Projet : pLama

1/ Décomposition du segment

Header , non chiffré : (0x00 - 0x08)

4 octets : (0x00 - 0x03)

Port source (P.Src)	Port destination (P.Dst)

1 octet : (0x04)

Chiffré	1st	Lst	ACK	M.Msb	M.Lsb	Err.Msb	Err.Lsb

4 octets : (0x05-0x08)

ID du segment (ID)

Data , possiblement chiffré : (0x09-0x4A)

---------- Chiffré avec la clé 1 ----------

1 octet : (0x09)

Nb Bytes de completions (COMP)

31 octets: (0x0A - 0x29)

---------- Chiffré avec la clé 2 ----------

32 octets : (0x2A - 0x4A)

1.1/ Détails

Flags , 1 pour VRAI et 0 pour FAUX

C	.	.	.	.	.	.	.	Précise si la donnée est chiffrée ou non , fixé au début de la communication
.	F	.	.	.	.	.	.	Premiers pacquets de la connexion , correspond au premier échange de clés
.	.	L	.	.	.	.	.	Derniers pacquets de la connexion , correspond à la rupture de connexion
.	.	.	A	.	.	.	.	Acknowledgment
.	.	.	.	W	X	.	.	Mode de communication , fixé au début de la communication
.	.	.	.	.	.	Y	Z	Code d'erreur

Erreurs,

WX décrivent un code d'erreur en base 2 , la correspondance code – signification nous donne :

W	X	Code	Signification
0	0	Roule Ma Poule (RMP)	Aucun problème
0	1	A l'Head (AH)	Erreur dans le header ou dans les clés
1	0	J'ai une banane coincé dans l'oreille (Banana)	Refusé
1	1	Format puant (« Fromage »)	Le paquet ne fais pas 74 octets

Modes,

YZ décrivent un mode en base 2 , la correspondance mode – impact nous donne :

Y	Z	Mode	Impact	Data totale transmissible max (DM)
0	0	Tout Petit (TP)	ID codé sur 1 octet	16,254 Ko
0	1	Petit (P)	ID codé sur 2 octets	4,128 Mo
1	0	Grand (G)	ID codé sur 3 octets	1056,96 Mo
1	1	Très Grand (TG)	ID codé sur 4 octets	270,58 Go

Chaque paquet contient 63o de data , il suffit de calculer $(2+2^{8*(mode+1)})*63$

2/ Fonctionnement Global

2.1/Chiffrement :

Le chiffrement est une option que le client (celui qui envoi) choisi d'activer ou non.
Si C = 0 alors la data restera en clair et les premiers échanges se feront avec data à 0.
Sinon , le client et le serveur procederont à un Échange de clés Diffie-Hellman basé sur les courbes elliptiques (le domaine est sec256k1) et 2 clés de 256 bits seront générées à l'issu de l'échanges , 1 par moitié de data , on les appelera K_0 . La fonction de chiffrement utilisé sera simplement un xor de la data avec la clé .Les clés sont à usage unique , on forge donc de nouvelles clés à chaque

nouveau paquet mais par soucis de sécurité on ne fera que dériver K_{n-1} , par récurrence on a :

$$K_n = K_0^{(n)}$$

→ Vulnérabilité à une MiM , si une personne intecepte le 1er échange .Possibilité pour rendre cette attaque inefficace: au préalable , le client dispose d'une clé publique (RSA) du serveur et lors du premier echange le serveur signe sa data.

2.2/ Comportements

Les flags 1st es Lst sont utilisé indépendament de l'ID (le premier paquet de donnée porte l'ID 0) et uniquement pour signifier une début de connexion (échange de clés) ou une fin de connexion (dernier paquet).

Cas parfait :

---------- Début de connexion ----------

Client :

C	1	0	0	W	X	0	0

ID = 0

Serveur :

C	1	0	1	W	X	0	0

ID = 0

---------- Fin de l'échange de clés ----------

Client :

.	0	0	0	.	.	0	0

ID = 0

Serveur :

.	0	0	1	.	.	0	0

ID = 0

.
.
.
.

Client :

.	0	1	0	.	.	0	0

ID = n

Serveur :

.	0	1	1	.	.	0	0

ID = n

---------- Fin de connexion ----------

- Le choix du mode et du chiffrement (en clair ou pas) est défini lors de l'initialisation , ensuite ils ne sont plus importants.
- La rupture de connexion est indiqué par ACK.Lst , RMP si la donnée a été traité , Banana sinon.
- Timeout au bout de 30sec.
- Seule des paquet de 74o sont autorisés , sinon erreur « Fromage »
-

Pendant l'échange de clé :

Erreurs :

- 1st à 0 → Err « AH »
- ID > 0 → Err « Banana »
- Les clés ne sont pas viables → Err « Banana »

Pendant l'échange de données :

En fonction du mode , pour l'ID , une partie du header ne sera pas lu , ainsi en mode TP , seulement l'octet 0x08 sera lu.
Le serveur lance un timer , et attend d'avoir reçu les paquet des ID 0-A , puis il les traite (déchiffre si besoin) et reset le timer puis fais la même avec les 10 paquets suivants.Soit il accepte les paquet (ACK.RMP) soit il les refuse (Err AH+Banana+Fromage)

Erreurs :

- 1st à 1 → Err «AH»
- Un ID différent de celui attendu → Err «Banana»
- Si le serveur reçoit 2 fois un paquet avec le même ID (qui ne provoque pas une erreur) il garde que le dernier.