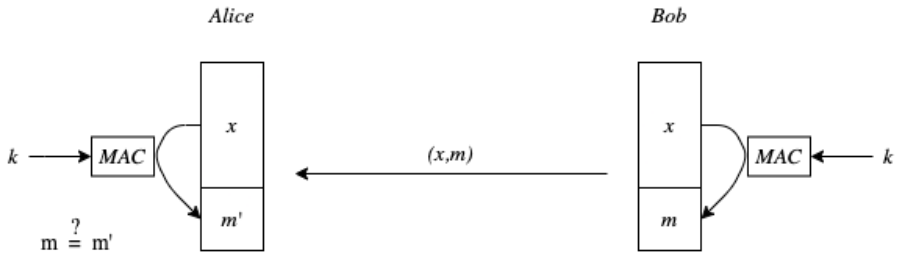
IFT3275 - Cours du 1er avril 2021

Professeur : Arnaud L'Heureux

1er avril 2021

MACs - Message Authentication Codes

$$m = MAC_k(x)$$



Bob est le seul autre à avoir la clé $k \rightarrow$ *Authenticité*.
Alice peut vérifier que le message n'a pas été changé $\rightarrow$ *Intégrité*.
Différence entre signature digitale et MAC: pas de non-répudiation. Alice sait que c'est Bob qui lui a envoyé le message, mais un parti tierce ne pourrait pas le savoir.

1 Propriétés du MAC

- *Checksum* cryptographique.
- Symmétrique: Les partis signature/vérification ont la même clé.
- Input de taille arbitraire. Output de taille fixe.
- Intégrité: Toute manipulation du message en transit sera détectée.
- Authenticité/Authentification: Le parti qui reçoit le message (le vérificateur du MAC) est assuré de l'origine du message.
- PAS de non-répudiation, car on n'utilise pas de clé privée (k symmétrique $\rightarrow$ les deux partis peuvent forger un MAC).

2 Comment créer un MAC

On a besoin d'une clé et d'un message.
Deux manières intuitives:

1. Chiffrement par blocs
2. Hachage

2.1 Hachage (HMAC)

Exemple d'utilisation: HTTPS utilise le protocole TLS (Transport Layer Security) dans lequel HMAC est utilisé.
Le *input* est séparé en t blocs. On commence avec un état initial/vecteur d'initialisation (IV). À chaque itération, on *feed* un bloc d'input dans la fonction de hachage.

On regarde deux approches:

- *Secret Prefix*

$$m = MAC_k(x) = h(k||x)$$

- *Secret Suffix*

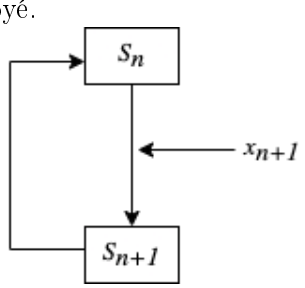
$$m = MAC_k(x) = h(x||k)$$

2.1.1 Attaque contre le Secret Prefix HMAC

Alice		*Bob*
	(x, m) $\longleftarrow$	$x = (x_1, x_2, \ldots, x_n)$ $m = h(k\|\|x_1, x_2, \ldots, x_n)$
	(x_0, m_0) $\longleftarrow$	
$m' = h(k\|\|x_1, x_2, \ldots, x_{n+1})$ $m' = m_0$	Eve ajoute un message à la fin de x: $x_0 = (x_1, \ldots, x_n, x_{n+1})$ $m_0 = h(m\|\|x_{n+1})$	

La fonction de hachage ne requiert que S_n et le prochain x_i pour obtenir l'état S_{n+1}. On peut donc concaténer x_{n+1} à la fin du message que Bob a envoyé.

$$m_0 = h(\underbrace{k||x_1, \ldots, x_n}_{m=S_n}||x_{n+1})$$



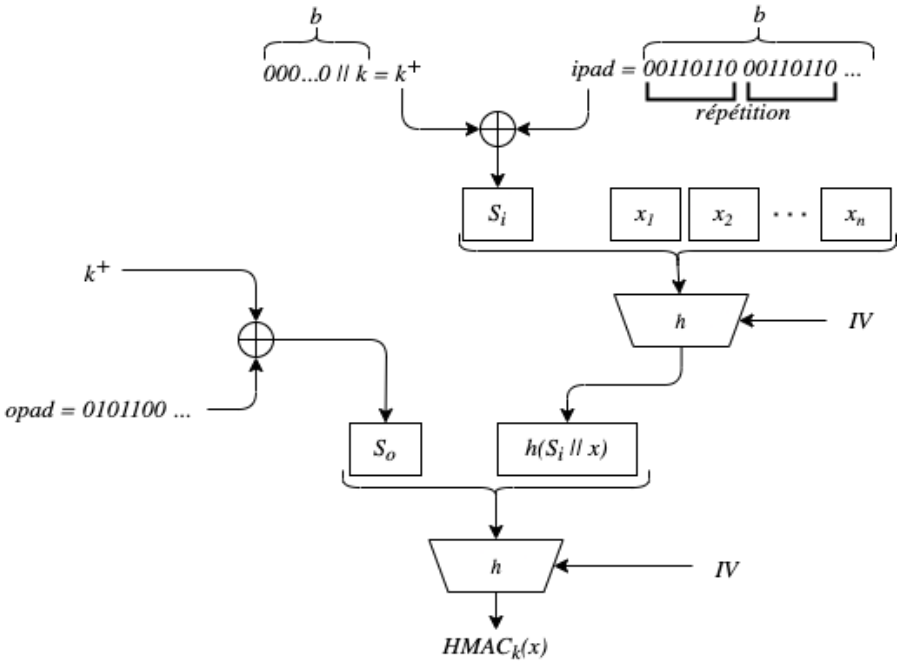
L'attaque ne fonctionne pas contre le *secret suffix*:
$m_0 = h(x_1, x_2, \ldots x_n||k||x_{n+1})$
Alice vérifie $h(x_0||k) \neq m_0$.

2.1.2 Attaque contre le Secret Suffix HMAC

$m = h(x||k)$
$m = h(x||k) = h(x_0||k)$ x est fixé, trouver x_0: $\rightarrow$ **collision faible**.

2.1.3 Solution aux attaques

Une encryption en 2 étapes avec *inner-* et *outerpad*:



2.2 Chiffrement par blocs (CBC-MAC)

Comment utiliser le chiffrement symmétrique pour avoir une sortie de taille fixe pour une entrée de taille arbitraire: