

ALCASAR : un contrôleur d'accès au réseau libre et gratuit
ALCASAR : a free & open source NAC (Network Access Controller)



www.alcasar.net



Documentation technique

Projet : Sécurisation des accès Internet	Auteur : Rexy with helps by alcasar team
Objet : Documentation technique du projet	Version : 3.5
Mots clés : Contrôleur d'accès au réseau, Network Access Control (NAC), portail captif, captive portal, coovachilli	Date : juin 2020

Table des matières

1 - Rappel de l'architecture.....	3
2 - Choix des constituants.....	3
2.1 - La passerelle d'interception.....	3
2.2 - Les autres constituants.....	4
3 - Schémas de principe :.....	5
4 - Fonction « interception / authentification ».....	7
4.1 - la passerelle « coova-chilli ».....	7
4.1.1 - Fonctionnement de l'interception (capture).....	7
4.1.2 - Exception à l'authentification.....	10
4.2 - Base de données des usagers.....	10
4.2.1 - Accès en mode graphique.....	13
4.2.2 - Accès en mode console.....	13
4.3 - Le serveur Radius.....	14
4.3.1 - <i>Module A.D./LDAP externe</i>	16
5 - Fonction « traçabilité et imputabilité ».....	17
5.1 - Journalisation principale.....	17
5.2 - journalisation accessoire.....	17
5.3 - Constitution de l'archive de traçabilité.....	17
6 - Fonction « filtrage ».....	18
6.1 - Filtrage de protocoles réseau.....	18
6.2 - Filtrage de noms de domaines, d'URLs et d'adresses IP.....	18
6.2.1 - Traitement de la liste de Toulouse.....	19
6.2.2 - Filtrage par usager/groupe.....	19
6.2.3 - Double filtrage de la WhiteList (WL).....	20
6.2.4 - Filtrage avec la BlackList (BL).....	21
6.3 - Antivirus WEB.....	21
7 - Fonction « Interface de gestion ».....	21
8 - Fonction « modules complémentaires ».....	22
8.1 - Import de comptes.....	22
8.2 - Auto-inscription par SMS.....	22
8.2.1 - Fonctionnement global.....	22
8.2.2 - Dialogue avec le modem-GSM – déblocage avec le code PUK.....	24
8.3 - Watchdog.....	25
8.4 - Statistiques réseau.....	25
8.5 - Contournement (by-pass).....	27
8.6 - Load balancing de connexions.....	27
8.7 - Re-Horodatage des fichiers journaux.....	28
8.8 - Sauvegardes.....	28
8.8.1 - Sauvegarde des journaux de traçabilité.....	28
8.8.2 - Sauvegarde de la base de données.....	28
8.8.3 - Le rapport d'activité hebdomadaire.....	28
9 - Annexes.....	29
9.1 - Coova-chilli.....	29
9.2 - Freeradius.....	29
9.3 - Dnsmasq.....	30
9.4 - Parefeu.....	30
9.5 - E ² Guardian.....	30
9.6 - Tinyproxy.....	30
9.7 - Ulogd.....	30
9.8 - HAVP + Clamav.....	30
9.9 - Distribution Mageia et ses dépôts.....	30
9.10 - Étude du remplacement de DNSMasq par Unbound.....	32
10 - Plan de tests.....	33
11 - Security tests.....	38

1 - Rappel de l'architecture




ALCASAR est positionné en coupure entre l'accès Internet et le réseau de consultation. Il permet d'authentifier les usagers, de contrôler les accès, de tracer les connexions effectuées, de protéger le réseau de consultation. Le cœur d'ALCASAR est constitué des éléments traditionnels d'un contrôleur d'accès au réseau (NAC = Network Access Control) :

- un serveur d'authentification (freeradius),
- une passerelle d'interconnexion (portail captif) et client radius (coovachilli)
- une base de données d'utilisateurs (mariadb)
- un pare-feu dynamique (netfilter + ipset)
- un système de filtrage (E²Guardian + clamd + DNSMasq)

2 - Choix des constituants

2.1 - La passerelle d'interception

Avec le pare-feu, la « passerelle d'interception » (ou portail captif) constitue le chef d'orchestre de cet ensemble. Pour choisir celle qui serait intégrée dans ALCASAR, les passerelles libres suivantes ont été évaluées au lancement du projet :

	NoCat 	Talweg	Wifidog 	Chillispot/ 
Site WEB	nocat.net	talweg.univ-metz.fr	dev.wifidog.org	www.chillispot.org www.coovachilli.org
Langage	C	C# sous mono	- C pour le programme principal - module PHP pour le serveur WEB	- C pour le programme principal - module CGI-BIN pour le serveur WEB (PERL ou C)
Description	NoCat est constitué de plusieurs éléments : « NoCatSplash » est le portail, « NoCatAuth » est utilisé pour l'authentification et « Splash Server » est le service permettant de générer les formulaires de connexion des utilisateurs. Le suivi de ce produit a été arrêté en 2005.	Talweg est un portail dont le contrôle d'accès au réseau est géré protocole par protocole. Tous les protocoles utilisables sur Internet ne sont pas encore intégrés.	WifiDog est composé de 2 modules : « Authentification Server » et « WifiDog Gateway ». Le serveur d'authentification doit être installé sur un serveur fixe alors que la passerelle peut être embarquée dans certains équipements réseau compatibles (routeur, passerelle ADSL, etc.).	Chillispot ne constitue que la partie centrale d'une architecture de type portail captif. Il implémente les 2 méthodes d'authentification (UAM et WPA). Il nécessite la connaissance et l'installation des autres services constitutifs du portail captif. Il est compatible radius.

	NoCat	Talweg	Wifidog	Chillispot
Simplicité d'installation				
Infrastructure nécessaire				
Performances & consommation réseau				
Gestion utilisateurs				
Sécurité authentification				
Sécurité communications				
Protocoles supportés				
Crédit temps				
Interface d'administration / Statistiques				

Légende:
 : Non Disponible.
 : Plus ou moins.

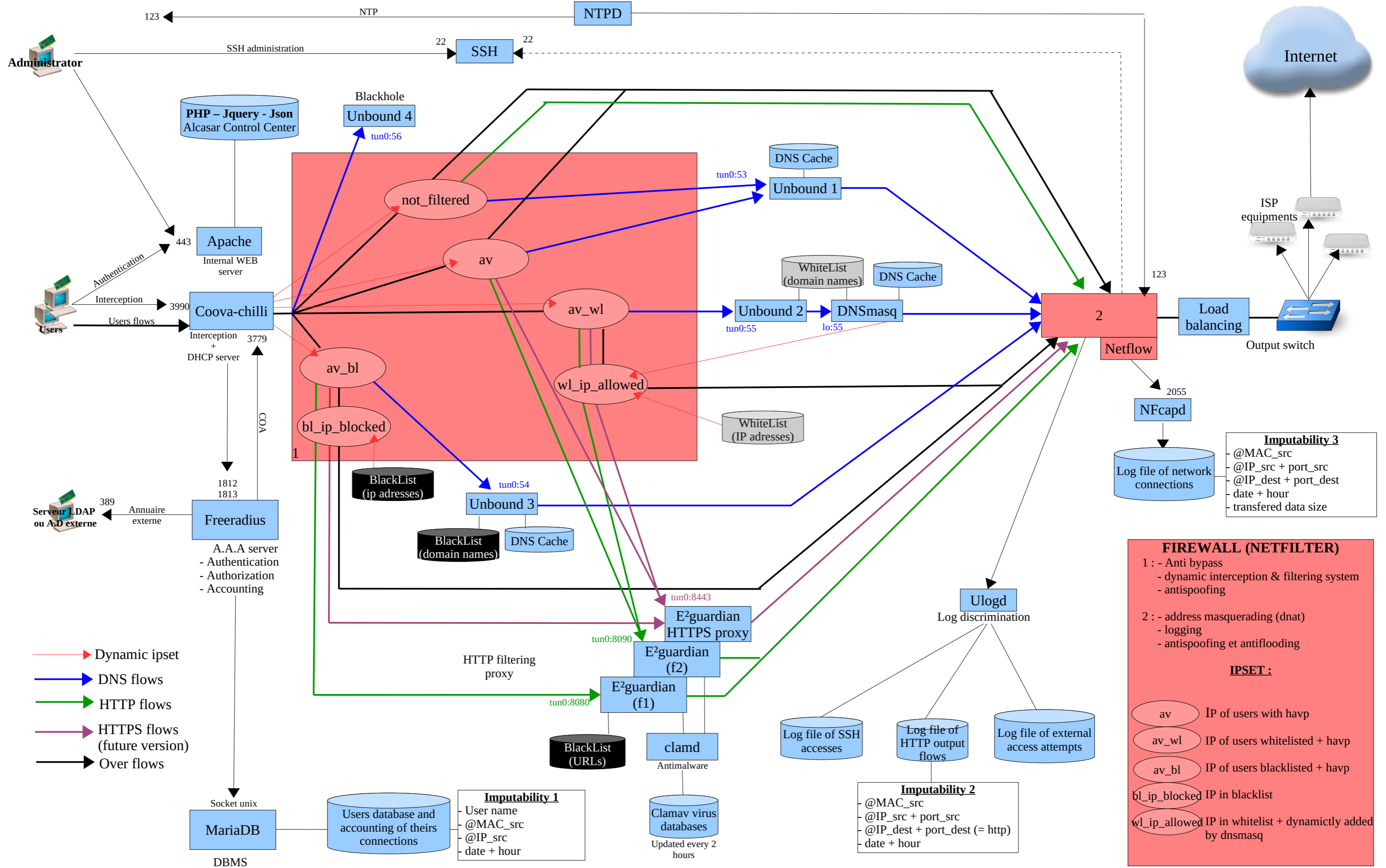
Bien que cette liste ne soit pas exhaustive, la passerelle « Chillispot » a été choisie. Depuis, elle a été remplacée par le clone (fork) « coova-chilli » dont le développement est plus actif (<http://coova.org/CoovaChilli>). Avant chaque nouvelle version d'ALCASAR, le code source du projet « coova-chilli » est récupéré, compilé spécifiquement pour ALCASAR et empaqueté (RPM) pour être intégré à la distribution Linux choisie pour ALCASAR.

3 - Schémas de principe :

ALCASAR peut être décomposé en cinq fonctions qui sont détaillées dans la suite du document :

- fonction « interception / authentification » réalisée par Coova-chilli, DNSMasq, Lighttpd et le couple (Freeradius , MariaDB). Possibilité de couple (Freeradius , LDAP externe) pour l'authentification ;
- fonction « traçabilité / imputabilité des connexions » constituée des flux Netflow, des journaux du pare-feu et du couple (Freeradius , MariaDB) ;
- fonction « filtrage » (de domaine, d'URL, d'adresses IP, de malwares et de protocoles réseau). Ces dispositifs de filtrage sont réalisés par le pare-feu (Netfilter), l'antimalware Clamav (clamd), 4 instances de DNSMasq et E²Guardian ;
- fonction « interface de gestion » réalisée en PHP / HTML4 & 5/ JQuery / PERL et servie par Lighttpd ;
- fonction « modules complémentaires ». Ces modules ont pour objectif d'améliorer la sécurité globale du portail (anti-contournement, anti-usurpation MAC/IP, chiffrement des fichiers journaux, gestion des certificats, IDS, etc.) ou d'enrichir les possibilités du portail (installation, mise à jour, by-pass, archivage, accélération de la consultation, cron, etc.)

ALCASAR – ARCHITECTURE



4 - Fonction « interception / authentification »

Un des objectifs d'ALCASAR est d'être le plus universel possible. Ainsi, la méthode d'interception et d'authentification choisie s'appuie sur l'« UAM » (Universal Access Method). Cette méthode n'utilise que des protocoles standards ne nécessitant qu'un navigateur WEB pour authentifier un usager situé sur un équipement de consultation. Parmi les autres méthodes, on peut citer celle exploitant des agents clients à installer sur les équipements de consultation (méthode exploitée par certains parefeux authentifiants) ou celle reposant sur des protocoles réseau dédiés (802.1X par exemple).

La fonction « interception / authentification » s'appuie sur la passerelle d'interception « Coova-chilli » (processus « chilli »), le serveur WEB « Lighttpd » (processus « httpd »), le serveur d'authentification « Freeradius » (processus « radiusd ») et le système de gestion de bases de données « MariaDB » (processus « mysqlmanager » et « mysqld »)

4.1 - la passerelle « coova-chilli »

Elle est lancée via son script de démarrage (`/etc/rc.d/init.d/chilli start`) qui a été légèrement adapté par le script d'installation (« `alcasar.sh` »). Ce script utilise le fichier de configuration (« `/etc/chilli.conf` »). Le processus « chilli » est alors lancé en mode « daemon ». Ce dernier crée l'interface virtuelle « tun0 »¹ liée en point à point à l'interface physique connectée au réseau de consultation (eth1). Cela lui permet de gérer sa propre table de résolution ARP en espace utilisateur. Une particularité dans cette gestion consiste à verrouiller les couples (@MAC , @IP) rencontrés sur le réseau de consultation. Un empoisonnement du cache ARP par le réseau est alors impossible (« cache poisoning »). Dans certains cas, ce comportement peut être bloquant (équipement reparamétré après avoir déjà généré des trames IP vers ALCASAR). La commande « `chilli-query list` » permet d'afficher et de contrôler le cache ARP de « chilli ». Cette commande est utilisée par l'interface de gestion (menu « ACTIVITÉ ») pour supprimer les mauvaises associations @IP/@MAC. Complémentaire à cette fonction d'anti-« cache poisoning » intégrée à « chilli », ALCASAR utilise un module spécifique de sécurité (`alcasar-watchdog.sh`) permettant d'éviter l'usurpation d'adresses MAC et d'adresses IP des stations de consultation connectées sur le réseau (cf. fonctions de sécurité).

4.1.1 - Fonctionnement de l'interception (capture)

Lorsqu'un équipement de consultation tente de se connecter sur une URL Internet (www.free.fr dans l'exemple qui suit) :

No. -	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.182.129	192.168.182.1	DNS	Standard query A www.free.fr
2	0.007977	192.168.182.1	192.168.182.129	DNS	Standard query response A 212.27.48.10
3	0.013100	192.168.182.129	212.27.48.10	TCP	iclpv-nlc > http [SYN] Seq=0 win=65535 Len=0 MSS=1460 WS=1
4	0.018826	212.27.48.10	192.168.182.129	TCP	http > iclpv-nlc [SYN, ACK] Seq=0 Ack=1 win=5840 Len=0 MSS=
5	0.025258	192.168.182.129	212.27.48.10	TCP	iclpv-nlc > http [ACK] Seq=1 Ack=1 win=96768 Len=0
6	0.095262	192.168.182.129	212.27.48.10	HTTP	GET / HTTP/1.1
7	0.112659	212.27.48.10	192.168.182.129	TCP	http > iclpv-nlc [ACK] Seq=1 Ack=384 win=6912 Len=0
8	0.119483	212.27.48.10	192.168.182.129	TCP	[TCP segment of a reassembled PDU]
9	0.122574	212.27.48.10	192.168.182.129	HTTP	HTTP/1.1 302 Moved Temporarily (text/html)
10	0.126880	212.27.48.10	192.168.182.129	TCP	http > iclpv-nlc [FIN, ACK] Seq=1511 Ack=384 win=6912 Len=0
11	0.131796	192.168.182.129	212.27.48.10	TCP	iclpv-nlc > http [ACK] Seq=384 Ack=1512 win=96768 Len=0
12	0.135296	192.168.182.129	212.27.48.10	TCP	iclpv-nlc > http [FIN, ACK] Seq=384 Ack=1512 win=96768 Len=0
13	0.142579	212.27.48.10	192.168.182.129	TCP	http > iclpv-nlc [ACK] Seq=1512 Ack=385 win=6912 Len=0
14	0.173829	192.168.182.129	192.168.182.1	TCP	iclpv-wsm > https [SYN] Seq=0 win=65535 Len=0 MSS=1460 WS=1
15	0.182402	192.168.182.1	192.168.182.129	TCP	https > iclpv-wsm [SYN, ACK] Seq=0 Ack=1 win=5840 Len=0 MSS=
16	0.182724	192.168.182.129	192.168.182.1	TCP	iclpv-wsm > https [ACK] Seq=1 Ack=1 win=96768 Len=0
17	0.259992	192.168.182.129	192.168.182.1	SSL	Client Hello
18	0.266048	192.168.182.1	192.168.182.129	TCP	https > iclpv-wsm [ACK] Seq=1 Ack=367 win=6912 Len=0
19	0.268301	192.168.182.1	192.168.182.129	TLSv1	Server Hello, Change Cipher Spec, Encrypted Handshake Mess

```
Transmission Control Protocol, Src Port: http (80), Dst Port: iclpv-nlc (3394), Seq: 1461, Ack: 384, Len: 0
[Reassembled TCP Segments (1510 bytes): #8(1460), #9(50)]
Hypertext Transfer Protocol
  HTTP/1.1 302 Moved Temporarily\r\n
    connection: close\r\n
    Cache-Control: no-cache, must-revalidate\r\n
    P3P: CP="IDC DSP COR ADM DEVI TAI1 PSA PSD IVA1 CON1 HIS OUR IND CNT"\r\n
    [truncated] Location: https://192.168.182.1/intercept.php?res=notyet&uamip=192.168.182.1&uamport=3990&challenge=6595f6
    Content-Type: text/html; charset=UTF-8\r\n
```

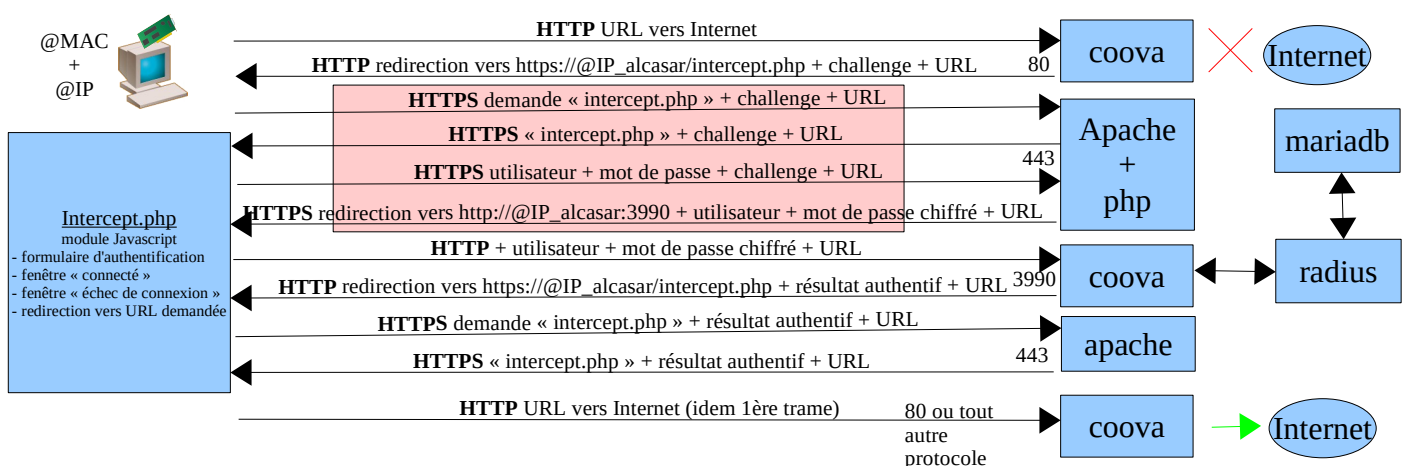
- [trame 1] La requête DNS de l'équipement est récupérée par le serveur DNS d'ALCASAR (dnsmasq).

1 - Les périphériques « Tap » et « Tun » des noyaux Linux sont des interfaces réseau virtuelles de niveau 2 (c.-à-d. Ethernet) pour « Tap » ou 3 (c.-à-d. IP) pour « Tun » permettant à des processus exécutés en espace utilisateur (les interfaces physiques fonctionnent en espace noyau) d'envoyer ou de recevoir des trames sur ces interfaces via les fichiers spéciaux (/dev/tapX ou /dev/tunX). Ces interfaces virtuelles peuvent être exploitées comme des interfaces physiques (configuration, émission/réception, routage). Ces interfaces autorisent un traitement sur les trames à la réception ou avant l'émission de celles-ci. L'interface Tap est souvent utilisée dans la création de tunnels RVP/VPN afin d'encapsuler un flux dans un autre (cf. projet « OpenVPN »).

Les tentatives de connexion vers d'autres serveurs DNS sont bloquées par le pare-feu interne. Cela permet de prévenir le contournement du DNS d'ALCASAR ainsi que les tunnels DNS.

- DNSMasq résout le domaine localement s'il est dans sa base (cf. fonctions de filtrage), sinon il transfère la requête vers les serveurs DNS Internet définis lors de l'installation d'ALCASAR. Les réponses sont retournées à l'équipement de consultation [trame 2].
- Une requête de connexion sur le port 80 (http) du serveur WEB est alors envoyée [trame 3] par la station de consultation. Cette requête est interceptée par « chilli » qui vérifie si un usager n'est pas déjà « autorisé » sur cet équipement :
 - Si tel est le cas, Chilli « ouvre la barrière » et laisse transiter toutes les trames de l'équipement quelque soit le protocole. Le pare-feu prend alors le relais. Il oriente les flux WEB vers la chaîne de filtrage WEB. Il filtre ou transfère les autres flux vers Internet (cf. fonction de filtrage).
 - Si tel n'est pas le cas, Chilli simule une connexion WEB standard [trames 4 à 6] et répond à la requête de l'équipement [trames 7 à 9] par une trame HTTP de redirection de service (« HTTP/1.0 302 Moved Temporarily ») contenant l'URL d'une « splash-page » avertissant de la redirection (directive « uamhomepage » du fichier `/etc/chilli.conf`). Dans ALCASAR cette « splash-page » a été supprimée afin de récupérer directement la page d'authentification définie par la primitive « uamserver » (URL de redirection : « `https://alcasar/intercept.php` ») [cf. détail de la trame 9]. Cette session se termine [trames 10 à 13] et le navigateur initie une session chiffrée avec le serveur WEB intégré dans ALCASAR (Lighttpd) afin de récupérer cette page [trame 14 et suivantes]. L'utilisateur renseigne les champs d'authentification (identifiant + mot de passe) qui sont envoyés de manière chiffrée à Lighttpd pour être traités (chiffrement du mot de passe avec une clé secrète partagée entre Lighttpd et chilli). Lighttpd retourne le résultat au navigateur afin que ce dernier redirige une nouvelle fois ces informations au processus « chilli » (port 3990²). Chilli les récupère afin de pouvoir requêter le serveur radius. Le résultat de cette requête est retourné au navigateur afin d'être traité par les scripts javascript de la page « intercept.php » (échec ou réussite de la connexion).
 - La communication entre chilli et Freeradius exploite le protocole « radius ». Les paramètres de cette communication sont définis à la fois dans le fichier « `/etc/raddb/client.conf` » et via les directives « `hs_radius` », « `hs_radius2` » et « `hs_radsecret` » du fichier « `/etc/chilli.conf` ».
- Pour la déconnexion, les navigateurs Web génèrent une requête adéquate sur le port d'écoute de Chilli (3990).

Cette phase d'interception peut être schématisée comme suit pour un usager non authentifié sur une station de consultation identifiée par son @MAC et son @IP :



2 - « chilli » écoute sur un port défini par la primitive « uamport » du fichier `/etc/chilli.conf` (3990 par défaut). Le format des requêtes envoyées sur ce port détermine l'action demandée (ex. « `@IP:3990/prelogin` » pour une demande de connexion, « `@IP:3990/logout` » pour une demande de déconnexion. La requête contient bien entendu l'ensemble des paramètres nécessaires au traitement de la demande (@MAC, challenge, identifiant, etc.).

4.1.2 - Exception à l'authentification

Coova-chilli à la possibilité de laisser transiter des trames spécifiques vers Internet sans authentification préalable. Cette possibilité est exploitée dans ALCASAR pour permettre la mise à jour automatique des antivirus et des patchs système. Les paramètres « uamallowed » et « uamdomain » pointent vers deux fichiers contenant la liste des adresses IP (ou adresse de réseaux) ou des noms de domaine joignables sans authentification (*/usr/local/etc/alcasar-uamallow* et */usr/local/etc/alcasar-uamdomain*).

4.2 - Base de données des usagers

La base de données des usagers est gérée par le SGBD « MariaDB ». Le schéma de cette base est entièrement compatible avec le service d'authentification Radius. La structure de cette base est mise en place lors de l'installation d'ALCASAR en exploitant un script SQL (cf. fonction « init_db » du script *alcasar.sh*) :

```
# Ajout d'une base vierge
mysql -u$DB_USER -p$radiuspwd $DB_RADIUS < $DIR_CONF/radiusd-db-vierge.sql
```

Le Modèle Conceptuel de Données (MCD) de cette base est le suivant :

BASE DE DONNÉES RADIUS V3.x
(gestion des utilisateurs)

Légende

En rouge : Clé Primaire. AI : Auto-Increment
* : Clé secondaire

- Tables normalisées Radius
- Tables complémentaires ALCASAR
- Tables inexploitées actuellement

1. badusers

id int(10)
UserName * varchar(30)
Date * datetime
 Reason varchar(200)
 Admin varchar(30)

2. mtotacct

mtotacctid bigint(21) - AI
username * varchar(64)
 acctdate * date
 connnum bigint(12)
 conntotduration bigint(12)
 connmaxduration bigint(12)
 connminduration bigint(12)
 inputoctets bigint(12)
 outputoctets bigint(12)
nasipaddress * varchar(15)

10. totacct

totacctid bigint(21) - AI
username * varchar(64)
 acctdate * date
 connnum bigint(12)
 conntotduration bigint(12)
 connmaxduration bigint(12)
 connminduration bigint(12)
 inputoctets bigint(12)
 outputoctets bigint(12)
nasipaddress * varchar(15)

12. userinfo

id int(10) - AI
username * varchar(64)
 name varchar(200)
 mail varchar(200)
department * varchar(200)
 workphone varchar(200)
 homphone varchar(200)
 mobile varchar(200)

4. radacct

radacctid bigint(21) - AI
 acctsessionid * varchar(32)
 acctuniqueid * varchar(64)
username * varchar(64)
 groupname varchar(64)
 realm varchar(64)
nasipaddress * varchar(15)
 nasportid varchar(15)
 nasporttype varchar(32)
 acctstarttime * datetime
 acctstoptime * datetime
 Acctsessiontime * int(12)
 acctauthentic varchar(32)
 connectinfo_start varchar(50)
 connectinfo_stop varchar(50)
 acctinputoctets bigint(20)
 acctoutputoctets bigint(20)
 calledstationid varchar(50)
 acctterminatecause varchar(32)
 servicetype varchar(32)
 framedprotocol varchar(32)
 framedipaddress * varchar(15)
 acctstartdelay int(12)
 acctstspdelay int(12)
 xascendsessionsvrkey varchar(10)

8. radpostauth

id int(11) - AI
 user varchar(64)
 pass varchar(64)
 reply varchar(32)
 date timestamp(14)

5. radcheck

id int(11) - AI
username * varchar(64)
 attribute varchar(64)
 op char(2)
 value varchar(253)

9. radreply

id int(11) - AI
username * varchar(64)
 attribute varchar(64)
 op char(2)
 value varchar(253)

11. radusergroup

username * varchar(64)
 groupname varchar(64)
 priority int(11)

7. radgroupreply

id int(11) - AI
groupname * varchar(64)
 attribute varchar(64)
 op char(2)
 value varchar(253)

6. radgroupcheck

id int(11) - AI
groupname * varchar(64)
 attribute varchar(64)
 op char(2)
 value varchar(253)

3. nas

id int(10)
nasname * varchar(128)
 shortname varchar(32)
 type varchar(30)
 ports int(5)
 secret varchar(60)
 community varchar(50)
 description varchar(200)

* dans les version < 2.0 : la table « radusergroup » s'appelait « usergroup » et le champs « groupname » de la table « radacct » n'existait pas
 * à partir de la version 2.6, le champs 'username' de la table 'userinfo' change de type pour être compatible avec les autres tables (basculé de varchar(30) en varchar(64))

BASE DE DONNÉES GAMMU V1.39
(gestion des SMS)
Version du schéma : 17

inbox	
UpdateInDB	timestamp
ReceivingDateTime	timestamp
Text	text
SenderNumber	° varchar(20)
Coding	enum('Default_No_Compression', 'Unicode_No_Compression', '8bit', 'Default_Compression', 'Unicode_Compression')
UDH	text
SMSCNumber	varchar(20)
Class	int
TextDecoded	° text
ID	° int
RecipientID	text
Processed	enum('false', 'true')
Status	int

SMS_ban_perm	
SenderNumber	° varchar(20)
Expiration	° varchar(255)
Perm	° int
date_add	° timestamp

SMS_ban_temp	
ID	° int
SenderNumber	° varchar(20)

SMS_country	
name	varchar(50)
id	varchar(20)
status	int

gammu	
Version	int

phones	
ID	text
UpdatedInDB	timestamp
InsertIntoDB	timestamp
TimeOut	timestamp
Send	enum('yes','no')
Receive	enum('yes','no')
IMEI	varchar(35)
IMSI	varchar(35)
NetCode	varchar(10)
NetName	varchar(35)
Client	text
Battery	int(11)
Signal	int(11)
Sent	int(11)
Received	int(11)

sentitems	
UpdatedInDB	timestamp
InsertIntoDB	timestamp
SendingDateTime	timestamp
DeliveryDateTime	timestamp
Text	text
DestinationNumber	varchar(20)
Coding	enum('Default_No_Compression', 'Unicode_No_Compression', '8bit', 'Default_Compression', 'Unicode_Compression')
UDH	text
SMSCNumber	varchar(20)
Class	int(11)
TextDecoded	text
ID	unsigned int
SenderID	varchar(255)
SequencePosition	int
Status	enum('SendingOK', 'SendingOKNoReport', 'SendingError', 'DeliveryOK', 'DeliveryFailed', 'DeliveryPending', 'DeliveryUnknown', 'Error')
StatusError	int
TPMR	int
RelativeValidity	int
CreatorID	text
StatusCode	int

outbox_multipart	
Text	text
Coding	enum('Default_No_Compression', 'Unicode_No_Compression', '8bit', 'Default_Compression', 'Unicode_Compression')
UDH	text
Class	int
TextDecoded	text
ID	unsigned int
SequencePosition	int
Status	enum('SendingOK', 'SendingOKNoReport', 'SendingError', 'DeliveryOK', 'DeliveryFailed', 'DeliveryPending', 'DeliveryUnknown', 'Error', 'Reserved')
StatusCode	int

outbox	
UpdateInDB	timestamp
InsertIntoDB	timestamp
SendingDateTime	timestamp
SendBefore	time
SendAfter	time
Text	text
DestinationNumber	varchar(20)
Coding	enum('Default_No_Compression', 'Unicode_No_Compression', '8bit', 'Default_Compression', 'Unicode_Compression')
UDH	text
Class	int
TextDecoded	text
ID	unsigned int
MultiPart	enum('false', 'true')
RelativeValidity	int
SenderID	varchar(255)
SendingTimeOut	timestamp
DeliveryReport	enum('default', 'yes', 'no')
CreatorID	text
Retries	int(3)
Priority	int
Status	enum('SendingOK', 'SendingOKNoReport', 'SendingError', 'DeliveryOK', 'DeliveryFailed', 'DeliveryPending', 'DeliveryUnknown', 'Error', 'Reserved')
StatusCode	int

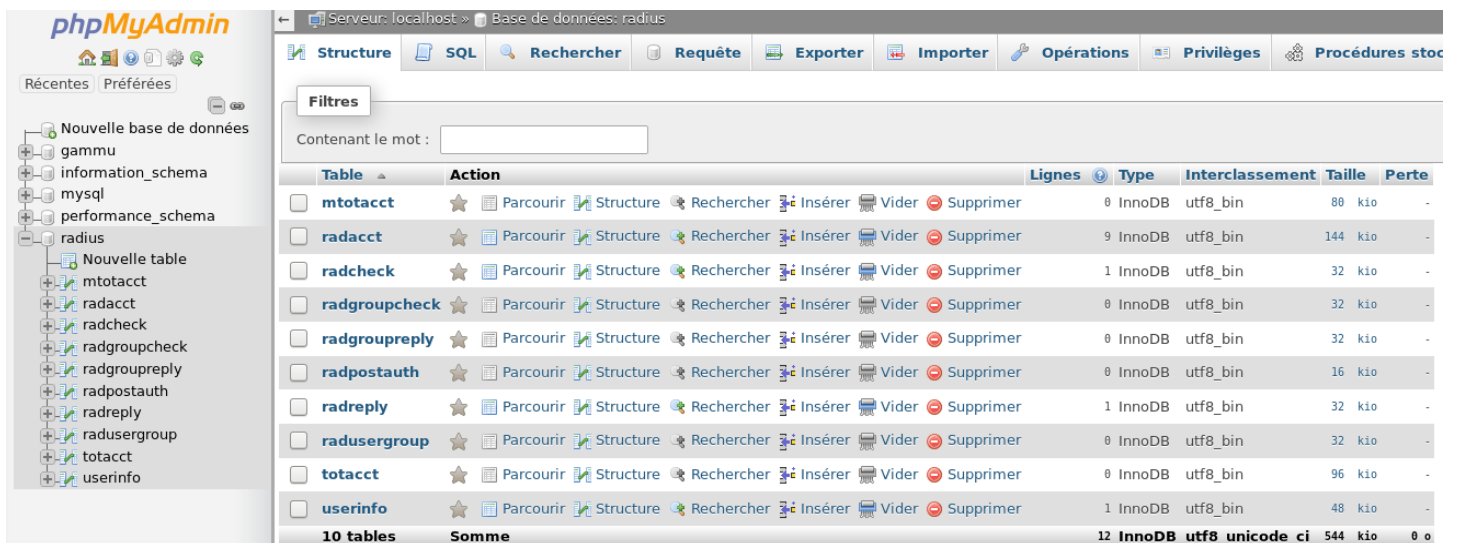
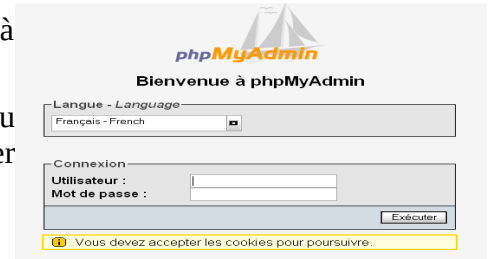
Légende

- En rouge : Index – Clé Primaire
- Table créée par Gammu-smsd
- Table ajoutée pour gérer les SMS
- Table inexploitée par ALCASAR
- ° : Colonne exploitée

4.2.1 - Accès en mode graphique

Il est possible d'afficher de manière graphique et pédagogique le contenu des bases exploitées pas ALCASAR. Voici la procédure exploitant l'interface WEB « phpmyadmin » (à supprimer après avoir été exploité).

- Récupérez la dernière archive de « phpmyadmin » et copiez-la dans le répertoire /var/www/html/. Installez « unzip » (urpmi « unzip ») et décompressez « phpmyadmin » (unzip nom_de_l'archive). Renommez le nom du répertoire de l'archive en « phpmyadmin » (mv nom_répertoire phpmyadmin) ;
- connectez-vous à la base à partir de votre station de consultation à l'URL : « https://alcasar.localdomain/phpmyadmin » ;
- récupérez les mots de passe du compte d'administration (root) ou du compte gestionnaire de la base « radius » dans le fichier « /root/ALCASAR-passwords.txt » ;
- identifiez-vous sur le SGBD soit en « root » soit en « radius » ;
- Vous pouvez maintenant accéder aux bases (« gammu » ou « radius ») ainsi qu'aux contenus des tables.



Quand vous avez terminé avec « phpmyadmin », désinstallez-le (rm -rf phpmyadmin »).

4.2.2 - Accès en mode console

Utilisez les login/mot_de_passe de votre fichier « /root/ALCASAR-password.txt »

Accès à l'administration globale

```
mysql -uroot -p
```

Entrez le mot de passe associé à l'administrateur (root)

ex : Voir les bases de données : « SHOW DATABASES ; »

« exit » pour quitter ; « help » pour voir les autres commandes.

```
MariaDB [(none)]> SHOW DATABASES ;
+-----+
| Database |
+-----+
| gammu    |
| information_schema |
| mysql    |
| performance_schema |
| radius   |
+-----+
5 rows in set (0.01 sec)
```

Accès à la base « radius »

```
mysql -uradius -p radius
```

Entrez le mot de passe associé à l'utilisateur « radius ».

ex : Voir les tables : « SHOW TABLES ; »

ex : Voir le contenu : « SELECT * FROM <tables_name> ; »

```
Tables_in_radius
+-----+
| mtotacct |
| radacct  |
| radcheck |
| radgroupcheck |
| radgroupreply |
| radpostauth |
| radreply |
| radusergroup |
| totacct  |
| userinfo |
+-----+
10 rows in set (0.01 sec)
```

4.3 - Le serveur Radius

Le service « freeradius » (radiusd) est utilisé comme unité d'authentification, d'autorisation et de journalisation (Authentication Authorization Accounting).

L'authentification exploite par défaut la base de données locale (mariadb). Un module LDAP additionnel a été intégré afin de pouvoir valider le couple login/MDP avec celui d'un annuaire externe compatible LDAP (Active Directory®, OpenLDAP, etc.).

L'autorisation et la journalisation exploitent uniquement la base de données local.

Les fichiers de configuration du serveur sont dans le répertoire « /etc/raddb ». La configuration de ce serveur est définie dans le fichier « radiusd.conf ». Le fichier « client.conf » contient les paramètres du seul client radius configuré par défaut (coovachilli). Deux fichiers de configuration spécifiques pour ALCASAR sont situés dans le répertoire « sites-available » (« alcasar » et « alcasar-with-ldap »). En fonction de la configuration, l'un ou l'autre est lié symboliquement au fichier « alcasar » situé dans le répertoire « sites-enabled ». Les modules radius (rlm : Radius Loadable Modules) exploités par ALCASAR sont situés dans le répertoire « mods-enabled ». Ce sont des liens symboliques pointant vers des fichiers du répertoire « mods-available ». Les modules suivants sont exploités pour ALCASAR :

- « sql » : définit les paramètres de connexion à la base de données des utilisateurs (mariadb) ;
- « pap » : permet de valider le mot de passe chiffré de l'utilisateur (Password Authentication Protocol) ;
- « attr_filter » : permet de filtrer (mise en forme) certains attributs avant de les traiter ;
- « expr » : utiliser pour effectuer des calculs numériques (*, /, +, sha256, etc.) et pour convertir des caractères (ex : « toupper » et « tolower ») ;
- « expiration » : permet de gérer la date d'expiration d'un compte (attribut : « Expiration ») ;
- « logintime » : permet de gérer le « temps de session max » (attribut : « Session-Timeout ») ;
- « sqlcounter » : permet de gérer les notions de temps maximum de connexion par jour (dailycounter = attribut « Max-Daily-Session), par mois (monthlycounter = attribut « Max-Monthly-Session ») et la notion de « temps autorisé après le premier login » (noresetcounter = attribut « Max-All-Session ») ;
- « ldap » : permet la connexion vers un annuaire externe (chargé uniquement si configuré)

Il est possible de tester le serveur radius en ligne de commande : `radtest <userLogin> <userPassword> 127.0.0.1 0 <secret_radius>` Rappel : « secret_radius » est le secret partagé entre le client radius (« coovachilli » dans le cas d'ALCASAR) et le serveur. Ce mot de passe est dans « /root/ALCASAR-passwords.txt »

```
Sent Access-Request Id 130 from 0.0.0.0:40775 to 127.0.0.1:1812 length 74
  User-Name = "test"
  User-Password = ██████████
  NAS-IP-Address = 172.16.0.1
  NAS-Port = 0
  Message-Authenticator = 0x00
  Cleartext-Password = ██████████
Received Access-Accept Id 130 from 127.0.0.1:1812 to 0.0.0.0:0 length 36
  Filter-Id = '00000000'
  Session-Timeout = 7200
```

Il est possible d'analyser le fonctionnement du serveur radius. Pour cela, arrêtez le DAEMON (`systemctl stop radiusd`) et relancez-le en mode « debug » (`radiusd -X`). Vous pouvez alors analyser les requêtes SQL lorsque des utilisateurs se connectent, se déconnectent, échouent à la connexion, etc.

Compteurs SQL exploités par ALCASAR

Déclaration des compteurs (section « authorize » de « /etc/raddb/sites-enabled/alcasar »)

```
authorize {
  sql
  noresetcounter
  dailycounter
  monthlycounter
  expiration
  logintime
```

```
} pap
```

Définition des compteurs SQL actuellement exploités (« /etc/raddb/mods-enabled/sqlcounter ») :

```
sqlcounter dailycounter {  
    sql_module_instance = sql  
    counter_name = Daily-Session-Time  
    check_name = Max-Daily-Session  
    reply_name = Session-Timeout  
    key = User-Name  
    reset = daily  
    $INCLUDE ${modconfdir}/sql/counter/mysql/${.:instance}.conf  
}
```

```
sqlcounter monthlycounter {  
    sql_module_instance = sql  
    counter_name = Monthly-Session-Time  
    check_name = Max-Monthly-Session  
    reply_name = Session-Timeout  
    key = User-Name  
    reset = monthly  
    $INCLUDE ${modconfdir}/sql/counter/mysql/${.:instance}.conf  
}
```

```
sqlcounter noresetcounter {  
    sql_module_instance = sql  
    counter_name = Max-All-Session-Time  
    check_name = Max-All-Session  
    key = User-Name  
    reset = never  
    $INCLUDE ${modconfdir}/sql/counter/mysql/${.:instance}.conf  
}
```

```
sqlcounter expire_on_login {  
    sql_module_instance = sql  
    counter_name = Alcasar-Expire-After-Initial-Login  
    check_name = Alcasar-Expire-After  
    key = User-Name  
    reset = never  
    $INCLUDE ${modconfdir}/sql/counter/mysql/${.:instance}.conf  
}
```

```
sqlcounter counterCoovaChilliMaxTotalOctetsDaily {  
    sql_module_instance = sql  
    counter_name = CoovaChilli-Max-Total-Octets-Daily  
    check_name = CoovaChilli-Max-Total-Octets-Daily  
    counter_type = data  
    reply_name = CoovaChilli-Max-Total-Octets  
    sqlmod-inst = sql  
    key = User-Name  
    reset = daily  
    query = "SELECT IFNULL((SUM(AcctInputOctets + AcctOutputOctets)),0) FROM radacct WHERE username='%${  
{key}}' AND UNIX_TIMESTAMP(AcctStartTime) + AcctSessionTime > '%%b'"  
}
```

```
sqlcounter counterCoovaChilliMaxTotalOctetsMonthly {  
    sql_module_instance = sql  
    counter_name = CoovaChilli-Max-Total-Octets-Monthly  
    check_name = CoovaChilli-Max-Total-Octets-Monthly  
    counter_type = data  
    reply_name = CoovaChilli-Max-Total-Octets  
    sqlmod-inst = sql  
    key = User-Name  
    reset = monthly  
    query = "SELECT IFNULL((SUM(AcctInputOctets + AcctOutputOctets)),0)FROM radacct WHERE username='%${  
{key}}' AND UNIX_TIMESTAMP(AcctStartTime) + AcctSessionTime > '%%b'"  
}
```


4.3.1 - Module A.D./LDAP externe

Freeradius peut interroger une base externe via le protocole LDAP. Les paramètres de connexion sont renseignés dans le fichier de configuration central d'ALCASAR (/usr/local/etc/alcasar.conf). Le script « alcasar-ldap.sh » se charge de configurer les modules LDAP de freeradius (« /etc/raddb/mods-available/ldap » et « /etc/raddb/sites-enabled/alcasar-with-ldap »).

Paramètres	Définition	Remarques
server	Nom du serveur LDAP (server = "ldap.example.com" ou server = "@IP")	Le port de connexion par défaut est 389. Pour le changer : @serveur:port
basedn	Base de recherche des usagers à authentifier	Voir l'exemple ci-dessous dans le cas d'Active Directory.
filter	Recherche de l'identifiant ou attribut pour l'authentification	<u>Pour un ldap standard</u> : filter = "(uid=%{Stripped-User-Name}:-%{User-Name})" <u>Pour Active Directory</u> : filter = "(samAccountName=%{Stripped-User-Name}:-%{User-Name})"
base_filter	Filtre de recherche ldap complémentaire	Exemples : - par défaut, vide - base_filter="(objectclass=radiusprofile)" - base_filter="(memberof=groupe_alcasar)"
identity	Compte possédant des droits en lecture sur l'annuaire.	Vide = connexion anonyme (LDAP) <u>Obligatoire pour Active Directory</u> (sur le serveur AD, créer un compte standard qui sera utilisé par ALCASAR pour l'interroger l'annuaire à distance).
password	Mot de passe associé au compte avec des droits de lecture sur l'annuaire ldap.	Vide = connexion anonyme (LDAP). <u>Obligatoire pour Active Directory.</u>

Par rapport à l'exemple d'annuaire présenté dans le document d'exploitation, les paramètres de ce fichier seraient les suivants :

```
basedn = "cn=Users,dc=serverad,dc=com"
filter = "(samAccountName=%{Stripped-User-Name}:-%{User-Name})"
identity= "cn=alcasar,dc=serverad,dc=com"
password = "*****"
```

Il est possible d'analyser la connexion avec le serveur d'annuaire externe à partir du poste ALCASAR après avoir installé le paquetage « openldap-clients » ([urpmi openldap-clients](#)).

La commande « ldapsearch -vWx -h @ip_A.D -b "cn=Users ;dc=serverad,dc=com" -D "alcasar@serverad.com" » permet de lister l'ensemble des usagers contenu dans « User ».

Les options utilisées dans cette commande sont les suivantes : -v : verbeux, -b : la base recherchée, -D : le dn de l'utilisateur autorisé à lancer une requête sur la base, -W : demande le mot de passe de manière interactive, -x : exploite l'authentification simple plutôt que SASL.

Sur ALCASAR, les usagers authentifiés de cette manière sont affectés dans le groupe d'utilisateurs nommé « ldap ». Il est ainsi possible de leur affecter des attributs particuliers. Pour modifier ce nom de groupe, il suffit d'éditer le fichier « /etc/raddb/sql/mysql/dialup.conf » et modifier la valeur « default_user_profile = "<nouveau groupe>" ». Relancer le service « radiusd » pour la prise en compte de la modification.

Il est aussi possible d'affecter des attributs ALCASAR à un seul usager authentifié A.D./LDAP. Pour cela il suffit de créer un usager ALCASAR ayant le même nom que celui qui est dans l'annuaire externe. Exemple d'utilisation : tous les élèves d'une école sont gérés dans un annuaire. Il est possible de limiter la bande passante ou les créneaux de connexion pour l'élève ayant abusé de téléchargements. Il suffit de créer un compte sur ALCASAR avec le nom de cet élève et de lui affecter des attributs particuliers.

5 - Fonction « traçabilité et imputabilité »

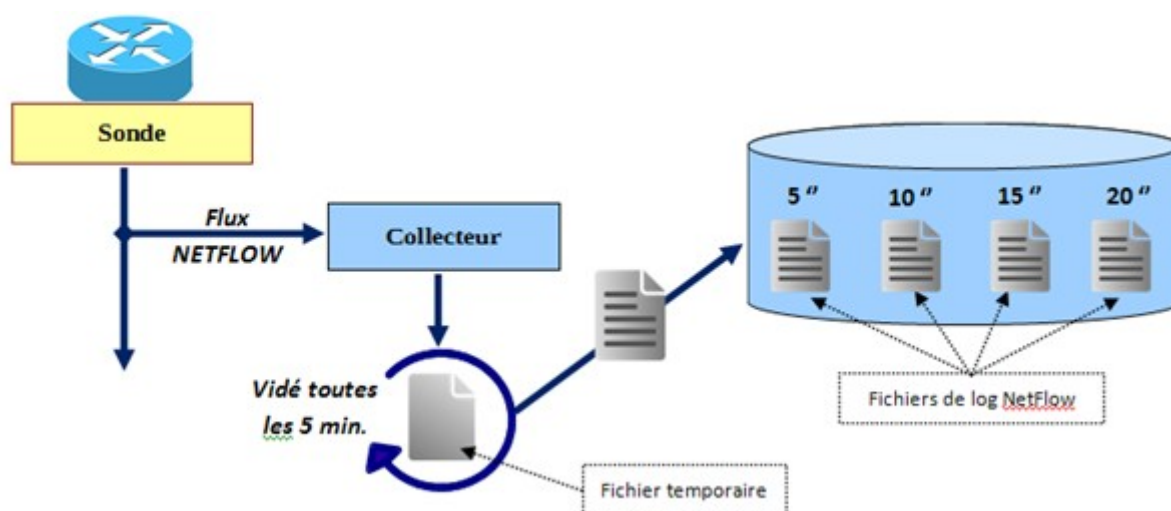
Plusieurs systèmes de traçabilité-imputabilité ont été évalués afin de définir celui qui serait exploité dans ALCASAR. Le tableau suivant résume le résultat de cette évaluation :

	NAGIOS	MRTG	CACTI	MUNIN	NETFLOW
Licence GPL					
Compatible <u>Mageia</u>					
Charge supplémentaire sur le système					
Espace de stockage nécessaire					
Facilité d'utilisation					
Interface d'administration					
Représentation des données sous forme de graphes					
Accessibilité aux données antérieures					
Contenu des fichiers journaux pour l'imputabilité					
Affichage de la charge par ports					

Satisfait : Convenable : Non satisfait :

5.1 - Journalisation principale

La traçabilité des connexions est principalement assurée par une sonde NetFlow qui est spécifiquement intégrée au noyau Linux exploité par ALCASAR. Le schéma de fonctionnement de cette sonde est le suivant :



Les flux NetFlow sont générés par la sonde via des règles du parefeu Netfilter (cible « -j Netflow »). Ces flux sont envoyés sur 127.0.0.1:2055. Ils sont récupérés par le collecteur nfcapd qui génère un fichier toutes les 5' dans le répertoire « `/var/log/nfsen/profile_data/live/alcasar_netflow/date_du_jour/` ».

Deux canaux distincts permettent de récupérer toutes les traces de connexion.

1. Le premier canal contient la trace de tous les flux des utilisateurs à l'exception des flux HTTP des utilisateurs avec attribut de filtrage (cf. 2ème canal). Ces traces sont générées par le module « netflow » du pare-feu. Ce canal génère un fichier toutes les 5' dans « `/var/log/nfsen/profile_data/live/alcasar_netflow/` » (un répertoire par jour). Chaque semaine, une archive est constituée sous le nom « `traceability-ALL-<date><heure>.tar.gz` » (cf. §5.3). Comme ces fichiers ne sont pas directement lisibles, il faut exploiter un **interpréteur Netflow** (ex. : « Nfdump ») comme suit : « `nfdump -R <fichier_au_format_netflow> -o extended -a` ». Une fois interprétée, chaque ligne est composée des champs suivants :

•	Date first seen	Duration	Proto	Src IP Addr:Port	Dst IP Addr:Port	Flags	Tos	Packets	Bytes	pps	bps	Bpp	Flows
•	2020-05-19 23:03:02.978	0.000	TCP	192.168.182.24:53955	→ 216.239.38.120:443S.	0	1	60	0	0	60	1
•	2020-05-19 23:15:10.898	133.038	TCP	192.168.182.24:49570	→ 216.239.38.120:5228S.	0	5	300	0	18	60	3
•	2020-07-04 14:45:29.870	0.000	TCP	192.168.182.7:48399	→ 104.85.39.74:80S.	0	1	60	0	0	60	1

2. Le deuxième canal contient les traces des flux HTTP des utilisateurs possédant un attribut de filtrage (antivirus et/ou blacklist et/ou whitelist). Ces traces sont générées par une règle « Ulog » du pare-feu sur le flux transitant dans le système de filtrage D’ALCASAR (E²Guardian + havp). Ces traces sont écrites dans le fichier « `/var/log/firewall/traceability.log` ». Chaque semaine, ce fichier est copié dans l’archive de traçabilité sous le nom « `traceability-HTTP-<date><heure>.tar.gz` » (cf. §5.3) avant d’être purgé. Dans ce fichier chaque ligne est composée des champs principaux suivants :

Oct 28 03:44:06 alcasar RULE_F_http ACCEPT IN=tun0 OUT= MAC= SRC=192.168.182.2 DST=77.67.27.11 LEN=52 TOS=00 PREC=0x00 TTL=128 ID=19347 DF PROTO=TCP SPT=62934 DPT=80 SEQ=161741080 ACK=0 WINDOW=8192 SYN URGP=0

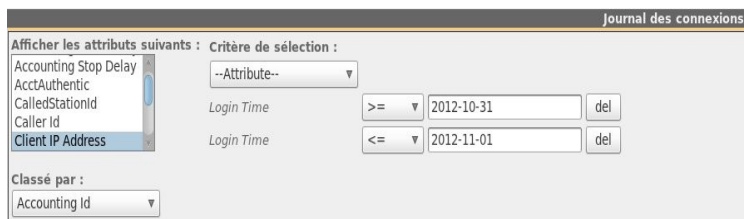
5.2 - journalisation accessoire

- Un canal Ulog génère les fichiers « `/var/log/firewall/ssh.log` » lié aux flux d’administration à distance via le protocole ssh.
- Un canal Ulog génère les fichiers « `/var/log/firewall/ext-access.log` » lié aux tentatives de connexions depuis Internet (fonction « bastion »). Pour ce canal, une protection est mise en place afin de ne pas charger trop le système en cas d’attaque par saturation (flooding).
- Le proxy de filtrage d’URL « E²Gurdian » génère des logs dans le répertoire « `/var/log/e2guardian` » sous le nom : « `access.log` ». Ils présentent les URL ayant été bloquées.
- Le proxy antivirus HAVP génère des logs dans « `/var/log/havp/` » sous le nom « `access.log` ». Ils présentent les virus détectés et bloqués par le couple (HAVP + libclamav).
- Le détecteur d’intrusion (IDS) « fail2ban » génère des log dans « `/var/log/fail2ban` ».

5.3 - Constitution de l’archive de traçabilité

Tous les lundis à 5h35, le gestionnaire de tâche « cron » lance le script « `alcasar-archive.sh` ». Ce script crée un fichier contenant une archive pour chaque canal de journalisation (cf. §5.1) et la base des utilisateurs. Il copie ce fichier sous le nom « `traceability-<date>-<heure>.tar.gz` » dans le répertoire « `/var/Save/log/` » afin d’être visible dans l’interface de gestion (ACC).

Pour imputer chaque trame, il faut extraire à partir du fichier de la base de données « `radius.sql` » (de la même semaine) le nom de l’usager connecté sur la station de consultation possédant l’adresse IP source. Cette dernière information peut être récupérée directement à partir de l’interface graphique d’ALCASAR (menu « statistique » + « connexions »). Exemple pour chercher les usagers connectés dans la journée du 31/10/2012 :



6 - Fonction « filtrage »

6.1 - Filtrage de protocoles réseau

Cette couche est gérée à l'aide du pare-feu intégré (NetFilter).

Le portail est configuré en mode 'bastion' vis-à-vis d'Internet. Il aiguille et contrôle les flux en provenance du réseau de consultation. Lors de l'installation, les règles du pare-feu sont mises en place.

Le fichier de configuration principal qui conditionne le fonctionnement de cova-chilli et des proxy web est « `/usr/local/bin/alcasar-iptables.sh` ». Il est déconseillé de le modifier afin d'éviter des effets de bords sur le fonctionnement global du portail.

Toutefois, certaines règles du pare-feu peuvent être surchargées pour permettre d'accéder à certaines fonctionnalités (accès SSH depuis l'extérieur pour l'administration par exemple).

Pour permettre ces paramétrages 'locaux', le fichier « `/usr/local/etc/alcasar-iptables-local.sh` » est appelé par le fichier de configuration principal du pare-feu. Les lignes pour l'administration externe par SSH sont commentées dans ce fichier pour exemple.

Par défaut, le portail autorise tous les protocoles lorsqu'une session utilisateur est ouverte. Cette fonction 'libertine' peut-être restreinte par une liste blanche de services autorisés. C'est le rôle du fichier « `/usr/local/etc/alcasar-filter-exceptions` » qui est appelé par le script principal du pare-feu si la variable FILTERING est positionnée à « yes ». Cette dernière est modifiable par le biais de l'interface de gestion. Dans ce cas-là, les services listés dans le fichier `alcasar-filter-exception` sont les seuls à être joignables depuis le réseau de consultation. Cette liste n'est pas exhaustive ; elle est modifiable par le biais de l'interface de gestion.

Pour forcer les usagers à passer par le service DNS du portail, le pare-feu effectue une redirection de port 53 vers `l'@IP` locale. Cet artifice permet de couper court aux éventuels tunnels DNS (sur le port 53 uniquement). Remarque : les seuls serveurs DNS interrogés par ALCASAR restent ceux qui ont été renseignés lors de l'installation et qui sont définis dans le fichier `/etc/dnsmasq.conf` (primitive 'server').

6.2 - Filtrage de noms de domaines, d'URLs et d'adresses IP

Ce filtrage s'appuie sur l'excellente liste de l'Université de Toulouse qui est organisée en répertoires. Chaque répertoire porte le nom d'une catégorie (adulte, secte, shopping, etc.). Chaque répertoire peut contenir 1 à 3 fichiers contenant la liste des « noms de domaine », des « URLs » et des « @IP » de cette catégorie. Un quatrième fichier permet de savoir si la catégorie est « noire » (blacklist) ou blanche (whitelist) ou les deux. ALCASAR traite cette liste afin de l'exploiter selon 3 techniques différentes :

1. filtrage de noms de domaine : ALCASAR s'appuie sur ses serveurs de DNS internes (« dnsmasq ») ;
2. filtrage d'URLs : ALCASAR s'appuie sur le proxy HTTP « E²Guardian » ;
3. filtrage d'adresses IP : C'est le pare-feu d'ALCASAR qui traite les @IP de la liste.

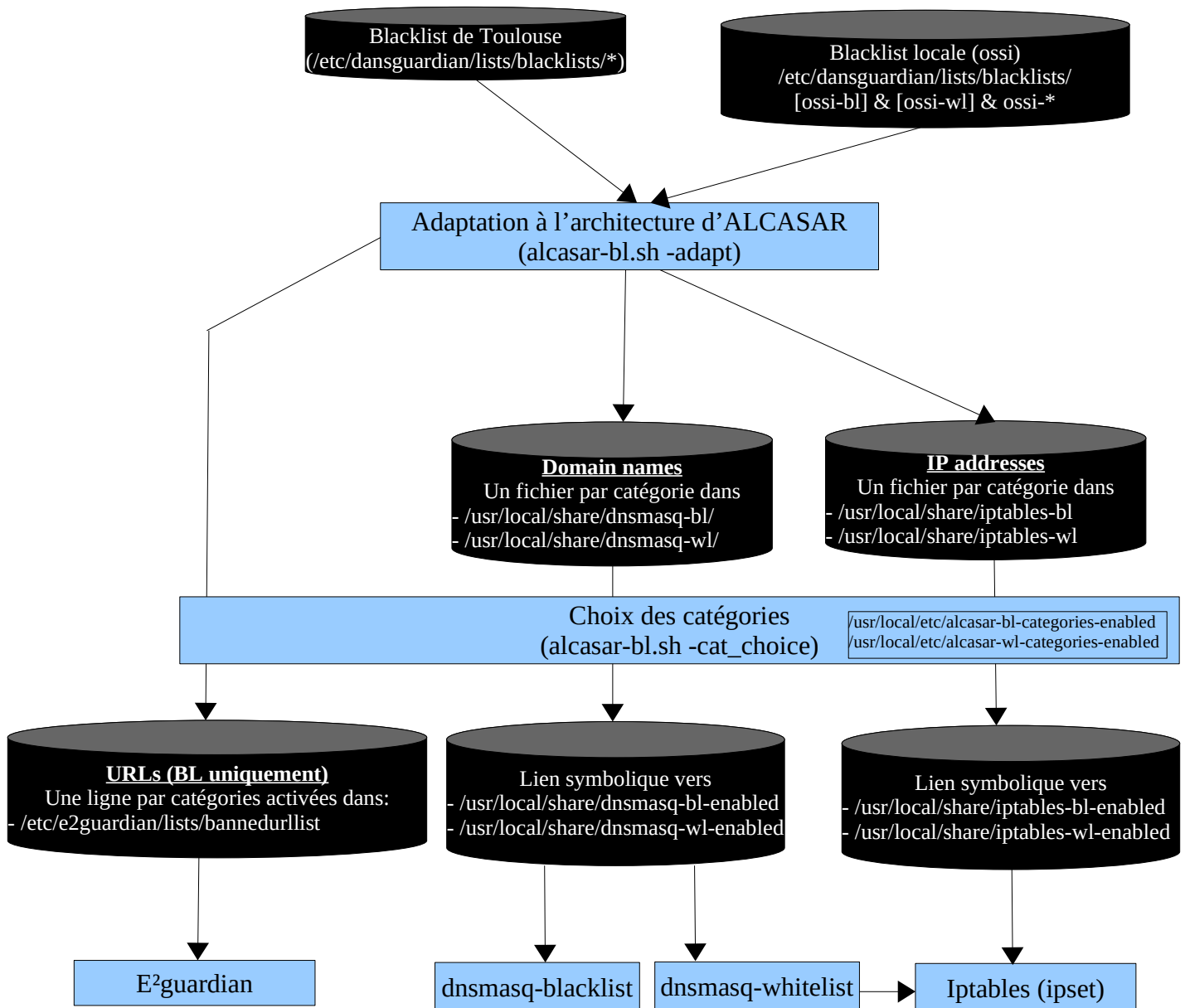
L'architecture d'ALCASAR rend le contournement du filtrage très compliqué. Celui-ci est toujours possible par l'ouverture d'un tunnel (VPN) à destination d'un équipement maîtrisé situé sur Internet dont l'internaute connaît l'adresse IP. Pour fonctionner, ce tunnel doit faire transiter l'ensemble des protocoles de la station de consultation (dont le DNS).

Quoi qu'il en soit, ce type de tunnel ne permet pas de contourner l'authentification. Ainsi, ALCASAR trace et impute les trames de ce tunnel. En cas de problème, et si l'enquête détermine que la sortie du tunnel est impliquée, le portail pourra être sollicité pour déterminer quel utilisateur a créé ce tunnel.

Dans le cas de la WhiteList, ce contournement par VPN n'est pas possible. En effet, ALCASAR tient à jour de manière dynamique les @IP autorisées en fonction des appels DNS réalisés par l'internaute.

6.2.1 - Traitement de la liste de Toulouse

Afin de permettre cette triple exploitation, le script « `alcasar-bl.sh` » effectue le traitement suivant lors de l'installation, de la mise à jour ou du choix des catégories de la Blacklist :



6.2.2 - Filtrage par usager/groupe

Quand un utilisateur réussi sont processus d'authentification, le daemon 'chilli' lance le script `/usr/local/bin/alcasar-conup.sh`. Ce script récupère par variable, l'ensemble des attributs de l'utilisateur. En fonction de l'attribut de filtrage « `Filter_Id` », le script positionne l'adresse IP de l'utilisateur dans l'IPSET correspondant.

Quand l'utilisateur se déconnecte, le daemon 'chilli' lance le script `/usr/local/bin/alcasar-condown.sh` qui retire l'adresse IP de l'utilisateur de l'IPSET correspondant.

Quatre IP_SET ont été créés pour gérer les différentes possibilités de filtrage. Le filtrage affecté à un utilisateur est défini par l'attribut radius « `Filter_Id` ». La liste suivante résume les valeurs des IPSET et de l'attribut « `Filter_Id` » :

- IPSET=« `not_filtered` » pour les usagers sans filtrage. « `Filter_Id` » : `Filter_Id=00000000`
- IPSET=« `havp` » pour les usagers filtrés avec l'antivirus (HAVP+LibClamav). `Filter_Id=00000001`
- IPSET=« `havp_bl` » pour les usagers filtrés avec l'antivirus et par la liste noire. `Filter_Id=00000011`
- IPSET= « `havp_wl` » pour les usagers filtrés avec l'antivirus et la liste blanche. `Filter_Id=00000101`

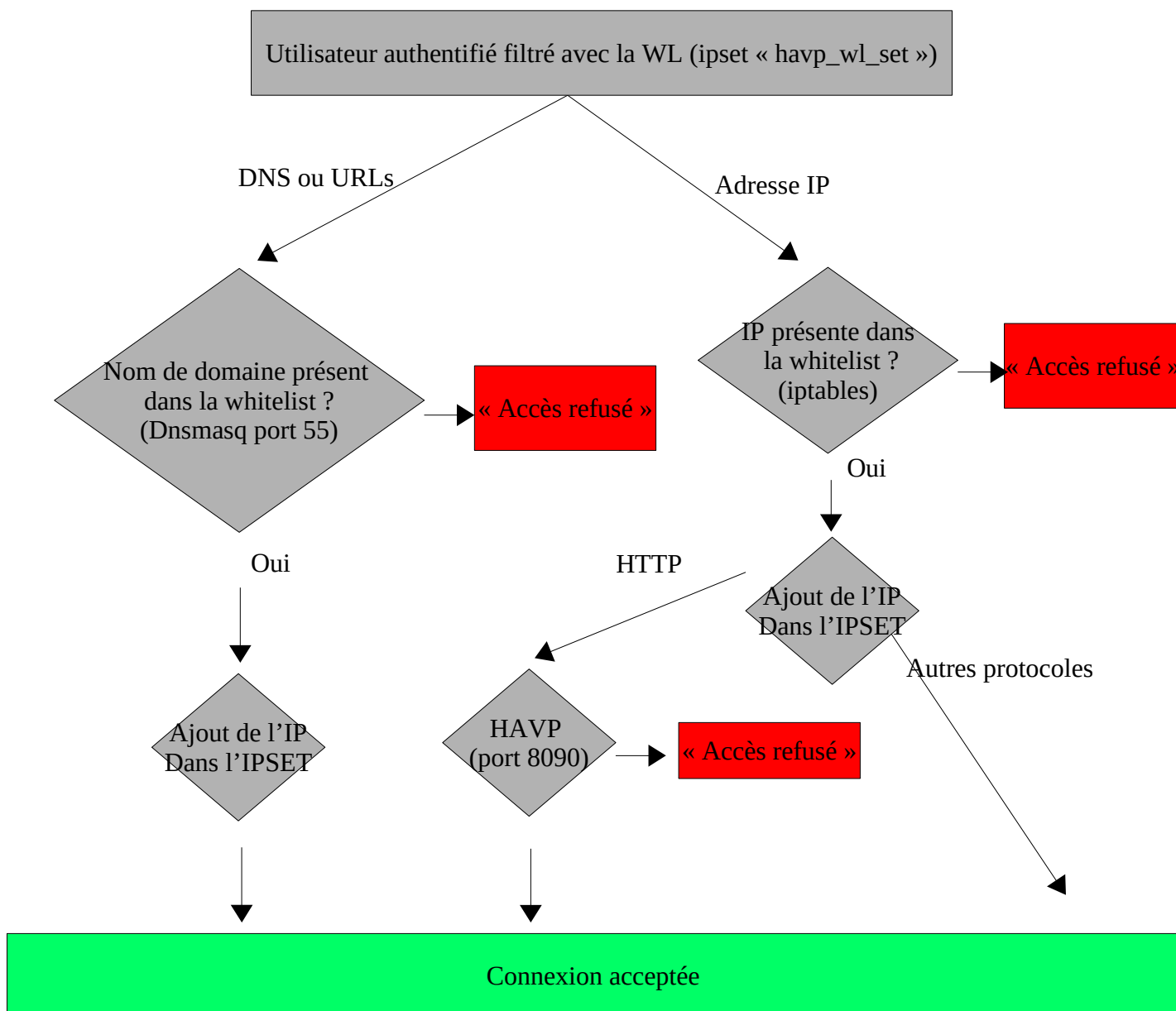
6.2.3 - Double filtrage de la WhiteList (WL)

Un utilisateur filtré avec la WL ne peut consulter que des sites ou des IP définis préalablement. La WL se décompose en plusieurs catégories de noms de domaine pouvant être sélectionnées et mises à jour dans l'ACC. La liste complète des noms de catégorie est située dans « [/usr/local/etc/alcasar-wl-categories](#) ». La liste des seules catégories activées est située dans « [usr/local/etc/alcasar-wl-categories-enabled](#) ». Ce sont des liens symboliques pointant vers la liste complète.

Les adresses IP des utilisateurs « whitelists » connectés (authentifiés) sont stockées dans l'IPSET « `havp_wl` ». Lorsqu'un utilisateur se connecte au portail avec cet ipset, ALCASAR va résoudre les requêtes de cet utilisateur via le daemon « `dnsmasq-whitelist` » (port UDP 55). Si le site en question est présent dans liste des sites « [/usr/local/share/dnsmasq-wl/*.conf](#) », le daemon peut résoudre le nom de domaine et récupérer l'@IP du site qui est retourné à l'utilisateur. Cette @IP est ajoutée simultanément à l'IPSET « `whitelist_ip_allowed` » afin d'autoriser l'utilisateur à l'atteindre.

Cette méthode permet d'éviter qu'un utilisateur « whitelisted » puisse contourner le filtrage en se connectant sur un site directement au moyen de son adresse IP (sans résolution de domaine).

Il est possible via l'ACC d'ajouter manuellement de nouveau [site/@IP](#). Les sites seront ajoutés dans le fichier [/usr/local/share/dnsmasq-wl/ossi.conf](#). Un lien symbolique sera alors créé pour que ces ajouts soient pris en compte ici [/usr/local/share/dnsmasq-wl-enabled/ossi](#). Les adresses IP seront ajoutées dans le fichier [/usr/local/share/ossi-ip-wl](#). Le script permettant cet ajout se trouve dans [/usr/local/sbin/alcasar-bl.sh](#). Voici un schéma récapitulatif du fonctionnement du double filtrage de la WL :



6.2.4 - Filtrage avec la BlackList (BL)

En utilisant ses serveurs DNS, ALCASAR va pouvoir déterminer si le site demandé par l'utilisateur est interdit. Si tel est le cas, l'utilisateur sera renvoyé vers [l'@IP](#) du portail (page de filtrage). Ce filtrage offre l'avantage de pouvoir interdire un nom de domaine, quel que soit le protocole demandé (HTTP, HTTPS, FTP, etc.). Ce type de filtrage est souvent appelé « DNS blackhole ».

Tout comme la WL, la BL est organisée en catégories. ALCASAR permet de sélectionner ces catégories via l'interface de gestion (ACC). La liste des noms de catégorie (`/usr/local/etc/alcasar-bl-categories`) et la liste des catégories activées sont situées dans le répertoire de configuration d'ALCASAR (`/usr/local/etc/alcasar-bl-categories-enabled`).

6.3 - Antimalware

L'antimalware « Clamav » (service « clamav-daemon.service ») est couplé au proxy WEB E2guardian afin d'analyser les flux WEB (en clair). C'est le daemon « clamd » qui reçoit les fichiers à analyser via le socket local « `/var/lib/clamav/clamd.socket` ». Les fichiers sont alors copiés dans « `/var/lib/clamav/tmp` » pour être analysés.

La base antivirus est située dans « `var/lib/clamav` ». Elle est mise à jour toutes les 4 heures via le daemon « freshclam » (service « clamav-freshclam.service »).

Un test de fonctionnement de filtrage antimalware peut être effectué via ce site : <http://2016.eicar.org/85-0-Download.html>

7 - Fonction « Interface de gestion »

Cette fonction (ALCASAR Control Center) est réalisée en PHP. Les possibilités de cette interface sont décrites dans la documentation d'exploitation. L'ACC est située dans le répertoire « `/var/www/html/acc` ».

Elle est protégée en accès par le module d'authentification « htdigest » de Lighttpd dont le fichier de configuration est « `/etc/httpd/conf/webapps.d/alcasar.conf` »

Le répertoire « `/usr/local/etc/digest/` » contient les fichiers des identifiants et des mots de passe des administrateurs en fonction de leur profil :

- key_all
- key_admin
- key_manager
- key_backup

8 - Fonction « modules complémentaires »

8.1 - Import de comptes

Dans le cadre de la gestion des comptes d'authentifications, il est possible d'importer une liste de comptes attachés à un groupe prédéfini. Cette fonctionnalité accessible depuis l'interface de gestion génère un fichier `<import-user>.pwd` pour chaque importation et ajoute les usagers dans le groupe (optionnel) de la base de données. Pour l'instant, seul le groupe peut-être attaché aux identifiants ; c'est-à-dire qu'aucun renseignement supplémentaire n'est importable pour le moment.

Le script `import_user.php` du répertoire « `/var/www/html/acc/manager/htdocs` » permet d'importer le fichier au format csv ou txt et le script `import_file.php` permet de ...

L'importation d'un fichier génère un fichier associé comportant les mots de passe en clair des utilisateurs importés. Ce dernier est téléchargeable pour être distribué aux usagers. Afin de les supprimer périodiquement, une tâche, planifiée toutes les 30min, cherche et supprime les fichiers datant de plus de 24h00.

Le script lancé est `alcasar-import-clean.sh`.

8.2 - Auto-inscription par SMS

8.2.1 - Fonctionnement global

L'administrateur peut mettre en place un module d'auto-inscription par SMS. Ce module fonctionne grâce au projet Gammu (plus précisément Gammu_smsd) qui va permettre de stoker en base de données (cf. schéma de cette base en P11) les SMS reçus par un MODEM-GSM connecté sur le port USB (clé-3G). Pour un bon fonctionnement, le code PIN de la carte SIM doit être renseigné dans le fichier de configuration de Gammu_smsd : « `/etc/gammu_smsd_conf` ». Les informations de connexion à la base de données sont écrites lors de l'installation d'ALCASAR.

Fonctionnement :

L'administrateur peut lancer `gammu_smsd` à partir de l'ACC (menu « Auto Enregistrement (SMS) »). Il est possible de suivre le journal d'évènements : « `tailf /var/log/gammu_smsd/gammu_smsd.log` »

À chaque lancement, le script « `/usr/local/bin/alcasar-sms.sh --start` » vérifie que le groupe « sms » est bien créé. Il le crée le cas échéant.

En fonctionnement, gammu_smsd dialogue avec le modem-GSM. Les SMS reçus par le modem-GSM sont alors récupérés puis stockés dans la table « inbox » de la base de données « gammu ».

Sur chaque réception de SMS, gammu_smsd, lance le script « `/usr/local/bin/alcasar-sms.sh --new_sms` ». Ce script permet de traiter le SMS selon le schéma de la page suivante.

Actuellement, l'administrateur à la possibilité :

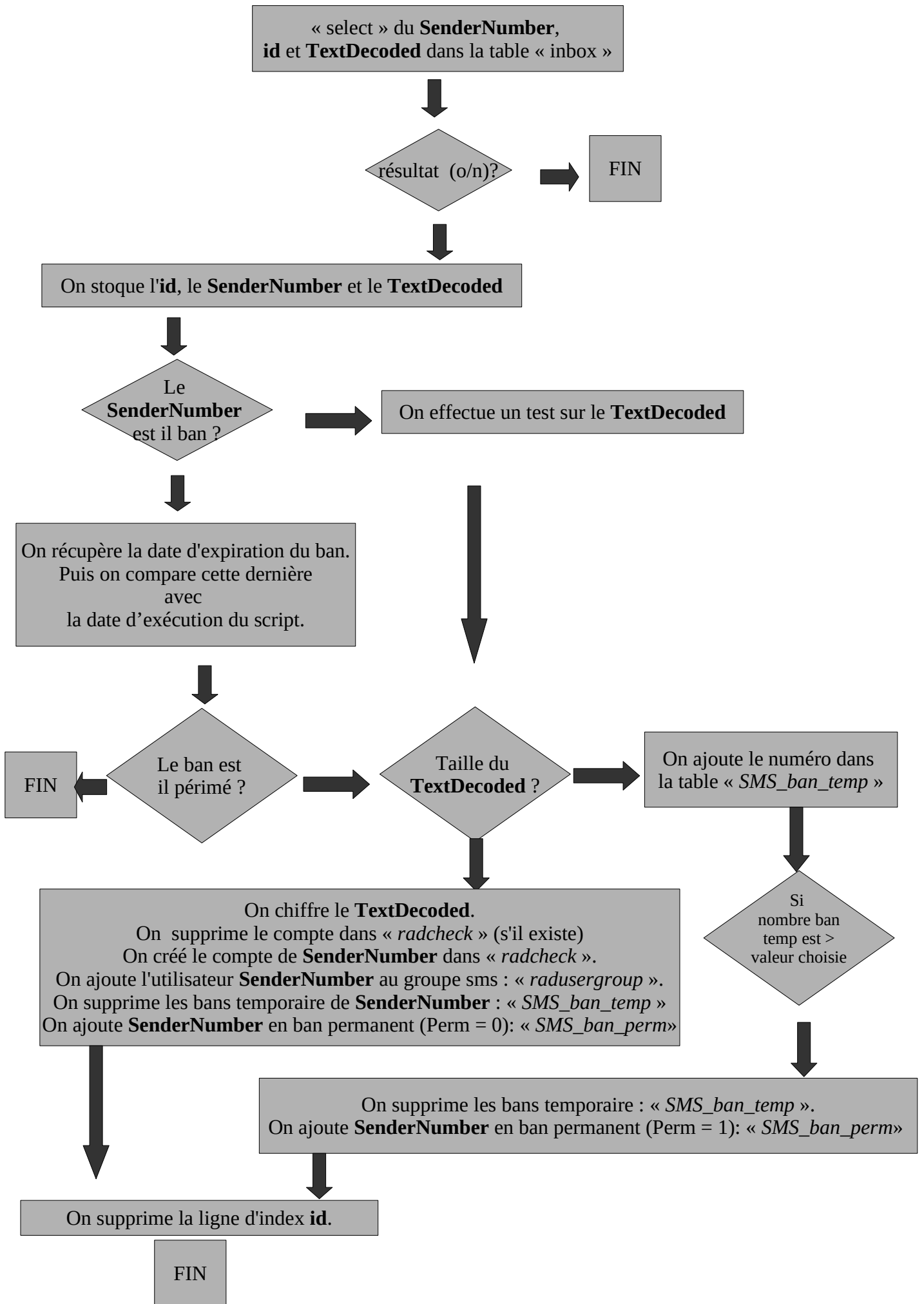
- Lancer et arrêter `gammu_smsd`
- Renseigner le code PIN de la carte SIM présente dans le modem-GSM.
- Renseigner la durée d'une session pour les comptes auto enregistrés.
- Renseigner le nombre de bannissements temporaire permis avant le bannissement permanent.
- Renseigner la durée d'un bannissement permanent (en jours).

On retrouve aussi sur cette page d'administration un tableau récapitulatif des comptes bannis :

- soit pour une raison de Compte existant
- soit pour une raison d'excès d'envois de SMS au serveur (Flood).

L'administrateur peut alors supprimer les numéros bannis. Cette action :

- supprime le numéro de l'expéditeur du SMS de la table des bannissements permanents (« `SMS_ban_perm` »),
- supprime le numéro de la table du groupe SMS (« `radusergroup` »)
- supprime le numéro de la table des comptes radius (« `radcheck` »).



8.2.2 - Dialogue avec le modem-GSM – déblocage avec le code PUK

« gammu-smsd » dialogue avec le modem-GSM au moyen de « commandes AT ». Ces commandes permettent de faire un grand nombre de choses, de la composition d'un numéro de téléphone à l'envoi de SMS, en passant par l'interrogation sur l'état du modem. La syntaxe de ces commandes suit le schéma suivant : « AT+commande ». Les deux premiers caractères (AT) sont l'abréviation du mot 'ATTention'. Ils permettent d'avertir le modem, afin qu'il prenne en compte la commande qui suit. La syntaxe « AT^commande » correspond aux commandes « étendues » pour les modem-GSM de type 3G.

- Liste de commandes AT: <http://www.activexperts.com/sms-component/at/etsi/>
- Liste de commandes AT spécifique Huawei : rechercher « HUAWEI UMTS Datacard Modem AT Command Interface Specification » dans un moteur de recherche.

Une fois « gammu-smsd » lancé, il vérifie l'état de la carte SIM en envoyant la commande : « AT+CPIN? ». Le modem-GSM répond en demandant le code PIN de la carte SIM. Gammu-smsd utilise alors le code PIN écrit dans le fichier de configuration « /etc/gammu_smsd_conf ».

Si ce code PIN est erroné, la carte SIM sera bloquée. Il faudra alors exploiter le code PUK pour la débloquent. La manipulation suivante permet de débloquent la carte SIM à l'aide d'un terminal et de commandes AT.

Arrêtez le service gammu via l'ACC ou via la commande « *alcasar-smsd.sh --stop* »

Dans un premier temps, installez « Minicom » sur votre système Linux : « *urpmi minicom* ». Modifiez la configuration de Minicom en lançant la commande « *minicom -s* ». Sélectionnez la 3e entrée (Serial port setup // Configuration du port série). Configurez le port série avec les paramètres suivants (cf. copie d'écran ci-dessous). Une fois les configurations effectuées, appuyer sur « échap » puis déplacer vous dans le menu pour enregistrer les modifications (Save setup as dfl // Enregistrer config. sous dfl). Vous pouvez alors quitter le menu (Exit from Minicom // Sortir de Minicom).

```
+-----[configuration]-----+
| Filenames and paths
| File transfer protocols
| Serial port setup
| Modem and dialing
| Screen and keyboard
| Save setup as dfl
| Save setup as..
| Exit
| Exit from Minicom
+-----+

A - Serial Device       : /dev/ttyUSB0
B - LockFile Location  : /var/lock
C - Callin Program     :
D - Callout Program    :
E - Bps/Par/Bits       : 115200 8N1
F - Hardware Flow Control : Yes
G - Software Flow Control : No

Change which setting?

| Screen and keyboard
| Save setup as dfl
| Save setup as..
| Exit
| Exit from Minicom
+-----+
```

```
+-----[configuration]-----+
| Filenames and paths
| File transfer protocols
| Serial port setup
| Modem and dialing
| Screen and keyboard
| Save setup as dfl
| Save setup as..
| Exit
| Exit from Minicom
+-----+
```

« AT+CPIN ? » permet de connaître l'état de la carte SIM. La copie d'écran ci-contre, montre la demande de code PIN de la carte SIM afin de pouvoir être exploitée : « +CPIN : SIM PIN ». La commande « AT+CPIN="xxxx" » (où xxxx correspond à votre code PIN), permet de renseigner le code PIN de la carte. Testez alors à nouveau l'état de la carte SIM.

Pour se connecter au modem-gsm, lancez la commande : « *minicom -c on* ». Vérifier la connexion au modem avec la commande suivante : « AT ». Si la configuration est correcte, le modem devrait renvoyer « OK ».

La commande

Dans le cas où la carte SIM est bloquée, le modem retourne « +CPIN : SIM PUK ». Récupérez le code PUK (disponible généralement en ligne sur le compte d'abonnement, ou fourni avec la carte SIM). Exécutez alors la commande « AT+CPIN="yyyyyyyyy","zzzz" » (où yyyyyyyyyy correspond à votre code PUK et zzzz correspond à votre nouveau code PIN).

```
AT+CPIN?
+CPIN: SIM PIN
OK
AT+CPIN="1234"
OK
AT+CPIN?
+CPIN: READY
OK
```

Remarque :

- Vous pouvez accéder au menu de Minicom via la combinaison de touche : « Ctrl+a » puis « z »
- Le modem vous retourne « OK » si la commande envoyée est correcte (syntactiquement) et reconnue.

8.3 - Watchdog

Ce script (« `alcasar-watchdog.sh` ») est lancé toutes les 3 minutes par le Daemon « `cron` ». Il permet de couvrir les fonctions suivantes :

- éviter les « oublis » de déconnexion (usager ayant fermé la fenêtre de connexion, panne d'équipement réseau, etc.) ;
- limiter le risque lié à l'usurpation d'adresse IP et d'adresse MAC sur le réseau de consultation (pirate interne) ;
- modifier la page WEB présentée aux usagers en cas de problèmes de connectivité détectés coté Internet

il est possible de désactiver le « `watchdog` » en commentant la ligne ci-dessous dans le fichier

« `/etc/cron.d/alcasar-watchdog` », puis, d'avertir « `cron` » de la modification : « `systemctl restart crond.service` »

```
# activation du "chien de garde" (watchdog) toutes les 3'
#*/3 * * * * root /usr/local/bin/alcasar-watchdog.sh > /dev/null 2>&1
```

8.4 - Statistiques réseau

Afin de protéger la vie privée des utilisateurs conformément aux préconisations de la CNIL, les statistiques de navigation ne comportent pas d'éléments permettant de lier les flux aux comptes utilisateur.

Il est possible via l'interface d'administration d'obtenir une représentation graphique de la charge réseau d'ALCASAR. Une sonde Netflow a été compilée à cet effet. Deux règles de pare-feu permettent de traiter tous les flux sortants par cette sonde

- Flux transitant dans les proxy HTTP internes : `$IPTABLES -A OUTPUT -o $EXTIF -p tcp --dport http -j NETFLOW`
- Flux sortant directement : `$IPTABLES -A FORWARD -i $TUNIF -s $PRIVATE_NETWORK_MASK -m state --state NEW -j NETFLOW`

Le module `ipt_NETFLOW` d'ALCASAR exporte ses informations sur le port 2055 de l'adresse de loopback.

La fonction de collecteur est prise en compte par le démon « `Nfcapd` » en écoute sur le port 2055. Il collecte les flux NetFlow et crée un fichier toutes les 5 minutes dans le répertoire « `/var/log/nfsen/profile-data/live/alcasar_netflow/` ».

Le module `Nfexpire` (installé avec le RPM « `nfdump` ») permet de réaliser une rotation sur les fichiers capturés par « `nfcapd` ». Une règle ajoutée à « `/etc/cron.d/alcasar-netflow` » permet d'actualiser tous les jours le délai d'expiration sur le répertoire contenant les fichiers de capture Netflow : `nfexpire -e /var/log/nfsen/profile-date/live/alcasar_netflow/ -t 1Y`

Tel quel, le format Netflow n'est pas lisible, en revanche il est possible à tout moment d'afficher le contenu des fichiers de capture de manière lisible. Il faut pour cela utiliser un **interpréteur Netflow** (ex : « `nfdump` ») comme suit : `nfdump -R <fichier_au_format_netflow> -o extended -a`

Tous les graphes et statistiques sur le trafic sont réalisés par `Nfsen`. Ce dernier permet à partir des données « Netflow » capturées de générer différents graphes relatifs à la charge du LAN. L'avantage de cet outil est qu'il ne crée qu'un seul fichier par graphe puisqu'il concatène le graphe déjà existant avec les nouvelles données en entrée. Ce mode de fonctionnement permet un gain de place non négligeable sur le disque dur. Le répertoire contenant tous les fichiers de `Nfsen` est « `/var/www/html/acc/manager/nfsen/` ».

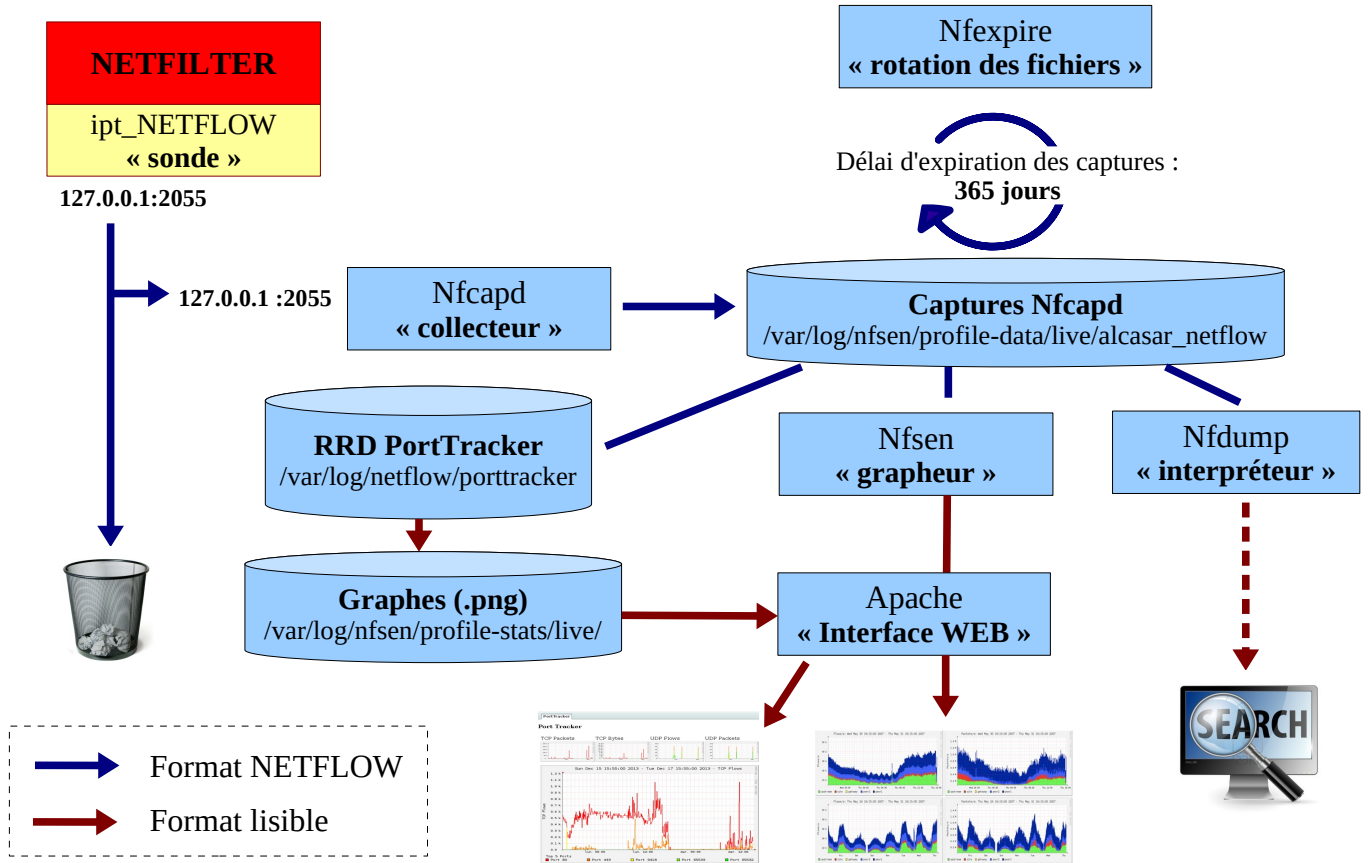
Une règle similaire est également nécessaire sur le profil créé par « `Nfsen` », à savoir « `live` » (cf `alcasar.sh`) : `nfsen -m live -e 365d`

Les données supprimées ne sont plus accessibles en tant que telles dans les tableaux de statistiques fournis par `Nfsen`. En revanche ces dernières ayant été concaténées par `Nfsen` lors de la réalisation des graphes, elles restent toujours visibles sur les graphes.

Un module complémentaire « **PortTracker** » a été ajouté à `Nfsen`. Ce dernier permet d'obtenir des statistiques de charge réseau par protocoles. Ces statistiques sont stockées dans une base de données de type RRD (« Round-Robin-database »). Ce type de base de données met en place via des algorithmes mathématiques un système de rotation des fichiers visant à supprimer les plus anciens lorsque de nouveaux arrivent. De cette manière la base de données conserve toujours sa taille initiale, qui est dans notre cas d'environ 8Go. Cette base est située dans le répertoire « `/var/log/netflow/porttracker` ».

Les fichiers « `.rrd` » de cette base de données doivent être accessibles à la fois par `Nfsen` et par `Lighttpd`. La base de données RRD a donc comme groupe propriétaire « **apache** ».

Grphe récapitulatif de l'architecture NETFLOW



8.5 - Contournement (by-pass)

En cas de problème technique concernant une des briques logicielles du portail (principalement « coova-chilli »), il est possible de court-circuiter le module d'authentification tout en maintenant le traçage des logs réseau (pare-feu).

Un script lancé localement en root `alcasar-bypass.sh -on | --off` permet au choix de mettre :

- en mode « On » le by-pass → le portail désactive les services coova-chilli, squid, E2Guardian
- en mode « Off » : le portail est en mode normal. Tous les services nécessaires sont activés.

8.6 - Load balancing de connexions

Le script `alcasar-load_balancing.sh` permet de disposer de plusieurs passerelles d'accès à l'Internet. Le script (actif ou non) est lancé au démarrage du serveur par `/etc/rc.local`.

Les interfaces virtuelles qui s'appuient sur eth0 sont créées au démarrage du script et sont définies dans le fichier `/usr/local/etc/alcasar.conf`. Le formatage de ces interfaces est de la forme : `WANx="active[1|0],@IPx/mask,GWx,Weight,MTUx` avec

- WANx avec x, un indice de 1 à ..., pour définir le nom de l'interface virtuelle,
- le premier paramètre (non traité pour le moment) qui prend en compte le côté actif de la carte ou non,
- `l@IP` / masque de l'interface virtuelle,
- @IP de la passerelle correspondant à cette interface virtuelle,
- le poids (weight) : un poids identique indique une répartition « égale » sur les interfaces de mêmes poids : la valeur par défaut est « 1 », identique à la passerelle par défaut,
- la valeur du MTU correspond à cette interface.

Pour être actif, le load_balancing nécessite de positionner le paramètre «MULTIWAN=on» (ou 'On').

Le test de connectivité qui s'effectue sur google et sur www.example.com (voir le paramètre TESTIPS dans le

script `alcasar-load_balancing.sh` pour modifier les @ip de tests) est actif lorsque le paramètre `FAILOVER` comporte une valeur différente de "0". Il est effectué tous les x secondes. La valeur doit être positive et entière.

Le script est dorénavant lancé par défaut au démarrage du serveur. Il peut être manipulé par la commande `alcasar-load_balancing.sh` avec les options suivantes :

- `start` : lancement du script qui crée les interfaces virtuelles, les monte et renseigne la table de routage,
- `stop` : arrêt du script et démontage des interfaces virtuelles
- `Status` : état du script (uniquement visible en mode failover) ; en effet, en dehors de ce mode, le script n'est pas lancé en mode « daemon ».

8.7 - Re-Horodatage des fichiers journaux

Lors de la réinstallation d'un serveur, il peut être utile de réinstaller les fichiers journaux d'origines (avant le crash). Afin que les fichiers disposent d'une date cohérente et que l'effacement des logs s'effectue régulièrement (au bout d'un an), les journaux doivent disposer de la date en relation avec leur rotation originale. C'est tout l'objet du script `alcasar-dateLog.sh` qui plaque les bons attributs 'date:heure' à partir du nom de fichier (qui comprend un suffixe <date>).

8.8 - Sauvegardes

Les sauvegardes d'ALCASAR sont disponibles sous 3 formes : l'archive des journaux de traçabilité, la base de données des utilisateurs et le rapport d'activité hebdomadaire.

8.8.1 - Sauvegarde des journaux de traçabilité

Les journaux du firewall (`/var/log/firewall/*`) et de la sonde Netflow (`/var/log/nfsen/profiles-data/live/alcasar-netflow`) sont « rotatés » chaque semaine. Chaque semaine (lundi à 5h35), une tâche planifiée appelle le script « `alcasar-archive.sh -now` » qui crée une archive compressée, constituée de ces journaux et de la base de données des utilisateurs. Cette archive est copiée dans le répertoire « `/var/Save/archive` » afin d'être disponible dans l'ACC pour téléchargement (fichier « `traceability-<date>.tar.gz` »). Afin de limiter la conservation des traces à 1 an, ce script efface toutes les archives dont la date de création est supérieure à 365 jours.

8.8.2 - Sauvegarde de la base de données

Chaque semaine (lundi à 04h45), le script « `alcasar-mysql.sh -dump` » vérifie, sauvegarde et compresse la base de données des utilisateurs dans le répertoire « `/var/Save/base` » sous la forme : « `alcasar-users-database-<date>.sql.gz` ».

Chaque nuit à 4h40, le script « `alcasar-mysql.sh --expire_user` » supprime les utilisateurs dont la date d'expiration est dépassée de plus de 7 jours.

8.8.3 - Le rapport d'activité hebdomadaire

Ce rapport au format « PDF » est généré par le script « `alcasar-activity_report.sh` » tous les dimanches matin à partir de 5h35. Les rapports sont disponibles via l'ACC. Ils sont stockés dans le répertoire « `/var/Save/activity_report/` ».

9 - Annexes

Ce chapitre reprend les fichiers de configuration spécifiques à ALCASAR.

9.1 - Coova-chilli

Les fichiers se situent sous « */etc/* , */etc/chilli* et */usr/local/etc* ».

- Fichier principal : *chilli.conf* (sous */etc*)
- Exceptions Domaines : *alcasar-uamdomain* (sous */usr/local/etc*)
- Exceptions URLs : *alcasar-uamallowed* (sous */usr/local/etc*)
- Exceptions d'authentification par MAC Adresses : *alcasar-macallowed* (sous */usr/local/etc*)
- L'association dynamique d'*@IP* statiques s'effectue par le biais du fichier : *alcasar-ethers* (sous */usr/local/etc*)

9.2 - Freeradius

Les fichiers du démon radius se situent sous « */etc/raddb* ».

- Fichier principal : *radiusd.conf*
- Fichier de connexion BDD : *sql.conf*
- Fichier des clients autorisés à requêter le service radiusd : *clients.conf*
- Fichier dédié : *alcasar* (sous */etc/raddb/sites-available* avec un lien symbolique qui lie les « *sites-enable* »)

Voici la liste des attributs (source : <https://raw.githubusercontent.com/coova/coova-chilli/master/doc/attributes>)

# Name	Type	Comment
User-name	String	Full username as entered by the user.
User-Password	String	Used for UAM as alternative to CHAP-Password and CHAP-Challenge.
CHAP-Password	String	Used for UAM CHAP Authentication
CHAP-Challenge	String	Used for UAM CHAP Authentication
EAP-Message	String	Used for WPA Authentication
NAS-IP-Address	IPAddr	IP address of Chilli (set by the "nasip" or "radiuslisten" option, and otherwise "0.0.0.0")
Service-Type	Integer	Set to Login (1) for normal authentication requests. The Access-Accept message from the radius server for configuration management messages must also be set to Administrative-User.
Framed-IP-Address	IPAddr	IP address of the user, which is configurable during MAC authentication in the Access-Accept.
Framed-IP-Netmask	IPAddr	IP netmask of the user, which is configurable during MAC authentication in the Access-Accept.
Filter-ID	String	Filter ID pass on to scripts possibly.
Reply-Message	String	Reason of reject if present.
State	String	Sent to chilli in Access-Accept or Access-Challenge. Used transparently in subsequent Access-Request.
Class	String	Copied transparently by chilli from Access-Accept to Accounting-Request.
Session-Timeout	Integer	Logout once session timeout is reached (seconds)
Idle-Timeout	Integer	Logout once idle timeout is reached (seconds)
Called-Station-ID	String	Set to the "nasmac" option or the MAC address of chilli.
Calling-Station-ID	String	MAC address of client
NAS-Identifier	String	Set to radiusnasid option if present.
Acct-Status-Type	Integer	1=Start, 2=Stop, 3=Interim-Update
Acct-Input-Octets	Integer	Number of octets received from client.
Acct-Output-Octets	Integer	Number of octets transmitted to client.
Acct-Session-ID	String	Unique ID to link Access-Request and Accounting-Request messages.
Acct-Session-Time	Integer	Session duration in seconds.
Acct-Input-Packets	Integer	Number of packets received from client.
Acct-Output-Packets	Integer	Number of packets transmitted to client.
Acct-Terminate-Cause	Integer	1=User-Request, 2=Lost-Carrier, 4=Idle-Timeout, 5=Session-Timeout, 11=NAS-Reboot
Acct-Input-Gigawords	Integer	Number of times the Acct-Input-Octets counter has wrapped around.
Acct-Output-Gigawords	Integer	Number of times the Acct-Output-Octets counter has wrapped around.
NAS-Port-Type	Integer	19=Wireless-IEEE-802.11
Message-Authenticator	String	Is always included in Access-Request. If present in Access-Accept, Access-Challenge or Access-reject chilli will validate that the Message-Authenticator is correct.
Acct-Interim-Interval	Integer	If present in Access-Accept chilli will generate interim accounting records with the specified interval (seconds).
WISPr-Location-ID	String	Location ID is set to the radiuslocationid option if present. Should be in the format
WISPr-Location-Name	String	Location Name is set to the radiuslocationname option if present. Should be in the format
WISPr-Logoff-URL	String	Included in Access-Request to notify the operator of the log off URL. Defaults to "http
WISPr-Redirection-URL	String	If present the client will be redirected to this URL once authenticated. This URL should include a link to WISPr-Logoff-URL in order to enable the client to log off.
WISPr-Bandwidth-Max-Up	Integer	Maximum transmit rate (b/s). Limits the bandwidth of the connection. Note that this attribute is specified in bits per second.
WISPr-Bandwidth-Max-Down	Integer	Maximum receive rate (b/s). Limits the bandwidth of the connection. Note that this attribute is specified in bits per second.
WISPr-Session-Terminate-Time	String	The time when the user should be disconnected in ISO 8601 format (YYYY-MM-DDThh
CoovaChilli-Max-Input-Octets	Integer	Maximum number of octets the user is allowed to transmit. After this limit has been reached the user will be disconnected.
CoovaChilli-Max-Output-Octets	Integer	Maximum number of octets the user is allowed to receive. After this limit has been reached the user will be disconnected.
CoovaChilli-Max-Total-Octets	Integer	Maximum total octets the user is allowed to send or receive. After this limit has been reached the user will be disconnected.
CoovaChilli-Bandwidth-Max-Up	Integer	Maximum bandwidth up
CoovaChilli-Bandwidth-Max-Down	Integer	Maximum bandwidth down
CoovaChilli-Config	String	Configurations passed between chilli and back-end as name value pairs
CoovaChilli-Lang	String	Language selected in user interface
CoovaChilli-Version	String	Contains the version of the running CoovaChilli
CoovaChilli-DHCP-Netmask	IPAddr	DHCP IP netmask of the user, which is configurable during MAC authentication in the Access-Accept.
CoovaChilli-DHCP-DNS1	IPAddr	DHCP DNS1 of the user, which is configurable during MAC authentication in the Access-Accept.
CoovaChilli-DHCP-DNS2	IPAddr	DHCP DNS2 of the user, which is configurable during MAC authentication in the Access-Accept.
CoovaChilli-DHCP-Gateway	IPAddr	DHCP Gateway of the user, which is configurable during MAC authentication in the Access-Accept.
CoovaChilli-DHCP-Domain	IPAddr	DHCP Domain of the user, which is configurable during MAC authentication in the Access-Accept.
MS-MPPE-Send-Key	String	Used for WPA
MS-MPPE-Recv-Key	String	Used for WPA

9.3 - Dnsmasq

En fonctionnement normal, 4 instances de Dnsmasq sont lancées (une instance en mode « forward » sur le port 53, une instance en mode « blacklist » sur port 54, une instance en mode whitelist sur le port 55 et une instance en mode « blackhole » sur le port 56). En mode secours (bypass), Dnsmasq fournit le service DHCP (c'est « coova-chilli » qui s'occupe de ce service en mode normal).

- Fichiers principaux : */etc/dnsmasq.conf* / */etc/dnsmasq-blacklist* / */etc/dnsmasq-whitelist* / */etc/dnsmasq-blackhole*
- les usagers sont redirigés sur une instance de DNSmasq en fonction de leur attribut de filtrage.

9.4 - Parefeu

- Fichier principal du pare-feu d'ALCASAR : *alcasar-iptables.sh* (sous */usr/local/bin*)
- Règles personnalisées du pare-feu : *alcasar-iptables-local.sh* (sous */usr/local/etc*)
- Fichier de filtrage Réseau (associé à *alcasar-nf.sh*) : *alcasar-iptables-exception*
- Activer/désactiver le filtrage web : *alcasar-bl.sh* (sous */usr/local/bin*)
- Fichier listant les classes de filtrage (associé à *alcasar-nf.sh*) : *alcasar-bl-categories-enabled* ; utilisée par le fichier *alcasar-bl.sh* pour le filtrage dnsmasq et E²Guardian.
- Fichier contenant la liste complète des domaines par classe issue de la liste noire de Toulouse : *alcasar-dnsfilter-available* (sous */usr/local/etc/*)
- Fichier du pare-feu d'ALCASAR utilisé en mode ByPass : *alcasar-iptables-bypass.sh* (sous */usr/local/bin*)

9.5 - E²Guardian

Les fichiers d'E²Guardian se situent sous « */etc/e2guardian* ».

- Fichier principal de configuration : *e2guardian.conf*
- Fichier concernant le groupe 1 utilisé par ALCASAR : *e2guardianf1.conf*
- Le répertoire « *lists* » contient les fichiers de filtrage proprement dits :
 - « *bannedsitelist* » : non exploité (cf. §6.2)
 - « *exceptionsitelist* » : non exploité
 - « *bannediplist* » : non exploité
 - « *exceptioniplist* » : exploité pour la liste des adresses IP en exception de filtrage
 - « *exceptionurllist* » exploité pour la liste des URLs réhabilitées
 - « *bannedurllist* » contient la liste des catégories d'URL à filtrer
 - « *blacklists* » : contient la BL de Toulouse. Les répertoires « *urls* » de chaque catégorie sont exploités pour le filtrage d'URLs.

9.6 - Tinyproxy

Le fichier de configuration de ce proxy léger est « */etc/tinyproxy.conf* ». Il est configuré en mode transparent. Il n'a été mis en place que pour éviter d'avoir à lancer plusieurs instances de l'antivirus (HAVP + libclamav).

9.7 - Ulogd

Le daemon ulogd centralise les logs du pare-feu (dissociés des logs 'messages') ; tous les journaux d'événements sont gérés en mode texte.

- Fichier de configuration : *ulogd.conf*
- Fichier concernant les flux Ssh extérieurs en provenance de *eth0* : *ulogd-ssh.conf*
- Fichier concernant les flux bloqués en provenance du réseau extérieur : *ulogd-ext-access.conf*

La rotation des logs s'effectue de manière hebdomadaire pour httpd, squid et tracability

9.8 - HAVP + Clamav

Le moteur HAVP est paramétré pour fonctionner avec la bibliothèque libClamav

- HAVP :
 - Fichier de configuration du moteur antivirus : *havp.config*
 - Un répertoire au format tmpfs (*/var/tmp/havp*) est utilisé pour accélérer le traitement du scan ; il est monté au démarrage du daemon havp et nettoyé et démonté à son arrêt.
- libClamav
 - la périodicité de mise à jour des signatures est paramétrée par défaut à 12 fois par jour.

9.9 - Distribution Mageia et ses dépôts

La distribution Mageia est utilisée comme système d'exploitation support pour ALCASAR. Les mises à jour et l'installation des paquets s'effectuent à l'aide des outils natifs : « *urpmi* ».

Les fichiers de configurations se trouvent sous */etc/urpmi* :

- source des miroirs : *urpmi.cfg* ;
- exceptions des mises à jour de paquets : *skip.list* ; permet d'exclure des mises à jour certains paquets pouvant éventuellement troubler le fonctionnement du portail.
- Pour effectuer une mise à jour automatique : *urpmi --auto-update --auto*
- Pour effectuer du ménage : *urpme --auto-orphans --auto*

9.10 - Étude du remplacement de DNSMasq par Unbound

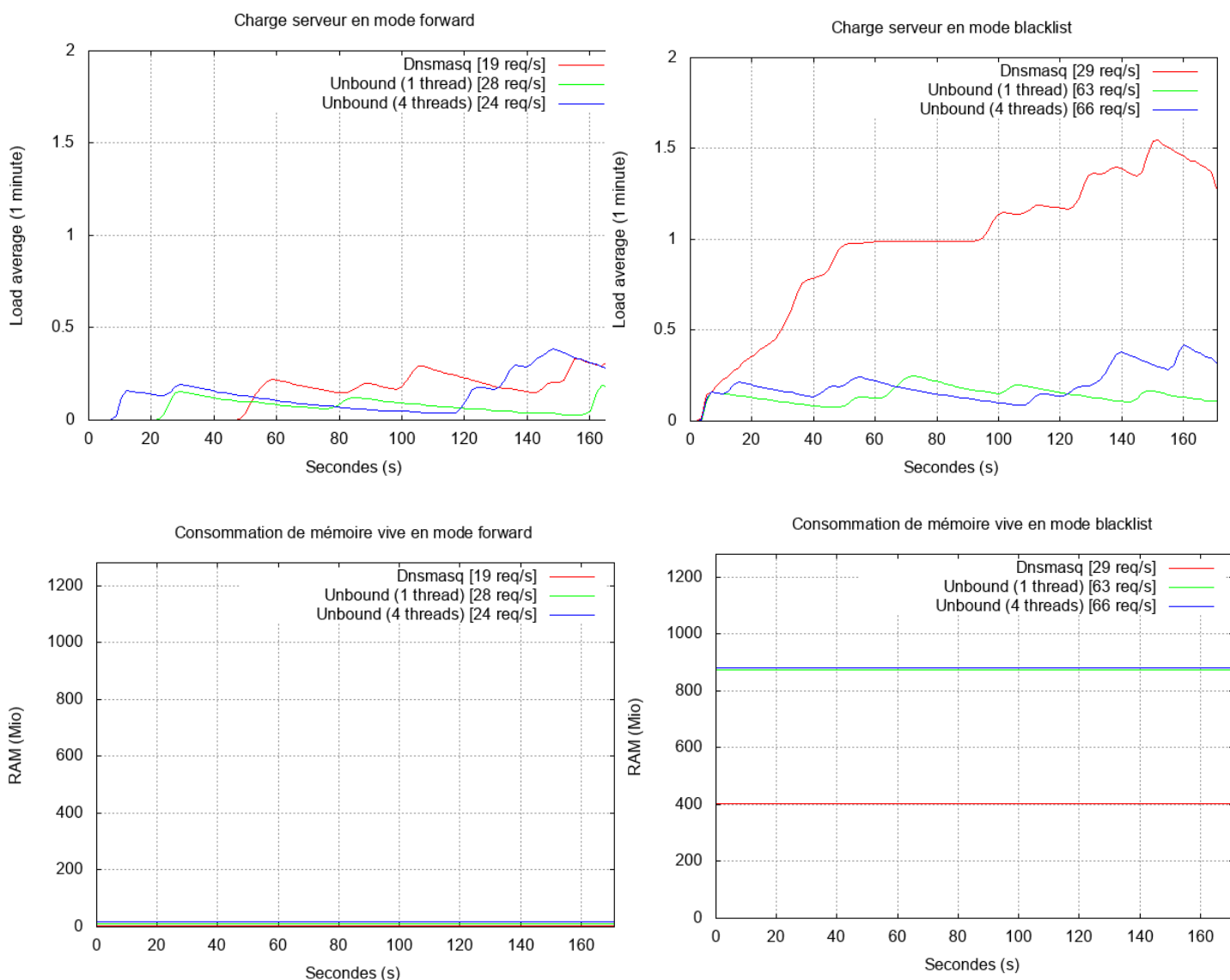
Étude réalisée par Lucas ECHARD

Problématique : À forte charge, le service « Dnsmasq » en mode blacklist consomme beaucoup de ressources et impacte les performances des serveurs ALCASAR.

Méthodologie : Un benchmark comparatif a été réalisé entre « Unbound » et « Dnsmasq ». Voici les conditions dans lesquelles les tests ont été réalisés :

- utilisation d'une machine virtuelle Mageia 6 (4 cœurs, 8 Gio de RAM) comprenant uniquement Unbound et Dnsmasq ainsi que les fichiers de blacklist/whitelist correspondant ;
- les services dnsmasq, dnsmasq-blacklist, unbound, unbound-blacklist ont été testés indépendamment ;
- une seconde VM faisant office de machine de consultation contenant un script générant des requêtes DNS (50 % whitelist, 50 % blacklist) pour stresser le serveur ;
- le « load average » et la consommation de mémoire vive liée au service testé sont sauvegardés à intervalle régulier sur le serveur.

Résultats : Voici ci-dessous les graphiques obtenus en fonction des données relevées.



Conclusion : comparé à « Dnsmasq », « Unbound » gère mieux la montée en charge au niveau du CPU mais consomme deux fois plus de RAM que « Dnsmasq » en mode blacklist.

Suite à cette étude, il a été décidé de lancer les travaux de migration vers « Unbound » à partir de la version 3.3 (fin 2018).

10 - Plan de tests

Installation / désinstallation / mise à jour

Function to test	Expected	3.5
Install script (alcasar.sh -i) <ul style="list-style-type: none"> • disconnect the wire on EXTIF • disconnect the router • Internet off (?) • Mode debug (set DEBUG_ALCASAR variable to « on ») • Install again (new installation) 	<ul style="list-style-type: none"> • The script stops • The script stops • The script stops • End of installation without error or warning • End of re-installation without error or warning 	<ul style="list-style-type: none"> ✓ ✓ ✓ ✓ ✓
After first reboot <ul style="list-style-type: none"> • All services needed are started • No error during the boot process • All ALCASAR services are started 	<ul style="list-style-type: none"> • “systemctl - - failed” must return 0 line • “journalctl -b grep 'failed error' ” • “alcasar-daemon.sh” 	<ul style="list-style-type: none"> ✓ ✓ ✓
Uninstall script (alcasar.sh -u) <ul style="list-style-type: none"> • Stop & uninstall all services • Network is working 	<ul style="list-style-type: none"> • All services are removed (ps fax) • ping Internet 	<ul style="list-style-type: none"> ✓ ✓
Minor update <ul style="list-style-type: none"> • Install on a running system (update mode) • verify that old parameters are take into account 	<ul style="list-style-type: none"> • End of re-installation without error or warning • network parameters • DNS (local & remote resolutions) • security certificates • admin accounts • users database 	<ul style="list-style-type: none"> ✓ ✓ ✓ ✓ ✓ ✓
Major update <ul style="list-style-type: none"> • Create a conf file in /var/tmp • Install Linux without formatting /var • Install ALCASAR • verify that old parameters are taken into account 	<ul style="list-style-type: none"> • alcasar-conf.sh -create (or ACC) • /var/tmp/alcasar-conf.tar.gz is still present • The conf file is taken into account • (see minor update) 	<ul style="list-style-type: none"> ✓ ✓ ✓ ✓

Function to test	Expected	3.5
<p>User management</p> <ul style="list-style-type: none"> • Create 4 users (“direct”, “havg_bl”, “havg_wl”, “havg”) each-one with a different filtering system. Set also other attributes to test. • havg test • havg_bl test • change a user attribute & test • create a group “student” with a filtering attribute. Set other attributes (ie : redirect_url). Create a user in this group and verify that the attributes are inherited. • Empty the database <ul style="list-style-type: none"> ◦ with the script “alcazar-mysql.sh -raz” ◦ via “ACC” • Import a users list (simple text file with and without password) • import a users database <ul style="list-style-type: none"> ◦ with the script “alcazar-mysql.sh -i xxx.sql” ◦ via ACC • Connection to an AD server & test remote users • Connection to a LDAP server & test remote users 	<p>The PDF voucher is ok. “ipset list -n“ to see the name of the ipset “ipset list xxx”, list the content of the ipset xxx (xxx = 'not_filtered', 'havg_bl', 'havg_wl' or 'havg')</p> <ul style="list-style-type: none"> • eicar test (search “eicar”+”download” on gg) • DNS blacklist test (www.warez.com & www.rael.org) • URL blacklist test (wawa-mania.eu/warez) • IP blacklist test (151.196.210.42 & 207.239.30.20) • URL with IP@ instead of domain name • Google safesearch • student group : check this user 'with edit a user'. Two columns should appear. The first one is group attributes and second one is user attributes • the connected users are disconnected. The database is empty. • pdf & txt password file is generated (remove after 24h) • do nothing for users already exist (just change their password if it exists in text file) • To get a SQL database sample. Go to /var/Save/base then use 'gzip -d' to extract. 	<p>✓</p> <p>✓</p> <p>✓</p> <p>✓</p> <p>✓</p> <p>✓</p> <p>X</p> <p>X</p> <p>✓</p> <p>✓</p> <p>✓</p> <p>X</p> <p>X</p> <p>X</p> <p>X</p>
<p>Exceptions</p> <ul style="list-style-type: none"> • Create a user with a PC MAC address as login and 'password' as password • Add domain names and IP addresses in exception form 	<ul style="list-style-type: none"> • The PC should connect to Internet without interception • Users should connect to these IP & domain names without interception 	<p>✓</p> <p>✓</p>
<p>Watchdog</p> <ul style="list-style-type: none"> • Disconnect the Ethernet cable on EXTIF • Connect again the Ethernet cable on EXTIF 	<ul style="list-style-type: none"> • All the DNS requests are send to the DNS blackhole (port 56). “iptables -nvL -tnat” to verify (first line redirect DNS to port 56) • for the users : Internet unavailable page • The main page alert that the access is down • Everything return to normal state 	<p>✓</p> <p>✓</p> <p>✓</p>

Function to test	Expected	3.5
Blacklist/Whitelist filtering The 3 following actions are performed when the admin wants to update BL with ACC <ul style="list-style-type: none"> • alcasar-bl.sh -download • alcasar-bl.sh -adapt • alcasar-bl.sh -reload • add and remove a 'blacklist' file containing IP and domain names • ipset list whitelist_ip_allowed • ipset list havp_wl • cat /usr/local/share/unbound-wl/ossi.conf • cat /usr/local/share/iptables-wl-enabled/ossi 	Download the last BL from Toulouse (saved in /tmp) compute the BL in order to create three sub-BL (domains, URL and IP) Reload the services that use the BL (dnsmasq-blacklist, dnsmasq-whitelist and netfilter) Verify that the IPs are really in the ipset (command "ipset test bl_ip_blocked @IP") Verify that the domain names are really blocked. Remove the file show ip allowed by the whitelist show current user filtered by the whitelist Show allowed website manually added with ACC the same for IP address	✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓
Network protocols filtering <ul style="list-style-type: none"> • switch the 1st filter on (http + https) • switch the 2nd filter on • Add a custom port (ex : ssh) 	Check that you can't connect to a ssh/ftp server Check that you can now connect to a ssh/ftp server Check that you can connect to that port	✓ ✓ ✓
Users activities <ul style="list-style-type: none"> • login & logout with the alcasar main page • logout with the URL : 'http://logout' • Change password • Import the CA certificate 		✓ ✓ ✓ ✓
SMS (autoregistration) <ul style="list-style-type: none"> • Check gammu (it's not a standard service) • Check the version of the database 	/var/log/gammu-smsd/gammu-smsd.log Compare the SQL creation script (in the RPM and in the folder "conf" of ALCASAR archive)	✓ ✓
After several days working <ul style="list-style-type: none"> • "journalctl -b grep 'failed error' " • logrotate process test 	View and analyze errors <ul style="list-style-type: none"> • logrotate -vf /etc/logrotate.conf (test all file in /etc/logrotate.d) • logrotate -vf /etc/logrotate.d/file_to_test 	
Admin <ul style="list-style-type: none"> • alcasar-bypass.sh -on • alcasar-bypass.sh -off • importation of an official certificate 	ACC isn't available. Users can surf without interception ACC is available. Users are intercepted	X X
Uninstall <ul style="list-style-type: none"> • alcasar.sh -u 	All services are stopped and the modified config files are removed.	✓

Function to test	Expected	3.5
Update a previous version	A conf file (alcasar-conf.tar.gz) is created in /var/tmp The parameters of the previous version are enabled (logo, admin accounts, users database, network parameters, SSL certificates, custom BL&WL).	

11 - Security tests

- internal audit with “lynis”
 - Lynis V2.7.5 - 06/2019 - Hardening index : [66]
- external audit with “openvas”

Nmap + NSE results

```
PORT      STATE    SERVICE  VERSION
22/tcp    open     ssh      OpenSSH 6.6 (protocol 2.0)
| ssh-hostkey:
| 1024 5b:39:5f:2a:cb:96:3e:60:a5:20:48:30:95:76:70:15 (DSA)
|_ 2048 bf:79:4c:a1:2e:b0:5d:f0:c8:52:f8:52:ff:1d:f9:eb (RSA)
53/tcp    open     domain   dnsmasq 2.75
55/tcp    open     domain   dnsmasq 2.75
| dns-nsid:
|_ bind.version: dnsmasq-2.75
56/tcp    open     domain   dnsmasq 2.75
| dns-nsid:
|_ bind.version: dnsmasq-2.75
80/tcp    open     http      Apache httpd
|_ http-favicon: Unknown favicon MD5: 725EB60B1CECCCF7DEF498E09422AA79
|_ http-methods: No Allow or Public header in OPTIONS response (status code 200)
|_ http-title: ALCASAR - Welcome on ALCASAR
443/tcp   open     ssl/http  Apache httpd
|_ http-favicon: Unknown favicon MD5: 725EB60B1CECCCF7DEF498E09422AA79
|_ http-methods: No Allow or Public header in OPTIONS response (status code 200)
|_ http-title: ALCASAR - Welcome on ALCASAR
| ssl-cert: Subject: commonName=alcasar.localdomain/organizationName=ALCASAR-Team/stateOrProvinceName=none/countryName=FR
| Issuer: commonName=ALCASAR-local-CA/organizationName=ALCASAR-Team/stateOrProvinceName=none/countryName=FR
| Public Key type: rsa
| Public Key bits: 2048
| Not valid before: 2016-04-25T11:20:45+00:00
| Not valid after: 2020-04-24T11:20:45+00:00
| MD5: a002 37c4 db82 2ef0 2a4c 1555 8193 b031
|_ SHA-1: 02c6 afa2 3ac8 0477 d1c0 3e7d 2c07 c264 7cf9 543e
|_ ssl-date: 1979-07-28T15:53:31+00:00; -36y332d12h47m10s from local time.
3990/tcp  open     tcpwrapped
53/udp    open     domain   dnsmasq 2.75
| dns-nsid:
|_ bind.version: dnsmasq-2.75
|_ dns-recursion: Recursion appears to be enabled
67/udp    open|filtered dhcpcd
123/udp   open     ntp       NTP v4
| ntp-info:
| receive time stamp: 2016-06-25T04:40:51
| version: ntpd 4.2.6p5@1.2349-o Fri Jun 3 18:26:59 UTC 2016 (1)
| processor: x86_64
| system: Linux/4.4.13-server-1.mga5
| leap: 0
| stratum: 3
| precision: -22
| rootdelay: 26.667
| rootdisp: 70.710
| refid: 178.32.54.53
| reftime: 0xdb1885ea.5d5387b5
| clock: 0xdb1889c9.cc628078
| peer: 30571
| tc: 10
| mintc: 3
| offset: 0.224
| frequency: -69.250
| sys_jitter: 0.916
| clk_jitter: 0.501
|_ clk_wander: 0.065
```