(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

COURBEVOIE

(11) **N° de publication :** **3 074 592**
(à n'utiliser que pour les commandes de reproduction)

(21) **N° d'enregistrement national :** **17 61505**

(51) Int Cl[8] : ***G 06 F 21/60*** *(2018.01)*

(12) **BREVET D'INVENTION** **B1**

(54) PROCEDE DE PARTAGE D'UNE CLE SERVANT A DERIVER DES CLES DE SESSION POUR CRYPTER ET AUTHENTIFIER DES COMMUNICATIONS ENTRE UN OBJET ET UN SERVEUR.

(22) **Date de dépôt :** 01.12.17.

(30) **Priorité :**

(43) **Date de mise à la disposition du public de la demande :** 07.06.19 Bulletin 19/23.

(45) **Date de la mise à disposition du public du brevet d'invention :** 25.10.19 Bulletin 19/43.

(56) **Liste des documents cités dans le rapport de recherche :**

Se reporter à la fin du présent fascicule

(60) **Références à d'autres documents nationaux apparentés :**

(○) **Demande(s) d'extension :**

(71) **Demandeur(s) :** *IDEMIA IDENTITY & SECURITY FRANCE Société par actions simplifiée* — FR.

(72) **Inventeur(s) :** BRINGER JULIEN et BOUATOU VINCENT.

(73) **Titulaire(s) :** IDEMIA IDENTITY & SECURITY FRANCE Société par actions simplifiée.

(74) **Mandataire(s) :** REGIMBEAU.

FR 3 074 592 - B1

DOMAINE DE L'INVENTION

La présente invention se rapporte au domaine des objets connectés.

L'invention concerne tout particulièrement un procédé de partage d'une clé servant à dériver des clés de session pour crypter et authentifier des communications entre un objet et un serveur.

L'invention trouve avantageusement application dans des réseaux d'objets connectés conformes à la spécification LoRaWAN®.

ETAT DE LA TECHNIQUE

La spécification LoRaWAN définit un protocole de communication pour un réseau d'objets connectés. Dans un tel réseau, un objet (parfois appelé terminal ou « end-device » en anglais) est amené à communiquer avec un serveur faisant partie d'un tel réseau.

La spécification LoRaWAN version 1 propose d'utiliser deux clés de session NwkSKey, AppSKey pour crypter et authentifier des communications entre l'objet et le serveur.

Par ailleurs, les clés de session NwkSKey, AppSKey doivent être dérivées d'une clé d'application AppKey, lorsque cet objet envoie au serveur une requête d'adhésion à un réseau d'objets connectés. Chaque objet dispose de sa propre clé d'application AppKey. Cette clé d'application AppKey est fournie par un fournisseur d'application.

Une contrainte importante est que la clé d'application AppKey doit être un secret partagé entre l'objet et le serveur.

Pour que cette contrainte soit respectée, la spécification LoRaWAN propose de précharger une valeur de clé d'application AppKey dans une mémoire de l'objet, au cours d'une étape de configuration de l'objet réalisée avant sa mise en service.

Un tel pré-chargement est toutefois contraignant. En effet, on ne connaît pas forcément, au moment où cette étape de configuration est mise en œuvre, avec quel serveur l'objet sera amené à communiquer une fois mis en service.

EXPOSE DE L'INVENTION

Un but de l'invention est de partager une clé entre un objet connecté et un serveur d'un réseau d'objets connectés d'une manière moins contraignante tout en étant sécurisée, la clé servant à dériver des clés de session pour crypter et authentifier des communications entre l'objet et le serveur.

Il est ainsi proposé, selon un premier aspect de l'invention, un procédé de partage d'une clé de référence entre un objet susceptible d'être connecté à un réseau d'objets connectés et au moins un serveur, le procédé comprenant des étapes de :

- application, par l'objet, d'une première fonction prédéterminée à au moins une première donnée et à une clé propre à un élément sécurisé de l'objet, de sorte à générer la clé de référence,
- transmission au serveur d'une requête d'adhésion de l'objet à un réseau d'objets connectés, la requête d'adhésion comprenant la première donnée, la clé propre à l'élément sécurisé n'étant en revanche pas transmise par l'objet au serveur,
- obtention, par le serveur, de la clé propre à l'élément sécurisé sur la base de la requête d'adhésion,
- application, par le serveur, de la première fonction prédéterminée à la première donnée et à la clé obtenue par le serveur, de sorte à obtenir la clé de référence.

Le procédé selon le premier aspect de l'invention, peut comprendre les caractéristiques optionnelles suivantes, prises seules ou en combinaison lorsque cela est techniquement possible.

Le procédé peut comprendre par ailleurs des étapes d'application, par l'objet, d'une deuxième fonction prédéterminée à une deuxième donnée et à la clé de référence, de sorte à générer un code d'intégrité, et de transmission au serveur du code d'intégrité généré. Dans ce cas, l'obtention par le serveur de la clé propre à l'élément sécurisé comprend les étapes suivantes :

- pour au moins une clé d'élément sécurisé candidate mémorisée par le serveur :
    a) application de la première fonction prédéterminée à la première donnée transmise dans la requête d'adhésion et à une clé d'élément sécurisé candidate mémorisée par le serveur, de sorte à générer une clé de référence candidate),
    b) application de la deuxième fonction prédéterminée à la deuxième donnée transmise dans la requête d'adhésion et à la clé de référence candidate générée), de sorte à générer un code d'intégrité candidat),
- sélection d'une clé de référence candidate ayant servi à générer un code d'intégrité candidat égal au code d'intégrité relatif à la requête d'adhésion.

Le procédé peut comprendre par ailleurs : l'application, par l'objet, d'une troisième fonction prédéterminée à un identifiant unique de l'élément sécurisé de l'objet et à une donnée d'aléa ayant une valeur comprise dans un ensemble fini prédéterminé, de sorte à générer un nonce ; la transmission du nonce au serveur ; pour au moins un identifiant unique d'élément sécurisé candidat et pour au moins une valeur candidate de l'ensemble fini prédéterminé, l'application de la troisième fonction prédéterminée à l'identifiant candidat et à la valeur candidate, de sorte à générer un nonce candidat ; et la sélection d'au moins

une clé d'élément sécurisé candidate associée, dans une table de correspondance, à un identifiant candidat ayant servi à générer un nonce candidat égal au nonce transmis dans la requête d'adhésion, dans lequel l'étape a) est mise en œuvre uniquement pour chaque clé d'élément sécurisé candidate sélectionnée.

La donnée d'aléa peut être un compteur incrémenté par l'objet pour chaque nouvelle requête d'adhésion.

La troisième fonction peut comprendre une fonction de hachage prenant en entrée l'identifiant de l'élément sécurisé de l'objet et la donnée d'aléa, suivie d'une troncature.

Le nonce peut être une donnée DevNonce conforme à la spécification LoRaWAN et/ou être incluse dans la requête d'adhésion.

L'obtention de la clé propre à l'élément sécurisé par le serveur peut comprend des étapes de : identification, dans une première table de correspondance, d'un identifiant d'élément sécurisé associé à la première donnée transmise dans la requête d'adhésion, et identification, dans une deuxième table de correspondance distincte de la première table de correspondance, d'une clé associée à l'identifiant unique d'élément sécurisé identifié dans la première table de correspondance.

La requête d'adhésion peut être un message conforme à la spécification LoRaWAN.

La première donnée peut être ou dépendre d'au moins une des données suivantes : un identifiant unique propre à l'objet, un identifiant d'application unique propre à un fournisseur d'application de l'objet, un nonce.

La clé de référence peut servir à dériver des clés de session pour crypter et authentifier des communications entre l'objet et le serveur ou sert à partager une autre clé entre l'objet et le serveur.

Il est également proposé, selon un deuxième aspect de l'invention, un système comprenant un objet susceptible d'être connecté à un réseau d'objets connectés, et au moins un serveur, dans lequel l'objet est configuré pour :

- appliquer une première fonction prédéterminée à au moins une première donnée et à une clé propre à un élément sécurisé de l'objet, de sorte à générer la clé de référence,
- transmettre au serveur une requête d'adhésion de l'objet au réseau d'objets connectés, la requête d'adhésion comprenant la première donnée, la clé propre à l'élément sécurisé n'étant en revanche pas transmise par l'objet au serveur,

et dans lequel le serveur est configuré pour :

- obtenir la clé propre à l'élément sécurisé sur la base de la requête d'adhésion,

- appliquer la première fonction prédéterminée à la première donnée et à la clé obtenue par le serveur, de sorte à obtenir la clé de référence.

DESCRIPTION DES FIGURES

D'autres caractéristiques, buts et avantages de l'invention ressortiront de la description qui suit, qui est purement illustrative et non limitative, et qui doit être lue en regard des dessins annexés sur lesquels :

- La figure 1 illustre de façon schématique un système comprenant un objet et un serveur pour réseau d'objets connectés, selon un mode de réalisation ;
- Les figures 2 et 3 illustrent des étapes d'un procédé de partage d'une clé entre l'objet et le serveur illustrés sur la figure 1, selon un mode de réalisation de l'invention.

Sur l'ensemble des figures, les éléments similaires portent des références identiques.

DESCRIPTION DETAILLEE DE L'INVENTION

On a schématiquement représenté **en figure 1** un objet 1 susceptible d'être connecté à un réseau d'objets connectés via un serveur 2.

L'objet 1 comprend un élément sécurisé 4, au moins un processeur 6, au moins une mémoire 8 et une interface de communication 10 avec le serveur 2.

Le processeur 6 est configuré pour exécuter des programmes d'ordinateur, et en particulier une première fonction f, voire une deuxième fonction g et une troisième fonction h qui seront détaillées dans la suite.

L'élément sécurisé 4 peut être un circuit physique dédié. Ce circuit dédié peut être amovible, par exemple au format UICC. Alternativement, l'élément sécurisé 4 est virtuel, dans le sens où il est formé par un programme d'ordinateur spécifique susceptible d'être exécuté par le processeur 6 de l'objet 1.

L'interface de communication 10 est par exemple du type radio sans fil.

Dans ce qui suit, on prendra l'exemple d'une interface de communication 10 destinée à émettre ou recevoir des messages conformes à la spécification LoRaWAN®, auquel cas le réseau d'objets connecté est un réseau également conforme à cette spécification LoRaWAN.

L'élément sécurisé 4 mémorise une clé KSE qui lui est propre.

L'élément sécurisé 4 dispose par ailleurs d'un identifiant unique IDSE qui lui est propre.

Par ailleurs, la mémoire 8 de l'objet 1 comprend au moins une mémoire non volatile est destiné à stocker des données de calcul de manière persistante (HDD, SSD, Flash, etc.), en particulier les données suivantes définies par la spécification LoRaWAN version 1 :

- DevEUI : un identifiant unique propre à l'objet 1, assimilable à un numéro de série.

- AppEUI : un identifiant d'application unique propre à un fournisseur d'application de l'objet 1. Il permet de classer différents objets tels que l'objet 1 par application.

Le serveur 2 comprend au moins un processeur 12, au moins une mémoire 14 et une interface de communication 16 avec au moins un objet 1 tel que l'objet 1 représenté en figure 1.

Le processeur 12 du serveur 2 est également configuré pour exécuter des programmes d'ordinateur, et en particulier la première fonction f, voire la deuxième fonction g et la troisième fonction h précitées.

La mémoire 14 du serveur 2 comprend au moins une mémoire non volatile est destiné à stocker des données de calcul de manière persistante (HDD, SSD, Flash, etc.), en particulier des tables de correspondances dont le contenu sera détaillé dans la suite.

L'interface de communication 16 est configurée pour interagir avec l'interface de communication 10 de l'objet 1 ; elle est par conséquent du même type.

Pour que l'objet 1 et le serveur 2 puisse communiquer l'un avec l'autre de manière sécurisée, au moins une clé de référence doit être partagée entre le client et le serveur.

La clé de référence sert par exemple à dériver des clés de session pour crypter et authentifier des communications entre l'objet 1 et le serveur.

Lorsque le réseau d'objets connectés est le réseau LoRaWAN version 1, cas non limitatif que l'on prendra en exemple dans la suite, cette clé de référence est la clé d'application AppKey, laquelle permet de dériver les clés de session NwkSKey, AppSKey lesquelles permettent de crypter et authentifier des communications entre l'objet 1 et le serveur 2.

En référence à la **figure 2**, un procédé de partage d'une clé de référence AppKey entre l'objet 1 et le serveur 2 selon un premier mode de réalisation comprend les étapes suivantes.

Le processeur 6 de l'objet 1 génère un nonce (étape 100). Ce nonce est destiné à être passé en paramètre d'une requête d'adhésion à un réseau d'objets connecté, transmise par l'objet 1 au serveur 2. Le serveur 2 garde trace de certaines valeurs de ce nonce utilisées par l'objet 1 par le passé, et ignore toute requête d'adhésion qui réutiliserait l'une de ces valeurs, de manière à assurer une protection contre des attaques par rejeu.

Ce nonce est la donnée DevNonce définie dans la spécification LoRaWAN. Il est codé sur deux octets.

Par ailleurs, l'objet 1 applique la première fonction f à au moins une première donnée, et à la clé KSE propre à l'élément sécurisé 4 de l'objet 1, de sorte à générer une clé de référence AppKey (étape 102).

Cette étape 102 peut être mise en œuvre par le processeur 6 de l'objet 1, mais est de préférence mis en œuvre par l'élément sécurisé 4 de façon à éviter que la clé KSE n'ait pas à être extraite hors de cet élément sécurisé 4.

La première donnée peut être l'une des données suivantes définies par la spécification LoRaWAN : DevEUI ou AppEUI (préalablement mémorisées dans la mémoire de l'objet 1) bien ou DevNonce (nonce calculé au cours de l'étape 100).

Au cours de sa durée de vie, l'objet 1 peut être amené à contenir différents éléments sécurisés. Par exemple, lorsque l'élément sécurisé 4 se présente sous la forme d'un circuit physique amovible, ce circuit peut être remplacé. La génération 102 de la clé AppKey par l'objet 1 connecté peut ainsi être mise en œuvre à chaque fois qu'un nouvel élément sécurisé est inclus dans l'objet 1 connecté, c'est-à-dire à chaque fois que la clé passée en entrée de la première fonction change au sein de l'objet 1.

En particulier, l'identifiant AppEUI peut être amené à évoluer au cours du cycle de vie de l'objet, par exemple si l'objet 1 change de propriétaire et/ou de serveur associé.

Le processeur 6 de l'objet 1 génère par ailleurs une requête d'adhésion à un réseau d'objets connectés (étape 106).

La requête d'adhésion est un message join-request conforme à la spécification LoRaWAN. Le message join-request comprend alors, de manière conventionnelle, les données suivantes : DevEUI, AppEUI et DevNonce. Par conséquent, la première donnée, ayant servi à calculer la clé de référence AppKey et étant susceptible d'être l'une des trois données qui précèdent, est incluse dans la requête d'adhésion.

L'objet 1 génère par ailleurs un code d'intégrité MIC relatif à la requête d'adhésion (étape 102). Ce code d'intégrité MIC a pour but d'authentifier la requête d'adhésion auprès d'un destinataire, autrement dit, de permettre à un tel destinataire de vérifier si la requête d'adhésion a été émis par un objet 1 agréé et si les données contenues dans la requête d'adhésion sont cohérentes.

Le code d'intégrité MIC est généré par application de la deuxième fonction g à à l'une des données comprises dans la requête d'adhésion générée et à la clé de référence AppKey (étape 104).

Cette donnée, dite dans la suite deuxième donnée, peut en particulier être la première donnée, ou bien distincte de la première donnée.

La deuxième fonction g est par exemple une fonction de chiffrement AES.

La requête d'adhésion est transmise par l'interface de communication au serveur 2 (étape 108). Le code d'intégrité MIC est également transmis au serveur 2.

Dans le cas du réseau LoRaWAN, le message join-request est encapsulé par le processeur 6 dans un message MAC, de même que le code d'intégrité MIC, avant leur envoi au serveur. Le message MAC est typiquement relayé au serveur 2 par un opérateur (LoRa Network Operator).

En référence à la **figure 3**, la requête d'adhésion est reçue par l'interface de communication 16 du serveur 2, qui le transmet au processeur 12 pour être traitée.

Le processeur 12 du serveur 2 obtient la clé KSE propre à l'élément sécurisé 4, en s'aidant de la requête d'adhésion reçue. Toutefois, cette obtention est mise en œuvre autrement que par une transmission de la clé KSE propre à l'élément sécurisé 4 par l'objet 1 au serveur. Cette obtention est mise en œuvre au moyen des étapes suivantes.

Ont été préalablement mémorisées dans la mémoire 14 du serveur 2 une pluralité de clés d'élément sécurisé candidates KSE(1), ..., KSE(n). L'une d'entre elles est susceptible d'être égale à la clé KSE de l'élément sécurisé 4 de l'objet 1.

Le processeur 12 met en œuvre une boucle comprenant au moins une itération, chaque itération utilisant une des clés d'élément sécurisé candidates KSE(1), ..., KSE(n).

Une itération d'indice i comprend les étapes 208, 210, 212 suivantes.

Le processeur 12 applique la première fonction prédéterminée f à la première donnée (DevEUI, AppEUI ou DevNonce) contenue dans la requête d'adhésion reçue, et à la clé d'élément sécurisé candidate KSE(i) mémorisée par le serveur 2, de sorte à générer une clé de référence candidate AppKey(i) (étape 208).

Ensuite, le processeur 12 applique la deuxième fonction prédéterminée g à la deuxième donnée (DevEUI, AppEUI, ou DevNonce) qui a été reçue par le serveur dans la requête d'adhésion, et à la clé de référence candidate AppKey(i), de sorte à générer un code d'intégrité candidat MIC(i) associé à la clé de référence candidate AppKey(i) (étape 210).

Le processeur 12 compare ensuite le code d'intégrité candidat MIC(i) au code d'intégrité MIC contenu dans la requête d'adhésion reçue par le serveur 2 (étape 212).

Si les deux codes ne sont pas égaux, une nouvelle itération est mise en œuvre pour une clé d'élément sécurisé candidate suivante KSE(i+1). Les étapes 208, 210, 212 sont alors répétées sur la base de la clé KSE(i+1).

Si le processeur 12 met en œuvre n itérations et qu'il est constaté qu'aucun des codes d'intégrité candidats MIC(1), ..., MIC(n) n'est égal au code d'intégrité MIC reçu, alors la requête d'adhésion est rejetée par le serveur.

En revanche, si deux codes comparés MIC(i) et MIC sont égaux, alors cela signifie que la requête d'adhésion reçue par le serveur 2 provient bien d'un objet 1 agréé. Mais cela signifie également que la clé de référence candidate AppKey(i), ayant permis de générer le

code d'intégrité candidat MIC(i), est égal à la clé AppKey détenue par l'objet 1, puisque l'objet 1 et le serveur ont chacun utilisé la même deuxième fonction prédéterminée g pour générer les deux codes d'intégrité comparés. Le processeur 12 du serveur 2 considère donc que la clé propre à l'élément sécurisé 4 est la clé AppKey(i).

En définitive, le processeur 12 sélectionne en tant que clé AppKey la clé de référence candidate ayant permis de générer un code d'intégrité candidat égal au code d'intégrité (MIC) relatif à la requête d'adhésion (étape 214).

De façon conventionnelle, le serveur 2 transmet à l'objet 1 un message de réponse « join-accept », lui indiquant que la requête d'adhésion a été acceptée par le serveur 2.

La clé de référence AppKey devient à ce stade une clé partagée entre l'objet 1 et le serveur 2.

Par la suite, la clé de référence AppKey partagée peut être utilisée conventionnellement pour dériver les clés de session NwkSKey, AppSKey pour crypter et authentifier des communications entre l'objet 1 et le serveur.

Bien entendu, le nombre maximal de calculs effectués par le serveur 2 pour récupérer la clé de référence AppKey au cours du procédé de partage qui précède est proportionnel au nombre n de clés candidates KSE(i) mémorisées par le serveur 2. Par conséquent, il est avantageusement mis les étapes additionnelles suivantes au cours du procédé, afin de réduire le nombre de calculs effectués en pratique par le serveur 2 pour récupérer cette clé de référence AppKey.

De retour à la figure 2, au cours de l'étape 100, le processeur 6 de l'objet 1 ou l'élément sécurisé 4 applique la troisième fonction prédéterminée h à l'identifiant unique (IDSE) de l'élément sécurisé 4 de l'objet 1 et à une donnée d'aléa ayant une valeur comprise dans un ensemble fini prédéterminé de m valeurs, de sorte à générer le DevNonce qui sera ensuite inclus dans la requête d'adhésion à transmettre au serveur.

La troisième fonction est par exemple la suivante :

$$DevNonce = trunc(hash(IDSE, counter))$$

où

- $counter$ est la donnée d'aléa,
- $hash$ désigne une fonction de hachage,
- $trunc$ est une troncature qui ne conserve que 2 octets du résultat de la fonction de hachage.

De retour à la figure 3, une table de correspondance entre des identifiants uniques d'éléments sécurisé IDSE(1), …, IDSE(n) et les clés d'éléments sécurisé KSE(1), …, KSE(n) discutées précédemment a été préalablement mémorisée dans la mémoire 14 du serveur 2.

Autrement dit, pour tout i allant de 1 à n, l'identifiant unique IDSE(i) et la clé KSE(i) se rapportent à un même élément sécurisé, et sont donc mis en correspondance dans la table de correspondance mémorisée dans la mémoire 14 du serveur 2.

Bien entendu, l'un des identifiants candidats IDSE(1), ..., IDSE(n) est susceptible d'être égal à l'identifiant unique IDSE mémorisé par l'élément sécurisé 4 de l'objet 1.

On suppose par ailleurs que le serveur connaît l'ensemble fini auquel appartient la donnée d'aléa.

Le processeur 12 du serveur 2 met en œuvre les étapes suivantes pour au moins un identifiant unique d'élément sécurisé candidat IDSE(i) d'indice i et pour au moins une valeur candidate de l'ensemble fini prédéterminé d'indice j.

Le processeur applique la troisième fonction prédéterminée h à l'identifiant candidat IDSE(i) et à la valeur candidate d'indice j, de sorte à générer un nonce candidat, que l'on nomme par convention DevNonce(i,j) (étape 200).

Le processeur compare le DevNonce reçu dans la requête d'adhésion avec chaque nonce candidat généré (202).

Si un nonce candidat généré DevNonce(i,j) est égal au DevNonce, alors le processeur 12 sélectionne la clé d'élément sécurisé KSE(i) associée, dans la table de correspondance, à l'identifiant unique d'élément sécurisé candidat IDSE(i) ayant servi à générer le nonce candidat DevNonce(i,j) (étape 204).

Si un nonce candidat généré DevNonce(i,j) est différent du DevNonce, alors le processeur 12 ne sélectionne pas, au cours de l'étape 204, la clé d'élément sécurisé KSE(i) associée, dans la table de correspondance, à l'identifiant unique d'élément sécurisé candidat IDSE(i) ayant servi à générer le nonce candidat DevNonce(i,j).

Le processeur répète par exemple les étapes 200, 202, 204 pour tout i allant de 1 à n, et pour tout élément de l'ensemble fini (de cardinal m).

Il est à noter que le nombre de clés d'éléments sécurisés candidats IDSE(i) sélectionné par le serveur peut être égal à 1, mais peut également être supérieur à 1. En effet, la troncature *trunc* de la troisième fonction h est susceptible de générer des collisions, c'est-à-dire produire des DevNonces(i,j) de même valeur à partir de données d'entrées de la troisième fonction pourtant différentes.

Les étapes 208, 210 et 212 sont ensuite mises en œuvre uniquement pour chaque clé d'élément sécurisé candidate sélectionnée au cours de l'étape 204.

Le calcul préalable 200 des nonces candidats permet ainsi de réaliser un pré-filtrage de clés d'éléments sécurisés KSE(i) candidates mémorisées dans la mémoire 14 du serveur 2, dont le nombre peut être extrêmement grand. Certes, les étapes additionnelles 200, 202,

204 de ce pré-filtrage induisent une charge de calcul supplémentaire ; toutefois, cette charge supplémentaire, relativement petite, est largement compensée par le fait que le nombre d'itérations faisant appel à la première fonction (dans l'étape 208) et à la deuxième fonction (dans l'étape 210) est fortement réduit. La charge de calcul globale consommée par le serveur pour obtenir la clé KSE de l'élément sécurisé 4 est donc en définitive fortement réduite grâce aux étapes 208, 210, 212.

Cette économie de charge de calcul est particulièrement avantageuse lorsque les calculs sont confiés à un module matériel de sécurité (« Hardware Security Module » en anglais, abrégé en HSM) ne disposant que de faibles ressources matérielles.

De plus, cette économie de charge de calcul est obtenue sans qu'il ait été besoin de modifier le format de la requête d'adhésion join-request. En effet, le nonce DevNonce, déjà imposé par la spécification LoRaWAN afin d'éviter des attaques par rejeu, est astucieusement calculé à partir d'une donnée caractéristique de l'élément sécurisé 4 (l'identifiant unique IDSE cet élément sécurisé), puis mis à profit par le serveur pour mettre en œuvre un pré-filtrage de clés KSE(i) pertinent côté serveur 2.

La donnée d'aléa utilisée pour calculer le DevNonce côté objet 1 est par exemple la valeur courante d'un compteur, et l'ensemble fini est constitué des entiers allant de 0 à m-1. Le compteur a donc m-1 comme valeur maximale. Le compteur est incrémenté par l'objet 1 à chaque fois que cet objet doit émettre une nouvelle requête d'adhésion (peu importe que cette requête soit acceptée ou non par le serveur). Le compteur est remis à zéro lorsqu'il a atteint sa valeur maximale m.

Le serveur 2 utilise également la valeur d'un deuxième compteur CO comme donnée d'entrée de la troisième fonction h pour générer un nonce candidat au cours de l'étape 200. Ce compteur est incrémenté par le serveur pour chaque mise en œuvre de l'étape 200. On note CO(i) la valeur du deuxième compteur au début du procédé.

Le premier compteur, côté objet 1, est incrémenté pour chaque nouvelle requête d'adhésion à émettre. Or, le serveur 2 n'est pas forcément au courant d'une telle émission, et il peut donc il y avoir une désynchronisation entre le premier compteur et le deuxième compteur. Il est dès lors avantageusement utilisé un écart prédéterminé Δ. Pour tout i allant de 1 à n, le serveur calcule Devnonce pour toutes les valeurs de compteur allant de CO(i) à CO(i)+ Δ. Il s'arrête dès qu'il trouve une valeur correcte.

Il est à noter que les étapes 200 à 214 peuvent être mises en œuvre par un seul est même serveur 2, ou bien peuvent être mises en œuvre par un ensemble de serveurs, chaque serveur ne mettant en œuvre qu'une partie des étapes. Par exemple, un premier serveur

peut se charger de réaliser le pré-filtrage de clés (étapes 200, 202, 204), et un deuxième serveur mettre en œuvre les étapes 208 à 214.

D'autres manières d'obtenir la clé KSE côté serveur peuvent être mises en œuvre.

Dans un deuxième mode de réalisation, le serveur – ou un ensemble de serveurs - met en œuvre les étapes suivantes.

Ont été préalablement mémorisées dans la mémoire 14 du serveur 2 deux tables de correspondance :

- une première table de correspondance entre des valeurs possible pour la première donnée (qui, on le rappelle, peut être DevEUI, AppEUI, ou DevNonce) et des identifiants de clés d'élément sécurisés IDSE(1), …, IDSE(n), et
- une deuxième table similaire à celle utilisée dans le premier mode de réalisation décrit au regard des figures 2 et 3, c'est-à-dire une table de correspondance entre des identifiants uniques d'éléments sécurisé IDSE(1), …, IDSE(n) et des clés d'éléments sécurisé KSE(1), …, KSE(n).

Le serveur identifie, dans la première table de correspondance, un identifiant d'élément sécurisé associé à la première donnée (DevEUI, AppEUI ou DevNonce) transmise dans la requête d'adhésion (join-request).

Le serveur identifie ensuite, dans la deuxième table de correspondance, une clé associée à l'identifiant unique d'élément sécurisé identifié dans la première table de correspondance. Cette clé est la clé KSE.

Le fait d'utiliser deux tables de correspondances apporte une certaine flexibilité de mise en œuvre. L'une des deux tables peut être mémorisée et parcourue par un premier serveur, et l'autre table être mémorisée et parcourue par un deuxième serveur différent du premier serveur ; en variante, les deux tables peuvent être mémorisées et parcourues par un seul et même serveur.

Il a été décrit précédemment un procédé de partage d'une clé de référence servant à dériver des clés de session pour crypter et authentifier des communications entre l'objet 1 et le serveur 2. Toutefois, la clé de référence peut avoir une destination différente. La clé de partage est par exemple une clé servant à partager une autre clé entre l'objet 1 et le serveur 2, cette autre clé ayant une destination prédéterminée. Le partage de l'autre clé peut comprendre le calcul d'une même valeur de clé côté objet 1 et côté serveur 2. En variante, au moins l'un de l'objet 1 et du serveur 2 peut connaître à l'avance la valeur de l'autre clé, mais ne pas savoir à l'avance que cette valeur a cette destination prédéterminée. Le partage de cette clé apporte la confirmation à l'entité connaissant à l'avance cette valeur doit être utilisée pour cette destination prédéterminée.

Par ailleurs, l'invention est également applicable à d'autres protocoles que celui défini dans la spécification LoRaWAN version 1. En particulier, la clé de référence peut être la clé nommée NwkKey dans la spécification LoRaWAN version 1.1.

REVENDICATIONS

1. Procédé de partage d'une clé de référence (AppKey) entre un objet (1) susceptible d'être connecté à un réseau d'objets connectés et au moins un serveur (2), le procédé comprenant des étapes de :

- application (102), par l'objet (1), d'une première fonction prédéterminée (f) à au moins une première donnée (DevEUI, AppEUI, DevNonce) et à une clé (KSE) propre à un élément sécurisé de l'objet (1), de sorte à générer la clé de référence (AppKey),
- transmission (108) au serveur (2) d'une requête d'adhésion (join-request) de l'objet (1) au réseau d'objets connectés, la requête d'adhésion (join-request) comprenant la première donnée (DevEUI, AppEUI, DevNonce), la clé (KSE) propre à l'élément sécurisé (4) n'étant en revanche pas transmise par l'objet (1) au serveur (2),
- obtention, par le serveur (2), de la clé (KSE) propre à l'élément sécurisé (4) sur la base de la requête d'adhésion (join-request),
- application (208), par le serveur (2), de la première fonction prédéterminée (f) à la première donnée (DevEUI, AppEUI, DevNonce) et à la clé (KSE) obtenue par le serveur (2), de sorte à obtenir la clé de référence (AppKey).

2. Procédé selon la revendication 1, comprenant par ailleurs des étapes de :

- application (104), par l'objet (1), d'une deuxième fonction prédéterminée (g) à une deuxième donnée (DevEUI, AppEUI, DevNonce) et à la clé de référence (AppKey), de sorte à générer un code d'intégrité (MIC),
- transmission (108) au serveur (2) du code d'intégrité (MIC) généré,

et dans lequel l'obtention par le serveur (2) de la clé propre à l'élément sécurisé (4) comprend des étapes de :

- pour au moins une clé d'élément sécurisé candidate mémorisée par le serveur (2) :
  - c) application (208) de la première fonction prédéterminée (f) à la première donnée (DevEUI, AppEUI, DevNonce) transmise dans la requête d'adhésion (join-request) et à une clé d'élément sécurisé candidate mémorisée par le serveur (2), de sorte à générer une clé de référence candidate (AppKey(i)),
  - d) application (210) de la deuxième fonction prédéterminée (g) à la deuxième donnée (DevEUI, AppEUI, DevNonce) transmise dans la requête d'adhésion (join-request) et à la clé de référence candidate générée (AppKey(i)), de sorte à générer un code d'intégrité candidat (MIC(i)),

- sélection d'une clé de référence candidate ayant servi à générer un code d'intégrité candidat égal au code d'intégrité (MIC) relatif à la requête d'adhésion (join-request).

3. Procédé selon la revendication 2, comprenant en outre des étapes de

- application (100), par l'objet (1), d'une troisième fonction prédéterminée (h) à un identifiant unique (IDSE) de l'élément sécurisé (4) de l'objet (1) et à une donnée d'aléa (counter) ayant une valeur comprise dans un ensemble fini prédéterminé, de sorte à générer un nonce (DevNonce),
- transmission (108) du nonce (DevNonce) au serveur (2),
- pour au moins un identifiant unique d'élément sécurisé candidat et pour au moins une valeur candidate de l'ensemble fini prédéterminé, application (200) de la troisième fonction prédéterminée (h) à l'identifiant candidat et à la valeur candidate, de sorte à générer un nonce candidat,
- sélection (204) d'au moins une clé d'élément sécurisé candidate associée, dans une table de correspondance, à un identifiant candidat ayant servi à générer un nonce candidat égal au nonce transmis dans la requête d'adhésion (join-request), dans lequel l'étape a) est mise en œuvre uniquement pour chaque clé d'élément sécurisé candidate sélectionnée.

4. Procédé selon la revendication 3, dans lequel la donnée d'aléa est un compteur incrémenté par l'objet (1) pour chaque nouvelle requête d'adhésion.

5. Procédé selon l'une des revendications 3 et 4, dans lequel la troisième fonction (h) comprend une fonction de hachage prenant en entrée l'identifiant (IDSE) de l'élément sécurisé (4) de l'objet (1) et la donnée d'aléa (counter), suivie d'une troncature.

6. Procédé selon l'une des revendications 3 à 5, dans lequel le nonce est une donnée DevNonce conforme à la spécification LoRaWAN et/ou est incluse dans la requête d'adhésion.

7. Procédé selon la revendication 1, dans lequel l'obtention de la clé propre à l'élément sécurisé (4) par le serveur (2) comprend des étapes de :

- identification, dans une première table de correspondance, d'un identifiant d'élément sécurisé associé à la première donnée (DevEUI, AppEUI, DevNonce) transmise dans la requête d'adhésion (join-request),

- identification, dans une deuxième table de correspondance distincte de la première table de correspondance, d'une clé associée à l'identifiant unique d'élément sécurisé identifié dans la première table de correspondance.

8. Procédé selon l'une des revendications 1 à 7, dans lequel :

- la requête d'adhésion (join-request) est un message conforme à la spécification LoRaWAN, et/ou
- lequel la première donnée est ou dépend d'au moins une des données suivantes :
  - un identifiant unique propre à l'objet (DevEUI),
  - un identifiant d'application unique propre à un fournisseur d'application de l'objet (AppEUI),
  - un nonce (DevNonce), et/ou
- la clé de référence sert à dériver des clés de session (NwkSKey, AppSKey) pour crypter et authentifier des communications entre l'objet (1) et le serveur (2) ou sert à partager une autre clé entre l'objet (1) et le serveur (2).

9. Système comprenant un objet (1) susceptible d'être connecté à un réseau d'objets connectés, et au moins un serveur (2), dans lequel l'objet (1) est configuré pour :

- appliquer une première fonction prédéterminée (f) à au moins une première donnée (DevEUI, AppEUI, DevNonce) et à une clé (KSE) propre à un élément sécurisé de l'objet (1), de sorte à générer la clé de référence (AppKey),
- transmettre au serveur (2) une requête d'adhésion (join-request) de l'objet (1) au réseau d'objets connectés, la requête d'adhésion (join-request) comprenant la première donnée (DevEUI, AppEUI, DevNonce), la clé (KSE) propre à l'élément sécurisé (4) n'étant en revanche pas transmise par l'objet (1) au serveur (2),

et dans lequel le serveur (2) est configuré pour :

- obtenir la clé (KSE) propre à l'élément sécurisé (4) sur la base de la requête d'adhésion (join-request),
- appliquer la première fonction prédéterminée (f) à la première donnée (DevEUI, AppEUI, DevNonce) et à la clé (KSE) obtenue par le serveur (2), de sorte à obtenir la clé de référence (AppKey).

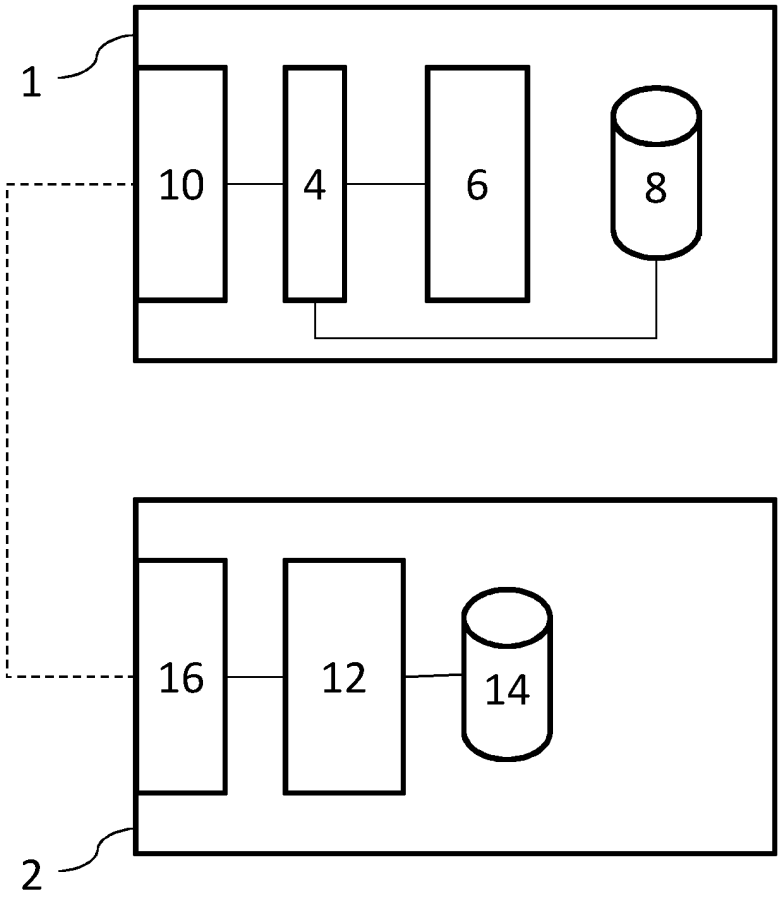

FIG. 1

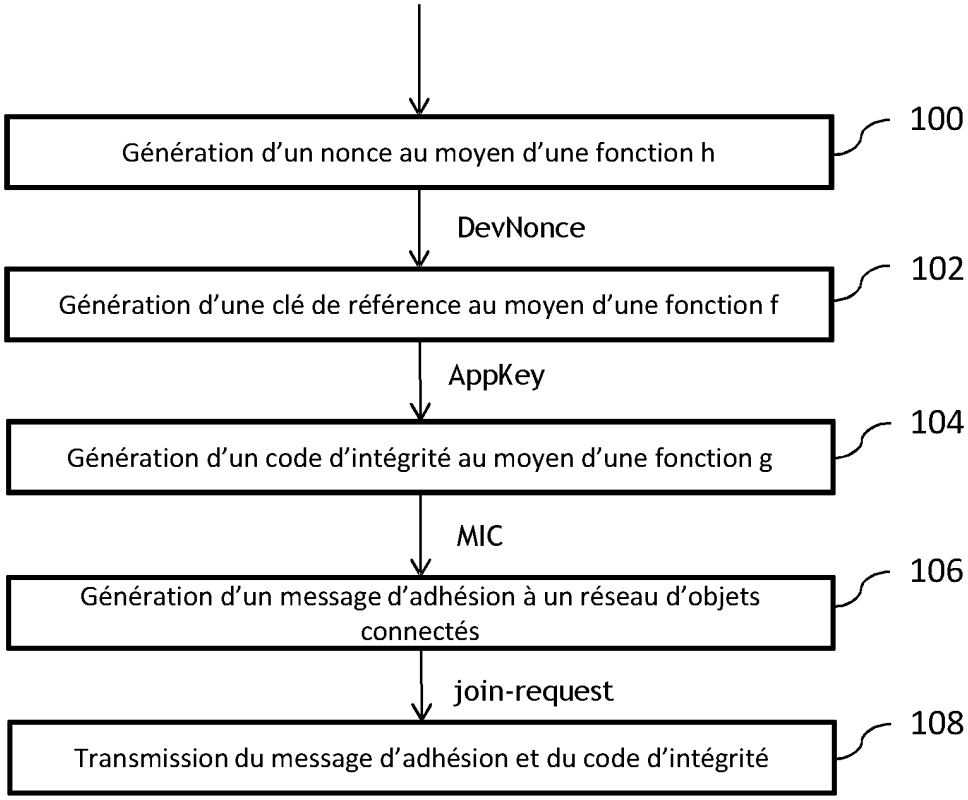
Génération d'un nonce au moyen d'une fonction h
100
DevNonce
Génération d'une clé de référence au moyen d'une fonction f
102
AppKey
Génération d'un code d'intégrité au moyen d'une fonction g
104
MIC
Génération d'un message d'adhésion à un réseau d'objets connectés
106
join-request
Transmission du message d'adhésion et du code d'intégrité
108

FIG. 2

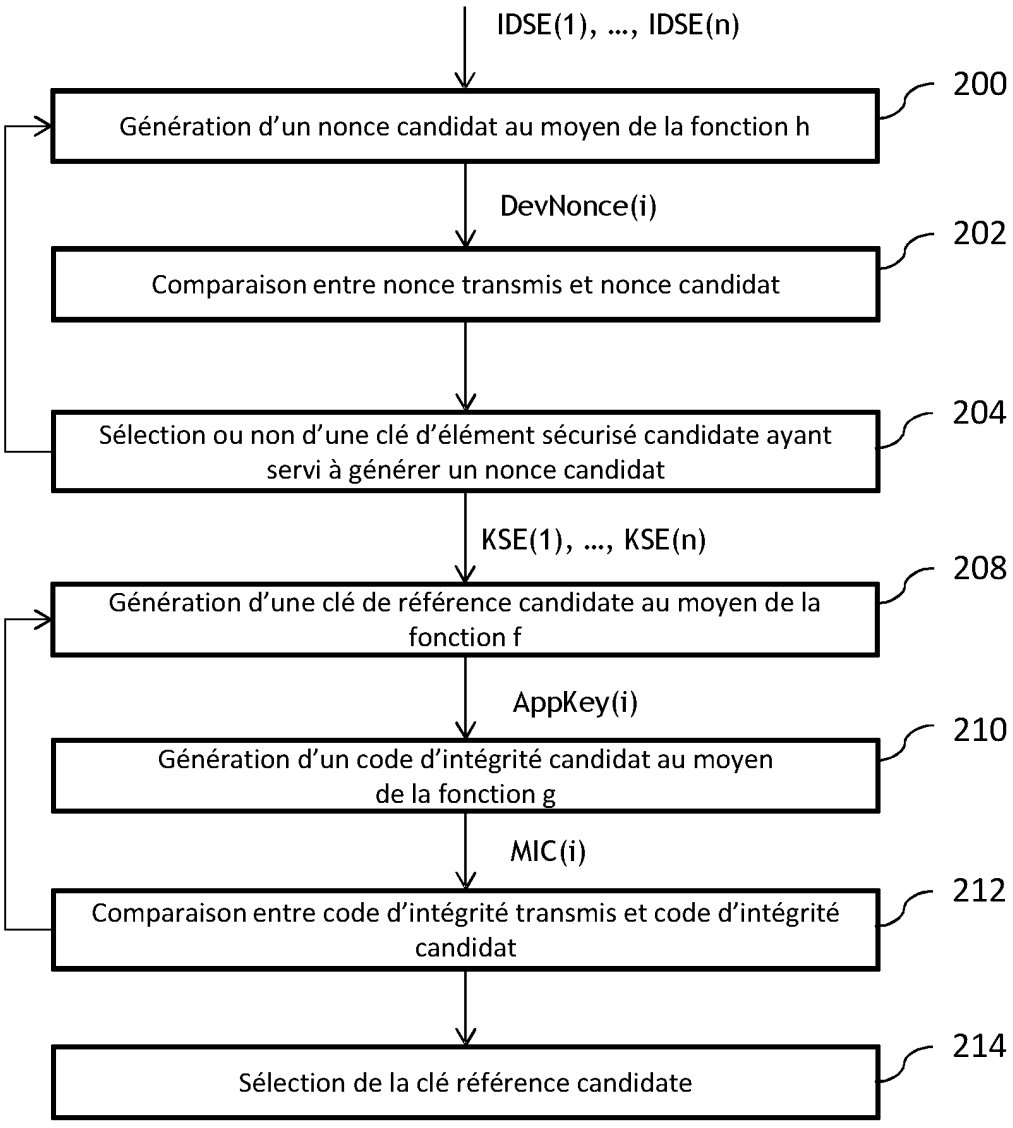
IDSE(1), ..., IDSE(n)
Génération d'un nonce candidat au moyen de la fonction h
200
DevNonce(i)
Comparaison entre nonce transmis et nonce candidat
202
Sélection ou non d'une clé d'élément sécurisé candidate ayant servi à générer un nonce candidat
204
KSE(1), ..., KSE(n)
Génération d'une clé de référence candidate au moyen de la fonction f
208
AppKey(i)
Génération d'un code d'intégrité candidat au moyen de la fonction g
210
MIC(i)
Comparaison entre code d'intégrité transmis et code d'intégrité candidat
212
Sélection de la clé référence candidate
214

FIG. 3

N° d'enregistrement national : FR1761505 N° de publication : FR3074592

RAPPORT DE RECHERCHE

articles L.612-14, L.612-53 à 69 du code de la propriété intellectuelle

OBJET DU RAPPORT DE RECHERCHE

L'I.N.P.I. annexe à chaque brevet un "RAPPORT DE RECHERCHE" citant les éléments de l'état de la technique qui peuvent être pris en considération pour apprécier la brevetabilité de l'invention, au sens des articles L. 611-11 (nouveauté) et L. 611-14 (activité inventive) du code de la propriété intellectuelle. Ce rapport porte sur les revendications du brevet qui définissent l'objet de l'invention et délimitent l'étendue de la protection.
Après délivrance, l'I.N.P.I. peut, à la requête de toute personne intéressée, formuler un "AVIS DOCUMENTAIRE" sur la base des documents cités dans ce rapport de recherche et de tout autre document que le requérant souhaite voir prendre en considération.

CONDITIONS D'ETABLISSEMENT DU PRESENT RAPPORT DE RECHERCHE

[x] Le demandeur a présenté des observations en réponse au rapport de recherche préliminaire.

[x] Le demandeur a maintenu les revendications.

☐ Le demandeur a modifié les revendications.

☐ Le demandeur a modifié la description pour en éliminer les éléments qui n'étaient plus en concordance avec les nouvelles revendications.

☐ Les tiers ont présenté des observations après publication du rapport de recherche préliminaire.

☐ Un rapport de recherche préliminaire complémentaire a été établi.

DOCUMENTS CITES DANS LE PRESENT RAPPORT DE RECHERCHE

La répartition des documents entre les rubriques 1, 2 et 3 tient compte, le cas échéant, des revendications déposées en dernier lieu et/ou des observations présentées.

[x] Les documents énumérés à la rubrique 1 ci-après sont susceptibles d'être pris en considération pour apprécier la brevetabilité de l'invention.

☐ Les documents énumérés à la rubrique 2 ci-après illustrent l'arrière-plan technologique général.

☐ Les documents énumérés à la rubrique 3 ci-après ont été cités en cours de procédure, mais leur pertinence dépend de la validité des priorités revendiquées.

☐ Aucun document n'a été cité en cours de procédure.

1. ELEMENTS DE L'ETAT DE LA TECHNIQUE SUSCEPTIBLES D'ETRE PRIS EN CONSIDERATION POUR APPRECIER LA BREVETABILITE DE L'INVENTION

Jaehyu Kim ET AL: "A Dual Key-Based Activation Scheme for Secure LoRaWAN", , 28 avril 2017 (2017-04-28), XP55416927, DOI: 10.1155/2017/6590713 Extrait de l'Internet: URL:http://downloads.hindawi.com/journals/ wcmc/aip/6590713.pdf [extrait le 2017-10-18]

NAOUI SARRA ET AL: "Trusted Third Party Based Key Management for Enhancing LoRaWAN Security", 2017 IEEE/ACS 14TH INTERNATIONAL CONFERENCE ON COMPUTER SYSTEMS AND APPLICATIONS (AICCSA), IEEE, 30 octobre 2017 (2017-10-30), pages 1306-1313, XP033330036, DOI: 10.1109/AICCSA.2017.73 [extrait le 2018-03-07]

ARAS EMEKCAN ET AL: "Exploring the Security Vulnerabilities of LoRa", 2017 3RD IEEE INTERNATIONAL CONFERENCE ON CYBERNETICS (CYBCON), IEEE, 21 juin 2017 (2017-06-21), pages 1-6, XP033126677, DOI: 10.1109/CYBCONF.2017.7985777 [extrait le 2017-07-19]

2. ELEMENTS DE L'ETAT DE LA TECHNIQUE ILLUSTRANT L'ARRIERE-PLAN TECHNOLOGIQUE GENERAL

NEANT

3. ELEMENTS DE L'ETAT DE LA TECHNIQUE DONT LA PERTINENCE DEPEND DE LA VALIDITE DES PRIORITES

NEANT