Throttle traffic to 100KB/s (800Kbps)
RelayBandwidthBurst 120 KB # But allow bursts up to 200KB/s (1600Kbps)

## Use these to restrict the maximum traffic per day, week, or month.
## Note that this threshold applies separately to sent and received bytes,
## not to their sum: setting "4 GB" may allow up to 8 GB total before
## hibernating.
##
## Set a maximum of 4 gigabytes each way per period.
AccountingMax 4 GB
## Each period starts daily at midnight (AccountingMax is per day)
#AccountingStart day 00:00
## Each period starts on the 3rd of the month at 15:00 (AccountingMax
## is per month)
AccountingStart month 1 00:00
```

FIGURE 11 – Fichier de configuration d'un relais Tor : spécifiquement concernant la bande passante.