

(12) DEMANDE INTERNATIONALE PUBLIÉE EN VERTU DU TRAITÉ DE COOPÉRATION EN MATIÈRE DE BREVETS (PCT)

(19) Organisation Mondiale de la
Propriété Intellectuelle
Bureau international

(43) Date de la publication internationale
09 janvier 2020 (09.01.2020)



(10) Numéro de publication internationale
WO 2020/008131 A1

(51) Classification internationale des brevets :
H04L 9/08 (2006.01)

(21) Numéro de la demande internationale :
PCT/FR2019/051616

(22) Date de dépôt international :
01 juillet 2019 (01.07.2019)

(25) Langue de dépôt : français

(26) Langue de publication : français

(30) Données relatives à la priorité :
1856170 04 juillet 2018 (04.07.2018) FR

(71) Déposant : VIACCESS [FR/FR] ; 9/11 ALLÉE DE
L'ARCHE TOUR EGÉE, 92400 COURBEVOIE (FR).

(72) Inventeur : CHIEZE, Quentin ; 38 RUE ROUGET DE
LISLE, 92800 PUTEAUX (FR).

(74) Mandataire : COLOMBO, Michel ; BREVINNOV chez
INNOVATION COMPETENCE GROUP, 310 AVENUE
BERTHELOT, 69372 LYON CEDEX 08 (FR).

(81) États désignés (sauf indication contraire, pour tout titre de protection nationale disponible) : AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) États désignés (sauf indication contraire, pour tout titre de protection régionale disponible) : ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), eurasien (AM, AZ, BY, KG, KZ, RU, TJ, TM),

(54) Title: METHOD FOR OBTAINING A SEQUENCE OF CRYPTOGRAPHIC KEYS

(54) Titre : PROCEDE D'OBTENTION D'UNE SUCCESSION DE CLES CRYPTOGRAPHIQUES

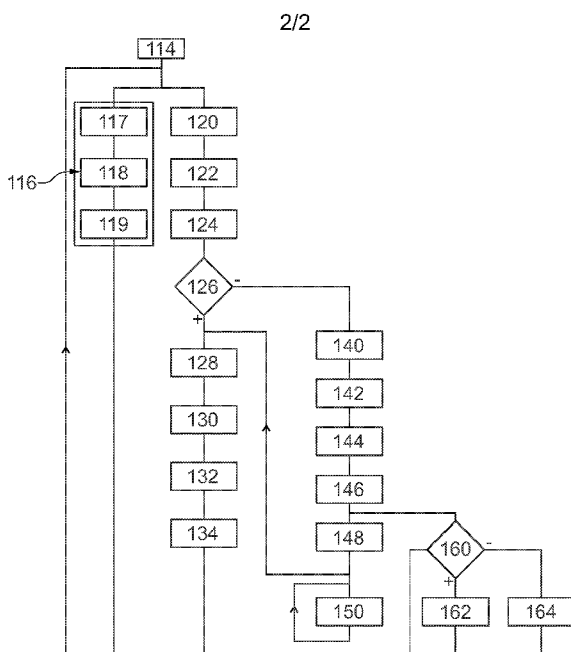


Fig. 2

(57) Abstract: The invention relates to a method for obtaining a sequence of L cryptographic keys $k_{1,m}$, $k_{i,m}$, $k_{i+1,m}$, $k_{L,m}$ in which, before an instant $t_{1,m}$, a group of receivers establishes (140) a first connection to a key server and receives, during said first connection, the information required to obtain the key $k_{1,m}$, and, subsequently, for each index i between 2 and L : the group of receivers obtains (150) the subsequent key $k_{i,m}$ by running a key derivation algorithm initialised by means of the previous key $k_{i-1,m}$ and without using any information other than that received during the first connection; and the average run-time $TC_{i,m}$ of the key derivation algorithm by the group of receivers in order to obtain the key $k_{i,m}$ is higher than $0,2V_{i-1,m}$, where $V_{i-1,m}$ is the duration of the validity interval of the previous key $k_{i-1,m}$.

(57) Abrégé : Procédé d'obtention d'une succession de L clés cryptographiques $k_{1,m}$, $k_{i,m}$, $k_{i+1,m}$, $k_{L,m}$ dans lequel : - avant un instant $t_{1,m}$, un groupe de récepteurs établit (140) une première connexion avec un serveur de clés et reçoit, lors de cette première connexion, les informations nécessaires pour obtenir la clé $k_{1,m}$, puis pour chaque indice i compris entre 2 et L : - le groupe de récepteurs obtient (150) la clé suivante $k_{i,m}$ en exécutant un algorithme de dérivation de clé initialisé à l'aide de la clé précédente $k_{i-1,m}$ et sans avoir recours à d'autres informations que celles reçues lors de la première connexion, et - le temps moyen $TC_{i,m}$ d'exécution de l'algorithme de dérivation de clé par le groupe de récepteurs pour obtenir la clé $k_{i,m}$ est supérieur à $0,2V_{i-1,m}$, où $V_{i-1,m}$ est la durée de l'intervalle de validité de la précédente clé $k_{i-1,m}$.

européen (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Publiée:

— avec rapport de recherche internationale (Art. 21(3))

PROCEDE D'OBTENTION D'UNE SUCCESSION DE CLES CRYPTOGRAPHIQUES

[001] L'invention concerne un procédé d'obtention d'une succession de clés cryptographiques ainsi qu'un procédé de transmission sécurisée d'un contenu numérique mettant en œuvre ce procédé d'obtention. L'invention concerne également un support d'enregistrement d'informations, un récepteur et un serveur de clés pour la mise en œuvre de ce procédé d'obtention d'une succession de clés.

[002] Il est connu des procédés d'obtention d'une succession de L clés cryptographiques $k_{1,m}$, ..., $k_{i,m}$, $k_{i+1,m}$, ..., $k_{L,m}$ par un groupe de récepteurs électroniques, où :

- l'indice i est un numéro d'ordre de la clé $k_{i,m}$ dans la succession de clés,
- L est un nombre entier supérieur ou égal à deux, et
- quel que soit l'indice i compris entre 1 et L , la clé $k_{i,m}$ est uniquement destinée à être utilisée pendant un intervalle de validité $[t_{i,m}, t_{i+1,m}[$ de durée $V_{i,m}$, où $t_{i,m}$ et $t_{i+1,m}$ sont, respectivement, les instants de début et de fin de cet intervalle de validité.

Dans ces procédés connus :

- avant l'instant $t_{1,m}$, le groupe de récepteurs établit une première connexion avec un serveur de clés et reçoit, lors de cette première connexion, les informations nécessaires pour obtenir la clé $k_{1,m}$, puis
- pour chaque indice i compris entre 2 et L , avant l'instant $t_{i,m}$, le groupe de récepteurs obtient la clé $k_{i,m}$.

[003] Un tel procédé est divulgué dans la demande EP2567500. Dans la demande EP2567500, chaque clé $k_{i,m}$ est utilisée pour traiter un segment $CP_{i,m}$ donné unique d'un contenu multimédia, dont la durée de jeu est appelée « cryptopériode ». Par abus de langage, le segment $CP_{i,m}$ est également appelé cryptopériode. La clé $k_{i,m}$ est donc valide pendant un intervalle de temps $[t_{i,m}; t_{i+1,m}[$ qui correspond à la cryptopériode $CP_{i,m}$, c'est-à-dire à l'intervalle de temps pendant lequel la cryptopériode $CP_{i,m}$ du contenu multimédia est jouée. Ainsi, pour pouvoir déchiffrer une cryptopériode $CP_{i,m}$, la clé $k_{i,m}$ doit être obtenue par le récepteur avant l'instant $t_{i,m}$. Idéalement, la clé $k_{i,m}$ doit être obtenue le plus tard possible avant l'instant $t_{i,m}$, de façon à être exposée le moins de temps possible à des tentatives d'attaque ou de cryptanalyse dans le récepteur. Typiquement, la clé $k_{i,m}$ est obtenue pendant la cryptopériode précédente, c'est-à-dire pendant l'intervalle $[t_{i-1,m}; t_{i,m}[$. Dans ces conditions, la clé $k_{i,m}$ est exposée à des tentatives d'attaque ou de cryptanalyse uniquement pendant une partie de l'intervalle $[t_{i-1,m}; t_{i,m}[$, c'est-à-dire généralement pendant au plus une durée de l'ordre d'une dizaine de secondes. La courte durée de cet intervalle rend difficile les tentatives d'attaque.

[004] Pour obtenir la clé $k_{i,m}$, le récepteur doit se connecter à un serveur de clés.

Ainsi, si rien de particulier n'est prévu, le récepteur doit se connecter au serveur de

clés à chaque cryptopériode. Etant donné que le nombre de récepteurs qui doivent se connecter au même serveur de clés peut être très élevé, c'est-à-dire supérieur à 1000 ou 10000, le nombre de connexions que le serveur de clés doit être capable de gérer pendant une cryptopériode est lui aussi très élevé.

5 [005] Pour limiter les ressources informatiques nécessaires pour implémenter un tel serveur de clés, la demande EP2567500 divulgue notamment de transmettre à chaque récepteur, lors d'une connexion au serveur de clés, non pas seulement la clé $k_{1,m}$ valide pour déchiffrer la cryptopériode $CP_{1,m}$ suivante, mais une succession $\{k_{1,m}; \dots; k_{L,m}\}$ de L clés pour déchiffrer les L cryptopériodes suivantes. Ainsi, le récepteur
10 n'a pas besoin de se connecter au serveur de clés à chaque cryptopériode mais seulement toutes les L cryptopériodes.

[006] La demande EP2567500 divulgue aussi qu'envoyer ainsi une telle succession de clés au récepteur, bien avant que ces clés soient utilisables, diminue la sécurité du procédé. En effet, par exemple, la clé $k_{L,m}$ est reçue, comme la succession
15 $\{k_{1,m}; \dots; k_{L,m}\}$ de L clés, avant la cryptopériode $CP_{1,m}$ et n'est utilisée qu'à partir de l'instant $t_{L,m}$. La clé $k_{L,m}$ est donc exposée aux tentatives d'attaques pendant L cryptopériodes successives. Par comparaison, cette même clé $k_{L,m}$ ne serait exposée aux tentatives d'attaque que pendant au plus la durée d'une seule cryptopériode si aucune clé $k_{i,m}$ n'était transmise à l'avance au récepteur.

20 [007] Pour remédier à ce problème tout en limitant le nombre de connexions simultanées au serveur de clés, la demande EP2567500 propose d'adapter le nombre L de clés transmises à l'avance à chaque récepteur en fonction de la probabilité que ce récepteur soit victime d'une tentative d'attaque. Ainsi, il est possible à la fois de diminuer le nombre de connexions à établir avec le serveur de clés tout
25 en maintenant un niveau de sécurité élevé.

[008] De l'état de la technique est également connu de :

- EP2460308A1, et

- Biming Tian et Al : "An Efficient Self-Healing Key Distribution Scheme", New Technologies, Mobility and Security, NTMS '08', IEEE, 5/11/2008, pages 1-5.

30 EP24660308 décrit une solution pour augmenter la robustesse d'un système sécurisé de transmission d'informations vis-à-vis de défaillances, tels qu'une perte de paquets, d'un réseau de transmission d'informations. Pour cela, seulement en cas de défaillance du réseau, un récepteur peut construire une clé de déchiffrement utilisable pendant un intervalle de temps $[i; i+1[$ à partir d'informations utilisées pour construire
35 la précédente clé de chiffrement. En cas de fonctionnement normal du réseau de transmission d'informations, aucune diminution du nombre de connexion du récepteur au serveur n'est obtenue.

[009] L'invention vise à résoudre le même problème que celui mentionné dans EP2567500 mais sans pour cela prendre en compte le niveau de sécurité du
40 récepteur. Elle a donc pour objet un tel procédé conforme à la revendication 1.

[0010] Les modes de réalisation de ce procédé d'obtention peuvent comporter une ou plusieurs des caractéristiques des revendications dépendantes.

[0011] L'invention a également pour objet un procédé de transmission sécurisée d'un contenu numérique.

5 [0012] L'invention a également pour objet un support d'enregistrement d'informations, lisibles par un microprocesseur, comportant des instructions pour la mise en œuvre d'un procédé objet de la présente demande, lorsque ces instructions sont exécutées par le microprocesseur.

10 [0013] L'invention a également pour objet un groupe de récepteurs pour la mise en œuvre du procédé d'obtention objet de la présente demande.

[0014] Enfin, l'invention a également pour objet un serveur de clés pour la mise en œuvre du procédé d'obtention objet de la présente demande.

[0015] L'invention sera mieux comprise à la lecture de la description qui va suivre, donnée uniquement à titre d'exemple non-limitatif et faite en se référant aux dessins
15 sur lesquels :

- la figure 1 est une illustration schématique d'un système sécurisé de transmission et de réception d'un contenu multimédia ;

- la figure 2 est un organigramme d'un procédé de transmission sécurisé d'un contenu multimédia mis en œuvre dans le système de la figure 1 ;

20 - les figures 3 et 4 sont des organigrammes de deux variantes différentes du procédé de la figure 2.

[0016] Chapitre I : Terminologies

[0017] Dans ces figures, les mêmes références sont utilisées pour désigner les mêmes éléments.

25 [0018] Dans la suite de cette description, les caractéristiques et fonctions bien connues de l'homme du métier ne sont pas décrites en détail.

[0019] Les exemples de modes de réalisation sont donnés dans le contexte particulier des systèmes d'accès conditionnels à des contenus multimédias. Par conséquent, la terminologie propre à ce contexte est employée. Ci-dessous quelques
30 précisions sont données sur cette terminologie. Toutefois, pour plus d'informations sur cette terminologie, le lecteur peut se reporter au document suivant : « *Functional Model of Conditional Access System* », EBU Review, Technical European Broadcasting Union, Brussels, BE, n° 266, 21 décembre 1995.

[0020] Ici, les termes « embrouiller », respectivement « désembrouiller », et
35 « chiffrer », respectivement « déchiffrer », sont considérés comme des synonymes.

[0021] Par « contenu multimédia » on désigne un contenu audio et/ou visuel destiné à être restitué sous une forme directement perceptible et compréhensible par un être humain. Typiquement, un contenu multimédia correspond à une succession d'images formant un film, une émission de télévision ou de la publicité. Un contenu
40 multimédia peut également être un contenu interactif tel qu'un jeu.

[0022] Une donnée « en clair » correspond à une donnée avant que celle-ci ne soit embrouillée ou chiffrée. Celle-ci peut donc être rendue directement compréhensible par un être humain sans avoir recours à des opérations de désembrouillage et sans que sa visualisation soit soumise à certaines conditions.

5 [0023] Pour sécuriser et soumettre la visualisation de contenus multimédias à certaines conditions, comme la souscription d'un abonnement payant par exemple, les contenus multimédias sont diffusés sous forme embrouillée et non pas en clair. Plus précisément, chaque contenu multimédia est divisé en une succession de cryptopériodes. Pendant toute la durée d'une cryptopériode, les conditions d'accès au
10 contenu multimédia embrouillé demeurent inchangées. En particulier, pendant toute la durée d'une cryptopériode, le contenu multimédia est embrouillé avec une même clé cryptographique connue sous le terme de « mot de contrôle ». Généralement, le mot de contrôle varie d'une cryptopériode à l'autre.

[0024] On désigne ici par « cryptogramme » d'une donnée, une information
15 insuffisante à elle seule pour retrouver la donnée en clair, c'est-à-dire la donnée telle qu'elle était avant l'application du chiffrement qui a permis d'en établir le cryptogramme. Ainsi, si la transmission du cryptogramme est interceptée, la seule connaissance du cryptogramme ne permet pas de retrouver la donnée en clair. Pour retrouver la donnée en clair, le cryptogramme doit être combiné avec une information
20 secrète. L'information secrète est typiquement une clé cryptographique permettant de déchiffrer ce cryptogramme. Toutefois, le cryptogramme peut aussi être une référence à une donnée stockée dans une table contenant une multitude de données similaires. Dans ce cas, l'information secrète est la table associant à chaque cryptogramme une donnée en clair.

25 [0025] Chapitre II : Notations

[0026] Les notations définies dans ce chapitre sont utilisées dans tout le texte de cette demande.

[0027] CP_m est la m-ième cryptopériode du contenu multimédia.

[0028] L'indice « m » est un numéro d'ordre qui identifie une position par rapport à
30 un point de référence. Le point de référence peut être une origine absolue indépendante du contenu multimédia ou une origine relative au contenu multimédia diffusé. Par la suite, le point de référence est une origine relative. Le point de référence est le début du contenu multimédia. Ainsi, la cryptopériode CP_1 est la première cryptopériode du contenu multimédia, la cryptopériode CP_2 est la deuxième
35 cryptopériode du contenu multimédia et ainsi de suite.

[0029] k_m est la clé cryptographique, connue sous le terme de « mot de contrôle », qui est uniquement utilisée pour embrouiller et désembrouiller la cryptopériode CP_m . La clé k_m est donc utilisée immédiatement après la clé précédente k_{m-1} et immédiatement avant la clé suivante k_{m+1} .

40 [0030] t_m et t_{m+1} sont, respectivement, les instants où débute et où se termine la

cryptopériode CP_m lorsqu'elle est jouée par un récepteur. L'instant t_1 correspond donc au début de la première cryptopériode CP_1 du contenu multimédia.

[0031] L'intervalle de temps $[t_m ; t_{m+1}[$ correspond à l'intervalle de temps pendant lequel la cryptopériode CP_m est jouée par le récepteur. L'intervalle $[t_m ; t_{m+1}[$ est aussi un intervalle de validité de la clé k_m qui a été utilisée pour embrouiller la cryptopériode CP_m . En effet, la clé k_m peut donc être utilisée pour désembrouiller le contenu multimédia uniquement pendant l'intervalle $[t_m ; t_{m+1}[$. En dehors de cet intervalle de temps, la clé k_m ne permet pas de désembrouiller correctement le contenu multimédia.

[0032] V_m est la durée de l'intervalle $[t_m ; t_{m+1}[$. Lorsque tous les intervalles $[t_m ; t_{m+1}[$ ont la même durée, et donc lorsque la durée V_m est indépendante de l'indice m , cette durée est simplement notée V .

[0033] ECM_m est un message ECM (Entitlement Control Message). Le message ECM_m est le message ECM contenant l'identifiant de la clé k_m qui permet de désembrouiller la cryptopériode CP_m .

[0034] SE_p est la succession de clés $\{k_1 ; \dots ; k_m ; k_{m+1} ; \dots ; k_N\}$, c'est-à-dire la séquence ordonnée des clés k_1 à k_N , où N est le nombre de clés dans la succession SE_p .

[0035] SR_m est la succession de clés $\{k_m ; k_{m+1} ; \dots ; k_{m+L-1}\}$, c'est-à-dire la séquence ordonnée des clés k_m à k_{m+L-1} , où L est le nombre de clés dans la succession SR_m . Le nombre L est systématiquement inférieur ou égal à un seuil préenregistré L_{max} . Le seuil L_{max} est inférieur à N , et par exemple deux ou dix ou cent fois inférieur au nombre N . Le nombre L_{max} est un nombre entier supérieur ou égal à deux, préenregistré dans la mémoire 110. Le nombre L_{max} est la longueur maximale d'une succession SR_m . Ainsi, chaque succession SR_m est beaucoup plus petite que la succession SE_p .

[0036] $k_{i,m}$ est la i -ième clé de la succession SR_m . L'indice i indique la position de la clé $k_{i,m}$ dans la succession SR_m par rapport à la première clé k_m de cette succession. L'indice i est le numéro d'ordre de la clé $k_{i,m}$ dans la succession SR_m . La valeur de l'indice i pour la première clé de la succession SR_m est égale à un. Ainsi, la clé $k_{i,m}$ est égale à la clé k_{i+m-1} . Cette même notation est aussi utilisée pour toute variable associée à la clé $k_{i,m}$. Par exemple, la notation $C_{i,m}$ désigne l'information de contrôle C_{i+m-1} associée à la clé $k_{i,m}$.

[0037] Chapitre III : Exemples de modes de réalisation :

[0038] La figure 1 représente un système 2 d'émission et de réception de contenus multimédias embrouillés. Par exemple, un contenu multimédia correspond à une séquence d'un programme audiovisuel tel qu'une émission de télévision ou un film.

[0039] Les contenus multimédias en clair sont générés par une ou plusieurs sources 4 et transmis à un dispositif 6 de diffusion. Le dispositif 6 diffuse les contenus

multimédias simultanément vers une multitude de récepteurs à travers un réseau 8 de transmission d'informations. Le nombre de récepteurs est généralement très élevé, c'est-à-dire supérieur à 1000 ou 10 000. Pour simplifier la figure 1, seuls trois récepteurs 10 à 12 sont représentés.

5 [0040] Le réseau 8 est typiquement un réseau grande distance de transmission d'informations tel que le réseau Internet ou un réseau satellitaire ou tout autre réseau de diffusion tel que celui utilisé pour la transmission de la télévision numérique terrestre (TNT).

[0041] Le dispositif 6 comprend un encodeur 16 qui compresse les contenus
10 multimédias qu'il reçoit. L'encodeur 16 traite des contenus multimédias numériques. Par exemple, cet encodeur fonctionne conformément à la norme MPEG2 (Moving Picture Expert Group – 2) ou la norme UIT-T H264.

[0042] Les contenus multimédias compressés sont dirigés vers une entrée 20 d'un embrouilleur 22. L'embrouilleur 22 embrouille chaque contenu multimédia compressé
15 pour conditionner sa visualisation à certaines conditions telles que l'achat d'un titre d'accès par les utilisateurs des récepteurs. Les contenus multimédias embrouillés sont restitués sur une sortie 24 raccordée à l'entrée d'un multiplexeur 26.

[0043] L'embrouilleur 22 embrouille chaque cryptopériode CP_m du contenu multimédia compressé à l'aide d'une clé k_m respective, connue sous le terme de «
20 mot de contrôle » dans le domaine des systèmes d'accès conditionnel. Typiquement, cet embrouillage est conforme à une norme telle que la norme DVB-CSA (Digital Video Broadcasting – Common Scrambling Algorithm), ISMA Cryp (Internet Streaming Media Alliance Cryp), SRTP (Secure Real-time Transport Protocol), AES (Advanced Encryption Standard), etc.

25 [0044] La durée V_m d'une cryptopériode CP_m est typiquement supérieure à cinq secondes et, de préférence, comprise entre 5 secondes et 10 minutes. Dans ce mode de réalisation, toutes les cryptopériodes CP_m ont la même durée V .

[0045] Le dispositif 6 comporte également un système 28 de contrôle d'accès. Le système 28 est plus connu sous l'acronyme CAS (Conditional Access System). Ici,
30 pour chaque cryptopériode CP_m , le système 28 :

- transmet à l'embrouilleur 22 la clé k_m à utiliser pour embrouiller cette cryptopériode, et

- génère, le message ECM_m contenant au moins l'identifiant Id_m de la clé k_m à utiliser pour désembrouiller la cryptopériode CP_m .

35 [0046] Le message ECM_m est ici associé à la cryptopériode CP_m par le multiplexeur 26. Pour cela, dans cet exemple, le message ECM_m et la cryptopériode CP_m sont synchronisés temporellement l'un avec à l'autre en les multiplexant dans le même signal audiovisuel transmis sur le réseau 8. Plus précisément, ici, le message ECM_m est transmis aux récepteurs pendant la cryptopériode CP_{m-1} qui précède
40 immédiatement la cryptopériode CP_m .

[0047] Ici, les récepteurs 10 à 12 sont identiques et seul le récepteur 10 est décrit plus en détail.

[0048] Le récepteur 10 comprend un module de réception 70 de contenus multimédias diffusés. Ce module 70 est raccordé à l'entrée d'un démultiplexeur 72. Le démultiplexeur 72 transmet d'un côté chaque cryptopériode CP_m embrouillée reçue à un désembrouilleur 74 et d'un autre côté les messages ECM et EMM (Entitlement Management Message) à un processeur 76.

[0049] Le processeur 76 traite des informations confidentielles telles que des clés cryptographiques. Pour préserver la confidentialité de ces informations, il est conçu pour être le plus robuste possible vis-à-vis des tentatives d'attaques menées par des pirates informatiques. Il est donc plus robuste vis-à-vis de ces attaques que les autres composants du récepteur 10. Cette robustesse est par exemple obtenue en implémentant un module logiciel dédié à la protection des informations secrètes.

[0050] Le processeur 76 est par exemple réalisé à l'aide d'un microprocesseur 77 programmable apte à exécuter des instructions enregistrées sur un support d'enregistrement d'informations. A cet effet, le processeur 76 comporte également une mémoire 78 contenant les instructions nécessaires pour l'exécution du procédé de la figure 2.

[0051] La mémoire 78 contient également par exemple :

- une clé secrète de cryptographie symétrique, unique et partagée avec un serveur 106 de clés, ou
- une clé privée de cryptographie asymétrique et un certificat cryptographique associé, c'est-à-dire la clé publique associée, pour authentifier le récepteur 10.

La mémoire 78 contient également une table locale 79 contenant les clés k_m actuellement disponibles.

[0052] Le désembrouilleur 74 désembrouille le contenu multimédia embrouillé à l'aide de la clé k_m transmise par le processeur 76. Le contenu multimédia désembrouillé est transmis à un décodeur 80 qui le décode. Le contenu multimédia décompressé ou décodé est transmis à une carte graphique 82 qui pilote l'affichage de ce contenu multimédia sur un afficheur 84 équipé d'un écran 86.

[0053] L'afficheur 84 affiche en clair le contenu multimédia sur l'écran 86.

[0054] Le récepteur 10 comprend également un émetteur-récepteur 88 permettant d'établir une connexion sécurisée entre le processeur 76 et une tête de réseau 90 par l'intermédiaire d'un réseau 92 de transmission d'informations. Par exemple, le réseau 92 est un réseau grande distance de transmission d'informations et plus précisément un réseau à commutation de paquets tel que le réseau Internet. La connexion sécurisée est par exemple un tunnel sécurisé à l'aide du certificat cryptographique du processeur 76.

[0055] La tête de réseau 90 comprend un module 100 de gestion des titres d'accès des différents utilisateurs du système 2. Ce module 100 est plus connu sous

le terme anglais de « subscriber authorization system ». Ce module 100 génère et maintient à jour une base de données 102. La base de données 102 associe à chaque identifiant d'utilisateur les titres d'accès acquis par cet utilisateur. Cette base de données 102 est stockée dans une mémoire 104.

5 [0056] La tête de réseau 90 comprend également un serveur 106 de clés. Le serveur 106 comprend notamment un générateur 108 de clés k_m et une mémoire 110.

[0057] La mémoire 110 contient :

- un compteur C_{nbc} du nombre de connexions par unité de temps,

10 - une table 112 contenant chaque clé k_m générée et, pour chaque clé k_m dont l'indice est supérieur ou égal à 1, une information de contrôle C_m ,

[0058] Le compteur C_{nbc} dénombre le nombre de connexions, par unité de temps, établies par l'ensemble des récepteurs du système 2 avec le serveur 106. Typiquement, ce compteur C_{nbc} contient le nombre de telles connexions recensées pendant une fenêtre de temps glissante de durée ΔT . Cette fenêtre glissante s'achève
15 à l'instant courant. La durée ΔT est par exemple comprise entre V et 24 heures ou entre V et 1 heure.

[0059] Typiquement, le serveur 106 est réalisé à partir d'un microprocesseur 114 programmable apte à exécuter des instructions enregistrées sur un support d'enregistrement d'informations. A cet effet, la mémoire 110 comprend également les
20 instructions pour l'exécution du procédé de la figure 2.

[0060] Le fonctionnement du système 2 va maintenant être décrit plus en détail en référence au procédé de la figure 2. Ici, on suppose qu'initialement, la table 79 est vide.

[0061] Le procédé débute, en réponse à une requête de transmission d'un
25 contenu, par une phase 114 d'initialisation des valeurs des différents paramètres nécessaires à la réalisation des étapes suivantes. Les valeurs de ces paramètres sont enregistrées dans la mémoire 110. Ces paramètres sont présentés au fur et à mesure de la description des étapes suivantes. Par conséquent, bien que chronologiquement située avant les étapes suivantes, dans cette description, le
30 réglage des valeurs de ces paramètres lors de la phase 114 est décrit après ces étapes. Une fois la phase 114 terminée, la transmission du contenu multimédia peut alors débuter.

[0062] Lors d'une étape 116, le générateur 108 génère, les unes après les autres, les clés k_m de la succession SE_p . Chaque clé k_m de cette succession est utilisée pour
35 embrouiller la cryptopériode correspondante CP_m du contenu multimédia à diffuser. Au cours du temps, le générateur 108 génère successivement les clés k_1 à k_N . Ici, le nombre N de clés dans la succession SE_p est par exemple égal ou supérieur au nombre de cryptopériodes CP_m du contenu multimédia à embrouiller.

[0063] Pour générer la succession SE_p , le générateur 108 commence par obtenir
40 la clé k_1 puis, pour tout indice m supérieur ou égal à deux, il dérive la clé suivante k_m

à partir de la clé précédente k_{m-1} en exécutant un premier algorithme D1 de dérivation de clé k_m .

[0064] Lors d'une opération 117, le générateur 108 obtient la clé k_1 , par exemple, par tirage aléatoire ou pseudo-aléatoire d'un nombre dans un ensemble E_k . Ici, l'ensemble E_k contient tous les nombres entiers dont la représentation binaire comporte au plus N_k bits. La valeur du nombre N_k est pré-enregistrée dans la mémoire 110. Par exemple, la valeur du nombre N_k est égale à 16, 32, 48 ou 56. La clé k_1 générée est ensuite enregistrée dans la table 112.

[0065] Ensuite, lors d'une opération 118, le générateur 108 dérive chaque clé suivante k_m de la clé précédente k_{m-1} en exécutant le même algorithme D1 paramétré par la valeur de la clé précédente k_{m-1} . Ainsi, la clé k_m ne peut pas être générée avant la clé k_{m-1} . Les clés k_m sont donc générées les unes après les autres.

[0066] Ici, l'algorithme D1 est aussi paramétré par un paramètre réglable de complexité, noté PC_p , qui permet, comme cela sera expliqué plus loin, d'augmenter ou de diminuer le nombre moyen d'opérations exécutées par les récepteurs pour obtenir la clé k_m à partir de la clé k_{m-1} . Le paramètre PC_p permet donc d'augmenter ou de diminuer le temps moyen $TC_{i,m}$ d'exécution d'un deuxième algorithme D2 de dérivation de la clé $k_{i,m}$ exécuté par les récepteurs. Le temps moyen $TC_{i,m}$ est le temps qui s'écoule, en moyenne, entre l'instant où un processeur 76 débute l'exécution de l'algorithme D2 pour obtenir la clé $k_{i,m}$ et l'instant où ce processeur 76 termine cette exécution de l'algorithme D2 car il a obtenu la clé $k_{i,m}$. Le temps moyen $TC_{i,m}$ correspond donc typiquement à la moyenne des temps mis par une multitude de processeurs 76 de terminaux du système 2 pour obtenir la clé $k_{i,m}$ en exécutant l'algorithme D2. Dans ce premier mode de réalisation, le paramètre PC_p est la taille d'un ensemble E_R de nombres entiers. L'ensemble E_R contient tous les nombres entiers dont la représentation binaire comporte au plus N_R bits. Par exemple, le nombre N_R est égal à la valeur du paramètre PC_p .

[0067] Dans ce mode de réalisation, l'algorithme D1 est un calcul de clés faisant intervenir une succession de tirages aléatoires d'un nombre R dans l'ensemble E_R . Par la suite, ce type d'algorithme est appelé « calcul de clés avec aléa ». Par exemple, à chaque exécution de l'algorithme D1, le générateur 108 réalise les opérations suivantes pour générer la clé suivante k_m :

1) Tirage aléatoire ou pseudo-aléatoire d'un nombre R_m dans l'ensemble E_R , puis

2) Calcul de la clé k_m à l'aide de la relation suivante : $k_m = F_1(R_m // k_{m-1})$, où :

- le symbole « // » désigne une opération de combinaison de la valeur de la clé K_{m-1} avec le nombre R_m , et

- F_1 est une fonction connue du générateur 108 et des récepteurs.

[0068] Par exemple, ici, l'opération « // » est l'opération « ou exclusif » habituellement désignée par le symbole XOR. La fonction F_1 est typiquement une fonction à sens-unique. Par exemple, la fonction F_1 est choisie dans le groupe G_1 de

fonctions à sens-unique constitué des fonctions de chiffrement symétriques, des fonctions de chiffrement asymétriques et des fonctions de hachage.

[0069] Dans ce mode de réalisation, lors d'une opération 119 supplémentaire, pour chaque clé k_m générée en exécutant l'algorithme D1, le générateur 108 génère

5 aussi une information de contrôle C_m . L'information de contrôle C_m est un paramètre de l'algorithme D2 nécessaire pour obtenir la clé k_m à partir de la clé k_{m-1} . Dans ce mode de réalisation, l'information de contrôle C_m est une information qui permet au récepteur d'obtenir la clé k_m à partir de la clé précédente k_{m-1} sans connaître le nombre R_m . Par exemple, l'information de contrôle C_m est calculée par le générateur
10 108 à l'aide de la relation suivante : $C_m = H_1(k_m)$, où H_1 est une fonction cryptographique à sens unique. Puisque H_1 est une fonction à sens unique, c'est-à-dire pour laquelle il est très difficile de calculer une pré-image à partir de son image, il n'est pas possible de déduire la clé k_m à partir de la seule connaissance de l'information de contrôle C_m . La fonction H_1 est typiquement elle aussi choisie dans le
15 groupe G_1 de fonctions. Ici, les fonctions F_1 et H_1 sont identiques. Par exemple, les fonctions F_1 et H_1 sont toutes les deux une même fonction de hachage.

[0070] Chaque clé k_m générée par l'exécution de l'algorithme D1 est enregistrée, associée à l'information de contrôle C_m , dans la table 112. Le générateur 108 transmet également chaque clé k_m au système 28.

20 [0071] L'étape 116 est déclenchée suffisamment tôt de manière à ce que, quelle que soit la valeur de l'indice m , la clé k_m est disponible à temps pour que l'embrouilleur 22 embrouille la cryptopériode CP_m avec la clé k_m . De plus, ici, l'exécution de l'étape 116 est déclenchée suffisamment tôt pour que, à chaque instant t_m , la table 112 contienne déjà au moins les clés k_m à $k_{m+L_{\max}}$ et les informations de
25 contrôle associées C_m à $C_{m+L_{\max}}$. Le nombre $L_{\max}$ est un nombre entier supérieur ou égal à un préenregistré dans la mémoire 110. Le nombre $L_{\max}$ est la longueur maximale d'une succession SR_m .

[0072] En parallèle de l'étape 116 ou après l'étape 116, lors d'une étape 120, le dispositif 6 divise le contenu multimédia en cryptopériodes successives, embrouille
30 chaque cryptopériode CP_m à l'aide de la clé k_m correspondante, puis transmet les cryptopériodes embrouillées. Les messages ECM_m contenant les identifiants Id_m des clés k_m sont multiplexés avec les cryptopériodes correspondantes du contenu multimédia transmis. Ce multiplexage permet de synchroniser la transmission de chaque identifiant Id_m avec la transmission de la cryptopériode CP_m du contenu
35 multimédia. Ici, l'identifiant Id_m est transmis aux récepteurs uniquement pendant la cryptopériode CP_{m-1} précédant la cryptopériode CP_m . Dans le cas de la première cryptopériode CP_1 , l'identifiant Id_1 est transmis aux récepteurs pendant un intervalle de temps $[t_0 ; t_1[$ précédant immédiatement la première cryptopériode CP_1 . La durée de l'intervalle $[t_0 ; t_1[$ est par exemple égale à la durée V d'une cryptopériode.

40 [0073] Le contenu multimédia embrouillé est reçu sensiblement en même temps

par chacun des récepteurs du système 2. Les étapes suivantes sont donc exécutées sensiblement en parallèle pour chacun de ces récepteurs. Les étapes suivantes sont décrites dans le cas particulier du récepteur 10.

[0074] Lors d'une étape 122, le signal audiovisuel contenant le contenu multimédia embrouillé et les messages ECM_m est reçu par le module 70 de réception.

[0075] Ensuite, lors d'une étape 124, le démultiplexeur 72 extrait, au fur et à mesure de leur réception, les cryptopériodes CP_m embrouillées du contenu multimédia embrouillé et les messages ECM_m. Le démultiplexeur 72 transmet les cryptopériodes CP_m embrouillées extraites vers le désembrouilleur 74. Les messages

ECM_m extraits sont quant à eux transmis au processeur 76.

[0076] Au moins en réponse à chaque première réception d'un message ECM_m, et au plus tard une durée prédéterminée d avant l'instant t_m, lors d'une étape 126, le processeur 76 vérifie s'il a déjà obtenu la clé k_{i,m}. Ici, pour cela, il recherche dans la table 79 si celle-ci contient déjà la clé k_m correspondant à l'identifiant Id_m contenu dans le message ECM_m reçu. La durée d est fixée par l'opérateur du système 2 comme majorant légèrement le temps nécessaire à l'obtention par un récepteur d'une clé k_{i,m} auprès du serveur 106.

[0077] Dans l'affirmative, lors d'une étape 128, le processeur 76 envoie au désembrouilleur 74 la clé k_m trouvée dans la table 79. Aucune connexion avec le serveur 106 n'est alors établie pour obtenir la clé k_m.

[0078] Ensuite, lors d'une étape 130, le désembrouilleur 74 désembrouille la cryptopériode CP_m reçue à l'aide de la clé k_m.

[0079] Puis, lors d'une étape 132, la cryptopériode CP_m désembrouillée est décodée par le décodeur 80 puis transmise à la carte vidéo 82.

[0080] Enfin, lors d'une étape 134, la carte vidéo 82 transforme la cryptopériode CP_m désembrouillée et décodée en un signal vidéo. Ici, ce signal vidéo est ensuite transmis au dispositif d'affichage 84.

[0081] En réponse, le dispositif 84 affiche la cryptopériode CP_m du contenu multimédia sur l'écran 86 de manière directement perceptible et compréhensible par un être humain.

[0082] Si lors de l'étape 126, la clé k_m correspondant à l'identifiant Id_m n'est pas contenue dans la table 79, alors le procédé se poursuit par une étape 140 et non pas directement par l'étape 128.

[0083] Lors de l'étape 140, le processeur 76 établit une connexion sécurisée avec le serveur 106 et transmet par l'intermédiaire de cette connexion une requête pour recevoir les informations nécessaires à l'obtention de la clé k_m. Par exemple, cette requête contient notamment l'identifiant Id_m de la clé k_m.

[0084] Cette requête est transmise au serveur 106 par l'intermédiaire de l'émetteur-récepteur 88 et du réseau 92. Tous les échanges d'informations entre le processeur 76 et le serveur 106 se font par l'intermédiaire d'un tunnel sécurisé établi

au travers du réseau 92. L'établissement du tunnel a requis l'authentification et l'identification du récepteur par le serveur 106, par exemple à l'aide du certificat cryptographique contenu dans la mémoire 78. Ainsi, le serveur 106 dispose de l'identifiant Id_T du récepteur qui lui envoie une requête.

5 [0085] Puisque la table 79 des récepteurs est initialement vide, l'étape 140 est systématiquement exécutée par chacun des récepteurs pendant l'intervalle de temps $[t_0 ; t_1[$ précédant immédiatement la première cryptopériode CP_1 à désembrouiller du contenu multimédia. Ensuite, l'étape 140 est exécutée à chaque fois que la clé k_m requise pour désembrouiller la cryptopériode CP_m est absente de la table 79.

10 [0086] La réception par le serveur 106 de cette requête l'informe que la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$. En réponse, lors d'une étape 142, le serveur 106 procède à la mise à jour du compteur C_{nbc} de connexions par unité de temps. Par exemple, le serveur 106 compte le nombre de connexions établies, y compris celle-ci, entre tous les récepteurs du système 2 et lui-même pendant la fenêtre glissante de
15 durée ΔT . Ici, le serveur 106 compte seulement les connexions lors desquelles les informations nécessaires pour obtenir une clé k_m sont requises.

[0087] Ensuite, lors d'une étape 144, le serveur 106 obtient la valeur d'un nombre entier L . Le nombre L permet d'ajuster le nombre de cryptopériodes qui vont s'écouler entre cette connexion au serveur 106 du récepteur 10 et la prochaine connexion
20 obligatoire du récepteur 10 au serveur 106. Plus précisément, le nombre L fixe le nombre maximum de clés suivantes que le récepteur 10 peut dériver à partir de la clé k_m sans se connecter à nouveau au serveur 106. Le nombre L fixe donc la longueur de la succession SR_m des clés k_m à k_{m+L-1} que le récepteur 10 peut obtenir à partir des seules informations contenues dans la réponse à sa requête.

25 [0088] Le nombre L est ici déterminé pour répartir de façon la plus uniforme possible les connexions des récepteurs au serveur 106. A cet effet, lors de la première connexion du récepteur 10, c'est-à-dire celle établie pour obtenir la clé k_1 , le serveur 106 choisit une première valeur du nombre L différente de celle choisie pour d'autres récepteurs du système 2. Par exemple, le serveur 106 tire aléatoirement
30 cette première valeur dans l'intervalle $[2 ; L_{max}]$. Dans un autre exemple, le serveur 106 tire aléatoirement la première valeur successivement dans des sous-intervalles formant une partition de l'intervalle $[2 ; L_{max}]$. Ensuite, lors des connexions suivantes du récepteur 10, le serveur 106 utilise une deuxième valeur du nombre L constante et identique pour tous les récepteurs du système 2. Une connexion suivante est une
35 connexion établie pour obtenir une clé k_m , où l'indice m est strictement supérieur à un. La deuxième valeur du nombre L est par exemple pré-enregistrée dans la mémoire 110 lors de la phase 114. Cette deuxième valeur du nombre L est également comprise entre 2 et L_{max} .

[0089] Lors d'une étape 146, en réponse à la requête du récepteur 10, le serveur
40 106 transmet au processeur 76, par l'intermédiaire de la connexion établie à l'étape

140, les informations nécessaires pour que le récepteur 10 puisse obtenir la séquence SR_m de clés sans que ce récepteur ait besoin d'établir une connexion suivante avec le serveur 106. Autrement dit, le serveur 106 transmet au récepteur 10 lors de cette connexion, toutes les informations nécessaires pour qu'il puisse obtenir

5 les clés $k_{1,m}$ à $k_{L,m}$. A cet effet, dans ce mode de réalisation, le serveur 106 transmet et le récepteur 10 reçoit lors de cette connexion les informations suivantes :

- la valeur actuelle du paramètre PC_p ,
- la clé $k_{1,m}$, et
- les informations de contrôle $C_{2,m}$ à $C_{L,m}$.

10 [0090] Après avoir transmis ces informations, la connexion entre le serveur 106 et le récepteur 10 est interrompue. La connexion est donc interrompue avant l'instant t_m où débute la cryptopériode CP_m .

[0091] Ensuite, lors d'une étape 148, le processeur 76 enregistre la clé $k_{1,m}$ reçue dans la table 79, puis le procédé retourne à l'étape 128. Ainsi, la clé $k_{1,m}$ est transmise

15 au désembrouilleur 74 avant l'instant t_m de sorte que la cryptopériode CP_m peut être correctement désembrouillée à temps.

[0092] En parallèle, lors d'une étape 150, le processeur 76 déclenche immédiatement l'obtention des clés suivantes : $k_{2,m}$ à $k_{L,m}$ à partir des informations reçues en réponse à sa requête. L'étape 150 est déclenchée systématiquement

20 après la réception de la clé $k_{1,m}$. En particulier, le déclenchement de l'étape 150 est indépendant de l'état de fonctionnement des réseaux 92 et 8.

[0093] Pour cela, lors de l'étape 150, le processeur 76 exécute l'algorithme D2 de dérivation de clé. L'algorithme D2 permet d'obtenir la clé $k_{i,m}$ suivante à partir de la clé $k_{i-1,m}$ précédente et également à partir, ici, des valeurs du paramètre PC_p et de

25 l'information de contrôle $C_{i,m}$. Ainsi, l'algorithme D2 est exécuté une première fois pour obtenir la clé $k_{2,m}$ à partir de la clé $k_{1,m}$ reçue, puis une deuxième fois pour obtenir la clé $k_{3,m}$ à partir de la clé $k_{2,m}$ et ainsi de suite jusqu'à obtenir la clé $k_{L,m}$ à partir de la clé $k_{L-1,m}$.

[0094] Ici, à chaque fois que l'algorithme D2 est exécuté par le processeur 76, ce processeur réalise les opérations suivantes pour obtenir la clé $k_{i,m}$:

30

- 1) Le processeur 76 tire aléatoirement un nombre R dans l'ensemble E_R , puis
- 2) Le processeur 76 calcule une clé candidate k_{cd} selon la relation suivante : $k_{cd} = F_1(R//k_{i-1,m})$, puis
- 3) Le processeur 76 calcule une information de contrôle C_{cd} selon la relation
- 35 suivante : $C_{cd} = H_1(k_{cd})$, puis
- 4) Le processeur 76 compare l'information de contrôle C_{cd} à l'information de contrôle $C_{i,m}$ reçue lors de l'étape 146, puis

5) Si les informations de contrôle C_{cd} et $C_{i,m}$ sont identiques, la clé k_{cd} est égale à la clé $k_{i,m}$ et la clé $k_{i,m}$ est donc obtenue. La clé $k_{i,m}$ est alors enregistrée dans la liste 79

40 et l'exécution de l'algorithme D2 est terminée. Dans le cas contraire, le processeur 76

retourne à l'opération 1). Ainsi, les opérations 1) à 5) sont réitérées en boucle jusqu'à ce que la clé $k_{i,m}$ soit obtenue, ou que le processeur 76 mette en œuvre l'étape 126 pour déterminer s'il a déjà obtenu la clé $k_{i,m}$, puis l'étape 140 pour l'obtenir auprès du serveur 106. Dans ce dernier cas, la clé $k_{i,m}$ n'a pas été obtenue par le récepteur au moins la durée d avant l'instant t_m , et l'étape 150 est interrompue avant son obtention par exécution de l'algorithme D2.

[0095] Les fonctions F_1 , H_1 et $//$ mises en œuvre par le processeur 76 sont les mêmes que celles mises en œuvre par le générateur 108 pour construire la succession SE_p .

10 [0096] La taille de l'ensemble E_R est déterminée à partir du paramètre PC_p reçu par le processeur 76 en même temps que la clé $k_{1,m}$ et des informations de contrôle $C_{i,m}$.

[0097] Le temps moyen $TC_{i,m}$ d'exécution de cet algorithme D2 dépend de la taille C_a de l'ensemble E_R . En effet, plus la taille C_a de l'ensemble E_R est grande, plus le nombre moyen de tirages aléatoires réalisés avant de tirer un nombre R égal au nombre R_m qui permet d'obtenir la clé $k_{i,m}$ est important. Ici, ce nombre moyen de tirages aléatoires est égal à $C_a/2$. Le temps moyen $TC_{i,m}$ pour obtenir la clé $k_{i,m}$ est donc donné par la relation suivante : $TC_{i,m} = (C_a/2)t_{1-5}$, où t_{1-5} est le temps nécessaire pour que le processeur 76 exécute une fois les opérations 1) à 5).

20 [0098] Après l'étape 146, par exemple en parallèle de l'étape 148, lors d'une étape 160, le serveur 106 compare le compteur C_{nbc} mis à jour à des seuils S_{nbc-h} et S_{nbc-i} prédéterminés enregistrés dans la mémoire 110. Le seuil S_{nbc-h} est égal ou strictement supérieur au nombre de connexions par unité de temps auquel on peut s'attendre si chaque récepteur arrive à calculer à temps chacune des clés $k_{2,m}$ à $k_{L,m}$, et ne se connecte donc au serveur 106, sauf la première fois, qu'après avoir obtenu, par exécution de l'algorithme D2, la dernière clé $k_{L,m}$ d'une succession de clés SR_m . Ainsi, le seuil S_{nbc-h} est supérieur ou égal à $N_{rec}/(L.V)$, où :

- N_{rec} est égal au nombre total de récepteurs du système 2 qui se connectent aux serveurs 106,

30 - le symbole « . » désigne l'opération de multiplication.

Le seuil S_{nbc-h} doit aussi être suffisamment petit pour permettre d'adapter la valeur du paramètre PC_p avant que le compteur C_{nbc} ne devienne très supérieur à $N_{rec}/(L.V)$. Par exemple, le seuil S_{nbc-h} est inférieur à $2N_{rec}/(L.V)$ ou inférieur à $1,5N_{rec}/(L.V)$.

[0099] Si la valeur actuelle du compteur C_{nbc} dépasse le seuil S_{nbc-h} , cela veut dire qu'un trop grand nombre de récepteurs sont incapables de terminer le calcul de la clé $k_{i,m}$ avant l'instant $t_{i,m}$. Cela se traduit par un nombre de connexions établies avec le serveur 106 bien plus important qu'initialement prévu. En réponse, lors d'une étape 162, le serveur 106 modifie la valeur du paramètre PC_p pour que les récepteurs puissent calculer plus rapidement les clés $k_{i,m}$ suivantes. Dans ce mode de réalisation, pour cela, la valeur du paramètre PC_p est diminuée afin de diminuer la taille C_a de

l'ensemble E_R . Ensuite, le procédé retourne à l'étape 116 pour générer les clés suivantes de la succession SE_p en prenant en compte la nouvelle valeur du paramètre PC_p .

[00100] Si la valeur du compteur C_{nbc} tombe en dessous du seuil S_{nbc-i_s} , cela veut dire qu'un trop petit nombre de récepteurs sont incapables de terminer le calcul de la clé $k_{i,m}$ avant l'instant $t_{i,m}$. Cela correspond généralement à une situation où le temps moyen $TC_{i,m}$ est trop petit devant la durée V . Le seuil S_{nbc-i} est strictement inférieur au seuil S_{nbc-h} et généralement proche de la limite $N_{rec}/(L.V)$. Par exemple, le seuil S_{nbc-i} appartient à l'intervalle $[N_{rec}/(L.V); 1,3N_{rec}/(L.V)]$ ou à l'intervalle $[N_{rec}/(L.V); 1,1N_{rec}/(L.V)]$. Dans ce cas, en réponse, lors d'une étape 164, le serveur 106 modifie la valeur du paramètre PC_p pour faire augmenter le temps moyen $TC_{i,m}$ de calcul de la clé $k_{i,m}$ puis retourne aux étapes 116 et 120. Pour cela, la valeur du paramètre PC_p est augmentée afin d'augmenter la taille C_a de l'ensemble E_R .

[00101] Si la valeur du compteur C_{nbc} est comprise entre les seuils S_{nbc-h} et S_{nbc-l} , alors le serveur 106 laisse inchangée la valeur actuelle du paramètre PC_p .

[00102] Des méthodes pour régler, lors de la phase 114, les différents paramètres précédemment décrits sont maintenant introduites.

[00103] La deuxième valeur préenregistrée du nombre L est, par exemple, fixée pour obtenir un nombre cible N_{cn} de connexions au serveur 106 par seconde. Le nombre N_{cn} est choisi par le concepteur du système 2. Pour cela, dans ce mode de réalisation particulier où la durée V de l'intervalle de validité de chaque clé k_m est la même, la deuxième valeur du nombre L est déterminée à l'aide de la relation suivante : $L = N_{rec}/(N_{cn}.V)$.

[00104] La valeur initiale du paramètre PC_p est déterminée pour que le temps moyen attendu $TC_{i,m}$ d'exécution de l'algorithme D2 par un récepteur avant d'obtenir la clé $k_{i,m}$ soit supérieur à $0,2V_{i-1,m}$ ou $0,5V_{i-1,m}$ ou $0,9V_{i-1,m}$, où $V_{i-1,m}$ est la durée de la cryptopériode $CP_{i-1,m}$.

[00105] Par exemple, la valeur initiale du paramètre PC_p est conçue pour que, quelle que soit la valeur de l'indice i comprise entre 2 et L , chaque temps moyen $TC_{i,m}$ respecte les conditions (1) à (3) suivantes :

(1)	$\forall i \in [3; L], \quad TC_{i,m} \geq \sum_{q=1}^{i-2} V_{q,m} - \sum_{q=2}^{i-1} TC_{q,m}$
(2)	$TC_{2,m} \geq 0,5V_{1,m}$

(3)	$\forall i \in [2; L], TC_{i,m} \leq V_{i-1,m}$
-----	---

[00106] Dans ce cas, quel que soit l'indice i supérieur ou égal à deux, la clé $k_{i,m}$ ne peut pas être obtenue avant l'instant $t_{i-1,m}$. Ainsi, la clé $k_{L,m}$ est exposée aux tentatives d'attaque uniquement au plus pendant l'intervalle $[t_{L-1,m}; t_{L,m}[$. L'intervalle $[t_{L-1,m}; t_{L,m}[$ est beaucoup plus court que l'intervalle $[t_{1,m}; t_{L,m}[$. L'intervalle $[t_{1,m}; t_{L,m}[$ correspond à l'intervalle de temps pendant lequel la clé $k_{L,m}$ est exposée aux tentatives d'attaque dans les procédés connus, tel que celui décrit dans la demande EP2567500, où la clé $k_{L,m}$ est transmise au récepteur en même temps que la clé $k_{1,m}$.

[00107] Dans ce mode de réalisation particulier où les durées des cryptopériodes CP_m sont toutes égales à V et où l'on impose que tous les temps moyens $TC_{i,m}$ soient égaux à une constante TC , alors les conditions (1) à (3) sont satisfaites, par exemple, en choisissant une valeur du paramètre PC_p tel que le temps moyen TC soit compris entre $(L-1)V/L$ et V .

[00108] Ici, la valeur initiale du paramètre PC_p est alors choisie en déterminant le plus grand nombre N_R pour lequel la condition suivante est vérifiée : $t_{1-5} \cdot C_a / 2 \leq V$, où

- t_{1-5} est le temps d'exécution par le récepteur des opérations 1) à 5) de l'étape 150,
- C_a est le nombre d'éléments de l'ensemble E_R , et C_a est égal à 2^{NR} , et
- V est la durée de chaque intervalle de validité des clés k_m .

[00109] Par exemple, lors de la phase 114, le temps t_{1-5} est mesuré expérimentalement sur un récepteur. La durée V est fixée et connue.

[00110] La figure 3 représente un procédé identique au procédé de la figure 2 sauf que les étapes 116 et 150 ont été remplacées, respectivement, par des étapes 180 et 182. Pour simplifier la figure 3 et les figures suivantes, seules les étapes modifiées sont représentées. Les étapes non modifiées et donc non représentées sont symbolisées sur ces figures par une ligne en pointillés.

[00111] Les étapes 180, 182 sont identiques, respectivement, aux étapes 116 et 150, sauf que les algorithmes D1 et D2 sont remplacés par, respectivement, des algorithmes D3 et D4.

[00112] L'algorithme D3 est un calcul de clés déterministe et non plus un calcul de clés avec aléa. Un calcul de clés déterministe, contrairement à un calcul de clés avec aléa, ne fait pas intervenir un tirage aléatoire susceptible de modifier de façon substantielle le temps moyen $TC_{i,m}$ d'exécution de l'algorithme D4.

[00113] L'algorithme D3 exécuté pour générer la clé k_m à partir de la clé k_{m-1} consiste à composer Q_m-1 fois une fonction cryptographique à sens unique H_2 avec elle-même. La clé k_m est donc obtenue selon la relation suivante : $k_m = H_2^{Q_m}(k_{m-1})$, où :

- H_2 est une fonction cryptographique à sens unique ;
- $H_2^{Q_m}$ indique que la fonction H_2 est composée avec elle-même Q_m-1 fois.

[00114] Typiquement, la fonction H_2 appartient au groupe G_1 de fonctions précédemment définies. Ici, la fonction H_2 est une fonction de hachage. La composition de la fonction H_2 avec elle-même consiste à appliquer une première fois la fonction H_2 à la clé k_{m-1} pour obtenir un premier résultat $H_2(k_{m-1})$, puis à appliquer
 5 une seconde fois la fonction H_2 au premier résultat $H_2(k_{m-1})$ pour obtenir un deuxième résultat $H_2^2(k_{m-1}) = H_2(H_2(k_{m-1}))$ et ainsi de suite Q_m fois. Dans ce mode de réalisation, les informations de contrôle comportent le paramètre Q_m .

[00115] La valeur du paramètre Q_m varie en fonction de l'indice m . Par exemple, pour chaque indice m supérieur ou égal à deux, la valeur du paramètre Q_m est tirée
 10 aléatoirement dans un ensemble E_Q de valeurs proches de V/t_{182} , où t_{182} est égal au temps qu'il faut au récepteur pour exécuter une fois la fonction H_2 . Par exemple, l'ensemble E_Q est l'ensemble des nombres entiers compris dans l'intervalle $[0,7V/t_{182}; 1,3V/t_{182}]$ ou de l'intervalle $[0,9V/t_{182}; 1,1V/t_{182}]$. Dans ce cas, les informations de contrôle transmises au récepteur 10 comportent chacune des valeurs des paramètres
 15 $Q_{2,m}$ à $Q_{L,m}$. Ainsi, quand le récepteur 10 se connecte au serveur 106 pour obtenir la succession SR_m , il reçoit en réponse :

- la clé $k_{1,m}$,
- les informations de contrôle $Q_{2,m}$ à $Q_{L,m}$.

[00116] L'ensemble E_Q est beaucoup plus petit que l'ensemble E_R . Par exemple,
 20 l'ensemble E_Q contient 10^3 ou 10^6 nombres entiers. La taille de l'ensemble E_Q est ici constante quelle que soit la valeur de l'indice m .

[00117] Le paramètre de complexité PC_p est dans ce mode de réalisation la valeur moyenne M_Q de l'ensemble E_Q et non pas le nombre d'entiers qu'il contient. La valeur moyenne M_Q est la moyenne des nombres entiers contenus dans cet ensemble E_Q en
 25 affectant à chacun de ces nombres entiers le même coefficient de pondération. Plus la valeur moyenne M_Q augmente, plus le temps $TC_{i,m}$ de calcul pour obtenir la clé $k_{i,m}$ à partir de la clé $k_{i-1,m}$ augmente aussi. Lors de la phase 114, l'ensemble E_Q est construit pour que sa valeur moyenne soit égale à V/t_{182} . Dans ce mode de réalisation, la valeur M_Q et donc le paramètre de complexité PC_p n'a pas besoin d'être
 30 transmis aux récepteurs.

[00118] Ensuite, lors des étapes 162 et 164, la composition de l'ensemble E_Q est modifiée pour diminuer sa valeur moyenne lors de l'étape 162 et, en alternance, augmenter sa moyenne lors de l'étape 164. Par exemple, pour augmenter la valeur moyenne de l'ensemble E_Q de ϵ , le nombre entier ϵ est ajouté à chacun des nombres
 35 entiers précédemment contenus dans l'ensemble E_Q .

[00119] Lors de l'étape 182, l'algorithme D4 exécuté par le récepteur pour obtenir la clé $k_{i,m}$ à partir de la clé $k_{i-1,m}$ est identique à l'algorithme D3 sauf que la valeur du paramètre $Q_{i,m}$ est obtenue à partir des informations de contrôle reçues. Autrement dit, lors de l'étape 182, le récepteur 10 obtient la clé $k_{i,m}$ à l'aide de la relation
 40 suivante : $k_{i,m} = H_2^{Q_{i,m}}(k_{i-1,m})$.

[00120] La figure 4 représente un procédé identique au procédé de la figure 3 sauf que les étapes 180 et 182 sont remplacées, respectivement, par des étapes 190 et 192. Les étapes 190 et 192 sont identiques aux étapes 180 et 182 sauf qu'au lieu d'utiliser la même fonction H_2 pour obtenir les clés k_m , le générateur 108 utilise une
 5 fonction H_{G5} et le récepteur 10 utilise de son côté une fonction H_{D6} . Ainsi, le calcul de la clé k_m suivante est réalisé par le générateur 108 à l'aide de la relation suivante : $k_m = H_{G5}^{Q_m}(k_{m-1})$.

[00121] Le processeur 76 calcule la clé $k_{i,m}$ à l'aide de la relation suivante : $k_{i,m} = H_{D6}^{Q_{i,m}}(k_{i-1,m})$. La fonction H_{G5} est conçue pour permettre de calculer la clé k_m à partir
 10 de la clé k_{m-1} de façon beaucoup plus rapide que ce qui est possible en mettant en œuvre la fonction H_{D6} . Pour cela, on utilise des fonctions cryptographiques H_{D6} à sens unique avec portes dérobées, telles que celles utilisées pour réaliser des chiffrements asymétriques. Le principe de fonctionnement de telles fonctions cryptographiques à
 15 sens unique avec portes dérobées est bien connu. Par exemple, il s'agit du même principe que celui utilisé dans des algorithmes de chiffrement asymétrique aussi connus que l'algorithme de chiffrement RSA (Rivest-Shamir-Adleman). Ainsi, par la suite, un seul exemple détaillé d'une telle fonction est décrit, sachant que l'homme du métier peut, à partir de cet exemple, déduire sans difficulté d'autres modes de réalisation possibles.

[00122] Par exemple, ici, le générateur 108 calcule la clé k_m à l'aide de la fonction H_{G5} définie par la relation suivante : $k_m = (k_{m-1}^{(e^{Q_m}[(P-1)(Q-1)])})[N]$, où :

- $(A^B)[C]$ désigne l'exponentiation modulaire, c'est à dire l'exponentiation du nombre A par le nombre B, le tout modulo C,

- P et Q sont des grands nombres premiers, c'est-à-dire des nombres premiers dont
 25 la représentation binaire comporte au moins 500 ou 1 000 bits, en outre distincts l'un de l'autre,

- N est égal au produit du nombre P par le nombre Q,

- e est un nombre premier avec le produit $(P-1)(Q-1)$, supérieur à 1 et non compris entre $a(P-1)(Q-1)$ et $a(P-1)(Q-1) + 2^s$, où « a » est un entier naturel non nul et, s un
 30 entier naturel typiquement égal à 80, ou supérieur, appelé paramètre de sécurité.

[00123] Le processeur 76 calcule la clé $k_{i,m}$ à l'aide de la fonction H_{D6} définie par la relation suivante : $k_{i,m} = ((k_{i-1,m}^e)[N])^{Q_{i,m}}$.

[00124] Les nombres P et Q sont seulement connus du générateur 108 et correspondent par exemple à sa clé privée. Les nombres N et e sont connus du
 35 générateur 108 et du récepteur 10 et correspondent alors à la clé publique du générateur 108. Le générateur 108, grâce à la connaissance des nombres P et Q, peut calculer la clé k_m à partir de la clé k_{m-1} en exécutant seulement deux exponentiations modulaires alors que pour obtenir la même clé, le processeur 76 doit réaliser $Q_{i,m}$ exponentiations modulaires.

[00125] Chapitre IV : Variantes :

[00126] Chapitre IV.1 : Variantes du procédé :

[00127] En variante, les informations de contrôle peuvent être omises. Par exemple, dans le cas d'un calcul déterministe de la clé $k_{i,m}$, si le nombre $Q_{i,m}$ de fois que la fonction H_2 doit être exécutée pour construire la clé $k_{i,m}$ à partir de la clé $k_{i-1,m}$

5 est une constante connue de tous les récepteurs, alors le serveur 106 n'a pas besoin de transmettre cette information de contrôle au récepteur à chaque connexion. Dans ce cas particulier, le paramètre PC_p est aussi une constante. Ainsi, le serveur 106 transmet uniquement la clé $k_{1,m}$ au récepteur et le récepteur dérive les clés $k_{2,m}$ à $k_{L,m}$ à partir de cette clé $k_{1,m}$ sans avoir besoin de recevoir d'autre information de la part du
10 serveur 106.

[00128] Le nombre N de clés de la succession SE_p peut être 10 ou 100 ou 1 000 fois supérieur au nombre L_{max} .

[00129] A un instant quelconque t_{p+1} , le générateur 108 peut arrêter d'utiliser la succession SE_p , y compris avant que la clé k_N ait été utilisée pour chiffrer la
15 cryptopériode CP_N . A partir de l'instant t_{p+1} , le générateur commence à utiliser une autre succession SE_{p+1} de clés. Dans ce cas, de préférence, avant l'instant t_{p+1} , le serveur 106 transmet un signal aux récepteurs pour leur indiquer que l'autre succession SE_{p+1} va être utilisée à partir de cet instant. En réponse, les récepteurs établissent immédiatement une connexion avec le serveur 106 pour recevoir les
20 informations nécessaires pour obtenir la clé $k_{1,m}$ et dériver les L clés suivantes $k_{i,m}$ de cette nouvelle succession SE_{p+1} .

[00130] Le nombre N de clés de la succession SE_p peut aussi être inférieur au nombre de cryptopériodes du contenu multimédia à embrouiller. Dans ce cas, après avoir généré la clé k_N , le générateur 108 commence la génération d'une autre
25 succession SE_{p+1} de clés destinées à embrouiller les cryptopériodes suivant la cryptopériode CP_N , par exemple CP_{N+1} à CP_{2N} ou CP_{N+1} à CP_{N+M} . La première clé k_{N+1} de la succession SE_{p+1} est générée indépendamment des clés de la succession SE_p et, en particulier, ne dépend pas de la clé k_N . Les nombres N et M doivent quand même le cas échéant être choisis supérieurs ou égaux au nombre L_{max} .

[00131] Le nombre N peut aussi être a priori indéterminé. Dans ce cas, le générateur 108 génère en permanence de nouvelles clés k_m pour la succession SE_p . Dans ce cas, de préférence, le générateur 108 génère une nouvelle succession SE_{p+1} en réponse à une commande extérieure de changement de succession de clés. Par exemple, cette commande de changement de succession de clés est émise
35 lorsqu'une attaque ou une tentative d'attaque de la succession SE_p de clés est détectée. La cryptanalyse réussie de l'algorithme de dérivation de clé peut être détectée, par exemple, à partir du fait qu'un nombre croissant de récepteurs n'ont plus besoin de se connecter au serveur 106 toutes les $L.V$ secondes pour être capables de désembrouiller correctement le contenu multimédia.

40 [00132] En variante, lors de l'étape 142, le serveur 106 ne met à jour le compteur

C_{nbc} que si la dernière clé obtenue n'est pas la dernière clé de la succession de clés courante, c'est-à-dire si la connexion en cours n'était pas attendue.

[00133] Dans un mode de réalisation simplifié, le nombre L n'est pas ajusté pour répartir plus uniformément dans le temps les connexions des récepteurs au serveur 106. Dans ce cas, par exemple, le nombre L est une constante par exemple prédéterminée, et est identique pour tous les récepteurs.

[00134] En variante, pour limiter la quantité d'informations transmises par le serveur 106 au récepteur 10 lors de chaque connexion, la valeur du paramètre Q_m est indépendante de l'indice m et ce paramètre Q_m est alors simplement noté Q . Dans ce cas, il est possible de choisir les nombres L et N égaux. Le générateur change alors systématiquement la succession SE_p après avoir généré L clés.

[00135] En variante, l'étape 160 n'est pas systématiquement mise en œuvre après l'étape 146. Par exemple, l'étape 160 est mise en œuvre périodiquement, avec une période prédéterminée, par exemple sous forme de nombre de mises en œuvre de l'étape 146, ou sous forme de durée. Lorsque, après l'étape 146, l'étape 160 n'est pas mise en œuvre, les étapes 162 et 164 sont également omises, et le serveur 106 laisse inchangée la valeur actuelle du paramètre PC_p .

[00136] D'autres méthodes de réglage de la valeur du paramètre PC_p sont possibles. Par exemple, le réglage de la valeur du paramètre PC_p est déclenché autrement qu'en comparant la valeur du compteur C_{nbc} au seuil S_{nbc-h} et S_{nbc-i} . Par exemple, le temps $TC_{i,m}$ d'exécution de l'algorithme de dérivation par le processeur 76 est mesuré, par exemple par le processeur 76, puis transmis au serveur 106. En réponse, le serveur 106 compare ce temps d'exécution mesuré à la durée V . Si le temps d'exécution mesuré est inférieur à la durée V , la valeur du paramètre de complexité PC_p est augmentée. Dans le cas contraire, elle est diminuée.

[00137] Dans un autre exemple, en réponse à la réception du temps $TC_{i,m}$ mesuré par le processeur 76, le serveur 106 calcule un temps moyen d'exécution de l'algorithme de dérivation, puis compare ce temps d'exécution moyen à la durée V . Si le temps d'exécution moyen est inférieur à la durée V , la valeur du paramètre de complexité PC_p est augmentée. Dans le cas contraire, elle est diminuée. Dans cet exemple, le serveur 106 calcule un temps moyen d'exécution de l'algorithme de dérivation de l'ensemble des récepteurs ou d'un échantillon des récepteurs, par exemple d'effectif prédéterminé, par exemple aléatoirement constitué. Dans cet exemple, alternativement ou simultanément, le serveur 106 calcule un temps moyen d'exécution de l'algorithme de dérivation sur une fenêtre temporelle glissante d'amplitude prédéterminée, par exemple de l'ordre de grandeur d'une minute, de dix minutes, d'une heure ou de dix heures.

[00138] En variante, la complexité de l'algorithme de dérivation n'est pas réglable. Dans ce cas, la valeur du paramètre PC_p est choisie lors, par exemple, de la phase 114 comme précédemment décrit. Ensuite, le paramètre PC_p ne peut plus être

modifié au cours de l'exécution du procédé de la figure 2. Dans ce mode de réalisation, les étapes 142, 160, 162 et 164 sont omises.

[00139] Le paramètre PC_p peut être réglé pour que le temps moyen $TC_{i,m}$ soit supérieur à la durée V . Dans ce cas, dans le contexte du mode de réalisation de la figure 2, cela entraîne que seul un sous-groupe PP, d'effectif inférieur à celui obtenu avec les réglages décrits plus haut du paramètre PC_p , des récepteurs aura tiré aléatoirement le nombre R qui permet d'obtenir la clé $k_{i,m}$ à temps, c'est-à-dire avant l'instant $t_{i,m}$. Les récepteurs qui n'appartiennent pas à ce sous-groupe PP devront se connecter au serveur 106 pour obtenir la clé $k_{i,m}$. Toutefois, ce réglage permet quand même de réduire le nombre des récepteurs qui se connectent à chaque cryptopériode au serveur 106, tout en augmentant la sécurité du système.

[00140] En variante, la connexion entre le serveur 106 et le récepteur 10 n'est pas interrompue après la transmission par le serveur des informations nécessaires pour construire la succession SR_m .

[00141] Chapitres IV.2 : Variantes des algorithmes de dérivation de clé

[00142] D'autres fonctions que celles précédemment décrites peuvent être utilisées. Par exemple, la fonction F_1 n'est pas nécessairement une fonction à sens unique. Elle peut être une fonction aussi simple que la fonction identité. Toutefois, dans ce cas, la fonction H_1 est différente de la fonction F_1 et reste une fonction à sens unique.

[00143] D'autres calculs déterministes de clés sont possibles. Par exemple, l'algorithme D3 peut être remplacé par un autre algorithme dans lequel le calcul de la clé k_m est réalisé à l'aide de la relation suivante : $k_m = H_3^Q(f(k_{m-1}, D_m))$ où :

- D_m est un petit nombre tiré aléatoirement dans un ensemble E_D contenant les nombres entiers dont la représentation binaire est au plus de 18 bits ou 10 bits,
- f est une fonction simple telle que l'addition ou la multiplication entre k_{m-1} et D_m , et
- H_3 est une fonction appartenant au groupe G_1 précédemment défini et de préférence une fonction de hachage, et
- Q est le paramètre de complexité.

[00144] Dans ce mode de réalisation, les informations de contrôle contiennent les données D_m et le paramètre Q . Les données D_m sont codées sur un nombre de bits de préférence au moins deux fois inférieur au nombre de bits nécessaires pour coder les paramètres Q_m définis dans le mode de réalisation de la figure 3. Par conséquent, la bande passante requise pour transmettre les informations de contrôle au récepteur 10 est réduite dans cette variante.

[00145] D'autres modes de réalisation de l'algorithme de dérivation sont possibles. Par exemple, au lieu de tirer aléatoirement chaque fois le nombre R dans l'ensemble E_R , le nombre R est initialisé à 0 puis incrémenté de 1 à chaque itération des opérations 1) à 5) de l'étape 150.

[00146] Chapitre IV.3 : Autres variantes :

[00147] Les procédés d'obtention d'une succession de clés ont été décrits ici dans le cas particulier où les clés générées et reçues sont des mots de contrôle utilisés directement pour chiffrer et déchiffrer des contenus multimédias. Toutefois, ces procédés peuvent être utilisés pour obtenir d'autres clés que des mots de contrôle.

5 Par exemple, les clés générées et reçues peuvent être des clés de session utilisées pour chiffrer et déchiffrer des mots de contrôle transmis au récepteur. Dans ce cas, les clés de session sont changées, en général, avec une fréquence 10, 100, 1000 ou 10000 fois inférieure à la fréquence de changement des mots de contrôle. Par exemple, la durée de l'intervalle de validité d'une clé de session est supérieure à une
10 minute, une heure, ou 24 heures. Dans ce cas, la valeur du paramètre de complexité PC_p est adaptée pour correspondre à une telle durée de l'intervalle de validité.

[00148] Les procédés d'obtention de succession de clés décrits ici peuvent aussi être utilisés pour obtenir des successions de clés qui sont utilisées pour chiffrer et déchiffrer des contenus numériques autres que des contenus multimédias. Par
15 exemple, la succession de clés obtenues peut être utilisée pour chiffrer ou déchiffrer des documents numériques tels que des fichiers de texte, ou toutes données échangées sur un canal de communication.

[00149] Ce qui a été décrit ici s'applique aussi à d'autres systèmes que des systèmes d'accès conditionnels. Par exemple, ce qui a été décrit ici s'applique à tout
20 système où une succession de clés, associées chacune à un intervalle de validité, est utilisée. Par exemple, l'enseignement donné ici se transpose sans difficulté particulière aux systèmes de gestion de droits numériques connu sous l'acronyme DRM (« Digital Rights Management »). Dans ces systèmes de DRM, chaque clé de la succession est obtenue à partir d'une licence. Cette licence contient généralement
25 l'intervalle de validité de la clé.

[00150] Ce qui vient d'être décrit dans le cas particulier où les clés $k_{i,m}$ sont des clés de déchiffrement peut aussi être appliqué au cas où les clés $k_{i,m}$ sont utilisées pour chiffrer un contenu numérique et non pas pour déchiffrer ce contenu numérique. De façon plus générale, ce qui a été décrit ici peut aussi être utilisé pour obtenir une
30 succession de clés cryptographiques utilisées à d'autres fins, telles que la vérification d'authenticité, la vérification d'intégrité de données numériques, l'initialisation d'un générateur pseudo-aléatoire ou autres.

[00151] En variante, la durée $V_{i,m}$ des intervalles de validité n'est pas constante. Dans ce cas, le temps $TC_{i,m}$ est par exemple différent pour obtenir chaque clé $k_{i,m}$. Le
35 temps $TC_{i,m}$ est réglé à l'aide, par exemple, de l'information de contrôle Q_m . Dans ce cas, typiquement, l'information de contrôle Q_m est choisie pour que le temps $TC_{i,m}$ soit compris entre $0,5V_{i-1,m}$ et $V_{i-1,m}$ et de préférence entre $0,9V_{i-1,m}$ et $V_{i-1,m}$. De préférence, l'information de contrôle Q_m est choisie pour que le temps d'exécution soit tel que le temps $TC_{i,m}$, quel que soit i appartenant à l'intervalle $[2;L]$, vérifie les conditions (1) à
40 (3) précédemment décrites. Lorsque ces conditions sont satisfaites, la clé $k_{i,m}$ peut au

plus tôt être obtenue par le récepteur dans l'intervalle $[t_{i-1,m}; t_{i,m}[$. Ainsi, la clé $k_{i,m}$ ne peut faire l'objet d'attaque que pendant l'intervalle $[t_{i-1,m}; t_{i,m}[$ de durée $V_{i-1,m}$.

[00152] En variante, la condition (2) précédente est omise.

[00153] Dans une autre variante, l'exécution de l'algorithme de dérivation est répartie sur un groupe de M récepteurs capables d'échanger entre eux des informations de façon sécurisée. Le nombre M est supérieur ou égal à deux et, de préférence, supérieur ou égal à 100 ou 1 000. Par exemple, les récepteurs d'un même groupe de récepteurs sont raccordés les uns aux autres par l'intermédiaire d'un réseau public ou privé. Un récepteur qui appartient à un groupe de récepteurs n'appartient pas à un autre groupe de récepteurs. Les échanges d'informations entre les récepteurs du groupe sont par exemple chiffrés à l'aide de clés connues seulement des récepteurs de ce groupe. Par exemple, dans le cas où l'algorithme de dérivation est un calcul de clé avec aléa, l'ensemble E_R est divisé en M sous-ensembles disjoints et de même taille. Chacun de ces sous-ensembles est affecté à un récepteur respectif du groupe. En réponse à la réception de la clé $k_{1,m}$, chaque récepteur du groupe essaie de calculer la clé $k_{2,m}$ comme décrit précédemment mais en choisissant le nombre R uniquement dans le sous-ensemble qui lui a été affecté. Le premier récepteur du groupe qui obtient la clé $k_{2,m}$ la transmet alors aux autres récepteurs du même groupe. Une fois que la clé $k_{2,m}$ a été distribuée à tous les récepteurs du même groupe, ces récepteurs arrêtent la précédente exécution de l'algorithme de dérivation pour obtenir la clé $k_{2,m}$ et commencent à exécuter l'algorithme de dérivation pour obtenir la clé $k_{3,m}$ de façon similaire à ce qui a été décrit pour la clé $k_{2,m}$.

[00154] Le groupe de récepteurs exécute ainsi l'algorithme de dérivation en un temps $TC_{i,m}$ égal à $TCr_{i,m}/M$, où $TCr_{i,m}$ est le temps moyen d'exécution du même algorithme de dérivation mais par un seul récepteur du groupe qui doit donc choisir le nombre R dans la totalité de l'ensemble E_R et non pas seulement dans un sous-ensemble. Cette variante permet donc d'augmenter la taille de l'ensemble E_R et donc d'augmenter la sécurité du procédé.

[00155] Dans le cas où l'algorithme de dérivation est un calcul déterministe de clé, il est aussi possible de répartir l'exécution de cet algorithme de dérivation sur chacun des récepteurs du groupe pour bénéficier de la puissance de calcul de l'ensemble des processeurs 76 de ces récepteurs. La répartition de l'exécution d'un algorithme entre différents microprocesseurs pour que des portions de cet algorithme soient exécutées en parallèle par chacun de ces microprocesseurs est bien connue et ne sera pas décrite ici en détail. Comme dans le cas d'un algorithme de dérivation de clé avec aléa, répartir l'exécution de l'algorithme de dérivation sur tous les récepteurs du groupe permet d'augmenter la sécurité car cela complexifie l'obtention de la clé. En particulier, cela augmente la puissance de calcul requise pour pouvoir exécuter suffisamment rapidement l'algorithme de dérivation par un récepteur illégal.

[00156] Si la durée TA_1 nécessaire à la récupération et à la distribution illégale d'une clé $k_{i,m}$ qui vient d'être obtenue par le récepteur 10 est connue, alors le paramètre de complexité PC_p peut être ajusté pour que le temps $TC_{i,m}$ soit seulement compris dans l'intervalle $[V-TA_1;V]$.

- 5 [00157] Si la durée TA_2 nécessaire à la récupération et la distribution illégale de la clé $k_{1,m}$ reçue par le récepteur 10 est connue et que l'algorithme de dérivation est connu d'un récepteur illégal, alors le paramètre de complexité peut être ajusté pour que le temps $TC_{i,m}$ vérifie seulement la condition suivante : $TA_2 + (L-1)TC_{i,m} \geq (L-1)V$.

[00158] Chapitre IV : Avantages des modes de réalisation décrits :

- 10 [00159] Dans les procédés décrits, le récepteur ne peut obtenir la clé $k_{i,m}$ qu'après avoir terminé le calcul de la clé $k_{i-1,m}$. Ainsi, le récepteur est obligé de calculer les clés $k_{i,m}$ dans l'ordre croissant des indices i . De plus, le temps moyen $TC_{i,m}$ de calcul d'une clé $k_{i,m}$ est long, c'est-à-dire ici supérieur ou égal à $0,2V_{i-1,m}$. Dès lors, bien que toutes les informations nécessaires pour obtenir les clés $k_{1,m}$ à $k_{L,m}$ soient reçues avant
15 l'instant $t_{1,m}$, les clés suivantes $k_{2,m}$ à $k_{L,m}$ ne peuvent être obtenues que bien après cet instant $t_{1,m}$. Par exemple, la clé $k_{L,m}$ est obtenue qu'après l'instant $t_R + TC_{2,m} + TC_{3,m} + \dots + TC_{L,m}$, où t_R est l'instant de réception des informations nécessaires pour obtenir chacune des clés $k_{1,m}$ à $k_{L,m}$. Par comparaison, dans les procédés connus comme celui décrit dans la demande EP2567500, la clé $k_{L,m}$ est obtenue à l'instant t_R .
20 Autrement dit, les procédés décrits ici retardent de $TC_{1,m} + \dots + TC_{L,m}$ l'obtention de la clé $k_{L,m}$. Puisque la clé $k_{L,m}$ est obtenue dans les récepteurs plus tard que dans les récepteurs connus, le temps disponible pour attaquer cette clé $k_{L,m}$ avant l'instant $t_{L,m}$ est plus court que dans les procédés connus, ce qui accroît la sécurité du procédé.

- [00160] Par ailleurs, comme dans le procédé EP2567500, pour obtenir les L clés de
25 la succession SR_m , le récepteur n'a besoin de se connecter qu'une seule fois au serveur de clé 106. Ainsi, le procédé permet toujours de réduire le nombre de connexions ou d'échanges d'informations entre le serveur 106 et chacun des récepteurs. Enfin, le procédé décrit peut être appliqué sans avoir à déterminer un niveau de sécurité associé à chacun des récepteurs.

- 30 [00161] Le fait d'adapter dynamiquement la valeur du paramètre PC_p en fonction de l'information selon laquelle la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$, permet de s'adapter automatiquement et au fil de l'eau aux performances réelles des récepteurs.

- [00162] Le fait d'utiliser différentes valeurs du nombre L pour différents récepteurs
35 ou groupes de récepteurs permet donc de mieux répartir dans le temps les différentes connexions au serveur 106.

- [00163] Le fait de répartir l'exécution de l'algorithme de dérivation sur plusieurs récepteurs permet d'augmenter la sécurité du procédé vis-à-vis des attaques par collusion. En effet, pour obtenir plus rapidement la succession SR_m , un attaquant peut
40 essayer de répartir l'exécution de l'algorithme de dérivation sur plusieurs récepteurs

illégaux. Or, le nombre de récepteurs légaux est généralement très supérieur au nombre de récepteurs illégaux. Cela permet donc d'augmenter la valeur du paramètre PC_p de façon à tenir compte de la détection d'une attaque par collusion. Il devient alors plus difficile de mettre en œuvre avec succès une telle attaque par collusion.

- 5 [00164] Le fait que le récepteur conserve la capacité d'établir une deuxième connexion avec le serveur 106 pour obtenir la clé $k_{i,m}$ s'il n'a pas réussi à obtenir cette clé à temps à partir des informations reçues dans la première connexion, permet de mettre en œuvre le procédé revendiqué avec un parc de récepteurs dans lequel certains récepteurs sont plus lents que d'autres ou que prévu pour exécuter
- 10 l'algorithme de dérivation. Cela permet le cas échéant de prendre en compte la diversité des récepteurs et la fluctuation de leurs performances de calcul.

- [00165] Le fait que l'algorithme de dérivation comporte la réitération $Q_{i,m}$ fois de la même fonction à sens unique permet d'avoir un temps moyen $TC_{i,m}$ qui peut être déterminé à l'avance et donc qui, plus sûrement, peut être systématiquement compris
- 15 dans l'intervalle $[V_{i-1,m}/2; V_{i-1,m}[$.

[00166] Le fait que la fonction à sens unique soit une fonction cryptographique à sens unique avec porte dérobée permet de diminuer le temps nécessaire pour générer la succession SE_p .

- [00167] Le fait que l'algorithme de dérivation de clé soit un calcul de clés avec aléa permet au générateur 108 d'exécuter beaucoup moins d'opérations pour calculer une
- 20 clé $k_{i,m}$ que ne doit en faire le récepteur pour calculer la même clé.

- [00168] Le fait de mettre en œuvre un algorithme de dérivation qui calcule la clé $k_{i,m}$ avec un aléa permet donc d'obtenir un temps d'exécution de cet algorithme de dérivation important du côté du récepteur et beaucoup plus faible du côté du
- 25 générateur.

[00169] Le fait de recevoir, lors de la première connexion, seulement les informations nécessaires pour obtenir une succession SR_m de L clés permet d'obliger le récepteur 10 à établir une nouvelle connexion avec le serveur 106 au plus tard après l'obtention de la clé $k_{L,m}$.

REVENDICATIONS

1. Procédé d'obtention d'une succession de L clés cryptographiques $k_{1,m}, \dots, k_{i,m}, k_{i+1,m}, \dots, k_{L,m}$ par un groupe de récepteurs électroniques, où :
 - 5 - l'indice i est un numéro d'ordre de la clé $k_{i,m}$ dans la succession de clés,
 - L est un nombre entier supérieur ou égal à deux, et
 - quel que soit l'indice i compris entre 1 et L, la clé $k_{i,m}$ est uniquement destinée à être utilisée pendant un intervalle de validité $[t_{i,m}, t_{i+1,m}[$ de durée $V_{i,m}$, où $t_{i,m}$ et $t_{i+1,m}$ sont, respectivement, les instants de début et de fin de cet intervalle de validité,
 - 10 dans lequel :
 - avant l'instant $t_{1,m}$, le groupe de récepteurs établit (140) une première connexion avec un serveur de clés et reçoit, lors de cette première connexion, les informations nécessaires pour obtenir la clé $k_{1,m}$, puis
 - pour chaque indice i compris entre 2 et L, avant l'instant $t_{i,m}$, le groupe de récepteurs
 - 15 obtient (150) la clé $k_{i,m}$, caractérisé en ce que pour chaque indice i compris entre 2 et L et dans l'ordre croissant de ces indices :
 - le groupe de récepteurs obtient (150 ; 182 ; 192) la clé suivante $k_{i,m}$ en exécutant un algorithme de dérivation de clé initialisé à l'aide de la clé précédente $k_{i-1,m}$ et sans
 - 20 avoir recours à d'autres informations que celles reçues lors de la première connexion, et
 - le temps moyen $TC_{i,m}$ d'exécution de l'algorithme de dérivation de clé par le groupe de récepteurs pour obtenir la clé $k_{i,m}$ est supérieur à $0,2V_{i-1,m}$.
- 25 2. Procédé selon la revendication 1, dans lequel pour au moins un indice i compris entre 2 et L :
 - après l'instant $t_{i-1,m}$ et avant l'instant $t_{i,m}$, le groupe de récepteurs vérifie (126) si l'obtention de la clé $k_{i,m}$ est terminée,
 - si l'obtention de la clé $k_{i,m}$ n'est pas terminée avant l'instant $t_{i,m}$, le groupe de
 - 30 récepteurs établit (140) une deuxième connexion avec le serveur de clés et lui transmet, par l'intermédiaire de cette deuxième connexion, l'information selon laquelle la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$ et, dans le cas contraire, le groupe de récepteurs ne transmet pas au serveur de clés l'information selon laquelle la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$.
 - 35
3. Procédé selon la revendication 2, dans lequel :
 - en réponse à la réception de l'information selon laquelle la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$, le serveur de clés modifie (162) la valeur d'un paramètre PC_p de complexité de l'algorithme de dérivation de clé, la valeur modifiée du
 - 40 paramètre PC_p correspondant à un temps moyen $TC_{i,m}$ d'exécution inférieur au temps

moyen d'exécution actuel, ou

- en réponse à une absence de réception de l'information selon laquelle la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$, le serveur de clé modifie (164) la valeur du paramètre PC_p pour qu'il corresponde à un temps moyen $TC_{i,m}$ d'exécution supérieur
5 au temps moyen d'exécution actuel.

4. Procédé selon la revendication 3, dans lequel, en réponse à la réception de l'information selon laquelle la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$:

- le serveur de clés met à jour (142) un compteur de nombre de connexions établies
10 par unité de temps, puis

- le serveur de clés compare (160) la valeur de ce compteur à un seuil haut prédéterminé, puis, si la valeur de ce compteur a franchi ce seuil haut prédéterminé, en réponse, le serveur de clés modifie (162) la valeur du paramètre de complexité pour diminuer le nombre de connexions établies par unité de temps, et/ou

15 - le serveur de clés compare (142) la valeur de ce compteur à un seuil bas prédéterminé, puis, si la valeur de ce compteur a franchi ce seuil bas prédéterminé, en réponse, le serveur de clés modifie (164) la valeur du paramètre de complexité pour augmenter le nombre de connexions établies par unité de temps.

20 5. Procédé selon l'une quelconque des revendications précédentes, dans lequel le procédé d'obtention d'une succession de clés est mis en œuvre pour un premier et un second groupes de récepteurs, et le serveur de clés utilise (144) une première valeur de l'indice L pour le premier groupe et une seconde valeur différente de l'indice L pour le second groupe de récepteurs.

25 6. Procédé selon l'une quelconque des revendications précédentes, dans lequel le groupe de récepteurs comporte plusieurs récepteurs et l'exécution de l'algorithme de dérivation est répartie sur chacun des récepteurs de ce groupe.

30 7. Procédé selon l'une quelconque des revendications 1 à 5, dans lequel le groupe de récepteurs comporte un seul récepteur et l'algorithme de dérivation est exécuté par ce seul récepteur.

8. Procédé selon l'une quelconque des revendications précédentes, dans
35 lequel pour au moins un indice i compris entre 2 et L :

- après l'instant $t_{i-1,m}$ et avant l'instant $t_{i,m}$, le groupe de récepteurs vérifie (126) si l'obtention de la clé $k_{i,m}$ est terminée,

- si l'obtention de la clé $k_{i,m}$ n'est pas terminée :

- le groupe de récepteurs établit (140) une deuxième connexion avec le serveur
40 de clés et reçoit, lors de cette deuxième connexion, les informations

nécessaires pour obtenir la clé $k_{i,m}$ sans avoir à exécuter l'algorithme de dérivation de clé, puis

- obtient la clé $k_{i,m}$ à partir des informations nécessaires reçues lors de la deuxième connexion et sans exécuter l'algorithme de dérivation de clé, puis
- 5 • le groupe de récepteurs obtient (150 ; 182 ; 192) la clé suivante $k_{i+1,m}$ en exécutant l'algorithme de dérivation de clé initialisé à l'aide de la clé précédente $k_{i,m}$ obtenue à l'aide des informations reçues lors de la deuxième connexion et sans avoir recours à d'autres informations que celles reçues lors de la deuxième connexion, et
- 10 - si l'obtention de la clé $k_{i,m}$ est terminée, le groupe de récepteurs n'établit pas cette deuxième connexion et obtient (150 ; 182 ; 192) la clé suivante $k_{i+1,m}$ en exécutant l'algorithme de dérivation de clé initialisé à l'aide de la clé précédente $k_{i,m}$ et sans avoir recours à d'autres informations que celles reçues lors de la première connexion.
- 15 9. Procédé selon l'une quelconque des revendications précédentes, dans lequel le groupe de récepteurs réalise les opérations suivantes lorsqu'il exécute l'algorithme de dérivation de clé :
- 1) l'initialisation de la valeur d'une variable à partir de la clé précédente $k_{i-1,m}$, puis
 - 2) la transformation de la valeur de cette variable à l'aide d'une fonction à sens
- 20 unique pour obtenir une nouvelle valeur de cette variable, puis
- 3) l'itération de l'opération 2) un nombre $Q_{i,m}$ prédéterminé de fois, en prenant comme valeur de la variable à transformer la nouvelle valeur de la variable obtenue à l'issue de la précédente itération de l'opération 2).
- 25 10. Procédé selon la revendication 9, dans lequel la fonction à sens unique est une fonction cryptographique à sens unique avec porte dérobée.
11. Procédé selon l'une quelconque des revendications 1 à 8, dans lequel le
- groupe de récepteurs réalise les opérations suivantes lorsqu'il exécute l'algorithme de
- 30 dérivation de clé :
- 1) le choix d'un nombre R dans un ensemble E_R , puis
 - 2) le calcul d'une clé candidate k_{cd} à l'aide de la relation $k_{cd} = F(R // k_{i-1,m})$, où F est une fonction prédéterminée et le symbole « // » désigne une opération de combinaison des nombres R et $k_{i-1,m}$, puis
- 35 3) le calcul d'une information de contrôle C_{cd} à l'aide de la relation $C_{cd} = H(k_{cd})$, où H est une fonction à sens unique, puis
- 4) la comparaison de l'information de contrôle C_{cd} à une information de contrôle $C_{i,m}$ reçue lors de la première connexion, et
 - 5) si les informations de contrôle C_{cd} et $C_{i,m}$ correspondent, alors la clé k_{cd} calculée est
- 40 égale à la clé $k_{i,m}$ qui est donc obtenue, sinon, le groupe de récepteurs réitère les

opérations 1) à 5) pour un autre nombre R de l'ensemble E_R .

12. Procédé selon l'une quelconque des revendications précédentes, dans lequel le temps moyen $TC_{i,m}$ d'exécution de l'algorithme de dérivation de clé par le
5 groupe de récepteurs pour obtenir la clé $k_{i,m}$ est inférieur ou égal à V_{i-1} .

13. Procédé de transmission sécurisé d'un contenu numérique, dans lequel :
- un dispositif divise (120) le contenu numérique en cryptopériodes successives, chiffre chaque cryptopériode CP_m avec une clé k_m respective d'une succession
10 ordonnée de clés k_1 à k_N , et transmet chaque cryptopériode chiffrée à un groupe de récepteurs,
- le groupe de récepteurs déchiffre (130) chaque cryptopériode chiffrée reçue à l'aide de la clé k_m correspondante de la succession ordonnée de clés,
caractérisé en ce que le groupe de récepteurs obtient (150 ; 182 ; 192) la succession
15 de clés en mettant en œuvre un procédé conforme à l'une quelconque des revendications précédentes.

14. Support d'enregistrement d'informations, lisible par un microprocesseur, caractérisé en ce qu'il comporte des instructions pour la mise en œuvre d'un procédé
20 conforme à l'une quelconque des revendications précédentes, lorsque ces instructions sont exécutées par ce microprocesseur.

15. Groupe de récepteurs pour la mise en œuvre d'un procédé conforme à l'une quelconque des revendications 1 à 13, chaque récepteur de ce groupe
25 comportant un microprocesseur (77) programmé pour réaliser les étapes suivantes :
- avant l'instant $t_{1,m}$, le groupe de récepteurs établit une première connexion avec un serveur de clés et reçoit, lors de cette première connexion, les informations nécessaires pour obtenir la clé $k_{1,m}$, puis
- pour chaque indice i compris entre 2 et L , avant l'instant $t_{i,m}$, le groupe de récepteurs
30 obtient la clé $k_{i,m}$,
caractérisé en ce que chaque microprocesseur est également programmé pour réaliser les étapes suivantes pour chaque indice i compris entre 2 et L et dans l'ordre croissant de ces indices :
- le groupe de récepteurs obtient la clé suivante $k_{i,m}$ en exécutant un algorithme de
35 dérivation de clé initialisé à l'aide de la clé précédente $k_{i-1,m}$ et sans avoir recours à d'autres informations que celles reçues lors de la première connexion, et
- le temps moyen $TC_{i,m}$ d'exécution de l'algorithme de dérivation de clé par le groupe de récepteurs pour obtenir la clé $k_{i,m}$ est supérieur à $0,2V_{i-1,m}$.

40 16. Serveur de clés pour la mise en œuvre d'un procédé conforme à l'une

quelconque des revendications 3 à 4, ce serveur comportant un microprocesseur (114) programmé pour réaliser les étapes suivantes :

- avant l'instant $t_{1,m}$, le serveur de clés établit une première connexion avec le groupe de récepteurs et transmet, lors de cette première connexion, les informations

5 nécessaires pour obtenir la clé $k_{1,m}$,

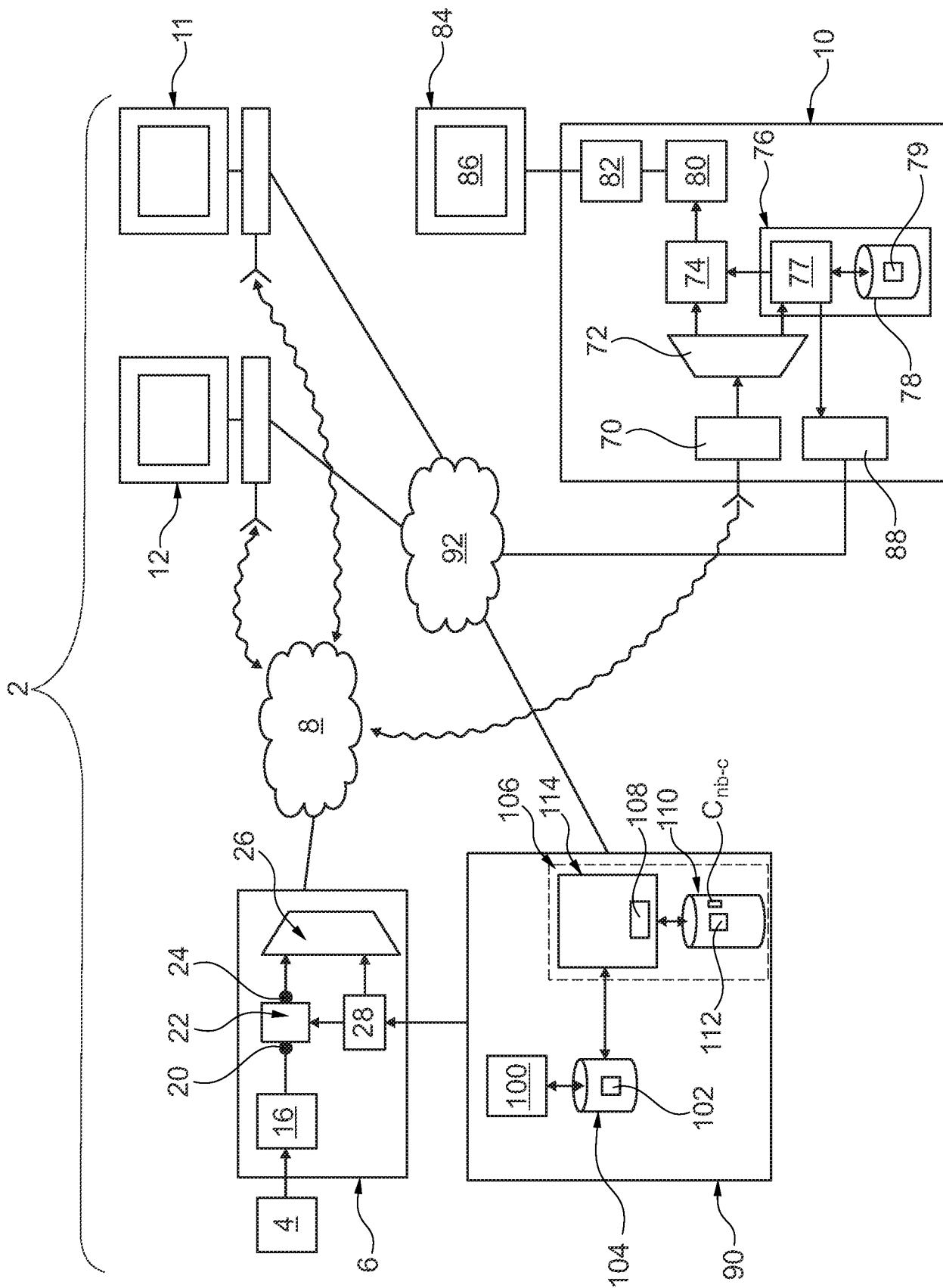
caractérisé en ce que le microprocesseur (114) est également programmé pour réaliser les étapes suivantes :

- en réponse à la réception de l'information selon laquelle la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$, le serveur de clés modifie la valeur d'un paramètre PC_p de

10 complexité de l'algorithme de dérivation de clé, la valeur modifiée du paramètre PC_p correspondant à un temps moyen $TC_{i,m}$ d'exécution inférieur au temps moyen d'exécution actuel, ou

- en réponse à une absence de réception de l'information selon laquelle la clé $k_{i,m}$ n'a pas pu être obtenue avant l'instant $t_{i,m}$, le serveur de clé modifie (164) la valeur du

15 paramètre PC_p pour qu'il corresponde à un temps moyen $TC_{i,m}$ d'exécution supérieur au temps moyen d'exécution actuel.



A 4x4 grid of 16 small squares, each containing a different geometric pattern of dots. The patterns are as follows:

- Row 1: A horizontal line of 4 dots; A single dot in the top-left corner; A 2x2 square of dots; A single dot in the bottom-right corner.
- Row 2: A single dot in the top-left corner; A horizontal line of 4 dots; A 2x2 square of dots; A single dot in the bottom-right corner.
- Row 3: A horizontal line of 4 dots; A single dot in the top-left corner; A 2x2 square of dots; A single dot in the bottom-right corner.
- Row 4: A horizontal line of 4 dots; A single dot in the top-left corner; A 2x2 square of dots; A single dot in the bottom-right corner.

2/2

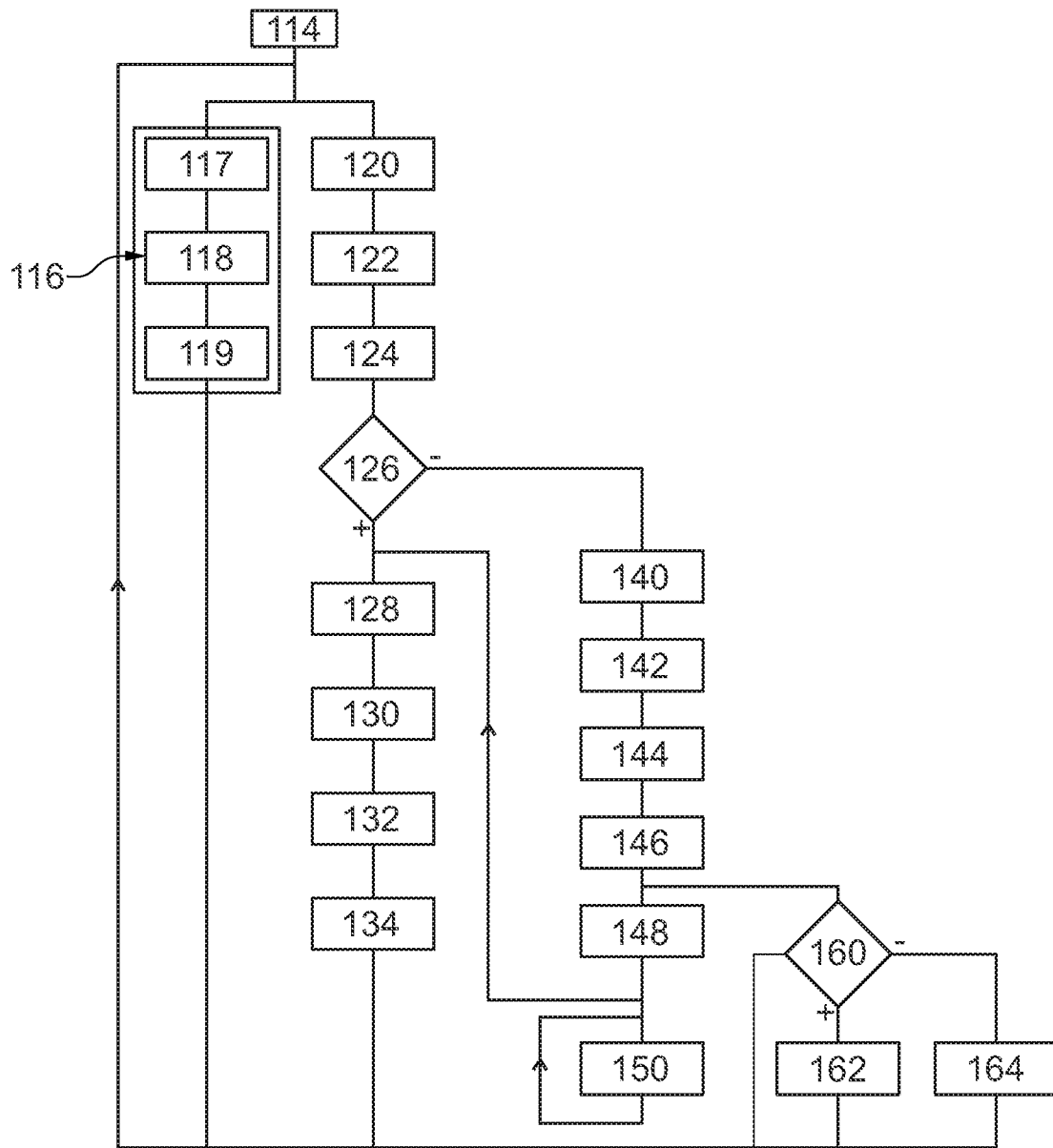


Fig. 2

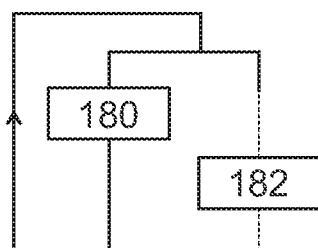


Fig. 3

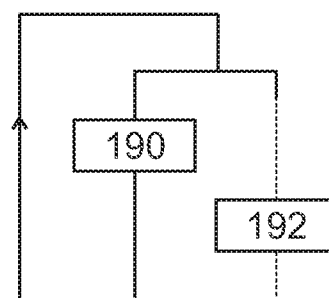


Fig. 4

INTERNATIONAL SEARCH REPORT

International application No.

PCT/FR2019/051616

A. CLASSIFICATION OF SUBJECT MATTER**H04L 9/08**(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 2460308 A1 (ERICSSON TELEFON AB L M [SE]) 06 June 2012 (2012-06-06) paragraphs [0022] - [0028]	1,2,5-10,12-15
T	BIMING TIAN ET AL. "An Efficient Self-Healing Key Distribution Scheme" <i>NEW TECHNOLOGIES, MOBILITY AND SECURITY, 2008. NTMS '08, IEEE, PISCATAWAY, NJ, USA</i> , 05 November 2008 (2008-11-05), pages 1-5 XP031366537 section V;abstract	
A	EP 2567500 A1 (VIACCESS SA [FR]) 13 March 2013 (2013-03-13) cited in the application pages 8-13	1-16

☐ Further documents are listed in the continuation of Box C.☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

09 September 2019

Date of mailing of the international search report

17 September 2019

Name and mailing address of the ISA/EP

European Patent Office
p.b. 5818, Patentlaan 2, 2280 HV Rijswijk
Netherlands

Telephone No. (+31-70)340-2040

Facsimile No. (+31-70)340-3016

Authorized officer

Billet, Olivier

Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/FR2019/051616

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
EP	2460308	A1	06 June 2012	EP	2460308	A1	06 June 2012
				US	2011026715	A1	03 February 2011
				WO	2011013028	A1	03 February 2011
EP	2567500	A1	13 March 2013	EP	2567500	A1	13 March 2013
				ES	2627916	T3	01 August 2017
				FR	2959905	A1	11 November 2011
				PL	2567500	T3	29 September 2017
				RU	2012151859	A	10 June 2014
				TW	201203990	A	16 January 2012
				US	2013046969	A1	21 February 2013
				WO	2011138333	A1	10 November 2011

RAPPORT DE RECHERCHE INTERNATIONALE

Demande internationale n°

PCT/FR2019/051616

A. CLASSEMENT DE L'OBJET DE LA DEMANDE INV. H04L9/08 ADD.		
Selon la classification internationale des brevets (CIB) ou à la fois selon la classification nationale et la CIB		
B. DOMAINES SUR LESQUELS LA RECHERCHE A PORTE Documentation minimale consultée (système de classification suivi des symboles de classement) H04L		
Documentation consultée autre que la documentation minimale dans la mesure où ces documents relèvent des domaines sur lesquels a porté la recherche		
Base de données électronique consultée au cours de la recherche internationale (nom de la base de données, et si cela est réalisable, termes de recherche utilisés) EPO-Internal, WPI Data		
C. DOCUMENTS CONSIDERES COMME PERTINENTS		
Catégorie*	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
X	EP 2 460 308 A1 (ERICSSON TELEFON AB L M [SE]) 6 juin 2012 (2012-06-06) alinéas [0022] - [0028]	1,2, 5-10, 12-15
T	BIMING TIAN ET AL: "An Efficient Self-Healing Key Distribution Scheme", NEW TECHNOLOGIES, MOBILITY AND SECURITY, 2008. NTMS '08, IEEE, PISCATAWAY, NJ, USA, 5 novembre 2008 (2008-11-05), pages 1-5, XP031366537, section V; abrégé	
A	EP 2 567 500 A1 (VIACCESS SA [FR]) 13 mars 2013 (2013-03-13) cité dans la demande pages 8-13	1-16
☐ Voir la suite du cadre C pour la fin de la liste des documents ☒ Les documents de familles de brevets sont indiqués en annexe		
* Catégories spéciales de documents cités: "A" document définissant l'état général de la technique, non considéré comme particulièrement pertinent "E" document antérieur, mais publié à la date de dépôt international ou après cette date "L" document pouvant jeter un doute sur une revendication de priorité ou cité pour déterminer la date de publication d'une autre citation ou pour une raison spéciale (telle qu'indiquée) "O" document se référant à une divulgation orale, à un usage, à une exposition ou tous autres moyens "P" document publié avant la date de dépôt international, mais postérieurement à la date de priorité revendiquée "T" document ultérieur publié après la date de dépôt international ou la date de priorité et n'appartenant pas à l'état de la technique pertinent, mais cité pour comprendre le principe ou la théorie constituant la base de l'invention "X" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme nouvelle ou comme impliquant une activité inventive par rapport au document considéré isolément "Y" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme impliquant une activité inventive lorsque le document est associé à un ou plusieurs autres documents de même nature, cette combinaison étant évidente pour une personne du métier "&" document qui fait partie de la même famille de brevets		
Date à laquelle la recherche internationale a été effectivement achevée 9 septembre 2019		Date d'expédition du présent rapport de recherche internationale 17/09/2019
Nom et adresse postale de l'administration chargée de la recherche internationale Office Européen des Brevets, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016		Fonctionnaire autorisé Billet, Olivier

RAPPORT DE RECHERCHE INTERNATIONALE

Renseignements relatifs aux membres de familles de brevets

Demande internationale n°

PCT/FR2019/051616

Document brevet cité au rapport de recherche		Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
EP 2460308	A1	06-06-2012	EP 2460308 A1	06-06-2012
			US 2011026715 A1	03-02-2011
			WO 2011013028 A1	03-02-2011

EP 2567500	A1	13-03-2013	EP 2567500 A1	13-03-2013
			ES 2627916 T3	01-08-2017
			FR 2959905 A1	11-11-2011
			PL 2567500 T3	29-09-2017
			RU 2012151859 A	10-06-2014
			TW 201203990 A	16-01-2012
			US 2013046969 A1	21-02-2013
			WO 2011138333 A1	10-11-2011
