Protection de l'accès au réseau

802.1x, Radius, EAP

Guillaume Piolle

Supélec – majeure SIS

7 octobre 2011

Introduction

Objectif

Protection de l'accès au réseau (*vs* accès aux services)
NAC : *Network Access Control*

AAA

- *Authentication :* Identification des machines ou des utilisateurs ;
- *Authorization :* Attribution de droits particuliers (jetons d'accès, configuration réseau, temps d'accès...) ;
- *Accounting :* Collecte de statistiques d'utilisation en vue de la facturation.

Solution étudiée : combinaison de 802.1x, EAP et Radius

802.1x

IEEE 802.1x Port-Based Network Access Control

Les trois acteurs du protocole

- *Supplicant :* Client cherchant à obtenir l'accès au réseau ;
- *Authenticator :* Point d'accès au réseau devant authentifier le client ;
- *Authentication server :* Autorité d'authentification.

802.1x

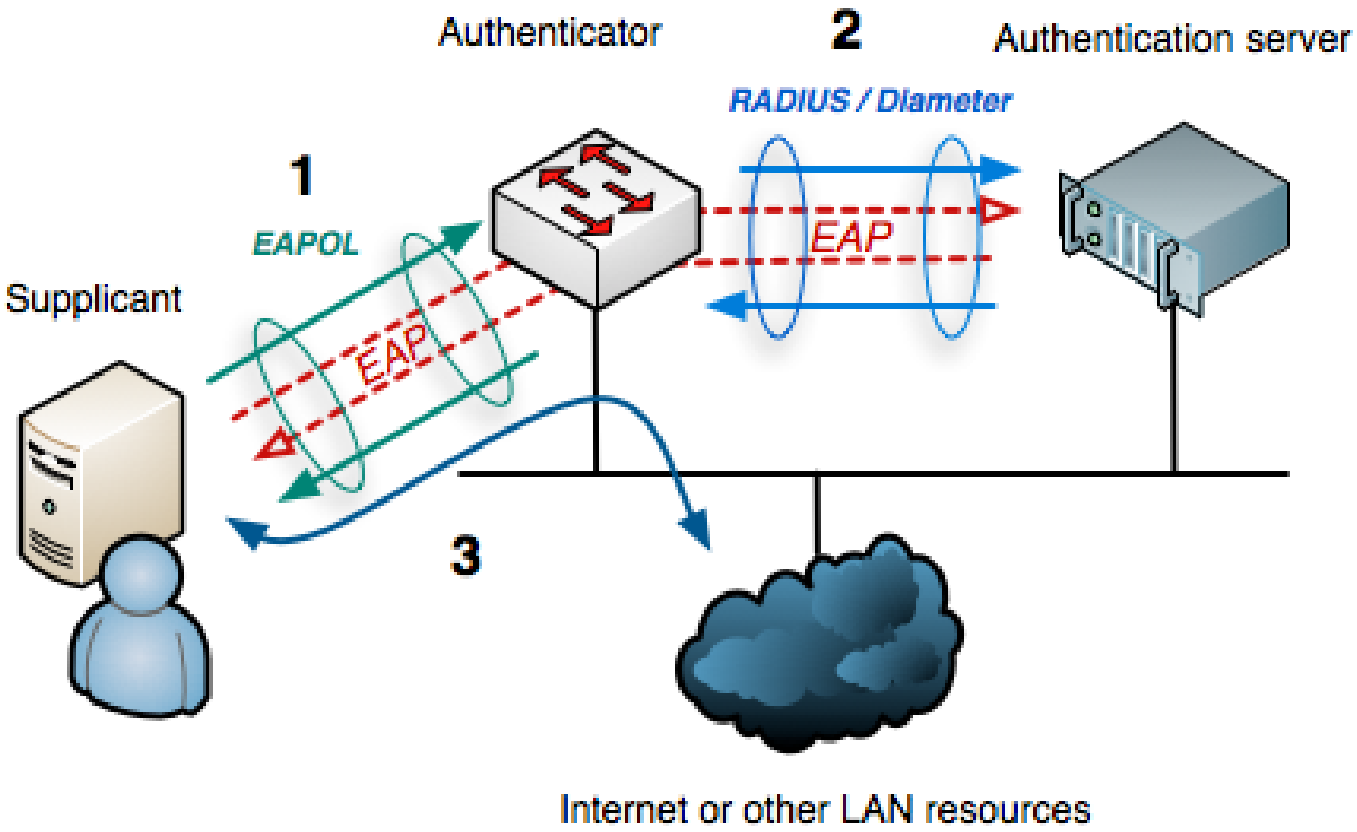
Principe : sur l'*authenticator*, un port physique est scindé en deux ports logiques :

- Un port non contrôlé mais dédié à la communication 802.1x ;
- Un port contrôlé (fermé jusqu'à confirmation de l'authentification, puis ouvert à tout trafic).

Authentification via le protocole d'encapsulation EAP (l'utilisation d'EAP sur 802 est définie par 802.1x)

Utilisation : surtout pour l'authentification wifi (WPA ou WPA2 « entreprise »)

802.1x



Arran Cudbard-Bell, GFDL

EAP : *Extensible Authentication Protocol* (RFC 3748)

Protocole de transport conçu pour l'encapsulation de protocoles d'authentification (indépendance entre transport et méthode d'authentification)

Permet d'encapsuler... à peu près n'importe quoi.
Intérêt : *lingua franca* compréhensible par les *authenticators*.

Dans le cadre de 802.1X :

- EAP over LAN (EAPOL) pour la communication *supplicant-authenticator* (paquet EAP encapsulé dans une trame 802)
- EAP over RADIUS pour la communication *authenticator-authentication server* (paquet EAP encapsulé dans un message RADIUS– de niveau supérieur !)

Principe

Six types de paquets :
Request, *Response*, *Success*, *Failure*, *Initiate**, *Finish**.
$(*)$ uniquement pour le protocole ERP
$+$ 4 types ajoutés par 802.1x pour gérer le 802.11 : *Start*, *Packet*, *Key* et *Logoff*.

Échange typique :

- $\leftarrow$ Request/Identify
- $\rightarrow$ Response/Identify
- $\leftarrow$ Request/MD5-Challenge
- $\rightarrow$ Response/MD5-Challence
- $\leftarrow$ Success

Format d'encapsulation

Structure des paquets EAP

- Code : 1 octet (de 1 à 6 pour *Request*, *Response*, *Success*, *Failure*, *Initiate*, *Finish*) ;
- Identifiant : 1 octet (pour l'association des requêtes et des réponses) ;
- Longueur : 2 octets ;
- Type : 1 octet (uniquement présent dans les échanges *Request*/*Response*, identifie le type de requête : identifiant, challenge de type donné, etc.) ;
- Données : variable, encapsulation des méthodes EAP (vide pour *Success* et *Failure*).

EAP-TLS (RFC 5216)

Encapsulation d'un tunnel TLS dans EAP, avec authentification bilatérale par certificat (lourd pour le client).
Le tunnel est fermé après l'authentification, mais la clé échangée peut être utilisée pour en reconstruire un.

Habituellement considérée comme la méthode EAP la plus sûre (?) et la plus facilement supportée.

EAP-PEAP (Cisco-MS-RSA), EAP-TTLS (Funk software, Certicom, RFC 5281)

Encapsulation d'un tunnel TTLS dans EAP, avec authentification du serveur mais pas du client (typique HTTPS).

EAP-TTLS

L'authentification se fait directement dans le tunnel (en utilisant les attributs RADIUS directement, par exemple).
Support répandu mais pas de support natif sous Windows.

EAP-PEAP

Un nouvel EAP est encapsulé dans le tunnel.
Moins standard que EAP-TTLS, mais très répandu, soutenu par Cisco et Microsoft.

EAP-MSCHAPv2 (Microsoft, RFC 2433)

Méthode la plus classique dans un EAP-PEAP, également nommé PEAPv0.
Variation du *Challenge Handshake Authentication Protocol* : challenge cryptographique redemandé à intervalles aléatoires.
Sûr seulement si encapsulée dans un tunnel après vérification du certificat serveur (MS-CHAPv2 est très vulnérable aux attaques en force brute).

Support le plus répandu après EAP-TLS.

EAP-IKEv2 (RFC 5106)

Établissement d'une clé de session à partir de crédences respectives du client et du serveur, de nature variables :

- Certificats X509 (IKEv2 original) ;
- Mot de passe ;
- Clé symétrique ;
- Paire de clés asymétriques.

Autres méthodes

Protocoles non traités

- EAP-OTP (*One-Time Password*) ;
- EAP-GTC (*Generic Token Card*, RFC 2284 et 3748) ;
- EAP-SIM (spécifique GSM, RFC 4186) ;
- EAP-AKA (spécifique UMTS, RFC 4187) ;
- EAP-PSK (*pre-shared key*, léger) ;
- EAP-MD5 (hérite de la vulnérabilité de MD5) ;
- EAP-LEAP (protocole Cisco conçu pour le WEP, MS-CHAP modifié, failles exploitées depuis 2004, très peu supporté) ;
- EAP-FAST (remplacement de EAP-LEAP par Cisco, RFC 4851).

Radius

Remote Authentication Dial In User Service

Livingston Enterprises 1991 (pour la protection de l'accès à NSFnet), puis standard IETF (RFC 2865, 2866)

Protocole simple mais vieillissant (toujours très utilisé cependant, notamment par les fournisseurs d'accès)

Authentication server dans l'architecture 802.1x

Principes

AAA

- *Authentification et autorisation* : RFC 2865
- *Accounting* : RFC 2866

L'*authenticator* s'appelle souvent RAS (*Remote Access Server*, ou NAS)
Radius peut utiliser PAP, CHAP, EAP...

Realms

Un identifiant de *Realm* peut être envoyé avec la demande de connexion (souvent un suffixe @ ajouté au login, parfois un préfixe).
Il identifie un serveur Radius particulier auquel la demande doit être transférée.

Le serveur Radius peut alors servir de mandataire pour la communication avec le serveur du realm correspondant. La liaison peut être tunnelée, par défaut avec MD5.

Utilisé pour le *roaming* ou les architectures d'organisations complexes (exemple : réseau Eduroam).

Protocole de communication

Protocole de la couche application, utilisation d'UDP en transport
Liaison client - RAS en LL (niveau 2, typiquement PPP ou 802), ou autre (HTTPS)
Liaison RAS-Radius tunnelée à l'aide de MD5 et d'un secret partagé (bien mais pas top).

Structure des paquets Radius

- Code : 1 octet (*Access-Request*, *Access-Accept*, *Access-Reject*, *Accounting-Request*, *Accounting-Response*, *Access-Challenge* + ...) ;
- Identifiant : 1 octet (correspondance des messages) ;
- Longueur : 2 octets ;
- *Authenticator* : 16 octets (tirés au hasard, utilisés, avec le secret partagé, pour chiffrer le mot de passe du client) ;
- Attributs : 0-4076 octets.

Protocole de communication

- Le RAS transmet un *Access-Request* ;
- Radius a trois réponses possibles :
  - *Access-Reject* ;
  - *Access-Challenge* (demande d'infos supplémentaires, challenge ou établissement d'un tunnel) ;
  - *Access-Accept*.

Chacun des trois messages de Radius peut avoir un attribut *Reply-Message* avec une une explication de rejet, un prompt de challenge, un message de bienvenue...

Autorisation

Infos contenues dans les attributs d'un *Access-Accept*. Typiquement :

- Adresse IP ;
- Pool d'adresse IP ;
- Durée maximale de connexion ;
- Access lists ;
- Priorité/QoS ;
- Paramètres de tunneling VPN ;
- Attribution d'un VLAN ;
- ...

Il faut que le RAS comprenne ces attributs et soit capable de les mettre en œuvre !

Accounting

- Radius envoie au RAS un message *Accounting-Request* avec un attribut *Acct-Status-Type* à `start` + l'ID de l'utilisateur, un identifiant de session et d'autres infos annexes ;
- Le RAS répond avec un message *Accounting-Response* ;
- Le RAS envoie à Radius des paquets *Accounting-Request* avec *Acct-Status-Type* à `interim-update`, avec des informations sur la session (utilisation de ressources, etc.) ;
- Le RAS envoie un message `stop` à Radius à la déconnexion, avec des statistiques globales et le motif de déconnexion.

Sources possibles pour l'authentification

(et également pour l'autorisation)

- Fichier de configuration ;
- annuaire X500, LDAP, Active Directory ;
- SGBDR ;
- Kerberos ;

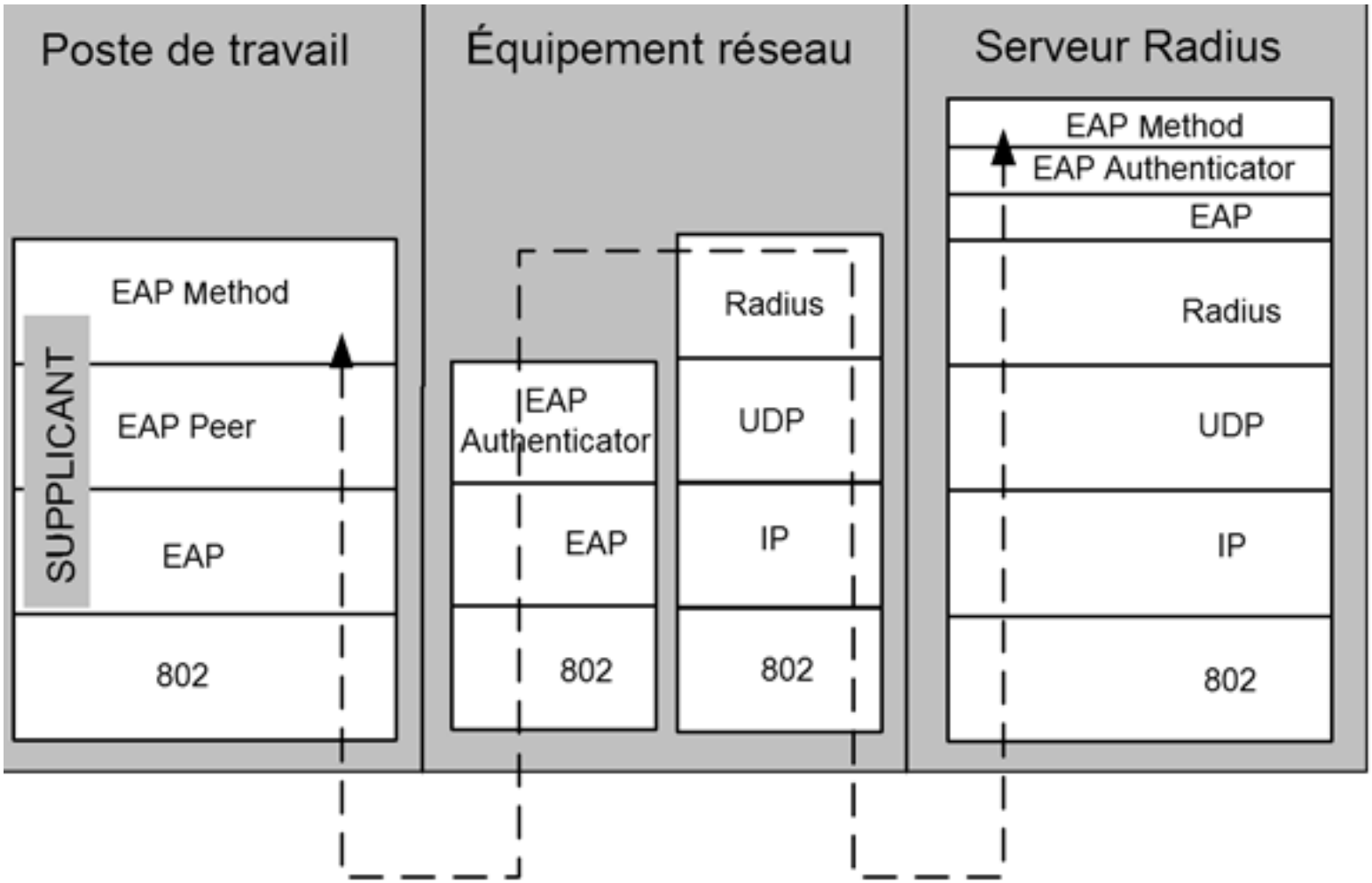
Diameter

Remplacement prévu de Radius, de plus en plus utilisé.

- Utilisation de TCP ou SCTP à la place d'UDP ;
- Utilisation d'IPsec ou de TLS ;
- Support partiel pour Radius ;
- Plus de place dans le format des paquets ;
- Amélioration des capacités de négociation ;
- Meilleur support du *roaming* ;
- Extensible (commandes, attributs) ;
- ...

Application de contrôle de débit/crédit, téléphonie mobile...

Intégration



Les couches relatives à EAP

La couche EAP

Elle reçoit et envoie les paquets vers la couche basse (802).
Les types Request, Success et Failure sont transmis à la couche EAP Peer.
Les paquets Response sont transmis à la couche EAP Authenticator.

Les couches relatives à EAP

Les couches EAP Peer et EAP Authenticator

EAP Peer : poste de travail.
EAP Authenticator : NAS/RAS, serveur RADIUS.
Interprétation du type de paquet (*(Request/Response)*, redirection vers la couche EAP method correspondant au protocole utilisé.

EAP method

Contient le code logiciel du protocole d'authentification utilisé.

Mise en œuvre de l'encapsulation

- Le *supplicant* envoie un paquet EAP au NAS ;
- Le NAS extrait le paquet EAP et le fait passer dans la couche Radius (et inversement au retour) : encapsulation de paquet EAP dans un attribut particulier de Radius : *EAP-Message* ;
- Sur le serveur Radius, un module spécifique décapsule la valeur de l'attribut EAP-Message et l'interprète en suivant le modèle de couches d'EAP.

On peut donc avoir, par exemple, à l'arrivée au serveur Radius : un paquet MS-CHAPv2, encapsulé dans de l'EAP, encapsulé dans un tunnel TTLS (PEAP), encapsulé dans de l'EAP, encapsulé dans du Radius(encapsulé dans de l'UDP, de l'IP et du 802).