MÉMOIRE DE FIN D'ÉTUDES
HOUSSENBAY Olivier

Entreprise d'accueil : Laboratoire de cryptologie et virologie opérationnelle $(C+V)^O$
38 Rue des Docteurs Calmette et Guérin, 53000 Laval

FreeRadius, un serveur d'authentification forte pour ALCASAR

FreeRadius, a strong authentication server for ALCASAR

M. Vincent GUYOT, Président du jury, Enseignant chercheur à l'ESIEA, Docteur en Informatique

M. Richard REY, Maître de stage, Ingénieur de recherche en sécurité informatique

M. Nicolas DEVILLY, Tuteur pédagogique, Chef d'entreprise de la société AtoutSens

Table des matières

Remerciements

Avant tout développement sur cette expérience, il apparaît opportun de commencer ce mémoire par des remerciements.

Mes premières pensées se tournent vers mon maître de stage M. Richard REY, ingénieur de recherche en sécurité informatique, ainsi que M. Eric Filiol, directeur de recherche, qui m'ont accueilli au Laboratoire CVO[1] de l'ESIEA[2] OUEST.

Je remercie M. REY pour avoir tout mis en œuvre pour que mon stage de fin d'études se passe dans les meilleures conditions ainsi que pour ses judicieux conseils formulés avec beaucoup de pédagogie tout au long du stage.

Je souhaiterai aussi remercier M. Nicolas DEVILLY, mon tuteur pédagogique avec qui j'ai entretenu de bons rapports. Il a été à l'écoute et m'a appris à adopter une bonne approche professionnelle au sein du Laboratoire CVO.

Et enfin, un grand merci à l'équipe du laboratoire qui m'a accompagné tout au long de cette expérience et qui a eu la gentillesse de faire de ce stage, un moment très profitable.

1 Cryptologie et Virologie Opérationelles
2 École Supérieure d'Informatique, Électronique et Automatique

Résumé

Comment déployer un système d'authentification forte au sein de ALCASAR : Application Libre pour le Contrôle d'Accès Sécurisé et Authentifié au Réseau ?

Le stage s'est déroulé en trois parties distinctes : une phase d'analyse et de remise à niveau, une approche approfondie du sujet et la mise en œuvre du travail à réaliser.

Les prémices du projet ont consisté à analyser le fonctionnement du serveur d'authentification Freeradius et des protocoles mis en place.

La vérification et la correction de la configuration du serveur Freeradius a permis de mieux connaître son principe de fonctionnement notamment sur les aspects d'arborescence des fichiers utilisés par le serveur, les compteurs SQL pour le calcul des autorisations et le protocole d'identification utilisé lors de l'authentification sur le serveur.

La dernière partie du stage consistait à reprendre toutes les notions acquises précédemment et les mettre en œuvre de manière à réaliser une authentification forte basée sur le standard IEEE 802.1X avec une méthode d'authentification bilatérale par certificat client/serveur.

Ce mémoire de stage au sein du laboratoire CVO de l'ESIEA-Ouest montre une démarche (problèmes et solutions proposées) possible pour appréhender et mettre en place une authentification forte avec un serveur Radius.

Abstract

How to deploy a strong based authentication in ALCASAR: Free Application for Secured and Authenticated Access Control to Network?

This internship has been done in three parts : an analysis and a review of the prerequisites, a deep approach about the subject then an implementation of the content involved.

Premises of the project consisted to analyze how the server authentication FreeRadius runs and the protocols inside.

Checking and patching have been processed through the configuration of Freeradius server to a better understanding of the operating principle particularly on file tree aspect used by the server, the SQL counters to compute authorizations and the identification protocol used during the authentication process on the server.

The last part of this internship was to take all the concepts previously acquired with the aim to implement a strong authentication based on the IEEE 802.1X standard with a bilateral client / server certificate authentication method.

This internship dissertation within the CVO laboratory ESIEA West shows a possible way (problems and given solutions) of understanding and implementing a strong authentication based on a Radius server.

I - Introduction

A) Entreprise d'accueil

L'ESIEA est implantée sur deux sites. On y retrouve le siège social à Paris 5e où les locaux permettent aux professeurs d'enseigner principalement dans les classes de 2e et 3e année du cycle d'ingénieur ainsi que le Mastère SI&S mais également d'autres locaux à Ivry-sur-Seine où la majorité des cours ont lieu. Le campus de Laval, où j'ai effectué mon stage, fût créé en 1993 grâce à des aides de la région et des collectivités locales afin de promouvoir la ville et offrir à la population un choix supplémentaire dans le cycle supérieur des études.

L'ESIEA Laval a pour vocation de fournir les mêmes enseignements qu'à Paris avec une réplication de certaines associations et la venue ponctuelle de certains enseignants de l'ESIEA Paris. Néanmoins, ce qui fait la particularité de ce site, réside dans la création du laboratoire en 2007 « Centre de virologie et de cryptologie opérationnelles » encadré par M. Eric Filiol et M. Richard Rey à partir de 2008. Plus tard, le label du laboratoire prendre le nom de CNS pour "Confiance Numérique et Sécurité".

M. Eric Filiol a été nommé à la tête du CNS après sa carrière militaire ce qui octroie au laboratoire des liens étroits avec les ministères de la Défense, de la Justice ou encore de l'Intérieur. Le laboratoire a ainsi pu encadrer des missions à réaliser pour ses différents ministères en prenant compte de la sécurité à mettre en place. Le CNS a dû être réaménagé à en 2011 afin de correspondre aux critères nécessaires pour la réalisation de sujets à caractères sensibles.
Le laboratoire assure les thèmes suivants :

- Cryptologie
- Analyse et conception de systèmes sténographiques
- Virologie informatique
- Analyse et études techniques du concept de guerre informatique
- Sécurité des environnements embarqués
- Algorithmes des structures complexes et applications
- Mécanismes de Flash Crash

Le laboratoire s'inscrit également dans une démarche de veille technologique afin d'appréhender les menaces d'aujourd'hui et celles de demain.

Depuis Janvier 2015, le CNS propose également à certains étudiants la possibilité d'effectuer un stage en tests d'intrusions avec l'accord des entreprises ciblées.
Des projets sont en développement avec, entre autres :

- Le projet DAVFI (Démonstrateurs d'AntiVirus Français et Internationaux) avec la mise en place de nouveaux modèles de détection et également une analyse en profondeur des fichiers octroyant un meilleur taux de détection des malwares actuels et à venir. Ce projet s'inscrit à partir de février 2015 dans la solution Uhuru Antimalware. Parallèlement, une version libre et gratuite est en cours de développement pour les systèmes Windows et Linux ("OpenDAVFI").

- Le projet Alcasar géré par M. Richard REY et développé par lui-même et un grand nombre de contributeurs. Ce projet a pour but de contrôler les accès à internet des utilisateurs présents sur le réseau via de nombreuses fonctionnalités garantissant trois grands principes de la sécurité : l'authentification, la non-répudiation et la traçabilité.

B) Le projet

Durant mon stage au sein du laboratoire CVO, j'ai dû réaliser plusieurs types de missions. Le stage s'est déroulé en trois parties :

- Une phase de remise à niveau réseau par la pratique de plusieurs travaux dirigés et encadrés par M. Rey
- Puis une rédaction du module pédagogique via une étude et application du protocole Radius et de ses dépendances.
- Enfin, la mise en place d'un mécanisme d'authentification forte dans ALCASAR.

La dernière partie correspond à la majeure partie de mon stage, mais les travaux préliminaires qui m'ont été confiés ont permis de mieux appréhender la problématique suivante :

Comment réaliser une authentification forte dans une solution existante (ALCASAR) ?

J'aborderai les différentes phases suivantes :

- Dans un premier temps, un état de l'art des différents moyens à disposition pour réaliser une authentification.
- Puis, le corps du document décrira le protocole choisi et son implémentation.
- Enfin, je porterai une réflexion sur les problèmes techniques que j'ai pu rencontrer au sein du stage ainsi que les possibles évolutions à mettre en œuvre au sein du projet.

II - État de l'art

L’objectif premier de mon stage a été de mettre en place un système d’authentification des usagers sur le réseau dans le cadre du projet ALCASAR. Tout d’abord, que signifie l’authentification ?

L’authentification est un des grands principes de la sécurité et consiste, dans un système d’information, à contrôler l’accès aux ressources par le biais d’une vérification de l’identité d’un utilisateur. L’authentification existe sous différentes formes. On retrouve une description générale de ce qu’est en mesure de faire un utilisateur pour s’authentifier :

- « Ce que l’utilisateur sait » (exemple : un mot de passe)
- « Ce qu’il sait faire » (exemple : une signature électronique)
- « Ce qu’il est » (exemple : la rétine des yeux, son empreinte digitale, …)
- « Ce qu’il possède (exemple : une carte à puce, une carte RFID, un authentificateur USB, …)

Il existe trois types d’authentification :

- L’authentification simple : un seul élément de vérification
- L’authentification forte : deux éléments ou plus de vérification
- L’authentification unique : une seule authentification pour accéder à plusieurs ressources

Quand l’authentification s’avère-t-elle nécessaire ?

Celle-ci intervient dans les réseaux énumérés ci-dessous :

- Sur un réseau local
- Sur un réseau local étendu : l’Intranet
- Sur l’Internet dans l’optique d’accéder à des ressources confidentielles ou commerciales

Cette liste n’est pas exhaustive, mais rassemble les principaux usages à ce système.

Les principaux protocoles utilisés sur le marché sont les suivants :

A) LDAP

Un serveur LDAP peut être communément appelé « Répertoire du système ». On le considère comme une base de données, essentiellement dans le but de stocker des informations sur diverses entités sur le réseau. En général, le serveur enregistre les données suivantes :

- Les noms d’utilisateurs et mots de passe
- Les droits conférés aux utilisateurs
- Les informations de configuration
- Les périphériques connectés sur le réseau

Lors d’une authentification d’un utilisateur sur le réseau, le serveur LDAP vérifie les informations et attribue les droits correspondants au profil renseigné. Un serveur LDAP fait office de cœur d’authentification du système.

B) Kerberos

L'objectif est de contrôler les accès à des services sur un serveur. Ce serveur d'authentification centralisé est en mesure d'authentifier les clients vers les serveurs mais l'inverse est également possible. Kerberos utilise une méthode de chiffrement qui ne repose pas sur l'utilisation de clefs publiques. Ce protocole est également responsable de reporter aux services sur le réseau l'identité d'un utilisateur connecté. Pour cela, il contrôle le nom d'utilisateur et le mot de passe d'une machine qui vient de s'authentifier et utilise un système de ticket pour l'identification future de l'utilisateur. Ce protocole base donc l'authentification des usagers par l'intermédiaire de tickets.

Le protocole est en mesure de remplacer la phase d'authentification d'un serveur LDAP mais n'exclut pas le besoin de celui-ci. En effet, il est possible de coupler Kerberos à un serveur LDAP qui jouerait le rôle du stockage des informations sur les utilisateurs.
Le système de tickets possède des avantages notables comparé à une authentification par un serveur LDAP. En effet, si le serveur LDAP tombe en panne, les services réseau vont suivre l'arrêt du serveur puisqu'ils nécessitent un système d'authentification. Par ailleurs, le système de tickets proposé par Kerberos permet d'authentifier les utilisateurs sans avoir à consulter d'autres serveurs par l'utilisation de signatures cryptographiques. De plus, envoyer les identifiants sur le réseau peut être risqué notamment via une attaque de « l'homme du milieu ». Kerberos possède une contre-mesure efficace puisque les tickets ne contiennent pas d'informations sensibles, ont une date d'expiration et ne peuvent être utilisés que pour le service demandé.

C) Radius

Ce protocole repose sur le principe des 3A : Authentication (Authentification) Authorization (Autorisation) Accounting (Comptabilisation).

Authentification : Le protocole permet d'effectuer une authentification distante, centraliser les données d'authentification et gérer les connexions des utilisateurs vers des services distants.

Autorisation : Le protocole vérifie les droits attribués à un utilisateur.

Comptabilisation : Le protocole est en mesure de journaliser l'activité d'un utilisateur sur le réseau pour effectuer entre autres une future facturation.

Ce protocole a trouvé sa réputation auprès des fournisseurs d'accès internet qui l'utilisent pour identifier les clients couplé à un serveur LDAP. Il est également employé pour gérer l'authentification sur des points d'accès WIFI. Il repose sur la couche UDP. En effet, Radius est un protocole dit « sans état » ce qui permet de simplifier la mise en œuvre du serveur puisque celui-ci n'attend pas d'accusé de réception.
On trouvera 4 types de paquets permettant d'effectuer une authentification Radius :

- Access-Request : Ce paquet est envoyé par le NAS qui contient les informations d'authentification du client.
- Acess-Accept : Ce paquet est envoyé par le serveur afin d'autoriser la connexion par vérification préalable des informations de l'utilisateur souhaitant se connecter.
- Access-Reject : Le serveur envoie ce paquet lorsqu'il refuse une connexion si l'authentification échoue ou si la connexion doit prendre fin.

- Access-Challenge : Le serveur envoie ce paquet afin de demander une réémission du paquet « Access-Request » ou pour obtenir des informations complémentaires.

D) Diameter

Diameter est un protocole d'authentification forte qui offre, tout comme Radius, le triple A (Authentification = Authentication, Autorisation = Authorization, Comptabilité = Accounting). Il est utilisé sur les réseaux mobiles (3G, 4G, ...) dans l'optique de fournir un accès au réseau et pour répondre à la problématique d'itinérance.

Lorsque la technologie IP a été introduite sur les réseaux de télécommunication, Diameter a été sélectionné pour être la référence des protocoles qui répondent au modèle AAA sur les réseaux fixe et mobile.

Le protocole Diameter (Diamètre) est un jeu de mots faisant hommage au protocole Radius (Rayon). Il a été adopté par des organismes de normalisation tels que 3GPP (3rd Generation Partnership Project) ainsi que l'ETSTI (European Telecommunications Standards Institute) comme étant le standard des fonctionnalités AAA pour la génération future des réseaux "Next-Generation Networks".

Diameter a été conçu sous les AAA afin de prendre en charge davantage de fonctionnalités par rapport à son prédécesseur telles que :

- La politique de contrôle
- Les règles dynamiques
- La qualité de service (QOS)
- L'allocation de la bande passante
- De nouveaux systèmes de tarification

Ce protocole peut également être étendu à de nouvelles applications par l'ajout de nouvelles commandes ou de couples d'attribut-valeur. Diameter est également le seul protocole capable de gérer des fonctionnalités temps réel pour des transactions.

Pour résumer, les atouts apportés par Diameter comparé au protocole Radius sont les suivants :

- Transmission fiable avec l'utilisation de protocoles avec état : TCP ou SCTP (Stream Control Transmission Protocol)
- Garantie de la délivrance des messages Diameter
- Possibilité de mettre en place des agents permettant la prise en charge de proxy(s), redirection(s), relai(s), ...
- Modularité illimitée afin de permettre l'extension du protocole sur diverses applications.

E) Le protocole 802.1X

Ce protocole a été créé par l’IEEE en Juin 2001. Il consiste à authentifier un client sur le

réseau. Pour ce faire, le protocole EAP (Extensible Authentication Protocol) est couplé au serveur Radius. Comment le protocole 802.1X entre en jeu avec Radius ?

Lors de l'initialisation de la connexion, le port est dans l'état « non contrôlé ». Cela signifie que seuls les paquets 802.1X permettant l'authentification sont autorisés. Lorsque l'authentification réussit, le port bascule alors dans l'état « contrôlé ». Ainsi, le trafic provenant du client est accepté et le client accède alors aux ressources partagées sur le réseau.

Le protocole 802.1X embarque plusieurs types d'EAP :

- EAP-TLS : Permet d'effectuer une authentification par certificat au niveau client et serveur
- EAP-TTLS : Permet d'effectuer une authentification par certificat ainsi que mot de passe par la création d'un tunnel sécurisé.
- EAP-MD5 : Permet d'effectuer une authentification par mot de passe.
- PEAP : Permet d'effectuer une authentification avec mot de passe via une encapsulation sécurisée.
- LEAP (propriétaire CISCO) : Même rôle que PEAP.

F) Les certificats X.509

Radius permet d'authentifier ses utilisateurs via un certificat. Ci-dessous, un tableau représentant les différentes implémentations d'un certificat :

	Avantages	Inconvénients
Certificat X.509 dans une carte a puce	+ multi-usage + robustesse de la méthode d'authentification + attitude similaire à la possession d'une carte bancaire	- vol, perte ou oubli de la carte a puce
Biométrie associée a un certificat X.509 dans un token USB	+ multi-usage + attitude similaire à la possession de clés (maison, voiture)	- technologie encore immature - vol, perte ou oubli du token USB - coût élevé
Biométrie associée a un certificat X.509 dans une carte a puce	+ multi-usage + attitude similaire à la possession d'une carte bancaire	- technologie encore immature - vol, perte ou oubli de la carte a puce - coût élevé
Certificat X.509 dans le navigateur de l'ordinateur	+ multi-usage + robustesse de la méthode d'authentification + confort d'utilisation pour l'utilisateur	- vol ou utilisation frauduleuse de l'ordinateur et copie de la clé privée associée au certificat
Certificat X.509 dans un token USB	+ multi-usage + robustesse de la méthode d'authentification + attitude similaire à la possession de clés (maison, voiture)	- vol, perte ou oubli du token

Figure 1: Avantages et inconvénients de l'implémentation des certificats X.509 (source http://repo.hackerzvoice.net)

Partie 1 : Radius

A) Historique

RADIUS a été conçu dans le but initial de contrôler l'authentification à distance. Ce besoin est né de la société « Merit Network » qui souhaitait identifier ses utilisateurs via la liaison téléphonique afin de fournir un accès distant à son réseau informatique. L'organisation à but non lucratif « Merit Network » a publié en 1991 une demande d'information ou « Request for Information » (RFI) qui spécifiait les fondements du protocole[3] RADIUS.

Après quelques mois, la société « Livingston Enterprises » répondit à cette demande par une description d'un serveur RADIUS. La proposition fut retenue par « Merit Network » qui donna le contrat à « Livingstone Enterprises ». Le protocole RADIUS « Remote Authentication Dial-in User Service » traduit littéralement par Authentification à distance composée par le service utilisateur est publié par l'IETF[4] en Janvier 1997 dans la RFC[5] 2058 et la RFC 2059.

Cette même année, la société « Livingstone Enterprises » leader dans l'accès à distance des réseaux informatiques, fut rachetée par Alcatel-Lucent. Le standard RADIUS a subi plusieurs évolutions au fil des années et à l'heure d'aujourd'hui, il est de facto le protocole le plus célèbre pour l'authentification des postes de travail, des terminaux mobiles, des équipements réseaux, etc...
La description actualisée du standard se trouve désormais dans la RFC 2865 et la RFC 2866 datant de Juin 2000.

B) Description du protocole

Le protocole RADIUS a évolué au fil des années. Il intègre aujourd'hui de nombreuses fonctionnalités. Auparavant ce protocole répondait uniquement aux besoins d'authentification et de traçabilité. Il s'est calqué sur le modèle du protocole AAA[6] qui permet de réaliser trois fondamentaux de l'accès à une ressource informatique :

- « Authentication » (Authentification) : c'est la fonction principale de sécurité qui consiste à prouver qu'une identité appartient bien à celui qui la présente. Elle peut être réalisé en comparant des « credentiels » (nom d'utilisateur/mot de passe), certificat numérique,etc… ;

- « Authorization » (Autorisation) : c'est la capacité à accéder, une fois l'authentification validée, à un service ou des ressources du système d 'information ;

- « Accounting » (Compatibilité) : c'est « journaliser » les accès, les temps de session, les ressources consommées, etc... afin de garantir la traçabilité des informations.

Le principe des échanges du protocole RADIUS est basé sur des requêtes/réponses (voir figure 2) avec des clients RADIUS aussi appelés NAS[7] ou plus simplement service d'authentification.

3 Définition de protocole en communication
4 IETF : Internet Engineering Task Force, groupe informel qui publie la plupart des nouveaux standards internet
5 RFC : Request For Comments, document publié par l'IETF
6 AAA : Authentication Authorization Accounting, Authentification Autorisation Comptabilité
7 NAS : Network Access Server, serveur d'accès au réseau

1. Qui peut être un client RADIUS ?

Un client RADIUS est un équipement ou bien une solution logicielle qui est capable d'envoyer des demandes de connexion (requêtes) et des messages à un serveur RADIUS. Il peut interpréter les réponses du serveur RADIUS et, de ce fait, valider ou non une authentification et/ou obtenir des autorisations. Par la suite, ces informations sont transmises au poste de travail souhaitant accéder à la ressource informatique.

Voici quelques exemples de client RADIUS :

- Serveur de connexion VPN[8], donne l'accès à distance aux ressources informatiques d'une organisation ;
- Points d'accès sans fil utilisant la technologie Wi-Fi[9], fournit un accès au réseau local d'une entreprise ;
- Commutateurs, fournit également l'accès au réseau local ;
- Proxy RADIUS, transfère les requêtes/réponses d'un client RADIUS vers un serveur RADIUS distant.

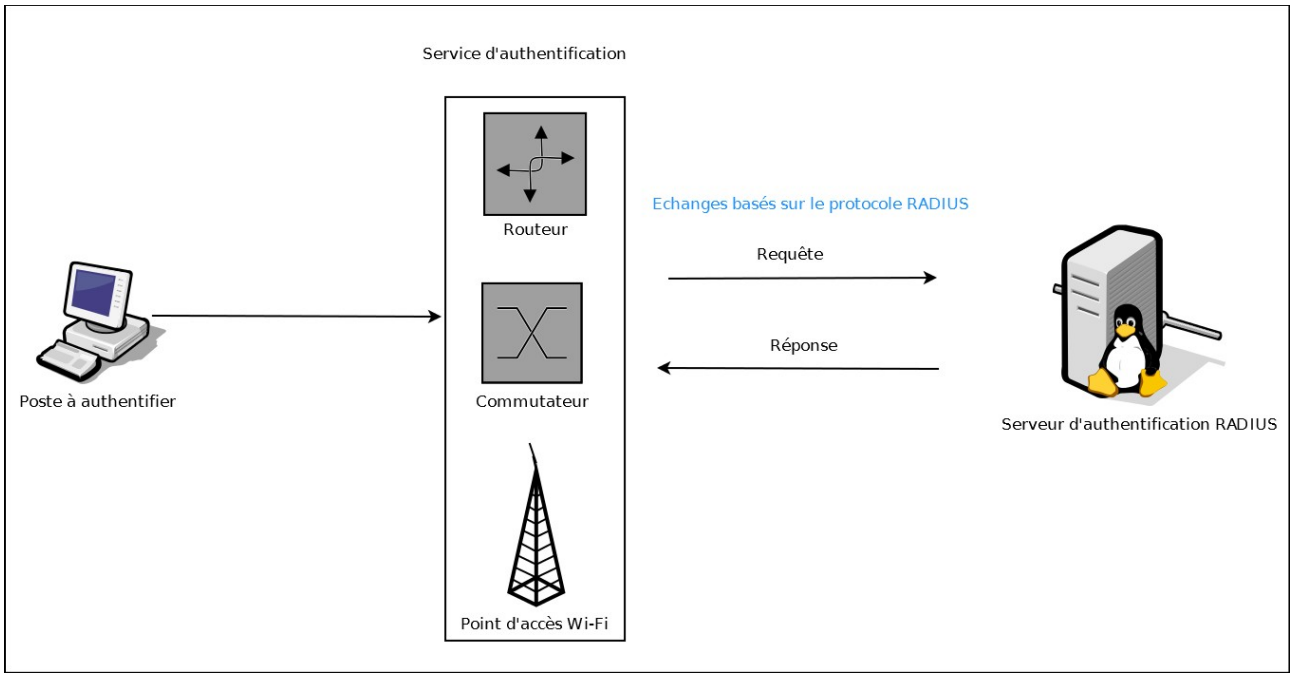

Figure 2 : Principe des échanges du protocole RADIUS

8 VPN : Virtual Private Network
9 Wi-Fi : Wireless Fidelity

2. Le protocole RADIUS dans le modèle OSI[10]

Le modèle OSI, développé par l'ISO[11], est un standard commun afin de normaliser les communications entre ordinateurs et plus généralement entre les systèmes informatiques. Il définit une architecture décomposée en plusieurs couches, où chacune a une fonctionnalité spécifique. Cela a permis d'établir un accord entre les différents constructeurs d'équipements informatiques pour ainsi homogénéiser l'interconnexion des systèmes hétérogènes.
Les couches du modèle OSI sont les suivantes :

- Application
- Présentation
- Session
- Transport
- Réseau
- Liaison
- Physique

Le protocole RADIUS se situe au niveau des couches hautes, dans la couche applicative du modèle OSI. Il se place au-dessus de la couche transport. Les données du protocole RADIUS sont acheminées par des segments du protocole UDP[12] et encapsulées dans des paquets IP[13] (voir figure 3 et 4).

Les ports d'écoute UDP utilisés pour accéder aux services proposés par le protocole RADIUS sont les suivants :

- 1812, reçoit les requêtes d'authentification et d'autorisations ;
- 1813, reçoit les requêtes d' « accounting » (comptabilité).

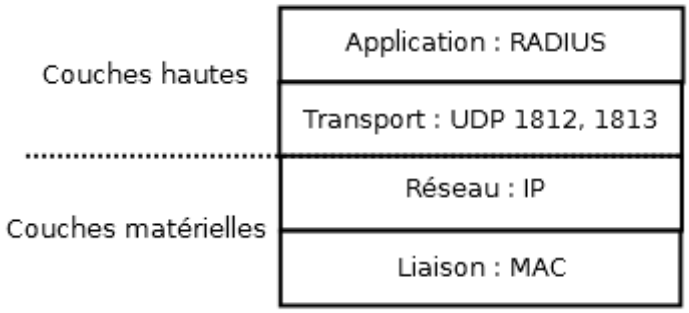

Figure 3 : Le protocole RADIUS au sein du modèle OSI

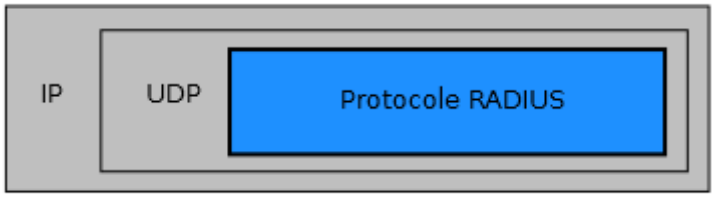

Figure 4 : Encapsulation du protocole RADIUS

10 OSI : Open Systems Interconnection
11 ISO : International Standard Organization
12 UDP : User Datagram Protocol, c'est de protocole de transmission des données
13 IP : Internet Protocol, fournit un service d'adressage permettant d'identifier les équipements informatique

3. Format des paquets RADIUS

Le protocole RADIUS utilise un format de paquet bien défini pour réaliser les transactions d'authentification, d'autorisation et de comptabilité. Le modèle des données RADIUS contient les champs ci-contre:

1. Code ;
2. ID ;
3. Longueur ;
4. « Authentificateur » ;
5. Attributs et Valeurs.

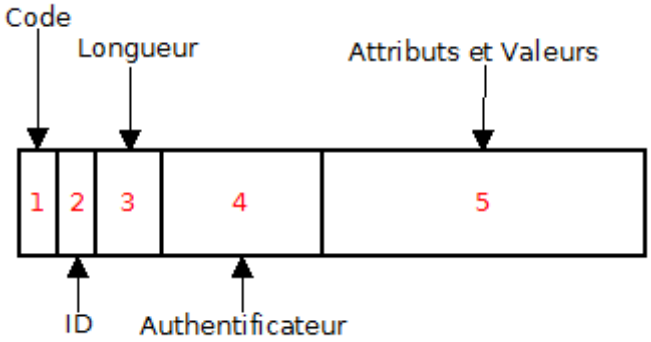

Figure 5 : Format des trames RADIUS

Ces champs accueillent des types de données décrites dans la RFC du protocole RADIUS.

Code
Ce champ identifie le type du paquet RADIUS. En effet, le protocole définit 255 types de paquets que l'on peut trouver dans la RFC 3575. Cependant, je vais citer uniquement les types que j'ai été amené à utiliser.
Il s'agit de :

- Access-Request
- Access-Challenge
- Access-Accept
- Access-Reject
- Accounting-Request
- Accounting-Response

Je détaillerai plus loin la fonction de ces différents types de paquets et leurs utilisations dans le protocole RADIUS.

ID
Ce champ permet de mettre en relation un identifiant au client RADIUS avec le couple requêtes/réponses.

Longueur
Ce champ contient la longueur totale des données RADIUS.

Authentificateur
Ce champ permet de vérifier l'intégrité des paquets envoyés par le serveur RADIUS. Il est calculé de manière aléatoire à partir d'un secret partagé (mot de passe connu par le serveur et le client) entre le client et le serveur. Ainsi, le client RADIUS peut s'assurer que la réponse lui vient bien du serveur RADIUS et non d'un usurpateur.

Attributs et valeurs
Ce champ contient **la charge utile** du protocole RADIUS. Il est de **longueur variable** en fonction des couples d'attributs/valeurs envoyés par le client RADIUS en requête ou par le serveur RADIUS en réponse.

Après cette description générale du protocole RADIUS, je vais exposer l'utilité du champ « attributs et valeurs ».

4. Les attributs RADIUS

Les informations contenues dans les requêtes/réponses d'authentification, d'autorisation et de comptabilités sont transportées par les couples attributs/valeurs du protocole. Le champ attribut et valeur joue un rôle primordial dans les échanges RADIUS. En effet, c'est lui qui va véhiculer les informations relatives à l'authentification, l'autorisation ou à la comptabilité. Chaque attribut est caractérisé par un numéro d'attribut, une longueur ainsi qu'une valeur (voir figure 6). Dans le jargon RADIUS, un couple attribut/valeur est appelé AVP "Attributes Value-Pair".
Un attribut peut prendre les valeurs suivantes :

- adresse IP ;
- date ;
- chaîne de caractère (par exemple un mot de passe) ;
- entier (un numéro de port) ;
- valeur binaire ou hexadécimale

.

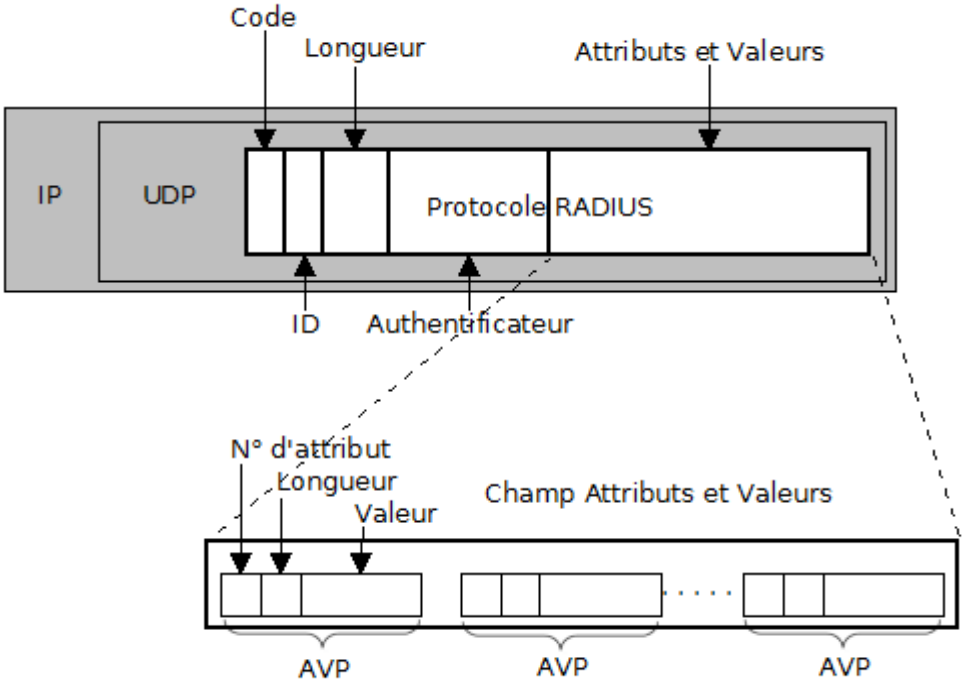

Figure 6 : Champ « attributs et valeurs » du protocole RADIUS

Dans les paquets RADIUS, le nom d'un attribut n'apparaît jamais. Seul le numéro de l'attribut est présent. La correspondance entre le nom de l'attribut et son numéro est faite grâce à un dictionnaire d'attributs implémenté sur le client et sur le serveur.

Les dictionnaires d'attributs contiennent la concordance entre les attributs standards et leurs numéros respectifs. Mais il peut également avoir des dictionnaires d' « attributs constructeurs ». J'aborderai la notion de ces attributs particuliers plus loin.

Le protocole RADIUS compte un grand nombre d'attributs (plus d'une centaine). La liste complète des attributs est consultable sur le site internet de l'organisation IANA[14] au lien suivant :

http://www.iana.org/assignments/radius-types/radius-types.xhtml#radius-types-2

Je vais définir certains de ces attributs afin de mieux appréhender leur utilisation au sein de l'authentification RADIUS.

"User-Name"

Cet attribut est envoyé par le client RADIUS lors de la demande d'authentification. Il peut correspondre à un identifiant utilisateur associé à un mot de passe. Dans le cas de l'authentification d'une machine, cet attribut peut prendre la valeur de l'adresse physique de celle-ci plus communément appelée adresse MAC[15]. Lors de l'utilisation d'un certificat électronique, la valeur reçue par l'attribut User-Name est le nom du porteur du certificat.

"User-Password"

Cet attribut contient le mot de passe associé à l'attribut User-Name. Il est également envoyé par le client RADIUS.

"NAS-IP-Address"

Cet attribut est renseigné par le client RADIUS (NAS). Il contient son adresse IP. Cet attribut peut imposer une restriction. Par exemple, un poste de travail ne pourra s'authentifier que s'il se connecte à partir d'un NAS possédant une adresse IP spécifique.

"Nas-Port"

Cet attribut permet d'ajouter une condition supplémentaire pour l'authentification. Prenons l'exemple d'un équipement réseau de type commutateur. Ainsi, un poste de travail est autorisé à s'authentifier uniquement s'il est connecté sur un port précis du commutateur. L'attribut « Nas-Port » donne le numéro de port sur lequel est connecté un poste utilisateur. Il est envoyé par le NAS lors de la demande d'authentification.

Dans la technologie sans fil, cet attribut perd son sens, car un point d'accès génère un port "virtuel" au moment de l'association du poste de travail à celui-ci.

"Called-Station-Id"

Cet attribut contient l'adresse MAC du NAS qui demande l'authentification au serveur RADIUS

"Calling-Station-Id"

Cet attribut contient l'adresse MAC du poste de travail souhaitant s'authentifier.

14 IANA : Internet Assigned Numbers Authority, Autorité qui gère la numérotation requise par les protocoles de communications

15 MAC : Media Access Control, adresse physique unique stockée dans toutes les cartes réseaux

Comme nous l'avons vu précédemment, lorsqu'un client RADIUS soumet une requête d'authentification, il renseigne différents attributs. Ces attributs sont définis par le standard RADIUS. Toutefois, il existe des attributs supplémentaires implémentés par les constructeurs d'équipement réseau. Ceux-ci offrent des fonctionnalités optionnelles afin de réaliser des opérations dédiées sur les équipements par le biais du protocole RADIUS.

5. Les attributs constructeurs

L'ajout d'attribut(s) constructeur permet aux administrateurs réseau de tirer un meilleur profit des capacités de leurs équipements réseau. En effet, ils offrent aux fournisseurs la possibilité de spécifier 255 attributs supplémentaires pour leurs matériels. L'implémentation d'un dictionnaire d'attributs constructeur permet le support de ceux-ci sur le serveur RADIUS. Bien évidemment, seuls les équipements constructeurs pourront interpréter la valeur d'un attribut propriétaire.

Cet attribut est appelé « Vendor Specific Attributes » (VSA) dans la terminologie RADIUS. Il est identifié par le numéro d'attribut 26. L'attribut VSA est similaire au couple attribut/valeur (AVP) RADIUS (voir figure 7). Chaque attribut VSA est constitué des champs suivants :

- N° d'attribut : c'est le numéro 26 qui est défini pour l'attribut VSA dans le standard RADIUS ;
- Longueur : contient la longueur totale de l'attribut ;
- « Vendor-id » : contient le code international du constructeur. Chaque constructeur possède un code qui lui est propre. On trouve la liste des codes constructeur dans la RFC 1700 ;
- N° d'attribut constructeur : contient le numéro d'attribut spécifié par le constructeur ;
- Longueur VSA : contient la longueur du champ VSA ;
- Valeur de l'attribut constructeur : contient la valeur de l'attribut VSA.

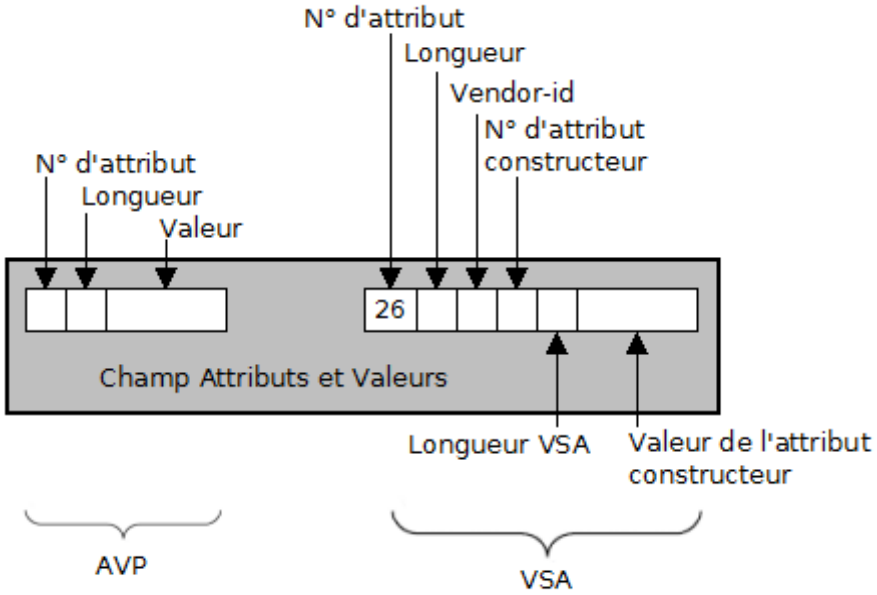

Figure 7 : Format des attributs VSA

6. Les types de paquets RADIUS

Comme mentionné dans la sous partie 3, chaque échange RADIUS est caractérisé par un paquet associé à un code. La figure ci-dessous présente quelques utilisations des types Radius :

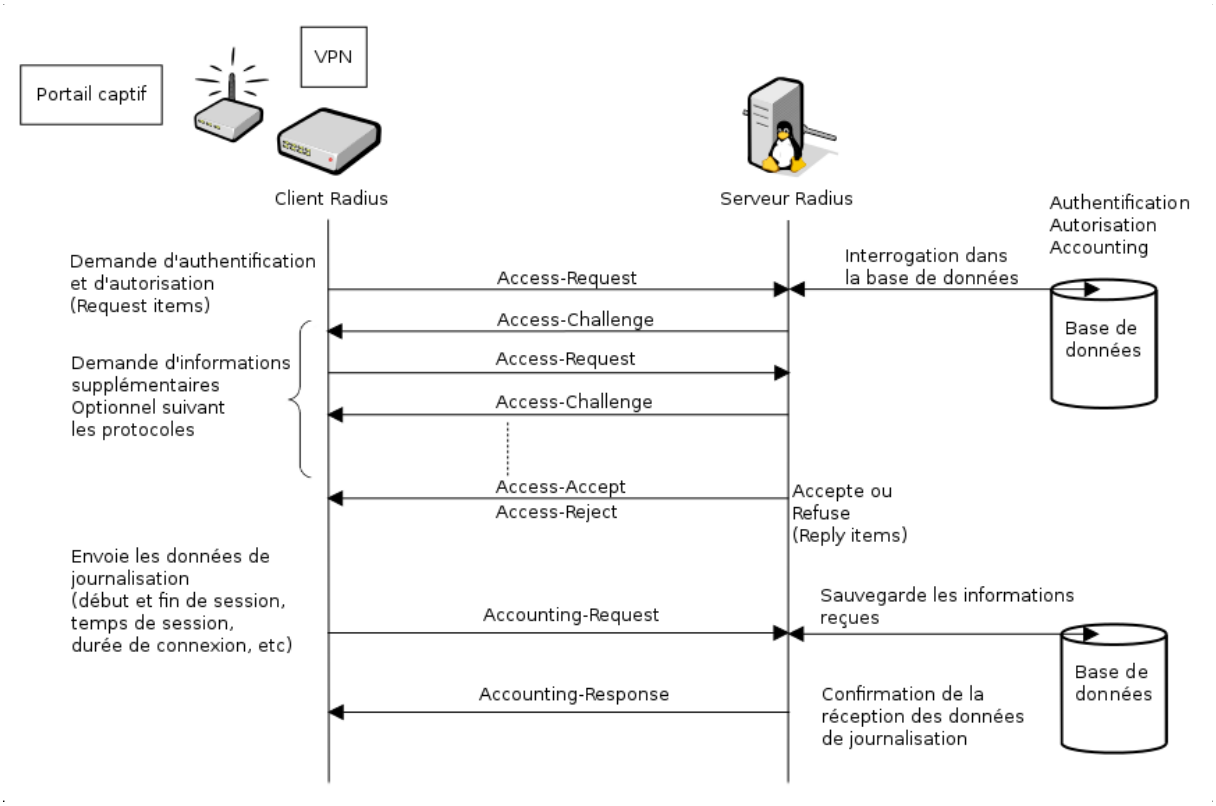

Figure 8: Les échanges au sein du protocole RADIUS

Le type « Access-Request »
Lors d'une authentification RADIUS, le dialogue est toujours initié par le NAS. Tout d'abord, il collecte les informations auprès du poste de travail souhaitant accéder à la ressource informatique, puis, il transmet ces informations sous forme de couples attributs/valeurs par le biais du protocole RADIUS vers le serveur RADIUS. Enfin, la demande d'authentification est alors acheminée par un paquet de type « Access-Request ».

Le type « Access-Challenge »
Une fois la demande d'authentification reçue par le serveur RADIUS, ce dernier peut demander des informations supplémentaires au NAS. Il y a alors émission d'un paquet de type « Access-Challenge » de la part du serveur vers le NAS. En conséquence, le serveur attendra la réception d'un nouveau paquet de type « Access-Request » de la part du client RADIUS afin d'obtenir les informations demandées.

Le type « Access-Accept »
Lorsque l'ensemble des données d'authentification émises par le client RADIUS au moyen de paquet « Access-Request » ont été validées par le serveur RADIUS. Celui-ci notifie le client du succès de l'authentification en envoyant un paquet de type « Access-Accept ». Ce paquet peut contenir des attributs constructeurs destinés au client Radius. Il peut également comporter des autorisations (attributs de retour) définissant des règles de sécurité à appliquer à la session du poste de travail authentifié.

Le type « Access-Reject »
Quand un ou plusieurs éléments d'authentification transmis par le client RADIUS sont incorrects ou qu'il ne respectent pas la politique de sécurité informatique, le serveur répond par un paquet de type « Access-Reject ».

Le type « Accounting-Request »
Une fois la phase d'authentification réussie, le client RADIUS envoie de manière périodique un paquet de type « Accounting-Request » au serveur. Ce paquet fait l'inventaire des traces enregistrées du poste authentifié telles que le temps de session, l'adresse IP attribuée, le nombre de paquets échangés sur le réseau, etc.

Le type « Accouting-Response »
Ce paquet est une réponse du serveur à la requête de traçabilité émise par le client. Il permet d'indiquer la réussite ou non de l'enregistrement des données.

Que souhaite-t-on authentifier et comment ?

Dans un système d'information, il y a des machines (ou équipements) et des utilisateurs.
Dans le cas de l'authentification des machines, une fois leur identification validée, elles gardent les mêmes droits d'accès au sein du système. Tout utilisateur sera donc dépendant des droits attribués à la machine.

En revanche, si on authentifie des utilisateurs et non plus des machines, alors chaque utilisateur bénéficiera de droits d'accès personnalisés. Les autorisations attribuées à la machine authentifiée dépendent de l'usager qui l'utilise.

Les données d'authentification à présenter au serveur RADIUS dépendent du niveau de sécurité que l'on souhaite adopter.

Partie 2 : Radius dans Alcasar

Mon travail sur l'authentification forte s'est axé sur principalement sur quatre briques de la solution ALCASAR. La passerelle d'interception Coova-Chilli, le serveur FreeRadius et la base de données SQL MariaDB.

Bien que ce soit une petite partie des fonctionnalités de ALCASAR cela reste l'un des principaux piliers de sécurité qu'offre cette application open-source.

La figure 9 place le contexte dans lequel j'ai travaillé dans le projet. Je détaillerai dans un premier temps les éléments du cadre A puis ceux du cadre B.

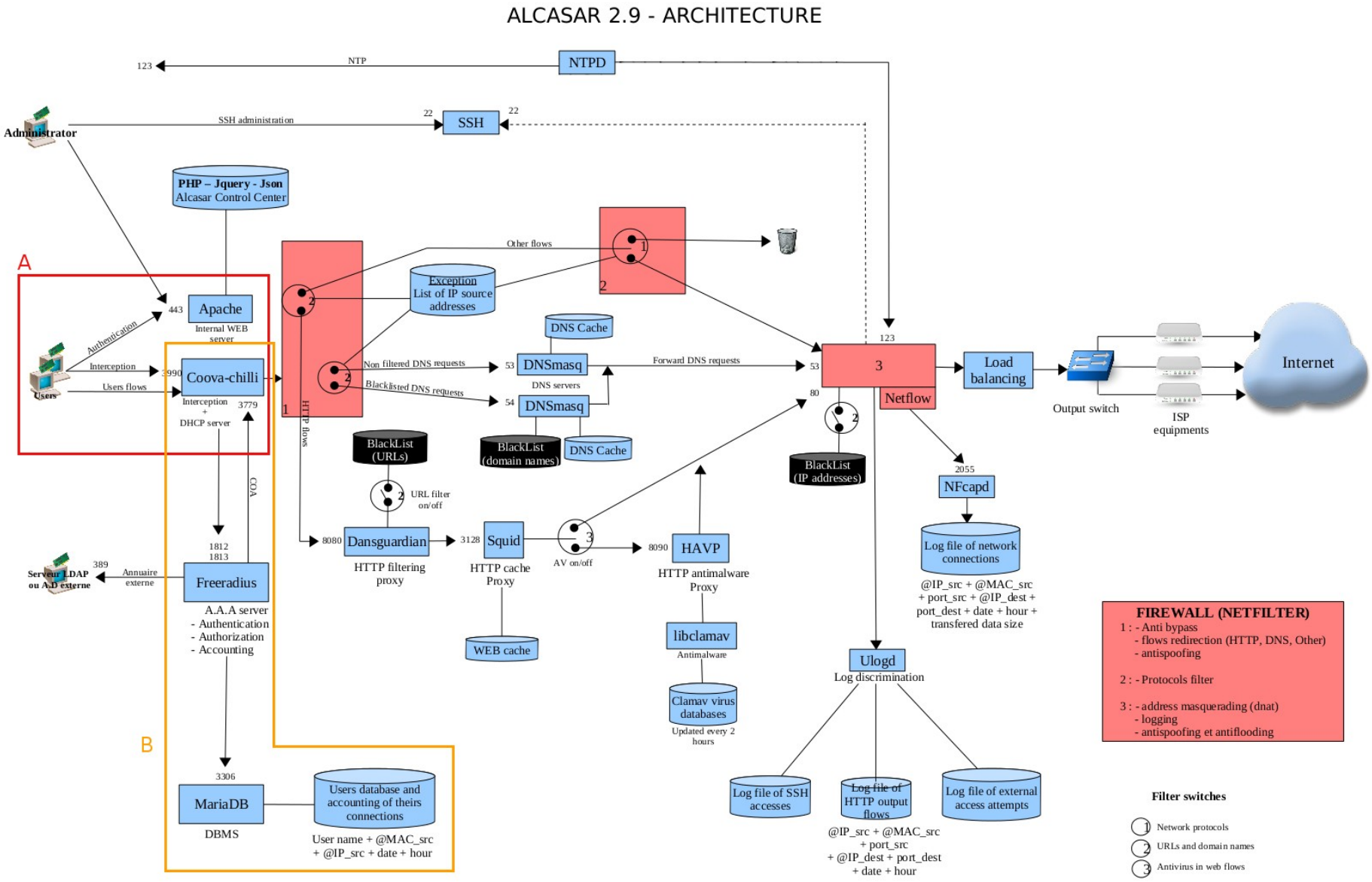

Figure 9: Schéma de principe ALCASAR

Après avoir effectué un travail de recherche sur FreeRADIUS et compris les principaux mécanismes d'authentification qu'il embarque, j'ai commencé la prise en main du serveur FreeRADIUS de Alcasar. Dans un premier temps, il a fallu analyser les échanges entre les différentes composantes rattachées au serveur.

Coova-chilli : C'est une passerelle d'interception qui bloque tout dispositif, non authentifié, du réseau local souhaitant accéder à l'Internet. Elle fait transiter tout le flux réseau grâce à une interface virtuelle TUN0[16]. Par ailleurs, cette composante intègre un mini serveur http qui écoute sur le port 80 afin de rediriger un équipement du réseau de consultation vers la page d'interception "intercept.php" fournie par le serveur Apache.

Elle traite également les données d'authentification envoyées par l'équipement souhaitant s'authentifier sur le réseau. De plus, elle est client RADIUS. Ainsi, c'est elle qui va générer des requêtes d'authentification destinées au serveur FreeRadius. Enfin, elle s'occupe aussi de collecter des données de comptabilité (durée de connexion, volume de données échangé, etc) et les transmet au serveur. Elle peut déconnecter un usager.

À mon arrivée, Coova permettait de réaliser une authentification UAM[17] à travers un navigateur Web. Mon objectif était de trouver une solution pour que Coova puisse également proposer un mécanisme d'authentification forte basé sur le standard IEEE 802.1X. J'ai donc étudié la procédure d'authentification qui était en place.

Dans un premier temps, je parlerai de l'authentification UAM puis dans un second temps,j'aborderai la solution IEEE 802.1X greffée au projet. Nous verrons donc comment le standard 802.1X réalisant une authentification forte peut cohabiter avec le mécanisme d'authentification universelle.

A) L'authentification UAM

La force de ce mécanisme est qu'il peut être déployé dans un milieu où les terminaux et les systèmes sont hétérogènes. En effet, il suffit que les dispositifs souhaitant s’authentifier disposent d'un navigateur Web.
Par ailleurs, l'équipe de ALCASAR a développé une solution sûre pour l'acheminement des données d'authentification depuis le navigateur de l'usager jusqu'à l'application.

1. Le mécanisme

Nous pouvons voir sur la figure 10 un usager s'authentifiant sur le réseau avec les différentes phases d'échanges opérées entre le navigateur Web de l'usager, Coova-Chilli, le serveur Web Apache et Radius.

Dans un premier temps, l'utilisateur souhaite accéder à une URL (exemple : http://google.fr). Cette requête est interceptée par le service Coova-Chilli qui vérifie si l'adresse IP de l'usager est autorisée à accéder à l'Internet.

16 Network TUNnel : c'est un tunnel réseau qui opère sur la couche 3 (gère les paquets IP)
17 UAM : Universal Access Method, méthode d'accès universelle

Dans le cas où une authentification est nécessaire :

- Coova-Chilli retourne une requête HTTP[18] de redirection de service au navigateur vers le serveur Apache (https://@IP_ALCASAR/intercept.php). L'URL de redirection contient le nom de domaine du serveur Web de ALCASAR. L'URI contient une demande de la ressource "intercept.php" avec en argument un challenge arbitraire ainsi que l'URL initialement demandée par l'utilisateur .

- Au travers d'une communication sécurisée en HTTPS[19], le navigateur demande la page "intercept.php". Cette page intègre un formulaire d'authentification, affiche à l'utilisateur le résultat des tentatives d'authentification, gère la redirection vers Coova-Chilli, etc ...

- L'utilisateur est invité à saisir son identifiant et mot de passe. Ces informations sont transmises, à travers le tunnel TLS, au serveur Apache.

- Ce dernier effectue des opérations cryptographiques sur le mot de passe puis renvoie le nouveau contenu au client Web avec, à nouveau une requête de redirection HTTP vers la passerelle d'interception Coova-Chilli.

- A la réception des données d'authentification, Coova-Chilli envoie une demande d'authentification au serveur FreeRadius.

- Une fois que la réponse de RADIUS parvient, la passerelle envoie une dernière requête de redirection contenant le résultat "authentif" vers le navigateur de l'utilisateur.

- Enfin, les informations sont acheminées à Apache qui interprète la réponse de Coova-Chilli et restitue le résultat sous forme visuelle pour l'utilisateur.

18 HTTP : HyperText Transfer Protocol
19 HTTPS : HyperText Transfer Protocol Secure

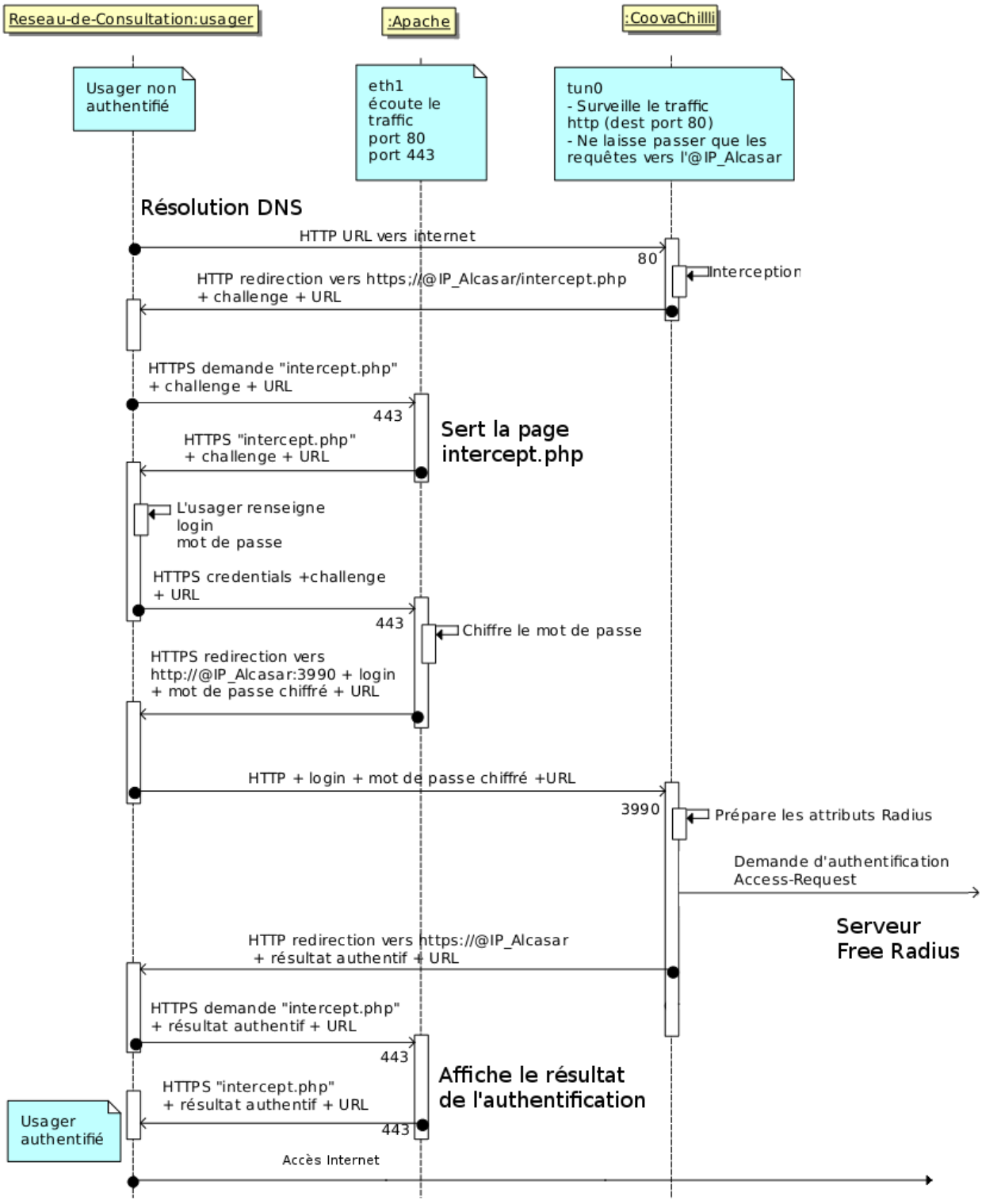

Figure 10: Schéma des échanges interception et authentification

Après l'explication des messages échangés lors de l'authentification d'un utilisateur sur le contrôleur d'accès au réseau, je vais parler des outils cryptographiques utilisés pour protéger l’acheminement du mot de passe de l'utilisateur jusqu'à la passerelle Coova-Chilli.

2. Sécurisation du mot de passe

La transmission du mot de passe se décompose en 10 étapes (cf. Figure 11) :

1. Lors de l'interception d'un utilisateur non autorisé sur le réseau, la passerelle Coova-Chilli génère une chaîne aléatoire que l'on appelle un "challenge" cryptographique. Ce "challenge" est envoyé au navigateur de l'utilisateur. Il interviendra dans la suite pour chiffrer le mot de passe.
2. L'utilisateur saisit son identifiant et son mot de passe. Le navigateur envoie ces informations au serveur Apache accompagné du "challenge" émis par Coova-Chilli à travers un canal sécurisé (SSL/TLS).
3. Le serveur Apache transmet les informations reçues au script "intercept.php". Le script complète le mot de passe par des "0" ("padding") pour obtenir une longueur de trente-deux caractères.
4. En parallèle, il convertit le "challenge" Cova-Chilli en base hexadécimale.
5. A l'installation d'ALCASAR, un secret partagé a été mis en place entre Apache et Coova-Chilli. Un nouveau challenge (newchal) est créé par concaténation entre le secret partagé (uamsecret) et le challenge "Coova-Chilli" (hexchal).
6. Puis, une empreinte du nouveau challenge est calculée en utilisant l'algorithme de hachage MD5.
7. Une opération logique de type OU EXCLUSIF est réalisée entre l'empreinte MD5 du nouveau challenge et le mot de passe obtenu à l'étape 3. Le masque appliqué au mot de passe permet de garantir un bon niveau de sécurité d'un point de vue cryptographique.
8. Le mot de passe chiffré est transmis au navigateur de l'utilisateur.
9. Le page "intercept.php" de l'usager transmet à son tour le mot de passe précédemment chiffré par Apache à la passerelle Coova-Chilli.
10. La passerelle applique le même masque calculé par Apache sur le mot de passe reçu pour obtenir le mot de passe en clair. Enfin, elle réalise une requête d'authentification au serveur Radius.

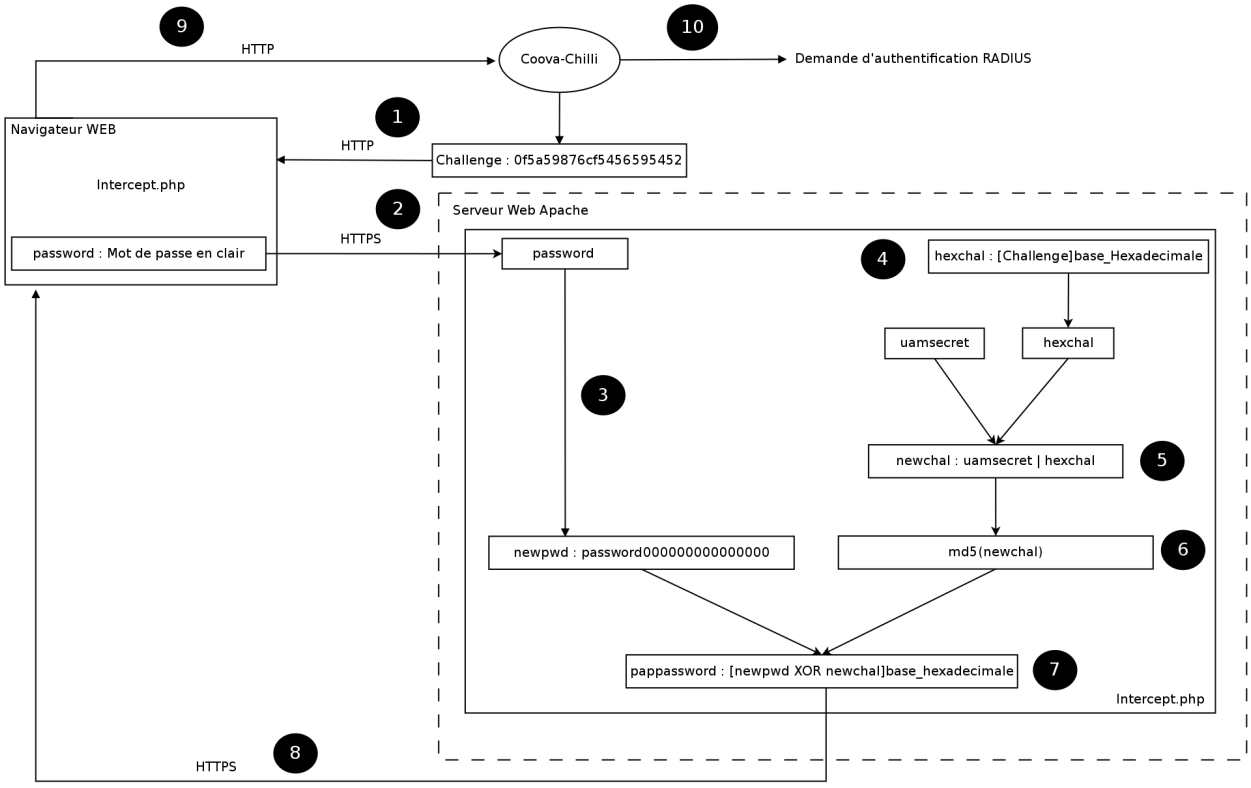

Figure 11: Schéma de principe de l'acheminement du mot de passe utilisateur

Algorithme de chiffrement du mot de passe

La fonction utilisée pour chiffrer le mot de passe utilisateur est un OU EXCLUSIF (cf. Figure 12), souvent appelé XOR (eXclusive OR). Elle prend en entrée deux opérandes, le mot de passe en clair de l'utilisateur et le challenge_Coova-Chilli. Le challenge est une clé symétrique aléatoire partagée par les services Coova-chilli et Apache. Elle n'est utilisée qu'une seule fois et sert à chiffrer et à déchiffrer. On parle alors de masque jetable ou encore de "one-time pad".

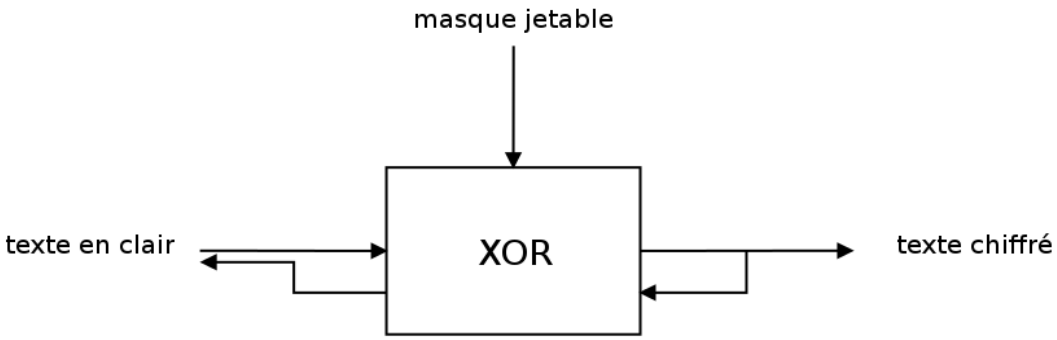

Figure 12: Algorithme de chiffrement utilisé pour la transmission du mot de passe lors d'une authentification

Côté Apache :

- challenge_Coova_Apache = md5 [uamsecret | challenge_Coova]
- mot_de_passe_en_clair **XOR** challenge_Coova_Apache = mot_de_passe_chiffré

Côté Coova-Chilli :

- challenge_Coova_Apache = md5 [uamsecret | challenge_Coova]
- challenge_Coova_Apache **XOR** mot_de_passe_en_chiffré = mot_de_passe_en_clair

Cet algorithme paraît simpliste et facile à casser car il suffit d'avoir la connaissance du challenge_Coova_Apache pour retrouver le mot de passe en clair.

Cependant, le challenge généré par Coova est différent à chaque nouvelle interception. Ce qui implique que le mot de passe chiffré d'un même utilisateur ne sera pas identique car le challenge Coova_Apache sera différent.

De plus, l'empreinte MD5 de ce challenge qui sert de clé de chiffrement est difficilement prédictible car elle nécessite la connaissance du secret commun entre Coova-Chilli et Apache. Une personne mal intentionnée souhaitant obtenir le mot de passe d'un utilisateur devra tenter de deviner le secret commun afin de produire le masque adéquat et déchiffrer le mot de passe.

La clé partagée entre Coova-Chilli et Apache est composée de huit caractères alphanumériques (lettres de l'alphabet latin de A à Z majuscules et minuscules, chiffres de 0 à 9).

Ci-dessous le code contenu dans le script d'installation (alcasar.sh) de ALCASAR qui génère le secret partagé :

```
secretuam=`cat /dev/urandom | tr -dc [:alnum:] | head -c8`
echo -n "Shared secret between the script 'intercept.php' and coova-chilli : " >> $PASSWD_FILE
echo "$secretuam" >> $PASSWD_FILE
```

Figure 13: Script générant le secret partagé entre Coova-Chilli et Apache

Par conséquent, il y a soixante-deux caractères possibles pour chaque caractère composant le secret partagé. Cela offre 62^8 possibilités soit 218 x 10^{12} clés possibles et donc autant de mots de passe à tester.

Nous avons vu dans cette première partie de l'authentification UAM les interactions de ALCASAR avec l'utilisateur pour récupérer les éléments d'authentifications. Je vais maintenant aborder la partie qui concerne le dialogue entre Coova-Chilli, FreeRadius et la base de données MariaDB (cf cadre B de la figure 9).

Nous verrons comment la passerelle Coova-Chilli réalise une authentification Radius auprès du service d'authentification FreeRadius. Je détaillerai également de quelle manière le serveur identifie un utilisateur et ainsi de quelle façon il calcule ses autorisations. Nous verrons aussi par quels procédés les informations de traçabilités sont stockées.

3. Demande d'authentification :

La Figure 14 décrit les échanges Radius de manière générale entre un client Radius et un serveur Radius. Le client Radius peut être représenté par Coova-Chilli.

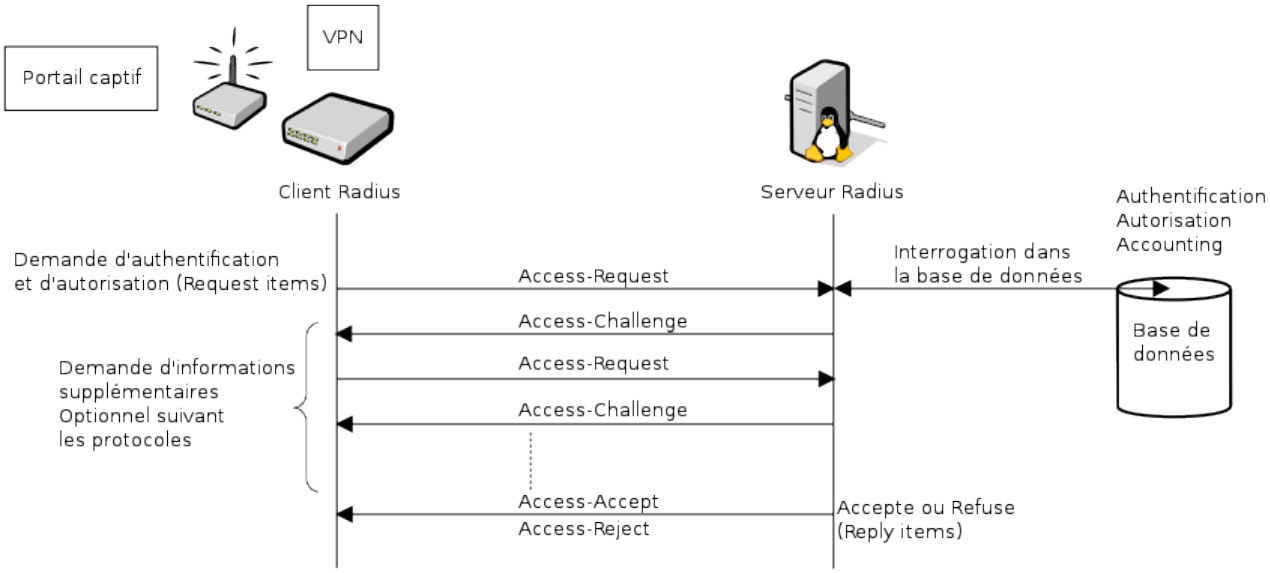

Figure 14: Échanges Radius lors d'une demande d'authentification

On retrouve ci-dessous les messages réseau échangés entre Coova-Chilli et FreeRadius (cf. Figure 15). On remarque que la trame n°1 est une demande d'authentification qui fait transiter le nom d'utilisateur et le mot de passe chiffré à l'aide d'un challenge et d'un secret partagé entre les 2 parties.

Ces messages ont été capturés sur la boucle locale (l0) de Alcasar. L'analyseur de trames réseau Wireshark a été utilisé pour visualiser les échanges.

La trame n°2 indique que l'utilisateur est autorisé à accéder au réseau.

La trame n°3 est une requête de comptabilité (voir détails Figure 16) qui contient les données de départ de la session de l'utilisateur.

La trame n°4 décrit la confirmation de l'enregistrement des données de session par le serveur d'authentification.

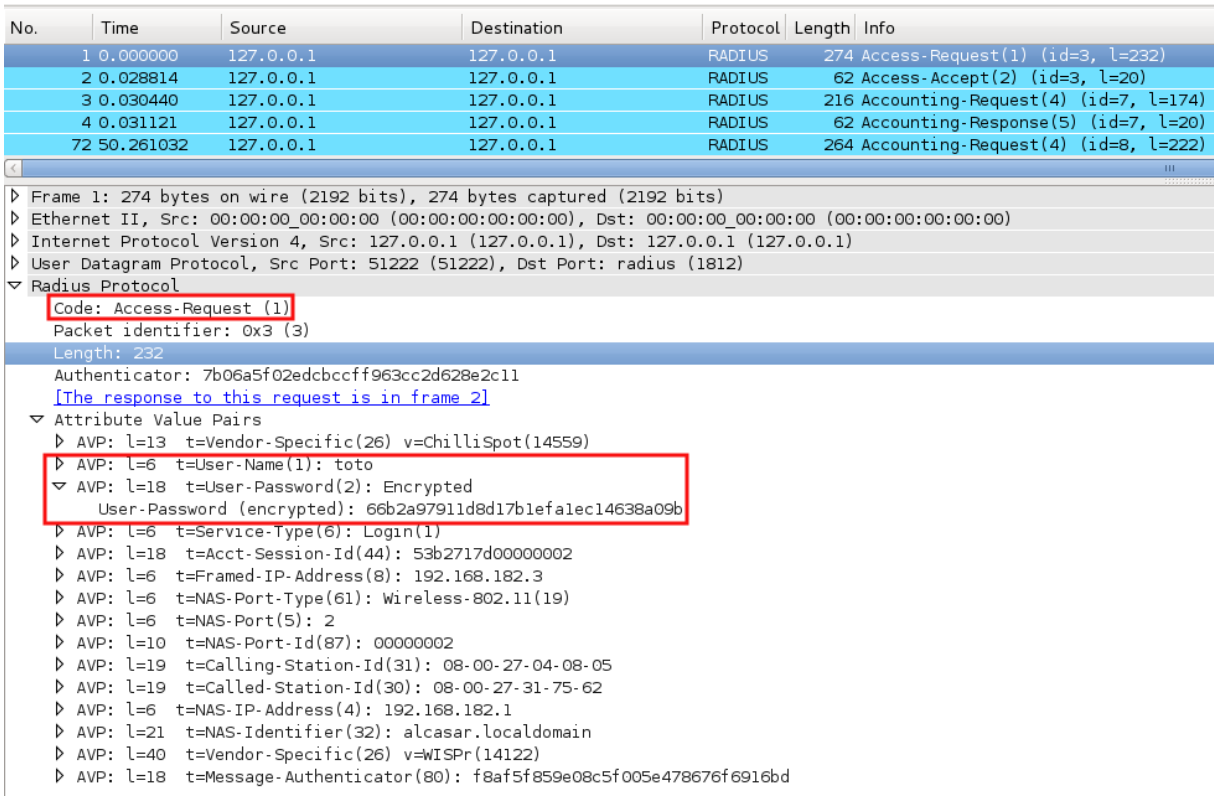

Figure 15: Demande d'authentification Radius par Coova-Chilli

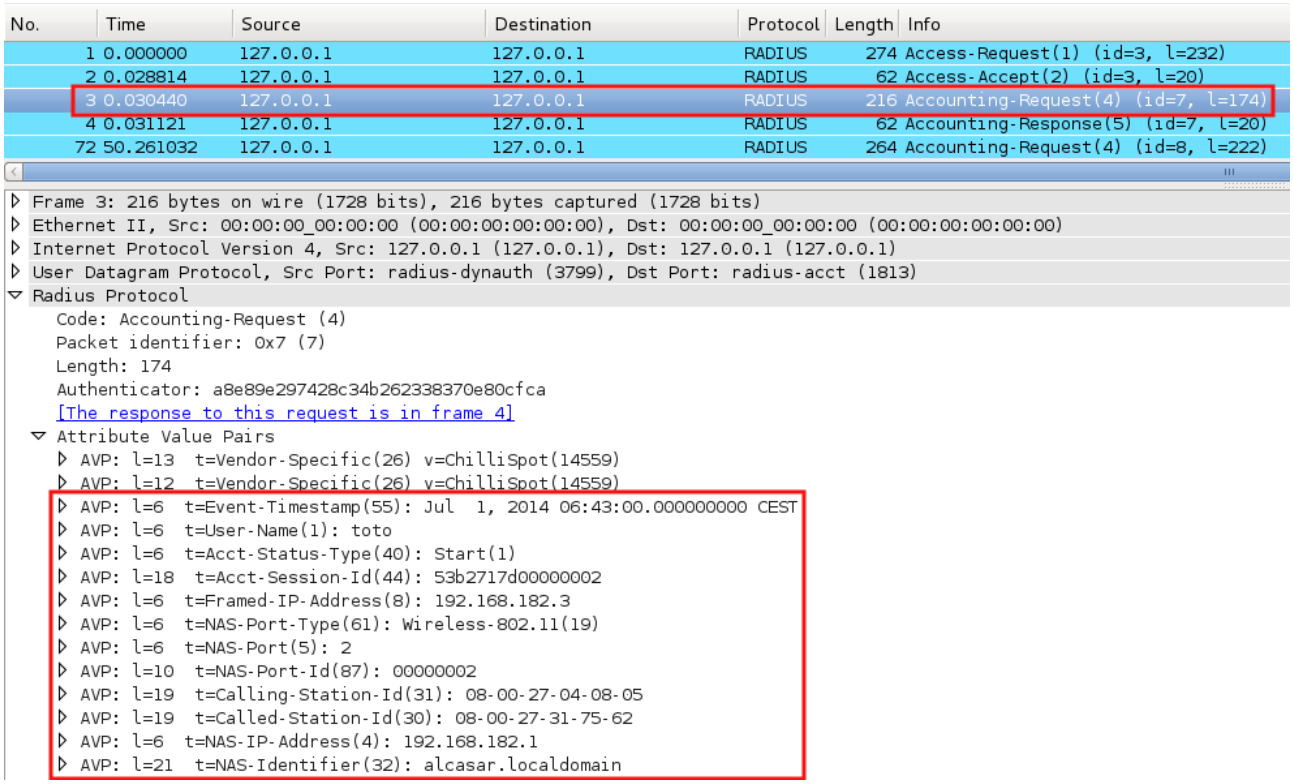

Figure 16: Demande d'enregistrement de données de comptabilité par Coova-Chilli

4. La base de données : MariaDB

Nous pouvons voir sur la figure ci-dessous la représentation des différentes tables de Radius. Les principales tables utilisées lors de l'authentification sont "radcheck", "radreply" et "radacct". Nous verrons donc en détail leurs spécificités.

```
MariaDB [radius]> show tables;
+-----------------+
| Tables_in_radius |
+-----------------+
| mtotacct        |
| radacct         |
| radcheck        |
| radgroupcheck   |
| radgroupreply   |
| radpostauth     |
| radreply        |
| radusergroup    |
| totacct         |
| userinfo        |
+-----------------+
```

Figure 17: Tables Radius

La table "radcheck" décrite ci-dessous montre comment les identifiants de session sont stockés.

La colonne "username" indique le nom d'utilisateur stocké dans la base, "attribute" spécifie un attribut de contrôle tandis que "value" correspondant à la valeur de l'attribut : dans l'exemple situé Figure 18, nous avons l'attribut de contrôle "Crypt-Password" qui correspond à l'empreinte du mot de passe de l'utilisateur. Le mot de passe est hashé en utilisant l'algorithme MD5(salted).

Dans les versions futures d'Alcasar (V2.9 et suivantes), l'algorithme a été remplacé par le SHA-256 pour des raisons de sécurité. De plus, un salage dynamique (salted) a été ajouté afin qu'un même mot de passe pour deux utilisateurs différents ne puisse pas produire la même empreinte. L'intérêt de conserver les mots de passe des utilisateurs sous forme d'empreinte permet qu'en cas de compromission de la base, un attaquant ne puisse pas exploiter directement le mot de passe. Cette table peut également contenir des attributs de contrôle. Par exemple : un temps de session maximal journalier ou bien une limitation sur la bande passante allouée.

```
MariaDB [radius]> select * from radcheck;
+----+-----------------+--------------------+----+-------------------------
| id | username        | attribute          | op | value
+----+-----------------+--------------------+----+-------------------------
|  1 | titi            | Crypt-Password     | := | $1$passwd$sGmzbelX1jCmgRsd
| 43 | lolo            | Crypt-Password     | := | $1$passwd$sGmzbelX1jCmgRsd
| 61 | E8-E7-32-49-17-C2 | Crypt-Password     | := | $1$passwd$qr0Ajhr12fZ475a2
| 60 | D8-C7-C8-C9-38-A9 | Crypt-Password     | := | $1$passwd$qr0Ajhr12fZ475a2
```

Figure 18: Table des attributs de contrôle Radius

La table "radreply" contient des attributs de réponse qui ont été préalablement renseignés dans le profil des utilisateurs. Ici, l'attribut "Filter-Id" est retourné à la passerelle Coova-Chilli lorsque l'authentification est réussie. Un script sur la passerelle permet de récupérer la valeur de "Filter-Id" et transmet cette information aux composantes de filtrage de Alcasar. Cette table peut aussi contenir un temps de session maximal autorisé à l'utilisateur.

```
MariaDB [radius]> select * from radreply;
+----+------------------+-----------+----+-------+
| id | username         | attribute | op | value |
+----+------------------+-----------+----+-------+
|  1 | titi             | Filter-Id | =  | Aucun |
| 18 | lolo             | Filter-Id | =  | Aucun |
| 16 | toto             | Filter-Id | := | Aucun |
```

Figure 19: Table des attributs de réponse Radius

La table "radacct" regroupe les données de comptabilité des utilisateurs. On y retrouve par exemple l'adresse IP de l'utilisateur, l'heure de début et de fin de chacune de ces sessions sur le portail captif, les données consommées en envoi et en réception, ...

```
MariaDB [radius]> show columns from radacct;
+----------------------+-------------+------+-----+---------+----------------+
| Field                | Type        | Null | Key | Default | Extra          |
+----------------------+-------------+------+-----+---------+----------------+
| radacctid            | bigint(21)  | NO   | PRI | NULL    | auto_increment |
| acctsessionid        | varchar(32) | NO   | MUL |         |                |
| acctuniqueid         | varchar(32) | NO   | MUL |         |                |
| username             | varchar(64) | NO   | MUL |         |                |
| groupname            | varchar(64) | NO   |     |         |                |
| realm                | varchar(64) | YES  |     |         |                |
| nasipaddress         | varchar(15) | NO   | MUL |         |                |
| nasportid            | varchar(15) | YES  |     | NULL    |                |
| nasporttype          | varchar(32) | YES  |     | NULL    |                |
| acctstarttime        | datetime    | YES  | MUL | NULL    |                |
| acctstoptime         | datetime    | YES  | MUL | NULL    |                |
| acctsessiontime      | int(12)     | YES  | MUL | NULL    |                |
| acctauthentic        | varchar(32) | YES  |     | NULL    |                |
| connectinfo_start    | varchar(50) | YES  |     | NULL    |                |
| connectinfo_stop     | varchar(50) | YES  |     | NULL    |                |
| acctinputoctets      | bigint(20)  | YES  |     | NULL    |                |
| acctoutputoctets     | bigint(20)  | YES  |     | NULL    |                |
| calledstationid      | varchar(50) | NO   |     |         |                |
| callingstationid     | varchar(50) | NO   |     |         |                |
| acctterminatecause   | varchar(32) | NO   |     |         |                |
| servicetype          | varchar(32) | YES  |     | NULL    |                |
| framedprotocol       | varchar(32) | YES  |     | NULL    |                |
| framedipaddress      | varchar(15) | NO   | MUL |         |                |
| acctstartdelay       | int(12)     | YES  |     | NULL    |                |
| acctstopdelay        | int(12)     | YES  |     | NULL    |                |
| xascendsessionsvrkey | varchar(10) | YES  |     | NULL    |                |
+----------------------+-------------+------+-----+---------+----------------+
```

Figure 20: Table des données de journalisation Radius

B) FreeRadius

Durant mon stage, j'ai été amené à optimiser les fichiers de configuration du serveur FreeRadius dans Alcasar. Cela a été possible en ayant pris connaissance en amont des mécanismes d'authentification. Dans un premier temps, je devais identifier les fichiers utiles à FreeRadius dans le cadre du projet Alcasar. Puis, dans un second temps, vérifier si les fichiers étaient correctement configurés. Enfin, il fallait que j'apporte corrections et améliorations afin d'optimiser le fonctionnement. Voir légende P. 34

Sur la figure 21, on retrouve une cartographie générale des fichiers FreeRadius.

Figure 21: Cartographie des fichiers de FreeRadius

Le schéma représenté ci-dessous montre les modules activés dans les différentes sections du fichier de configuration principal.

instantiate{} : Section permettant de définir un chaînage des différents modules fournis par FreeRadius. Ceux-ci sont préchargés au moment où le serveur démarre. Dans le cas d'une procédure d'authentification, les modules ne pourront s'exécuter qu'uniquement dans l'ordre dans lequel ils ont été initialisés.

modules {} : Section qui permet de définir les chemins des fichiers de configuration de chaque module. Les versions d'ALCASAR à partir de la 2.9 sont en mesure d'appeler seulement les modules requis pour les besoins du projet. En effet, les modifications que j'ai apportées ont permis de limiter au nécessaire le nombre de fonctionnalités lors du lancement du serveur FreeRadius.

Les modules permettent d'exécuter les tâches suivantes :

- Effectuer des requêtes avec la base de données MariaDB ("sql")
- Interroger un annuaire ("ldap")
- Calculer les temps de connexion à l'aide de compteurs sql ("logintime, dailycounter" et "monthlycounter").
- Définir une date d'expiration des comptes utilisateurs ("expiration")
- Contrôler un mot de passe ("pap")

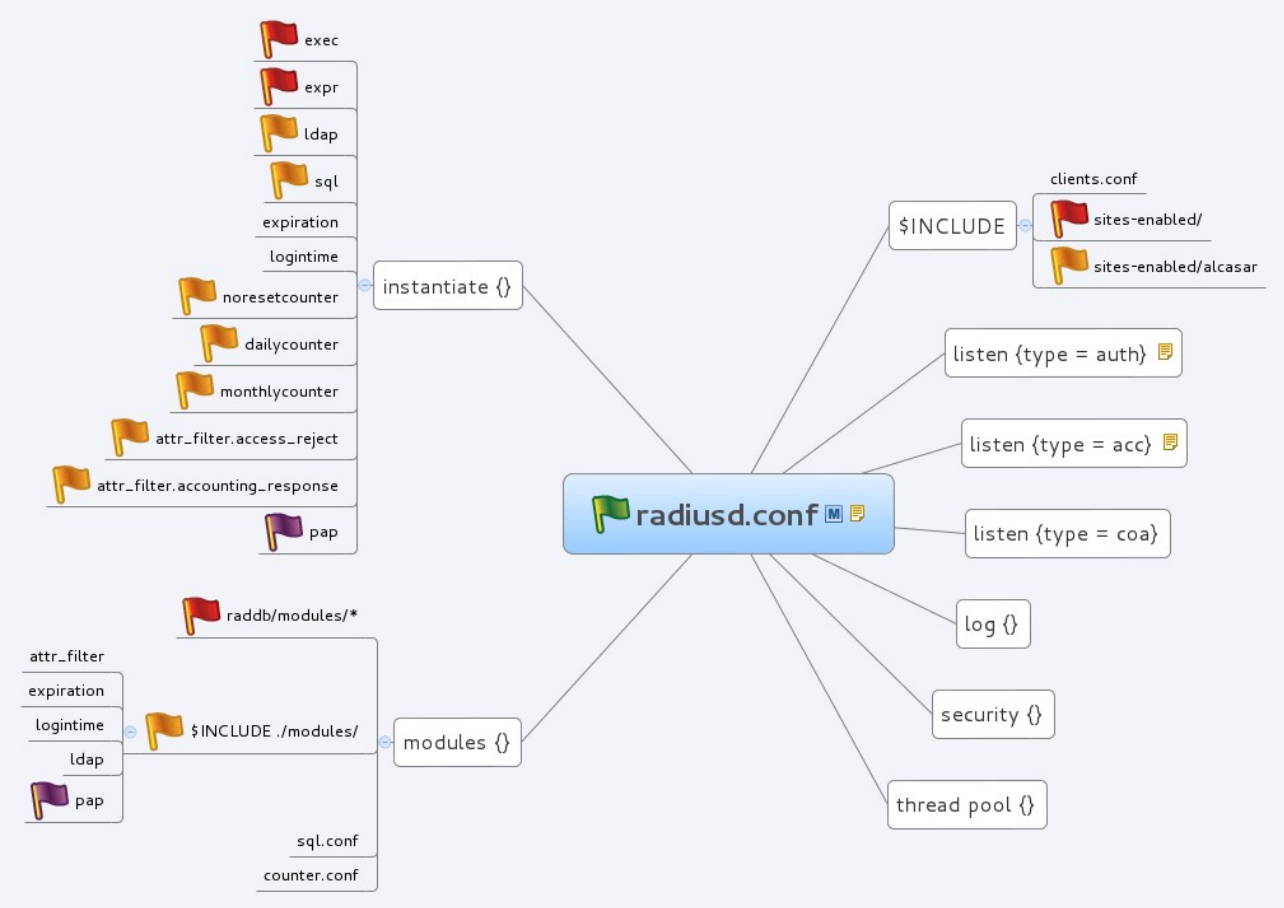

Figure 22: Représentation du fichier de configuration principal de FreeRadius

Le schéma suivant définit la politique adoptée pour le modèle AAA. On retrouve les modules nécessaires pour les sections d'authentification (authentication {}), d'autorisation (authorize {}) et de comptabilité (accounting {}).

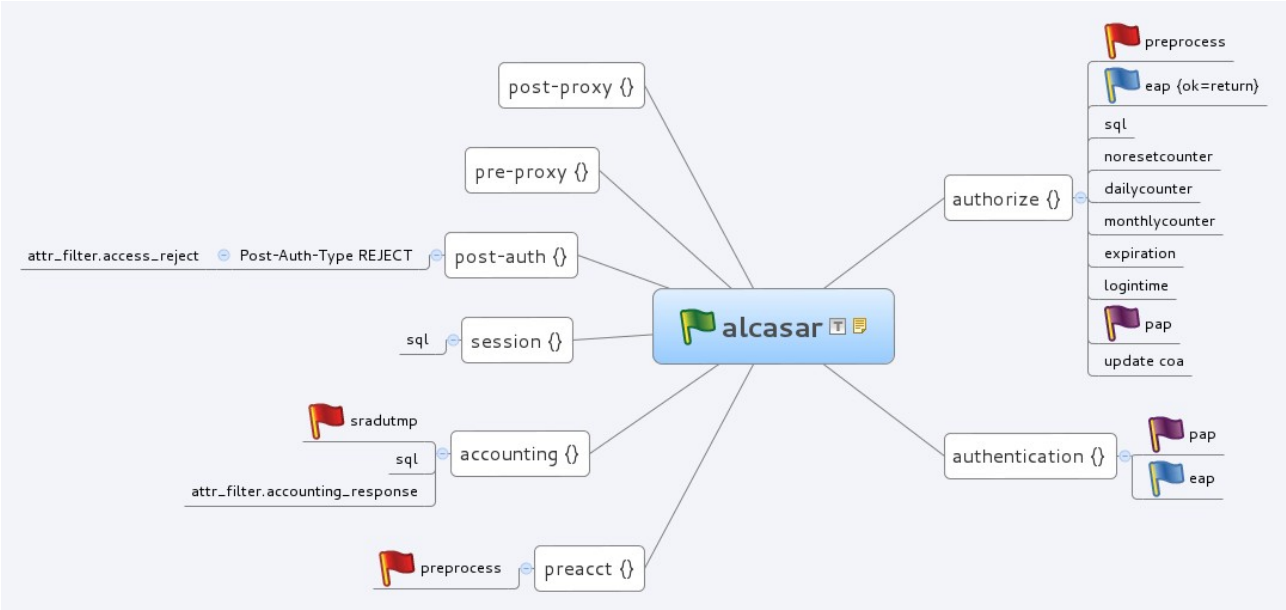

Figure 23: Représentation des sections de configuration du serveur virtuel ALCASAR au sein de FreeRadius

Légende

Drapeau vert : Fichiers utiles à la configuration de FreeRadius pour Alcasar
Drapeau bleu : Fichiers / modules nécessaires pour l'authentification 802.1X.
Drapeau orange : Ajout de fichiers / modules
Drapeau rouge : Suppression de fichier / modules

C) Authentification 802.1X

Le protocole 802.1X engendre une communication indirecte entre le poste de travail usager et le serveur Radius.

Ainsi, le processus d'authentification se fait dans un premier temps entre l'agent 802.1X appelé supplicant et le NAS qui est un équipement réseau (authentificateur) grâce à EAP[20]. Ce dernier relaye les messages EAP encapsulés dans des paquets Radius au service d'authentification. Le protocole Radius possède un attribut EAP-message qui permet d'acheminer les messages entre l'agent 802.1X et le client Radius.

Sur le schéma ci-dessous, on observe les différentes couches logicielles qui doivent être implémentées pour déployer le protocole 802.1X. En effet, l'utilisateur doit posséder sur son poste de travail un agent qui puisse dialoguer avec un équipement réseau en utilisant le protocole EAP. L'équipement réseau, quant à lui, doit prendre en charge le protocole EAP et le protocole Radius pour échanger avec le serveur d'authentification. Enfin, le service d'authentification doit avoir un module EAP lui permettant de récupérer les messages EAP issus de l'agent 802.1X.

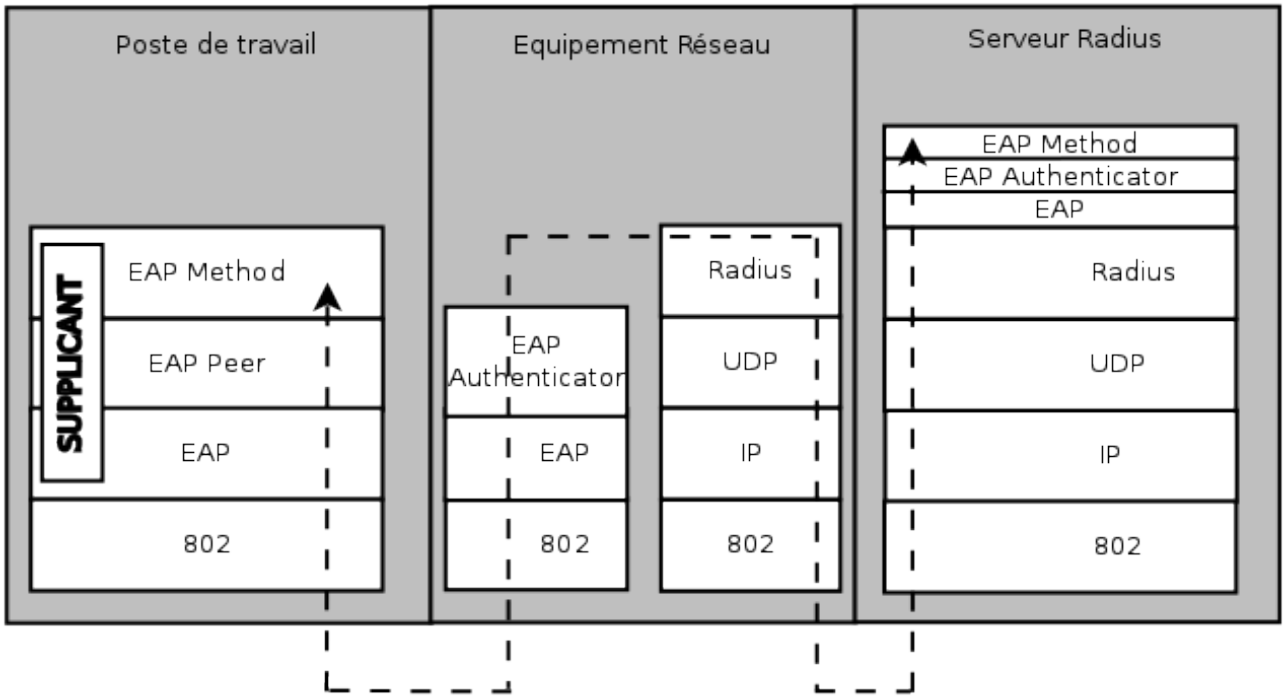

Figure 24: Implémentation des couches EAP par les différents acteurs entrant en jeu dans le protocole 802.1X

20 EAP : Extensible Authentication Protocol

On retrouve sur la figure 25, une vue générale du modèle de fonctionnement du protocole 802.1X.

La première étape consiste à établir un dialogue entre le système souhaitant s'authentifier et le serveur d'authentification.

Ce dialogue est obligatoirement réalisé au moyen du protocole EAP. Puis, si le système à authentifier répond aux critères requis, le serveur d'authentification envoie un message au client Radius afin d'autoriser celui-ci. Le système autorisé peut alors accéder au réseau.

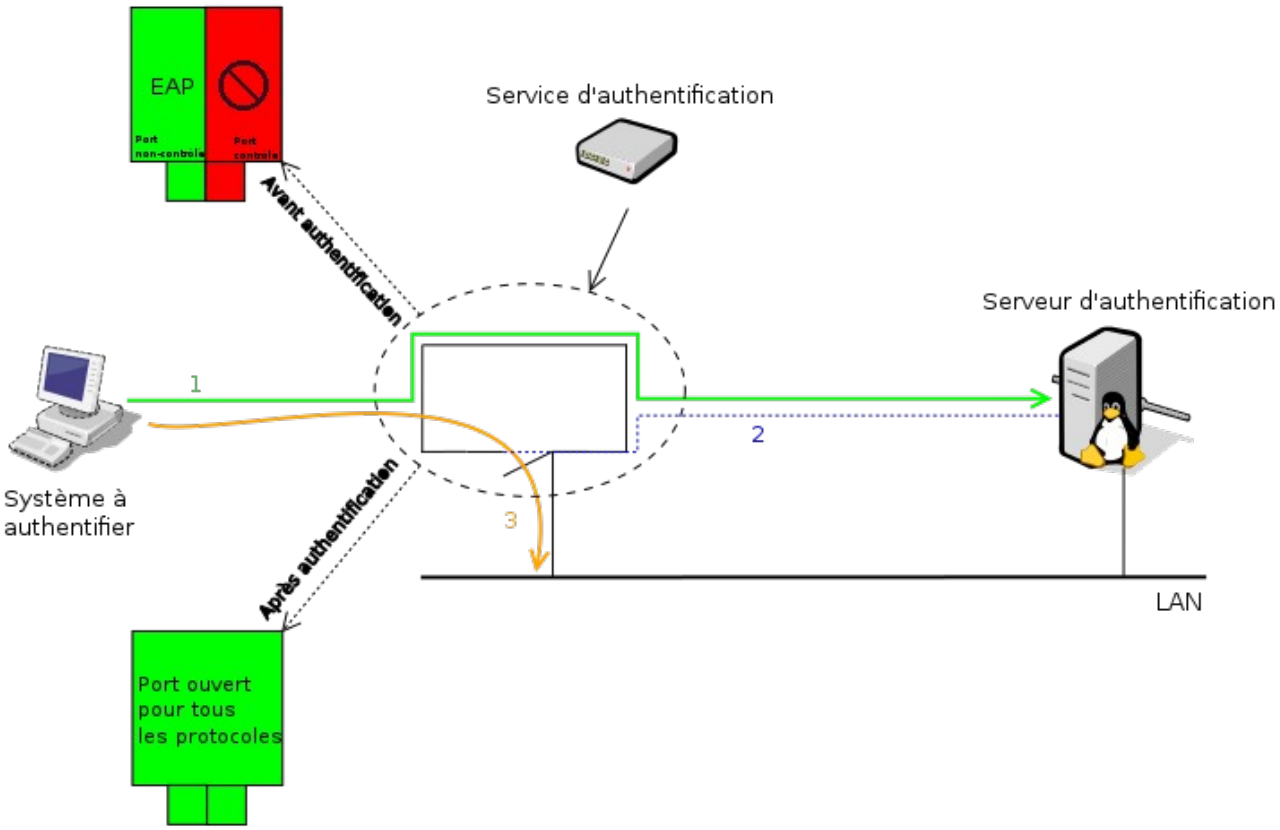

Figure 25: Schéma de principe de l'accès au réseau basé sur le contrôle de port

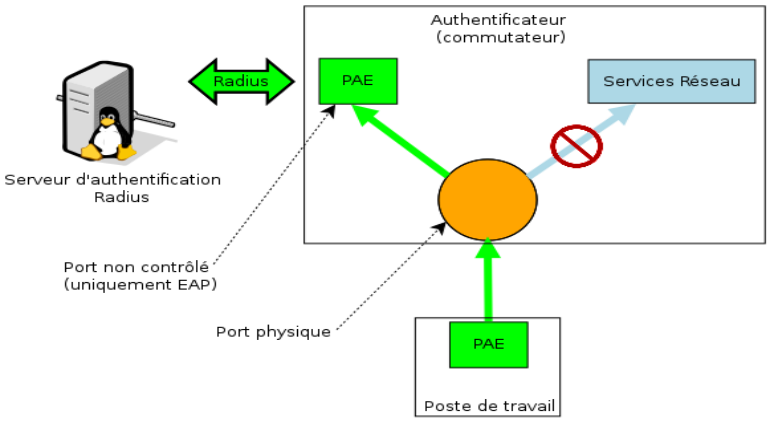

Figure 26: État du port du client Radius avant l'authentification 802.1X

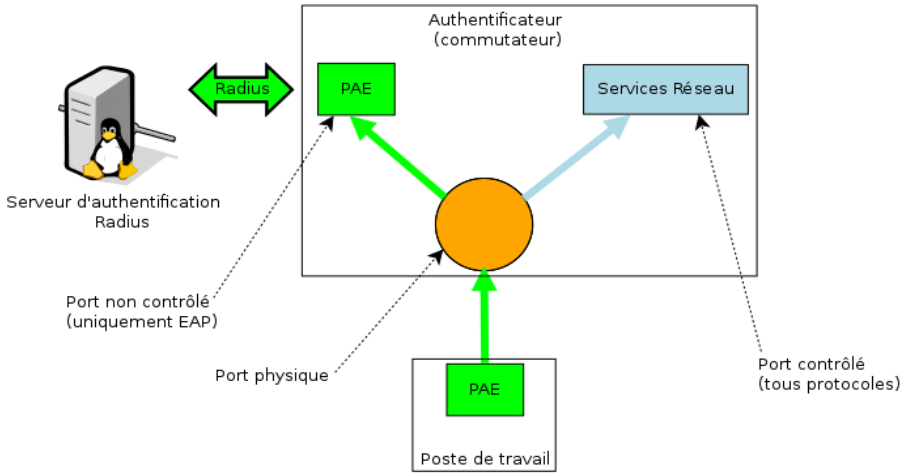

Figure 27: État du port du client Radius après l'authentification 802.1X

Dans les figures 26 et 27 le bloc PAE : Port Access Entity est le composant logique du modèle 802.1X. C'est lui qui permet les échanges de message EAP entre l'authentificateur et le supplicant.

Les couches EAP intègrent plusieurs méthodes d'authentification. Par conséquent, plusieurs mécanismes d'identification peuvent être mis en place. Le tableau ci-dessous fait une comparaison de quatre méthodes d'authentification.

Méthode d'authentification	EAP-MD5[21]	EAP-TLS[22]	EAP-PEAP[23]	EAP-TTLS[24]
Mode d'authentification	Unidirectionnel : authentification du client	Mutuel : certificats (client/serveur)	Mutuel : login/mot de passe (client) et certificat (serveur)	Mutuel : login/mot de passe (client) et certificat (serveur)
Protocole d'authentification de mot de passe	Fonction de hachage MD5 mot de passe sous forme d'empreinte numérique	Non	EAP-GTC[25] EAP-MSCHAPv2[26]	PAP[27] CHAP[28] MS-CHAP EAP-MD5 EAP-MSCHAPv2
Tunnel TLS	Non	Oui mais non utilisé	Oui	Oui
PKI[29] Client	Non	Oui	Oui (optionnelle)	Oui (optionnelle)
PKI Serveur	Non	Oui	Oui	Oui
Compatibilité avec des protocoles non EAP	Non	Non	Non	Oui
Niveau de sécurité	Faible, vulnérable au MITM et la fonction de hachage est vulnérable aux attaques par dictionnaire	Très bon	Bon	Très bon, peut être utilisé avec des serveurs d'authentification qui ne prennent pas en charge EAP
Distribution dynamique de clef de chiffrement pour les communication sans fil	Non	Oui	Oui	Oui

21 MD5 : Message Digest 5
22 TLS:Transport Layer Security
23 PEAP : Protected Extensible Authentication Protocol
24 TTLS : Tunneled Transport Layer Security
25 GTC : Generic Token Card
26 MS-CHAP : Microsoft Challenge Handshake Authentication Protocol
27 PAP : Password Authentication Protocol
28 CHAP : Challenge Handshake Authentication Protocol
29 PKI : Public Key Infrastructure

L'utilité du protocole EAP est de transporter la méthode d'authentification et le protocole d'authentification.

La méthode d'authentification choisie pour le protocole 802.1X au sein d'ALCASAR est l'EAP-TLS. Cette méthode offre une authentification forte car elle repose sur deux critères d'authentification : ce que l'utilisateur connaît (mot de passe / code PIN) et ce que l'utilisateur possède (certificat électronique embarqué dans une carte à puce / authentificateur USB).

Au sein du projet, je devais utiliser la solution Coova-Chilli dans le mécanisme d'authentification. Il a donc fallu paramétrer celui-ci en proxy Radius afin qu'il puisse relayer les messages des équipements réseaux vers le serveur Radius. Cela a permis de conserver la fonction de journalisation des utilisateurs connectés sur le réseau de consultation Alcasar.

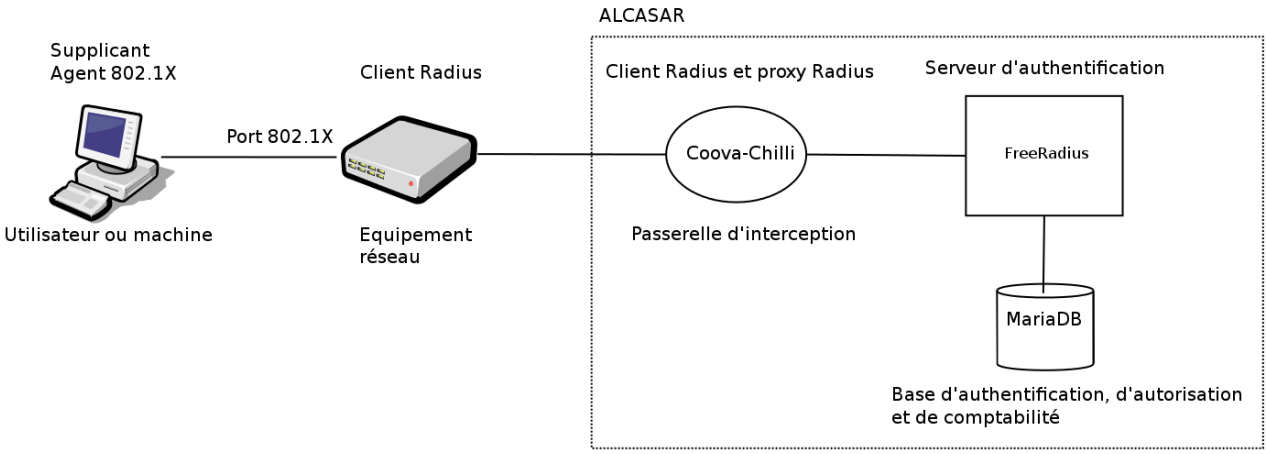

Figure 28: Implémentation de 802.1X dans ALCASAR

D) Problèmes rencontrés : Authentification 802.1X

Durant mon stage, j'ai été confronté à un problème de corruption des paquets. Lors de l'authentification d'un usager, en utilisant la méthode EAP-TLS, le dialogue entre l'agent 802.1X et le client Radius s'interrompait lorsque l'agent envoyait le certificat client.

On remarque sur la figure ci-dessous que la taille attendue dans l'en-tête 802.1X (1408 octets) ne correspond pas à la taille réelle donnée par la méthode EAP-TLS (2450 octets). Dans la représentation hexadécimale, on peut également identifier un fragment du certificat utilisateur.

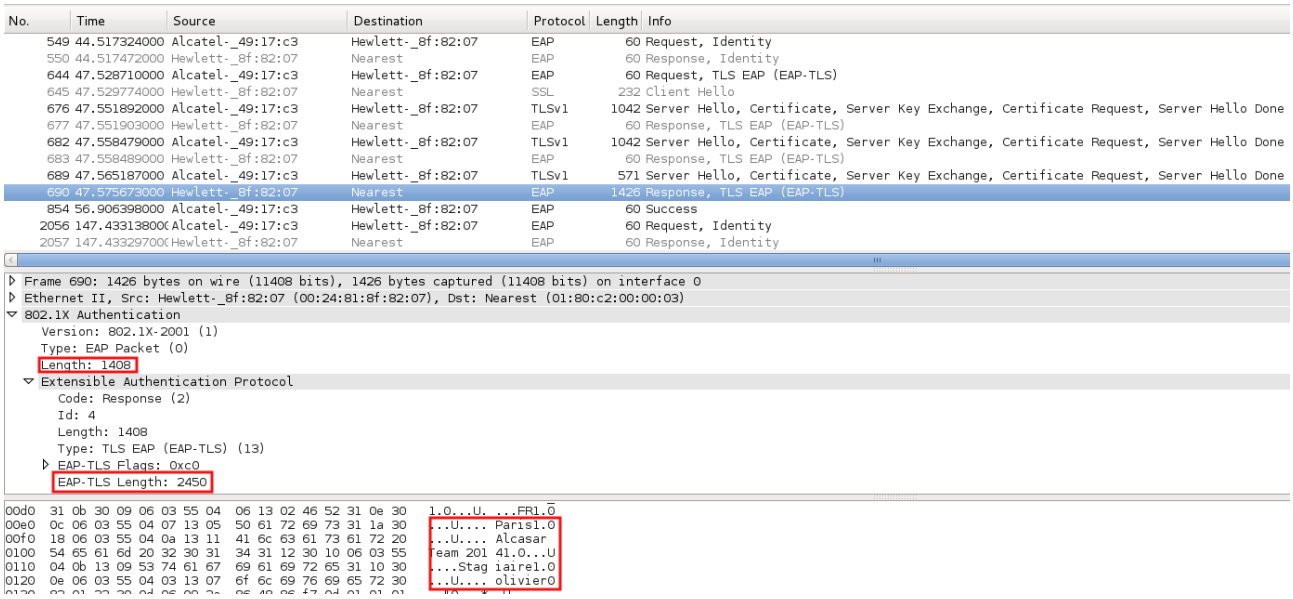

Figure 29: Dialogue entre supplicant et Coova-Chilli interrompu lors de l'utilisation de la méthode EAP-TLS.

La figure ci-dessous représente les retours ICMP par Coova-Chilli lors de l'authentification EAP-TLS. Grâce à ces messages, j'ai pu identifier le problème qui portait sur la taille des paquets envoyés par l'agent 802.1X du poste utilisateur. Le protocole EAP opère sur la couche liaison du modèle OSI et est encapsulé par le protocole Ethernet.

Ce protocole ne peut gérer la fragmentation des paquets. En effet, seuls les protocoles de plus haut niveau comme TCP sont en mesure de réaliser cette opération grâce au contrôle de la taille des paquets en fonction de la MTU[30] définie.

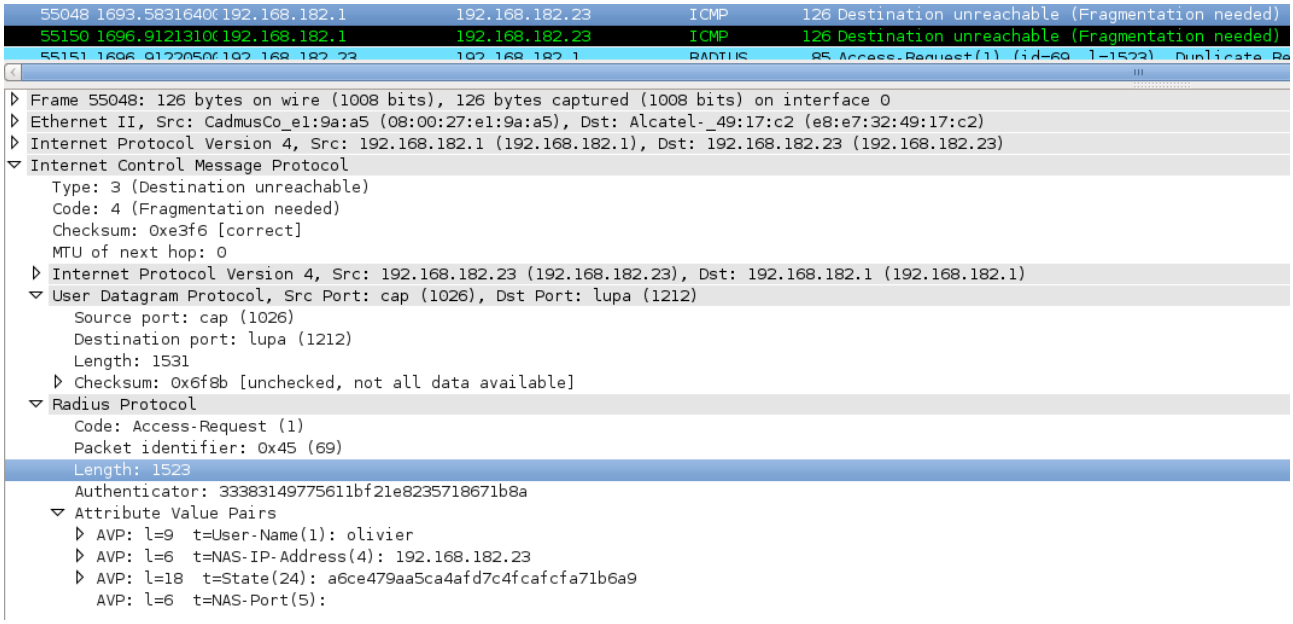

Figure 30: Messages ICMP retournés par Coova-Chilli

30 MTU : Maximum Transmission Unit

Ci-dessous, on observe une authentification EAP-TLS complète après résolution du problème. Cette résolution a eu lieu par l'ajout d'un attribut de configuration au sein de l'agent 802.1X définissant la possibilité de fragmentation des paquets.

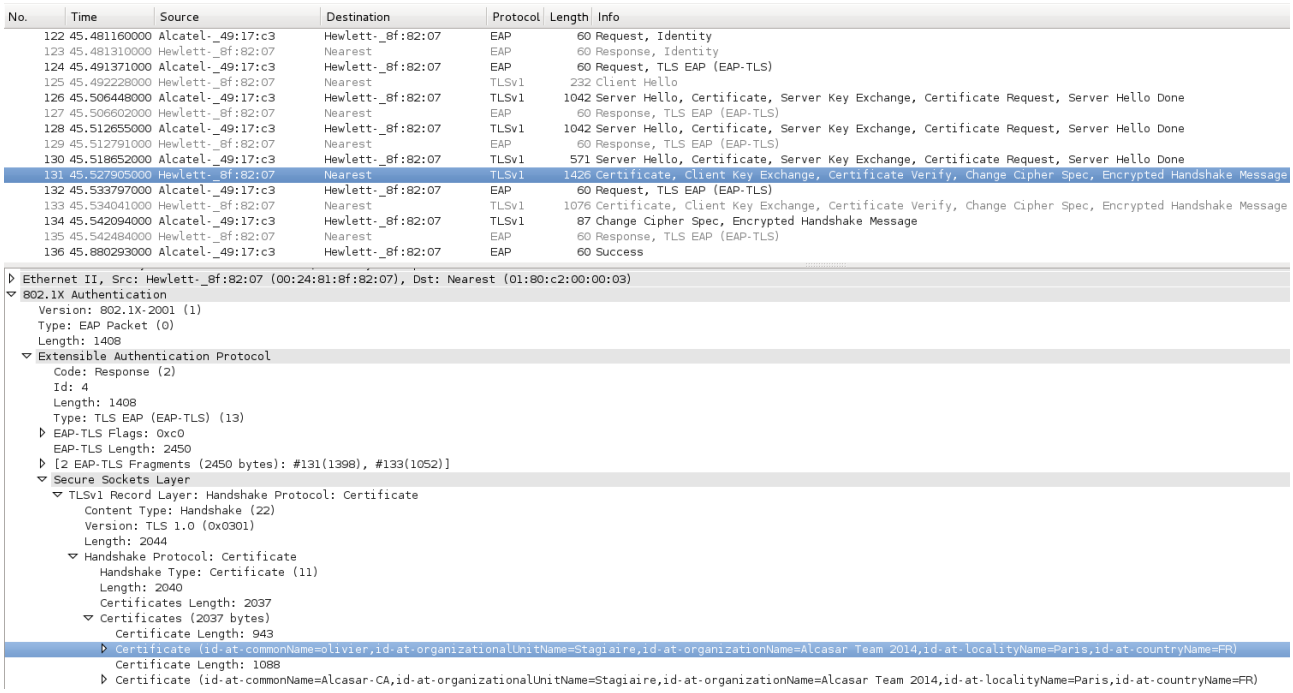

Figure 31: Côté supplicant méthode EAP-TLS avec coova en proxy Radius succès

E) Problèmes rencontrés : Point d'accès Wi-Fi

J'ai également rencontré des difficultés lorsque j'ai mis en œuvre des authentifications Radius sans fil dans le cadre de l'écriture des travaux dirigés Radius (cf. Annexe 1 P. 91 et P. 92). J'ai été amené à utiliser un point d'accès Netgear WNR 3400v3 ainsi qu'un point d'accès Alcatel-Lucent 105.

Le firmware initial du constructeur Netgear ne fournissait pas les couches logicielles nécessaires pour réaliser une authentification via Radius. En effet, j'ai dû remplacer le firmware propriétaire par un firmware libre (dd-wrt) compatible avec le modèle.

La mise à jour du firmware a permis d'étendre les fonctionnalités du point d'accès et ainsi permettre une authentification forte via le protocole Radius.

Malheureusement, le point d'accès en tant que client Radius ne répondait qu'à un seul des 3 critères AAA, soit l'authentification. Le routeur a montré ses limites quant à la comptabilité et à l'autorisation où il n'a pu remplir sa tâche.

Par conséquent, j'ai dû utiliser un autre équipement, l'Alcatel-Lucent 105 où j'ai rencontré une impasse. Ce dernier est un point d'accès bien plus évolué que le Netgear car il prend nativement les fonctionnalités du protocole Radius. On y retrouve 2 critères sur 3 soit l'authentification et la

comptabilité. Concernant le dernier critère qui est l'autorisation, je me suis aperçu que le modèle ne pouvait réaliser cette opération que par l'utilisation d'un logiciel tiers appartenant à la firme Alcatel-Lucent. Ce logiciel étant très coûteux, je n'ai pas pu l'acquérir. J'ai donc fait des recherches infructueuses sur un éventuel firmware libre, ce qui m'a mené à contacter le constructeur à l'origine du produit Alcatel-Lucent 105. En effet, Alcatel-Lucent n'est pas l'OEM (Original Equipment Manufacturer traduit littéralement par Fabriquant d'Équipement d'Origine) mais un revendeur officiel pour ce type de produit.

L'objet de ma requête à l'intention d'Aruba était d'obtenir de leur part le firmware correspondant à ce modèle (prenant en charge le triple A) ce à quoi ils m'ont répondu qu'ils n'étaient pas en mesure de le fournir car le numéro de série était enregistré sous la marque Alcatel-Lucent.

En résumé, du Netgear ou de l'Alcatel-Lucent, les tests n'ont pu être réalisés entièrement avec la technologie sans fil.

Figure 33: Netgear WNR 3400v3

Figure 32: Alcatel-Lucent 105

F) Améliorations & Perspectives

- L'implémentation du protocole 802.1X étant réalisée, une ouverture possible serait de configurer la passerelle d'interception afin qu'elle puisse gérer deux réseaux utilisant des méthodes d'authentification différentes. Ci-dessous, un schéma de principe de cette possible réalisation future.

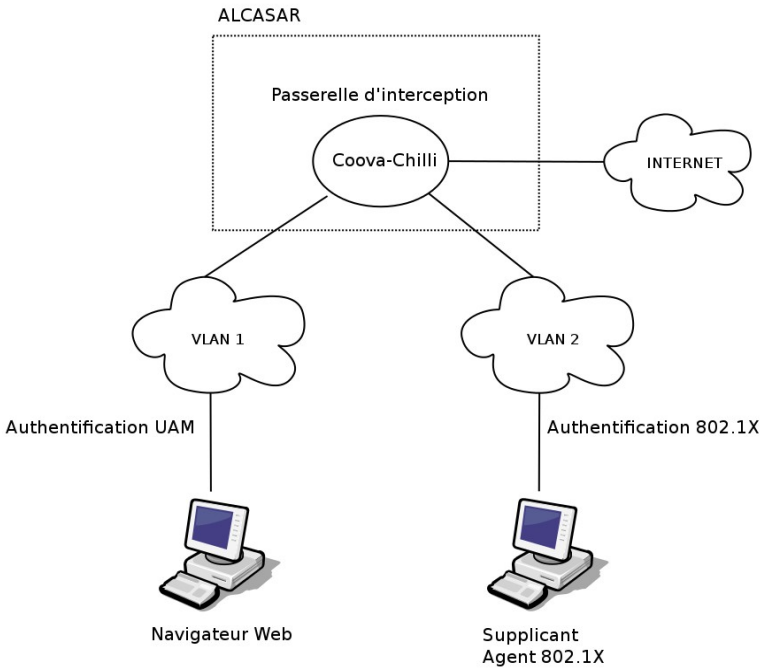

Figure 34: Gestion de deux réseaux virtuels par Coova-Chilli intégrant l'authentification UAM et 802.1X

- Une autre amélioration envisageable consisterait à exploiter la fonctionnalité COA proposé par FreeRadius afin de modifier les autorisations d'un utilisateur ou de couper la connexion à un usager connecté de manière autonome sur un équipement compatible 802.1X.

- Mettre en œuvre l'implémentation d'un agent 802.1X pour qu'il puisse interagir de manière intuitive avec les utilisateurs sur des machines en libre accès.

Conclusion

Les différentes tâches que j'ai effectuées au sein du laboratoire de Laval m'ont permis de m'enrichir sur les aspects techniques où j'ai pu aborder de nombreux sujets sur les réseaux et la sécurisation de ceux-ci.

Les différentes interactions que j'ai eues avec mon maître de stage et les autres membres du laboratoire m'ont également enseigné de nombreux aspects notamment sur le comportement à adopter en entreprise, la collaboration et l'esprit d'équipe.

L'ambiance au sein du stage était de rigueur puisque j'étais dans un cadre agréable avec une véritable entraide entre les membres du CVO.

Sur le plan technique, j'ai exploré de manière détaillée de nombreuses facettes du monde des réseaux informatiques mais aussi du système Linux. Je retiens de nombreuses notions au sein de ce stage et, en particulier :

- La prise en main de la distribution Mageia
- Les caractéristiques d'un portail captif
- L'authentification
- La gestion d'une base d'usagers
- Le fonctionnement des équipements réseau (commutateurs, bornes WIFI)
- Les attaques sur les réseaux informatiques

Pour résumer, je qualifierai ce stage d'une réelle opportunité et je suis très reconnaissant de toute l'équipe du CVO et notamment mon maître de stage M. Richard REY ainsi que mon tuteur pédagogique M. Nicolas DEVILLY qui m'ont apporté le savoir nécessaire pour mener à bien l'ensemble du projet.

Étant à présent au Mastère SI&S (Sécurité de l’Information et des Systèmes) de l'ESIEA, j'ai pour ambition de poursuivre dans le domaine de la sécurité informatique dont le facteur de départ a été ce stage.

Bibliographie et Webographie

Radius
Serge Bordères, Authentification réseau avec Radius, Édition Eyrolles

http://en.wikipedia.org/wiki/RADIUS

http://en.wikipedia.org/wiki/IEEE_802.1X

Authentification
http://fr.wikipedia.org/wiki/Authentification

http://www.nolot.eu/Download/Cours/reseaux/m2pro/SESY0708/proto-authentification.pdf

http://repo.hackerzvoice.net/depot_madchat/sysadm/unix.seku/Authentification_de_A_a_Z.pdf

http://en.wikipedia.org/wiki/Extensible_Authentication_Protocol

CNS
http://www.esiea.fr/recherche/expertise-confiance-numerique-securite

Projet Alcasar
http://www.alcasar.net/

Projet Davfi
https://www.davfi.fr/

Projet Uhuru
http://uhuru-am.com

FreeRadius
http://freeradius.org/

http://en.wikipedia.org/wiki/FreeRADIUS

http://www.tldp.org/HOWTO/8021X-HOWTO/freeradius.html

http://deployingradius.com/

Diameter
http://en.wikipedia.org/wiki/Diameter_%28protocol%29

Table des figures