

CRYPTA

Ludovic Perret

Sorbonne Universités, UPMC Univ Paris 06, INRIA Paris
LIP6, PolSyS Project, Paris, France

GitHub CRYPTA : <https://github.com/lperret/EtuCrypto.git>
DRAFT – Ne pas imprimer.

2017 – 2018



Plan du cours

- 1 Cryptographie symétrique
- 2 Fonction de hachage
- 3 (In)Sécurité dans les réseaux sans fil – WEP/WPA
- 4 Internet Security Protocol – IPSec
- 5 BEAST

Plan du cours

1 Cryptographie symétrique

- RC4
- Trivium
- Salsa20
- Chiffrement par bloc – Mode opératoire

2 Fonction de hachage

3 (In)Sécurité dans les réseaux sans fil – WEP/WPA

4 Internet Security Protocol – IPSec

5 BEAST

RC4 [R. Rivest, 1987]

- Génération d'une suite chiffrante à partir d'un tableau S de 256 **octets** et d'une clef K .

Phase d'initialisation (RC4-KSA)

- Pour $i, 0 \leq i \leq 255$ faire $S[i] := i$ FinPour
- $j := 0$
- Pour $i, 0 \leq i \leq 255$ faire
 - $j := (j + S[i] + K[i \bmod \text{len}(K)]) \bmod 2^8$
 - Échanger($S[i], S[j]$)
- FinPour

RC4 [R. Rivest, 1987]

Génération de la suite chiffrante (RC4-PRGA)

- $i := 0 \quad j := 0$
- **Pour** $k, 0 \leq k \leq \text{no} - 1$ **faire**
 - $i := (i + 1) \bmod 2^8 \quad j := (j + S[i]) \bmod 2^8$
 - Échanger($S[i], S[j]$)
 - $t := (S[i] + S[j]) \bmod 2^8$
 - $\text{SuiteChiffrante}[k] := S[t]$
- **FinPour**
- **Return** SuiteChiffrante

Faiblesses de RC4 [N. J. AlFardan, D. J. Bernstein, K. G. Paterson, B. Poettering, Jacob C. N. Schuldt]

Biais sur la suite chiffrante : <http://www.isg.rhul.ac.uk/tls/>

ECRYPT II

↓←↑↑←↓

Source : <http://www.ecrypt.eu.org/stream/>

- Nouvelle génération de chiffrement à flot
- 2004 – 2008
- Révision en 2012

Profile 1 (Soft, 128 bit)	Profile 2 (Hard, 80 bit)
HC-128	Grain v1
Rabbit	MICKEY 2.0
Salsa20/12	Trivium
SOSEMANUK	

Compétition eStream



Source : <http://www.ecrypt.eu.org/stream/>

- Nouvelle génération de chiffrement par flot, 2004 – 2008
- Révision en 2012

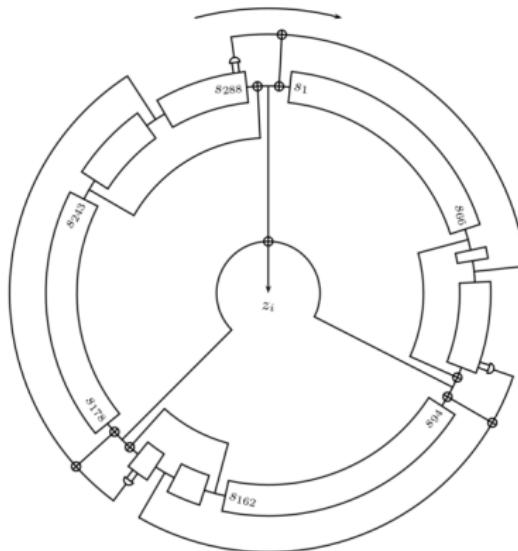
Type 1 (Soft)	Type 2 (Hard)
HC-128	Grain v1
Rabbit	MICKEY 2.0
Salsa20/12	Trivium
SOSEMANUK	

Plan du cours

1 Cryptographie symétrique

- RC4
- Trivium
- Salsa20
 - Fonction de hachage
 - Fonction d'expansion
 - Suite Chiffrante
- Chiffrement par bloc – Mode opératoire

Trivium (Christophe De Cannière, Bart Preneel)



Source : <http://www.ecrypt.eu.org/stream/e2-trivium.html>

- Clé de 80 bit, registre de 288 bit
- 3 registres de 93, 84 et 111 bit
- Suite chiffrante de taille $\leq 2^{64}$ (pour une clé secrète)

Description de Trivium

Principe

- Inti. clé (80 bit) et IV (80 bit)
- Génération de flot

for $i := 1$ to N do

$$t_1 := s_{66} + s_{93}$$

$$t_2 := s_{162} + s_{177}$$

$$t_3 := s_{243} + s_{288}$$

$$z_i := t_1 + t_2 + t_3$$

Description de Trivium

Principe

- Inti. clé (80 bit) et IV (80 bit)
- Génération de flot

for $i := 1$ to N do

$$t_1 := s_{66} + s_{93}$$

$$t_2 := s_{162} + s_{177}$$

$$t_3 := s_{243} + s_{288}$$

$$Z_i := t_1 + t_2 + t_3$$

$$t_1 := s_{66} + s_{93} + s_{91}s_{92} + s_{171}$$

$$t_2 := s_{162} + s_{177} + s_{175}s_{176} + s_{264}$$

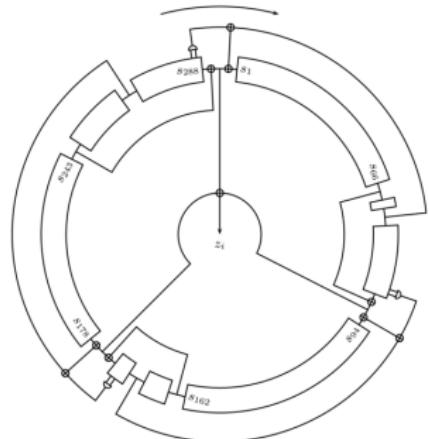
$$t_3 := s_{243} + s_{288} + s_{286}s_{287} + s_{69}$$

$$(s_1, s_2, \dots, s_{93}) := (\textcolor{red}{t}_3, s_1, \dots, s_{92})$$

$$(s_{94}, s_{95}, \dots, s_{177}) := (\textcolor{red}{t}_1, s_{94}, \dots, s_{176})$$

$$(s_{178}, s_{279}, \dots, s_{288}) := (\textcolor{red}{t}_2, s_{178}, \dots, s_{287})$$

end for



Description de Trivium – Initialisation

$(s_1, s_2, \dots, s_{93}) := (K_1, K_2, \dots, K_{80}, 0, \dots, 0)$

$(s_{94}, s_{95}, \dots, s_{177}) := (\text{IV}_1, \text{IV}_2, \dots, \text{IV}_{80}, 0, \dots, 0)$

$(s_{178}, s_{279}, \dots, s_{288}) := (0, 0, \dots, 1, 1, 1)$

for $i := 1$ to 4×288 do

$t_1 := s_{66} + s_{93}$

$t_2 := s_{162} + s_{177}$

$t_3 := s_{243} + s_{288}$

$z_i := t_1 + t_2 + t_3$

Description de Trivium – Initialisation

$(s_1, s_2, \dots, s_{93}) := (K_1, K_2, \dots, K_{80}, 0, \dots, 0)$
 $(s_{94}, s_{95}, \dots, s_{177}) := (\text{IV}_1, \text{IV}_2, \dots, \text{IV}_{80}, 0, \dots, 0)$
 $(s_{178}, s_{279}, \dots, s_{288}) := (0, 0, \dots, 1, 1, 1)$

for $i := 1$ to 4×288 do

$$t_1 := s_{66} + s_{93}$$

$$t_2 := s_{162} + s_{177}$$

$$t_3 := s_{243} + s_{288}$$

$$z_i := t_1 + t_2 + t_3$$

$$t_1 := t_1 + s_{91}s_{92} + s_{171}$$

$$t_2 := t_2 + s_{175}s_{176} + s_{264}$$

$$t_3 := t_3 + s_{286}s_{287} + s_{69}$$

$$(s_1, s_2, \dots, s_{93}) := (\textcolor{red}{t_3}, s_1, \dots, s_{92})$$

$$(s_{94}, s_{95}, \dots, s_{177}) := (\textcolor{red}{t_1}, s_{94}, \dots, s_{176})$$

$$(s_{178}, s_{279}, \dots, s_{288}) := (\textcolor{red}{t_2}, s_{178}, \dots, s_{287})$$

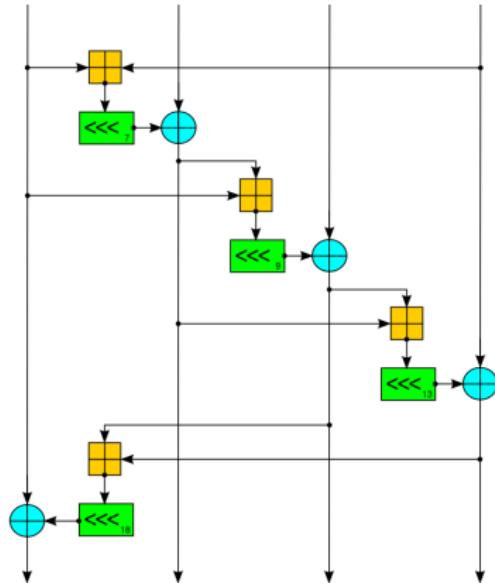
end for

Plan du cours

1 Cryptographie symétrique

- RC4
- Trivium
- **Salsa20**
 - Fonction de hachage
 - Fonction d'expansion
 - Suite Chiffrante
- Chiffrement par bloc – Mode opératoire

Salsa20 (Dan Bernstein)



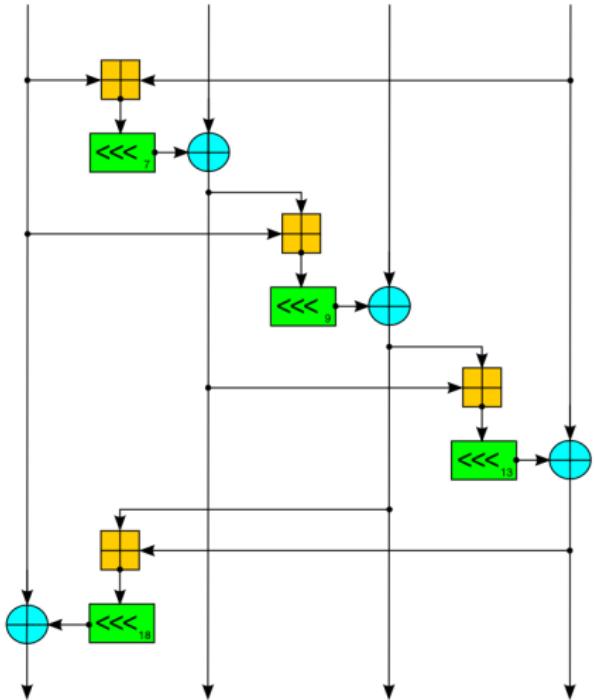
- ARX (Add, Rotate, XOR)
- Clé de 128 ou 256 bit
- Hachage en mode compteur

Plan du cours

1 Cryptographie symétrique

- RC4
- Trivium
- Salsa20
 - Fonction de hachage
 - Fonction d'expansion
 - Suite Chiffrante
- Chiffrement par bloc – Mode opératoire

Quarterround



$$y_0, y_1, y_2, y_3 \in \{0, \dots, 2^{32} - 1\}$$

$$\begin{aligned}z_1 &:= y_1 \oplus ((y_0 \boxplus y_3) \lll 7) \\z_2 &:= y_2 \oplus ((z_1 \boxplus y_0) \lll 9) \\z_3 &:= y_3 \oplus ((z_2 \boxplus z_1) \lll 13) \\z_0 &:= y_0 \oplus ((z_3 \boxplus z_2) \lll 18)\end{aligned}$$

- \oplus : addition modulo 2
- \boxplus : addition modulo 2^{32}
- \lll : rotation à gauche

RowRound

$$y_0, y_1, y_2, y_3, \dots, y_{15} \in \{0, \dots, 2^{32} - 1\}$$

$$(z_0, z_1, z_2, z_3) := \text{quarterround}(y_0, y_1, y_2, y_3)$$

$$(z_5, z_6, z_7, z_4) := \text{quarterround}(y_5, y_6, y_7, y_4)$$

$$(z_{10}, z_{11}, z_8, z_9) := \text{quarterround}(y_{10}, y_{11}, y_8, y_9)$$

$$(z_{15}, z_{12}, z_{13}, z_{14}) := \text{quarterround}(y_{15}, y_{12}, y_{13}, y_{14})$$

$$\begin{pmatrix} y_0 & y_1 & y_2 & y_3 \\ y_4 & y_5 & y_6 & y_7 \\ y_8 & y_9 & y_{10} & y_{11} \\ y_{12} & y_{13} & y_{14} & y_{15} \end{pmatrix}$$

ColumnRound

$$x_0, x_1, x_2, x_3, \dots, x_{15} \in \{0, \dots, 2^{32} - 1\}$$

$$(y_0, y_4, z_8, y_{12}) := \text{quarterround}(x_0, x_4, x_8, x_{12})$$

$$(y_5, y_9, y_{13}, y_1) := \text{quarterround}(x_5, x_9, x_{13}, x_1)$$

$$(y_{10}, y_{14}, y_2, y_6) := \text{quarterround}(x_{10}, x_{14}, x_2, x_6)$$

$$(y_{15}, y_3, y_7, y_{11}) := \text{quarterround}(x_{15}, x_3, x_7, x_{11})$$

$$\begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix}$$

Hachage Salsa20

$x = (x_0, x_1, x_2, \dots, x_{15})$ avec $x_i \in \{0, \dots, 2^{32} - 1\}$

$$\text{HachSalsa20}(x) = x + \text{doubleround}^{12}(x),$$

avec $\text{doubleround}(x) = \text{rowround}(\text{columnround}(x))$.

Hachage Salsa20

$x = (x_0, x_1, x_2, \dots, x_{15})$ avec $x_i \in \{0, \dots, 2^{32} - 1\}$

$$\text{HachSalsa20}(x) = x + \text{doubleround}^{12}(x),$$

avec $\text{doubleround}(x) = \text{rowround}(\text{columnround}(x))$.

$$\text{HachSalsa20} : 16 \cdot 32 = 512 \text{ bit} \rightarrow 16 \cdot 32 = 512 \text{ bit}$$

Plan du cours

1 Cryptographie symétrique

- RC4
- Trivium
- **Salsa20**
 - Fonction de hachage
 - **Fonction d'expansion**
 - Suite Chiffrante
- Chiffrement par bloc – Mode opératoire

Expansion Salsa20

k_0, k_1, n des séquences de $16 \cdot 8 = 128$ bit :

$$\text{ExpSalsa20}_{k_0, k_1}(n) = \text{HachSalsa20}(\sigma_0, k_0, \sigma_1, n, \sigma_2, k_1, \sigma_3)$$

$\sigma_0, \sigma_1, \sigma_2, \sigma_3$ des séquences fixées de $4 \cdot 8$ bit.

$$\text{ExpSalsa20} : 256 \times 128 \text{ bit} \rightarrow 16 \cdot 32 = 512 \text{ bit}$$

Expansion Salsa20

k_0, k_1, n des séquences de $16 \cdot 8 = 128$ bit :

$$\text{ExpSalsa20}_{k_0, k_1}(n) = \text{HachSalsa20}(\sigma_0, k_0, \sigma_1, n, \sigma_2, k_1, \sigma_3)$$

$\sigma_0, \sigma_1, \sigma_2, \sigma_3$ des séquences fixées de $4 \cdot 8$ bit.

$$\text{ExpSalsa20} : 256 \times 128 \text{ bit} \rightarrow 16 \cdot 32 = 512 \text{ bit}$$

k, n des séquences $16 \cdot 8 = 128$ bit :

$$\text{ExpSalsa20}_k(n) = \text{HachSalsa20}(\tau_0, k, \tau_1, n, \tau_2, k, \tau_3)$$

$\tau_0, \tau_1, \tau_2, \tau_3$ des séquences fixées de $4 \cdot 8$ bit.

$$\text{ExpSalsa20} : 128 \times 128 \text{ bit} \rightarrow 16 \cdot 32 = 512 \text{ bit}$$

Plan du cours

1 Cryptographie symétrique

- RC4
- Trivium
- **Salsa20**
 - Fonction de hachage
 - Fonction d'expansion
 - **Suite Chiffrante**
- Chiffrement par bloc – Mode opératoire

Chiffrement Salsa20

- k une séquence de $16 \cdot 8 = 128$ bit ou $32 \cdot 8 = 256$ bit
- v une séquence de $8 \cdot 8 = 64$ bit :

$$\text{Salsa20}_k(v) = [(\text{ExpSalsa20}_k(v, i)) | i \in \{0, \dots, 2^{64} - 1\}],$$

avec $\underline{i} = (i_0, i_1, \dots, i_7)$ une séquence de $8 \cdot 8$ bit tel que

$$i = i_0 + 2^8 i_1 + \dots + 2^{56} i_7.$$

Chiffrement Salsa20

- k une séquence de $16 \cdot 8 = 128$ bit ou $32 \cdot 8 = 256$ bit
- v une séquence de $8 \cdot 8 = 64$ bit :

$$\text{Salsa20}_k(v) = [(\text{ExpSalsa20}_k(v, i)) | i \in \{0, \dots, 2^{64} - 1\}],$$

avec $\underline{i} = (i_0, i_1, \dots, i_7)$ une séquence de $8 \cdot 8$ bit tel que

$$i = i_0 + 2^8 i_1 + \dots + 2^{56} i_7.$$

m une séquence de ℓ octets, avec $\ell \in \{0, \dots, 2^{70}\}$.

$$c = m \oplus \text{Salsa20}_k(v).$$

Chiffrement Salsa20

- k une séquence de $16 \cdot 8 = 128$ bit ou $32 \cdot 8 = 256$ bit
- v une séquence de $8 \cdot 8 = 64$ bit :

$$\text{Salsa20}_k(v) = [(\text{ExpSalsa20}_k(v, i)) | i \in \{0, \dots, 2^{64} - 1\}],$$

avec $\underline{i} = (i_0, i_1, \dots, i_7)$ une séquence de $8 \cdot 8$ bit tel que

$$i = i_0 + 2^8 i_1 + \dots + 2^{56} i_7.$$

m une séquence de ℓ octets, avec $\ell \in \{0, \dots, 2^{70}\}$.

$$c = m \oplus \text{Salsa20}_k(v).$$

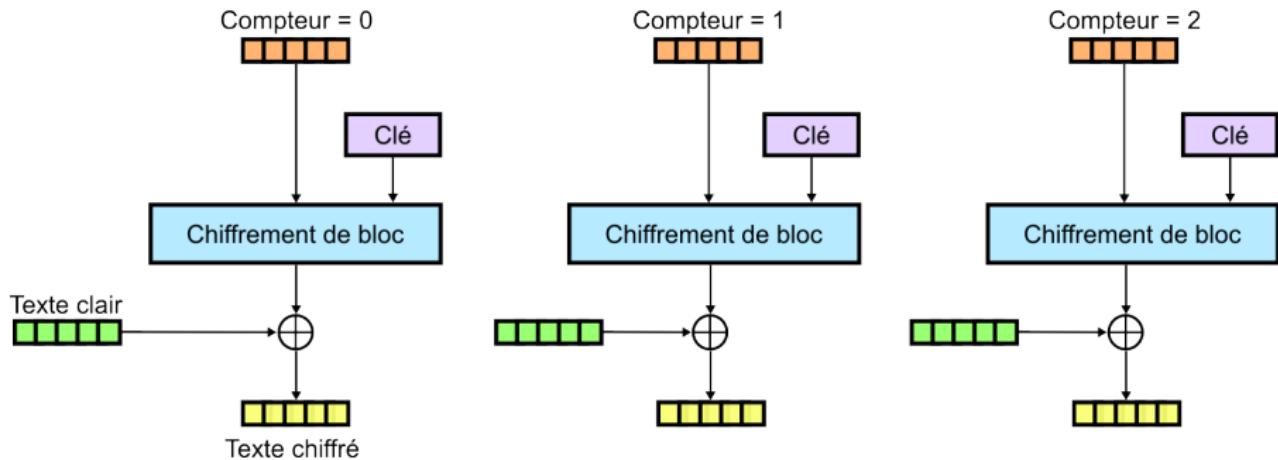
- Variante de Salsa20 (ChaCha20) dans TLS
- Fonction de hachage BLAKE/BLAKE2

Plan du cours

1 Cryptographie symétrique

- RC4
- Trivium
- Salsa20
 - Fonction de hachage
 - Fonction d'expansion
 - Suite Chiffrante
- Chiffrement par bloc – Mode opératoire

Mode CTR



Source : <https://fr.wikipedia.org/>

Plan du cours

2

Fonction de hachage

- Généralités
- Merkle-Damgård
 - SHA2
 - SHA3

Plan du cours

2

Fonction de hachage

- Généralités
- Merkle-Damgård
 - SHA2
 - SHA3

Paradoxe des anniversaires

Proposition

Soient $H : X \rightarrow Y$, x_1, \dots, x_k des éléments distincts de X tirés aléatoirement, et $y_i = H(x_i)$, pour tout i , $1 \leq i \leq k$.

$$\Pr(\exists \text{ collision}) \approx 1 - e^{-\frac{k(k-1)}{2N}}, \text{ avec } N = |Y|.$$

Paradoxe des anniversaires

Démonstration.

- On suppose que les y_i sont des éléments aléatoires de Y .

Paradoxe des anniversaires

Démonstration.

- On suppose que les y_i sont des éléments aléatoires de Y .
- Nous avons $N = |Y|$. La probabilité que $y_{i+1} \notin \{y_1, \dots, y_i\}$ est $p_{i+1} = (1 - i/N)$.

Paradoxe des anniversaires

Démonstration.

- On suppose que les y_i sont des éléments aléatoires de Y .
- Nous avons $N = |Y|$. La probabilité que $y_{i+1} \notin \{y_1, \dots, y_i\}$ est $p_{i+1} = (1 - i/N)$.
- La probabilité que les y_1, \dots, y_k – tirés dans cet ordre – soient distincts est

$$P = \prod_{i=0}^{k-1} p_{i+1} = \prod_{i=0}^{k-1} (1 - i/N).$$

Paradoxe des anniversaires

Démonstration.

- On suppose que les y_i sont des éléments aléatoires de Y .
- Nous avons $N = |Y|$. La probabilité que $y_{i+1} \notin \{y_1, \dots, y_i\}$ est $p_{i+1} = (1 - i/N)$.
- La probabilité que les y_1, \dots, y_k – tirés dans cet ordre – soient distincts est

$$P = \prod_{i=0}^{k-1} p_{i+1} = \prod_{i=0}^{k-1} (1 - i/N).$$

- La probabilité de **non-collision** est donc P .

Paradoxe des anniversaires

Démonstration.

- On suppose que les y_i sont des éléments aléatoires de Y .
- Nous avons $N = |Y|$. La probabilité que $y_{i+1} \notin \{y_1, \dots, y_i\}$ est $p_{i+1} = (1 - i/N)$.
- La probabilité que les y_1, \dots, y_k – tirés dans cet ordre – soient distincts est

$$P = \prod_{i=0}^{k-1} p_{i+1} = \prod_{i=0}^{k-1} (1 - i/N).$$

- La probabilité de **non-collision** est donc P .
- En approchant $1 - x$ par e^{-x} pour x proche de 0, on obtient :
$$P \simeq \prod_{i=0}^{k-1} e^{-\frac{i}{N}} = e^{-\frac{k(k-1)}{2N}}.$$



Collision

Proposition

Soit $H : X \rightarrow Y$ une fonction de hachage, avec $|X| \geq |Y|$ et $|Y| = N$.
Pour trouver une collision avec probabilité $\geq 1/2$, il "suffit" de hacher :

$$\mathcal{O}(\sqrt{N}) \text{ éléments de } X.$$

Autrement dit ...

Pour avoir une probabilité $\geq 1/2$ de trouver une collision, il suffit de hacher un peu plus de \sqrt{N} éléments de X .

Preuve

Démonstration.

Notons $\epsilon = 1 - P$, la probabilité d'avoir au moins une collision.
Exprimons k en fonction de ϵ et N :

$$\epsilon \simeq 1 - e^{-\frac{k(k-1)}{2N}} \Rightarrow -\frac{k(k-1)}{2N} \simeq \ln(1 - \epsilon).$$

Ainsi, $k^2 - k \simeq 2N \ln(\frac{1}{1-\epsilon})$. En ignorant le terme $-k$, on obtient :

$$k \simeq \sqrt{2N \ln \left(\frac{1}{1-\epsilon} \right)}.$$

Pour $\epsilon = 1/2$, on trouve $k \simeq 1.18 \cdot \sqrt{N}$.



Illustration

- Supposons que X est un ensemble d'individus
- Y l'ensemble des 365 jours d'une année non bissextile
- $H(x)$, le jour de l'anniversaire d'une personne de X (on suppose que X comporte plus de 365 personnes)
- On obtient $k \simeq 1.18 \cdot \sqrt{365} \simeq 1.18 \cdot 19.10 \simeq 22.5$

Plan du cours

2

Fonction de hachage

- Généralités
- Merkle-Damgård
 - SHA2
 - SHA3

Fonction de compression

Problème

Comment gérer une donnée de taille variable ?

Définition

fonction de compression : fonction qui transforme toute chaîne d'une taille fixée $r + n$ en une chaîne de taille n .

$$f : \{0, 1\}^{r+n} \mapsto \{0, 1\}^n.$$

Construction de Merle-Damgård – (I)

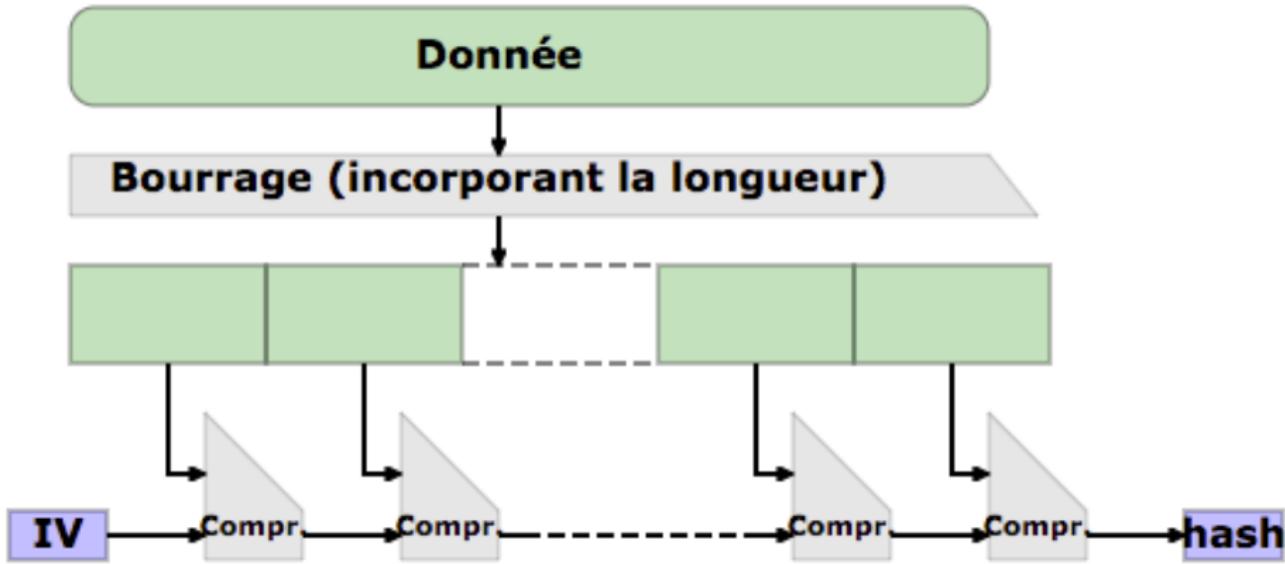
La chaîne x (de longueur arbitraire) à hacher subit un prétraitement (padding) qui la transforme en t blocs de r bits x_1, \dots, x_t .

- $\text{IV} \in \{0, 1\}^n$ une valeur initiale (ou vecteur d'initialisation),
- $f : \{0, 1\}^r \times \{0, 1\}^n \mapsto \{0, 1\}^n$ une fonction de **compression**.

On calcule :

$$H_0 = \text{IV}, \quad H_i = f(H_{i-1}, x_i), \quad 1 \leq i \leq t.$$

Merkle-Damgård



Source : <https://fr.wikipedia.org/>

Remarque

Fonction de hachage \iff Fonction de compression

Plan du cours

2

Fonction de hachage

- Généralités
- Merkle-Damgård
 - SHA2
- SHA3

SHA2 (SHA256, SHA384 et SHA512)

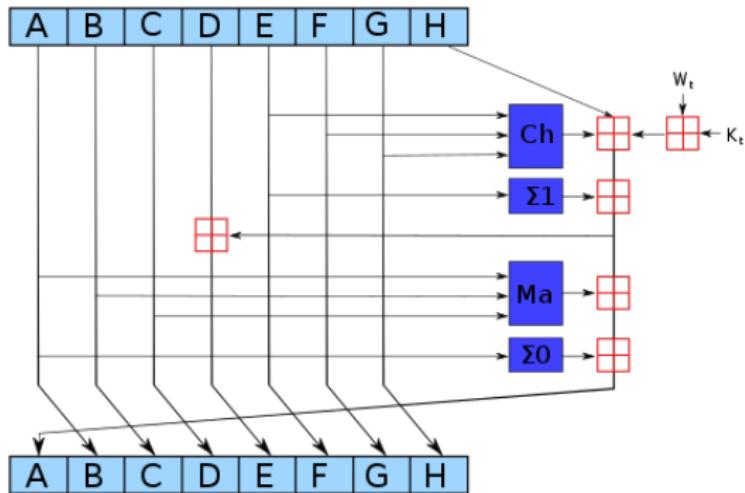
SHA = Secure Hash Algorithm

- Merkle-Damgård
- Fonction de compression ; taille des blocs $\in \{512, 1024\}$ bit
- Emprunte $\in \{224, 256, 384, 512\}$ bit

Fonction de Compression – SHA256

- Variables de chaînage de 256 bits (A, B, C, D, E, F, G, H)
- 64 étapes élémentaires (tours)
- Expansion du bloc de message
16 mots (32 bits) vers 64 mots

SHA2 – Tour



Source : <https://fr.wikipedia.org/>

$$Ch(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$$

$$Ma(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$

$$\Sigma_0(x) = \text{ROT}^2(x) \oplus \text{ROT}^{13}(x) \oplus \text{ROT}^{22}(x)$$

$$\Sigma_1(x) = \text{ROT}^6(x) \oplus \text{ROT}^{11}(x) \oplus \text{ROT}^{25}(x)$$

SHA12 – (II)

Expansion de message : $W_i = m_i, \forall i, 0 \leq i \leq 15$, et

$$W_t = \sigma_0(W_{t-2}) + W_{t-7} + \sigma_0(W_{t-15}) + W_{t-16},$$

pour $t, 16 \leq i \leq 63$.

$$\sigma_0(x) = \text{ROT}^7(x) \oplus \text{ROT}^{18}(x) \oplus \text{SHR}^3(x)$$

$$\sigma_1(x) = \text{ROT}^{17}(x) \oplus \text{ROT}^{19}(x) \oplus \text{SHR}^{10}(x)$$

Plan du cours

2

Fonction de hachage

- Généralités
- Merkle-Damgård
 - SHA2
- SHA3

Compétition SHA3



Xiaoyun Wang, Hongbo Yu.

How to Break MD5 and Other Hash Functions.
EUROCRYPT 2005.

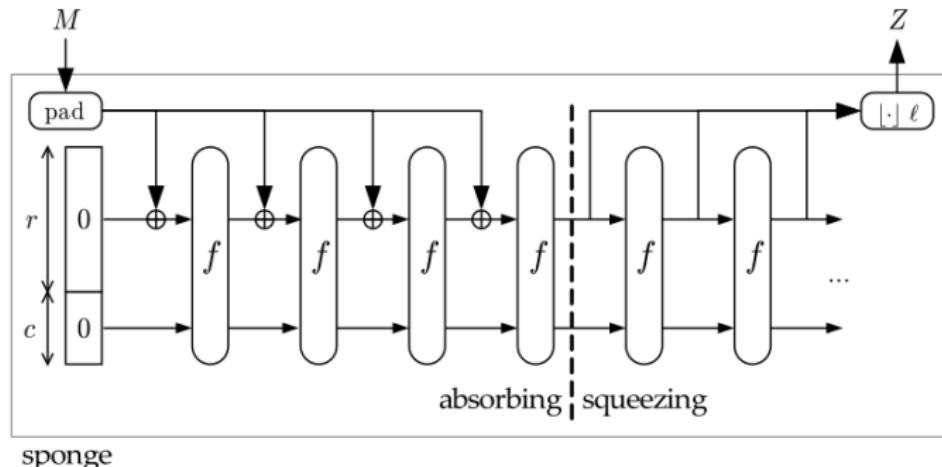
Nouveau standard

- 64 soumissions (2008)
- 14 candidats en phase 2
- 5 candidats en phase 3 (2010)

SHA3, 2012

- Keccak (G. Bertoni, J. Daemen, M. Peeters, G. Van Assche)

Construction sponge



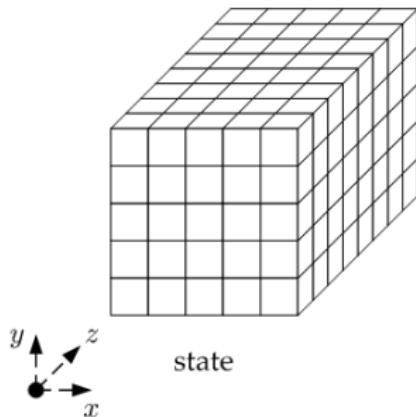
Source : https://keccak.team/sponge_duplex.html

- r , le **taux** (taille d'un bloc)
- c , la **capacité**
- f , permutation sur $b = r + c$ bit

Niveau de sécurité

$$2^{c/2}$$

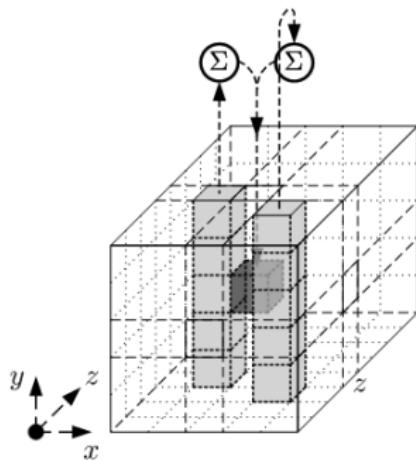
Keccak-f – Structure de donnée



Source : <https://keccak.team>

- 2^ℓ tableaux de 5×5 bit, avec $\ell \in \{1, 2, 5, 8, 16, 32, 64\}$
- $b = 25 \times 2^\ell$

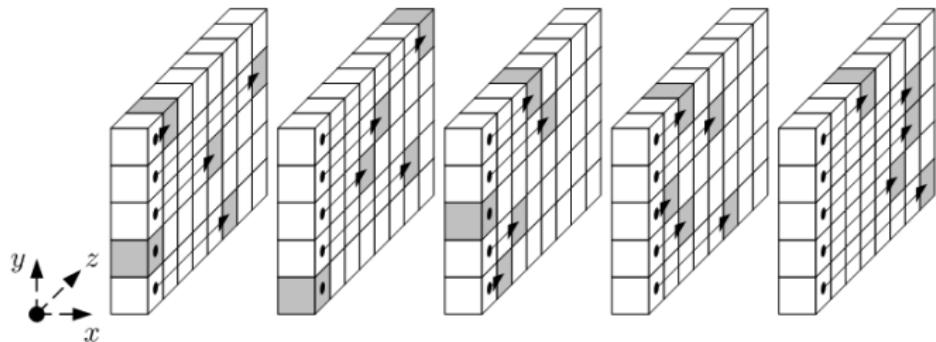
Keccak-f – Fonction θ



Source : <https://keccak.team>

$$a[i][j][k] := a[i][j][k] \oplus \sum_{j'=0}^4 a[i-1][j'][k] \oplus \sum_{j'=0}^4 a[i+1][j'][k-1]$$

Keccak-f – Fonction ρ



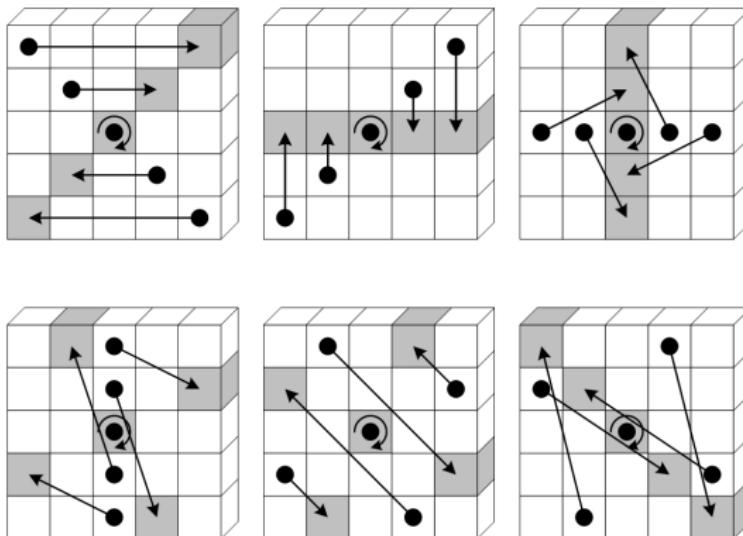
Source : <https://keccak.team>

$$a[i][j][k] := a[i][j][k - (t + 1)(t + 2)/2],$$

avec $t = -1$ si $i = j = 0$; sinon $t, 0 \leq t \leq 24$ et :

$$\binom{i}{j} \equiv \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}^t \begin{pmatrix} 0 \\ 1 \end{pmatrix} \pmod{5}.$$

Keccak-f – Fonction π

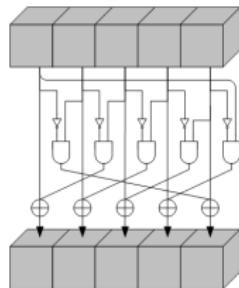


Source : <https://keccak.team>

$$a[i][j] := a[i'][j'], \text{ avec}$$

$$\binom{i}{j} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \binom{i'}{j'}$$

Keccak-f – Fonction χ



Source : <https://keccak.team>

$$a[i] := a[i] + (a[i+1] + 1)a[i+2].$$

Keccak-f

On répète $n_r = 12 + 2\ell$ fois :

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta.$$

avec ι :

$$a := a + RC[i_r].$$

Keccak-f

On répète $n_r = 12 + 2\ell$ fois :

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta.$$

avec ι :

$$a := a + RC[i_r].$$

SHA3

	sortie	r	c	Collision
SHA3-224	224	1152	448	112
SHA3-256	256	1088	512	128
SHA3-384	384	832	768	192

Plan du cours

3

(In)Sécurité dans les réseaux sans fil – WEP/WPA

- Wired Equivalent Privacy (WEP)
 - Chiffrement
 - Authentification
 - Sécurité
- Wi-Fi Protected Access (WPA/WPA2)

Plan du cours

3

(In)Sécurité dans les réseaux sans fil – WEP/WPA

- **Wired Equivalent Privacy (WEP)**
 - Chiffrement
 - Authentification
 - Sécurité
- Wi-Fi Protected Access (WPA/WPA2)

WEP

Wired Equivalent Privacy (a.k.a. *Weak Encryption Protocol*)

- **Protocole** permettant de sécuriser les réseaux sans fil de type Wi-Fi
- Fournir aux réseaux sans fil une confidentialité comparable à celle d'un réseau local filaire classique
- Norme de sécurité introduite dans IEEE 802.11 (1999)



WEP – Quelques Caractéristiques

- Utilisation d'une **clef secrète** partagée entre les entités communicantes et l'**Acces Point (AP)**
 - ▶ **Chiffrement à clef secrète (RC4)** pour assurer la confidentialité des données (**trames, paquets**)
 - ▶ **Code linéaire CRC32** pour assurer l'intégrité



CRC32

Cyclic Redundancy Check

Le contrôle de redondance cyclique (CRC) est un moyen de tester l'intégrité des données (i.e. détecter des erreurs).

- Code correcteur ; pas une primitive cryptographique

CRC32 – Encodage

- **Encodage.** Pour un CRC_n , on ajoute $n (= 32)$ bit au message (trame) à envoyer.
- $\mathbf{M}' = X^n \cdot \mathbf{M}(X)$ est le message correspondant aux bit de la trame $\mathbf{M}(X)$ auquel nous avons ajouter n zéros.

Exemple

Nous pouvons voir **0110101001** comme le polynôme :

$$0 \cdot X^9 + X^8 + X^7 + 0 \cdot X^6 + X^5 + 0 \cdot X^4 + X^3 + 0 \cdot X^2 + 0 \cdot X^1 + X^0. \\ X^8 + X^7 + X^5 + X^3 + 1.$$

CRC32 – Principe

- CRC_{M'} est le **reste de la division Euclidienne** de M'(X) par un polynôme **G(X)** fixé :

$$M'(X) = Q(X)G(X) \oplus \text{CRC}_{M'}(X),$$

avec **CRC_{M'}(X) = 0** ou $\deg(\text{CRC}_{M'}(X)) < \deg(G(X))$.

- ▶ Pour CRC32, **G(X) =**

$$x^{32} \oplus x^{26} \oplus x^{23} \oplus x^{22} \oplus x^{16} \oplus x^{12} \oplus x^{11} \oplus x^{10} \oplus x^8 \oplus x^7 \oplus x^5 \oplus x^4 \oplus x^2 \oplus x \oplus 1.$$

- On transmet :

$$M \parallel \text{CRC}_{M'}.$$

CRC32 – Linéarité

$$\begin{aligned}\mathbf{M}'(X) &= \mathbf{Q}'(X)\mathbf{G}(X) \oplus \textcolor{red}{\text{CRC}_{\mathbf{M}'}(X)}, \\ \mathbf{M}''(X) &= \mathbf{Q}''(X)\mathbf{G}(X) \oplus \textcolor{red}{\text{CRC}_{\mathbf{M}''}(X)}.\end{aligned}$$

- Nous avons $\mathbf{M}'(X) \oplus \mathbf{M}''(X)$:

$$(\mathbf{Q}'(X) \oplus \mathbf{Q}''(X))\mathbf{G}(X) \oplus (\textcolor{red}{\text{CRC}_{\mathbf{M}'}(X)} \oplus \textcolor{red}{\text{CRC}_{\mathbf{M}''}(X)}),$$

avec $\text{CRC}_{\mathbf{M}'}(X) \oplus \text{CRC}_{\mathbf{M}''}(X) = 0$ ou
 $\deg(\text{CRC}_{\mathbf{M}'}(X) \oplus \text{CRC}_{\mathbf{M}''}(X)) < \deg(G(X)).$

$$\boxed{\text{CRC}_{\mathbf{M}' \oplus \mathbf{M}''} = \text{CRC}_{\mathbf{M}'} \oplus \text{CRC}_{\mathbf{M}''}}$$

Plan du cours

3

(In)Sécurité dans les réseaux sans fil – WEP/WPA

- **Wired Equivalent Privacy (WEP)**
 - Chiffrement
 - Authentification
 - Sécurité
- Wi-Fi Protected Access (WPA/WPA2)

Caractéristiques

- Une même clef **K** (**root key**) partagée par les entités
 - ▶ 40 bits (5 octets)
 - ▶ 104 bits (13 octets)
 - ▶ 232 bits (29 octets)
- On ajoute (concatène) à cette clef un IV de **24 bit**.

K || IV, per packet key.

- **Pas de précision** (initialement) dans 802.11 sur la gestion de l'IV
 - ▶ Génération aléatoire ?
 - ▶ On incrémente l'IV ?

Chiffrement WEP

Principe

Soit \mathbf{K} la clef partagée (root key).

- **Formatage.** Soit \mathbf{M} le message (trame) à chiffrer.

$$\mathbf{M}' = \mathbf{M} \parallel \text{ICV}(\mathbf{M}), \text{ avec } \text{ICV}(\mathbf{M}) = \text{CRC32}(\mathbf{M}),$$

ICV étant Integrity Check Value.

- **Chiffrement.** On génère un $\text{IV} \in \mathbb{F}_2^{24}$ (i.e. 24 bits) et :

$$\mathbf{C} = \mathbf{M}' \oplus \text{RC4}(\mathbf{K} \parallel \text{IV}).$$

Rafraîchissement de IV

- On note :

$$\mathbf{C}_1 = \mathbf{M}'_1 \oplus \text{RC4}(\mathbf{K} \parallel \text{IV}) \quad \mathbf{C}_2 = \mathbf{M}'_2 \oplus \text{RC4}(\mathbf{K} \parallel \text{IV}).$$

- Nous avons une *collision* :

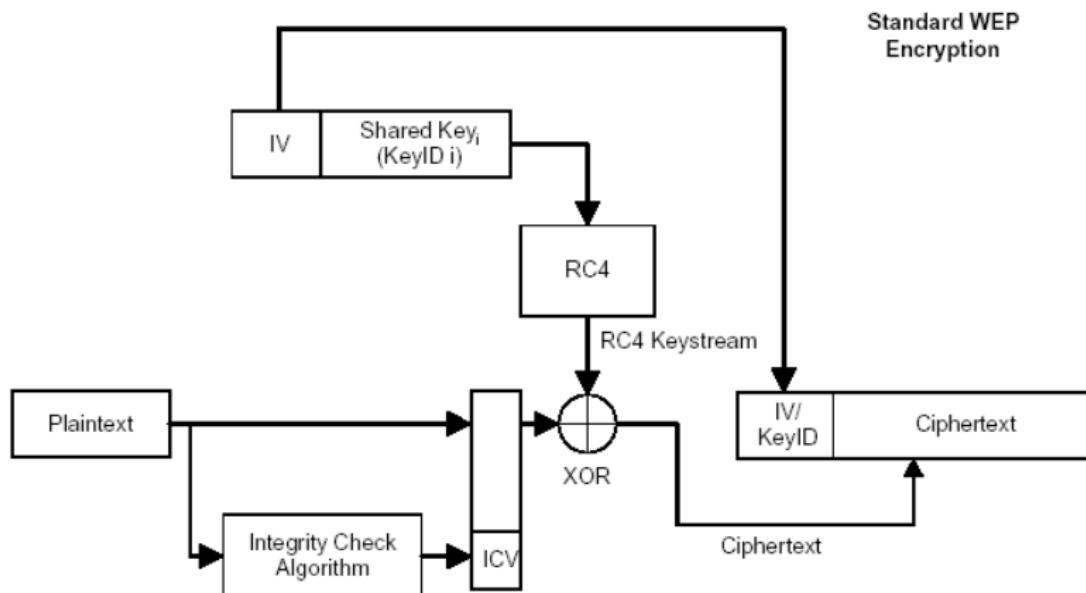
$$\mathbf{C}_1 \oplus \mathbf{C}_2 = \mathbf{M}'_1 \oplus \mathbf{M}'_2.$$

⇒ Changement de l'IV nécessaire.

Trame 802.11

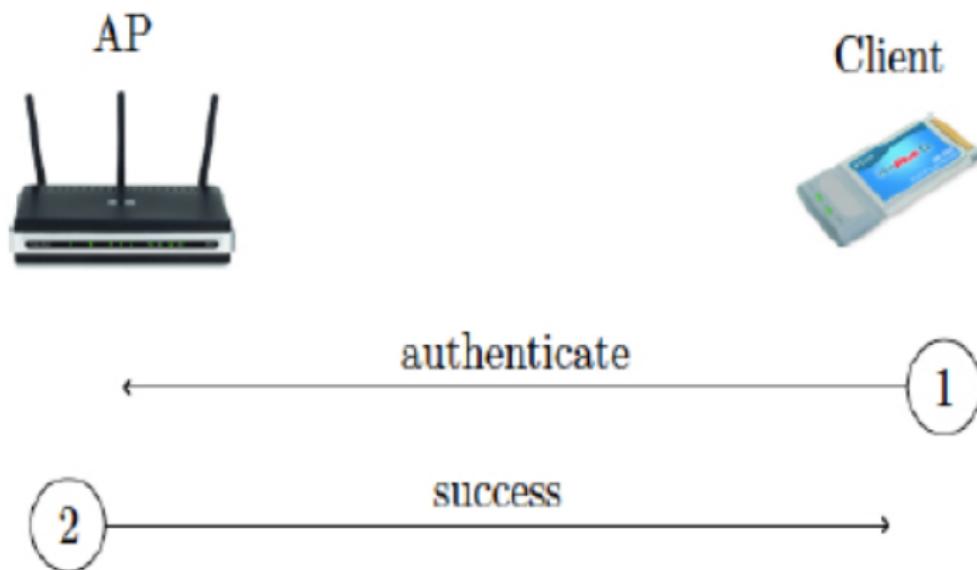


Chiffrement WEP – Résumé



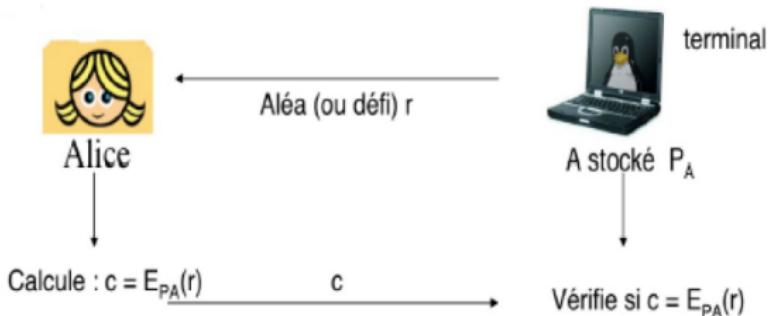
Open System Authentication – WEP

- authentification automatique



Authentification – Principe

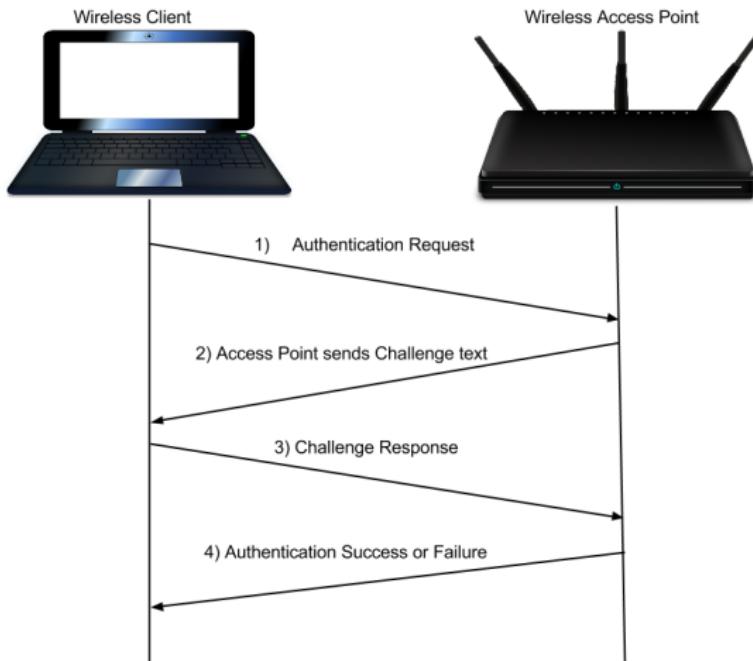
- Les entités partagent une clef secrète PA.



Shared Key Authentication

- Défi (aléa) **R** d'une taille de 128 bit.
- On chiffre l'aléa avec la méthode précédente :

$$\mathbf{R} \oplus \text{RC4}(\mathbf{K} \parallel \text{IV}).$$



Faiblesses de RC4

- Attaque FMS de S. R. Fluhrer, I. Mantin, et A. Shamir (2001).
 - ▶ Recouvrement de clé avec 9.000.000 requêtes (probabilité de succès 1/2)
- Korek-FMS (2001)
 - ▶ Recouvrement de clé avec 700.000 requêtes (probabilité de succès 1/2)
- Attaque de A. Klein (2005)
 - ▶ Recouvrement de clé avec 43.000 requêtes (probabilité de succès 1/2)
- Attaque PTW de A. Pyshkin, E. Tews, et R.-P. Weinmann (2005)
 - ▶ Recouvrement de clé avec 35.000 requêtes (probabilité de succès 1/2)
- Tornado attaque de P. Sepehrdad, S. Vaudenay, et M. Vuagnoux (2011)
- Ensemble des biais sur RC4 par N. J. AlFardan, D. J. Bernstein, K. G. Paterson, B. Poettering, Jacob C. N. Schuldt (2013).
- ...

Faiblesses du protocole

- Attaque de W. A. Arbaugh (2001)
- **Attaque Chochop de Korek (2004)**
- Attaque par fragmentation de A. Bittau, M. Handley, et J. Lackey (2006).
- ...



<https://www.aircrack-ng.org/>

Injection de paquet

Rejeu

Possible de **réinjecter** une trame chiffrée dans le réseau.

- Aucun mécanisme prévu dans WEP contre le rejeu.

Fausse authentification – SKA

Authentification sans clé

- Dans la phase d'authentification :

$$\mathbf{R} \oplus \text{RC4}(\mathbf{K} \parallel \text{IV}),$$

avec un aléa **R publique** d'une taille de 128 bit.

- Nous pouvons **extraire** $\text{RC4}(\mathbf{K} \parallel \text{IV})$.
- Pour tout challenge **R'**, nous pouvons **construire un authentifiant valide** :

$$\mathbf{R}' \oplus \text{RC4}(\mathbf{K} \parallel \text{IV}).$$

Modification de paquet

Linéarité du CRC

$$\text{ICV}(\mathbf{M}' \oplus \mathbf{M}) = \text{ICV}(\mathbf{M}) \oplus \text{ICV}(\mathbf{M}').$$

$$\text{RC4}(\mathbf{K} \parallel \text{IV}) \oplus (\mathbf{M} \oplus \mathbf{M}' \parallel \text{ICV}(\mathbf{M} \oplus \mathbf{M}'))$$

$$\text{RC4}(\mathbf{K} \parallel \text{IV}) \oplus (\mathbf{M} \oplus \mathbf{M}' \parallel \text{ICV}(\mathbf{M}) \oplus \text{ICV}(\mathbf{M}'))$$

$$\text{RC4}(\mathbf{K} \parallel \text{IV}) \oplus (\mathbf{M} \parallel \text{ICV}(\mathbf{M})) \oplus (\mathbf{M}' \parallel \text{ICV}(\mathbf{M}'))$$

⇒ Possibilité d'injecter des paquets modifiés dans le réseau.

Attaque Chochop – (I)

- Dans CRC32, $\mathbf{G}(X) =$

$$X^{32} \oplus X^{26} \oplus X^{23} \oplus X^{22} \oplus X^{16} \oplus X^{12} \oplus X^{11} \oplus X^{10} \oplus X^8 \oplus X^7 \oplus X^5 \oplus X^4 \oplus X^2 \oplus X \oplus 1.$$

- $\mathbf{M}' = \mathbf{M} \parallel \text{ICV}(\mathbf{M})$, avec $\text{ICV}(\mathbf{M}) = \text{CRC32}(\mathbf{M})$.
- Soit P_{CHECK} un polynôme de degré < 32 tel que :

$$\mathbf{M}'(X) \equiv P_{\text{CHECK}} \bmod \mathbf{G}(X).$$

Attaque Chochop – (II)

- Soit $\mathbf{P}_7(X)$ un polynôme de degré < 8 tel que :

$$\mathbf{M}'(X) = \mathbf{Q}(X) \cdot X^8 \oplus \mathbf{P}_7(X).$$

- Ainsi :

$$\mathbf{Q}(X) \cdot X^8 \equiv \mathbf{P}_7(X) \oplus P_{\text{CHECK}} \pmod{\mathbf{G}(X)}.$$

- Nous avons $\text{pgcd}(X^8, \mathbf{G}(X)) = 1$.

- Il existe donc $\mathbf{R}_{\text{Inv}} \equiv (X^8)^{-1} \pmod{\mathbf{G}(X)} =$

$$X^{31} \oplus X^{29} \oplus X^{27} \oplus X^{24} \oplus X^{23} \oplus X^{22} \oplus X^{20} \oplus X^{17} \oplus X^{16} \oplus X^{15} \oplus X^{14} \oplus X^{13} \oplus X^{10} \oplus X^9 \oplus X^7 \oplus X^5 \oplus X^2 \oplus X.$$

Attaque Chochop – (III)

- Comme $\mathbf{Q}(X) \cdot X^8 \equiv \mathbf{P}_7(X) \oplus P_{\text{CHECK}} \pmod{\mathbf{G}(X)}$:

$$\mathbf{Q} \equiv \mathbf{R}_{\text{Inv}} \cdot (\mathbf{P}_7 \oplus P_{\text{CHECK}}) \pmod{\mathbf{G}(X)}.$$

- Soit $\mathbf{P}_{\text{Cor}} = P_{\text{CHECK}} \oplus \mathbf{R}_{\text{Inv}} \cdot (\mathbf{P}_7 \oplus P_{\text{CHECK}})$, alors :

$$\mathbf{Q}' = \mathbf{Q} \oplus \mathbf{P}_{\text{Cor}} \equiv P_{\text{CHECK}} \pmod{\mathbf{G}(X)}.$$

- On suppose l'existence d'un oracle \mathcal{O}_{CRC} qui retourne 1 si une trame est valide, et 0 sinon.
 - ▶ $\mathcal{O}_{\text{CRC}}(\mathbf{Q}') = 1$, nous avons retrouvé le dernier octet de \mathbf{M}'

Comment construire \mathcal{O}_{CRC} ?

- L'attaquant possède deux stations A et B dans le réseau. Il transmet à B via A une trame \mathbf{Q}' . La trame est relayée à B par l'AP ssi \mathbf{Q}' est valide.
- L'attaquant possède une machine dans le réseau. Il transmet trame \mathbf{Q}' à une adresse aléatoire. Un message d'erreur est retourné par l'AP si la trame est valide.

Plan du cours

3

(In)Sécurité dans les réseaux sans fil – WEP/WPA

- Wired Equivalent Privacy (WEP)
 - Chiffrement
 - Authentification
 - Sécurité
- Wi-Fi Protected Access (WPA/WPA2)

Evolutions

Wi-Fi Protected Access (WPA et WPA2) [802.11i, 2004]

Authentification

- Serveur d'identification 802.1X (Extensible Authentication Protocol, EAP) pour distribuer les clefs.
 - ▶ Mode dégradé pre-shared key (PSK)

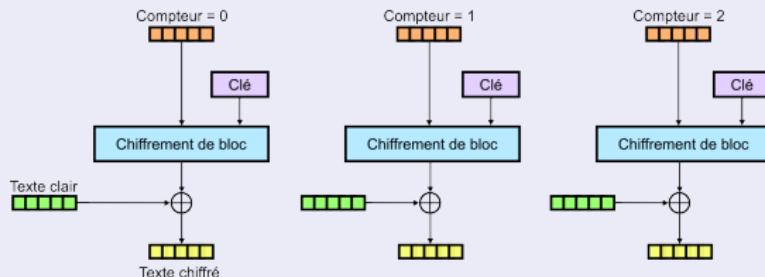
Confidentialité

- Temporal Key Integrity Protocol (TKIP)
 - ▶ code d'authentification de message : message integrity code (MIC)
 - ▶ un compteur pour les vecteurs d'initialisation
 - ▶ gestion dynamique des clefs de chiffrement
- Utilisation de RC4 WPA dans (clef de 128 bits, IV de 48 bits)
- Utilisation de AES dans WPA2

AES-CCMP

Chiffrement et Authentification

- Mode CTR
- CBC-MAC



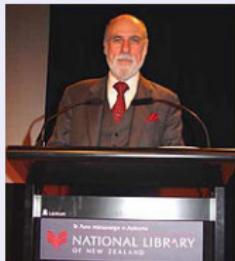
Source : <https://fr.wikipedia.org/>

Plan du cours

- 4 Internet Security Protocol – IPSec
 - Les protocoles AH et ESP
 - Modes
 - Gestion des clefs

Internet Protocol (IP)

Il permet le transport des paquets de données.



Internet Security protocol (IPSec)

- Extension de sécurité pour IP
 - ▶ RFC 2401, 2402, 2406 et 2408
- Intégré à IPv6 (optionnel pour IPv4)

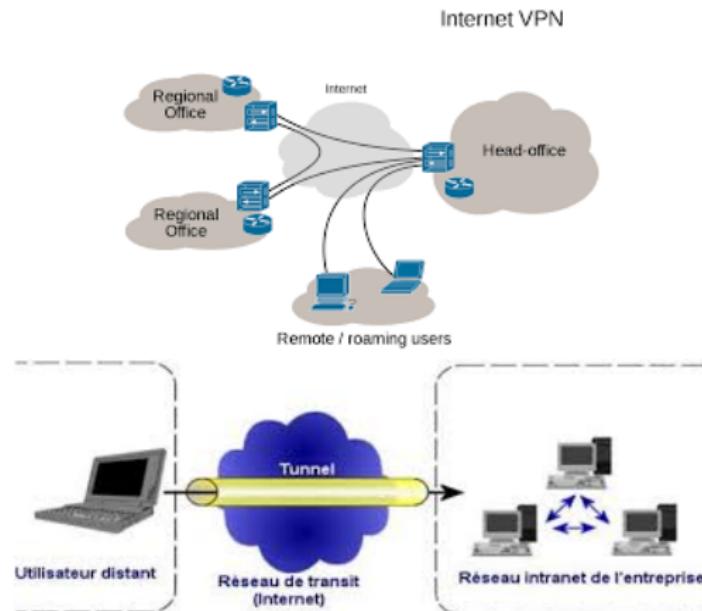


IPSec – Services

- Confidentialité des données
 - ▶ Chiffrements symétriques (RC5, IDEA, CAST, Blowfish, AES, ...)
 - ▶ Mode opératoires (CTR)
- protection partielle contre l'analyse du trafic
 - ▶ Chiffrement des en-têtes des paquets IP pour masquer – par exemple – les adresses source et destination réelles
 - ▶ protection contre le **rejet**
- Authentification des données
 - ▶ HMAC

IPSec – Applications

- Création de réseaux privés virtuels (VPN)



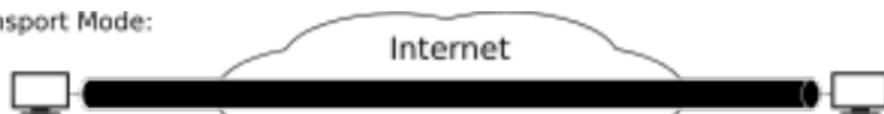
Architecture

- AH (*Authentication Header*)
- ESP (*Encapsulating Security Payload*)

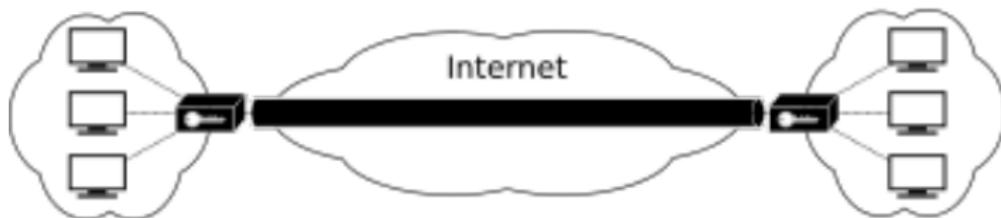
Architecture

- AH (*Authentication Header*)
- ESP (*Encapsulating Security Payload*)
- Mode transport (ESP)
- Mode tunnel (AH)

Transport Mode:



Tunnel Mode:



Association de Sécurité

- + services et algorithmes utilisés sont paramétrables

Définition

Une **Association de Sécurité (AS)** permet de fixer le type d'extension de sécurité (AH ou ESP) à mettre en place sur une communication ainsi que les services et les paramètres de sécurité (algorithmes et clefs de chiffrement, ...).

- Une relation unidirectionnelle entre un émetteur et un récepteur
- Une AS est identifiée de manière unique par un triplet
 - ▶ une chaîne SPI (Security Parameters Index) (32 bits),
 - ▶ l'adresse du destinataire d'un paquet IP,
 - ▶ le protocole de sécurité AH ou ESP.

SADB (Security Association DataBase)

- Ensemble des SA établies
 - ▶ règles nécessaires au fonctionnement de cette AS (SPI) :
 - ★ **Mode** de sécurisation du transport
 - ★ Algorithmes
 - ★ ...

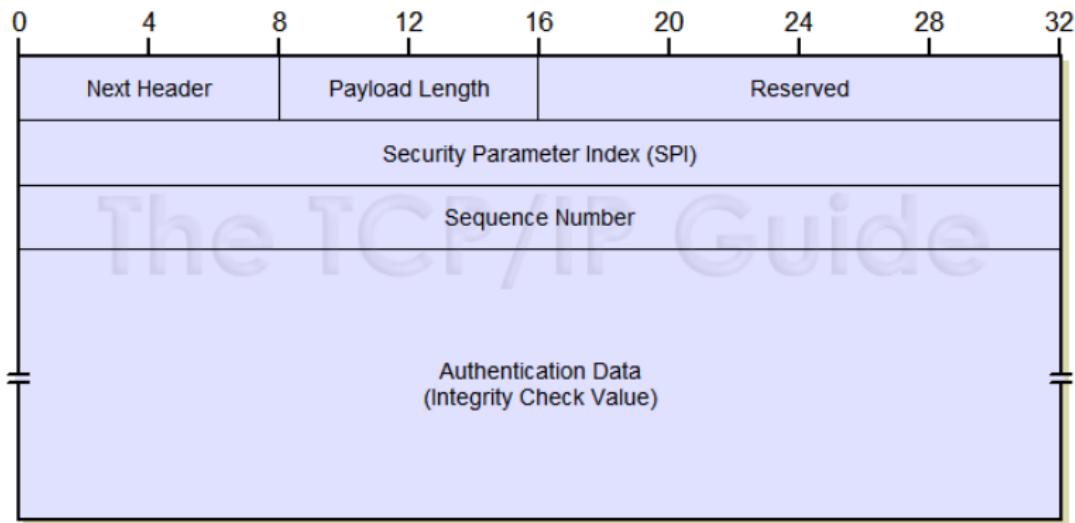
Plan du cours

- 4 Internet Security Protocol – IPSec
 - Les protocoles AH et ESP
 - Modes
 - Gestion des clefs

Authentication Header (AH)

- Authentification des paquets IP sans chiffrement des données
 - ▶ pas de confidentialité.
- On ajoute au paquet IP classique un champ additionnel – **AH (Authentication Header)** – permettant de vérifier l'authenticité des données.
- Un numéro de séquence permet de détecter les tentatives de rejet.

En-tête AH



- Le SPI combiné à l'adresse du destinataire et du protocole IPsec (AH ou ESP), identifie l'association de sécurité (AS) utilisée pour construire cette extension.
- Le numéro de séquence sur 32 bits permet de détecter les rejeux de paquet IP.
- Le champ authenticateur garantit l'intégrité du paquet.

Encapsulating Security Payload (ESP)

- ESP a pour rôle premier d'assurer la **confidentialité** mais peut aussi assurer l'authenticité des données.
- On génère, à partir d'une trame IP classique, une nouvelle trame dans laquelle les données et éventuellement l'en-tête original, sont chiffrés.
- ESP peut également assurer l'authenticité des données par ajout d'un bloc d'authentification.
- Protection contre le rejeu par le biais d'un numéro de séquence.

Structure de l'ESP

Encapsulating Security Payload

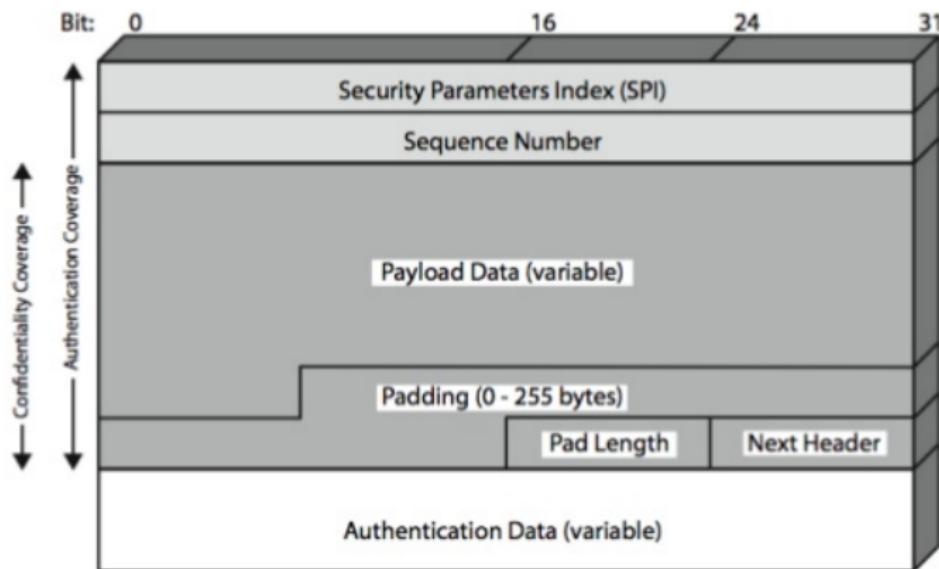


Fig-1. ESP Format

Plan du cours

4

Internet Security Protocol – IPSec

- Les protocoles AH et ESP
- **Modes**
- Gestion des clefs

Modes

Mode Transport

- Protection du contenu du paquet IP sans toucher à l'en-tête.

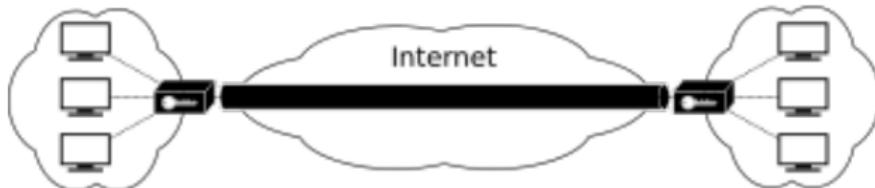
Mode Tunnel

- La protection porte sur l'ensemble des champs des paquets IP arrivant à l'entrée d'un tunnel.
- Il permet la création de tunnels par "*encapsulation*" de chaque paquet IP dans un nouveau paquet.

Transport Mode:



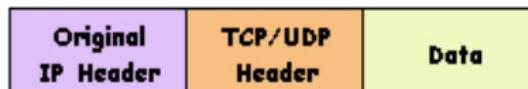
Tunnel Mode:



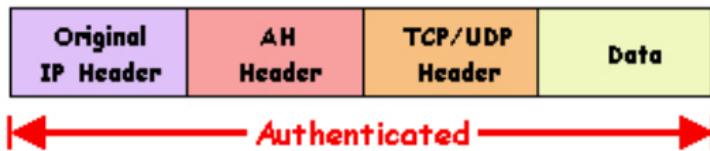
AH Transport/Tunnel

IPSec Authentication Header (AH): IP protocol number 51

Before applying AH



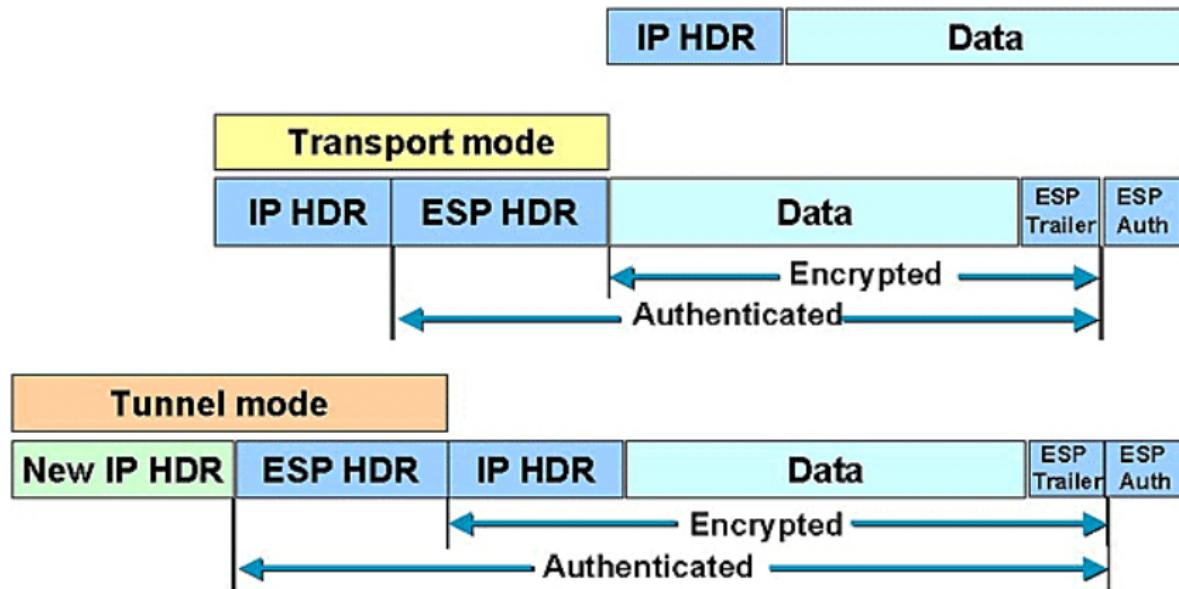
IPSec Transport Mode: After applying AH



IPSec Tunnel Mode: After applying AH



ESP Transport/Tunnel



Plan du cours

4

Internet Security Protocol – IPSec

- Les protocoles AH et ESP
- Modes
- Gestion des clefs

Gestion des clefs

Pour établir une communication sécurisée, il faut :

- partager une SA
- partager des clefs

IPSec supporte deux types de gestions des clef secrètes :

- **Manuelle.** Un administrateur configure manuellement chaque système avec ses propres clefs.
 - + pratique dans un environnement statique et de petite taille
 - définition totalement statique des associations de sécurité
 - non-renouvellement des clefs
- **Automatique.** Un système automatisé permet une création à la demande de clefs pour les associations de sécurité.
 - ▶ ISAKMP : cadre général
 - ⇒ Besoin de spécifier l'échange des clés (typiquement, IKE (Internet Key Exchange)).

Internet Security Association and Key Management Protocol – ISAKMP

- Il permet d'établir, de négocier, de modifier ou de supprimer des SA et leurs attributs.
 - ▶ ISAKMP fonctionne en deux phases.

Phase 1

- Un ensemble d'attributs relatifs à la sécurité est négocié
- Les identités des entités sont authentifiées et des clefs sont générées. Ces éléments constituent une première association de sécurité, dite **SA ISAKMP**.
- Le **SA ISAKMP** est bidirectionnel et servira à protéger l'ensemble des échanges ISAKMP futurs.

Internet Security Association and Key Management Protocol – ISAKMP

Phase 2

- Négociation des associations de sécurité IPsec.
- Chaque négociation aboutit à deux SA symétriques, une dans chaque sens de la communication.



Fielded Capability: End-to-End VPN SPIN 9 Design Review

The overall classification for this brief is:

TOP SECRET//COMINT//REL USA, AUS, CAN, GBR, NZL//20320108

(U//FOUO) Exploited Protocols

(TS//SI//REL) **IPsec automatic key management protocols** establish security associations between communicants. A **security association** (SA) is a relationship between a source and a destination that includes a session key and other parameters. The VPN capability exploits the following key management protocols:

- (U) **ISAKMP** – Internet Security Association and Key Management Protocol (RFC2407, RFC2408) provides the authentication and key exchange framework.
- (U) **IKE** – Internet Key Exchange (RFC2409) provides the authentication and key exchange mechanisms.

(U//FOUO) Exploited Protocols (continued)

(TS//SI//REL) **IPsec security protocols** provide integrity, confidentiality, and authentication for higher layer IP protocols. IPsec security protocols use security associations previously established either manually or by automatic key management protocols (IKE). The VPN capability targets SA's that are established by IKE. The VPN capability exploits the following security protocol:

- (U) **ESP** - Encapsulating Security Payload (RFC2406) provides traffic confidentiality (via encryption) and optionally provides authentication and integrity protection.

Weak Diffie-Hellman and the Logjam Attack

Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice

David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, and Paul Zimmermann

22nd ACM Conference on Computer and Communications Security (CCS '15), Denver, CO, October 2015

Best Paper Award Winner

Weak Diffie-Hellman and the Logjam Attack

Principe de l'attaque

- Attaque MITM contre Diffie-Hellman
- Forcer la négociation vers une instance faible Diffie-Hellman
 - ▶ Diffie-Hellman en mode éphémère (DHE)
 - ▶ export-grade cryptography DHE_EXPORT, groupe de 512 bits au plus.

Weak Diffie-Hellman and the Logjam Attack

Websites, mail servers, and other TLS-dependent services that support DHE_EXPORT ciphers are at risk for the Logjam attack. We use Internet-wide scanning to measure who is vulnerable.

Protocol	Vulnerable to Logjam
HTTPS – Top 1 Million Domains	8.4%
HTTPS – Browser Trusted Sites	3.4%
SMTP+StartTLS – IPv4 Address Space	14.8%
POP3S – IPv4 Address Space	8.9%
IMAPS – IPv4 Address Space	8.4%

Websites that use one of a few commonly shared 1024-bit Diffie-Hellman groups may be susceptible to passive eavesdropping from an attacker with nation-state resources. Here, we show how various protocols would be affected if a single 1024-bit group were broken in each protocol, assuming a typical up-to-date client (e.g., most recent version of OpenSSH or up-to-date installation of Chrome).

Vulnerable if most common 1024-bit group is broken	
HTTPS – Top 1 Million Domains	17.9%
HTTPS – Browser Trusted Sites	6.6%
SSH – IPv4 Address Space	25.7%
IKEv1 (IPsec VPNs) – IPv4 Address Space	66.1%

Weak Diffie-Hellman and the Logjam Attack

We carried out this computation against the *most common 512-bit prime used for TLS and demonstrate that the Logjam attack can be used to downgrade connections to 80% of TLS servers supporting DHE_EXPORT*. We further estimate that an academic team can break a *768-bit prime and that a nation-state can break a 1024-bit prime*. Breaking the single, *most common 1024-bit prime used by web servers would allow passive eavesdropping on connections to 18% of the Top 1 Million HTTPS domains*. A second prime would allow *passive decryption of connections to 66% of VPN servers and 26% of SSH servers*. A close reading of published NSA leaks shows that the agency's attacks on VPNs are consistent with having achieved such a *break*. We conclude that moving to stronger key exchange methods should be a priority for the Internet community

Plan du cours

5

BEAST

Attaque BACPA (Bard & Dai)

$$E : \begin{matrix} \{0,1\}^s \\ k \end{matrix} \times \begin{matrix} \{0,1\}^n \\ x \end{matrix} \rightarrow \{0,1\}^n$$
$$E_k(x) \mapsto$$

Attaque BACPA (Bard & Dai)

$$E : \begin{matrix} \{0,1\}^s \\ k \end{matrix} \times \begin{matrix} \{0,1\}^n \\ x \end{matrix} \rightarrow \begin{matrix} \{0,1\}^n \\ E_k(x) \end{matrix}$$

Mode CBC

Soit IV un vecteur d'initialisation de n bits. On chiffre m_1, \dots, m_t :

$$c_0 = \text{IV}, \quad c_i = E_k(m_i \oplus c_{i-1}), \quad 1 \leq i \leq t.$$

Pour déchiffrer :

$$m_0 = c_0 = \text{IV}, \quad m_i = D_k(c_i) \oplus c_{i-1}, \quad 1 \leq i \leq t.$$

Attaque BACPA (Bard & Dai)

Mode CBC

Soit IV un vecteur d'initialisation de n bits. On chiffre m_1, \dots, m_t :

$$c_0 = \text{IV}, \quad c_i = E_k(m_i \oplus c_{i-1}), \quad 1 \leq i \leq t.$$

Pour déchiffrer :

$$m_0 = c_0 = \text{IV}, \quad m_i = D_k(c_i) \oplus c_{i-1}, \quad 1 \leq i \leq t.$$

Mode CBC dans SSL&TLS

C'est un mode CBC **dégradé**.

- Le premier message est chiffré avec IV aléatoire.
- Le dernier bloc du chiffré d'un **message j** est utilisé comme vecteur d'initialisation pour le chiffrement du **message $j + 1$** .

Attaque BACPA (Bard & Dai)

Blockwise-Adaptive Chosen-Plaintext Attack

- Oracle \mathcal{O} qui retourne le chiffrement d'un message donné.

Attaque BACPA (Bard & Dai)

Blockwise-Adaptative Chosen-Plaintext Attack

- Oracle \mathcal{O} qui retourne le chiffrement d'un message donné.

Objectif

Soient c_1, \dots, c_t le chiffrement en mode CBC dégradé de m_1, \dots, m_t avec IV comme vecteur initialisation, i.e.

$$c_1, \dots, c_t \leftarrow \text{CBC_}E_k(m_1 \parallel \dots \parallel m_t).$$

- Comment utiliser \mathcal{O} pour retrouver m_j ?

Attaque BACPA (Bard & Dai)

Principe

Soit $m' = m^* \oplus c_{j-1} \oplus c_t \| \cdots \| m'_t$. On note :

$$c'_1, \dots, c'_t \leftarrow \text{CBC_}E_k(m'_1 \| \cdots \| m'_t).$$

- Nous avons $c_j = E_k(m_j \oplus c_{j-1})$. Montrer que :

$$c'_1 = c_j \iff m^* = m_j.$$

Attaque BACPA (Bard & Dai)

Principe

Soit $m' = m^* \oplus c_{j-1} \oplus c_t \| \dots \| m'_t$. On note :

$$c'_1, \dots, c'_t \leftarrow \text{CBC_}E_k(m'_1 \| \dots \| m'_t).$$

Nous avons $c_j = E_k(m_j \oplus c_{j-1})$:

$$c'_1 = E_k(m'_1 \oplus c_t),$$

$$c'_1 = E_k(m^* \oplus c_{j-1} \oplus c_t \oplus c_t) = E_k(m^* \oplus c_{j-1}).$$

Ainsi :

$$c'_1 = c_j \iff m^* = m_j.$$

Attaque BACPA (Bard & Dai)

Limite

Soit $m' = m^* \oplus c_{j-1} \oplus c_t \| \dots \| m'_t$. On note :

$$c'_1, \dots, c'_t \leftarrow \text{CBC_}E_k(m'_1 \| \dots \| m'_t).$$

Nous avons :

$$c'_1 = c_j \iff m^* = m_j.$$

⇒ Il faut deviner le bloc m_j .

Attaque BEAST (Duong & Rizzo)

Blockwise-Adaptative Chosen-Boundary Plaintext Attack

- ⇒ On cherche à retrouver $m[1]$ (i.e. premier octet du message) avec l'oracle de déchiffrement \mathcal{O} .
- ▶ $m'_1 = c_0 \oplus c_t \oplus (r\|g)$, avec g un octet.

Attaque BEAST (Duong & Rizzo)

Principe

Soit $m'_1 = c_0 \oplus c_t \oplus (r \| g)$. Nous avons :

$$c_1 = E_k(m'_1 \oplus c_t),$$

$$c_1 = E_k(c_0 \oplus c_t \oplus (r \| g) \oplus c_t) = E_k((r \| g) \oplus c_0).$$

Ainsi :

$$c_1^* = c_1' \iff m[1] = g.$$

⇒ On devine le message octet par octet !!