Nom : Groupe ISC2il- Poste :

Laboratoire Réseaux et Applications 2023-2024

Marc SCHAEFER
Marc.Schaefer@he-arc.ch

http://www.he-arc.ch/ingenierie

22 novembre 2023

Neuchâtel, le 22 novembre 2023

Ces laboratoires ont pour but de permettre aux étudiants de 2e année ISC-il d'approfondir les éléments théoriques et de mettre en pratique des concepts avancés pertinents à la formation bachelor en développement logiciel et multimédia (nouvelles technologies, technologies multimédia, service de base, gestion réseau et quelques éléments de développement réseau : développement LDAP, VoIP WebRTC, ...).

Licence et droits d'auteurs

Marc SCHAEFER
Maître d'enseignement
Ing. inf. dipl. EPFL
http://he-arc.ch/ingenierie/

Table des matières

0. Introduction

Contenu du chapitre

— objectifs du laboratoire *Réseaux et Applications*
— votre implication
— évaluation
— description du laboratoire de téléinformatique

1

Objectifs du chapitre

— vous savez ce qui est attendu de vous
— vous connaissez les modalités d'évaluation
— vous connaissez les éléments essentiels du laboratoire

Objectifs du laboratoire Réseaux et Applications – 0.1

- appliquer et approfondir des éléments théoriques du cursus *réseaux* de l'orientation IL, en particulier
  - nouveaux protocoles : IPv6
  - multimédia : sans-fil, VoIP, QoS
- tout cela, en général, de manière virtualisée, avec des préconfigurations de réseaux définies par logiciel
- en option, développer un peu de code (baquet, WebRTC ...)
- s'autonomiser (recherches documentaires, questions adéquates, travail régulier, etc)

2

Planification du semestre du laboratoire La planification initiale du cours est disponible sur le Gitlab.

Chaque groupe de deux étudiants vient au moins **une fois toutes les 2 semaines** : il est possible de venir la semaine suivante en cas d'empêchement – sous réserve de place dans la salle.

Errata et indications complémentaires S'il y a des erreurs dans ce polycopié (errata), vous trouverez également les corrections dans le Gitlab, ainsi que des indications complémentaires, des références, etc.

Disponibilité de l'enseignant Les enseignants sont disponibles durant les cours, mais aussi par e-mail (*Marc.Schaefer@he-arc.ch*, *Fabrizio.Albertetti@he-arc.ch*) et sur rendez-vous.

Votre implication – 0.2

- pour chaque labo :
  - préparer la partie théorique éventuelle à l'avance
  - effectuer les expérimentations et recueillir les résultats importants durant le laboratoire (capture d'écran *et* captures Wireshark en format pcap pour pouvoir mieux les analyser)
  - répondre aux questions après le laboratoire et avant le laboratoire suivant (pas celles pour le rapport)
  - au début labo suivant : répondre au questionnaire noté portant sur le labo précédent (par groupe de 2)
- pour les deux laboratoires pour lesquels vous avez choisi de rendre un rapport :
  - faire comme pour chaque laboratoire (cf ci-dessus)
  - en plus rédiger, pour la 1ère et la seconde échéance, un rapport selon les directives (contenant également les réponses aux questions pour le rapport, reliant à la théorie, etc) : utiliser les semaines sans cours et le laboratoire additionnel prévus pour !

3

avant	**pendant**	**après**
– lire chaque labo – dans certains cas, éléments théoriques à lire et résumer – préparer des questions, envoyer par e–mail ou poser pendant le labo	– faire des observations rapides – prendre des captures réseau et d'écran – lors d'attentes, répondre aux questions – ne pas rester bloqué, appeler l'enseignant	– si vous rendez le rapport: – structurer – résumer – lier à la théorie – références – questions rapport (voir recommandations) – dans tous les cas: préparer les questions

Evaluation – 0.3

— questionnaire pour chaque laboratoire
— deux rapports évalués (un environ deux-tiers du semestre, un à la fin)

4

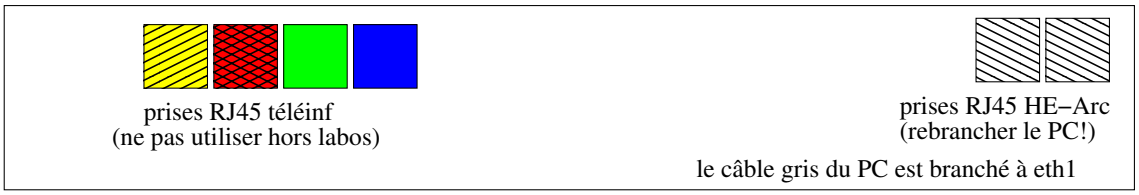
Laboratoire de téléinformatique – 0.4

— chaque table
  — prises réseau labo : jaune (VLAN 77, 157.26.77.0/24), rouge (VLAN 16, 192.168.1.0/24, IPv6), vert (VLAN 77 en hub ; ou autre), bleu : eth0
  — 2 prises HE-Arc pédagogique (VLAN 10)

5

Labo 304

Table étudiant



les 2 prises utilisables pour vos laptops, mais rebranchez le PC fixe à la fin du cours!

Légende

réseau pédagogique Ecole HE−Arc

réseau téléinf (sous−réseau 157.26.77.0/24, VLAN 77, switché)

réseau externe non sécurisé (connexion ADSL, 192.168.1.0/24 et IPv6, VLAN 16, switché)

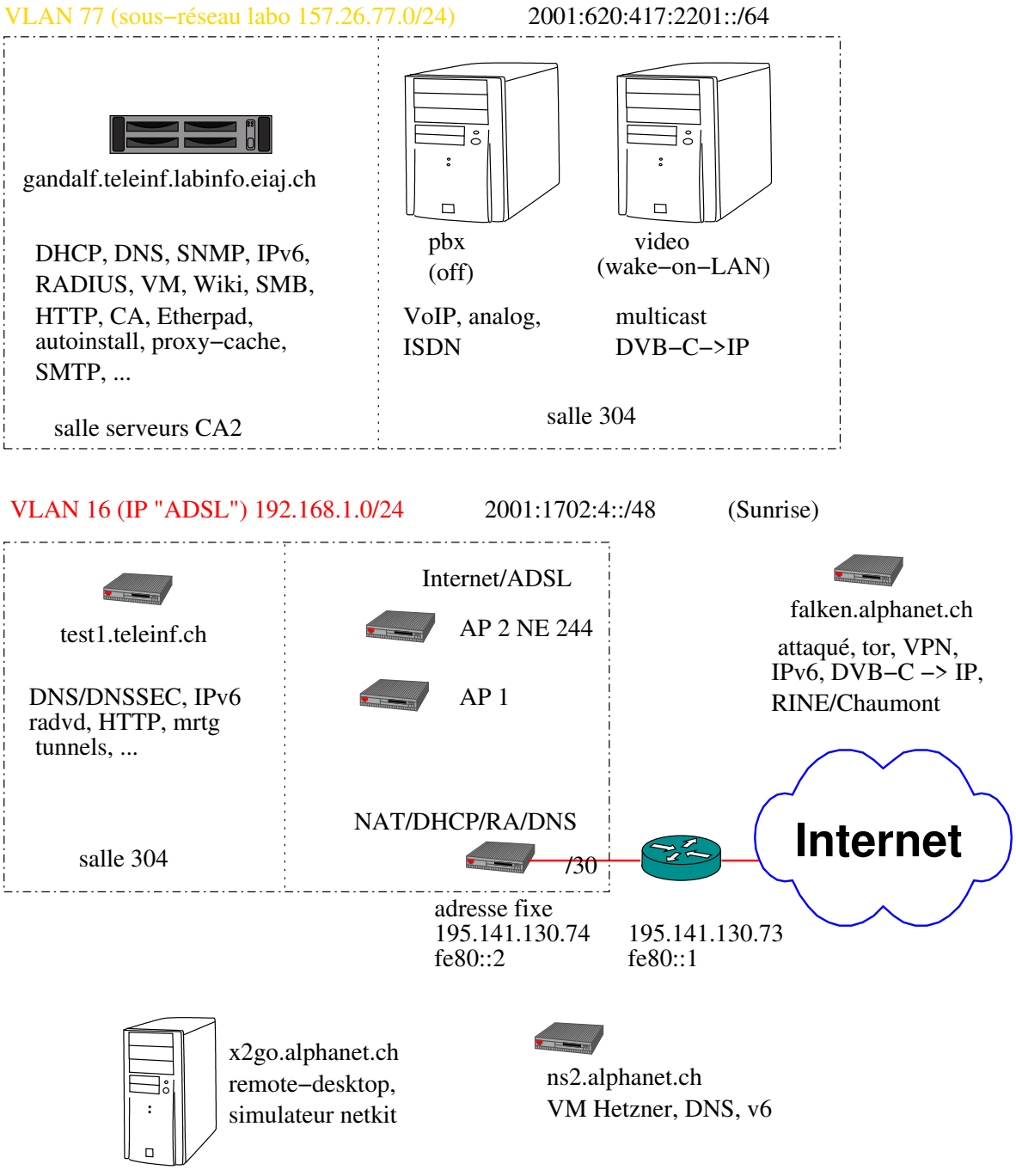
divers (hub 77, ISDN, analogique, xDSL)

carte réseau supplémentaire du PC (eth0)

Diagramme conceptuel des services et réseaux du laboratoire

Services labo TINF

les VLAN 10, 77 et 16 sont accessibles
en 244, 304, 306 ainsi qu'au PTSI St–Imier.



Installation de Netkit – 0.5

Netkit est utilisé par certains laboratoires, voici comment l'installer sur une machine réelle ou virtuelle Debian GNU/Linux buster.

Il est *déjà* installé sur la machine virtuelle fournie. Dans ce cas, il n'y a *rien* à faire, sinon lancer le `source ~/NETKIT/SOURCE_ME` dans chaque shell.

6

Installation de netkit (si nécessaire) Le VPN – ou le WiFi ou filaire à la HE-Arc – est nécessaire durant l'installation.

```
wget -O /tmp/netkit-install \
   https://www.alphanet.ch/~schaefer/tmp/netkit-install

# mettre à jour liste des paquets logiciels
# éventuellement ajouter l'option
#    --allow-releaseinfo-change
sudo apt-get update

# mettre à jour les packages
sudo apt-get -u dist-upgrade

# éventuellement redémarrer si le kernel a été changé

# le script doit vous dire que tout est OK
bash /tmp/netkit-install
```

1. Labo IPv6

Objectifs – 1.1

7

— expliquer les changements principaux entre IPv4 et IPv6
— expliquer la notion d'adresse IPv6 et de portée des adresses IPv6
— exploiter en parallèle IPv4 et IPv6
— expliquer le fonctionnement du remplaçant de ARP en IPv6, le protocole NDP et ses différences par rapport à ARP
— mettre en place un petit réseau IPv6
— expliquer la configuration automatique d'adresse avec un *Router Advertisement Daemon* (y.c. variantes sphère privée)
— mettre en place un tunnel IPv6 sur IPv4

Le protocole IPv6 Le protocole IP version 6 est actuellement au début de sa phase productive. Ce qui n'empêche pas de nombreux pays et fournisseurs de déployer ce protocole de manière étendue. Prévu comme successeur de la version actuelle (IP version 4) dès le milieu des années 90, il tarde à s'imposer pour diverses raisons (IPv4 a repris des innovations d'IPv6 comme le chiffrement ou la qualité de service ; le problème du nombre d'adresses a été reporté par les plages privées (NAT/PAT) ; le coût de transition est important et n'a pas encore trouvé de valeur économique pour l'utilisateur final).

Il est cependant à prévoir que dans les prochaines années, IPv6 sera au moins déployé en parallèle à IPv4 quasiment partout, pour ensuite le remplacer, en particulier dans les domaines spécifiques où ses avantages (sécurité intrinsèque, plus grandes plages d'adresses [1] et meilleur routage ; meilleure performance ; simplification de la configuration et qualité de service) le rendront incontournable. Le fait que les postes clients supportent aujourd'hui tous l'IPv6 dans une certaine mesure (Microsoft Windows Vista, 7 8 et 10, GNU/Linux, Mac OS X) est un signal dans la bonne direction.

Calcul d'adresses Comme en IPv4, IPv6 utilise la notation **CIDR** : on exprime un sous-réseau sous la forme `préfixe/fixe` où `fixe` indique le nombre de bits qui sont fixes dans le préfixe – et donc `128 - fixe` indique les bits libres.

1. Cablecom a déployé ses clients privés en IPv4 émulé sur CATV-IPv6 – DSLite – et en IPv6 avec un /64, la plupart des opérateurs mobiles font du *Carrier Grade NAT* en IPv4 ; conséquence : les utilisateurs finaux sont en adresse privée v4 et n'ont plus d'adresse v4 directement accessible.

Introduction théorique à IPv6 (partie 1) – 1.2

Contexte général

- raréfaction des adresses et sous-réseaux disponibles en particulier pour les pays émergents et les nouveaux besoins
- augmentation de la taille des tables de routage dans les routeurs
- le NAT/PAT détruit la nature *end-to-end* d'Internet et pose des problèmes de performance
- le P2P va dans la direction d'une adresse par équipement, voire par application
- les besoins des applications ne cessent de croître
- IPv4 a plus de 40 ans : préparons-nous au changement !
- problème principalement *stratégique*

8

Plus grand adressage en IPv6 Pour contourner la limite actuelle des adresses IP (version 4) sur 4 octets, c'est finalement la proposition SIPP qui a été choisie plutôt que les projets TUBA (TCP/UDP with Bigger Address – OSI/CLNP) ou CATNIP (OSI/CLNP, Novell/IPX, IP). Les nouvelles adresses IP seront ainsi codées sur 16 octets, ce qui conduira à un nouvel entête IP simplifié sur 40 octets au lieu des 20 octets actuels !

Routage par préfixe Optimise les tables de routage.

Plus de nécessité de NAT/PAT On peut toujours l'utiliser, mais cela ne devrait plus être nécessaire contrairement à IPv4.

Problème stratégique IPv6 est déjà déployé, parfois même par défaut. La question est : y-a-t-il déjà une application importante qui ne fonctionnerait plus en IPv4 ? Combien faut-il investir pour l'IPv6, alors que le bénéfice, suivant le type d'acteur, n'est pas immédiat et le coût certain ?

IPv4 : rappels

- adressage 32 bits (environ 4 milliards d'adresses)
  - beaucoup de *gaspillage* (anciennes classe A, zones réservées, taille des tables de routage...)
  - peu d'adresses véritablement utilisables
- entête complexe et signification des champs évolutive (peu claire parfois : p.ex. champ TOS)
- le passage du modèle des 5 classes (A à E) au subnetting/norme CIDR a augmenté la flexibilité, limité le nombre d'entrées de table de routage, mais la fragmentation de l'adressage jusqu'au /22 voire /24 encombre la mémoire des routeurs de bord du réseau
- la fragmentation dans les routeurs baisse la performance
- la transmission de données multimédia n'est pas réalisable hors de réseaux contrôlés (qualité de service)
- la sécurité et la cryptographie sont souvent relégués aux couches supérieures

9

On peut tout résoudre, ou presque, en IPv4, mais c'est souvent complexe et lourd :

- qualité de service (DiffServ, RSVP)
- mobilité IP
- sécurité (IPsec, VPN, DNSSEC, ...)
- NAT/PAT pour le problème de taille d'adressage
- anycast (load-balancing par GeoDNS, load-balancer)
- don't fragment / PMTUDISC
- ECN (Enhanced Congestion Notification)

De plus, ces améliorations sont souvent confinées à un seul réseau d'opérateur, alors que IPv6 propose des mécanismes pour les généraliser (même si, par exemple pour la qualité de service, voire la mobilité, cela dépend, en fait, d'un déploiement NGN full).

IPv6 : le successeur

nouveautés

- gestion optimisée des options, notamment par les routeurs, grâce au chaînage des entêtes
- autoconfiguration des adresses et nouveau procédé remplaçant ARP (NDP)
- qualité de service (pour le multimédia), gestion de la congestion (ECN) et intégration facilitée à MPLS
- authentification et chiffrement (optionnels, mais implémentation obligatoire)
- mobilité IP
- protection vie privée (chiffrement, adresses aléatoires…)

Les changements principaux d'IPv6

adressage

- étendu (128 bits, 16 octets)
- nouveau schéma (préfixes)
- sous-réseaux par défaut en /64

entête

- objectif : gain de performance
- entête principal IP de taille fixe
- simplification (des champs, suppression du checksum IP, de la fragmentation par les routeurs…)

Intégration IPv4/v6

- les couches supérieures (4 : TCP, UDP, ICMP ; 7 : HTTP, FTP, etc) sont pas ou peu touchées
  - tous les protocoles spécifiant des adresses dans leurs entêtes sont touchés (FTP PORT, PASV ; SIP ; etc)
  - quelques options particulières (p.ex. les jumbograms, RFC-2675) nécessitent des changements à TCP et UDP
  - checksums obligatoires en TCP et UDP

11

Stratégies de transition Voir https://en.wikipedia.org/wiki/IPv6_transition_mechanism, par exemple SIT (*Simple Internet Transition*).

Intégration IPv4/v6 : implémentations

- dual-stack (natif, le mieux)
- IPv6 natif, IPv4 en émulation : p.ex. UPC avec DSLite
- encapsulation / tunnelling, plus ou moins efficace
  - tunnel efficace chez le fournisseur lui-même : p.ex. Swisscom 6rd
  - tunnel vers un fournisseur tiers, moins efficace (délai notamment)
  - dans tous les cas causera des problèmes de MTU

12

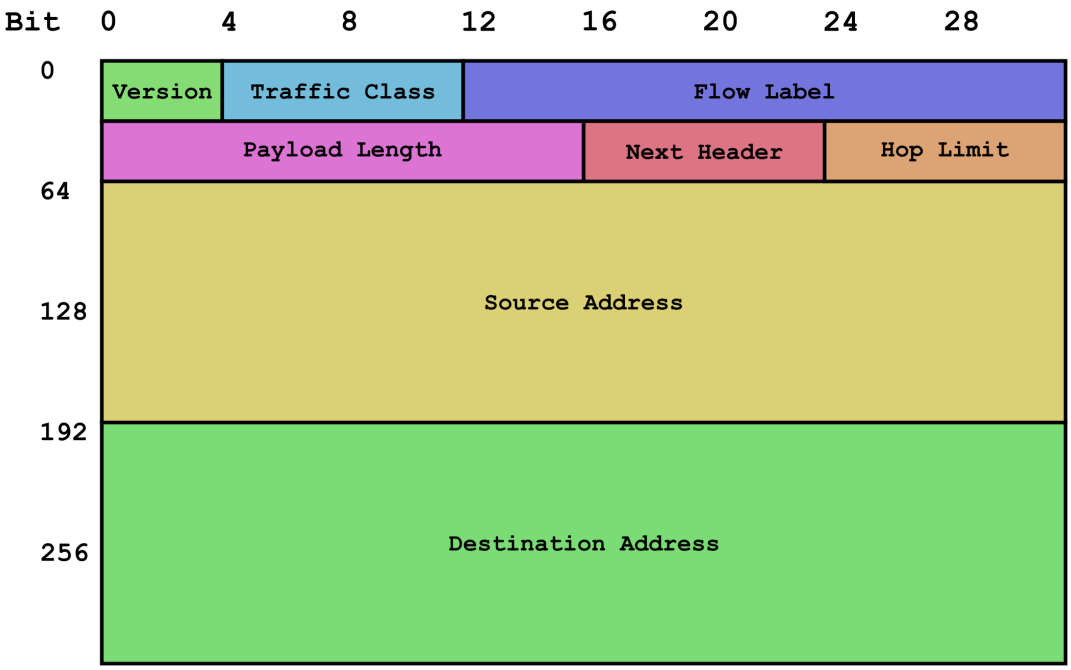
- passerelles couches supérieures (proxy-application, gateway couche 7)

Problème de MTU avec les tunnels Un entête supplémentaire de tunnel consomme des octets du payload car le MTU couche 2 reste identique. Le MTU disponible pour les couches supérieures est diminué d'autant, ce qui peut poser des problèmes en particulier si la détection de MTU (*Path MTU Discovery*) est bloquée (firewall).

Dans ce dernier cas, on peut appliquer le même *work-around* que pour le PPPoE dans l'xDSL, soit le *TCP MSS clamping* : à l'ouverture d'une connexion TCP, le MSS négocié est subrepticement altéré par le routeur.

Augmentation du délai avec les tunnels Sur un exemple pratique, l'IPv6 natif via init7 sur google.com prend 11.2 ms, et en tunnel via Hurricane Electric 16 à 20 ms. Cela peut être pire si le serveur de tunnel est assez loin en terme de hops.

Entête IPv6



(source : https://fr.m.wikipedia.org/wiki/Fichier:IPv6_header_rv1.png)

13

Description des champs C'est plus simple qu'en IPv4 !

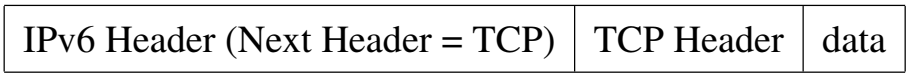
- Version : 6 (la 5 est réservée, la 4 est IPv4)
- Class et Flow Label : utilisé pour QoS (notamment le Flow Label pour MPLS)
- Payload Length : 0 à 65535 [1] – ne compte pas l'entête fixe v6
- Next Header : type de l'entête qui suit immédiatement (ICMPv6, UDPv6, TCPv6 ou entêtes de chaînage – options et contrôle)
- Hop Limit : décrémenté à chaque routeur, le datagramme est jeté lorsqu'il atteint zéro. Remplace le champ TTL d'IPv4. Utilisé également pour la sécurité : certains messages de contrôle ICMP au sein d'un sous-réseau (domaine de diffusion Ethernet) ne sont valides qu'avec Hop Limit = 255 (notamment NDP)

Le champ *checksum* a notamment disparu : on compte sur la couche 2 ainsi que sur la couche 4 (checksums TCP, et checksum UDP maintenant obligatoires).

1. un mode étendu (*jumbogram*) existe via un entête chaîné de type *hop by hop*, voir http://tools.ietf.org/html/rfc2675, par exemple pour usage en data center

Chaînage d'entête (Next Header)

— cas le plus courant : le routeur peut simplement router le datagramme, Next Header contient TCP, UDP ou ICMPv6, l'entête fait la taille fixe de 40 octets, par exemple avec TCP :



14

— exemple (complexe) en cas de chaînage(s) : l'entête de base fait 40 octets, et on y ajoute les entêtes chaînés ; le routeur peut devoir interpréter certains des entêtes chaînés en plus :

— l'ordre obligatoire des entêtes chaînés optimise le routage

Protocoles usuels dans le cas le plus courant

protocole	**Next Header** (décimal)
ICMPv6	58
TCP	6
UDP	17

Protocoles en cas de chaînage

entête chaîné	**description**	**Next Header** (décimal)
Hop-by-Hop	options à considérer par tous les équipements ; jumbogram	0
Destination Options	à traiter par la première destination	60
Routing	source routing	43
Fragment	gestion de la fragmentation	44
IPsec Authentification Header (AH)	signature du datagramme	51
IPsec Encrypted Security Payload (ESP)	chiffrement du datagramme	50
Destination Options	à traiter par la dernière destination	60

Chaque entête contient potentiellement un entête suivant (Next Header). L'ordre obligatoire des entêtes est spécifié ci-dessus, ce qui permet d'optimiser le routage (les routeurs s'arrêtent de traiter et transmettent simplement le datagramme).

Exemple de chaînage : fragmentation

Lorsque le MTU de l'interface de sortie du routeur est plus faible :

- en IPv4 que la taille du datagramme IP, deux cas possibles :
  - le drapeau *Don't fragment* est mis : le datagramme est rejeté (erreur ICMP ; on peut appliquer le PMTU-DISC)
  - sinon, le datagramme est fragmenté par le premier routeur concerné par le problème du MTU, et défragmenté à l'arrivée (terminal)
- en IPv6
  - les routeurs ne fragmentent *jamais*, ils se bornent à retourner le message ICMP Packet Too Big avec l'indication du MTU maximum possible
  - c'est au terminal émetteur de reconnaître l'erreur et d'envoyer des datagrammes de la bonne taille ou de fragmenter par chaînage
  - un datagramme plus petit ou égal à 1280 octets passera toujours

15

Path MTU Discovery Le RFC-1191 décrit le *Path MTU Discovery* / PMTU-DISC. Il permet à un terminal de détecter le MTU minimum dans un chemin quelconque, consulter par exemple `http://www.znep.com/~marcs/mtu/`. Toutefois, la mise en place de filtrage ICMP non intelligent (firewall) aboutit à empêcher le *Path MTU Discovery* (PMTU-DISC). Ce problème crée des parties d'Internet v6 non accessibles depuis des tunnels IPv6 sur IPv4, uniquement en natif.

Comment envoyer des datagrammes de la bonne taille ? L'erreur ICMP devrait permettre d'adapter la taille : le terminal émetteur peut alors adapter le MSS TCP par exemple. Une autre solution non adaptative est de ne jamais dépasser 1280 octets en UDP.

En alternative, il peut aussi décider de fragmenter le datagramme en plusieurs datagrammes :

- l'entête de chaque datagramme-fragment est exactement le même que l'original, sauf que le champ Next Header de l'entête IPv6 mis à Fragment, et qu'un entête intercalaire Fragment est ajouté (avec son Next Header avec la valeur précédente : p.ex. UDP)
- comme en IPv4 il y a les notions de *fragment offset*, *message ID* et *last fragment*, champs situés dans l'entête chaîné Fragment
- les routeurs transmettent tous les datagrammes en n'examinant pas les entêtes au-delà de l'entête IPv6 (gain de performance)
- comme auparavant en IPv4, c'est le destinataire final qui défragmente (avec une minuterie, des tampons, etc), ce qui est un risque de sécurité et donc peut être bloqué (firewall)
- comme auparavant, une perte d'un fragment signifie la réémission éventuelle par la couche supérieure de l'émetteur de tous les fragments.

Adressage

16 octets (128 bits)

notation

- hexadécimale : `2001:0db8:85a3:0000:0000:8a2e:0370:7334`
  (équivalente à : `2001:0db8:85a3:0:0:8a2e:370:7334`)
- forme simplifiée (élimination des zéros) :
  `2001:0db8:85a3::8a2e:0370:7334` ou encore
  `2001:db8:85a3::8a2e:370:7334`; attention aux ambiguités !
- intégration IPv4 possible

16

préfixes (similaire à CIDR IPv4)

Ambiguités `2001::FFD3::57ab` est *interdit* car ambigu ; il faut spécifier : `2001:0:0:ffd3::57ab` ou `2001::ffd3:0:0:57ab` en fonction de ce qui est voulu.

Intégration IPv4 Par exemple `2001:0db8:85a3::192.168.42.24`, que l'on peut aussi écrire `2001:db8:85a3::c0a8:2a18`.

Préfixes

- les adresses sont groupées par préfixes, noté comme une adresse IPv6 avec les bits hors préfixe à zéro, puis une spécification similaire au CIDR d'IPv4
- par exemple : `2001::/8` signifie que seuls les 8 premiers bits sont fixes, les 120 autres bits peuvent varier et former autant d'adresses de ce préfixe

Types d'adresses (variantes)

unicast
- — désigne une interface réseau unique, dans la portée (*scope*) spécifiée
- — portées : *global*, *link-local*, *unique (site) local*

anycast (n'existe pas tel quel en IPv4)
- — désigne n'importe quelle interface réseau (UNE) parmi N (groupe)
- — on ne peut distinguer les adresses anycast dans leur forme des adresses unicast (normales)

multicast
- — toutes les interfaces du groupe considéré
- — commence par le préfixe `FF00::/8`

17

Il n'y a pas à proprement parler de *broadcast* en IPv6 !

anycast Le choix de l'interface parmi N est effectué selon des critères de distance via le protocole de routage. La 1ère adresse d'un sous-réseau (avec tous les bits libres à zéro) est réservée pour l'adresse anycast du routeur [1].

1. https://tools.ietf.org/html/rfc4291#section-2.6.1

Multicast

Les adresses multicast ont également une portée (scope) :

8	4	4	112 (bits)
11111111	0RPT	scope	group id

Le protocole *Neighbour Discovery Protocol* (NDP) utilise le multicast : par exemple
18 `FF02::1:FF00:0/104`

- les 24 derniers bits formés de l'adresse unicast (ou anycast) de l'adresse IP recherchée
- multicast, uniquement pour ce lien (link-local, domaine de diffusion Ethernet p.ex.)
- sollicited-node [a]
- avantage par rapport au ARP d'IPv4 : toutes les interfaces ne sont pas dérangées

a. node/nœud au sens couche 3 : routeur ou terminal

Le format d'adresse multicast en détail

- T=1 : groupe multicast connu de l'IANA
- P et R : voir RFC-3306 et 3956
- scope (portée) :

0x1	interface-local (loopback)
0x2	link-local
0x3	admin-local
0x5	site-local
0x8	organisation-local
0xE	global

- group ID
  - usuellement, seuls les derniers 32 bits de l'adresse sont utilisés pour la spécification du groupe (comme pour les multicast IPv4)
  - groupes réservés : par exemple 1 pour tous les machines ; 2 pour tous les routeurs

Exemple NDP

```
 1 0.000000    fe80::211:22ff:fe33… ff02::1:ff33:4401    ICMPv6     86 Neighbor Solicitation for fe80::211:22ff:fe33:4401 from 00:11:22:33:4…
 2 0.000044    fe80::211:22ff:fe33… fe80::211:22ff:fe33… ICMPv6     86 Neighbor Advertisement fe80::211:22ff:fe33:4401 (sol, ovr) is at 00:1…
 3 0.000165    fe80::211:22ff:fe33… fe80::211:22ff:fe33… ICMPv6    118 Echo (ping) request id=0x0902, seq=1, hop limit=64 (reply in 4)
 4 0.000181    fe80::211:22ff:fe33… fe80::211:22ff:fe33… ICMPv6    118 Echo (ping) reply id=0x0902, seq=1, hop limit=64 (request in 3)
```

Problèmes des adresses privées d'IPv4

Exemple 1

— deux sites ou entreprises sont reliés par VPN (application B2B, accès aux ressources mutuelles, etc)
— on se rend compte que les deux sites utilisent en interne la *même* plage d'adresses privées IPv4 (p.ex. 192.168.1.0/24)

19

Exemple 2

— un gateway agit comme proxy entre un réseau de terrain IP et un accès Internet d'un FAI quelconque, le problème se corse s'il s'agit d'un équipement grand public
— risque : la plage d'adresse privée du réseau de terrain peut être la même que celle derrière le routeur du FAI

Solution en IPv6 : unique local addresses (ULA)

Work-around en IPv4

— problème 1 : résoudre cela par des règles complexes de firewall (NAT, PAT, masquerading)
— problème 2 : adapter dynamiquement les adresses du réseau de terrain en fonction de ce qui se passe du côté FAI, 172.16.0.0/12 est aussi un candidat "moins" utilisé

Unique (site) local addresses (ULA) [1]

Les adresses privées sont remplacées par des adresses "uniques" locales, avec un préfixe les identifiant (scope, portée). L'allocation des plages peut se faire

`fc00::/8` via un registre centralisé pour tout Internet (n'existe pas, en fait)

`fd00::/8` via des nombres *aléatoires* de 40 bits – choisis [2] via un protocole décrit dans le RFC-4193 [3] qui limitent les risques de collision (donc préfixe de 48 bits)

1. http://en.wikipedia.org/wiki/Unique_local_address
2. https://cd34.com/rfc4193/
3. http://tools.ietf.org/html/rfc4193

Portée des adresses (scope) et préfixes globaux

portée	**préfixe (binaire)**	**préfixe (hexadécimal)**	**fraction**
non assigné	`0000 0000`	`::0/8`	1/256
réservé	`0000 001`	`200::/7`	1/128
global unicast	`001`	`2000::/3`	1/8
link-local unicast	`1111 1110 10`	`fe80::/10`	1/1024
reservé (anc. site-local unicast)	`1111 1110 11`	`fec0::/10`	1/1024
unique local prefixes	`1111 110`	`fc00::/7`	1/128
random unique local prefixes	`1111 1101`	`fd00::/8`	1/256
multicast	`1111 1111`	`ff00::/8`	1/256

(voir http://www.iana.org/assignments/ipv6-address-space)

Spécificités Microsoft Attention, les documentations Microsoft prévoient que certaines adresses sous `fec0::/10` sont prévues pour des services DNS automatiques. C'est une violation du standard, en effet, fec0 : :/10 ne doit plus être utilisé. Pour référence, les adresses proposées incorrectement [1] par Microsoft sont : `fec0:0:0:ffff:1`, `2` et `3`.

Exercices Faire les exercices 1 et 2 page 50.

1. http://technet.microsoft.com/en-us/library/cc783049%28v=ws.10%29.aspx

Routage par préfixe

Une adresse de type *global unicast* se construit comme suit :

global routing prefix	subnet ID	interface ID (64 bits)

Les deux premiers champs peuvent être de diverses longueurs, au total maximum 64 bits. Le premier champ sert au routage externe, le deuxième au routage interne. L'interface ID identifie la machine.

21

Routage par préfixe Le global routing prefix est utilisé pour les protocoles de routages externes (inter-AS : EGP ; on y route globalement plutôt des /48 par exemple, ou plus grand – typiquement /32).

Le subnet ID est utilisé dans les protocoles de routages internes (intra-AS : RIP, OSPF).

L'interface ID quant à elle est définie de plusieurs manières (voir slide suivant) : en général [1], l'interface ID comporte 64 bits (comme un subnet /64), ce qui laisse 64 bits pour l'adresse de machine d'un sous-réseau.

1. http://en.wikipedia.org/wiki/IPv6_subnetting_reference

Attribution des adresses

— comme en IPv4 : attribution manuelle (adresse IP, envergure du sous-réseau, routeur)
— nouveau
  — chaque interface a *automatiquement* une adresse de portée (scope) link-local, construite par le terminal de manière unique pour l'interface considérée, dans `fe80::/10`
  — attribution automatique depuis un préfixe communiqué par un routeur (RA, *Router Advertisement*), de diverses portées – en général est en portée globale et permet de sortir du sous-réseau
— similaire : DHCPv6, qui peut ne s'occuper que de la partie *non adresse* (serveurs DNS, indication serveur TFTP, etc)

22

Addresses automatiques link-local
— préfixe : `fe80::/10`
— 64 bits inférieurs : valeur EUI-64 (p.ex. construite par l'adresse MAC) : interface ID
— ne permet pas de sortir, comme les adresses aléatoires d'IPv4 (169.254.0.0/16, RFC-5735)

Router Advertisement (RA) Le *Router Advertisement* [1] (régulier, ou produit sur demande via *Router Sollicitation* – RS) est un message ICMPv6 qui permet au terminal de fixer son adresse IP à partir d'un préfixe diffusé par le RA et éventuellement de configurer une route par défaut pour sortir du sous-réseau.

La nouvelle adresse est le préfixe diffusé par le RA suivi de l'adresse MAC (RFC-2462 SLAAC EUI-64 ; ou une valeur aléatoire à des fins de protection de la vie privée, RFC-4941)

Norme EUI-64 Cette norme fait correspondre une adresse MAC (couche 2, p.ex. Ethernet à 6 x 8 = 48 bits) aux 64 bits de poids faibles d'une adresse IPv6, en ajoutant la valeur sur 16 bits `fffe` au milieu.

1. http://www.networksorcery.com/enp/protocol/icmp/msg9.htm

Adresses prédéfinies

unicast

`0:0:0:0:0:0:0:0` ou `::0`	adresse non définie
`0:0:0:0:0:0:0:1` ou `::1`	adresse loopback

anycast

subnet prefix (N bits)	(128 - N) bits à zéro	sera traité par un des routeurs du subnet

23

multicast

`ff00::/128` à `ff0f::/128`	réservées
`ff01::1/128`	all-nodes interface-local (loopback)
`ff02::1/128`	all-nodes link-local (broadcast Ethernet)
`ff0X::2/128`	all-routers : avec X 1, 2 ou 5 (interface-local, link-local ou site-local)
`ff02::1:FFXX:XXXX`	sollicited-node address (NDP)

(dans le tableau ci-dessus, *node* a le sens couche 3 : routeur ou terminal)

Adresse non définie Elle sert pour spécifier une adresse inconnue, par exemple pour bind(2) pour recevoir des connexions ou des flux sur toutes les adresses locales de la machine. En IPv4, c'était `INADDR_ANY` (`0.0.0.0`).

Sollicited-node Address Sert à la découverte d'une adresse MAC pour une adresse IPv6 donnée, sans besoin de broadcast : seules les machines ayant des adresses ressemblant (24 bits inférieurs) à l'adresse cherchée vont recevoir le message multicast (*Neighbour Discovery Protocol* – NDP). Cela fonctionne car la machine recherchée s'est préalablement abonnée au groupe multicast concerné (*Multicast Listener*, envoyé à `ff02::16`).

D'ailleurs, avant d'utiliser une adresse, une machine va vérifier que personne n'a son adresse grâce à une requête NDP sur soi-même (self), voir page 30.

Résumé : les adresses d'une machine

terminal et routeur (sens couche 3 de : node/nœud)

- une adresse de portée link-local pour chacune de ses interfaces
- des adresses unicast ou anycast configurées automatiquement ou manuellement
- l'adresse *loopback*
- s'abonner aux groupes multicast
  - all-nodes
  - sollicited-node (pour toutes ses adresses de toutes les interfaces)
  - d'autres groupes auxquels il veut faire partie

24

routeur, en plus

- anycast de type subnet router, pour toutes les interfaces pour lesquelles il agit comme routeur
- multicast all-routers

ICMPv6

Rôle étendu

- Router Advertisement ("DHCP light")
  - remplace partiellement le DHCP sans le suivi des allocations (stateless !)
  - fonctionne par publication du préfixe du sous-réseau et de routes : les machines construisent leur adresse via EUI-64 (MAC ou aléatoire)
  - en option, autres fonctions : p.ex. RRDNS pour les serveur DNS
  - alternative plus complète : DHCPv6
- Neighbour Discovery Protocol ("le ARP de v6")
  - validation d'adresses (éviter les collisions)
  - conversion d'adresses couche 3 vers couche 2 : remplaçant totalement ARP
  - utilisation de groupes multicast formés des derniers bits de l'adresse IPv6, à la place des broadcasts
- IGMP (gestion des groupes multicast classiques)

25

Exemple de recherche d'une station (NDP, "ARP de v6") Neighbour Discovery Protocol :

```
⊞ Frame 108 (86 bytes on wire, 86 bytes captured)
⊞ Ethernet II, Src: 28:37:37:0a:cc:f0 (28:37:37:0a:cc:f0), Dst: IPv6mcas
⊟ Internet Protocol Version 6
  ⊞ 0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 32
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source: fe80::2a37:37ff:fe0a:ccf0 (fe80::2a37:37ff:fe0a:ccf0)
    Destination: ff02::1:ff17:99a3 (ff02::1:ff17:99a3)
⊟ Internet Control Message Protocol v6
    Type: 135 (Neighbor solicitation)
    Code: 0
    Checksum: 0x339d [correct]
    Target: fe80::226:bbff:fe17:99a3 (fe80::226:bbff:fe17:99a3)
  ⊞ ICMPv6 Option (Source link-layer address)
```

Exemple de démarrage d'une station (sans radvd ni DHCPv6) La station s'auto-allouera une adresse IPv6 manuelle, ici `fe80::1:2ff:fe30:0`.

1) Multicast Listener (abonnement à un groupe multicast) :

```
+ Frame 42 (110 bytes on wire, 110 bytes captured)
- Ethernet II, Src: 9c:8e:99:3c:55:23 (9c:8e:99:3c:55:23), Dst: IPv6mcast_
  + Destination: IPv6mcast_00:00:00:16 (33:33:00:00:00:16)
  + Source: 9c:8e:99:3c:55:23 (9c:8e:99:3c:55:23)
    Type: IPv6 (0x86dd)
- Internet Protocol Version 6
  + 0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 56
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source: fe80::9e8e:99ff:fe3c:5523 (fe80::9e8e:99ff:fe3c:5523)
    Destination: ff02::16 (ff02::16)
  - Hop-by-Hop Option
      Next header: ICMPv6 (0x3a)
      Length: 0 (8 bytes)
      Router alert: MLD (4 bytes)
      PadN: 2 bytes
- Internet Control Message Protocol v6
    Type: 143 (Multicast Listener Report Message v2)
    Code: 0 (Should always be zero)
    Checksum: 0x8d71 [correct]
  + Changed to exclude: ff02::1:ff30:0 (ff02::1:ff30:0)
  + Changed to exclude: ff02::1:ff3c:5523 (ff02::1:ff3c:5523)
```

(voir aussi la partie V, et constater le Next Header hop-by-hop, le Hop Limit de 1, et l'adresse destination `ff02::16`, all multicast MLDV2 routers ; l'abonnement au groupe multicast fonctionne selon RFC-3810 : éviter tout sauf l'adresse indiquée).

2) NDP self (détection de duplicat d'adresse) :

```
+ Frame 43 (78 bytes on wire, 78 bytes captured)
- Ethernet II, Src: 9c:8e:99:3c:55:23 (9c:8e:99:3c:55:23), Dst: IPv6mcas
  + Destination: IPv6mcast_ff:30:00:00 (33:33:ff:30:00:00)
  + Source: 9c:8e:99:3c:55:23 (9c:8e:99:3c:55:23)
    Type: IPv6 (0x86dd)
- Internet Protocol Version 6
  + 0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 24
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source: :: (::)
    Destination: ff02::1:ff30:0 (ff02::1:ff30:0)
- Internet Control Message Protocol v6
    Type: 135 (Neighbor solicitation)
    Code: 0
    Checksum: 0x7ac6 [correct]
    Target: fe80::1:2ff:fe30:0 (fe80::1:2ff:fe30:0)
```

3) Router Sollicitation (requête à un radvd, un *routeur advertisement daemon*) :

```
⊞ Frame 45 (70 bytes on wire, 70 bytes captured)
⊟ Ethernet II, Src: 9c:8e:99:3c:55:23 (9c:8e:99:3c:55:23), Dst: IPv6mcas
  ⊞ Destination: IPv6mcast_00:00:00:02 (33:33:00:00:00:02)
  ⊞ Source: 9c:8e:99:3c:55:23 (9c:8e:99:3c:55:23)
    Type: IPv6 (0x86dd)
⊟ Internet Protocol Version 6
  ⊞ 0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 16
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source: fe80::1:2ff:fe30:0 (fe80::1:2ff:fe30:0)
    Destination: ff02::2 (ff02::2)
⊟ Internet Control Message Protocol v6
    Type: 133 (Router solicitation)
    Code: 0
    Checksum: 0xf00e [correct]
  ⊟ ICMPv6 Option (Source link-layer address)
      Type: Source link-layer address (1)
      Length: 8
      Link-layer address: 9c:8e:99:3c:55:23
```

Fin de la théorie partie 1 : faire les parties I à III dès la page 39

Introduction théorique à IPv6 (partie 2) – 1.3

DHCPv6

- les adresses peuvent être attribuées par un radvd (*Router Advertisement Daemon*) via ICMPv6 donc soit avec l'adresse MAC (suivi possible), soit adresse aléatoire (suivi impossible)
- le routeur est configuré aussi par radvd, le serveur DNS peut être configuré globalement (site-local p.ex.) ou par une extension radvd (RRDNS)
- en alternative, DHCPv6 peut fournir des adresses
  - en fonction des adresses MAC
  - dans un pool dynamique
- en entreprise, DHCPv6 sera déployé en général, notamment pour les autres informations (DNS, NTP, TFTP, hostname, etc) et le suivi des allocations

26

Réseaux IoT Pour des réseaux de terrain, la configuration avec un radvd peut en général suffire.

Autres différences avec IPv4

- NAT/PAT : plus nécessaire, sauf dans des cas très particuliers
- DNS
  - forward : champ `AAAA`
  - reverse : inversé, sous `ip6.arpa`
- options
  - le champ Next Header permet de chaîner des options dans un ordre défini
  - cette implémentation permet un traitement efficace par les routeurs
- fragmentation
  - déjà en IPv4, le Path MTU Discovery et le drapeau *don't fragment* rendait rare l'utilisation de la fragmentation
  - IPv6 la supprime sur les routeurs ; en cas de nécessité de taille de datagramme plus basse, un datagramme ICMPv6 *Packet Too Big est envoyé*, avec l'indication du MTU concerné (le terminal peut adapter ou fragmenter)
- Explicit Congestion Notification (ECN) : en standard

27

Explicit Congestion Notification (ECN)

- permet à un routeur de marquer les datagrammes d'un flux en congestion avant de devoir jeter et donc permet d'informer l'émetteur via le récepteur
- option de TCP IPv4 (ECN) ; implémenté de base en IPv6 (champ Traffic Class) – si les routeurs le supportent
- particulièrement utile en data center

Mobilité IP

Problème en IPv4

— routage triangulaire [a] : le trafic émetteur vers mobile circule via le Home agent, mais le retour est direct, ce qui peut poser des problèmes de performance et d'anti-spoofing [b]

Solution en IPv6

— les entêtes chaînés permettent de gérer le problème

28

— solution à l'avenir : RFC-3693 [c]

Qualité de service et intégration MPLS

IPv6 a une intégration directe MPLS grâce au champ Flow Class.

a. http://en.wikipedia.org/wiki/Triangular_routing
b. http://en.wikipedia.org/wiki/Anti-Spoofing
c. http://tools.ietf.org/html/rfc3963

Qualité de service et intégration MPLS IPv4 avait initialement 3 bits de priorité et 3 bits encodaient le type de trafic (bits TOS). L'interprétation de ces bits a varié (champ DSCP pour DIFFSERV) et leur signification ne dépassait en général pas un domaine administratif (réseau d'opérateur p.ex.)

IPv6 définit deux champs dans l'entête de base :

— champ traffic class : 8 bits de priorité selon le type de trafic (DIFFSERV)

— champ flow label : 20 bits permettant d'identifier un flux rapidement (compatible avec la réservation/commutation rapide de flux MPLS), RFC-3697 [1] (INTSERV dans une certaine mesure)

Un autre avantage notamment pour les applications 1 vers n (radio, TV) est le véritable support de multi-/any-casting à travers tout le réseau, remplaçant des configurations manuelles complexes.

En pratique toutefois et sans assistance des NGN full, il n'est pas possible non plus en IPv6 de faire du multicast sur l'Internet global.

1. http://www.faqs.org/rfcs/rfc3697.html

Freins à l'acceptation d'IPv6

Freins

- pas d'avantage commercial perçu (pas de *killer app*)
- deux réseaux en parallèle à maintenir, peut-être des années durant !
- compétences à acquérir (plus de 40 RFCs !)
- coûts (formation, matériel)

29

Ce qui se passe chez les FAI : de rien, au tunnel, puis au natif.

Quant aux fournisseurs de service / contenus : ils sont généralement disponibles en v4 *et* v6.

Evolution chez les FAI

- au départ "support" par tunnels IPv6 sur IPv4, évt. efficaces grâce à 6rd (p.ex. Swisscom) qui, sur une infrastructure backhaul IPv4 permet un accès tunnel sur un routeur v4/v6 très près du backhaul
- de plus en plus, migration complète à IPv6 de l'ensemble du backbone, avec IPv4 proposé en compatibilité (méchanismes de transition MAP-T et MAP-E) ; exemple : UPC migre le routeur des clients finaux en v6, avec passerelle transparente v4 (plus d'adresse v4 globalement accessible avec DSLite !)

Sécurité

— attack surface
— problèmes d'implémentations
— problèmes de protocoles
— problèmes de configuration
— problèmes de savoir

30

Amélioration progressive !

Attack Surface Adresses MAC utilisées dans les adresses IPv6
— IP-tracking facilité
— plus un problème depuis que certains équipement comme Android utilisent des adresses MAC aléatoires à chaque connexion on peut aussi activer les *privacy options* ou un serveur DHCPv6

Un firewall IPv4 ne gère pas IPv6 : un firewall IPv6 doit gérer une complexité plus importante (p.ex. entêtes chaînés). Faire en particulier attention aux tunnels IPv6 invisibles pour un firewall IPv4 !

IPv6 est-il plus immunisé contre les scans d'adresses (énumération) ?
— oui : plage d'adresses énormes
— non : multicast "all-hosts", politiques locales uniformes, énumération DNS

Implementation Issues Par exemple : crashes en envoyant des paquets DHCPv6 corrompus sur certains mobiles.

Protocol Issues Microsoft Windows peut résoudre des noms de machine via des procédés spécifiques : un routeur ISATAP peut être intercepté ou du *cache poisoning* utilisé.

Il y a des problèmes potentiels de *cache poisoning* sur mobile, Mac OS X et GNU/Linux Ubuntu/Fedora concernant du *cache poisoning* avec zeroconf ou mDNS.

Configuration Issues Filtrer ICMPv6 sans réfléchir est une mauvaise idée (risque de coinçages liée à des MTU plus faibles en raison de tunnels).

Microsoft Windows dès Vista et 7 implémentent par défaut un tunneling par défaut, Teredo [1], ce qui signifie que toute machine de ce type même derrière un firewall (ne filtrant pas IPv4 UDP port 3544) est accessible directement par IPv6 et les machines dans le même sous-réseau, qu'elles soit Microsoft ou non, sont accessibles entre elles automatiquement en IPv6 (IPv4 multicast).

Knowledge Issues IPv6 peut offrir en standard le chiffrement ESP ou la signature AH, donc certains protocoles ont abandonné l'authentification (p.ex. OSPFv3) : encore faut-il qu'IPsec soit mis en place !

Voir aussi
- http://www.gont.com.ar/talks/lacnog2010/fgont-lacnog2010-ipv6-security.pdf
- http://www.si6networks.com/presentations/h2hc2011/fgont-h2hc2011-ipv6-security.pdf
- http://www.si6networks.com/presentations/HES2012/fgont-hes2012-recent-advances-in-ipv pdf

1. http://en.wikipedia.org/wiki/Teredo_tunneling

Ambiguité

— `http://46.140.72.222:8080/` – accéder à 46.140.72.222 en HTTP sur le port TCP/8080
— `http://::1:8080/` – ambigu ! solution : `http://[::1]:8080/`

31

Conclusion

— IPv6 se déploie au fur et à mesure
— IPv4 est probablement là pour coexister encore longtemps, d'autant plus qu'il n'y a pas encore d'application ou de service dépendants d'IPv6

Références

Voir fin du labo page 56.

Etat du déploiement

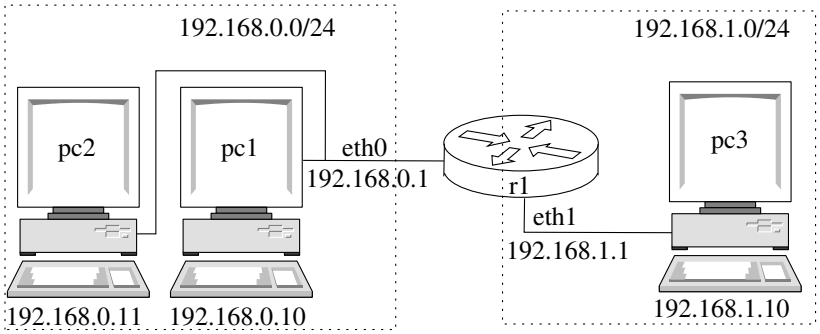
qui	**déploiement**
fournisseurs de contenu (Google, etc)	déjà effectué
gros FAI	déjà effectué
FAI client final	pas encore, ou alors en tunnels efficaces 6rd, voire (UPC) en natif v6 /64 avec émulation v4 CGNAT
entreprise	souvent encore peu déployé : — services en dual-stack ou par reverse proxy — interne souvent en v4 encore, accès Internet v6 inexistant ou via proxy — cloud, voir fournisseurs de contenu ci-dessus
IoT	quelques réseaux de terrain se sont mis au v6 (6LowPAN p.ex.)

Fin de la théorie partie 2 : faire les parties IV, V et questions

Partie I : adresses de portée link-locale – 1.4

32

Réseau de base IPv4 :



objectifs

- découvrir qu'une machine va automatiquement configurer une adresse IPv6 par interface, d'une portée particulière (*link local scope*)
- observer comment, par défaut, on peut utiliser l'adresse MAC [a] de l'interface pour générer cette adresse (SLAAC)
- expérimenter avec la *portée locale lien* (*link local scope*)

a. ce qui pose un problème de protection de la sphère privée

Manipulations et observations : préconfiguration IPv6

1. il peut être utile de télécharger le PDF des labos (à chacun des 4 labos, peut changer), dans la VM Linux, soit via Firefox, soit via un partage avec le host [1] – on peut ensuite copier (sélection avec la souris) et coller (bouton du milieu ou 2 boutons avec les `xterm`)
2. sourcez le script netkit : `source ~/NETKIT/SOURCE_ME`
3. installez le laboratoire : `netkit-lab ipv6`
4. vous pouvez consulter la configuration du laboratoire (`lab.conf`), ainsi que les fichiers de démarrage des machines netkits (`*.startup`)
5. démarrez les machines netkit avec : `lstart -p0` [2]
6. sur `pc1`, consultez la configuration réseau (adresses MAC couche 2 et adresses IP couche 3, v4 et v6) avec la commande `ip addr show`
   - sur l'interface *loopback* `lo`, observez les deux adresses de portée *host* v4 et v6
   - sur l'interface Ethernet `eth0`, observez l'adresse MAC et les adresses IP v4 (configurée par le fichier `pc1.startup` et v6 (préconfiguration automatique)

1. recommandations : faire un `sudo adduser demo vboxfs` puis se relogguer pour accéder facilement au partage ; ne pas mettre le partage sur Microsoft Drive

2. depuis le répertoire du laboratoire ; en cas de nécessité vous pouvez tout arrêter avec `lcrash; lclean`

7. nous allons commencer par forcer une préconfiguration IPv6 en capturant les messages émis, sachant qu'aucun serveur de distribution d'adresse IPv6 (que cela soit DHCPv6 ou un serveur radvd) ne sont activés ; pour ce faire, sur la machine netkit `pc1`, tapez la commande (une seule ligne) suivante : `ip link set down dev eth0; sleep 2; ip link set up dev eth0; tcpdump -e -s 0 -n -w /hostlab/capture.pcap -i eth0`
8. après environ 15 secondes, vous pouvez terminer la capture avec `CTRL-c`, ce qui devrait mentionner 5 paquets reçus [3], puis lancer `wireshark capture.pcap &` depuis le répertoire du laboratoire sur la VM Linux (pas dans la machine netkit pc1) pour observer les trames capturées.
9. quels sont les datagrammes v6 échangés ; quels sont les protocoles associés et que notez-vous concernant leur adresse source et destination au fur et à mesure des échanges indications : en particulier, de quel type sont les adresses de destination ?
10. une préconfiguration IPv6 a eu lieu : voir la commande `ip addr show` [4] : comment a été choisie l'adresse sur l'interface `eth0` ? découpez cette adresse en un *préfixe* (une catégorie [5], ici) et quelque chose qui ressemble beaucoup [6] à l'adresse couche 2 de l'interface
11. quelle est la portée [7] de cette adresse (indication : voir le *préfixe* ci-dessus dans la note de bas de page)
12. observez que `ping6` sur votre propre adresse IPv6 (sans le suffixe `/64`) listée à l'interface `eth0` ne fonctionne pas (erreur *Invalid argument*)
13. observez que `ip -6 addr show` liste les interfaces et les numérote (tout à gauche) ; que se passe-t-il si vous tentez un `ping6` sur votre propre adresse IPv6 en y ajoutant l'identificateur de zone `%N` (`N` : numéro d'interface ou de portée/scope) ou `%eth0` (nom d'interface) ? pourquoi est-ce nécessaire [8] de préciser le *scope* [9] pour cette catégorie d'adresses ?
14. observez que l'adresse `fe80::211:22ff:fe33:4410%eth0` répond mais que vous ne pouvez pas atteindre les DNS publics de Google sous `2001:4860:4860::8888` [10], pourquoi ? (lire le message d'erreur)
15. effacez ou renommez le fichier du laboratoire `capture.pcap` puis relancez la capture `tcpdump` sur `pc1`, puis testez un ping sur l'adresse IPv6 de portée *link-local* de l'inter-

3. au moins 3 de types différents
4. ajouter `-6` comme option pour limiter la sortie aux informations v6 seules ; `ip` est le successeur plus complet de l'ancienne commande générique `/sbin/ifconfig` – plus forcément installée par défaut sous Linux – exemples : pour ajouter une adresse manuelle dans un sous-réseau séparé : `ip addr add 192.168.250.42/24 dev eth0` ; pour obtenir l'adresse MAC on fait `ip link show eth0`
5. https://fr.wikipedia.org/wiki/Adresse_IPv6#Cat.C3.A9gories_d.27adresses
6. https://fr.wikipedia.org/wiki/IPv6#Attribution_des_adresses_IPv6
7. scope ; http://en.wikipedia.org/wiki/IPv6#Addressing
8. avec certaines commandes ping, sans préciser l'identificateur de zone, la première interface est choisie ; sous Microsoft Windows, plusieurs essais sont effectués sur plusieurs interfaces – dans ce laboratoire les commandes utilisées refusent lorsque c'est ambigu
9. comparez cette notion de portée avec l'encapsulation, dans deux classes différentes de programmation orientée-objet, de deux variables de classe de même nom située dans chacune de ces classes : en particulier comment on accède à ces variables de classe depuis l'extérieur (notation double-pointée)
10. cette adresse est de portée ou *scope* global, pas besoin de préciser le numéro de *scope*

face `eth0` de `pc1` depuis `pc2` et observez **comment** fonctionne la conversion d'une adresse IPv6 en couche 2 (vers adresse MAC) ; vous pouvez supprimer l'équivalent du cache ARP en IPv6 avec `ip -6 neighbour flush dev eth0`

(a) quel est le type Ethernet utilisé ici ? (indication : en v4 le type serait `0x0800`)

(b) comment s'appelle le protocole utilisé qui remplace le protocole ARP d'IPv4 ? en quelle couche est-il ?

(c) en particulier vous devriez observer du trafic vers des adresses de type *multicast* : comment l'adresse du groupe multicast est construite ? qui est en membre, et en quoi est-ce mieux que le *broadcast* utilisé en v4 par ARP ? (indication : qui sera dérangé ?)

(d) consultez la version v6 de la table ARPv4 (`ip -6 neighbour`) – si elle est vide, refaites le ping du point 14 et réessayez !

16. quel type de datagrammes a un *Hop Limit* de 255 ? et pourquoi est-ce important pour la sécurité ?
indications :
— ce type de datagramme est détruit s'il est reçu avec un *Hop Limit* résiduel différent
— ce type de datagramme doit avoir une portée d'un sous-réseau, et non pas du réseau couche 3 entier
— analogie : en v4, le protocole ARP ne traverse évidemment pas les routeurs, vu que c'est un protocole couche 2.

Partie II : configuration IPv6 manuelle native – 1.5

objectifs

- configurer manuellement une adresse globale IPv6 au sein d'un sous-réseau IPv6 /64 *natif*
- ajouter une route v6 par défaut (vers le routeur [a] v6)

33

a. souvent appelé passerelle ou *gateway* par abus de langage

Indications Nous allons configurer manuellement les adresses globales (publiques) suivantes, sans utiliser de serveur de configuration d'adresse dynamique (DHCPv6 ou radvd). Le sous-réseau de gauche sera `2001:620:417:2200::/64` et le sous-réseau de droite `2001:620:417:2201::/64`.

machine.interface	adresse v6
pc1.eth0	`2001:620:417:2200::10/64`
pc2.eth0	`2001:620:417:2200::11/64`
r1.eth0	`2001:620:417:2200::1/64`
r1.eth1	`2001:620:417:2201::1/64`
pc3.eth0	`2001:620:417:2201::10/64`

N'oubliez pas de configurer une route par défaut v6 sur chacune des machines (sauf le routeur, qui n'a pas ici accès à Internet).

Dans ce cas, il s'agit d'IPv6 natif (pas de tunnel IPv6 sur IPv4), transporté sur Ethernet dans des trames Ethernet 0x86dd. IPv4 est toujours disponible en parallèle (trames Ethernet 0x0800).

Manipulations et observations

1. configurez toutes les machines y compris le routeur
   - adresse IP et sous-réseau (route implicite) : `ip -6 addr add ADRESSE/64 dev ethX`

   - route par défaut pour `pc[1-3]` : `ip -6 route add default via XXX`, avec XXX à choix soit l'adresse *globale* de l'interface du routeur où l'on est branché, ou l'adresse de portée *lien* correspondante (dans tous les cas sans le `/64`)
   - activez la fonction de routage v6 sur le routeur `r1` : `sysctl net.ipv6.conf.all.forwarding=1`
2. testez que toutes les machines et y compris le routeur peuvent s'entre-pinguer en v6, aussi à travers deux sous-réseaux différents – il n'y a pas besoin de préciser le scope vu que les adresses sont globales
3. testez qu'il n'y a toujours pas de route sur le routeur pour sortir sur Internet, p.ex. avec l'adresse Google `2001:4860:4860::8888`
4. imaginons qu'on désire mettre en place un serveur web, mais qu'il peut être soit actif sur `pc1` ou `pc2`, et qu'on ne désire pas changer ni le DNS ni l'adresse IP dans les clients qui vont y accéder, que pourrait-on faire ? (indication : on peut avoir plus d'une adresse IPv6 active)
5. testez un `mtr` de `pc1` à `pc3` (en v6)
6. sur votre VM Linux (pas netkit), en capturant sur l'interface réseau extérieure (voir `ip addr show`) avec Wireshark [1], obtenez l'adresse IPv6 de `www.switch.ch` avec la commande `host www.switch.ch` : quel est le type d'enregistrement du DNS qui permet de le faire ? cette requête a-t-elle été acheminée en v4 ou en v6 ? est-ce une bonne pratique ?
7. faites `sudo apt-get install whois` et déterminez à l'aide de la commande `whois` à qui appartient le préfixe (sous-réseau /64) ainsi configuré et de quel sous-réseau plus grand il fait partie (nous ne devrions pas utiliser de plages publiques qui ne nous ont pas été assignées, mais ici notre réseau netkit n'est pas branché à Internet)

1. `sudo wireshark` si vous n'avez pas installé Wireshark avec les droits utilisateurs

Partie III : configuration IPv6 automatique – 1.6

objectifs

- découvrir la configuration automatique avec un `radvd`, plus simplifiée qu'avec le protocole DHCPv6
- expérimenter avec les *Privacy Extensions*
- comparez avec votre laptop sous Microsoft

34

Indications Nous allons revenir à la configuration de base v4 du laboratoire puis ajouter un serveur `radvd` sur le routeur `r1`, de manière à ce que tous les PCs puissent obtenir une adresse automatiquement.

`radvd` utilise les messages *Router Advertisement* et est un peu plus simple que le DHCPv6, qui lui sera nécessaire si l'on veut configurer plus que juste le préfixe du sous-réseau et donc l'adresse IP de chaque machine, comme par exemple des adresses IPv6 fixes dépendant de l'adresse MAC ou des paramètres réseaux supplémentaires : serveur de temps, etc.

Manipulations et observations

1. redémarrez le laboratoire (`lcrash; lclean; lstart -p0`) de manière à ce que la configuration manuelle des machines soit perdue
2. configurez les adresses IPv6 de `r1` comme dans la partie II et configurez le serveur `radvd` sur `r1` comme suit :

```
ip -6 addr add 2001:620:417:2200::1/64 dev eth0
ip -6 addr add 2001:620:417:2201::1/64 dev eth1

sysctl net.ipv6.conf.all.forwarding=1

cat > /etc/radvd.conf <<EOF
interface eth0 {
   AdvSendAdvert on;
```

```
    MinRtrAdvInterval 3;
    MaxRtrAdvInterval 10;
    prefix 2001:620:417:2200::/64 {
        AdvOnLink on;
        AdvAutonomous on;
        AdvRouterAddr on;
    };
};

interface eth1 {
    AdvSendAdvert on;
    MinRtrAdvInterval 3;
    MaxRtrAdvInterval 10;
    prefix 2001:620:417:2201::/64 {
        AdvOnLink on;
        AdvAutonomous on;
        AdvRouterAddr on;
    };
};
EOF

    /etc/init.d/radvd start
```

3. obtenez une adresse IPv6 (composée d'une valeur fixée par l'adresse MAC et du préfixe annoncé par le `radvd`) ainsi que la route par défaut en faisant la commande qui suit dans `pc1` et `pc3:ip link set down dev eth0; sleep 2; ip link set up dev eth0;` et consultez les adresses IPv6 globales ainsi obtenues – il se peut que cela marche tout seul d'ailleurs
4. quelle est la route par défaut sur `pc1` (consultez avec `netstat -rn6`) ?
5. expliquez comment ces machines ont construit leur adresse IPv6, et s'il y a un risque qu'un site Internet puisse vous suivre même si vous changez de préfixe (en option, consultez https://home.regit.org/2011/04/ipv6-privacy/ pour voir comment améliorer)
6. sur `pc2`, on va activer les *Privacy Extensions* : `sysctl -w net.ipv6.conf.eth0.use_tempaddr=2` puis faire comme pour les autres machines, mais ajouter une capture `tcpdump` comme on avait fait précédemment (voir point 7 en page 40, mais également capturer sur `r1.eth0`) : y-a-t-il une différence dans la façon dont l'adresse IPv6 est construite, et est-ce mieux pour la sphère privée ?
7. analysez le protocole utilisé pour obtenir le préfixe et la route par défaut (au moins 2 types de messages échangés)
8. comment une machine vérifie-t-elle qu'une adresse n'est pas déjà prise (c'est le même protocole qu'en préconfiguration *link-local* `fe80`)
9. constatez que les adresses de portées *link-local* sont toujours présentes
10. testez que les pings fonctionnent comme avant entre `pc2` et `pc3`
11. en option, comparez avec votre laptop sous Microsoft Windows, si il a déjà accès IPv6 : comment l'adresse IPv6 est-elle configurée et les *Privacy Extensions* sont-elles utilisées ? Que disent http://www.whatismyipaddress.com et http://ipv6-test1.alphanet.ch

Partie IV : transport d'IPv6 – 1.7

objectif

— mettre en place un tunnel pour accéder IPv6 via un réseau IPv4

35

Indications Lorsque l'on n'a pas une connexion *native* IPv6, il est possible de transporter des datagrammes IPv6 de différentes manières, par exemple à l'intérieur de datagrammes IPv4.

Les protocoles les plus classiques pour IPv6-sur-IPv4 sont 6rd, SIT ou GRE. Ici, nous allons utiliser un simple tunnel SIT.

Nous allons considérer notre réseau, avec le routeur `r1` uniquement en v4, et `pc1` et `pc3` en v4 et en v6.

Manipulations et observations

1. redémarrez le laboratoire (`lcrash; lclean; lstart -p0`) de manière à ce que la configuration manuelle des machines soit perdue
2. vérifiez que ni `pc1`, `pc3` ni `r1` n'ont d'adresse IPv6 globales [1], mais seulement IPv4
3. configurez `pc1` comme suit :

```
# création du tunnel SIT
# (si nécessaire: supprimer avec ip tunnel del test1)
# pour rappel: les backslash signifient tout taper sur la
# même ligne
ip tunnel add test1 mode sit local 192.168.0.10 \
                             remote 192.168.1.10 \
```

1. il y aura des adresses de portée lien

```
                                ttl 255
ip link set test1 up

# configuration de l'adresse et de la route v6
ip -6 addr add 2001:520:417:2200::1/64 dev test1
# ip route add ::/0 dev test1 # si l'autre bout a accès Internet
```

4. configurez `pc3` comme suit :

```
# création du tunnel SIT
# (si nécessaire: supprimer avec ip tunnel del test1)
ip tunnel add test1 mode sit local 192.168.1.10 \
                             remote 192.168.0.10 \
                             ttl 255
ip link set test1 up

# configuration de l'adresse et de la route v6
ip -6 addr add 2001:520:417:2200::2/64 dev test1
# ip route add ::/0 dev test1 # si l'autre bout a accès Internet
```

5. lancez une capture `tcpdump` sur `pc3`, sur l'interface `eth0` [2]
6. faites `ping6 2001:520:417:2200::2` depuis `pc1`
7. expliquez l'encapsulation IPv6 dans IPv4 (p.ex. grâce à Wireshark sur la capture acquise)
8. y-a-t-il une différence de délai entre un `ping` en v4 (vers l'adresse v4 de `pc3`) et en v6 ? et si le tunnel traverse de nombreux routeurs et liens Internet ?
9. que pensez-vous de la sécurité de ce tunnel ? (est-ce un vrai VPN authentifié et chiffré, ou juste de l'acheminement, en clair, de datagrammes v6 sur v4 ?)
10. y-a-t-il un impact sur le MTU disponible pour les applications, par comparaison à l'IPv6 natif ?

2. si vous capturez sur `test1` vous ne verrez pas l'encapsulation mais juste du v6

Partie V : analyse de trames – 1.8

Dans le Gitlab, vous trouvez une archive avec plusieurs fichiers de capture pcap, lisible avec Wireshark ou tcpdump.

Expliquez ce qui s'est passé, à chaque fois, en consultant le README.

36

Partie VI : certification (optionnelle) – 1.9

Effectuez tout ou partie du programme de certification gratuit d'Hurricane Electric ! Vous pouvez la faire sous GNU/Linux, Microsoft Windows ou Mac OS X.

Est optionnel même si vous rendez le rapport (bonus).

37

Indications Cette partie ne peut s'effectuer que chez vous ou éventuellement à la HE-Arc sans VPN : la liaison ADSL du laboratoire de téléinformatique ne dispose que d'une seule adresse IP fixe, et le transport IPv6-sur-IPv4 utilisé par Hurricane Electric ne l'autorise pas, de plus notre routeur ADSL ne supporte pas le protocole 41. En théorie ça devrait marcher sur réseau jaune (77), à vérifier.

Inscrivez-vous sur Hurricane Electric (un tunnel broker pour faire de l'IPv6 sur IPv4, voir http://ipv6.he.net/certification/).

Commencez par répondre aux questions. Cela vous donnera un niveau de certification minimal et vous pourrez ensuite créer un tunnel.

ATTENTION : le mode de transport utilisé par Hurricane Electric ("protocol 41") n'est pas forcément compatible avec un modem ADSL en mode routeur (il vous faut peut-être activer ce mode). Par contre, avec un modem ADSL en mode modem ou une liaison CATV (câble téléréseau), cela devrait marcher sans problème.

Après avoir répondu aux questions, vous pouvez ensuite cliquer "this link", vous logguer sous tunnelbroker.net, créer un tunnel (en indiquant votre adresse IPv4 externe, voire votre adresse interne si vous avez un NAT), cliquer sur le tunnel, puis voir les commandes proposées (qui dépendent de l'OS et de sa version).

Si cela ne marche pas, essayez de capturer quand même et tentez d'expliquer.

Questions – 1.10

Toutes ces questions ainsi que le laboratoire (grandes lignes) font partie de la matière du prochain questionnaire.

Aidez-vous des références en page 56.

1. pour les 3 portées IPv6 ci-dessous, indiquez dans quel cas chacune est utilisée : (vous pouvez aussi réfléchir à l'équivalence dans le monde IPv4, si elle existe)

scope (portée)	**exemple d'application pratique**
global	
unique (site) local (ULA)	
link-local	

2. voici des adresses de machine, indiquez la portée de ces adresses :

adresse	**scope** (portée)
`2001:4dd0:ff00:36::2`	
`fe80::21c:c4ff:fecc:c76d`	
`fdfe:dcba:9876::2`	
`::1`	

3. votre FAI vous a remis une plage d'adresse IPv6 : `2001:1702:4::/48`

   (a) de combien d'adresses, au total, dispose cette plage (ce sous-réseau) ? est-ce beaucoup ?

   (b) combien de sous-réseaux votre entreprise pourrait-elle définir dans cette plage, sachant qu'en règle générale un sous-réseau est un /64 ?

   (c) votre entreprise a juste besoin de 3 sous-réseaux de taille classique pour le moment : choisissez-en 3 au sein de la plage ci-dessus et donnez les 3 adresses de 3 sous-réseaux en notation CIDR – faites au plus simple !

   (d) montrez approximativement quelle serait l'adresse IP de votre laptop, s'il était situé dans le premier de ces sous-réseaux, en sachant qu'il n'y a pas de serveur DHCPv6 et que le respect de la sphère privée n'est pas important ici

(e) et si nous sommes concernés par le respect de la vie privée ?

4. quelle est la taille minimale usuelle d'un sous-réseau IPv6 ?

5. pourquoi l'on ne peut atteindre une adresse IPv6 de portée locale-lien qu'en indiquant le scope de l'interface considérée ?

6. injecter du trafic du protocole NDP depuis un *autre sous-réseau* est-elle une attaque de sécurité fonctionnelle ? sinon pourquoi ? (indication : l'attaqué peut-il détecter que ce datagramme a traversé un routeur alors qu'il ne devrait pas ?)

7. pourquoi le support du *multicasting* est obligatoire pour toute machine IPv6 ?

8. comment peut-on émuler un *broadcast* IP en IPv6 ?

9. pourquoi l'abbréviation d'adresse IPv6 `2001:520:417:2201::42::1` est interdite ?

10. quelles sont les solutions qui permettent la coexistence d'IPv4 et d'IPv6, en plus de machines et de réseaux totalement *dual-stack* (natifs) ?

11. pourquoi l'IPv6 non natif est, en général, une mauvaise idée (deux raisons) ?

12. pourquoi peut-il être dangereux de laisser l'IPv6 activé sur une machine dans un réseau IPv4 en présence d'un firewall IPv4 protégeant ce réseau IPv4 ? (indication : *Teredo*)

13. comment fonctionne la détection qu'une adresse IPv6 est déjà utilisée dans un sous-réseau donné ?

14. à quelles adresses *multicast* un routeur doit-il répondre ?

15. pour quelle raison le checksum d'entête a été supprimé d'IPv6 ? (indication : que fait la couche 2 Ethernet par exemple ?) (optionnel : y voyez-vous un risque résiduel [1] malgré cela ?)

1. les checksums d'entête TCP et UDP existent toujours, sont obligatoires, et intègrent également un résumé de l'entête v6

Questions pour le rapport – 1.11

Ces questions ne font pas partie de la matière pour le questionnaire.

1. reprenez la question 3 de taille de plage en page 50 : pour se représenter ce nombre qui est gigantesque, plusieurs idées, faites-en une seule :
   (a) le nombre d'Avogadro [2] définit, en chimie, le nombre d'entités qui se trouvent dans une *mole* : si l'on avait autant d'atome de carbone que d'adresses dans cette plage IPv6, quelle serait la masse approximative de carbone résultante ?
   (b) calculez la hauteur produite en empilant N pièces de deux francs, avec N le nombre d'adresses IPv6 dans cette plage – pour un bon résultat, convertissez en années-lumières, puis en diamètres de notre galaxie, la Voie lactée.
   (c) calculez la densité d'adresses de ce sous-réseau par millimètre carré de la surface de la Terre

2. IPv4 peut, optionnellement, implémenter la notification de congestion via le protocole ECN [3] (plutôt que de simplement détruire les datagrammes une fois que c'est trop tard). Quel mécanisme est utilisé en IPv6 ?

3. en supposant un flux de données utiles (couche 7) de 1300 octets, comparez l'efficacité intrinsèque des protocoles IPv4 et IPv6, en supposant l'usage du protocole de couche 4 UDP.

2. https://fr.wikipedia.org/wiki/Nombre_d'Avogadro – il permet de lier la masse atomique d'un atome de carbone (isotope 12) carbone, environ 12, à la masse d'une mole de carbone soit 12 grammes

3. http://cvs.alphanet.ch/cgi-bin/cvsweb/~checkout~/schaefer/public/cours/EISI/teleinfo_2/cours/ECN/RELEASES/ecn_complement.pdf?rev=HEAD;content-type=application%2Fpdf

4. décrivez divers modes de qualité de service en IPv6

5. pour quelle raison le transport d'IPv6 en MPLS est facilité ?

6. en quoi consiste le mode 6to4 http://wiki.linuxquestions.org/wiki/6to4 et quels sont ses avantages et inconvénients ?

7. le NAT/PAT est parfois utilisé, en IPv4, pour assigner divers services (ports) à divers serveurs. Comment résoudre le problème sans NAT/PAT en IPv6 ?

8. Swisscom propose des tunnels efficaces IPv6 via 6rd, en consultant le schéma d'adressage http://1.bp.blogspot.com/-MaC9m1pD2Rg/T8t1BfGNomI/AAAAAAAABvA/jR_mmRXdNko/s640/IPv6+Format.PNG expliquez le principe

9. comment fonctionnent les options IPv6 ? (indication : où sont-elles stockées en v6, et y-a-t-il un ordre spécifique et pourquoi ?)

10. en Ethernet classique sur IPv4, les *switches* peuvent être configurés pour détecter des réponses de faux serveurs DHCP et les supprimer (*DHCP snooping*) : qu'existe-t-il en IPv6 ?

11. comment choisit-on le préfixe d'un sous-réseau privé IPv6 (avec des adresses de portée *Unique site Local (ULA)*, voir http://tools.ietf.org/html/rfc4193#section-3.1) et pourquoi le risque d'adresses similaires avec d'autres sous-réseaux du même type est minime contrairement à IPv4 ?

Références – 1.12

- http://en.wikipedia.org/wiki/IPv6
- RFC-4193 [a] : Unique local IPv6 unicast addresses, générateur en-ligne [b] et code Python [c]
- IP version 6 addressing RFC-4291 [d]
- http://tldp.org/HOWTO/Linux+IPv6-HOWTO/
- utilitaire http://www.deepspace6.net/projects/ipv6calc.html
- http://www.gentoo.org/doc/en/ipv6.xml

38

a. http://tools.ietf.org/html/rfc4193
b. https://cd34.com/rfc4193/
c. https://github.com/andrewlkho/ulagen
d. http://rfc-ref.org/RFC-TEXTS/4291/index.html

- http ://www.ipv6.org/
- cours ECN de Marc Schaefer (plus donné), voir http://cvs.alphanet.ch/cgi-bin/cvsweb/~checkout~/schaefer/public/cours/EISI/teleinfo_2/cours/ECN/RELEASES/ecn_complement.pdf?rev=HEAD;content-type=application%2Fpdf
- IPv6 Essentials, Sivia HAGEN, ISBN 978-0-596-10058-2, O'Reilly, 2nd edition, 2006.
- RFCs
  - RFC-2460 - Internet Protocol, Version 6 (IPv6) Specification
  - RFC-4291 - IP Version 6 Addressing Architecture
- utilitaire ipv6calc (calcul et validation d'adresses)
- http://www.shorewall.net/Linuxfest-2009.pdf
- http://www.blogg.ch/uploads/IPv6-deployment-for-the-IPv4-clueful-he
  pdf
- http://www.blogg.ch/uploads/Native-IPv6-via-xdsl-how-to-tweak-your-
  pdf
- sécurité
  - http://www.cpni.gov.uk/Documents/Publications/2011/2011mar22-info
    viewpoint_security_implications_of_IPv6.pdf
  - http://www.si6networks.com/publications/articles.html
  - http://www.si6networks.com/presentations/hacklu2011/fgont-hacklu2
    pdf
  - http://searchenterprisewan.techtarget.com/tip/IPv6-firewall-secur
  - DHCPv6/RA-snooping (similaire au DHCP snooping en IPv4 par les switches) http:

//tools.ietf.org/id/draft-gont-opsec-dhcpv6-shield-00.txt
— http://www.si6networks.com/presentations/hip2012/fgont-hip2012-hacki
pdf
— http://ipv6securitylab.org/ipv6toolbox.html
— config OpenWRT : http://bruno.kerouanton.net/blog/2012/01/14/ipv6-et-moi
— journée de test 2012 : http://www.pcinpact.com/news/68388-ipv6-launch-day-de
htm
— MOOC https://www.fun-mooc.fr/courses/course-v1:MinesTelecom+04012+
session04/info
— livre http://www.editions-eni.fr/livre/ipv6-principes-et-mise-en-oeuvr
accessible via https://aai-logon.hes-so.ch/eni

2. Labo QoS

Objectifs – 2.1

39

— découvrir comment on peut assurer la qualité de service au sein d'un réseau d'opérateur
— expérimenter un cas concret avec l'approche *differentiated services* : le *traffic shaping*, dans le cas d'une liaison Internet classique à prioriser
— concevoir une application de simulation de l'approche de validation du contrat de service (UPC, *baquet*) – voir page 65 et Gitlab baquet.md

support de cours complémentaire : qos.md (Gitlab).

La qualité de service La QoS est une mesure de la capacité d'un *canal* [1] à transmettre des données conformément aux besoins de l'utilisateur.

Par exemple, on peut utiliser un réseau IP pour transmettre de la voix, des images fixes et mobiles (vidéo), des données informatiques. Chacun de ces flux a des **besoins** différents :

— les données informatiques demandent des débits minimaux, un taux d'erreur faible
— la téléphonie demande un débit constant, un délai acceptable ($<$ 150ms de bout en bout) et ne variant que peu (éviter la gigue de phase ou *jitter*), la préservation de l'ordre des données et un taux d'erreur acceptable
— la vidéo compressée demande un débit moyen élevé, la possibilité de dépasser ce débit lors de courtes pointes de trafic, et ne demande un délai faible que pour les applications de vidéo conférences.

L'évaluation de la QoS se base sur certains **critères** comme par exemple : débit, taux d'erreur et de perte, délai, gigue de phase (*jitter*, variation du délai), disponibilité, garantie de préservation d'ordre des paquets, etc.

1. ici, dans le contexte des réseaux d'opérateur, un flot virtuel implémenté au-dessus d'un réseau complexe à paquets (avec ou sans connexion)

Principes de la QoS – 2.2

Sans QoS : on peut aussi facturer ce qui se passe (*95-percentile*), en acceptant des surcharges temporaires (surdimensionnement), tout en augmentant régulièrement la capacité du réseau en fonction de la progression des besoins réels.

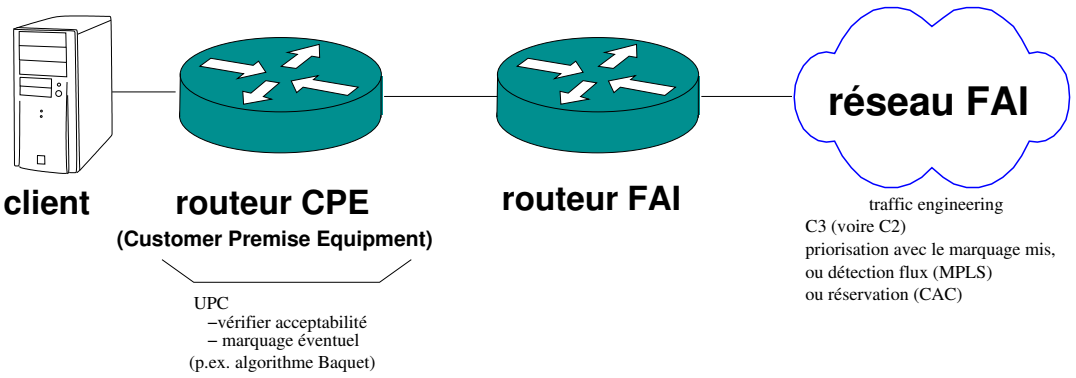
Avec QoS (besoins, critères, contrat SLA) : assurer la QoS désirée (*traffic engineering*)

— réservation (*integrated services*, *CAC*)

— priorisation (*differentiated services*, *traffic shaping*)

40

option : valider les flux client à l'entrée du réseau (*UPC*)



Modèles d'exploitation Les modèles possibles sont :

1. facturer ce qui se passe, ce qui à long terme permet un lissage des infrastructures nécessaires, mais ne protège pas contre des surcharges temporaires (surdimensionnement nécessaire) : c'est souvent le modèle retenu par les opérateurs Internet ou les datacenters pour les clients *commerciaux*
2. dans un réseau d'opérateur, on applique souvent le traffic engineering : on définit des critères de QoS et, l'une des deux stratégies suivantes :
   (a) *réserver*, à l'acceptation de connexion ou de flux, le débit nécessaire (CAC, *Connection Admission Control*), refuser le flux ou la connexion si le réseau ne peut l'assurer – *integrated services*
   (b) prioriser en cas de surcharge et jeter ou ralentir le trafic non prioritaire avec du *traffic shaping – differentiated services*

Dans tous les cas, on peut avoir intérêt à vérifier que certains critères ne sont pas dépassés par l'utilisateur à l'*entrée du réseau* (UPC, *User Parameter Control*) ; cette validation [1] peut néanmoins se limiter à marquer le trafic qui violerait des critères pour le jeter en priorité sur un nœud du réseau lorsque cela devient nécessaire

1. utilisant par exemple le Baquet, voir page 65.

Modèle simple de facturation a posteriori Un modèle simple d'exploitation est le *95 percentile* : on va simplement mesurer le débit toutes les 5 minutes, et à la fin du mois on supprime les 5% des mesures les plus grandes : la facturation intervient sur le débit le plus grand restant.

Concrétisation des besoins du client dans un SLA Les *besoins du client*, concrétisés sous la forme de valeurs minimum et maximum pour tout ou partie des critères (voir page 58), sont formalisés en un **contrat**, le *Service Level Agreement* (SLA) entre l'utilisateur et l'exploitant du réseau (opérateur). La portée de ce contrat ne dépasse jamais les bornes du réseau de l'opérateur : en particulier, des SLA pour tout Internet sont aujourd'hui illusoires [2], à part le simple et classique *best effort*.

Assurer la qualité de service par traffic engineering Le *traffic engineering* est une technique utilisée pour analyser, classifier, contrôler et réguler la distribution du trafic au sein d'un réseau, et notamment pour garantir le traitement différencié de flux multiplexés sur un même canal.

Un opérateur peut assurer le SLA demandé par différentes méthodes (qui peuvent être combinées) :

surdimensionnement la performance du réseau est sur-dimensionnée (routeurs, switches, lignes) : lors d'un usage normal du réseau, la qualité de service sera facilement atteinte ; le coût prohibitif de cette méthode fait qu'elle n'est en général appliquée qu'au sein d'un *réseau d'entreprise*. Elle peut utiliser des VLANs et de la priorisation simple sur les équipements (voir méthode *differentiated services*) lorsque le trafic augmente.

integrated services le débit / les chemins sont *réservés* en fonction des flux (acceptés au sein d'un protocole d'admission – *Connection and Admission Control*, CAC), ce qui supprime tout *overbooking* et garantit le débit et le délai par des moyens déterministes : cela exige des protocoles en couches 2 et/ou 3 capables de réserver du débit et des chemins, comme par exemple ATM, RSVP, MPLS, PBT [3] ... : l'inconvénient de cette méthode est sa complexité. Les flux concrets des clients doivent être *validés* en permanence à leur entrée pour vérifier qu'ils ne dépassent pas le profile négocié (*Usage Parameter Control*, UPC)

differentiated services l'approche est ici une *priorisation* du trafic sur les équipements, que l'on appelle aussi *traffic shaping* : l'objectif est d'assurer en général les critères par des moyens plus simples, par queues de priorités : le trafic est classé par type de flux, marqué (p.ex. les bit TOS ou DIFFSERV de IP ; 802.1p/q en couche 2 ; labels MPLS) et priorisé sur les équipements (routeurs, switches). L'avantage de cette approche est évidemment qu'elle est simple, et fonctionne en général bien. Le RFC-3290 (*An Informal Management Model for Diffserv Routers*) documente un modèle applicable aux équipements réseau et définissant des concepts comme des filtres, des queues, des ordonnanceurs de trafic, etc.

De fait, l'approche *differentiated services*, la plus couramment mise en œuvre, nécessite un surdimensionnement léger, car elle ne se comporte pas bien en cas de surcharge (perte de données des trafics moins prioritaires). Elle se combine souvent en plus à des éléments de la solution d'*integrated services* (détection de flux et de profiles, validation des profiles négociés UPC, évt. réservation partielle

2. des tentatives ont été effectuées pour des interconnexions de réseaux d'opérateur utilisant MPLS, et potentiellement l'interconnexion NGN-full des NGN-IMS devraient offrir de la qualité de service à travers les opérateurs compatibles

3. https://fr.wikipedia.org/wiki/Provider_Backbone_Bridge_Traffic_Engineering, utilise les VLANs 802.1q comme des tunnels spécifiant le chemin

de chemins). Par exemple, une approche mixte de priorisation sur le *backhaul* (réseau d'amenée) et de réservation sur le core MPLS peut être appliquée.

Validation des flux des clients A l'entrée dans le réseau d'opérateur, les flux des clients doivent être validés (UPC) pour détecter des dépassements du contrat de service.

S'ils le dépassent, l'excédent est soit directement éliminé ou marqué comme retardable ou éliminable en cas de congestion sur un équipement au sein du réseau de l'opérateur.

Des pics temporaires sont toutefois acceptés s'ils rentrent dans les paramètres du SLA – voir l'exemple du *baquet* en page 65 – dans ce cas, les paquets acceptés sont envoyés, sans attendre, dans le réseau.

Traffic shaping Un cas particulier du traffic engineering est le *trafic shaping*, qui consiste à intervenir sur le trafic en marquant, retardant légèrement, voire supprimant, le trafic excédentaire.

Le trafic excédentaire est défini, lorsqu'il y a une surcharge, comme :

— du trafic de priorité inférieure, alors que du trafic de priorité supérieure existe (priorisation)
— du trafic excédant certaines caractéristiques, comme le débit moyen, la taille d'un burst, etc (UPC)

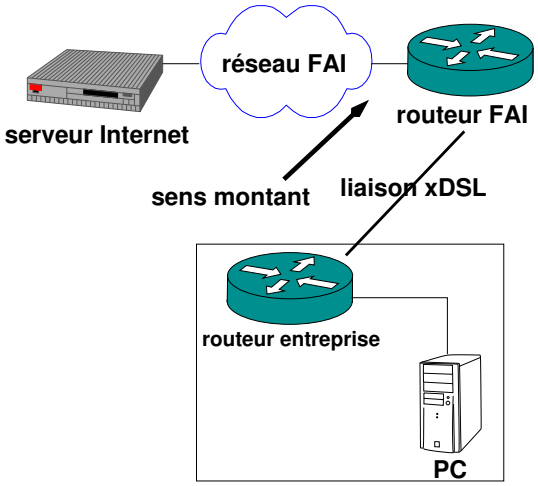
On découvre très vite la notion de *queue de priorité* et d'algorithme de *scheduling*.

Il y a plusieurs implémentations [4] possibles pour les queues de priorités et leurs algorithmes de scheduling, variant suivant leur stratégies (marquer, retarder, supprimer) et l'impact sur le trafic des classes de priorités inférieures en présence de surcharge (blocage, ralentissement). On peut rapprocher ces concepts à ceux du scheduler d'un système d'exploitation (real-time avec interblocage pour les systèmes temps-réel, time-shared avec quanta comme sous UNIX ...).

4. voir par exemple l'implémentation du kernel Linux de queues de priorités pour le traffic shaping sous http://linux-ip.net/articles/Traffic-Control-HOWTO/classless-qdiscs.html

Differentiated services : cas concret – 2.3

Soit la liaison Internet xDSL suivante :



41

objectif : améliorer la performance des applications Internet avec du *traffic shaping*

moyen : machines virtuelles *netkit*

Cas idéal Dans le cas idéal, le routeur du côté fournisseur va prioriser le trafic descendant et le routeur du côté entreprise va prioriser le trafic montant. En effet, il vaut mieux prioriser à l'entrée qu'à la sortie. Nous allons procéder différemment, en supposant en première approche que le routeur du fournisseur n'est pas modifiable et faire au mieux sur le routeur de l'entreprise.

Caractéristique d'une liaison Internet classique En général, les opérateurs dimensionnent les routeurs vers le client avec de grandes queues de traitement. Par défaut, cela implique de grands délais en cas de surcharge de trafic sur la liaison vers le client.

Simulation Ci-après, on appellera *host* la machine Linux qui contient l'installation de *netkit*, et *machines virtuelles (VM)* les machines lancées par netkit, soient :

`serveurinternet` un serveur quelque part sur Internet

`routeurfai` le routeur du FAI

`routeurentreprise` le routeur de l'entreprise

`pc` le PC dans l'entreprise

Manipulations et observations S'il faut lancer plusieurs commandes en parallèle, utilisez l'outil de multiplexeur de terminal `screen` (voir page 65).

1. lancez un terminal (p.ex. `xterm` ou `mate-terminal` avec ALT-F2)
2. puis :
```
# à faire à chaque terminal
source ~/NETKIT/SOURCE_ME
# une fois le labo installé: cd ~/NETKIT/LABOS/qos/qos

# à faire une fois (installation du labo)
mkdir -p ~/NETKIT/LABOS
netkit-lab qos
cd qos
```
3. consultez le fichier `lab.conf` et comprenez comment les liaisons entre machines virtuelles sont configurées
4. lancez le laboratoire avec `lstart -p0` [1] (pour terminer : `lcrash; lclean`)
5. observez le lancement des 4 machines virtuelles et placez-les comme désiré sur l'écran
6. consultez le fichier `routeurfai.startup` et visualisez les commandes – agissant sur le système de queue du routeur – permettant de définir des grandes queues de traitement, via un Token Bucket Filter (baquet) avec un délai (voir page 65) – *traffic shaping*
   (a) sur `routeurfai`, effectuez `tc qdisc list`
   (b) que représentent `rate` et `burst` par rapport au modèle du baquet ?

   ensuite, vous pouvez minimiser le `routeurfai` – nous n'y changerons rien
7. testez que `pc` peut atteindre `serveurinternet` en pinguant 192.168.1.42 depuis la fenêtre de `pc`
8. vérifiez que vous pouvez accéder au répertoire du labo situé sur le *host*, avec `ls /hostlab` depuis une des VMs
9. consultez la section *Evaluation de performance* (page 64) et évaluez-la :
   (a) mesurez le débit montant entre `pc` et `serveurinternet`
   (b) mesurez le débit descendant
   (c) mesurez les `ping`
      — à vide
      — avec du débit descendant (voir outil `screen` à la page 65 pour pouvoir lancer simultanément le trafic et le `ping`)
      — avec du débit montant

      (indication : prenez toujours le maximum, en enlevant les valeurs aberrantes)
10. répondez aux questions :
    (a) est-ce que la performance, chaque sens pris séparément, correspond à celle prévue ? (cf la configuration `routeurfai.startup`)
    (b) si vous faites des transferts dans les deux sens, vous devrier observer que saturer le débit montant ralentit également le débit descendant, pourquoi ? (indication : fonctionnement de TCP)
    (c) qu'observez-vous si vous faites des `ping` simultanément à des transferts ? expliquez.
11. uniquement si vous rendez le rapport : avec l'outil `gnuplot` sur le Linux et sa commande `load "plot.gnu"` vous pouvez afficher en boucle les statistiques sur la latence, par exemple avec le fichier `plot.gnu` suivant :

1. si votre VM est particulièrement lente, sans l'option de parallélisation `-p0`

```
plot "plot.dat" with lines title "latency [ms]"
pause 5
reread
```

si vous stockez les résultats `ping` en temps réel d'une VM netkit ainsi :

```
(DELAY=5
 START=$(date '+%s')
 while :
 do
    PING=$(ping -c 1 -i 1 192.168.1.42 | tail -1 | awk -F / '{print $5;}')
    if [ "$PING" != "" ]; then
       echo "$(( (`date '+%s'` - $START) / $DELAY )) $PING"
    fi
    sleep $DELAY
 done) > /hostlab/plot.dat
```

12. uniquement si vous rendez le rapport : utilisez la commande `iperf -u` pour simuler du trafic UDP et évaluer les pertes et éventuellement les réordonnancements, en spécifiant le débit indiqué [2], puis en l'augmentant pour voir les dégats :

```
# montant
serveurinternet# iperf -s -u
pc# iperf -c 192.168.1.42 -u -b 200k

# descendant
serveurinternet# iperf -c 192.168.2.10 -u -b 1000k
pc# iperf -s -u
```

Evaluation de la performance Outils disponibles pour TCP : `nc` (netcat) permet d'établir des flux réseau TCP, voire UDP et la commande `buffer` permet d'évaluer la performance en temps réel – n'hésitez toutefois pas à la relancer pour éviter l'effet cumulatif du passé.

côté `serveurinternet` (à lancer *d'abord*, car ce sont des serveurs TCP qui attendent une connexion) :

— réception continue de données du `pc` :

```
nc -l -p 7000 -q 0 | buffer -S 32768 -s 32768 -o /dev/null
```

— envoi continu de données vers `pc` :

```
nc -l -p 7001 -q 0 < /dev/zero
```

N'hésitez pas à utiliser la boucle `while :; do ... done` pour relancer ces deux services dès que le client termine.

côté `pc` (à lancer rapidement ensuite) :

— envoi continu de données vers `serveurinternet` :

```
buffer -S 32768 -s 32768 -i /dev/zero \
   | nc -q 0 192.168.1.42 7000
```

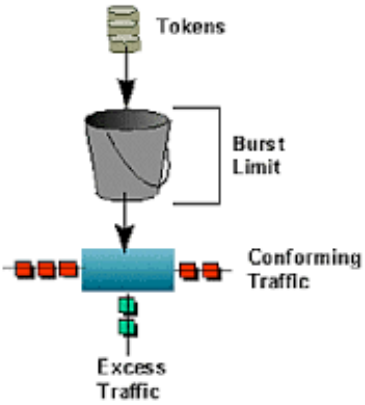
(les valeurs de débits affichées par ce buffer se stabiliseront moins vite qu'un buffer côté `serveurinternet`, en raison des tampons de TCP)

— réception continue de données de `serveurinternet`

```
nc -q 0 192.168.1.42 7001 \
   | buffer -S 32768 -s 32768 -o /dev/null
```

2. UDP, contrairement à TCP, n'a pas d'adaptation de débit en fonction des délais et pertes, et on prend une marge de sécurité car iperf envoie en burst !

Algorithme de lissage du baquet Le baquet, ou *token bucket* permet de lisser du trafic en acceptant des pointes momentanées de trafic si le trafic moyen reste acceptable. Il est appliqué notamment à l'entrée du réseau (UPC) mais peut aussi être appliqué comme critère d'une queue de priorité sur un routeur par exemple.



NB : le baquet est un modèle : les paquets ne circulent pas *via* le baquet : ils passent soit *sans délai*, ou soit sont détruits (ou marqués pour une destruction en priorité, ultérieure). Une variante peut associer un court délai avant la destruction.

Les paramètres du baquet sont :

— le débit de sortie (débit moyen)
— la hauteur d'eau stockable dans le seau (ce qui dépasse le débit moyen) : capacité de résister aux bursts ou aux variations de débit d'entrée

Le baquet est normalement vide. En cas d'un pic (*burst*) de trafic, il est possible que le baquet puisse stocker le burst, puis si le débit en entrée le permet, l'absorber à terme. De même, des variations de débit moyen d'entrée sont possibles dans une certaine limite. Si le burst est trop important, ou si le débit d'entrée est en moyenne supérieur au débit de sortie, le baquet débordera et des paquets seront détruits.

Multiplexeur de terminal screen `screen` permet de multiplexer plusieurs terminaux dans le même terminal – ici la même fenêtre `xterm` – qui survivent la fin d'un terminal.

Son principe est assez simple : on lance `screen`, on tape la touche ENTREE pour lancer un premier terminal. On peut créer un shell, par défaut dans sa fenêtre avec CTRL-A c (CTRL-A suivi de `c`). On la supprime avec CTRL-A k. On navigue dans les fenêtres (ou sous-fenêtres, voir fonction de *split screen* ci-dessous) avec CTRL-A ESPACE (ou en indiquant leur numéro), on liste les fenêtres avec CTRL-A w. L'aide est disponible avec CTRL-A ?.

Le mode *split screen* est aussi intéressant avec ses commandes CTRL-A suivi de S (*split* vertical ; c'est une *majuscule*, sensible à la casse !), X (supprimer), TAB (naviguer).

Attention : CTRL-A x (minuscule) verrouille screen : le mot de passe root est root.

Référence : `http://aperiodic.net/screen/quick_reference`

Traffic shaping simple – 2.4

Introduction

On a constaté le problème de *ping*, qui est dû à une surcharge et à des queues de traitement trop grandes. Pour la performance montante, cela ne concerne pas le routeur du FAI, mais bien le débit en sortie du routeur de l'entreprise !

42 Une première mesure simple est donc la limitation du débit montant à la sortie du routeur d'entreprise, ce qui devrait avoir pour résultat de stabiliser, dans une certaine mesure les délais.

Limitation du débit montant

C'est ici facile : changer la configuration du routeur d'entreprise pour limiter le débit montant, puis prioriser le trafic *ping* (ou voix-sur-IP) par exemple.

voir introduction page 67.

Le problème Nous avons constaté qu'un trafic informatique peut perturber le trafic (ici en ICMP). Le rôle du traffic engineering est d'assurer que les différents SLA sont respectés. Le problème qui se pose ici est multiple :

1. il y a un débit de sortie qui est plus faible que le débit de l'interface
2. mais surtout : l'équipement du fournisseur (le simulateur ici) a des queues de traitement trop grandes
3. enfin, le trafic audio n'est pas géré de manière prioritaire par rapport au trafic informatique.

Il est évident que ce qu'il faudrait faire, ce sont des queues de priorité à la montée (sur le routeur d'entreprise), mais aussi à la descente (sur le routeur du FAI). Malheureusement, bien souvent, ce dernier n'est pas modifiable. Nous allons d'abord résoudre ce qui peut l'être, puis réduire le problème par diverses astuces !

Manipulations et observations : limitation du débit montant

1. sur le `routeurentreprise`, limitez le débit montant, en appliquant une queue TBF [1] sur l'interface sortante, avec un très faible délai de stockage, par exemple 5 ms :
   `tc qdisc add dev eth0 root tbf rate 400kbit latency 5ms burst 1540`
   (pour supprimer, changer le `add` en `del`)

1. Token Bucket Filter, donc très proche du baquet décrit en page 65

2. testez que les délais mesurés (ici ping ICMP, pour p.ex. simuler de la voix-sur-IP) sont meilleurs en présence de trafic informatique TCP montant, avec cette limitation
3. montrez dans quelle mesure le trafic informatique TCP est ralenti par cette limite et expliquez pourquoi.
4. consultez les statistiques des queues avec `tc -s qdisc ls`

Traffic engineering / shaping sous Linux Linux propose une implémentation complète de *traffic engineering* / *traffic* shaping que nous allons mettre en œuvre.

Quelques informations rapides :

— sauf dans de rares cas, on limitera toujours la sortie vers une interface réseau. En effet, limiter l'entrée n'a pas vraiment de sens : les paquets ont déjà occupé le débit disponible.
— pour limiter le débit d'une interface à un maximum défini, le moyen classique est le *baquet* ou *Token Bucket* (voir page 65), qui garantit en moyenne une limite de débit, avec la possibilité de courtes pointes de trafic. Les paramètres d'un Token Bucket Filter (TBF) sont : le débit du baquet, la taille du baquet, une option de queue (délai avant de définitivement détruire les paquets hors limite), et le débit de pointe maximal.
— lorsque l'on désire donner des chances égales à divers types de traffic, le *Stochastic Fair Queueing* est un bon candidat
— pour classer le trafic en fonction de critères (p.ex. bits TOS, numéro de ports, etc), on peut utiliser des filtres
— les queues de priorités proprement dites sont par exemple de type FIFO, de type stochastique (round-robin par flux), etc.
— il est possible d'empiler les queues afin d'obtenir des limitations complexes liées à des types de trafic particuliers (*Classful Queuing Disciplines*, voir http://linux-ip.net/articles/Traffic-Control-HOWTO/classful-qdiscs.html)

La terminologie est décrite notamment dans le document http://lartc.org/howto/lartc.qdisc.terminology.html. Les types de queues sont bien décrites dans http://linux-ip.net/articles/Traffic-Control-HOWTO/classless-qdiscs.html#qs-tbf

Marquage du trafic classifié – 2.5

La limitation seule du trafic mène à des *pertes* de paquets pour tous les types de trafic, en cas de surcharge. Un véritable système de *traffic shaping* nécessite de la limitation (manipulation précédente), de la classification, par exemple par marquage d'un flux (cette manipulation) et de la priorisation (par queues de priorité classifiées, la manipulation du slide suivant).

43

Pour améliorer ce comportement, on va étudier ici la classification par marquage du trafic par type de flux.

Manipulations et observations : marquage du trafic La limitation du trafic montant a permis de diminuer le jitter. Cependant, le trafic audio, par exemple n'est pas prioritaire et donc peut être perdu au même titre que du trafic informatique. L'idée de la classification par marquage est de donner ensuite une plus grande priorité à un certain [1] type de flux sur les différents équipements traversés.

1. supprimez la *qdisc* introduites précédemment sur le `routeurentreprise` :
   `tc qdisc del dev eth0 root`
2. créez deux flux pproducteurs TCP montants, depuis le `pc` vers le `serveurinternet` (qui doit avoir les consommateurs déjà lancés, p.ex. dans `screen`, voir page 65) :
   ```
   buffer -S 32768 -s 32768 -i /dev/zero | nc -q 0 192.168.1.42 7001
   buffer -S 32768 -s 32768 -i /dev/zero | nc -q 0 192.168.1.42 7002
   ```
   vous devriez observer qu'après quelques secondes de stabilisation, les deux flux se partagent le débit montant à peu près à égalité. Ne pas hésiter à redémarrer les flux si vous voulez observer un changement, car la commande `buffer` calcule une vitesse totale et non instantanée. recommandation : exploitez `screen` en mode *split screen* (voir page 65) pour voir les deux débits simultanément.
3. notre but est de rendre un des deux flux reconnaissable facilement comme plus prioritaires que l'autre : nous allons simplement jouer avec les bits TOS [2] en fonction du numéro de port

1. ici, nous allons utiliser des flux TCP comme exemple
2. *Type of Service* : un champ de l'entête IP, aussi interprétable comme champ DIFFSERV

destination : rendez un des flux qualifié (marqué) en *type of service* sur le `pc` : *minimisation du délai* (voir ci-dessous) et vérifiez avec `tcpdump` que le bit est bien mis.

4. en pratique, un routeur surchargé pourrait laisser tomber ou retarder en priorité du trafic non ainsi marqué (ne pas faire).

Indications :

— soit les applications vont positionner les bits TOS, soit nous pouvons le faire à l'aide de règles de firewall, par exemple sur le `pc` :

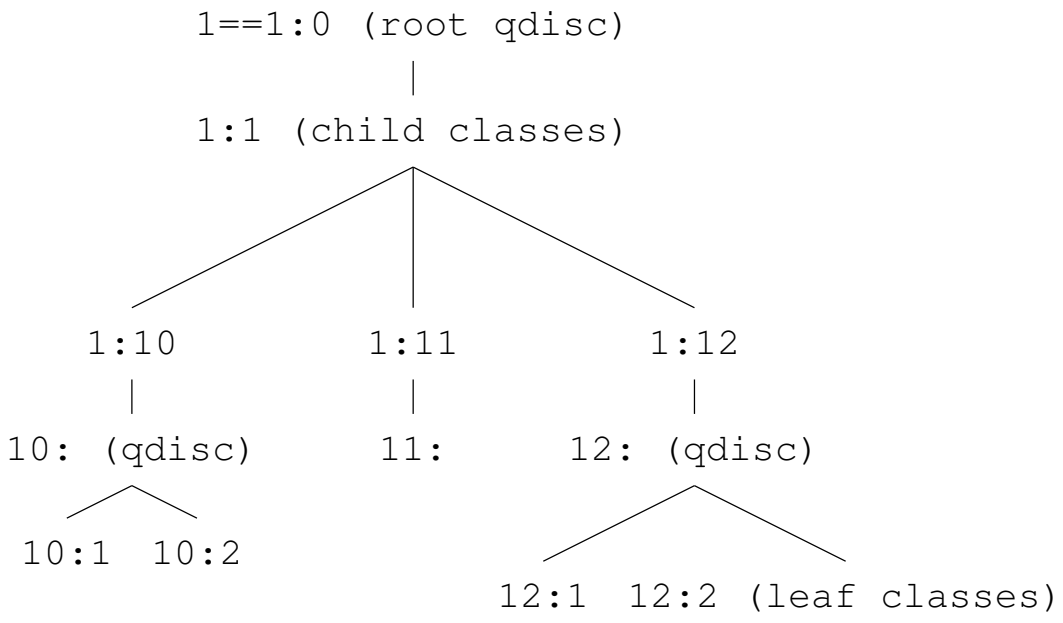
```
iptables -I OUTPUT -t mangle -p tcp --dport 7002 \
         -j TOS --set-tos Minimize-Delay
```

— on peut visualiser le bit TOS configuré sur `routeurentreprise` avec par exemple :

```
tcpdump -vvvv -i eth0 -n 'dst 192.168.1.42 && port 7002' \
   | head -1
```

Classification et traffic shaping hiérarchique – 2.6

Il reste à implémenter une véritable priorisation du trafic par queues de priorités hiérarchiques, par exemple associées à un *Hierarchical Token Bucket*.



44

Manipulations et observations : classes hiérarchiques et classification

1. déconfigurer la règle de firewall sur le `pc`, en remplaçant `-I` en `-D` la règle de firewall ajoutée précemment, ou en supprimant l'ensemble des règles du type avec `iptables -t mangle -F`
2. sur le `routeurentreprise`, configurer un HTB (*Hierarchical Token Bucket* pour l'interface montante, comprenant une limitation de trafic par classe en assignant un débit pour chacun des types de trafic, avec deux niveaux d'arbre :
   — $\frac{3}{4}$ du débit pour le port destination TCP 7001
   — $\frac{1}{4}$ décomposé en
   — $\frac{3}{4}$ du trafic restant pour tout le trafic IP restant
   — $\frac{1}{4}$ pour port destination TCP 7002
   (cela signifie que la concurrence est à 2 niveaux : si vous préférez, vous pouvez gérer plutôt une répartition plus plate avec concurrence entre les 3 classes comme dans l'exemple en page 71)
3. générez du trafic et visualisez l'effet sur les statistiques de chaque classe et sous-classe (`tc -s class show dev eth0`)
4. en option : configurez chacune des classes du bas de la hiérarchie pour être liées à des *qdisc* de type SFQ (*Stochastic Fair Queuing*) et vérifiez que l'effet est visible sur la répartition des débits (si vous rendez le rapport, donnez l'idée sans forcément implémenter)
5. consultez `http://lartc.org/wondershaper/` et donnez les buts et les éléments principaux du fonctionnement du script.

Indications

- le baquet hiérarchique (*Hierarchical Token Bucket*, HTB) permet de combiner TBF et p.ex. PRIO (cf http://luxik.cdi.cz/~devik/qos/htb/manual/userg.htm)
  - on crée un arbre HTB ainsi :
    `tc qdisc add dev eth0 root handle 1: htb default 11`
    (indication : racine `1:`, classe par défaut `1:11`)
  - la classe de la racine sera alors `1:1`
  - les feuilles de l'arbre (ou sous-arbres) seront les classes `1:10, 1:11, 1:12`, etc.
  - on déclare une classe via tc class en indiquant où l'on attache cette classe dans l'arbre et l'identification de la classe
  - on peut lister les classes définies avec `tc class show dev eth0`
- on peut utiliser des marques déjà portées par les datagrammes (p.ex. champ TOS), ou classifier par type de trafic (cf http://lartc.org/howto/lartc.qdisc.filters.html), par exemple
  `tc filter add dev eth0 parent 1:10 protocol ip prio 1 u32 match ip sport 80 0xffff flowid 10:1`
- dans l'arborescence, les qdisc sont toujours des `:0`
- les classes ne sont jamais traitées plus rapidement que leur parent

Exemple de traffic shaping multi-classe avec classification simple (plate)

```
DEV=eth0

# limitons le débit total en sortie
# (a adapter à la realite)

MAX_RATE=400000
HIGH_CLASS=300000
MIDDLE_CLASS=75000
LOW_CLASS=25000

# supprimer les qdisc existantes
tc qdisc del dev $DEV root

# ajouter une queue de type baquet hiérarchique de nom 1:
# tout trafic non classifié sera affectée à la classe 1:11
tc qdisc add dev $DEV root handle 1: htb default 11

tc class add dev $DEV parent 1: classid 1:1 htb rate $MAX_RATE \
   ceil $MAX_RATE

# définissons trois sous-arbres (ou feuilles) de type htb
# chacun peut utiliser tout le débit disponible, mais partagera avec
# ses voisins de même niveau en fonction du "rate" indiqué.
# donc nous n'avons pas les deux niveaux de la donnée ici
tc class add dev $DEV parent 1:1 classid 1:10 htb rate $HIGH_CLASS \
   ceil $MAX_RATE
tc class add dev $DEV parent 1:1 classid 1:11 htb rate $MIDDLE_CLASS \
```

```
   ceil $MAX_RATE
tc class add dev $DEV parent 1:1 classid 1:12 htb rate $LOW_CLASS \
   ceil $MAX_RATE

# à part les bits TOS via une pfifo qdisc empilée, on peut aussi
# discriminer sur d'autres champs, par exemple le numéro de port:
# (ici 7001 plus prioritaire que le reste du trafic qui lui plus
# que 7002)
tc filter add dev $DEV protocol ip parent 1:0 prio 1 \
   u32 match ip dport 7001 0xffff flowid 1:10

# la pire classe
tc filter add dev $DEV protocol ip parent 1:0 prio 1 \
   u32 match ip dport 7002 0xffff flowid 1:12
```

Trafic descendant – 2.7

Idéalement : limiter et prioriser le trafic sur le routeur du FAI de la même manière que nous l'avons fait pour le trafic montant sur le routeur d'entreprise.

En effet, il est plus adéquat de faire du *traffic shaping* avant que les données soient reçues de l'autre côté !

Mais ici on suppose que c'est impossible : quelles possibilités nous restent encore ?

Le cas du trafic descendant Idéalement, le trafic descendant devrait être limité et classé par ordre de priorité par le routeur du fournisseur. Ce n'est en règle générale pas le cas. Il n'est donc pas facile de limiter le trafic descendant : on peut cependant utiliser des astuces (voir les questions).

Références QoS – 2.8

- http://www.ietf.org/rfc/rfc3290.txt
- http://lartc.org/lartc.html (si indisponible, consulter les fichiers attachés dans le document Gitlab LaboQoS)
- http://linux-ip.net/articles/Traffic-Control-HOWTO/classless-qdiscs.html#qs-tbf

46

- http://www.netqos.com/resourceroom/calculator/index.html
- http://lartc.org/, http://lartc.org/wondershaper/
- http://www.voip-info.org/wiki/view/QoS+with+Linux+using+PRIO+and+HTB
- http://luxik.cdi.cz/~devik/qos/htb/manual/userg.htm
- test qualité UDP (notamment pertes et réordonnancements) : `iperf -u`
- voir aussi déroulé.

Questions – 2.9

Toutes ces questions ainsi que le laboratoire (grandes lignes) font partie de la matière du prochain questionnaire.

1. TCP contient un algorithme qui permet de réguler automatiquement son débit de sortie en fonction du débit effectif disponible de bout en bout, expliquez en quelques mots son principe. (si vous ne vous rappelez plus de ce que nous avons fait l'an passé, consultez https://web.archive.org/web/20210326084654/http://www.ssfnet.org/Exchange/tcp/tcpTutorialNotes.html, partie *Congestion Control*)

2. pourquoi un MTU plus faible peut améliorer la qualité de service ? (indication : penser à l'impact sur les critères de qualité de service (par exemple le *jitter* ou gigue de phase) du temps d'attente avant de transmettre sur un canal, car il est occupé par un message en transmission)

3. que doit faire un routeur de l'usager (CPE, *Customer Premise Equipment*) d'un opérateur (p.ex. Swisscom) s'il voit entrer des paquets avec des bits TOS ou DIFFSERV activés, avec un débit ne correspondant pas au SLA ? (p.ex : l'utilisateur n'a pas acheté le service, par exemple voix-sur-IP, correspondant) ?

4. imaginez que vous deviez limiter un téléchargement (p.ex. une connexion HTTP, FTP, etc) dans le sens descendant. Sachant que le protocole de couche 4 est TCP, que pourriez-vous faire dans *le sens montant*, sens que l'on ne touche en général pas sauf exceptions, pour diminuer la vitesse du sens descendant ?

5. corollaire : en supposant que le sens *montant* est saturé, comment feriez-vous pour assurer malgré tout un bon débit descendant ?

6. pourquoi ne gérer le *traffic shaping* que sur `routeurentreprise` (et pas en plus sur `routeurfai`) produit de bons résultats mais peut ne pas être suffisant ? (corollaire : en général, il est mieux de faire le *traffic shaping* sur le trafic sortant, pourquoi ?)

7. quel type de queue est recommandé pour garantir un traitement juste (*fair*) du trafic ? est-ce que le *Token Bucket* correspond à ce besoin ? (voir page 61)

Questions pour le rapport – 2.10

Ces questions ne font pas partie de la matière pour le questionnaire.

1. qu'est-ce que le routage de la patate chaude, et en quoi peut-il améliorer la qualité de service dans un réseau surchargé ?

2. avec quel outil pourriez-vous évaluer le nombre de paquets reçus en désordre ?

3. proposez une répartition de débit entre classes plus logique dans un réseau réel : pensez à un réseau IP général, aux applications de gestion (SNMP) au trafic interactif (SSH/TELNET), etc.

4. donnez les pistes de conception d'une application simulant l'algorithme du Baquet (cf page 65 et complément Gitlab) et implémentez-la.

5. implémentez la requête SQL qui détermine le 95-percentile, avec une table définie ainsi avec PostgreSQL :

```
CREATE TABLE mesures(
   id SERIAL NOT NULL PRIMARY KEY,
   stamp TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
   bitrate NUMERIC(10,0) NOT NULL);
```

   (indication : PostgreSQL dispose d'une fonction d'agrégation)

6. on a vu que les FAI configurent en général de grandes queues sur les routeurs. Le protocole QUIC proposé par Google pour remplacer TCP modélise le *tuyau* et diminue son débit s'il voit le délai augmenter. Pour rappel, TCP adapte son débit à la baisse lorsqu'il rencontre des pertes. Discutez les avantages de QUIC sur TCP sur ce point.

3. Labo sans-fil et multimédia

Objectifs – 3.1

47

— décrire les applications du sans-fil
— décrire les différences avec le filaire, en particulier pour le WiFi
— comparer les caractéristiques des technologies sans-fil (topologie, bandes de fréquence privées ou publiques)
— décrire les concepts de déploiement de certaines technologies
— déployer une intégration de données LoRaWAN/TTN via réseau coopératif https://thethingsnetwork.org

Déploiement de technologies En particulier, nous étudierons :

— WiFi classique en mode routage (voire routage/NAT) ou bridge, avec options multimédia modernes (WMM/802.11e), pour LAN/RLE
— faisceau hertzien, avec l'exemple d'un pont WiFi à moyenne distance pour relier deux bâtiments ou deux sites
— réseau de capteur LoRa, passerelle communautaire Internet LoRaWAN et intégration des données

Applications du sans-fil – 3.2

Dernier kilomètre

— opérateur : GSM (3-4-5 G), WiMax

LAN/RLE

48 — WiFi, LiFi
  — réseau sans-fil classique avec point d'accès (AP), bridgé ou routé sur le RLE
  — faisceaux hertziens courts en WiFi (pont WiFi) entre bâtiments voire sites
— dans certains cas : 4G – réseaux privés : exemple aéroport – et 5G via les *network slices*

Réseaux GSM d'entreprise Certaines applications particulières (p.ex. réseaux de capteurs d'avion) peuvent utiliser des satellites géostationnaires en vol et à terre, des réseaux privés d'aéroport, en particulier en 4G, bientôt en 5G (ce qui permet de consolider les autres usages, par exemple téléphonie et LAN dans un seul réseau plutôt que des réseaux épars [1]).

1. https://www.usinenouvelle.com/article/les-aeroports-parisiens-se-dotent-d-un-reseau-4g-N985819

Réseaux de terrain (industriels)

— faible à moyenne distance pour acquisitions de données et contrôle industriel, par exemple : domotique : mesh WiFi 6, Bluetooth Low Energy (BLE), Z-Wave, Zigbee, ...
— longue distance : LoRa, Sigfox
— avec *gateway* vers Internet (acquisition de données et contrôle)
  — LoRaWAN (réseau propre, coopératif ou commercial)
  — gateways spécifiques pour les autres protocoles
— réseaux mondiaux d'opérateurs LTE-m

PAN (Personal Area Network)

P.ex. le Bluetooth ou 6LoWPAN.

Exercice 1

49

Réseaux de terrains (industriels) On distingue, par contrainte temps-réel croissante : l'acquisition de données, le contrôle (allumer ou éteindre un équipement par exemple) et la commande (p.ex. d'axes, possible uniquement en filaire, p.ex. avec bus CAN).

En sans-fil, à courte distance, de nombreux protocoles existent (livret A5 P+R section 4.4.2, page 57) pour l'acquisition de données ou le contrôle. A longue distance, LoRaWAN est une solution, même si ce n'est pas le seul système disponible (tableau comparatif livret A5 figure 4.3 page 58).

La consommation d'énergie peut être très minime avec certaines technologies (Z-Wave, LoRa/LoRaWAN, BLE, Wifi 6) : un équipement de mesure sur pile peut prendre des transmettre toutes les 20 minutes pendant 10 ans, par exemple.

Le LTE-m est un réseau WAN à basse consommation, déployé de manière mondiale par les opérateurs.

Le livret A5 P+R section 4.4 contient plus de détails sur ces technologies.

Différences du WiFi avec le filaire – 3.3

— pas de garantie de détection de collision et collisions beaucoup plus probables
  — évitement des collisions CSMA/CA (se dégrade en CSMA/CD normal)
  — réservation de temps au moment venu
  — retransmission en couche 2 (désactivable)
— QoS plus difficile que la simple priorisation 802.1p
  — queues de priorité de transmission dynamiques
  — slots de transmission statiques, éventuellement dynamiques (TDMA)
— chiffrement et authentification
— déploiement en mode infrastructure (AP relais) ou ad-hoc (mesh possible)

50

Exercice 3

Voir les compléments dans le déroulé du Gitlab sous Technologies WiFi.

Authentification et chiffrement couche 2 car pas de câble
variantes
— clé unique partagée (secret partagé) WEP-WPA/WPA2 classique
— login et session sécurisée par utilisateur (versions WPA2/Entreprise) évt. avec centralisation de l'authentification avec RADIUS

Toute la sécurité dépend de comment les clés sont générées puis utilisées (cf Crypto&Sécurité ISC3). En général, le chiffrement est effectué par le matériel (symétrique), mais les clés symétriques sont perturbées à chaque transmission et changées régulièrement, par exemple via un cryptosystème asymétrique. En pratique, le problème est la compatibilité des implémentations des systèmes d'authentification, qui amènent parfois à choisir sub-optimalement question sécurité.

NB : des technologies similaires peuvent être exploitées en filaire également suivant les besoins (802.1x).

Communication directe entre station En mode infrastructure (avec AP), le WiFi ne permet par défaut pas la communication directe entre station, ce qui a un avantage de distance (rayon du réseau possible autour de l'AP) mais un inconvénient de performance. On peut soit, dans les versions récentes du WiFi passer en WiFi Direct, ou dans le cas sans-fil général, adopter une topologie plus *mesh* (mode ad-hoc du WiFi, mode normal du LoRa), en y ajoutant éventuellement des fonctions de routage mesh.

Topologies – 3.4

- PtMP : WiFi avec AP central (mode infrastructure), Z-Wave en mode centralisé
- PTP : WiFi faisceaux hertzien à moyenne distance (p.ex. 6 km)
- mesh : WiFi mode ad-hoc, LoRa, BLE, Zigbee, Z-Wave
- PtMP à gateway
  - LoRaWAN : conversion couche 7 de LoRa vers pile IP
51 - Z-Wave peut aussi être exploité dans ce mode centralisé
  - GSM : communication remonte au réseau core
  - WiMax : communication remonte au réseau core

Définitions

PtMP Point to Multi Point : les terminaux ne communiquent en général pas directement entre eux mais via l'AP

PTP Point to Point

mesh communication directe entre équipements, parfois protocoles de relais de trame / routage entre terminaux qui agissent comme nœuds) du réseau

PtMP à gateway les terminaux ne peuvent pas communiquer entre eux sans support couche réseau, voire application pour LoRaWAN

Bandes de fréquences – 3.5

— privées ou réservées : opérateur ou entreprise avec des besoins particuliers ; puissances importantes
  — GSM 3-4-5 G
  — WiMax
— publiques ou partagées/libres : déployable par opérateur, ou entreprise : puissance
52 limitée
  — Bluetooth et BLE
  — Zigbee, Z-Wave
  — WiFi
  — LoRa (& LoRaWAN) – avec limites de temps de transmission total

plus de détails livret A5 P+R chapitre 6
Exercice 7

Limitation de la puissance rayonnée

L'ORNI [1] pose les bases légales suisses des contraintes sur le maximum de puissance des émetteurs. Ces valeurs limites reposent sur deux principes [2] :

1. la cohabitation d'utilisateurs (en particulier dans les bandes publiques)
2. la santé publique (par le principe de précaution, voir livret A5 P+R section 6.2.6.3 page 75)

Par exemple, le WiFi est limité à une puissance rayonnée (voir livret A5 P+R section 6.2.6.2) de 100 mW (20 dBm) en 2.4 GHz et de 1 W (30 dBm) en général en 5 GHz [3]. Vu la formule 6.9 du livret A5 P+R (page 77), ces deux fréquences ont des distances potentielles comparables à ces puissances respectives – mais des comportements différents face aux obstacles éventuels.

Bilan de liaison En particulier, les faisceaux hertziens doivent pouvoir être dimensionnés correctement pour fonctionner et respecter les contraintes légales. Voir la section 6.2.4 du livret A5 P+R (page 73).

1. https://www.admin.ch/opc/fr/classified-compilation/19996141/index.html
2. voir aussi livret A5, section 6.2.6, page 74
3. mais consulter https://en.wikipedia.org/wiki/List_of_WLAN_channels pour toute la vérité !

Routage (3) ou commutation/bridging (2) – 3.6

Exemple du WiFi

- accès Internet en mode routage, voire NAT
- LAN (filaire) et WiFi de l'entreprise bridgés (switchés)
- LAN (filaire) et WiFi sous forme de deux sous-réseaux distincts, routés
- deux sites reliés ensemble par un pont WiFi (bridge, switch)

53

L'AP peut cumuler d'autres fonctions comme serveur DHCP, serveur web QoS…

Il peut être intéressant de configurer les interfaces de gestion des équipements réseau dans un VLAN séparé.

Modes d'exploitation

- commuter (switcher, bridger) le LAN filaire Ethernet avec le WiFi, de manière à ce que chaque terminal derrière l'AP, exploité en mode **bridge**, soit dans le même sous-réseau que les équipements filaires – les broadcasts et protocoles à découvertes vont fonctionner entre le sans-fil et le filaire !
- séparer le sous-réseau WiFi du sous-réseau réseau de l'entreprise en exploitant l'AP en mode **routage** : en général le plus simple
- cas particulier : étendre la portée d'une couche 2 / d'un sous-réseau, y compris entre deux bâtiments : commutation ou bridging via un **pont WiFi** formé de deux AP en mode bridge, éventuellement avec des antennes à gain (faisceau hertzien court)
- la liaison à Internet, si gérée par l'AP, se fait quasiment toujours en routage – avec du NAT la plupart du temps

Sécurisation des nœuds Il peut être nécessaire, en particulier en mode bridge, de séparer le réseau de gestion des AP du réseau transporté, avec des VLAN 802.11q.

Problème du relais de trame via l'AP En Ethernet, il faut deux adresses MAC : une pour l'émetteur, une pour le récepteur. En WiFi, avec un AP, l'émetteur envoie sa trame via l'AP, l'AP la réémet en direction de la destination : il faut donc 3 adresses MAC dans l'entête couche 2 WiFi : AP, destination, source.

Le problème se complique dans le cadre d'un pont WiFi exploité en bridge Ethernet des deux côtés (deux AP en mode *Wireless Distribution System* – WDS) : il faut utiliser les quatre [1] adresses MAC de l'entête WiFi !

En alternative, de la translation d'adresses couche 2 peut-être exploitée (le LAT de couche 2 (*Link-layer Address Translation*), équivalent du NAT de couche 3 et du PAT de couche 4), cachant tous les terminaux derrière l'adresse MAC de l'AP de chaque côté, avec une compatibilité acceptable en général.

Le problème ne se pose bien sûr pas en mode *routage*.

1. pas supporté par tous les AP correctement

Formats de données d'échange – 3.7

— JSON
— XML
— Protobuf
— ASN.1/DER
— texte quelconque
54 — binaire quelconque

leur standardisation, structuration et potentiel de validation diffère ; ils peuvent être combinés

Les formats (syntaxes de transfert) Beaucoup de formats utilisés aujourd'hui sont structurés et validables. Parmi les formats textes, citons JSON (populaire sur le web) et XML (populaire en entreprise). Parmi les formats binaires-structurés : Google Protobuf, ASN.1/DER-BER.

Les autres formats texte et binaires ne sont pas forcément structurés ni validables.

Google Protobuf [1] est populaire en embarqué ou quand la performance ou la taille des données est un critère.

Les formats standardisés *télécom* DER/BER (syntaxe de transfert) d'ASN.1 (syntaxe abstraite) sont utilisés dans certains protocoles réseau binaires-structurés (p.ex. protocole et certificats HTTPS X.509, SNMP, etc).

Sur chaque machine, une syntaxe locale dépendant de la plateforme matérielle et du langage et permettant un traitement efficace, sera utilisée (classes, structures de données du langage, etc).

Voir le livret A5 Réseaux, chapitre 6.2 et figure 6.1 pour les notions de syntaxe locale, de transfert et abstraite ; et le chapitre 7 pour les divers formats classiques et modernes.

Formats combinés Le format utilisé peut être différent dans le transport et dans les données proprement dites : par exemple une intégration HTTP/Webhook TTNv3 envoie par défaut du JSON (structuré) dont un des champs est le *payload* binaire propriétaire du périphérique concerné.

Très souvent, l'XML et le JSON sont transportés compressés par une option du protocole HTTP.

1. Protocol buffers ; https://developers.google.com/protocol-buffers

Intégration de données LoRaWAN/TTN – 3.8

à l'aide du réseau coopératif TTN et en particulier de sa console

- logguez-vous avec votre compte (par groupe) sur TTN et créez une application TTN
- y intégrer (associer) un capteur LoRaWAN
- visualiser la réception de données LoRaWAN dans TTN
- décoder côté serveur TTN en Javascript
- coder une intégration TTN simple (HTTP Webhook sur conteneur externe)
- mettre en place une intégration TTN plus complexe

55

Principes de LoRaWAN Le *Long Range Wide Area Network* est un protocole de réseau sans-fil à base consommation. C'est une technologie propriétaire de *Semtech*, dont de nombreux composants sont open source, permettant à brancher des terminaux IoT à des services sur Internet, via des *gateway* de couche 7.

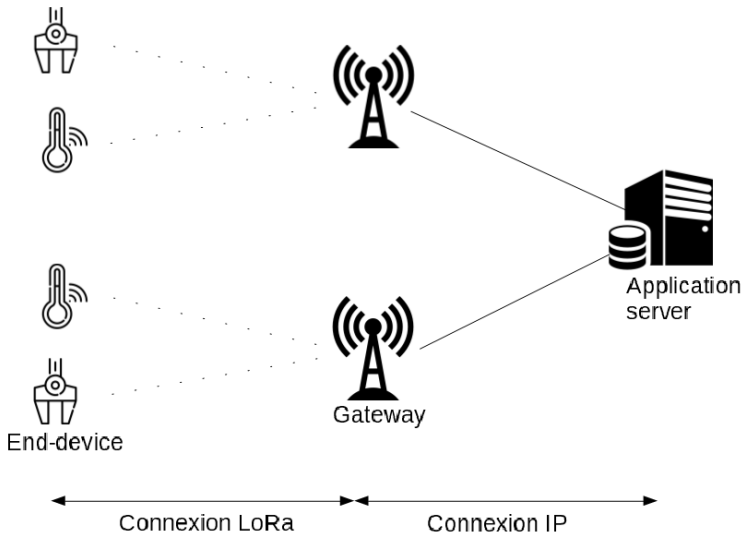


FIGURE 1 – Architecture LoRaWAN – https://fr.wikipedia.org/wiki/LoRaWAN

Les exploitations de gateways LoRaWAN peuvent être commerciales [1] ou coopératives, comme le réseau mondial *The Things Network* – TTN, https://thethingsnetwork.org : on trouve donc dans la plupart des villes des gateways disponibles. On peut aussi monter son propre gateway TTN ou privé : par exemple ALPHANET exploite un gateway TTN à Chaumont.

TTN offre en plus toute une pile applicative permettant d'acquérir ou d'envoyer des données, d'implémenter des translateurs entre les formats binaires des *payload* spécifiques des paquets de chaque type de terminal et par exemple JSON, ainsi que de connecter des bases de données ou de vraies applications.

La topologie est en étoile, les terminaux ne communiquant qu'avec un gateway LoRa à proximité (à vue, de l'ordre de 10km, jusqu'à 40km dans de rares cas), qui lui-même relaye les paquets vers un service applicatif sur Internet (*uplink*) ou vers les terminaux (*downlink*).

Le réseau d'accès entre un terminal et le(s) gateway(s) est LoRa. Le réseau d'amenée jusqu'au serveur d'application passe par Internet. Les débits sont faibles (moins de 300 kbit/s et au minimum 50kbit/s). La modulation utilisée est à faible radiation [2] et consommation d'énergie, ce qui permet des terminaux sur piles avec une durée de vie de 10 à 15 ans.

Les fréquences utilisées sont publiques [3], donc limitées en puissance et aussi en temps maximum d'émission par jour : LoRaWAN et en particulier TTN ne permettent que des envois de données de faible quantité vers Internet et la restriction est encore plus grande pour les données vers les terminaux (*downlink*).

Les terminaux peuvent implémenter trois classes de service (A : envoi vers la gateway, puis reste allumé pour deux tentatives de réceptions de données ; B : fenêtres de réception à temps fixes ; C : activité possible en continu, au prix d'une consommation d'énergie importante).

Il y a plusieurs modes d'exploitation pour les clés de chiffrement de session : une configuration sur l'équipement (ABP), ou une configuration automatique par échange avec le réseau (OTAA), basée sur la clé d'application, ce que nous allons utiliser ici.

Plusieurs informations sont nécessaires pour intégrer un terminal (objet connecté) dans un réseau LoRaWAN :

DeviceAddress L'adresse MAC du terminal (32 bits, auto-détectée)

DevEUI L'identificateur du terminal (64 bits)

AppEUI L'identificateur de l'application

AppKey La clé de l'application

En pratique et dans notre laboratoire, on va utiliser les identificateurs et clés générés par le fabricant et stockés dans les terminaux pour relier à TTN, et des clés de session (Network et App) générées automatiquement. En effet, une application TTN peut être liée à plusieurs AppEUI et AppKey (par terminal, appelés *devices* dans TTN). A cela s'y ajoutent des clés d'API pour interagir entre l'application TTN et des services Internet (*Integrations*).

1. en Suisse, Swisscom exploite un réseau : https://www.swisscom.ch/fr/business/enterprise/offre/iot/lpn.html
2. *Chirp spread spectrum* – modulation à étalement de spectre : le message est répété à des fréquences différentes ce qui est un avantage pour la réception et pour éliminer p.ex. les effets Doppler et pour le prix des équipements ; de plus il y a une certaine orthogonalité qui permet potentiellement l'envoi de trames par plusieurs équipements en même temps
3. bande d'instruments ISM, fréquences différentes en Europe et aux Etats-Unis, en Europe c'est en général entre 863 et 870 MHz

Références : https://fr.wikipedia.org/wiki/LoRaWAN et documentations TTN.

Manipulations et observations Utilisez les explications détaillées dans les paragraphes qui suivent pour effectuer les différentes étapes suivantes :

1. vous logguer avec votre compte par groupe sur TTNv3
2. créer une application TTN
3. intégrer (associer) un capteur LoRaWAN à cette application (avec les IDs et clés sur l'auto-collant dans la boîte)
4. visualiser la réception de données LoRaWAN dans TTN et déterminer par quel gateway elles passent (voir aussi https://ttnmapper.org/)
5. tester la simulation de données
6. tester, puis compléter un décodeur de données Javascript dans TTN
7. tester l'accès à votre conteneur ALPHANET-DS ou à votre cloud préféré
8. configurer et coder une intégration HTTP (Webhook) vers votre application dans votre conteneur ALPHANET-DS, par exemple en logguant les données reçues via un script PHP ou en les affichant à l'écran
9. faire quelque chose de cette intégration en parsant des valeurs particulières (p.ex. graphe de données munin ou autre) (en option : travaillez sous un utilisateur non `root`)
10. configurer et réaliser un autre type d'intégration, à choix
11. si vous rendez le rapport : utilisez le format *protobuf* à la place de JSON pour votre intégration HTTP/Webhook et comparez JSON et protobuf.

Création d'une application TTNv3 Logguez-vous avec le compte TTN Étudiant (un par groupe) pré-créé sur le cluster TTN Europe 1 sous https://www.thethingsnetwork.org/, avec comme Username `ra-labo3-x`, où `x` est le numéro du LHT65 qui vous a été remis et le mot de passe qui vous a été indiqué :

FIGURE 2 – Etiquette du numéro 5 – exemple
– adapter !

Créez une application via Console[4] > Europe 1 > Create an application[5], avec l'Application ID `ra-labo3-a-x`[6], où `x` est toujours le numéro du LHT65 déjà utilisé pour le compte.

4. https://console.cloud.thethings.network/
5. lien direct https://eu1.cloud.thethings.network/console/applications/add
6. doit être globalement unique et ne peut être réutilisé : si existe déjà, prenez cette existante

Equipement LHT65 L'équipement est un capteur de terrain, pouvant mesurer diverses valeurs, comme par exemple la température et l'humidité, sur pile d'une durée maximum de 10 ans, et qui est configuré par défaut pour envoyer toutes les 20 minutes ses données au réseau TTN (gateway le plus proche).

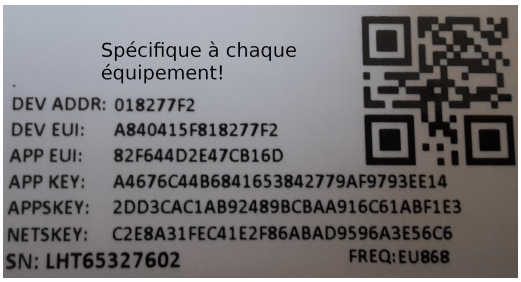


FIGURE 3 – Autocollant de préconfiguration du capteur LHT65 (clés, IDs) – adapter !

Comme beaucoup d'équipements LoRaWAN, il est assez spartiate mais aussi assez bon marché (une trentaine de CHF).

Informations commerciales et générales : http://www.dragino.com/products/lora-lorawan-e
item/151-lht65.html ; informations techniques : http://www.dragino.com/downloads/
index.php?dir=LHT65/

Association de votre LHT65 Accédez à votre application TTNv3 puis activer le lien `+ Add end device`, puis l'onglet `Manually` :

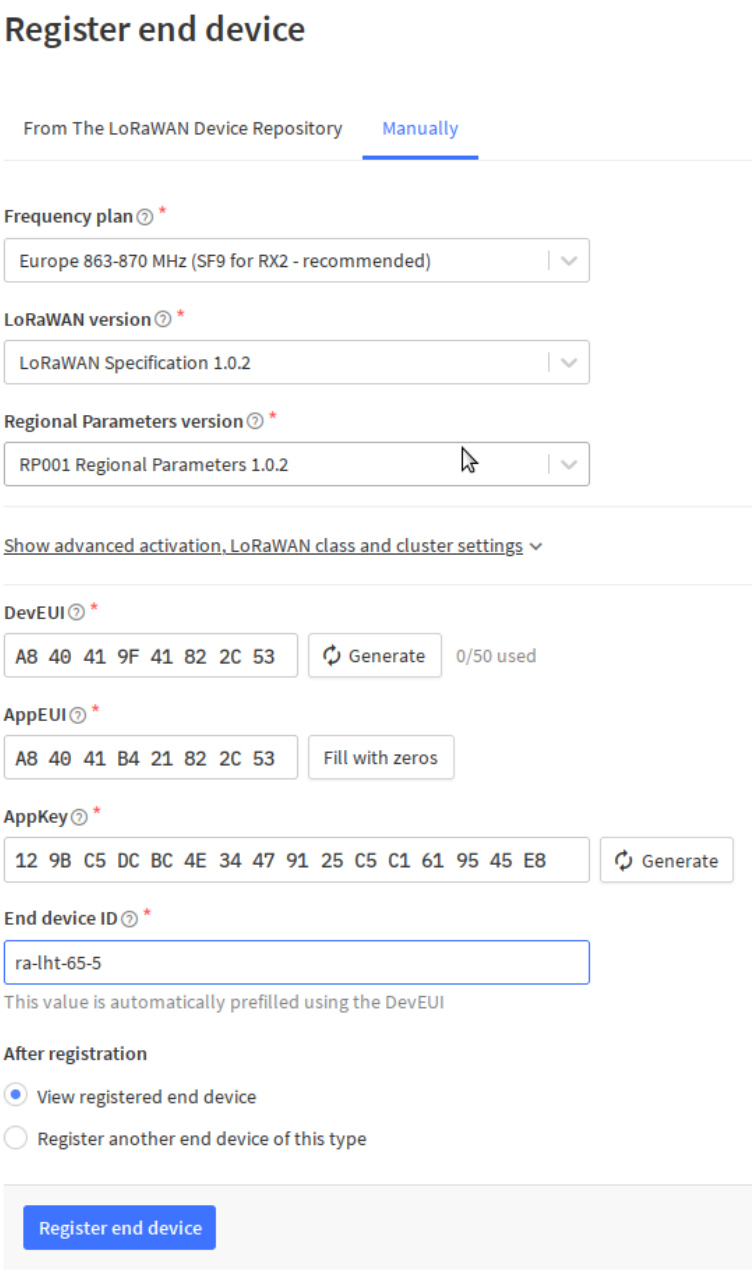


FIGURE 4 – Enregistrement manuel LHT65

Entrez les `Dev EUI`, `App EUI` et App Key en fonction de l'étiquette dans la boîte de votre LHT65. Comme `End device ID` utilisez : `ra-lht-65-x` avec `x` le numéro du LHT65 comme auparavant.

Attention : un Device ne peut être associé qu'à une application, et son Dev EUI et son nom doit être unique.

Exemple de paquet reçu Visualiser le trafic de ce périphérique dans TTN : sous *Live data*, puis *enclenchez* le LHT65 (si pas déjà fait, voir ci-dessous). Si nécessaire, pressez pendant 1 à 2 secondes

le bouton `ACT` pour générer du trafic :

Exemple d'un *payload* :

Vous pouvez aussi consulter les compteurs de trafic juste au-dessous de l'icône du device.

Déterminez par quel(s) gateway(s) les messages sont passés en visualisant le message complet en JSON (cliquer sur le message). Ca pourrait être `hearc-campus-arc2` ou `hearc-ing-tinf-ca2` (dans la salle).

Enclenchement/déclenchement et status du LHT65 Attendez d'avoir déclaré le LHT65 dans TTN pour l'enclencher !

Pour allumer (enclencher) (passage du mode d'endormissement profond économique au mode normal), pressez plus de 3 secondes sur son bouton `ACT` et la LED verte devrait flasher 5 fois. Elle devrait ensuite, si elle a joint le réseau TTN LoRaWAN, rester allumée pendant 5 secondes.

Pour éteindre (déclencher et économiser la batterie), pressez le bouton `ACT` 5 fois en succession rapide, la LED rouge s'allumera pendant 5 secondes.

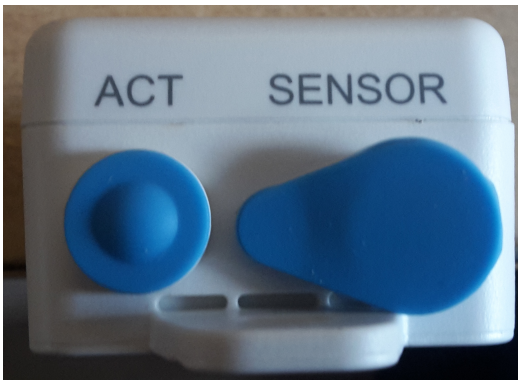


FIGURE 5 – Bouton ACT et slot de branchement du thermomètre du capteur LHT65

Vous pouvez aussi presser sur `ACT` entre 1 et 3 secondes et le LHT65 enverra un packet *uplink*, et la LED clignotera en bleu (senseur extérieur connecté) ou en rouge (senseur extérieur non connecté). Ou refaire la procédure avec 5 secondes.

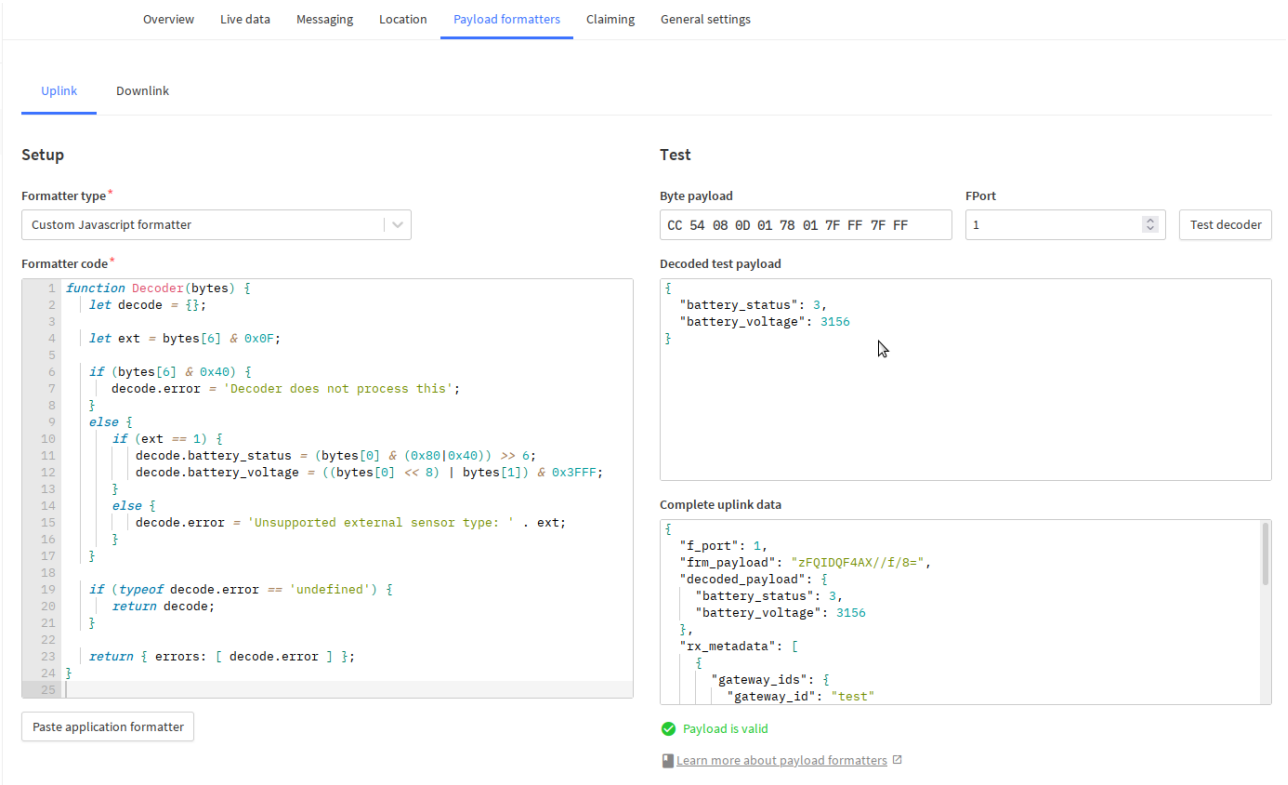
Simulation de payload

La simulation de payload permet d'injecter dans TTN des données sans que le périphérique ni LoRaWAN doivent fonctionner.

Simulez un *payload* sous Messaging > Uplink p.ex. le payload `CC 54 08 0D 01 78 01 7F FF 7F FF` : il devrait être visible dans les datas et dans la future intégration.

Ecrire un décodeur Uplink TTN en Javascript Vous trouvez la spécification du format du payload ici : https://www.dragino.com/downloads/downloads/LHT65/UserManual/LHT65_Temperature_Humidity_Sensor_UserManual_v1.8.5.pdf, dès la page 15.

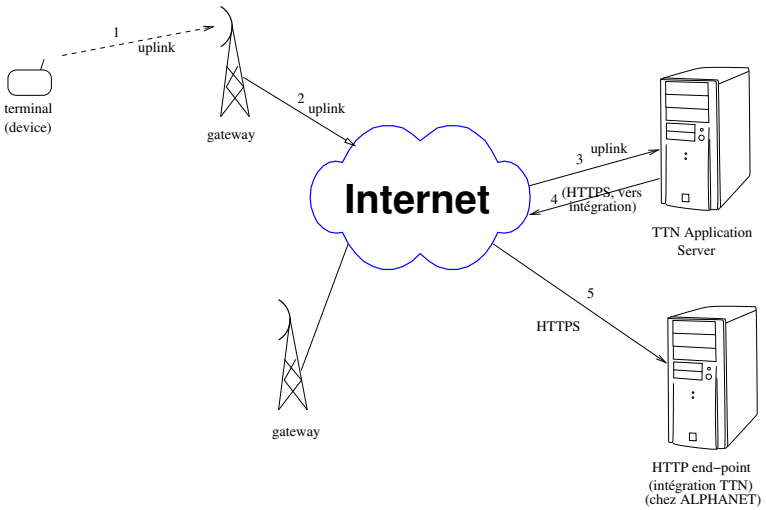
Vous trouvez un exemple de décodeur en Javascript, qui décode le status et le voltage de la batterie en mV sous `code/payload-decode.js`. Testez ce code sous votre périphérique comme ci-dessous :



Ensuite, ajoutez le code pour pour décoder au minimum les deux températures et testez.

NB : vous pouvez utiliser les outils de développeur de votre navigateur et leur console pour tester la syntaxe Javascript, et/ou l'IDE fourni par TTN. Si vous faites Save, alors vous pouvez même voir votre décodage dans l'application (sous Live Data). Augmentez un peu la température en réchauffant le capteur avec votre main puis en cliquant moins de 3 secondes !

Exemple complet d'une application TTN
Il s'agit de sept capteurs LHT65 LoRaWAN à Fontainemelon, Chaumont et Grimentz, envoyant leurs données à divers gateways TTN et déclarés comme *devices* TTN configurés dans une application TTN, qui alimentent une intégration HTTPS (Webhook) qui envoie les données immédiatement à un script Perl situé à l'URL `https://www.alphanet.ch/~schaefer/cgi-bin/ttn-demo.pl` via la méthode POST.



Aucun downlink n'a été configuré dans ce cas (pas d'ordre à envoyer au terminal).

Consulter `https://git.alphanet.ch/gitweb/?p=ttn-demo;a=tree` pour le code source entier :

`ttn-demo.pl` intégration HTTP (Webhook) traitant les données provenant de TTN

`lib/LoraWAN/handlers/lht65.pm` module de décodage LHT65

Exemple de log produit par `ttn-demo.pl` qui est ensuite parsé pour obtenir un graphe munin :

```
Wed Jan 20 14:10:26 CET 2021: battery 3 battery_raw 2.915 \
humidity 92.7 temp_2 5.25 temperature 2.61
```

NB : il est évident que l'intégration HTTP faite ici est sécurisée, travaille avec Apache2 comme serveur web et pas sous root. Par contre, ce qui est encore perfectionnable est qu'il manque l'authentification.

Tester l'accès à votre conteneur ALPHANET-DS
Nous utilisons ce cloud car il a des adresses IP externes, ce qui est *nécessaire* lorsqu'on veut utiliser l'intégration HTTP de TTN (PUSH), qui est une des plus simples à tester et utiliser, mais n'est pas forcément très performante. Tous les conteneurs ont la même adresse IP, mais des ports différents (pour les services SSH, HTTP, voire HTTPS si utile).

Informations nécessaires
- nom ou adresse IP : tinf.cust.ds.alphanet.ch / 193.72.186.153
- numéros de port externes SSH et HTTP : voir votre billet
- utilisateur et mot de passe : voir votre billet

Exemple d'accès à un conteneur et de test de fonctionnement simple : ci-après, l'utilisateur est user-0, les numéros de ports externes sont 2222 (SSH) et 8080 (HTTP) et correspondent aux numéros internes respectifs 22 et 80. Adaptez en fonction du billet.

```
votre-laptop% ssh -p 2222 user-0@193.72.186.153
conteneur% sudo apt-get update && sudo apt-get dist-upgrade
conteneur% sudo apt-get install netcat-traditional
conteneur% sudo nc -l -p 80
```

Vous pouvez maintenant tester que si vous allez sur http://193.72.186.153:8080/script.php (adapter le port) avec un client WWW, `nc` affichera la requête GET correspondante et les entêtes client, par exemple :

```
GET /script.php HTTP/1.1
Host: 193.72.186.153:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:98.0) Gecko/20100101 Firefox/98.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0
Accept-Language: en-US,en;q=0.7,fr-CH;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Vous pouvez alors terminer `nc` avec CTRL-c.

Installation de `php` en *standalone* (sans serveur web Apache)

```
sudo apt-get install php-cli && sudo apt-get clean
mkdir html
cd html
```

Ensuite, lancer le serveur web minimal PHP pour servir le répertoire courant – pas super de tourner sous root[7] : `sudo php -S 0.0.0.0:80`

Déployer votre application Vous pouvez maintenant copier votre script PHP avec
`votre-laptop% scp -P 2222 script.php user-0@193.72.186.153:html/`

Exemple simple de script PHP :

```
<?php

$stderr = fopen('php://stderr', 'w');

fwrite($stderr, file_get_contents('php://input') . "\n");
```

7. nécessaire car le port 80 est un port privilégié; alternative : redirection de port sur un port > 1023 (non privilégié), par exemple https://catonmat.net/unprivileged-programs-privileged-ports

Pour tester ce script comme la future intégration TTN le ferait, vous pouvez par exemple utiliser l'outil `POST` [8] :

```
votre-laptop% echo '{ "a": "b"}' \
                  | POST -E http://193.72.186.153:8080/script.php
```

Produit :

```
POST http://193.72.186.153:8080/script.php
User-Agent: lwp-request/6.36 libwww-perl/6.36
Content-Length: 12
Content-Type: application/x-www-form-urlencoded

200 OK
Connection: close
Date: Fri, 22 Jan 2021 07:59:25 GMT
Host: 193.72.186.153:8080
Content-Type: text/html; charset=UTF-8
Client-Date: Fri, 22 Jan 2021 07:59:25 GMT
Client-Peer: 193.72.186.153:8080
Client-Response-Num: 1
X-Powered-By: PHP/7.3.19-1~deb10u1
```

Et le script affiche, sur la console du serveur PHP standalone :

```
{ "a": "b"}

[Fri Jan 22 07:59:25 2021] 193.72.186.129:53350 [200]: /script.php
```

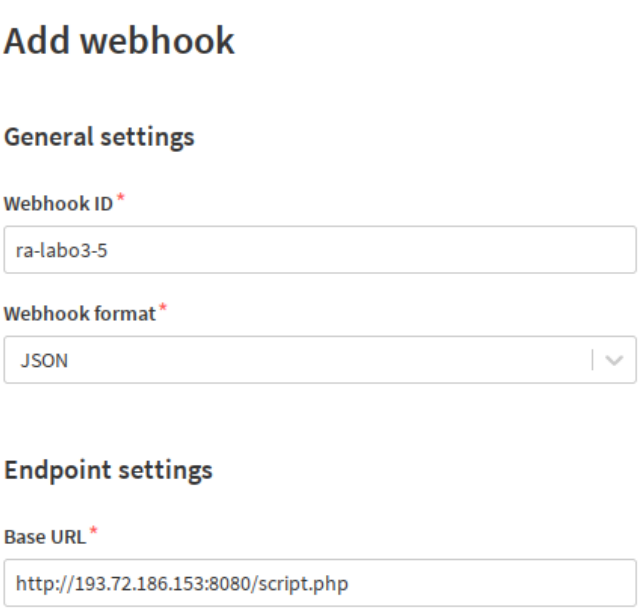
Tester votre intégration Dans ce conteneur, l'URL de l'intégration HTTP (Webhook) dans TTN sera alors `http://193.72.186.153:8080/script.php` [9] et correspondra au port 80 du serveur web (ici PHP standalone) dans le conteneur.

Vous pouvez commencer par tester pour voir si votre script loggue bien quelque chose, puis activer l'intégration [10].

Créer cette intégration HTTP JSON ainsi : `Applications` > `Integrations` > `Webhooks` > `Custom` (voir page suivante).

N'oubliez pas d'activer les messages de type `Uplink message` !

8. package `libwww-perl`; alternative : `curl` voir https://gist.github.com/subfuzion/08c5d85437d5d4f00e58
9. aussi DNS http://tinf.cust.ds.alphanet.ch:8080/script.php
10. consulter : https://www.thethingsindustries.com/docs/integrations/webhooks/



Notre script de test affiche alors, par exemple, dès qu'une donnée est envoyée (bouton) ou simulée par l'interface web, et envoyée par POST HTTP : (remis en page)

```
{"end_device_ids":{"device_id":"ra-lht-65-5","application_ids":{
"application_id":"ra-labo3-a-5"},"dev_eui":"A840419F41822C53",
"join_eui":"A84041B421822C53"},"correlation_ids":
["as:up:01FZWVFHACPTGTTQYRNFEHX62J",
"rpc:/ttn.lorawan.v3.AppAs/SimulateUplink:74c5ec60-2564-4e2c-8e08-2435fae9706f"],
"received_at":"2022-04-05T12:34:04.748850022Z", "uplink_message":{"f_port":1,

   "frm_payload": "zFQIDQF4AX//f/8=",
   "decoded_payload": {"battery_status": 3, "battery_voltage": 3156},

"rx_metadata":[{"gateway_ids":{"gateway_id":"test"},"rssi":42,
"channel_rssi":42, "snr":4.2}],"settings":{"data_rate":
{"lora":{"bandwidth":125000,"spreading_factor":7}}}},"simulated":true}

[Tue Apr  5 12:34:04 2022] 34.255.49.188:40495 [200]: /script.php
```

On voit ci-dessus le *payload* brut (sous `frm_payload`), encodé base64) et du décodage Javascript côté TTN grâce à notre script de démonstration (sous `decoded_payload`)[11]. Il s'agit ici d'un message simulé. Le JSON d'un vrai message est beaucoup plus grand (indication par exemple des gateway, etc).

Le payload proprement dit dépend de l'équipement, vous pouvez utiliser les liens sur le LHT-65 en début de ce labo ou l'exemple de code en Perl pour extraire p.ex. la température. En alternative, pourquoi ne pas utiliser votre pré-décodage en Javascript côté TTN et donc les valeurs symboliques ?

Si vous rendez le labo, faites quelque chose avec ces données comme demandé dans le slide, que cela soit dans votre script PHP ou dans une autre des intégrations possibles avec TTN. Cela pourrait être de faire un graphe, par exemple.

11. En PHP, utiliser : https://www.php.net/manual/fr/function.json-decode.php pour le décodage du message JSON et https://www.php.net/manual/fr/function.base64-decode.php pour le décodage du payload en base64.

Questions – 3.9

Toutes ces questions ainsi que le laboratoire (grandes lignes) font partie de la matière du prochain questionnaire.

1. remplissez le tableau ci-dessous concernant l'adéquation de normes sans-fil (WiFi 802.11 mais pas seulement) – la première ligne est déjà remplie :

type d'utilisation	**normes ou technologies envisageables**
réseau sans-fil pour ordinateurs et mobiles domestique	802.11, évt. Bluetooth ou LIFI (lumière)
réseau de terrain (p.ex. pour domotique)	
liaison de quelques kilomètres entre deux bâtiments qui se voient	
dernier kilomètre (boucle locale, WLL) pour opérateur télécom	
réseau IoT à grand périmètre	

2. quelle est la différence principale entre un access-point exploité en mode *bridge* et en mode *routeur* ?

3. Max constate que son laptop dispose des débits mesurés suivants :
   — environ 4 à 5 mégabytes par seconde pour l'accès Internet
   — environ 2 mégabytes/s pour le transfert vers un deuxième laptop
   Il utilise un réseau WiFi 802.11g en mode infrastructure. Aidez-vous d'un petit schéma (laptop 1, access-point, laptop 2, routeur Internet) si nécessaire !
   Est-ce que les valeurs mesurées sont correctes et pourquoi ?

   Avancés (bonus) : que pouvez-vous suggérer pour améliorer le débit inter-laptops ? (deux idées)

4. un AP WiFi exploité en mode *routeur* n'est pas qu'un routeur idéal, listez les fonctions supplémentaires que l'access point exploite dans ce mode : pensez aux fonctions de gestion du réseau local (WiFi+filaire, souvent bridgé quand même sous la forme d'un sous-réseau unique), mais aussi aux fonctions nécessaires pour permettre aux machines du au réseau local d'accéder à Internet (par exemple si elles sont en adresses privées)

5. à quoi sert l'activation de la fonction *client isolation* d'un AP ? en quoi c'est similaire aux *VLAN privés* ?

6. comment configurer les adresses IP de gestion des AP dans un pont WiFi exploité en mode bridge (en supposant qu'on les veut disjointes du sous-réseau bridgé) ?

7. en utilisant la feuille de calcul HTML/Javascript du Gitlab ou le lien Internet `http://wireless.saitis.net/wlan_calc_fr.html`, calculez la liaison suivante :
   — site 1 : Cernier ; site 2 : Fontainemelon distance : 600 m, bonne visibilité directe sans obstacles même autour
   — côté Cernier : access point ZyXEL, câble 2m (type Aircom), vers une antenne à gain
   — côté Fontainemelon : Orinoco PCMCIA Silver/Gold, câble 10 cm
   — puissance de sortie de l'access point : 30 mW (ZyXEL)
   — puissance max autorisée dans un secteur à l'antenne émettrice : 100 mW (limite légale à 2.4 Ghz !)
   — sensibilité du récepteur : similaire à Orinoco PCMCIA Silver/Gold
   — vitesse au moins 1 MBps

   Déterminez le **gain** de l'antenne émettrice pour que la liaison soit **faisable** et vérifiez qu'elle reste **légale**. Donnez le **débit** envisageable. Faites une copie d'écran du bilan. Voir aussi la section 6.2.4 du livret A5 P+R (page 73).

8. quel est l'avantage, pour un WiFi servant à des accès de mobiles en entreprise, d'utiliser une authentification EAP basée sur un *backend* RADIUS plutôt qu'un secret partagé ?

9. listez les fonctions multimédia 802.11e (WMM) proposées par l'access point `WAPS-APG600H`

10. dans notre intégration TTN HTTP Webhook, les données circulent-elles en clair ? TTN s'authentifie-t-il à notre application ?

Questions pour le rapport – 3.10

Ces questions ne font pas partie de la matière pour le questionnaire.

1. Amélie veut relier deux LAN Ethernet via un bridge WiFi (en couche 2, donc). Cela signifie que le même sous-réseau (même domaine de diffusion Ethernet) sera actif sur les deux LANs et sur le sans-fil. Cela veut dire aussi que les protocoles classiques comme ARP vont être utilisés. Il y un access-point bridgé sur chaque LAN, exploité en mode bridge et non pas routeur.

   Amélie capture le trafic du côté du LAN2 et elle constate que toutes les machines du LAN1 y possèdent la même adresse MAC ! Tout semble toutefois fonctionner normalement.

   (a) est-ce normal ? que se passe-t-il probablement ? quel est le risque éventuel ?

   (b) consultez le format de trame 802.11 et expliquez comment les 4 adresses devraient être utilisées, si cette fonction bizarre n'était pas activée

2. le chiffrement AES nécessite une clé de 256 bits : donnez un exemple de secret partagé qui permette d'atteindre cette longueur de clé (et donc qui contienne une entropie suffisante) : estimez si nécessaire et justifiez votre démarche (indication : évaluer l'entropie d'un mot de passe trivial comme `toto`, puis proposez une démarche puis un exemple meilleur, voire parfait ; voir https://en.wikipedia.org/wiki/Password_strength#Entropy_as_a_measure_of_password_strength) ; avancés : comment peut-on, si un mot de passe n'a pas une entropie suffisante, *stretcher*[12] la clé pour améliorer la situation

3. qu'est-ce qu'un *portail captif* dans le contexte de la sécurisation d'un réseau sans-fil sans chiffrement ? y-a-t-il des problèmes avec cette stratégie ?

4. faites un schéma couche 3 et couche 2 du pont WiFi dans la variante où les équipements réseau sont administrés dans un VLAN séparé, placez un PC d'administration et indiquez quels ports sont en *trunk* et quel port est en *access*

12. voir https://en.wikipedia.org/wiki/Key_stretching et https://www.schneier.com/academic/paperfiles/paper-low-entropy.pdf

5. avec deux antennes à gain 30 dB, quelle distance pourriez-vous atteindre ? quel serait le problème de cette liaison ?

6. donnez des arguments pour l'exploitation en mode *bridge* et en mode *routeur* au sein de l'entreprise (les réseaux filaires et sans-fils sont respectivement switchés et routés)

7. Alex se rend compte que parfois les communications voix-sur-IP sont mauvaises au sein de son réseau WiFi : la variance de délai est telle qu'il doit mettre de grand buffers dans les applications, ce qui rend la communication peu interactive. Il a pourtant mis en place du 802.11n avec du 802.11e (ses AP sont "compatibles WMM"). En debuggant un peu plus, il se rend compte que parfois des trames se perdent.

   Que se passe-t-il et que faut-il faire ?

   Il peut y avoir plusieurs idées plus ou moins simples.

8. Soit un réseau de capteurs que vous avez codés, en 802.11. Vous remarquez, trop tard, que le chip WiFi embarqué ne supporte pas le chiffrement en couche 2 (ou alors du vieux WEP à 64 bits cassable en quelques secondes). Que pouvez-vous faire pour améliorer la sécurité ? Pourquoi ce n'est pas idéal dans ce cas ?

9. Listez les mesures à prendre pour sécuriser un réseau WiFi et classez-les de la meilleure à la moins utile.

Fin du laboratoire – 3.11

Dès que vous avez fini (copies d'écran prises) et que vous ne pensez pas vouloir revenir sur ce labo plus tard :

Veuillez déconfigurer les décodages Javascript, les intégrations de votre compte TTN. Laissez le device et l'application (ne pas les supprimer !)

56

Rendez ensuite la boîte complète avec tous les accessoires, LHT65 éteint (voir page 92), y compris le billet avec les clés imprimées et les accès conteneurs.

4. Labo voix-sur-IP

Objectifs – 4.1

57

— découvrir les principes des protocoles de voix-sur-IP
  — signalisation (SIP)
  — négociation de contenu (SDP)
  — codecs et transport de l'audio (RTP)
— configuration d'équipements et de logiciels (y.c. WebRTC)
— interconnexion VoIP sur Internet, multiplexage et trunks, intégration aux réseaux publics et virtualisation de la VoIP dans le cloud
— intégration aux réseaux publics (ISDN/RNIS, POTS, VoIP, ENUM/E.164)
— sensibilisation à la sécurité

support de cours complémentaire : Protocole SIP (Gitlab) et ancien cours Asterisk [a]

a. http://www.cril.ch/pub/CRIL/MarcSCHAEFERHomePage/asterisk.pdf

Voix-sur-IP et téléphonie sur IP Les concepts de voix-sur-IP et téléphonie sur IP sont en général interchangeables, même si, en contexte entreprise, on entend plutôt la deuxième expression [1].

Toutefois, on peut considérer que la voix-sur-IP désigne les technologies qui envoient du trafic *voix* sur un réseau IP, alors que la téléphonie sur IP concerne les technologies qui, en plus, vont permettre l'interconnexion à un réseau public de téléphonie et supportent des services variés attendus sur ce genre de réseau (y compris par exemple, non voix, comme la messagerie texte ou multimédia).

Sauvegardez les captures pcap Ne faites pas uniquement des copies d'écran, mais sauvegardez également les captures en format `pcap` (compatible `Wireshark` et `tcpdump`), cela peut vous être utile pour analyser certains échanges après coup.

1. https://solutions.lesechos.fr/tech/c/la-telephonie-ip-definition-et-avantages-pour-les-entreprises-30306/

Principes de la voix-sur-IP : résumé théorique – 4.2

pour (vidéo)téléphoner sur IP, il faut :

- établir des appels (appeler, être appelé, raccrocher, transférer) : **signalisation**, p.ex. protocole SIP
- transmettre des **flux audio**
  - négocier le codec du flux (voire de flux multiples – vidéo, texte – transmis simultanément), p.ex. **SDP**
  - le **codec** : G.711 (A/μ law), GSM, G.726, G.729, . . .avec des caractéristiques spécifiques (qualité perçue, débit, résistance aux erreurs, etc)
  - le **flux** contenant les données audio ainsi encodées : p.ex. **RTP**

58

Protocoles de voix-sur-IP Il y a principalement trois familles :

1. universel, standard, d'avenir : SIP/SDP/RTP
2. standard, au départ répandu en entreprise, en disparition : H.323
3. très répandu, propriétaire : Skype

Le central Asterisk propose également le protocole IAX2, qui est ouvert et optimise les liens inter-centraux (multiplexage du payload de plusieurs communications, ce que SIP ne sait pas faire). Il passe aussi plus facilement les firewall et NAT (simple flux UDP pour signalisation et audio).

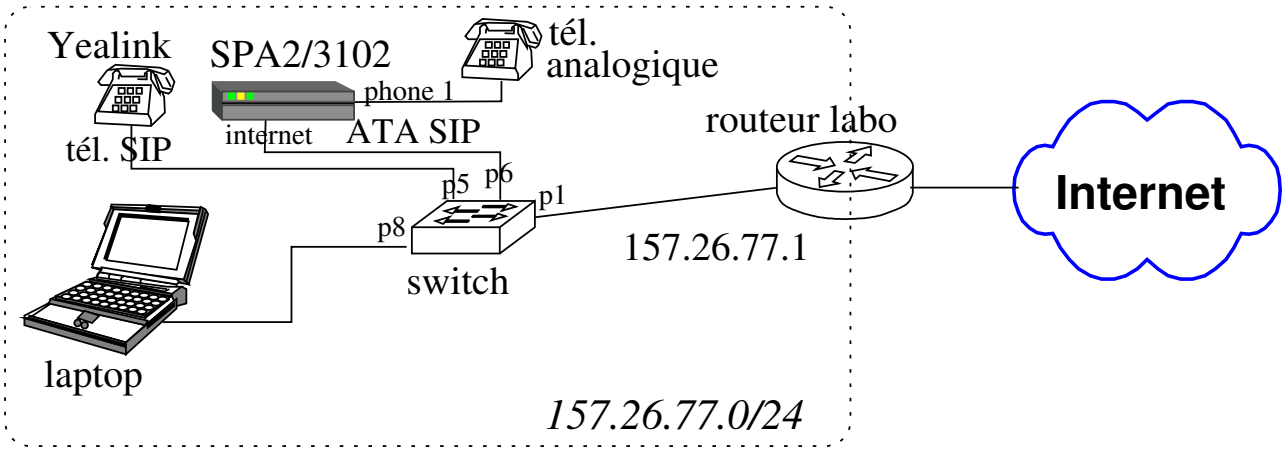
Transport de flux audio RTP transporte un seul flux audio, en général, par exemple encodant 20 millisecondes d'audio (compromis), soit 160 octets utiles en G.711. Il y a donc un coût important des entêtes (efficacité intrinsèque) et un surdébit brut nécessaire. Il existe d'autres protocoles comme p.ex. IAX2 qui permet de multiplexer les flux.

Mise en place d'Asterisk – 4.3

— Asterisk est un logiciel de central téléphonique (PABX), qui offre des fonctions de proxy (annuaire) SIP, des fonctions de conversion de codecs, de passerelles vers la téléphonie fixe (cartes spécifiques), des menus interactifs vocaux (IVR), des fonctions de visiophonie, de conférence, ...
— *dual-licensing* : en GPL, ou, pour certains besoins d'intégrateurs, en propriétaire
59 — exemple d'applications réalisées par l'enseignant
  — système d'authentification pour accès sans-fil public
  — redirection d'appel pour des chauffeurs de taxi
  — conférences téléphoniques du comité scientifique du MAS-ICT
  — système d'alarme avec quittance et désactivation par téléphone
  — blocage d'appels publicitaires

voir aussi le Wiki très complet http://www.voip-info.org/

Mise en place matérielle Le câblage recommandé est proposé ci-dessous :



Branchez un de vos deux laptops en filaire sur le même switch (adaptateurs USB/Ethernet à disposition). En WiFi cela ne fonctionnera pas pour différentes raisons (firewall, authentification, etc).

Description des équipements de voix-sur-IP

SPA2102/3102 il s'agit d'un ATA, qui permet d'intégrer des équipements analogiques à la voix-sur-IP (protocole SIP/RTP ici) via IP sur Ethernet ; il existe en deux modèles dans le laboratoire : SPA2102 avec deux ports pour téléphones, et SPA3102 avec 1 port téléphone (FXS) et un port ligne (FXO) pour brancher à une ligne analogique d'un opérateur télécom ; dans les deux cas l'équipement peut faire également routeur/NAT, bridge, etc

Yealink TP-22 un téléphone SIP/RTP, directement intégrable à IP sur Ethernet

Installation d'Asterisk Assurez-vous avant de la démarrer que la machine virtuelle VirtualBox Debian Buster fournie est en mode bridge sur l'interface *filaire* appropriée, de manière à pouvoir utiliser des équipements matériels contactant votre serveur de téléphonie sur le laptop. **Il faut** regénérer l'adresse MAC virtuelle avant de démarrer la VM Linux. Pour éviter des problèmes, désactivez le WiFi sur votre host et vérifiez que celui-ci a une adresse IP dans 157.26.77.0/24.

Pour l'installation automatique des outils, il faut être à la HE-Arc ou avoir son VPN enclenché : plus simple est de faire (*pas* sous `root`) :

```
wget fs.teleinf.labinfo.eiaj.ch/samba/scratch/R+A/asterisk/auto-install

# contrôle d'intégrité
# 7cf6edb8c5b1b8e5eecf18b3cef8f806954fec78d3ab9f40414d40dd3db6ac50
sha256sum auto-install

# activation
chmod a+rx auto-install

# installation: quelques demandes sudo
./auto-install
```

N'hésitez pas à consulter le script `auto-install` pour comprendre son fonctionnement.

Découverte d'Asterisk – 4.4

— le répertoire `/etc/asterisk` contient la configuration, avec notamment :

sip.conf configuration du protocole SIP et des accès des équipements SIP

extensions.conf configuration des extensions du central (quel numéro correspond à quel service)

60 — on peut communiquer avec le serveur Asterisk notamment grâce à sa console après lancement (option `-rcvvvvvvvvv`)

— on demande à Asterisk de relire ses fichiers de configuration avec `sudo systemctl reload asterisk`[a], ou en utilisant la console Asterisk (`reload`, `sip reload`, ...)

— parfois un `sudo systemctl restart asterisk`[b] est nécessaire – détruira les appels en cours

a. ou `sudo service asterisk reload`
b. ou `sudo service asterisk restart`, mais *pas* `sudo asterisk restart` !

Manipulations et observations

1. affichez la fin du fichier de configuration du protocole SIP : `sudo tail -40 /etc/asterisk/sip.conf`[1]
2. les mots de passe préconfigurés sont-ils stockés hachés ou en clair ?
3. quelles sont les permissions du fichier `sip.conf` et pourquoi y-a-t-il cette restriction ? (indication : `ls -l /etc/asterisk/sip.conf`)
4. quels sont les téléphones SIP configurés par le script `auto-install` ?
5. interagissez avec la console Asterisk
   (a) lancez la console avec `sudo asterisk -rcvvvvvvvvvvv`[2] – voir exemple ci-dessous
   (b) ouvrez un deuxième terminal, lancez une nouvelle console, qu'observez-vous dans la 1ère console ?
   (c) redémarrez Asterisk, que se passe-t-il dans les 2 consoles ?
6. déterminez les téléphones SIP pré-configurés dynamiquement depuis la console Asterisk (indication : `help sip`, en particulier `help sip show`)

1. alternative : commandes `more` ou `less`, ou édition avec `sudo gedit /etc/asterisk/sip.conf`
2. beaucoup de `v`, cf `man asterisk`

Exemple de sortie de la console Asterisk

```
pclabo@debian:~$ sudo asterisk -rcvvvvvvvvvvvvv
[sudo] password for pclabo:
Asterisk 1.6.2.9-2+squeeze4, Copyright (C) 1999 - 2010 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for detail
s.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
  == Parsing '/etc/asterisk/asterisk.conf':   == Found
  == Parsing '/etc/asterisk/extconfig.conf':   == Found
Connected to Asterisk 1.6.2.9-2+squeeze4 currently running on debian (pid = 21634)
Verbosity was 11 and is now 13
debian*CLI> reload
  == Parsing '/etc/asterisk/extconfig.conf':   == Found
  == Parsing '/etc/asterisk/manager.conf':   == Found
  == Parsing '/etc/asterisk/manager.d/README.conf':   == Found
  == Parsing '/etc/asterisk/logger.conf':   == Found
[ ... ]
```

Debugging En cas de problème, on peut debugger de différentes façons :

1. consulter les logs en temps réel : `sudo tail -f /var/log/asterisk/messages`
2. en cas de panne de démarrage d'Asterisk, lancer interactivement avec :
   ```
   # ouvrir shell root
   sudo -s

   # lancer asterisk sous l'utilisateur asterisk, en mode console interactive
   # (sortir avec: stop now)
   su - asterisk -s /bin/bash -c '/usr/sbin/asterisk -cvvvvvvv'
   ```
3. debugger les échanges SIP en activant le debug SIP : `sip set debug on`[3] (désactivation avec `sip set debug off`); consultez `help sip` si nécessaire.

3. on peut aussi limiter à une adresse IP ou à un *peer* donné

Connexion de téléphones – 4.5

Matériel : un téléphone IP Yealink et un ATA SPA2102 ou 3102 avec un téléphone analogique

- brancher et obtenir l'adresse IP de chaque équipement SIP
- configurer les téléphones par interface Web
- effectuez les manipulations et observations ci-dessous
  - vérifier l'enregistrement dans Asterisk
  - tester l'établissement d'appels et le fonctionnement audio

61

(voir configuration dès page 110)

Manipulations et observations En vous aidant des informations ci-dessous :

1. configurez vos deux téléphones (selon les indication dès la page 110) et vérifiez qu'ils s'enregistrent bien dans Asterisk : il y aura un message automatique sur la console (exemple en page 112) et une information d'état avec la commande `sip show peers`
2. capturez l'enregistrement [1] SIP d'un client au serveur Asterisk
   - `Wireshark` sur l'interface réseau utilisée
   - évt. filtrer avec
     `sip`
   - utilisez la fonction Statistics > Flow Graph (options par défaut : Displayed packets, General flow, Standard) pour voir les échanges entre Asterisk et le téléphone)
   - consultez si nécessaire le détail des champs SIP des paquets intéressants (il y aura 2 REGISTER, 1 erreur 401 et statut 200 OK) – voir aussi la question 1 en page 119
3. testez l'appel (p.ex. 200 appelle 201) – vous pouvez accélérer l'appel avec un # final)
4. capturez l'établissement d'appel SIP
   - filtrez par exemple avec `sip`
   - faites d'abord un appel court pour bien visualiser la *signalisation*
   - faites un schéma d'interaction, p.ex. avec le Flow Graph : enregistrement, appel, mise en place des canaux audio, fin de l'appel

1. contiendra notamment une opération `SIP REGISTER` : le plus simple est de couper le courant de l'équipement

— expliquez les protocoles de couches 4+ utilisés, les ports utilisés, indiquez si Asterisk s'authentifie

5. capturez une partie des flux audio RTP (laissez la communication active suffisamment longtemps)
   — filtrez p.ex. avec `sip || rtp`
   — le trafic est-il direct [2] entre équipements ou circule-t-il via [3] Asterisk ?
   — forcez le passage par Asterisk en changeant `canreinvite=yes` en `no` dans la configuration `sip.conf` du compte `200`, puis rechargez avec `sip reload` : vérifiez que l'ensemble du trafic audio passe maintenant par Asterisk
   — évaluez le débit brut [4] en consultant par exemple la taille totale des paquets RTP et le temps moyen entre chaque paquet, puis comparez avec le début net (débit causé par l'audio seulement, vous pouvez cliquer tout en bas sous Payload pour voir sa taille en octet)
   — les données audio sont elles vraisemblablement en clair ?
   — quel est le codec négocié en SDP ?
   — si vous tapez une touche du clavier du téléphone (DTMF), comment celle-ci est-elle transmise [5] (en-bande : sous forme d'un son ; ou hors-bande : sous forme d'un paquet SIP ou d'une métadonnée RTP) ?

Informations nécessaires

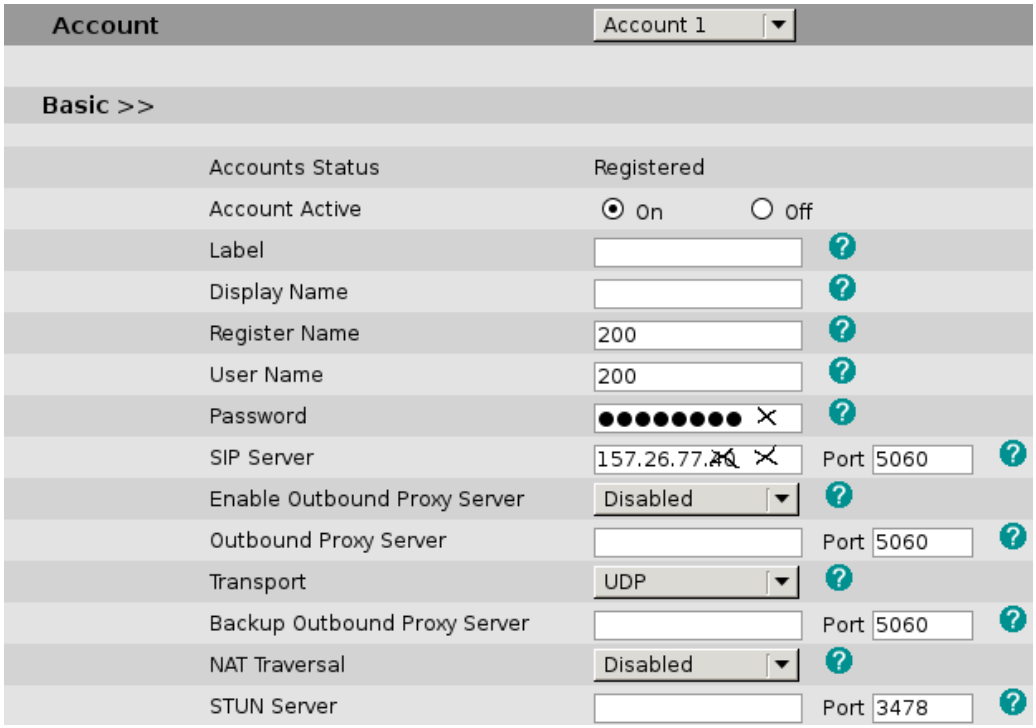
sip proxy	adresse IP de votre serveur Asterisk (cf commande `ip addr show`, prenez l'adresse qui est dans le sous-réseau 77)
phone number	voir `/etc/asterisk/sip.conf`, 200-201
account	idem
pin	voir le mot de passe (`secret`) pour cette entrée dans `/etc/asterisk/sip.conf`

Chaque téléphone aura un numéro unique (200 pour le Yealink, 201 pour l'ATA).

Téléphones Yealink TP-22 (à brancher sur le port Internet de préférence)

— deux fois la 4e touche à droite (Menu > Enter) pour l'adresse IP
— adresse Web : `http://157.26.77.X`, avec X ci-dessus
— login : `admin` [6], mot de passe Web : `admin`
— configurer le compte SIP sous Account
   — Account active : ON
   — Register et username : 200
   — Password : cf `sip.conf` pour le compte 200
   — SIP server : adresse IP de votre serveur Asterisk
— en cas de nécessité : reset du téléphone avec bouton OK pendant 10 secondes

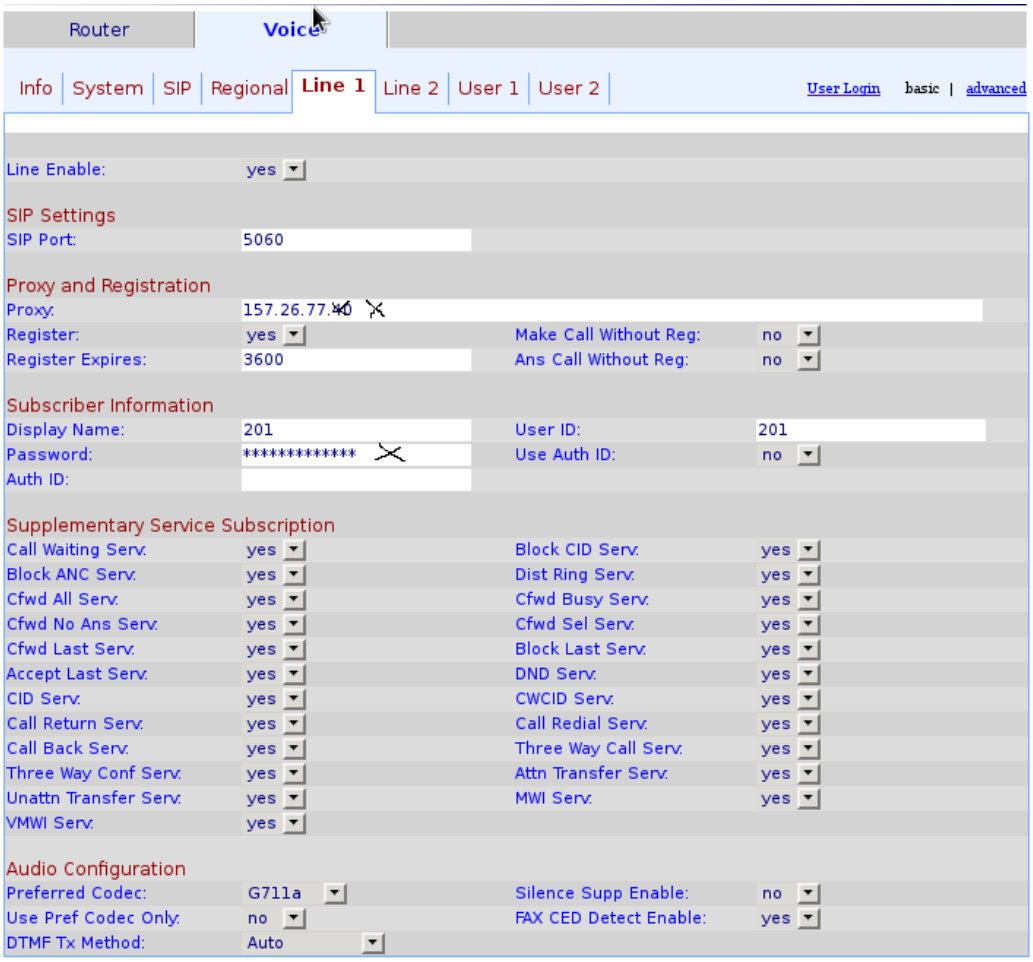
2. il se peut que les premiers échanges RTP passent par Asterisk, mais vous verrez tout de suite si cela s'arrête : soit par la capture, soit en observant les LEDs du switch
3. http://www.voip-info.org/wiki/view/Asterisk+SIP+media+path
4. débit total, y compris entêtes
5. http://www.voip-info.org/wiki/view/Asterisk+sip+dtmfmode
6. en cas de déploiement réel, pensez également à mettre un mot de passe à l'utilisateur, voir menu Sécurité



ATA SPA2102 et 3102 Doit être branché sur le port Internet (WAN), et **pas** Ethernet (par défaut : sous-réseau privé avec DHCP, ce qui interférerait avec le sous-réseau du laboratoire). Le téléphone analogique doit être connecté au *phone 1*.

- avec un téléphone analogique branché sur *phone 1*, composez dans le vide 4 étoiles (`****`) puis lorsque le message audio vient, les touches `877771#` [7], puis confirmez avec `1` et raccrochez : cela va réinitialiser la configuration utilisateur de l'ATA
- de même, composez `****` puis `73738#` et confirmez avec `1` et raccrochez : cela réinitialisera la configuration de base de l'ATA
- activez l'interface d'administration par WAN (Internet) avec `****` puis `7932#`, puis `1#`, puis `1`
- obtenez l'adresse IP du côté WAN (Internet) de l'ATA avec `****` puis `110#`
- connectez vous à l'interface Web de cette adresse IP, activez la fonction `Admin login`, puis modifiez la configuration de la *Ligne 1* comme ci-dessous : entrez le mot de passe correspondant au numéro et adaptez l'adresse du *SIP Server* (c'est votre PC de laboratoire).

7. si ne marche pas, vérifiez d'abord que votre téléphone analogique envoie des sons DTMF et non pas des clacs multiples (composition par impulsion) : dans ce dernier cas pressez longtemps sur la touche `*` pour passer en mode DTMF; réessayez aussi plus lentement, si code inconnu, passez à la suite



Dans un cas réel, on configurerait le mot de passe des comptes `admin` et `user` et on configurerait éventuellement en mode bridge.

Exemple de messages Asterisk et état Après configuration, voici des exemples de messages Asterisk lorsque des téléphones matériels s'enregistrent en SIP et une commande spécifique d'interrogation d'état :

```
 -- Registered SIP '200' at 157.26.77.116:5062
    > Saved useragent "Yealink SIP-T22P 7.61.0.80" for peer 200
[Feb 28 11:28:00] NOTICE[14806]: chan_sip.c:23583 handle_response_peerpoke: Peer '200'
   is now Reachable. (49ms / 1000ms)

 -- Registered SIP '201' at 157.26.77.115:5060
    > Saved useragent "Linksys/SPA2102-3.3.6" for peer 201
[Feb 28 11:42:51] NOTICE[14806]: chan_sip.c:23583 handle_response_peerpoke: Peer '201'
   is now Reachable. (5ms / 1000ms)

poste-403*CLI> sip show peers
Name/username Host            Dyn Forcerport ACL Port     Status
200/200       157.26.77.116    D                1720     OK (152 ms)
201/201       157.26.77.115    D                5062     OK (253 ms)
```

Dial plan et services Asterisk – 4.6

— un téléphone (un login SIP spécifique) a un `contexte` associé qui définit dans quel `dial plan` numéroter
— un dial-plan est composé d'extensions
— les extensions sont les numéros appelables, et se décomposent en séquences numérotées effectuant des opérations

62

Manipulations et observations

1. consultez le fichier `/etc/asterisk/extensions.conf` et déterminez quels services du PABX vous pouvez appeler depuis votre téléphone SIP.
   — ils sont, dans cette configuration spécifique d'Asterisk, sous le dial-plan `service-numbers` que vous pouvez aussi interroger dans une console Asterisk avec : `dialplan show service-numbers` (utilisez la touche TAB pour compléter !)
2. vérifiez si votre téléphone peut appeler le contexte / dial-plan `service-numbers` avec `dialplan show VOTRE-CONTEXTE` avec `VOTRE-CONTEXTE` le contexte dans lequel votre téléphone agit (voir `sip.conf`)
3. regardez la séquence (dans `extensions.conf`) qui traite le numéro `600`, voyez qu'il n'est pas dans le contexte du téléphone par défaut et déterminez comment on l'appelle (indication : on entre dans le dialplan `demo` avec un `goto`)
4. au début de la séquence du numéro `508` dans le dialplan `service-numbers`, ajoutez un `Answer`, puis un `Playback` d'un fichier avant de lancer la fonction `Echo` (voir exemple du `600` et ci-après *séquence complexe*) ; rechargez Asterisk [1] et testez.

1. `reload` dans la console Asterisk suffit

Exemple de séquence complexe Asterisk Exemple classique [2] :

```
exten => 6123,1,faire quelque chose
exten => 6123,2,faire autre chose
exten => 6123,4,encore faire quelque chose
```

Dans l'exemple ci-dessus, si l'on tape 6123 dans le contexte, les numéros d'opération 1 et 2 seront exécutés : ensuite l'appel sera raccroché, car il manque le 3 pour que le 4 soit exécuté.

On peut simplifier la numérotation en remplaçant les numéros de séquences subséquents à `1` par `n` (numérotation automatique).

2. voir aussi https://wiki.asterisk.org/wiki/display/AST/Contexts,+Extensions,+and+Priorities

Interconnexion VoIP – 4.7

On va montrer un type d'interconnexion entre centraux Asterisk, à l'aide du protocole IAX2. Ici, l'interconnexion sera par le sous-réseau du laboratoire 157.26.77.0/24 où toutes les VM sont bridgées :

— appel d'un téléphone d'un autre groupe en VoIP et évaluer la qualité et les délais

63 La vidéo https://www.youtube.com/watch?v=OPHVNSyG0OM décrit le design et l'implémentation d'architectures VoIP modernes.

Relier au réseau public en VoIP Pour relier au réseau public, il faut un opérateur de téléphonie sur IP, un protocole de signalisation, un protocole de transport d'audio et un ou des codec(s) supportés.

La plupart du temps, le protocole utilisé sur Internet sera SIP, avec transport RTP et codec G.711 [1]. Même en cas de *trunk* [2] SIP, aucun multiplexage de signalisation ni de transport n'est effectué dans SIP (plusieurs communications sont multiplexées par IP).

Dans notre cas, nous allons utiliser le protocole open source IAX2 d'Asterisk, qui permet un multiplexage de plusieurs communications sur le même flot UDP. Il a aussi l'avantage de mieux passer les NAT.

Utiliser un VPN permet d'assurer la sécurité des communications (intégrité, confidentialité) même si des protocoles de couche 7 sécurisés ne sont pas utilisés (SIPS et SRTP). Toutefois, il peut ajouter des délais.

1. PCM 8000 échantillons par seconde de 8 bits, donc 64 kbit/s ; avec une taille d'escalier non constante selon la loi A (Europe) ou μ (Etats-Unis) pour privilégier les détails des faibles amplitude
2. réunion de plusieurs numéros d'appels dans un échange logique

Appel d'un téléphone d'un autre étudiant Demander à un autre étudiant son adresse IP sous 157.26.77.0/24 et ajouter un contexte supplémentaire dans `/etc/asterisk/extensions.conf`:

```
[inter-dial]
exten => _7XXXXXX,1,Dial(IAX2/157.26.77.$[ 0 + ${EXTEN:1:3}]/${EXTEN:4})
```

Inclure ce contexte dans `local-sip-in`. Soit le numéro `7002500`, expliquer quelle adresse IP sera contactée en protocole IAX2 et quelle extension distante sera appelée.

Par défaut, IAX2 nécessite une authentification, quelques exemples sont à disposition dans `/etc/asterisk/iax.conf`. Ici, nous allons simplement utiliser une authentification par adresse IP, ce qui n'est pas un gros problème dans un réseau privé.

Sous `root` ou `sudo` :

```
mv /etc/asterisk/iax.conf /etc/asterisk/iax.conf.OLD
cat > /etc/asterisk/iax.conf <<"EOF"
[general]
disallow=all
allow=alaw
jitterbuffer=no


[guest]
type=user
context=local-sip-in
permit=157.26.77.0/24

EOF
```

Faire `sudo systemctl reload asterisk` pour activer les modifications de configuration.

Lancer la console Asterisk pour voir l'appel puis appeler le service 500 (éventuellement ensuite le 600 pour l'écho) sur un autre serveur, en composant le bon numéro commençant par 7.

Evaluer la qualité générale et le délai.

Eléments à choix – 4.8

Effectuer au moins un, à choix, si vous voulez rendre le laboratoire !

Ces compléments peuvent aussi vous être utiles pour répondre aux questions.

- utilisez un client SIP logiciel (p.ex. dans le navigateur avec API WebRTC [a] avec pile SIP/RTP) et montrez les protocoles utilisés
- contrôlez Asterisk par logiciel (interface Manager : TCP brut, ou par une API de votre langage préféré, y.c. REST)
- utilisez un autre codec (p.ex. G729, pas supporté par Asterisk – licence – mais par les téléphones) et évaluez le débit approximatif et la qualité perçue
- faites une conversion de codec par Asterisk et évaluez la performance nécessaire
- créez un système de menus vocaux (IVR)

a. https://www.doubango.org/sipml5/

64

Autre codec Déterminez (par la capture RTP ou les options négociés par SDP sur SIP) quel est le codec utilisé par défaut et indiquez ses paramètres (débit, échantillons, etc).

P.ex. :

```
sudo egrep '^allow=' /etc/asterisk/sip.conf | sort -u
allow=alaw
```

Configurez un autre codec p.ex. GSM ou ADPCM (G.726). Comparez ces codecs. Pour ce faire il vous faut modifier `/etc/asterisk/sip.conf` (lignes allow) pour chaque téléphone, voire, si les codecs y ont été désactivés, la configuration des téléphones eux-mêmes.

Fonctions simples de centraux Le but est ici, en modifiant extensions.conf, de créer des fonctions classiques de central. Par exemple une fonction d'appel sur plusieurs terminaux. Nous allons configurer l'extension 300 pour appeler les deux téléphones.

Sachant que le séparateur & dans un argument de la commande `Dial` permet d'appeler plusieurs destinataires simultanément, configurez une extension 300 qui appelle vos deux téléphones.

Trunks IAX2 IAX2 est le protocole natif d'Asterisk, utilisant un seul port UDP et très performant pour les trunks.

L'idée est ici que si l'on compose 8pp200, on appelle le téléphone SIP 200 de l'Asterisk du poste pp. Par exemple 804200 pour pp=04

Déterminez où cela est déjà préconfiguré, comprenez le fonctionnement, adaptez la configuration et testez avec votre collègue (ou le poste de l'enseignant, poste-400, numéro 800500 s'il l'a préparé). N'hésitez pas à capturer.

E.164/ENUM Faites quelques requêtes DNS grâce à la commande host, p.ex.

```
host -t naptr 8.9.6.8.6.2.0.3.1.8.6.9.4.e164.arpa

8.9.6.8.6.2.0.3.1.8.6.9.4.e164.arpa has NAPTR record 20 10 "u"
    "E2U+sip"
    "!^\\+49681302(.*)$!sip:\\1@enum.rz.uni-saarland.de!" .
8.9.6.8.6.2.0.3.1.8.6.9.4.e164.arpa has NAPTR record 30 10 "u"
   "E2U+h323"
    "!^\\+49681302(.*)$!h323:\\1@h323server.rz.uni-saarland.de!" .
8.9.6.8.6.2.0.3.1.8.6.9.4.e164.arpa has NAPTR record 10 10 "u"
   "E2U+http"
    "!^.*$!http://www.rz.uni-saarland.de/projekte/!" .
8.9.6.8.6.2.0.3.1.8.6.9.4.e164.arpa has NAPTR record 10 10 "u"
   "E2U+iax2"
    "!^\\+49681302(.*)$!iax2:guest@enum.rz.uni-saarland.de/\\1!" .
8.9.6.8.6.2.0.3.1.8.6.9.4.e164.arpa has NAPTR record 10 20 "u"
   "E2U+iax"
    "!^\\+49681302(.*)$!iax:guest@enum.rz.uni-saarland.de/\\1!"
```

Vous y reconnaîtrez facilement du pattern-matching en expressions régulières et remplacements ! Deux espaces de nommages existent : l'officiel (e164.arpa) et le communautaire [1]

Que pensez-vous de l'impact sur la sécurité générale ?

Provisioning En télécom, on parlait de "provisioning" pour la planification de la mise à disposition de ressources (p.ex. commander le matériel, réserver les équipes d'installation, ...) Aujourd'hui, cela désigne plutôt, au sein d'un réseau informatique, la gestion de configuration automatisée, pour réaliser un service donné.

Effectuez les étapes suivantes :

1. réfléchissez aux étapes à réaliser pour que Mme Dupont, nouvelle secrétaire récemment engagée, puisse téléphoner depuis sa place de travail et dispose de son propre numéro interne et d'une boîte vocale
2. pour un type de téléphone donné (Polycom [2] IP 335), documentez-vous sur les phases de configuration qui peuvent être automatisées, et si le temps le permet, mettez en place une telle solution
3. déterminez quels fichiers changer puis écrivez une ébauche d'un script qui adapte automatiquement la configuration de votre serveur Asterisk en fonction des besoins exprimés

1. http://www.nrenum.net/
2. http://docs.polycom.com/global/documents/whitepapers/uc-software-provisioning-best-practices-whitepaper.pdf

Questions – 4.9

Toutes ces questions ainsi que le laboratoire (grandes lignes) font partie de la matière du prochain questionnaire.

1. lorsqu'un mot de passe est stocké en clair dans la configuration Asterisk, quel protocole d'authentification est utilisé pour ne pas le transmettre en clair sur le réseau ? (consultez par exemple votre capture du REGISTER SIP et expliquez en particulier pourquoi la 1ère demande REGISTER produit une erreur 401, et en quoi le deuxième REGISTER est différent : il y a peut-être une information transmise par le serveur dans l'erreur 401 qui est utilisée ?)

   indications
   — notion de *nonce* : `https://en.wikipedia.org/wiki/Cryptographic_nonce` et de protocole d'authentification *challenge-response* (voir schéma du protocole)
   — ainsi que, plus généralement, la notion de *hachage cryptographique* : `https://fr.wikibooks.org/wiki/Les_fonctions_de_hachage_cryptographiques`
   — et une application connexe, le hachage avec graine aléatoire connue pour les mots de passe afin de rendre une attaque par *Rainbow tables* moins facile : voir
   `sudo grep root /etc/shadow` pour un exemple de hash
   `man 5 shadow` pour le format de ce fichier
   `man 3 crypt` pour les types de hachages qui y sont utilisés (sous NOTES, Glibc notes)

2. que pensez-vous de mettre ces téléphones SIP sur un réseau général ? (indication : y-a-t-il des attaques de sécurité contre le firmware, cf wiki Asterisk) que proposez-vous ?

3. si vous exploitez un Asterisk accessible par Internet, que devriez-vous vérifier dans la configuration ?

4. comment faire, avec Asterisk, pour ne pas devoir accélérer la composition d'un numéro avec un # final ? (indication : soit le téléphone doit connaître la structure du numéro de téléphone et donc tout ou partie du *dial plan*, soit il doit poser la question à Asterisk)

5. SIP et RTP fonctionnent-ils bien en présence d'un NAT/firewall (p.ex. accès ADSL via un routeur) ? En quoi le service STUN est utile ?

6. quels sont les avantages et inconvénients de faire passer le trafic audio (RTP) via le serveur Asterisk ? dans quel(s) cas c'est obligatoire ?

7. montrez par un calcul d'efficacité intrinsèque [3] que le débit brut audio mesuré en codec G.711 était attendu ; expliquez la conséquence si l'on a 30 appels téléphoniques en parallèle et proposez une solution pour diminuer le débit brut total

8. quel protocole permet de chiffrer RTP ?

9. pourquoi est-ce une bonne idée de détecter les DTMF aux bords analogiques d'un réseau VoIP et de transmettre, à l'intérieur de ce réseau, les DTMF sous forme *hors-bande* ?

10. comment connecter un téléphone analogique, voire ISDN, mais non voix-sur-IP à un central voix-sur-IP ?

3. voir livre Protocoles et réseaux

11. quel est le rôle du protocole SS7 ? (voir par exemple : https://fr.wikipedia.org/wiki/Signaling_System_7, section historique)

12. dans les NGN, avec un core IMS, quel est le protocole de signalisation moderne utilisé ? (voir par exemple https://fr.wikipedia.org/wiki/IP_multimedia_subsystem#Gateways ou https://en.wikipedia.org/wiki/Next-generation_network#Underlying_technology_components)

Questions pour le rapport – 4.10

Ces questions ne font pas partie de la matière pour le questionnaire.

1. quel autre logiciel qu'Asterisk existe-t-il comme PABX (central téléphonique complet) ?

2. quelle alternative plus simple qu'Asterisk, effectuant uniquement la fonction de proxy SIP, existe ?

3. Asterisk supporte-il quels autres modes d'authentification que par mot de passe simple via authentification *challenge-response* ?

4. proposez une méthode expérimentale simple permettant d'évaluer le délai audio d'une communication téléphonique

5. donnez quelques fournisseur de téléphonie sur IP en Suisse et quels protocoles et codecs supportent-ils ? sont-ils sécurisés ?

6. on aimerait brancher Asterisk au réseau Swisscom, via la prise téléphone analogique du routeur xDSL Swisscom, quel équipement nous faudrait-il ? est-ce possible de connecter directement et nativement en voix-sur-IP ?

7. comment peut-on déployer de manière efficace quelques centaines de téléphones configurés automatiquement ? (exemple, nom ou type de produit, URL)

8. qu'est-ce que le protocole *Skinny* ?

9. comment E.164/ENUM permet-il d'intégrer la téléphonie classique à Internet ?

10. peut-on faire des appels vidéo avec Asterisk ?

11. existe-t-il une passerelle entre Asterisk et MICROSOFT SKYPE ? y-a-t-il des limitations ?

12. comment faire du routage d'appel intelligent basé sur les coûts (*Least Cost Routing*) avec Asterisk ?

13. Asterisk peut être contrôlé programmatiquement par des fichiers de queue, un socket TCP et un protocole spécifique, ou par la technologie REST : donnez un exemple de code PHP, pas forcément 100% fonctionnel, pour lister les communications en cours.

14. quel est l'équivalent Cisco de l'Asterisk Manager Interface ?

15. évaluez la sécurité des ATA utilisés (p.ex. le fait qu'ils sont configurables par DTMF)
16. comment assurer la haute fiabilité en cas de panne de la liaison fixe VoIP principale (p.ex. ligne DSL tombe en panne) : donnez un exemple *clé en main* par le fournisseur et un exemple de produit intégrant à la VoIP plusieurs liaisons sans fil GSM (p.ex. VoLTE).
Indication : Swisscom et Beronet VoLTE gateway.