Cours 5. Codes d'authentification de message

HAI709I – Cryptographie

Bruno Grenet

Université de Montpellier – Faculté des Sciences

Chiffrement et authentification

Chiffrement symétrique (en résumé)

- Un Expéditeur E veut envoyer un message m à un Destinataire D
- E et D possèdent une clef secrète partagée k
- E calcule $c \leftarrow \mathrm{Enc}_k(m)$ et envoie c
- D rençoit c et calcule $m \leftarrow \mathrm{Dec}_k(c)$
- Un attaquant $\mathcal{A}$ qui voit c a une probabilité négligeable de calculer m (avec plusieurs définitions possibles de sécurité)

Problème

- $\mathcal{A}$ pourrait intercepter c et envoyer $c' \neq c$
- Exemple : si Enc_k est basé sur $\oplus$, modifier un bit de c modifie un bit de m

$\rightarrow$ Comment assurer que le chiffré c a bien été envoyé par E ?

Solution

- Calculer une *étiquette* t associée au message m, infalsifiable

Définition d'un code d'authentification de message

Définition

Un **code d'authentification de message** (MAC) est un triplet d'algorithmes polynomiaux $(\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ où

- Gen prend en entrée 1^n et renvoie une clef k
- Mac prend en entrée k et un message $m \in \{0,1\}^*$ et renvoie une *étiquette* t
- Vrfy prend en entrée k, m et t et renvoie 1 si l'étiquette est *valide*, et 0 sinon.

Un MAC est à **longueur fixée $\ell(n)$** s'il n'accepte que des messages de longueur $\ell(n)$.

Remarques

- Le MAC traite *a priori* des messages non chiffrés
- Gen est probabiliste, Mac peut l'être, et Vrfy ne l'est pas
- Si Mac est déterministe, $\mathrm{Vrfy}_k(m,t)$ est simplement le test « $\mathrm{Mac}_k(m) = t$? »

Sécurité d'un MAC

Un MAC est sûr si un attaquant ne peut pas créer d'étiquette valide.

Expérience de sécurité

Entrée : paramètre de sécurité 1^n

1. Protocole : $k \leftarrow \text{Gen}(1^n)$
2. Attaquant : calcule $\text{Mac}_k(m_i)$ pour $m_1, \ldots, m_t$ qu'il choisit comme il veut
3. Attaquant : produit un couple (m, t), où $m \notin \{m_1, \ldots, m_t\}$

Succès de l'attaquant si $\text{Vrfy}_k(m, t) = 1$

Définition

Un MAC est existentiellement infalsifiable par une attaque adaptative à clairs choisis si pour tout APP, $\Pr[\text{Vrfy}_k(m, t) = 1] \leq \text{negl}(n)$.

Attaques sur des MAC

L'attaque par rejeu

- Si $\mathcal{A}$ voit passer (m, t), il pourra toujours *rejouer* (m, t), càd renvoyer le même message, authentifié
- Les MAC ne sont pas faits pour se prémunir de ce genre d'attaque
- S'en protéger dépend des applications. Possibilités :
  - Ajouter un horodatage au message : $t \leftarrow \mathrm{Mac}_k(m \| h)$ où h est l'heure courante
  - Ajouter un compteur de message : $t \leftarrow \mathrm{Mac}_k(m \| cpt)$

L'attaque par temps de calcul

- Hypothèses :
  - $\mathrm{Vrfy}_k(m, t)$ calcule $t' \leftarrow \mathrm{Mac}_k(m)$ puis teste si $t = t'$
  - Le test $t = t'$ termine dès qu'on a trouvé i tel que $t_{[i]} = t'_{[i]}$ `strncmp`
- Pour $i = 0$ à $n - 1$ (longueur de t en octet),
  - on teste les couples $(m, t^0), \ldots, (m, t^{255})$ où $t^j = t_{[0,i[} \| j \| 0 \ldots 0$
  - en mesurant les temps de calcul, on trouve j
- Attaque utilisée sur la Xbox 360

Chiffrement authentifié

On sait chiffrer un message, et l'authentifier. Comment combiner les deux ?

Chiffrer-et-authentifier — SSH

- $m \mapsto (c, t)$ où $c = \mathrm{Enc}_{k_E}(m)$ et $t = \mathrm{Mac}_{k_M}(m)$
- Pas sûr : t peut dévoiler de l'information sur m

Authentifier-puis-chiffrer — SSL

- $m \mapsto c$ où $c = \mathrm{Enc}_{k_E}(m \| t)$ et $t = \mathrm{Mac}_{k_M}(m)$
- Pas sûr :
  - à la réception : $m \| t \leftarrow \mathrm{Dec}_{k_E}(c)$ puis $\mathrm{Vrfy}_{k_M}(m, t)$
  - le déchiffrement peut renvoyer une erreur $\rightarrow$ attaque *par bourrage*

Chiffrer-puis-authentifier — IPsec (VPN)

- $m \mapsto (c, t)$ où $c = \mathrm{Enc}_{k_E}(m)$ et $t = \mathrm{Mac}_{k_M}(c)$
- Sûr (si le chiffrement et le MAC sont sûrs) !

1. Définition et sécurité

2. Construction de MACs

MAC de longueur fixée

Construction

- Primitive : fonction pseudo-aléatoire $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$
- Mac : sur l'entrée k, m, renvoie $F_k(m)$
- Vrfy : sur l'entrée k, m, t, teste si $t = F_k(m)$

Théorème

Si F est pseudo-aléatoire, alors (Mac, Vrfy) est un MAC sûr

MAC de longueur quelconque

MAC de longueur fixée $\rightarrow$ MAC de longueur quelconque ?

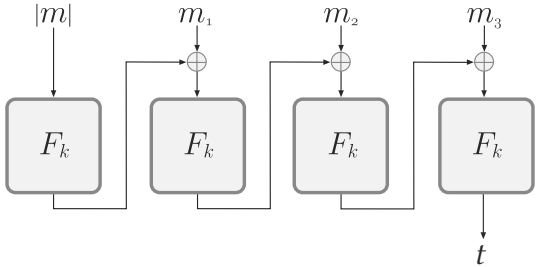
Mauvaise idée

- $m = m_1 \| \cdots \| m_s \rightarrow t = t_1 \| \cdots \| t_s = \mathrm{Mac}_k(t_1) \| \cdots \| \mathrm{Mac}_k(t_s)$
- Problèmes similaires au mode ECB pour le chiffrement
- Ex.: si on a $(m_1^0 \| m_2^0, t_1^0 \| t_2^0)$ et $(m_1^1 \| m_2^1, t_1^1 \| t_2^1)$, on crée $(m_1^0 \| m_2^1, t_1^0 \| t_2^1)$

Idée correcte

- Ajouter de l'info à **chaque** bloc avant de calculer t_i :
  - numéro i du bloc $\rightarrow$ empêcher de changer l'ordre
  - longueur ℓ du message $\rightarrow$ empêcher de tronquer le message
  - identifiant unique r (aléatoire) $\rightarrow$ empêcher les *recombinaisons*
- Solution : $t_i \leftarrow \mathrm{Mac}_k(r \| \ell \| i \| m_i)$
  - Peut être montré sûr
  - Nécessite de diminuer la longueur des blocs
  - Lourd à mettre en œuvre

CBC-MAC : directement de longueur quelconque



Théorème (difficile)

Si F est pseudo-aléatoire, alors CBC-MAC est sûr

Hash-and-MAC

Utiliser une fonction de hachage pour passer d'un MAC de longueur fixée à un MAC de longueur quelconque

Construction

- Primitives :
  - $(\mathsf{Mac}, \mathsf{Vrfy})$: MAC pour les messages de longueur $\ell(n)$
  - (Gen, H) : fonction de hachage avec taille de sortie $\ell(n)$
- MAC $(\mathsf{Gen}', \mathsf{Mac}', \mathsf{Vrfy}')$ pour les messages de longueur quelconque :
  - $\mathsf{Gen}'(1^n)$: tire $k \in \{0,1\}^n$ uniformément, et $s \leftarrow \mathsf{Gen}(1^n)$
  - $\mathsf{Mac}'_{k,s}(m)$: renvoie $\mathsf{Mac}_k(H^s(m))$
  - $\mathsf{Vrfy}'_{k,s}(m,t)$: renvoie $\mathsf{Vrfy}_k(H^s(m),t)$

Théorème

Si $(\mathsf{Mac}, \mathsf{Vrfy})$ est sûr pour les messages de longueur $\ell(n)$ et que (Gen, H) est résistante aux collisions, alors $(\mathsf{Gen}', \mathsf{Mac}', \mathsf{Vrfy}')$ est sûr pour les messages de longueur quelconque.

Preuve de sécurité pour *Hash-and-MAC*

Soit $\mathcal{A}$ un attaquant contre $(Gen', Tag', Vrfy')$ et (m^*, t^*) le couple renvoyé par $\mathcal{A}$.

Cas 1 $\mathcal{A}$ a interrogé Tag'_k sur un message m tq $H^s(m) = H^s(m^*)$ $\Rightarrow$ $\mathcal{A}$ a trouvé une collision

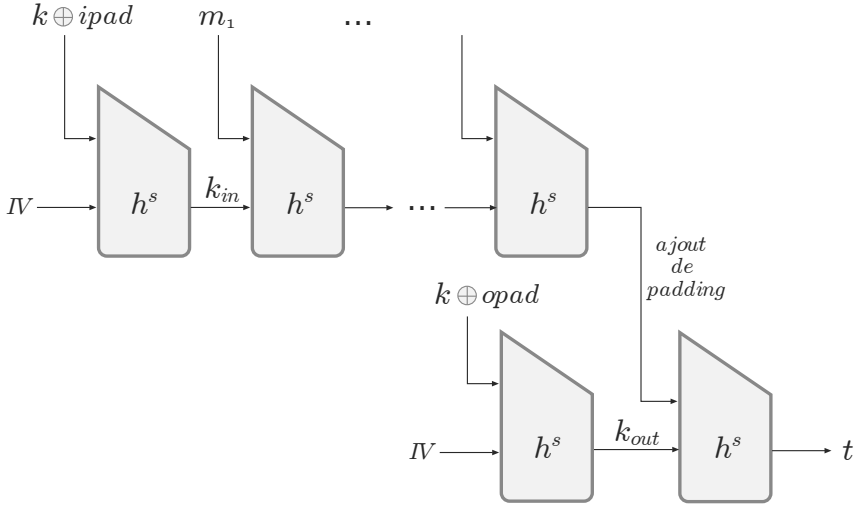
$\Rightarrow$ Proba $negl(n)$

Cas 2 $\mathcal{A}$ a interrogé Tag_k sur des éléments $\neq H^s(m^*)$

$\Rightarrow$ la proba qu'il ait trouvé $(H^s(m^*), t^*)$ valide est $negl(n)$.

$\Rightarrow$ Proba de succès $negl(n)$.

MAC purement basé sur une fonction de hachage : HMAC



Conclusion

Chiffrer ne suffit **jamais**

- Canal de communication non sécurisé mais authentifié : parfois utile
- Canal de communication sécurisé mais non authentifié : toujours inutile

Codes d'authentification de messages

- Nombreuses constructions existantes : AMAC, BMAC, CMAC, DMAC, EMAC, FMAC, GMAC, HMAC, IMAC, JMAC, KMAC, LMAC, MMAC, NMAC, OMAC, PMAC, QMAC, RMAC, SMAC, TMAC, UMAC, VMAC, WMAC, XMAC, YMAC, ZMAC, PelicanMAC, SandwichMAC
- Utilisation soit de chiffre par blocs, soit de fonctions de hachage…
- … soit les deux, mais moins pratique !

Chiffrement authentifié

- Objectif : fournir à la fois chiffrement et authentification
- Trois solutions :
  - Chiffrer-et-authentifier : attaques connues
  - Authentifier-puis-chiffrer : attaques connues
  - Chiffrer-puis-authentifier : sécurité prouvée !