Hachage. Tables. Intégrité. RSA.

Sergey Kirgizov

Plan

Structure du module

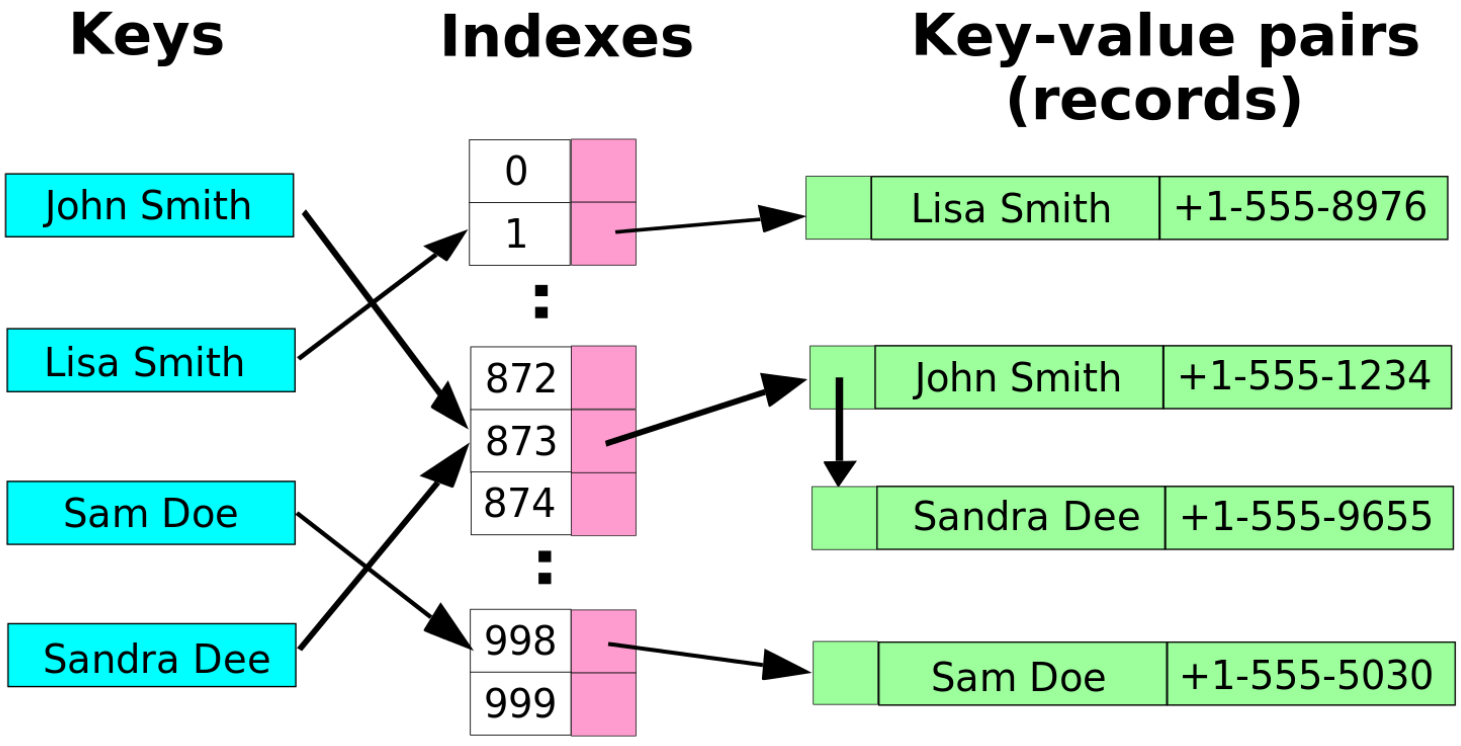
8 CM, 4 séances TD, 7 séances TP

Le lien :
`https://kirgizov.link/teaching/polytech-dijon/sic/`

Les notes :

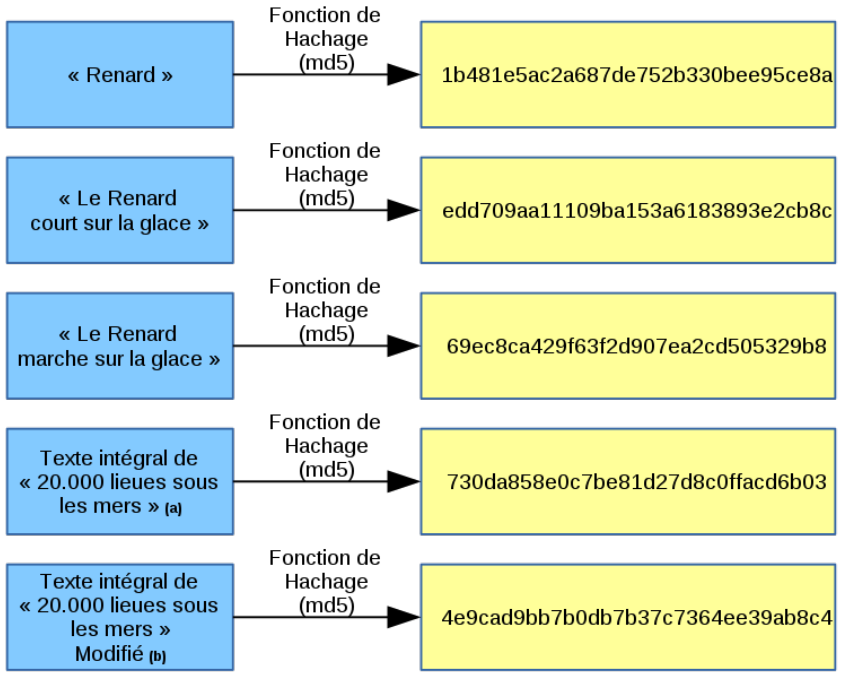
- un contrôle intermédiaire
- un contrôle terminal
- un projet à faire pendent les TPs, autour de la compréhension de la blockchain

Table de hachage



src : Helix84 @ Wikipedia
`https://fr.wikipedia.org/wiki/Table_de_hachage`

Fonction de hachage



Exemples de hachages de textes par la fonction md5 ; (a) le texte utilisé est la version libre de Vingt mille lieues sous les mers du projet Gutenberg ; (b) la version modifiée est le même fichier texte, le 10e caractère de la 1000e ligne ayant été remplacé par le caractère “*”.

– Unique Nitrogen, Wikipedia

Propriétés

- **Compression (avec pertes)**
  Large ensemble depart, “petit” ensemble d’arrivée.
  Exemple : $\{0,1\}* \to 2^{128}$ — toutes les chaîne de bits → mots de 128 bits.

- **Uniformité**
  Les nombres des inputs ayant le même hash sont réparties uniformément.
  Antidote contre les collisions. Très utile pour les tables de hachage.

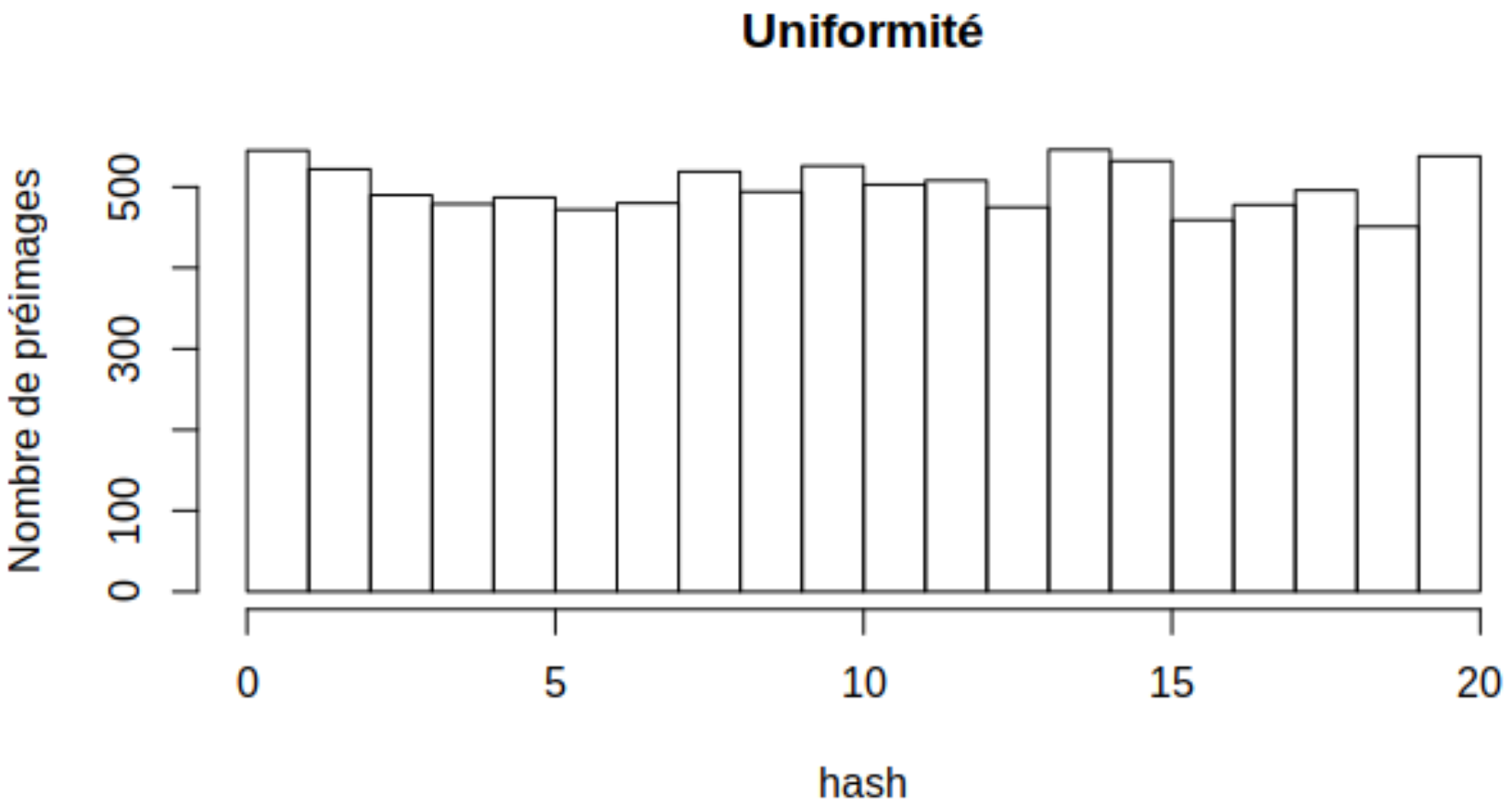
- **Fonction à sens unique (one way)**
  Facile à calculer, difficile à trouver la préimage.

- **Effet avalanche**
  si pour toute inversion d’un seul bit en entrée alors chaque bit en sortie a une chance sur deux d’être modifié. — Webster et Tavares, 1985

- **ou continuité**
  Si on utilise les hashs pour comparer les préimages.

Uniformité

Uniformité



Quelle est la probabilité d'une collision ?

Collision : $x \neq y$ t.q. $h(x) = h(y)$

Paradoxe des anniversaires

Quelle est la probabilité de coïncidence de 2 anniversaires dans dans un groupe de vingt-trois personnes ?

Paradoxe des anniversaires

Quelle est la probabilité de coïncidence de 2 anniversaires dans dans un groupe de vingt-trois personnes ?

$\approx 51\%$

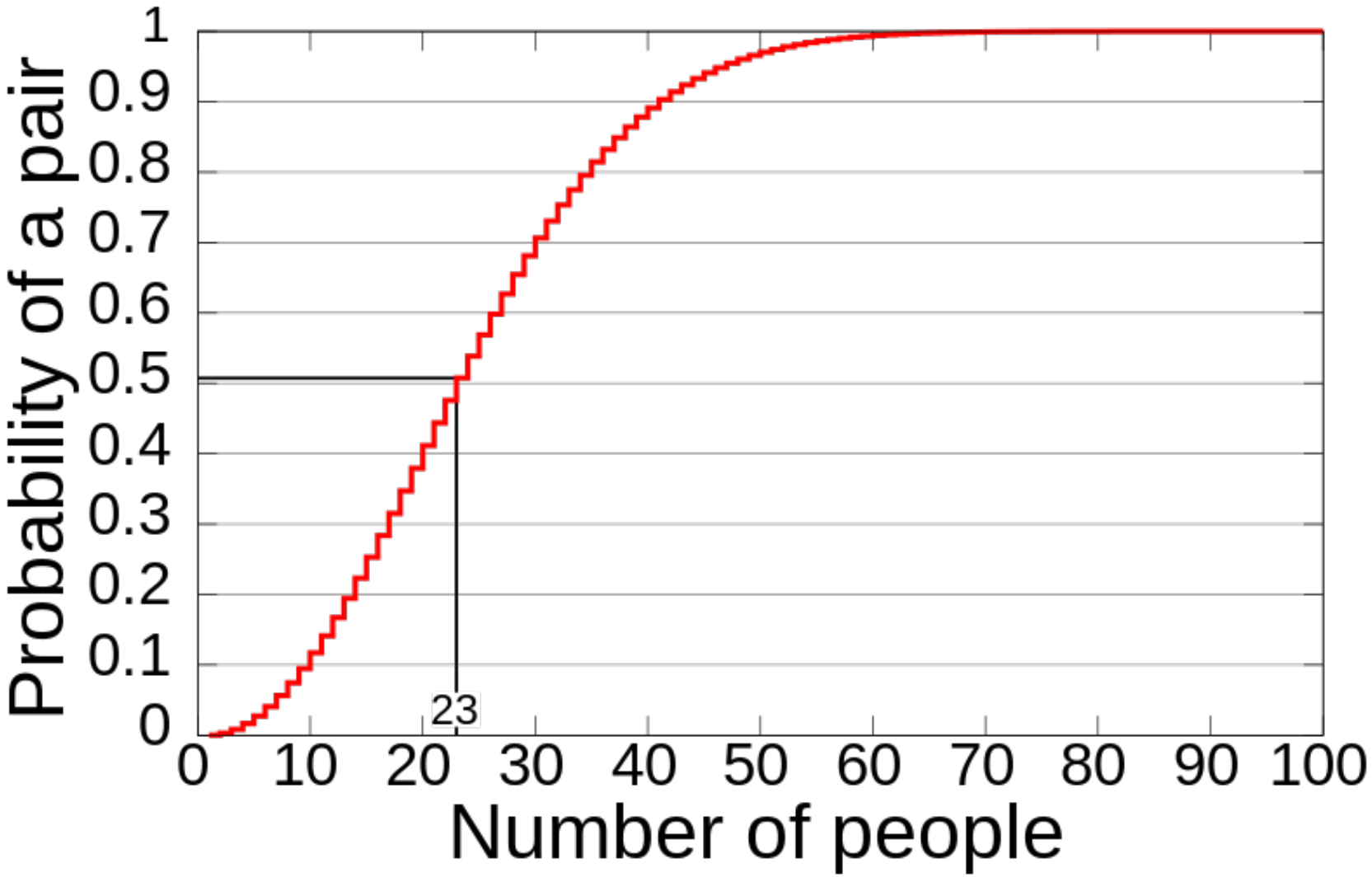
Paradoxe des anniversaires

Probabilité $\bar{p}(n)$ que tous les n anniversaires soient différents.

$$\bar{p}(n) = \left(1 - \frac{1}{365}\right) \times \left(1 - \frac{2}{365}\right) \times \cdots \times \left(1 - \frac{n-1}{365}\right)$$

$$1 - \bar{p}(23) \approx 0.507$$

Paradoxe des anniversaires



Rajkiran g @ Wikipédia

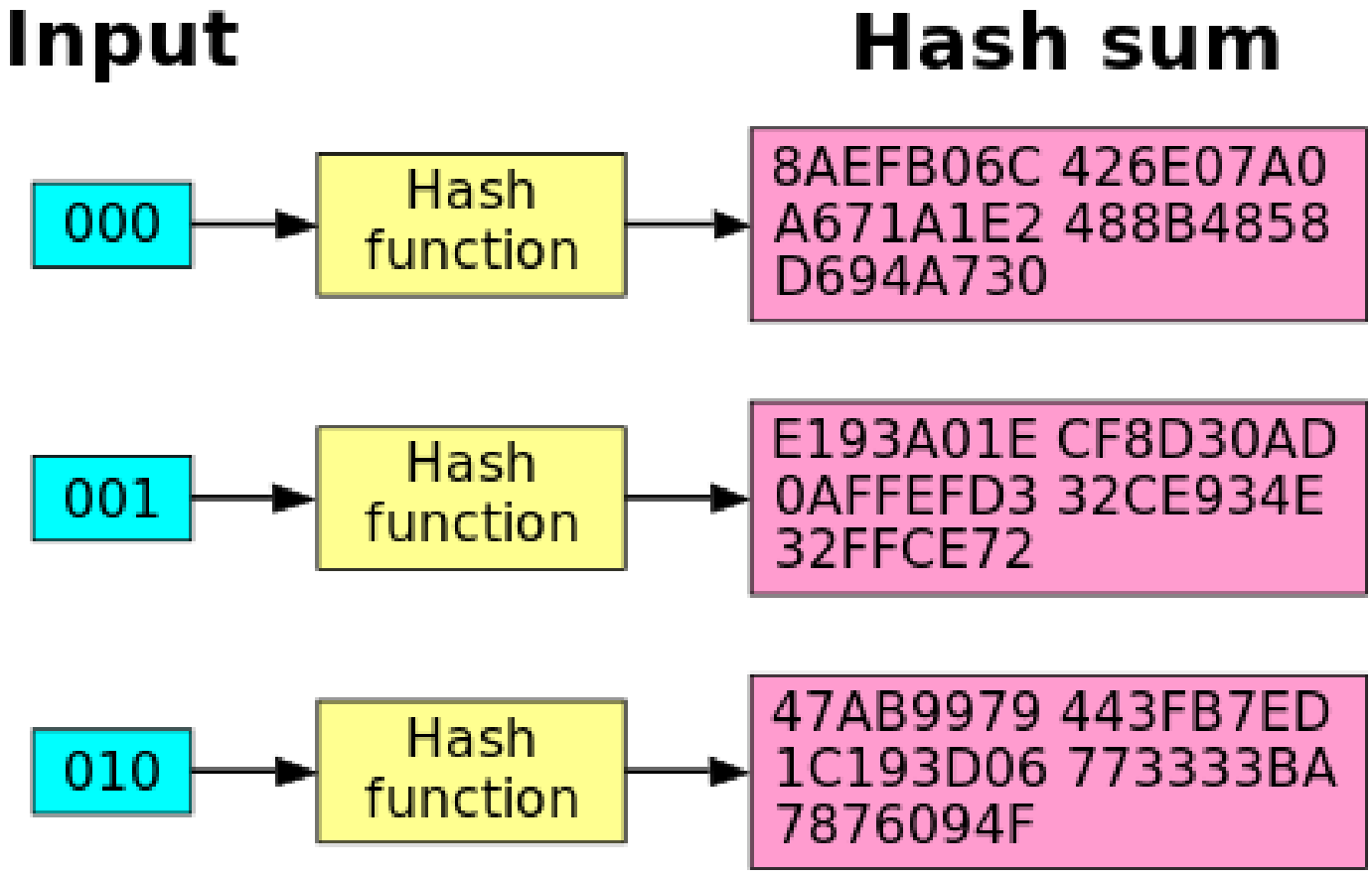
En TD

La fonction de hachage respecte l'uniformité.
La longeur du hash c'est 128 bit.

Quelle est probabilité de coïncidence probabilité de coïncidence cette fonction au 1 000 000 000 entrées differentes ?

Effet avalanche

Effet avalanche



Le pendule double

CET OBJET EST CHAOTIQUE ! (double pendule)

`https://www.youtube.com/watch?v=m923FsfhNYE`

Sens unique

Levin's paper

The Tale of One-Way Functions*

Leonid A. Levin†
Boston University‡

All the king's horses, and all the king's men,
Couldn't put Humpty together again.

Abstract

The existence of one-way functions (owf) is arguably the most important problem in computer theory. The article discusses and refines a number of concepts relevant to this problem. For instance, it gives the first combinatorial complete owf, i.e., a function which is one-way if any function is. There are surprisingly many subtleties in basic definitions. Some of these subtleties are discussed or hinted at in the literature and some are overlooked. Here, a unified approach is attempted.

1 Introduction I: Inverting Functions

From time immemorial, humanity has gotten frequent, often cruel, reminders that many things are easier to do than to reverse. When the foundations of mathematics started to be seriously analyzed, this experience immediately found a formal expression.

Leonid Levin

https://www.cs.bu.edu/~lnd/

Les applications

Applications

Stockage

- L'identification des données plus rapide
- Stockage un espace virtuel très grand
- Somme de contrôle, intégrité des données
- Hashtables

  Dans le cas des tables de hachage, l'ensemble d'arrivée est petit.
- Arbres de Merkle

Cryptographie

- Contrôle d'accès (hashs salés de mots de passe, etc)
- Signatures et HMACs

Préimage

$f : X \to Y$

Préimage d'un $y \in Y$ c'est $x \in X$ t.q. $f(x) = y$.

Attaques

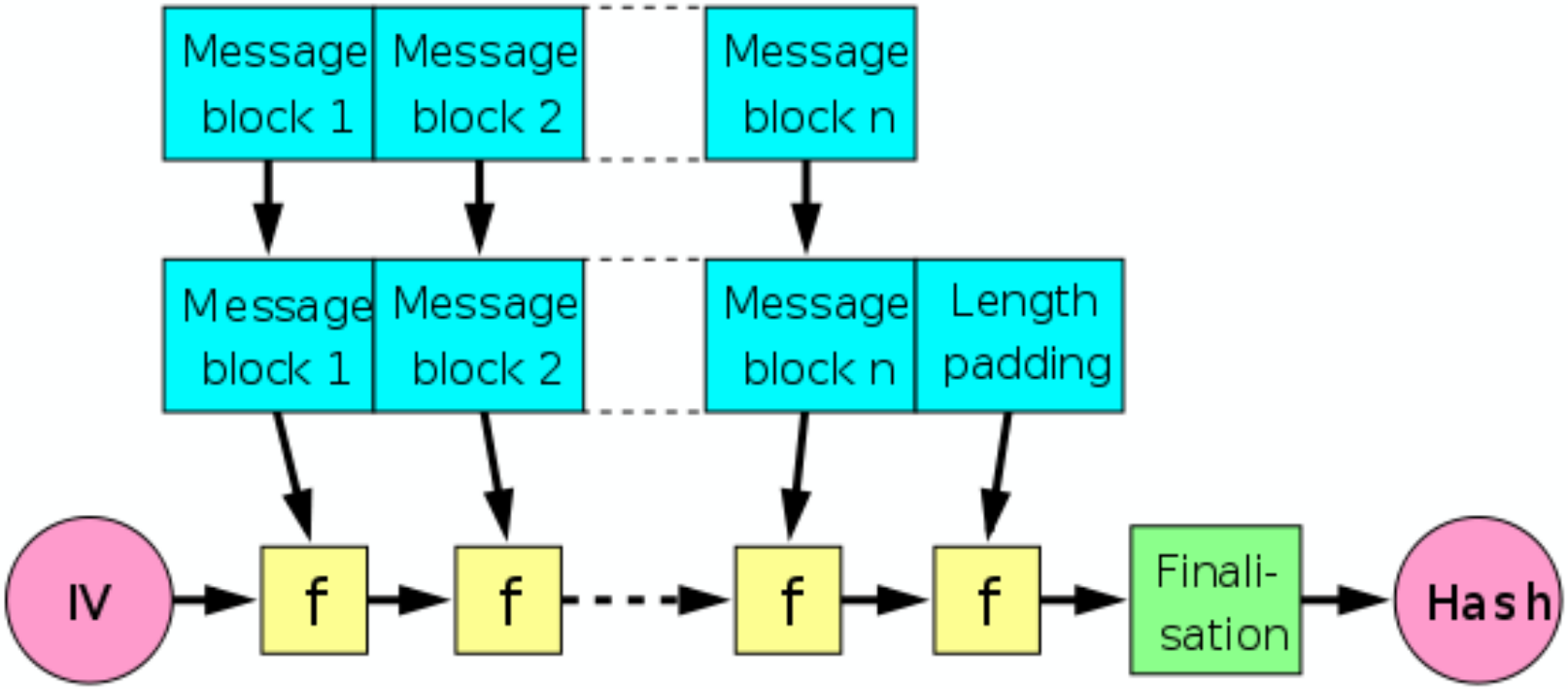
h : Entrées $\rightarrow$ Sorties
h c'est une fonction de hachage.

- **Collision** Trouver x et y, t.q. $x \neq y$ et $h(x) = h(y)$.
  Exemple : SHA-1 collision, "6.500" years of single-CPU : `https://shattered.io`
- **Préimage** Pour une valeur de sortie spécifiée q, trouver un x, t.q. $h(x) = q$.
- **Deuxième Préimage** Pour une entrée spécifiée x, trouver un y, t.q. $x \neq y$ et $h(x) = h(y)$.

Méthodes et techniques : force brute, rainbow tables, etc.
La science de cryptanalyse.

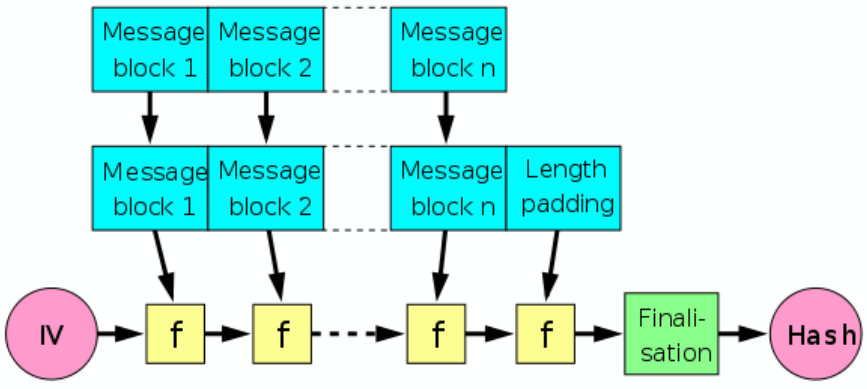
Comment construire une fonction de hachage ?

La construction de Merkle-Damgård



en.wikipedia.org/wiki/Merkle–Damgård_construction

La construction de Merkle-Damgård



- $f(x, y)$: une fonction de compression à sens unique, par bloc de taille fix.
- $f(x, y)$: par exemple, un chiffrement par bloc de taille fix. (XOR ?)
- *IV* c'est un vecteur d'initialisation.

La construction de Merkle-Damgård I

Est-ce que cela entraîne l'apparition de nouvelles collisions ?

La taille de block est 128 bit.
Le taille du message est 379bit.
Donc il faut faire du rembourrage (*padding*) !

$$M_1 = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C}^{123bit}$$

$$M_1' = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C00000}^{128bit}$$

$$h(M_1) = h(M_1') = f\Big(f\big(f(IV, A), B\big), C00000\Big)$$

La construction de Merkle-Damgård I

Est-ce que cela entraîne l'apparition de nouvelles collisions ?

La taille de block est 128 bit.
Le taille du message est 379bit.
Donc il faut faire du rembourrage (*padding*) !

$$M_1 = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C}^{123bit}$$

$$M_1' = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C00000}^{128bit}$$

$$h(M_1) = h(M_1') = f\Big(f\big(f(IV, A), B\big), C00000\Big)$$

Imaginons maintenant un autre message $M_2 = ABD$, où $D = C00$

$$h(M_2) = h(M_2') = f\Big(f\big(f(IV, A), B\big), D000\Big)$$

$$M_1 \neq M_2, h(M_1) = h(M_2)$$

Collision !

Collision ! Ce rembourrage n'est pas bien !

La construction de Merkle-Damgård II

Est-ce que cela entraîne l'apparition de nouvelles collisions ?

La taille de block est 128 bit.
Le taille du message est 379bit.
Donc il faut faire du rembourrage (*padding*) !

$$M_1 = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C}^{123bit}$$

$$M_1' = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C10000}^{128bit}$$

$$h(M_1) = h(M_1') = f\Big(f\big(f(IV, A), B\big), C10000\Big)$$

La construction de Merkle-Damgård II

Est-ce que cela entraîne l'apparition de nouvelles collisions ?

La taille de block est 128 bit.
Le taille du message est 379bit.
Donc il faut faire du rembourrage (*padding*) !

$$M_1 = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C}^{123bit}$$

$$M_1' = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C10000}^{128bit}$$

$$h(M_1) = h(M_1') = f\Big(f\big(f(IV, A), B\big), C10000\Big)$$

Imaginons maintenant un autre message $M_2 = ABD$, où $D = C00$

$$h(M_2) = h(M_2') = f\Big(f\big(f(IV, A), B\big), D100\Big)$$

$$M_1 \neq M_2, h(M_1) \neq h(M_2)$$

pas de collision si f n'est fait pas une.

La construction de Merkle-Damgård III

Est-ce que cela entraîne l'apparition de nouvelles collisions ?

$$M_1 = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C}^{123bit}$$

$$M_1' = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C10000}^{128bit}$$

$$h(M_1) = h(M_1') = f\Big(f\big(f(IV, A), B\big), C10000\Big)$$

La construction de Merkle-Damgård III

Est-ce que cela entraîne l'apparition de nouvelles collisions ?

$$M_1 = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C}^{123bit}$$

$$M_1' = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C10000}^{128bit}$$

$$h(M_1) = h(M_1') = f\Big(f\big(f(IV, A), B\big), C10000\Big)$$

Imaginons maintenant un autre message $M_2 = ZABC$, t.q. $f(IV, Z) = IV$

$$h(M_1) = h(M_1') = f\Big(f\big(f\big(f(IV, Z), A\big), B\big), C10000\Big)$$

$$M_1 \neq M_2, h(M_1) = h(M_2)$$

Collision !

La construction de Merkle-Damgård IV

Est-ce que cela entraîne l'apparition de nouvelles collisions ?

Faire attention aux suffixes !

padding $= 1000\cdots00\cdots$ la taille en binaire

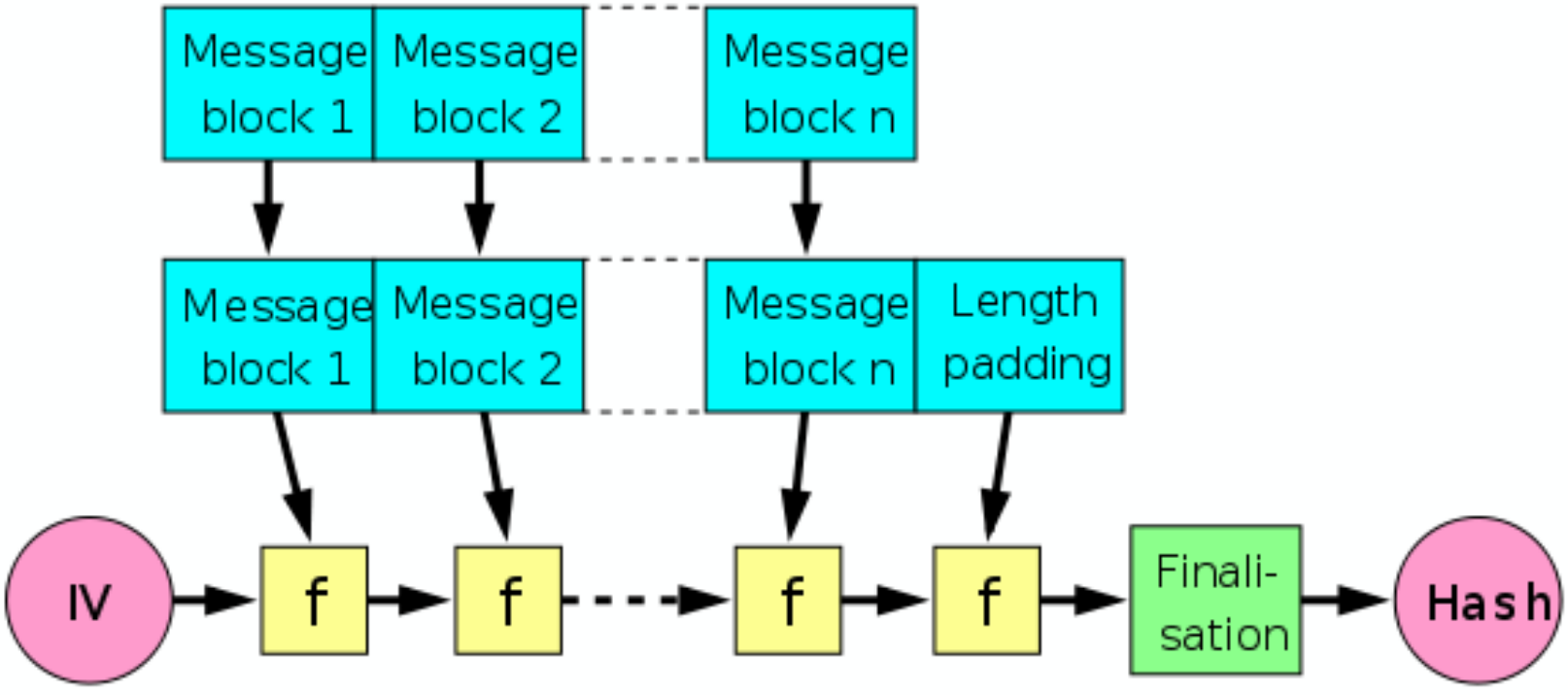
$$M_1 = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C}^{123bit}$$

$$M_1' = \overbrace{A}^{128bit}\ \overbrace{B}^{128bit}\ \overbrace{C10000}^{128bit}\overbrace{000\cdots\underbrace{1101110011}_{\text{la taille}}}^{128bit}$$

C'est mieux ! *Merkle–Damgård strengthening* !

Autres attaques sont possibles :
`https://en.wikipedia.org/wiki/Length_extension_attack`
`https://archive.org/details/pdfy-HaRR7XMfTOb_RrU0`

La construction de Merkle-Damgård



en.wikipedia.org/wiki/Merkle–Damgård_construction

Plus d'info

Il faut voir des cours sur la cryptographie... par exemple

- `http://cseweb.ucsd.edu/~mihir/courses.html`
- `http://cseweb.ucsd.edu/~mihir/papers/gb.html`

Ralph Merkle et Ivan Damgård

`http://www.daimi.au.dk/ivan/`
Denmark, Aarhus Universitet

`https://www.merkle.com/`
PhD de Stanford University
Singularity University
Alcor Life Extension Foundation, etc...

Ralph Merkle

> *While an undergraduate, Merkle devised Merkle's Puzzles, a scheme for communication over an insecure channel, as part of a class project. The scheme is now recognized to be an early example of public key cryptography.*
>
> — Wikipedia

1979 – La construction de Merkle–Damgård a été présentée dans la elle thèse de Ralph Merkle.

Ivan Damgård

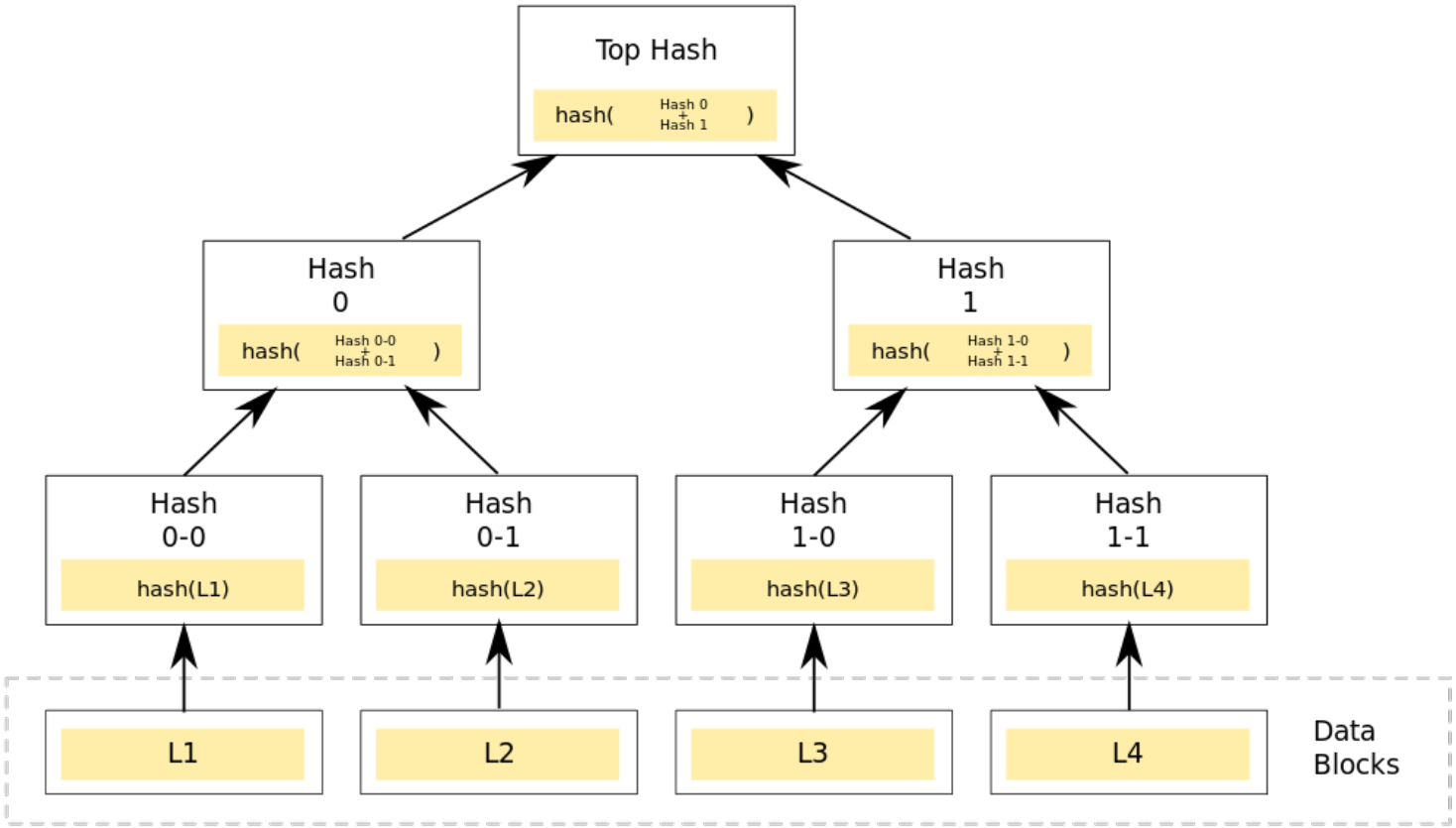
He is known among other things for the Merkle–Damgård construction used in most modern cryptographic hash functions such as SHA-1 and MD5. He discovered the structure independently of Ralph Merkle and published it in 1989.

– Wikipedia

Exemples

- MD4
- MD5
- SHA-1
- SHA-2
- ChaCha
- Salsa20
- Rumba20

Arbres de Merkle



On ajoute également les données dans tout les noeuds.

Git — the stupid content tracker

man 1 git

- “In **2002**, the Linux kernel project began using a proprietary DVCS called BitKeeper”
- “In **2005**, the relationship between the community that developed the Linux kernel and the commercial company that developed BitKeeper broke down.”
- Git a apparu en 2005.
- Premier commit `https://github.com/git/git/commit/e83c5163316f89bfbde7d9ab23ca2e25604af290`

Plus d'info : `https://git-scm.com/book/en/v2/Getting-Started-A-Short-History-of-Git`

Plus que **1300 personnes**, dont

Linus Torvalds et

Junio C Hamano

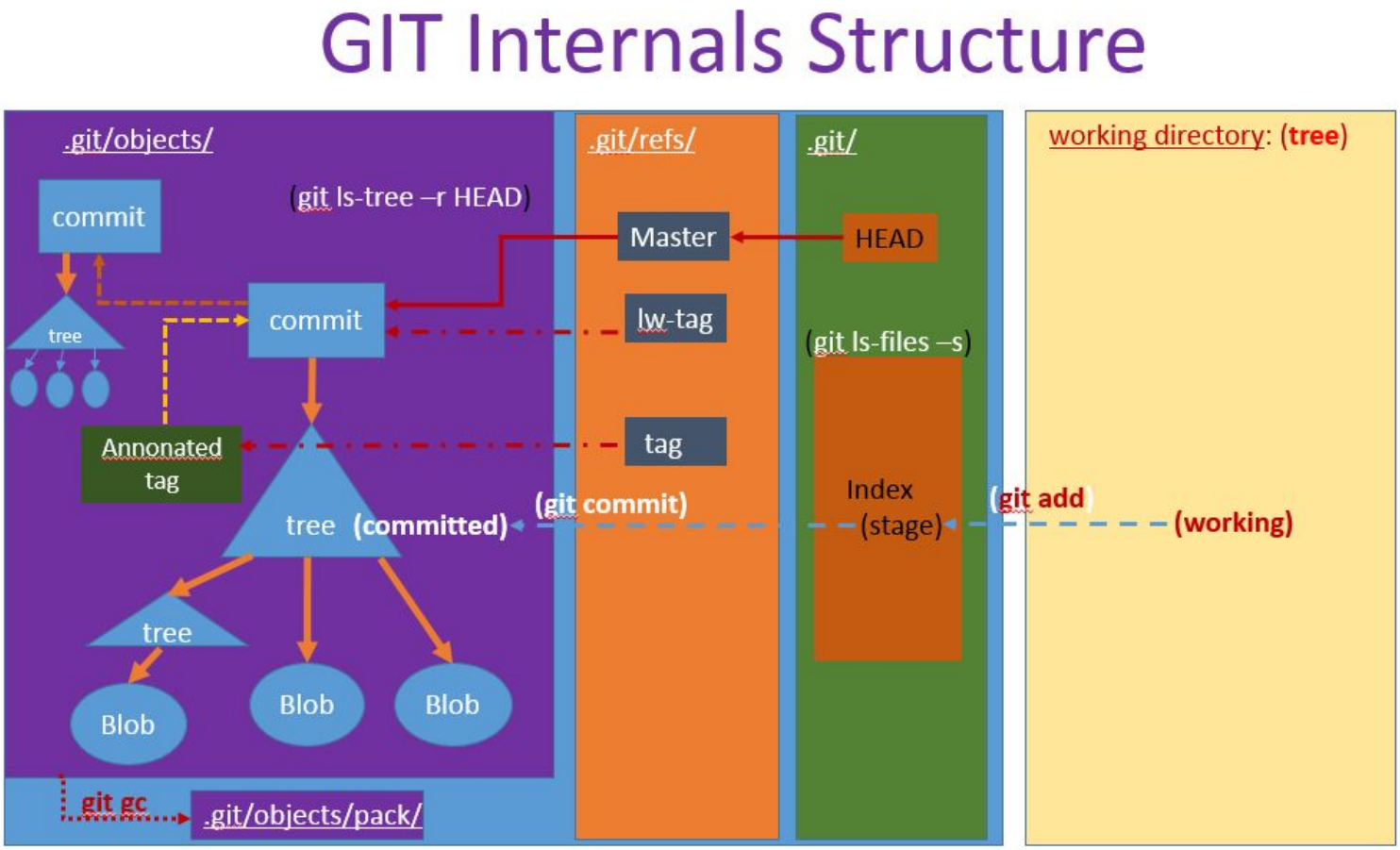
Choix techniques

- Le système distribué (`pull`, `push`)
- Écrit en : ????

Choix techniques

- Le système distribué (`pull`, `push`)
- Écrit en : C, Shell Unix, Perl, Tcl, Python, Makefile et C++

Aperçu



http://weng-blog.com/2017/01/git-internal/

Autres liens

- `man git`
- **Initial revision of "git", the information manager from hell**
  `https://github.com/git/git/commit/e83c5163316f89bfbde7d9ab23ca2e25604af290`
- **“Tech Talk : Linus Torvalds on git”**
  `https://m.youtube.com/watch?v=4XpnKHJAok8` 1h10
- `https://git-scm.com/book/fr/v2/Les-tripes-de-Git-Plomberie-et-porcelaine`
- `https://git-scm.com/book/en/v1/Git-Internals`

Chiffrement asymétrique. RSA

Signatures électroniques

Pretty Good Privacy

Le monde post-quantique ?