

FASCICULE DE BREVET D'INVENTION

21 Numéro de dépôt : 1202000386

22 Date de dépôt : 23/10/2020

30 Priorité(s) :
FR n° 1912020 du 27/10/2019

24 Délivré le : 07/06/2021

45 Publié le : 31/08/2021

73 Titulaire(s) :
LEX PERSONA,
2 Rue Gustave Eiffel,
10430 ROSIERS PRES TROYES (FR)72 Inventeur(s) :
DEVORET François (FR)
SCHWEBLIN Benoit (FR)
PASQUIER Julien (MA)74 Mandataire : **Cabinet BONNY & ASSOCIÉS,**
B.P. 869, YAOUNDE (CM).54 Titre : **Système ouvert et sécurisé de traitement de demande de signature électronique et procédé associé.**

57 Abrégé :

L'invention concerne un système ouvert et sécurisé de traitement de demandes de signature électronique de documents comprenant une application métier, chargée de créer une demande de signature d'au moins un document pour un utilisateur signataire, ladite application métier étant hébergée par une entité demandeur. Le système est caractérisé en ce qu'il permet de signer des documents avec un certificat serveur créé à la volée et en ce qu'il comprend en outre trois services indépendants, un service gestionnaire de demandes qui héberge les demandes de signatures et les documents joints; une page de consentement, constitué d'une application HTML, étant apte à gérer le protocole de consentement en s'interfaçant avec un service sécurisé d'identification de l'utilisateur et en générant une paire de clés client (KcliPub, KcliPriv) et un service gestionnaire de preuves, hébergé par un fournisseur de signature, collectant les preuves de signature et comprenant un composant serveur hébergeant la clé cryptographique du signataire, ledit service gestionnaire de preuves étant apte à générer une paire de clés serveur (KservPriv et KservPub) à dériver une clé secrète Ks en utilisant la clé publique du client (Kclipub) et la clé privée du serveur (KservPriv) ; à générer une clé publique (KPub) et une clé privée, (KPriv) du signataire, à créer une demande de certificat, à récupérer un certificat auprès d'une autorité de certification, et en ce que la clé secrète Ks étant utilisée pour envelopper la clé privée du signataire.

L'invention concerne en outre les procédés de traitement et d'élaboration de signature électronique de document mis en œuvre dans le système ci-dessus.

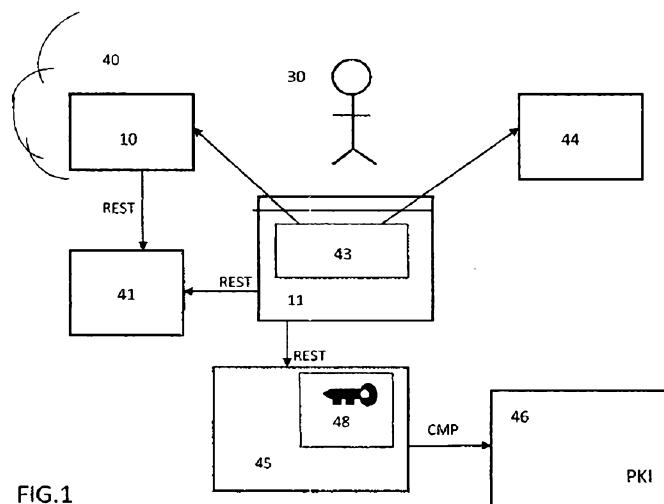


FIG.1

Figure 1

Description

Titre de l'invention : Système ouvert et sécurisé de traitement de demande de signature électronique et procédé associé

[1] DOMAINE TECHNIQUE DE L'INVENTION

[2] L'invention se rapporte au domaine de la signature électronique. Plus particulièrement, l'invention concerne un système ouvert et sécurisé pour traiter des demandes de signatures de documents électronique. L'invention concerne en outre un procédé d'élaboration et de traitement d'une demande de signature.

[3] ETAT DE LA TECHNIQUE ANTERIEUR

[4] La signature électronique consiste principalement à permettre à un utilisateur humain de chiffrer l'empreinte d'un document à signer, avec une clé privée correspondant à une clé publique associée à son identité, cette clé privée étant généralement protégée par un dispositif cryptographique et un code secret, le résultat du chiffrement devant ensuite être incorporé ou associé au document à signer de manière à constituer une preuve. Au cours de cette opération, il est nécessaire de s'assurer que l'association entre la clé publique et l'identité du signataire soit certifiée par une autorité compatible avec les exigences de sécurité et de confiance associée à la signature électronique, que cette certification soit vérifiée comme étant toujours valide, et que le signataire soit bien d'accord avec le contenu à signer.

[5] Par ailleurs, l'enchaînement des tâches de calcul, de gestion et de vérifications nécessaires à la réalisation d'une signature électronique est excessivement complexe. En effet, les algorithmes sur lesquels s'appuient les calculs doivent eux-mêmes être compatibles avec les exigences de sécurité et de confiance. De plus, les données à signer ne sont pas forcément accessibles directement par le processus de signature mais peuvent être distantes, et ces mêmes données à signer doivent pouvoir être encadrées par des éléments de contexte comme la date et l'heure de la signature, la chaîne de certification du signataire, son rôle, un lieu de signature, une politique de signature, etc. Par ailleurs, la clé privée peut être sur un dispositif cryptographique local ou distant de l'utilisateur, et

l'environnement même de ces opérations est tantôt sur le poste de travail de l'utilisateur, mais peut être également distant ou s'exécuter en mode client-serveur dans un navigateur Web, ou encore sur un smartphone ou une tablette.

[6] Document EP 1393144 B1 divulgue un procédé et un système basé sur le Web permettant la signature juridiquement exécutoire de documents dans un environnement Web. Le système comprend un premier moyen d'accès pour accéder à l'environnement Web depuis un système électronique, et comprend également une pluralité de modules. Un module de rendu du document pour présenter à l'utilisateur une représentation Web du document, un module d'information juridique pour présenter à l'utilisateur, dans l'environnement Web, de l'information juridique relative à la signature électronique du document, et pour obtenir l'accord de l'utilisateur de cette information juridique. Un module d'approbation des documents pour intégrer la signature de l'utilisateur au document, avec l'accord de l'utilisateur de l'information juridique. Le système comprend également un module de journalisation pour générer un journal des processus de la signature du document en associant ce journal de processus avec le document signé. Enfin, un module de distribution de document pour rendre le document signé disponible. Ce document concerne la traçabilité du processus. Il y a un besoin particulier de rationaliser le processus de signature électronique et aussi de masquer la complexité du processus aux utilisateurs.

[7] Document EP 3423982 A1 (demandeur Lex persona) divulgue un système ouvert et sécurisé de signature électronique comprenant une application métier qui effectue une demande de signature d'un document auprès d'un gestionnaire de signature pour un utilisateur. L'application métier définit le cahier des charges de la ou des signatures, par exemple le contenu à signer, la sélection d'un utilisateur signataire, d'un format de signature, etc... Le gestionnaire de signature coordonne la demande de signature et sert d'intermédiaire entre l'application métier et l'utilisateur en vérifiant l'habilitation de l'application métier, l'identité de l'utilisateur signataire ; en récupérant le document à signer, en préparant la demande de signature avec les calculs d'empreintes des données à signer, l'envoi d'une notification de la demande de signature à des services de signatures de l'utilisateur. L'utilisateur, au moyen des services de signatures,

peut contrôler l'exécution du processus de signature en activant la clé privée correspondant à un certificat de l'utilisateur répondant aux critères de sélection envoyés au gestionnaire de signature par l'application métier en vue du chiffrement de l'empreinte des données à signer. Dans ce document, le processus de signature électronique est décomposé en tâches indépendantes dont les interactions entre elles sont sécurisées. Ce document montre une rationalisation du processus de signature électronique et masque la complexité aux utilisateurs de la signature électronique et aux applications métiers qui souhaitent la mettre en œuvre.

[8] Le but de la présente invention est de permettre la création à la volée d'un certificat de signature sur le poste même du signataire et de signer un document. La présente invention simplifie considérablement la problématique de signature à distance et notamment la contrainte de prouver le contrôle de la clé privée par son titulaire, mais aussi la problématique de la sécurisation de cette clé privée gérée de manière centralisée ou bien la confidentialité des informations signées.

[9] Un autre but de l'invention est d'améliorer le niveau de sécurité de la transaction de signature et de permettre une identité substantielle ou renforcée mais aussi la génération d'un identifiant de la transaction de la signature permettant ainsi de générer à la volée une identité numérique spécifiquement créée pour une transaction particulière.

[10] EXPOSE DE L'INVENTION

[11] La présente invention a pour but de simplifier la problématique de signature électronique d'un document à distance, de sécuriser la gestion des clés privées et aussi d'améliorer davantage le niveau de sécurité ainsi que la confidentialité des informations qui sont signées. Pour ce faire, il est proposé un système ouvert et sécurisé de traitement de demande de signature électronique de documents comprenant une application métier, chargée de créer une demande de signature d'au moins un document pour un utilisateur signataire, ladite application métier étant hébergée par une entité demandeur, le système est caractérisé en ce qu'il permet de signer des documents avec un certificat serveur créé à la volée et en ce qu'il comprend en outre trois services indépendants, un service gestionnaire de demande qui héberge les demandes de signatures et les documents joints;

une page de consentement, constitué d'une application HTML, étant apte à gérer le protocole de consentement en s'interfaçant avec un service sécurisé d'identification de l'utilisateur et en générant une paire de clés client, KcliPub et KcliPriv, dans le navigateur web du signataire, et un service gestionnaire de preuve, hébergé par un fournisseur de signature, collectant les preuves de signature et comprenant un composant serveur hébergeant la clé cryptographique du signataire, ledit service gestionnaire de preuve étant apte à récupérer un certificat auprès d'une autorité de certification, à générer une paire de clés serveur, KservPub, KservPriv à dériver une clé secrète Ks en utilisant la clé publique du client et la clé privée du serveur, et à générer une clé publique KPub et une clé privée (KPriv) du signataire, la clé secrète Ks étant utilisée pour envelopper la clé privée du signataire.

[12] Selon des caractéristiques particulières, le service gestionnaire de demandes est protégé par une clé API de telle sorte que seule l'application métier est apte à lui faire des demandes de signature et en ce que ledit service gestionnaire de demandes étant apte à générer une requête pour la demande de signature, puis à générer un identifiant aléatoire unique pour un ou plusieurs documents de la demande de signature et à calculer le hash du ou desdits documents, à calculer le hash de ladite requête de demande de signature, à créer un jeton de la demande, avec une durée de vie limitée, à utiliser une phrase secrète, partagée avec le gestionnaire de preuve, pour signer ledit jeton de la demande, puis modifier le statut de la demande de signature en 'démarré' et renvoyer le jeton signé de la demande à l'application métier.

[13] Selon des caractéristiques particulières, la page de consentement est générée sur le navigateur de l'utilisateur, à la demande de l'application métier qui transmet le jeton de la demande via le service de demande de signature, ladite page de consentement génère une paire de clé client (kcliPriv et kcliPub) puis s'interface avec le service d'identification de l'utilisateur, via un nouvel onglet sur le navigateur, une fois l'utilisateur authentifié par le fournisseur d'identité, celui-ci transmet un code à l'utilisateur, le gestionnaire de preuve demande ensuite au fournisseur d'identité un jeton d'identification en échange du code, selon le protocole OpenId Connect.

- [14] Selon des caractéristiques particulières, la page de consentement est servie par le service gestionnaire de preuve, de telle sorte que le gestionnaire de preuve renvoie la chaîne de certificat et la clé publique du serveur à savoir KservPub à la page de consentement, ladite page de consentement en utilisant la clé privé du client (Kclipriv) et la clé publique du serveur (KservPub) est apte à dériver la clé secrète Ks, le signataire étant le seul à disposer de ladite clé secrète Ks, nécessaire pour décomposer sa clé privée, Kpriv, via ladite page de consentement qui s'exécute dans son navigateur web.
- [15] Selon des caractéristiques particulières, le gestionnaire de preuve est apte à valider le jeton de la demande, à générer un ID de preuve et à créer un fichier de preuves, l'ID de preuve étant dérivé du jeton de demande de telle sorte que la création de preuves ne puisse pas être rejouée.
- [16] Selon des caractéristiques particulières, le service d'identification avec lequel s'interface la page de consentement est telle qu'il permet d'obtenir une identité substantielle ou renforcée de l'utilisateur et de créer un certificat qualifié.
- [17] Selon des caractéristiques particulières, la page de consentement est apte à récupérer la demande de signature et la liste des documents à signer du gestionnaire de demandes, la page de consentement présentant les documents au signataire, lui permettant de les lire avant de les signer.
- [18] Selon des caractéristiques particulières, le gestionnaire de demandes est apte à générer la valeur des documents à signer, DTBS, et à envoyer à la page de consentement en y attachant un code d'authentification, HMAC, ladite page de consentement est apte à envoyer les HMAC et la clé secrète Ks, et que le gestionnaire de preuve étant apte à valider les HMAC et décompresser la clé privée KPriv du signataire à l'aide de la clé secrète Ks, ledit gestionnaire de preuve signe ensuite les valeurs DTBS, avec la clé privé KPriv du signataire et ajoute les valeurs DTBS et la signature au fichier de preuves.
- [19] Selon des caractéristiques particulières, le fichier de preuves est exporté au format XML, ledit fichier étant signé par un certificat de sécurité et est envoyé à la page de consentement qui le transfert au gestionnaire de demandes pour la validation et le stockage dudit fichier de preuves.

- [20] Selon des caractéristiques particulières, le gestionnaire de demandes après avoir reçu le fichier de preuves au format XML, est apte à produire les documents signés de manière asynchrone et à informer l'application métier de ces documents, l'application métier étant apte à récupérer les documents signés et le fichier de preuves XML à l'aide d'une clé API, après avoir archivés le fichier XML et les documents signés, l'application métier étant apte à transmettre les documents signés au signataire.
- [21] Selon des caractéristiques particulières, le système comprend en outre une application native qui accède à la clé cryptographique du signataire, ladite application native pouvant s'exécuter sur l'ordinateur du signataire ou sur son smartphone, et permettant audit signataire de signer des documents avec son propre certificat en utilisant soit son ordinateur, soit une application mobile s'il souhaite signer sur son smartphone.
- [22] Selon des caractéristiques particulières, la page de consentement après avoir récupéré les documents à signer sur le gestionnaire de demandes et après avoir obtenu le consentement du signataire est apte à ouvrir l'application native soit sur l'ordinateur soit sur le smartphone du signataire, ladite application native étant apte à demander au signataire de choisir un certificat de signature, à récupérer le DTBS des documents à signer sur le gestionnaire de demandes, à envoyer le DTBS signé au gestionnaire de preuves, demander de produire le fichier de preuves, et à transférer le fichier de preuves au gestionnaire de demandes pour générer les documents signés de manière asynchrone.
- [23] L'invention concerne également un procédé d'élaboration et de traitement d'une demande de signature mis en œuvre dans le système ci-dessus, comprenant une application métier pour effectuer une demande de signature, un service de gestionnaire de demandes qui prend en compte la demande de signature depuis l'application métier, une page de consentement pour gérer le protocole de consentement, et un service gestionnaire de preuves pour générer et mettre à jour un fichier de preuves, le procédé étant caractérisé en ce qu'il permet la signature d'un ou plusieurs documents avec un certificat serveur et en ce qu'il comprend les étapes suivantes :

- lancement de l'application métier par un utilisateur pour signer un document ;
- lancement de la demande de signature par l'application métier auprès du gestionnaire de demandes puis transfère du ou des documents à signer et la redirection de l'utilisateur vers la page de consentement;
- identification du signataire par la page de consentement avec le fournisseur d'identité via le protocole OpenID Connect ;
- récupération de la demande de signature et le ou les documents à signer par la page de consentement auprès du gestionnaire de demande ;
- demande de consentement pour le ou les documents à signer par la page de consentement auprès du signataire ;
- génération de la clé de signature sur le composant serveur du gestionnaire de preuve, récupération du DTBS du ou des documents à signer auprès du gestionnaire de demandes par la page de consentement ;
- demande de signature de DTBS du ou des documents auprès du serveur et la production d'un fichier de preuves auprès de gestionnaire de preuve par la page de consentement ;
- transfère du fichier de preuves au gestionnaire de demande pour générer les documents signés de manière asynchrone ;
- redirection du signataire vers gestionnaire de demandes ;
- envoi du ou des documents signés et le fichier de preuves à l'application métier ;
- archivage du ou des documents signés et le fichier de preuves par l'application métier.

[24] Le procédé comprend en outre les étapes suivantes :

- validation de jeton de demande, génération d'un ID de preuve, et la création d'un fichier de preuves par le gestionnaire de preuves ;
- génération d'une paire de clés client KcliPriv, Kclipub par la page de consentement ;
- création de la clé du signataire par le gestionnaire de preuves ;
- génération d'une paires de clés privée et publique du serveur et l'utilisation de la clé privée du serveur et la clé publique du client pour dériver une clé

secrète Ks qui enveloppe la clé privée du signataire, par le gestionnaire de preuve;

- demande de certificat à l'autorité de certification et renvoie la chaîne de certificat et la clé publique du serveur à la page de consentement par le gestionnaire de preuves ;
- génération des valeurs DTBS de la demande et des documents par le gestionnaire de demandes ;
- transmission des valeurs DTBS par la page de consentement (43) au gestionnaire de preuves ;
- signature des valeurs DTBS avec la clé privée du signataire et ajout les valeurs DTBS et la signature au fichier de preuves ;
- export du fichier de preuves au format XML et renvoie à la page de consentement qui le transmet ensuite au gestionnaire de demandes ;
- production des documents signés et envoie desdits documents à l'application métier ;
- archivage du fichier de preuves au format XML et les documents signés par l'application métier et ensuite le transfert des documents signés au signataire.

[25] L'invention concerne également le procédé d'élaboration et de traitement d'une demande de signature mis en œuvre dans le système ci-dessus, et comprenant en outre une application native qui accède à une clé cryptographique du signataire. Le procédé, après la récupération des documents à signer sur le gestionnaire de demandes et après avoir obtenu le consentement du signataire, comprend les étapes suivantes :

- ouverture de l'application native soit sur l'ordinateur du signataire soit sur son smartphone ;
- demande de choix d'un certificat de signature par l'application native ;
- récupération du DTBS des documents à signer sur le gestionnaire de demandes, puis envoi le DTBS signé au gestionnaire de preuves par l'application native;
- demande de production du fichier de preuves, et le transfère du fichier de preuves au gestionnaire de demandes pour générer les documents signés de manière asynchrone ;

- fermeture de l'application native et la redirection du signataire vers l'application métier;
- récupération des documents signés par l'application métier;
- récupération du fichier de preuves et archivage des fichiers ainsi que les documents signés par l'application métier.

[26] BREVE DESCRIPTION DES FIGURES

[27] D'autres caractéristiques, détails et avantages de l'invention ressortiront à la lecture de la description qui suit, en référence aux figures annexées, qui illustrent :

[Fig.1] illustre l'architecture générale du système selon la présente invention;

[Fig.2] représente un diagramme simplifié du serveur de signature avec un certificat crée à la volé selon la présente invention ;

[Fig.3] illustre l'architecture du système dans le cas d'utilisation d'un jeton d'authentification et signature locale ;

[Fig.4] représente un diagramme simplifié dans le cas de signature locale.

[28] Pour plus de clarté, les éléments identiques ou similaires sont repérés par des signes de références identiques sur l'ensemble des figures.

[29] DESCRIPTION DETAILLEE

[30] L'architecture générale du système selon la présente invention représente d'une part, un utilisateur 30 qui doit signer un document et d'autre part l'environnement internet d'un gestionnaire de demande de signature. Un utilisateur 30 est une personne physique qui souhaite ou doit signer un ou plusieurs documents.

[31] La distinction entre une signature effectuée à l'initiative même de l'utilisateur ou bien sollicitée par un tiers (autre utilisateur) est essentielle. En effet, l'expérience utilisateur est très différente car, dans le premier cas, celle-ci implique nécessairement une préparation liée au choix du document, à sa rédaction, à la sélection de l'identité numérique et à sa mise en place, à l'éventuelle politique de signature à appliquer, etc., alors que dans le deuxième cas, elle exige une facilité d'action particulière quant à l'accès au document et à

l'identité numérique du signataire pour se focaliser sur la valeur probatoire de la transaction, en contraignant éventuellement l'utilisateur, avant de pouvoir signer, à lire l'intégralité du document, à s'authentifier pour prouver son identité numérique, etc.

- [32] L'architecture du système telle que représentée sur la figure 1 comprend une application métier 10, ladite application métier 10 peut être développée et exécutée dans des environnements variés tels que des serveurs Web, des navigateurs Internet, dans un environnement natif PC ou Mac, ou encore depuis un téléphone portable ou une tablette. L'application métier est à l'origine du processus de signature, ainsi, toute demande de signature, qu'elle soit effectuée à l'initiative de l'utilisateur signataire 30 lui-même, ou qu'elle soit effectuée par un tiers en vue de faire signer un document, doit nécessairement passer par cette application métier 10. Ladite application métier 10 est conçue de sorte qu'elle est apte à effectuer une demande de signature d'au moins d'un document auprès d'un service gestionnaire de demandes 41, ledit gestionnaire de demandes 41 est un composant serveur qui prend en compte une demande de signature venant de ladite application métier 10 et il est hébergé par une entité demandeur 40. Le système comprend en outre deux autres services indépendants à savoir un service gestionnaire ou une page de consentement 43 et un service gestionnaire de preuves 45. La page de consentement 43 gère le protocole de consentement et est une application HTML qui pilote le signataire 30 tout au long du processus de consentement et de la création de la signature. Ce composant réside dans le navigateur Internet 11 du signataire 30. Le service gestionnaire de preuves 45 est un composant serveur qui collecte les preuves, ce composant est hébergé par le fournisseur de signature 50 (LEXPERSONA). La référence 48 est un composant serveur hébergeant la clé cryptographique du signataire 30. Ce composant est intégré au gestionnaire de preuves 45. L'autorité de certification 46 est une infrastructure de clé publique PKI qui émet le certificat de signataire, ce composant est hébergé par le fournisseur de signature 50. Le fournisseur d'identité 44 est un serveur d'autorisation conforme à Open ID Connect qui émet le jeton d'identification d'un signataire à partir duquel le certificat de signataire est dérivé. Ce composant est hébergé par un fournisseur d'enregistrement 60.

[33] La figure 2 illustre le diagramme simplifié du processus de signature d'un document à distance. Le signataire 30 navigue avec son navigateur internet 11 sur l'application métier 10 où il doit signer certains documents. L'application métier 10 lance une demande de signature sur le gestionnaire de demandes 41 et lui transfère les documents à signer. Le service gestionnaire de demandes 41 redirige le signataire 30 vers la page de consentement 43. Ledit page de consentement 43 identifie le signataire 30 en s'interfaçant avec un fournisseur d'identité 44 à l'aide du protocole Open ID Connect. Ensuite, la page de consentement 43 récupère la demande de signature ainsi que les documents à signer sur le gestionnaire de demandes 41. La page de consentement 43 montre les documents à signer au signataire 30 en lui demandant son consentement. La page de consentement demande la génération de la clé de signataire sur le composant serveur 48 qui est intégré au composant gestionnaire de preuves 45. La page de consentement 43 récupère le DTBS des documents à signer sur le gestionnaire de demandes 41 et demande au serveur 48 de signer le DTBS et au gestionnaire de preuves 45 de produire le fichier de preuves, pour transférer au gestionnaire de demandes 41 afin que ledit gestionnaire de demandes 41 puisse générer les documents signés de manière asynchrones. Ensuite, la page de consentement 43 redirige le signataire vers l'application métier 10. Une fois les documents signés produites par le gestionnaire de demandes 41, l'application métier 10 les récupère. L'application métier 10 récupère aussi le fichier de preuves et l'archive avec les documents signés.

[34] Nous allons décrire comment une application métier 10 fournit une demande de signature sur le gestionnaire de demandes 41. En effet, l'application métier 10 lance la création d'une demande de signature auprès du gestionnaire de demandes 41 qui est protégé par une clé API. L'application métier 10 est la seule application apte à faire une demande de signature auprès du gestionnaire de demandes 41. Il est à noter que, l'application métier 10 communique avec l'API REST du gestionnaire de demandes 41 en arrière plan afin que la clé de l'API reste secrète. Le gestionnaire de demandes 41 génère ensuite un ID de demande de signature et l'envoi à l'application métier 10, ladite application métier 10 télécharge un document à signer dans le gestionnaire de demandes 41 en transmettant l'ID de demande en tant que paramètre. De même, le gestionnaire

de demandes 41 génère un ID unique du document et le renvoi à l'application métier 10. Les étapes précédentes sont répétées autant de fois qu'il y ait de document à signer. Une fois les documents téléchargés, l'application métier 10 demande au gestionnaire de demandes 41 de lancer la demande de signature.

- [35] Le gestionnaire de demandes 41 calcul le hachage de demande ainsi que les hachages des documents à partir du JSON de demande et des JSON des documents (incluant les hachages de documents). Le JSON de demande incluant un ID de demande. Le gestionnaire de demandes 41 crée également le jeton de demande contenant l'ID de demande et son hachage et utilise une phrase secrète partagée avec le gestionnaire de preuves 45 pour le signer. En suite, le gestionnaire de demandes 41 modifie le statut de la demande en démarré et renvoie le jeton de la demande à l'application métier 10. Le jeton de la demande a une durée de vie de cinq minutes. Une durée plus courte, poserait des problèmes si les horloges du gestionnaire de demandes 41 et le gestionnaire de preuves 45 ne sont pas correctement synchronisées.

Création d'une page de consentement :

- [36] Le signataire est redirigé vers l'URL de la page de consentement 43 par l'application métier 10, en transmettant le jeton de la demande en tant que paramètre. Le gestionnaire de preuves 45, derrière l'URL de la page de consentement 43 valide le jeton de la demande, génère un ID de preuve et crée le fichier de preuves, en renvoyant une redirection HTTP avec l'ID de preuve en tant que paramètre. L'ID de preuve est dérivé du jeton de la demande afin que la création de preuve ne puisse pas être rejouée. En effet, la redirection empêche la tentative de création de preuves à chaque fois que le signataire actualise l'onglet de son navigateur. En suite, le signataire ouvre l'URL de la page de consentement 43 en transmettant l'ID de preuve en tant que paramètre. Le gestionnaire de preuves 45 renvoie une page HTML comprenant l'ID de la demande, le hachage de la demande, l'URL d'autorisation du fournisseur d'identité 44, les feuilles de style, etc.. à la page de consentement 43.

Récupération de l'identité du signataire et le déroulement d'autorisation Open ID Connect :

[37] Une fois la page de consentement 43 signifiée, le signataire 30 la charge dans son navigateur 11. Tout d'abord, la page de consentement génère une paire de clés publique et privée du client (KcliPriv et KcliPub). L'URL autorisé est ensuite renseignée avec un état contenant l'ID de preuve, un identifiant aléatoire unique pour un ou plusieurs documents et la clé publique du client. Une redirection vers le fournisseur d'identité 44 aura lieu via un bouton présenté au signataire 30. Le bouton permet également à la page d'autorisation de s'ouvrir sans aucune restriction du navigateur. Une fois le bouton cliqué, l'URL d'autorisation du fournisseur d'identité 44 est ouverte dans un nouvel onglet de navigateur 11, laissant ainsi la page de consentement 43 en arrière plan. Le fait de garder la page de consentement 43 ouvert permet de conserver la clé privée du client (KcliPriv) dans la RAM et de limiter les risques d'attaque sur l'ordinateur du signataire 30. Une fois le signataire 30 s'est authentifié et a accordé l'autorisation d'accéder à son identité, le fournisseur d'identité 44 le redirige vers une page de rappel en lui transmettant un code. Le navigateur du signataire 30 ouvre la page de rappel en transmettant l'ID de preuve en tant que paramètre avec le code. Le gestionnaire de preuves 45 demande au fournisseur d'identité 44 d'échanger le code avec un jeton d'identification tel que spécifié par le protocole Open ID Connect. Le fournisseur d'identité 44 renvoie le jeton d'identification au gestionnaire de preuves 45. Ledit gestionnaire de preuves 45 crée le jeton de signataire à partir du jeton ID et met à jour le fichier de preuves et une page HTML est renvoyée, y compris le jeton de signataire. A noter que le jeton de signataire contient le hachage de la demande et certains éléments de base fournie par le fournisseur d'identités 44, telle que le nom du signataire, l'adresse de la messagerie et l'identificateur. Une fois ouvert par le navigateur 11 du signataire, la page de rappel envoie le jeton de signataire à la page de consentement 43, à l'aide d'un message inter-onglet. Puisque les deux pages sont sur le même domaine, il n'y a pas de restriction du navigateur.

Consentement du signataire :

[38] La page de consentement 43 récupère la demande de signature du gestionnaire de demandes 41 en transmettant l'ID de la demande et le jeton de signataire en tant que paramètre. Le gestionnaire de demandes 41 renvoie le JSON de la demande après avoir vérifié que le jeton de signataire contient les

éléments d'identité attendus ainsi que le hachage de la demande attendu. La page de consentement 43 récupère la liste des documents du gestionnaire de demandes 41 en transmettant l'ID de la demande et le jeton de signataire en tant que paramètre. Le gestionnaire de demandes 41 renvoie les JSON des documents après avoir vérifié que le jeton de signature contient les éléments d'identité et le hachage de la demande prévu. La page de consentement 43 garantit que le hachage de la demande est cohérent avec le JSON de la demande et les JSON des documents. Pour chaque document que le signataire doit ou veut lire ou télécharger, la page de consentement 43 récupère le contenu du document en transmettant comme paramètre l'ID du document et le jeton de signataire. Le gestionnaire de demandes 41 renvoie le contenu du document après avoir vérifié que le jeton de signataire contient les éléments d'identité attendues et le hachage de demande prévu. Avant de présenter le contenu du document au signataire, la page de consentement 43 garantit que le hachage du contenu est cohérent avec le hachage contenu dans le document JSON. En suite la page de consentement 43 présente les documents au signataire 30, lui permettant de les lire. Une fois qu'il accepte, le signataire clique sur le bouton «J'accepte».

Création de la clé de signataire

[39] La clé du signataire est créée et protégée par le gestionnaire de preuves 45. En effet, la page de consentement 43 demande au gestionnaire de preuves 45 de créer la clé de signataire 30, en passant l'ID de preuve, le jeton de signataire, le JSON de demande, les JSON de documents et la clé publique de client (kcliPub) en tant que paramètres. Le gestionnaire de preuves 45 s'assure que le hachage de la demande est cohérent avec le JSON de la demande et avec les JSON des documents. Le gestionnaire de preuves 45 vérifie également que le nombre aléatoire unique contenu dans le jeton d'ID produit par le fournisseur d'identité 44 correspondant au hachage du défi et à la clé publique du client (kcliPub). Ensuite, le gestionnaire de preuves 45 génère une paire de clés serveur (kservPriv et kservPub) et utilise la clé privée du serveur (kservPriv) et la clé publique du client (kcliPub) pour dériver une clé secrète (ks) avec l'algorithme Diffie-Hellman. La paire de clés (kPriv et kPub) du signataire est ensuite créée par le gestionnaire de preuves 45 avec une demande de certificat, et la clé

secrète (ks) est utilisée pour envelopper la clé privée (kPriv) du signataire. Une fois que la clé du signataire a été encapsulée, la clé secrète (ks) est détruite par le gestionnaire de preuves 45. La demande de certificat est envoyée à l'autorité de certification 46. L'autorité de certification 46 renvoie le certificat au gestionnaire de preuve 45. Après avoir stocké la clé privée encapsulée et mis à jour le fichier de preuves avec le certificat de signataire, le gestionnaire de preuves 45 renvoie la chaîne de certificats et la clé publique du serveur (kservPub) à la page de consentement 43. La page de consentement 43 utilise la clé privée du client (kcliPriv) et la clé publique du serveur (kservPub) pour dériver la clé secrète (ks) avec l'algorithme Diffie-Hellman. À ce stade, le signataire 30, via la page de consentement 43 qui s'exécute dans son propre navigateur 11, est le seul à disposer de la clé secrète (ks) nécessaire pour décompresser la clé privée du signataire (kPriv) dans le gestionnaire de preuves.

[40] Nous décrivons comment les documents DTBS sont générés par le gestionnaire de demandes 41 et comment ils sont signés par le gestionnaire de preuves 45 à l'aide de la clé du signataire 30. En effet, pour chaque document à signer, la page de consentement 43 envoie la chaîne de certificats au gestionnaire de demandes 41 avec l'ID de document et le jeton de signataire. Après avoir vérifié que le jeton de signataire contient les éléments d'identité attendus et le hachage de la demande, le gestionnaire de demandes 41 génère la valeur DTBS et la renvoie à la page de consentement 43 en y attachant un HMAC. Le HMAC est calculé à partir de la valeur DTBS et de l'ID de document à l'aide de la phrase secrète partagée avec le gestionnaire de preuves 45. Une fois que toutes les valeurs DTBS ont été générées, la page de consentement 43 les envoie au gestionnaire de preuves 45 avec les HMAC et la clé secrète (ks). Le gestionnaire de preuves 45 valide les HMAC et décompresse la clé privée du signataire (kPriv) avec la clé secrète (ks). Le gestionnaire de preuves 45 signe ensuite les valeurs DTBS avec la clé privée du signataire (kPriv) et ajoute les valeurs DTBS et la signature au fichier de preuves. Le fichier de preuves est exporté au format XML, signé par le gestionnaire de preuves 45 avec un certificat de scellement et est renvoyé à la page de consentement 43. Ladite page de consentement 43 transmet le fichier de preuves XML au gestionnaire de demandes 41. Le gestionnaire de demandes 41 valide et stocke le fichier de

preuves XML, modifie le statut de la demande en 'signé' et renvoie une réponse vide à la page de consentement 43.

[41] Le gestionnaire de demandes 41 termine le traitement de la demande en produisant les documents signés en arrière-plan. En effet, après avoir reçu le fichier de preuves XML, le gestionnaire de demandes 41 lance une tâche en arrière-plan pour produire les documents signés et terminer le traitement de la demande. Pendant ce temps, la page de consentement 43 redirige le signataire 30 vers l'URL de rappel de l'application métier 10. Le navigateur du signataire 30 ouvre la page de rappel de l'application métier 10. Selon le cas d'utilisation de l'application métier 10, la page de rappel peut demander au signataire 30 d'attendre pendant la préparation des documents signés ou peut l'informer que les documents signés seront disponibles ultérieurement. Une fois que le gestionnaire de demandes 41 a produit les documents signés, le statut de la demande est modifié pour être terminé. Le gestionnaire de demandes 41 informe l'application métier 10 via un Webhook. L'application métier 10 récupère les documents signés à partir du gestionnaire de demandes 41 à l'aide de sa clé API. Le gestionnaire de demandes 41 renvoie le fichier de preuves XML à l'application métier 10. Une fois le fichier de preuve XML et les documents signés archivés, l'application métier 10 transmet les documents signés au signataire 30.

[42] A noter que un Webhook est une méthode permettant d'augmenter ou de modifier le comportement d'une page Web , ou d'une application Web , avec des rappels personnalisés.

[43] L'invention permet également à un signataire de signer des documents avec son propre certificat, un jeton cryptographique, par exemple, en utilisant une application locale s'il souhaite signer sur son ordinateur ou s'il souhaite utiliser son smartphone. Les différents composants impliqués dans la création de la signature locale sont illustrés sur la figure 3. Cette architecture comprend une application métier 10 qui est chargée de créer les demandes de signature. Ce composant est hébergé par l'entité demandeur 40. Le gestionnaire de demandes 41 est un composant du serveur qui héberge les demandes de signature et les documents joints. Ce composant est hébergé par l'entité demandeur 40. La page de consentement 43 est une application HTML qui pilote le signataire 30 tout au

long du processus de consentement et de la création de la signature. Ce composant réside dans le navigateur Web 11 du signataire 30. Le gestionnaire de preuves 45 est un composant serveur qui collecte les preuves de signature. Ce composant est hébergé par le fournisseur de signature 50 (LEX PERSONA). Le système comprend en outre une application native locale 47 qui peut être installée sur un ordinateur fixe du bureau ou sur un smartphone et qui accède à la clé cryptographique du signataire pour signer les documents. Ce composant s'exécute sur l'ordinateur du signataire ou sur son smartphone.

[44] La figure 4 illustrant un flux simplifié de ce mode de signature locale. Dans ce mode de signature locale, une fois que la page de consentement 43 montre les documents au signataire 30 et demande son consentement, ledit page de consentement 43 ouvre l'application locale bureau/mobile, ladite application locale 47 bureau/mobile demande au signataire 30 de choisir un certificat de signature. L'application locale bureau/mobile récupère le DTBS des documents à signer sur le gestionnaire de demandes 41 et ensuite envoie le DTBS signé au gestionnaire de preuves 45 et lui demande de produire le fichier de preuves. Le fichier de preuves est transféré au gestionnaire de demandes 41 pour générer les documents signés de manière asynchrones. Ensuite, la page de consentement 43 redirige le signataire 30 vers l'application métier 10. Le signataire accède à l'application métier 10, et une fois que les documents signés ont été produits par le gestionnaire de demandes 41, l'application métier 10 les récupère. L'application métier 10 récupère le fichier de preuves et l'archive avec les documents signés.

Ouverture de l'application locale bureau/Mobile :

[45] Le signataire 30 passe de la page de consentement 43 à l'application locale 47 bureau/mobile afin de signer avec son jeton cryptographique. Dans ce mode de signature locale, la page de consentement 43 demande du signataire 30 à installer l'application locale 47 sur son bureau ou sur son smartphone ou à l'ouvrir si l'application est déjà installée en transmettant comme paramètres l'ID de demande, l'ID de preuve et un ID d'instance aléatoire. Si la page de consentement 43 est ouverte sur un navigateur Web du bureau, le signataire peut passer à application Mobile sur son smartphone à l'aide d'un code QR.

Sinon, l'application locale bureau/mobile est ouverte via un URL personnalisé. Une fois ouverte, l'application locale bureau/mobile appelle le gestionnaire de demandes 41 pour définir l'ID d'instance correspondant à l'ID de demande spécifié. Le but est que la page de consentement 43 soit notifiée lorsque l'application locale 47 bureau/mobile a été ouverte. Le gestionnaire de demandes 41 renvoie le JSON de la demande. L'application locale bureau/mobile récupère la liste de documents du gestionnaire de demandes 41 en transmettant l'ID de demande en tant que paramètre. Le gestionnaire de demandes 41 renvoie les JSON des documents. Le signataire 30 est invité à choisir un certificat de signature et ensuite le signataire clique sur le bouton "Signer".

- [46] Dans ce mode de signature locale, le gestionnaire de preuves 45 renvoie la chaîne de certificats à l'application locale bureau/mobile. Pour chaque document à signer, l'application locale bureau/mobile envoie la chaîne de certificats au gestionnaire de demandes 41 avec l'ID du document. Le gestionnaire de demandes 41 génère la valeur DTBS et la renvoie à la page de consentement 43. Une fois que toutes les valeurs DTBS ont été générées, l'application locale, bureau/mobile, les signe à l'aide de la clé du signature et les envoie au gestionnaire de preuves 45 avec les HMAC et les valeurs de signature. Le gestionnaire de preuves 45 valide les HMAC et les valeurs de signature, puis ajoute le DTBS et les valeurs de signature au fichier de preuves. Le fichier de preuves est exporté au format XML, signé par le gestionnaire de preuves 45 avec un certificat de scellement et renvoyé à l'application locale, bureau/mobile. La page de consentement 43 transmet le fichier de preuves XML au gestionnaire de demandes 41. Le gestionnaire de demandes 41 valide et stocke le fichier de preuves, modifie le statut de la demande en signé et renvoie une réponse vide à la page de consentement 43. Ensuite, l'application locale, bureau/mobile se ferme et ramène le signataire 30 à la page de consentement 43. |

Revendications

[Revendication 1] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents comprenant une application métier (10), chargée de créer une demande de signature d'au moins un document pour un utilisateur signataire (30), ladite application métier (10) étant hébergée par une entité demandeur (40), le système est caractérisé en ce qu'il permet de signer des documents avec un certificat serveur créé à la volée et en ce qu'il comprend en outre un service gestionnaire de demandes (41) qui héberge les demandes de signatures et les documents joints; une page de consentement (43), générée sur le navigateur de l'utilisateur à la demande de l'application métier (10) ; et un service gestionnaire de preuves (45), hébergé par un fournisseur de signature (50), collectant les preuves de signature ; la page de consentement (43) constituée d'une application HTML étant apte à gérer le protocole de consentement en générant une paire de clés client, KcliPub, KcliPriv et en s'interfaçant avec un service sécurisé d'identification (44) de l'utilisateur (30), via un nouvel onglet sur le navigateur ; une fois l'utilisateur authentifié par le fournisseur d'identité (44), celui-ci transmet un code à l'utilisateur (30), le gestionnaire de preuves (45) fait la demande d'un jeton d'identification, en échange du code, auprès de fournisseur d'identité (44), selon le protocole OpenId Connect, ledit service gestionnaire de preuves (45) comprend en outre un composant serveur hébergeant la clé cryptographique du signataire (30), et étant apte à générer une paire de clés serveur, KservPriv et KservPub, à dériver une clé secrète Ks en utilisant la clé publique du client, Kclipub et la clé privée du serveur, KservPriv, à générer une clé publique KPub et une clé privée, KPriv du signataire (30), à créer une demande de certificat, à récupérer un certificat auprès d'une autorité de certification (46) ; que la clé secrète Ks étant utilisée pour envelopper la clé privée du signataire en renvoyant la chaîne de certificat et la clé publique du serveur, KservPub à la page de consentement (43), ladite page de consentement en utilisant la clé privé du client Kclipriv, et la clé publique du serveur, KservPub est apte à dériver la clé secrète Ks, et que le signataire (30) est le seul à disposer de ladite clé secrète Ks,

nécessaire pour décomposer sa clé privée Kpriv via ladite page de consentement (43) qui s'exécute dans son navigateur web, le gestionnaire de preuves (45) étant apte à décompresser la clé privée du signataire kPriv, à l'aide de la clé secrète ks, pour signer la valeur DTBS, documents à signer, avec la clé privée du signataire kPriv, et à ajouter la valeur DTBS et la signature au fichier de preuves.

[Revendication 2] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents selon la revendication 1 caractérisé en ce que le service gestionnaire de demandes (41) est protégé par une clé API de telle sorte que seule l'application métier (10) est apte à lui faire des demandes de signature et en ce que ledit service gestionnaire de demandes (41) étant apte à générer une requête pour la demande de signature, puis à générer un identifiant aléatoire unique pour un ou plusieurs documents de la demande de signature et à calculer le hash du ou desdits documents, à calculer le hash de ladite requête de demande de signature, à créer un jeton de la demande, avec une durée de vie limitée, à utiliser une phrase secrète, partagée avec le gestionnaire de preuves (45), pour signer ledit jeton de la demande, puis modifier le statut de la demande de signature en démarré et renvoyer le jeton signé de la demande à l'application métier (10).

[Revendication 3] Système ouvert et sécurisé de demandes de signature électronique de document selon l'une des revendications précédente caractérisé en ce que le gestionnaire de preuves (45) est apte à valider le jeton de la demande, à générer un ID de preuve et à créer un fichier de preuve, l'ID de preuve étant dérivé du jeton de la demande de telle sorte que la création de preuve ne puisse pas être rejouée.

[Revendication 4] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents selon la revendication 3 caractérisé en ce que le service d'identification (44) avec lequel s'interface la page de consentement (43) est de telle qu'il permet d'obtenir une identité substantielle ou renforcée de l'utilisateur (30) et de créer un certificat qualifié.

[Revendication 5] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents selon l'une des revendications précédente, caractérisé en ce que la page de consentement (43) est apte à

recupérer la demande de signature et la liste des documents à signer du gestionnaire de demandes (41), la page de consentement présentant les documents au signataire (30), lui permettant de les lire avant de les signer.

[Revendication 6] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents selon l'une des revendications précédente, caractérisé en ce le gestionnaire de demandes (41) est apte à générer la valeur des documents à signer, DTBS, et à envoyer à la page de consentement (43) en y attachant un code d'authentification, HMAC, ladite page de consentement (43) est apte à envoyer les HMAC et la clé secrète Ks, et que le gestionnaire de preuve (45) étant apte à valider les HMAC et décompresser la clé privée KPriv du signataire à l'aide de la clé secrète Ks, ledit gestionnaire de preuve (45) signe ensuite les valeurs DTBS, avec la clé privé KPriv du signataire et ajoute les valeurs DTBS et le signature au fichier de preuves.

[Revendication 7] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents selon la revendication précédente dans lequel le fichier de preuves est exporté au format XML, ledit fichier étant signé par un certificat de sécurité et envoyé à la page de consentement (43) qui le transfert au gestionnaire de demandes (41) pour la validation et le stockage dudit fichier de preuves.

[Revendication 8] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents selon la revendication précédente dans lequel, le gestionnaire de demandes (41), après avoir reçu le fichier de preuve au format XML, est apte à produire les documents signés et à informer l'application métier (10) de ces documents, l'application métier (10) étant apte à récupérer les documents signés et le fichier de preuve XML à l'aide d'une clé API, et dans lequel après avoir archivés le fichier XML et les documents signés, l'application métier (10) étant apte à transmettre les documents signés au signataire (30).

[Revendication 9] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents selon la revendication 1 caractérisé en ce que le système comprend en outre une application native (47) qui accède à la clé cryptographique du signataire (30), ladite application native

(47) pouvant s'exécuter soit sur l'ordinateur du signataire (30) soit sur son smartphone, et permettant audit signataire (30) de signer des documents avec son propre certificat en utilisant soit son ordinateur, soit une application mobile s'il souhaite signer sur son smartphone.

[Revendication 10] Système ouvert et sécurisé de traitement de demandes de signature électronique de documents selon la revendication 11, caractérisé en ce que, la page de consentement (43) après avoir récupéré les documents à signer sur le gestionnaire de demandes (41) et après avoir obtenu le consentement du signataire (30) est apte à ouvrir l'application native (47) soit sur l'ordinateur bureau ou sur le smartphone du signataire (30), ladite application native (47) étant apte à demander au signataire (30) de choisir un certificat de signature, à récupérer le DTBS des documents à signer sur le gestionnaire de documents (41), à envoyer le DTBS signé au gestionnaire de preuves (45), demander de produire le fichier de preuves, et à transférer le fichier de preuves au gestionnaire de demandes (41) pour générer les documents signés de manière asynchrone.

[Revendication 11] Procédé d'élaboration et de traitement d'une demande de signature mis en œuvre dans le système comprenant une application métier (10) pour effectuer une demande de signature, un service de gestionnaire de demandes (41) qui prend en compte la demande de signature depuis l'application métier (10), une page de consentement (43) pour gérer le protocole de consentement, et un service gestionnaire de preuves (45) pour générer et mettre à jour un fichier de preuves, selon l'une des revendication 1 à 10, le procédé étant caractérisé en ce qu'il permet la signature d'un ou plusieurs documents avec un certificat serveur et en ce qu'il comprend les étapes suivante :

- lancement de l'application métier (10) par un utilisateur (30) pour signer un document (20) ;
- lancement de la demande de signature par l'application métier (10) auprès du gestionnaire de demandes (41) puis transfère du ou des documents à signer et la redirection de l'utilisateur (30) vers la page de consentement (43);
- identification du signataire (30) par la page de consentement (43) avec le fournisseur d'identité via le protocole OpenID Connect ;

- récupération de la demande de signature et le ou les documents à signer par la page de consentement (43) auprès du gestionnaire de demandes (41) ;
- demande de consentement pour le ou les documents à signer par la page de consentement (43) auprès du signataire (30) ;
- génération de la clé de signature sur le composant serveur (48) du gestionnaire de preuve (45), récupération du DTBS du ou des documents à signer auprès du gestionnaire de demandes (41) par la page de consentement (43) ;
- demande de signature de DTBS du ou des documents auprès du serveur (48) et la production d'un fichier de preuves auprès de gestionnaire de preuves (45) par la page de consentement (43) ;
- transfère du fichier de preuves au gestionnaire de demandes (41) pour générer les documents signés de manière asynchrone ;
- redirection du signataire (30) vers gestionnaire de demandes (41) ;
- envoi du ou des documents signés et le fichier de preuves à l'application métier (10) ;
- archivage du ou des documents signés et le fichier de preuves par l'application métier (10).

[Revendication 12] Procédé d'élaboration et de traitement de demandes de signature selon la revendication précédente caractérisé en ce qu'il comprend en outre les étapes suivantes :

- validation de jeton de demande, génération d'un ID de preuve, et la création d'un fichier de preuves par le gestionnaire de preuves (45) ;
- génération d'une paire de clés client KcliPriv, Kclipub par la page de consentement (43) ;
- création de la clé du signataire par le gestionnaire de preuves (45) ;
- génération par le gestionnaire de preuves (45) d'une paires de clés privée et publique du serveur et l'utilisation de la clé privée du serveur et la clé publique du client pour dériver une clé secrète Ks qui enveloppe la clé privée du signataire (30);
- demande de certificat à l'autorité de certification et renvoie la chaine de certificat et la clé publique du serveur à la page de consentement (43) par le gestionnaire de preuve (45) ;

- génération des valeurs DTBS de la demande et des documents par le gestionnaire de demandes (41) ;
- transmission des valeurs DTBS par la page de consentement (43) au gestionnaire de preuves (45) ;
- signature des valeurs DTBS avec la clé privée du signataire (30) et ajout les valeurs DTBS et la signature au fichier de preuves ;
- export du fichier de preuves au format XML et renvoie à la page de consentement (43) qui le transmet ensuite au gestionnaire de demandes (41) ;
- production des documents signés et envoi desdits documents à l'application métier (10) ;
- archivage du fichier de preuves au format XML et les documents signés par l'application métier (10) et ensuite le transfert des documents signés au signataire (30).

[Revendication 13] Procédé d'élaboration et de traitement d'une demande de signature mis en œuvre dans le système des revendications 11 et 12, comprenant une application métier (10) pour effectuer une demande de signature, un service de gestionnaire de demandes (41) qui prend en compte la demande de signature depuis l'application métier (10), une page de consentement (43) pour gérer le protocole de consentement, un service gestionnaire de preuves (45) pour générer et mettre à jour un fichier de preuves, une application native (47) qui accède à une clé cryptographique du signataire (30), caractérisé en ce que le procédé après la récupération des documents à signer sur le gestionnaire de demandes (41) et après avoir obtenu le consentement du signataire (30) comprend les étapes suivantes :

- ouverture de l'application native (47) soit sur l'ordinateur du signataire (30) ou sur son smartphone ;
- demande de choix d'un certificat de signature par l'application native (47) ;
- récupération du DTBS des documents à signer sur le gestionnaire de documents (41), puis envoi le DTBS signé au gestionnaire de preuve (45) par l'application native (47) ;
- demande de production du fichier de preuves, et le transfère du fichier de preuves au gestionnaire de demandes (41) pour générer les documents signés de manière asynchrone par l'application native (47) ;

- fermeture de l'application native (47) et la redirection du signataire (30) vers l'application métier (10);
- récupération des documents signés par l'application métier (10) ;
- récupération du fichier de preuves et archivage des fichiers ainsi que les documents signés par l'application métier (10).

Abrégé

Système ouvert et sécurisé de traitement de demande de signature électronique et procédé associé

L'invention concerne un système ouvert et sécurisé de traitement de demandes de signature électronique de documents comprenant une application métier, chargée de créer une demande de signature d'au moins un document pour un utilisateur signataire, ladite application métier étant hébergée par une entité demandeur. Le système est caractérisé en ce qu'il permet de signer des documents avec un certificat serveur créé à la volée et en ce qu'il comprend en outre trois services indépendants, un service gestionnaire de demandes qui héberge les demandes de signatures et les documents joints; une page de consentement, constitué d'une application HTML, étant apte à gérer le protocole de consentement en s'interfaçant avec un service sécurisé d'identification de l'utilisateur et en générant une paire de clés client (KcliPub, KcliPriv) et un service gestionnaire de preuves, hébergé par un fournisseur de signature, collectant les preuves de signature et comprenant un composant serveur hébergeant la clé cryptographique du signataire, ledit service gestionnaire de preuves étant apte à générer une paire de clés serveur (KservPriv et KservPub) à dériver une clé secrète Ks en utilisant la clé publique du client (Kclipub) et la clé privée du serveur (KservPriv) ; à générer une clé publique (KPub) et une clé privée, (KPriv) du signataire, à créer une demande de certificat, à récupérer un certificat auprès d'une autorité de certification, et en ce que la clé secrète Ks étant utilisée pour envelopper la clé privée du signataire. L'invention concerne en outre les procédés de traitement et d'élaboration de signature électronique de document mis en œuvre dans le système ci-dessus.

Figure pour abrégé : Figure 1

[Fig.1]

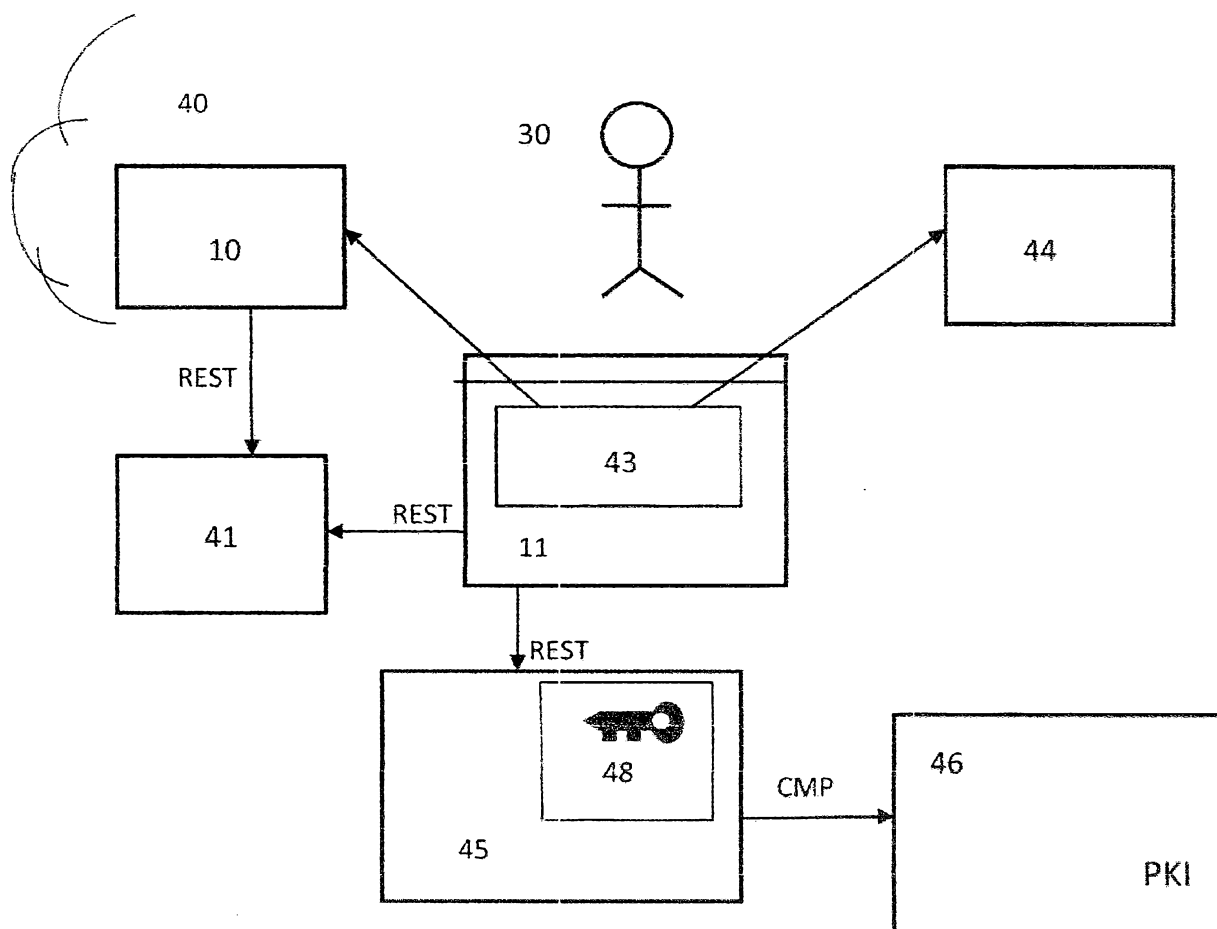


FIG.1

[Fig.2]

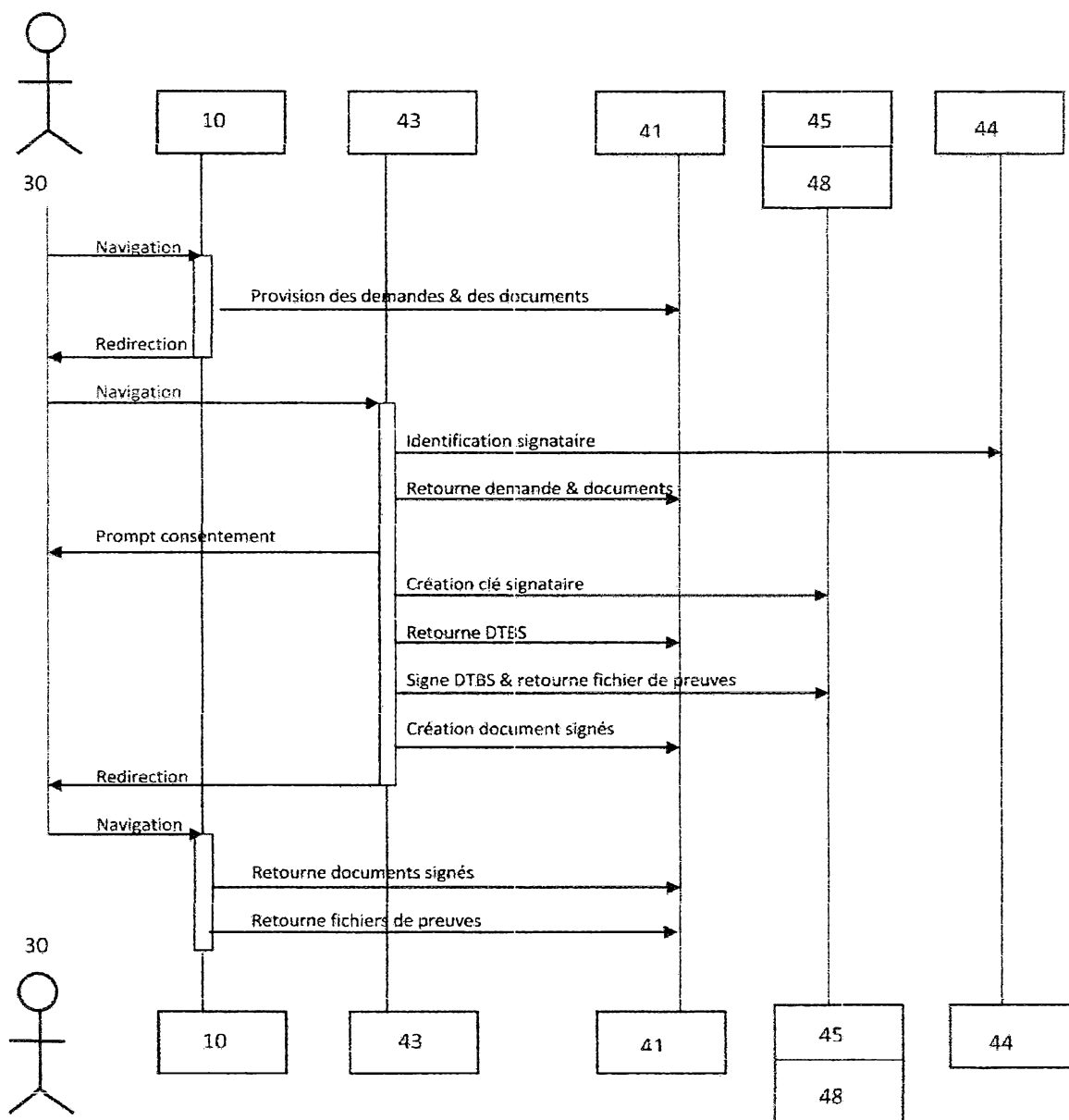


FIG.2

[Fig.3]

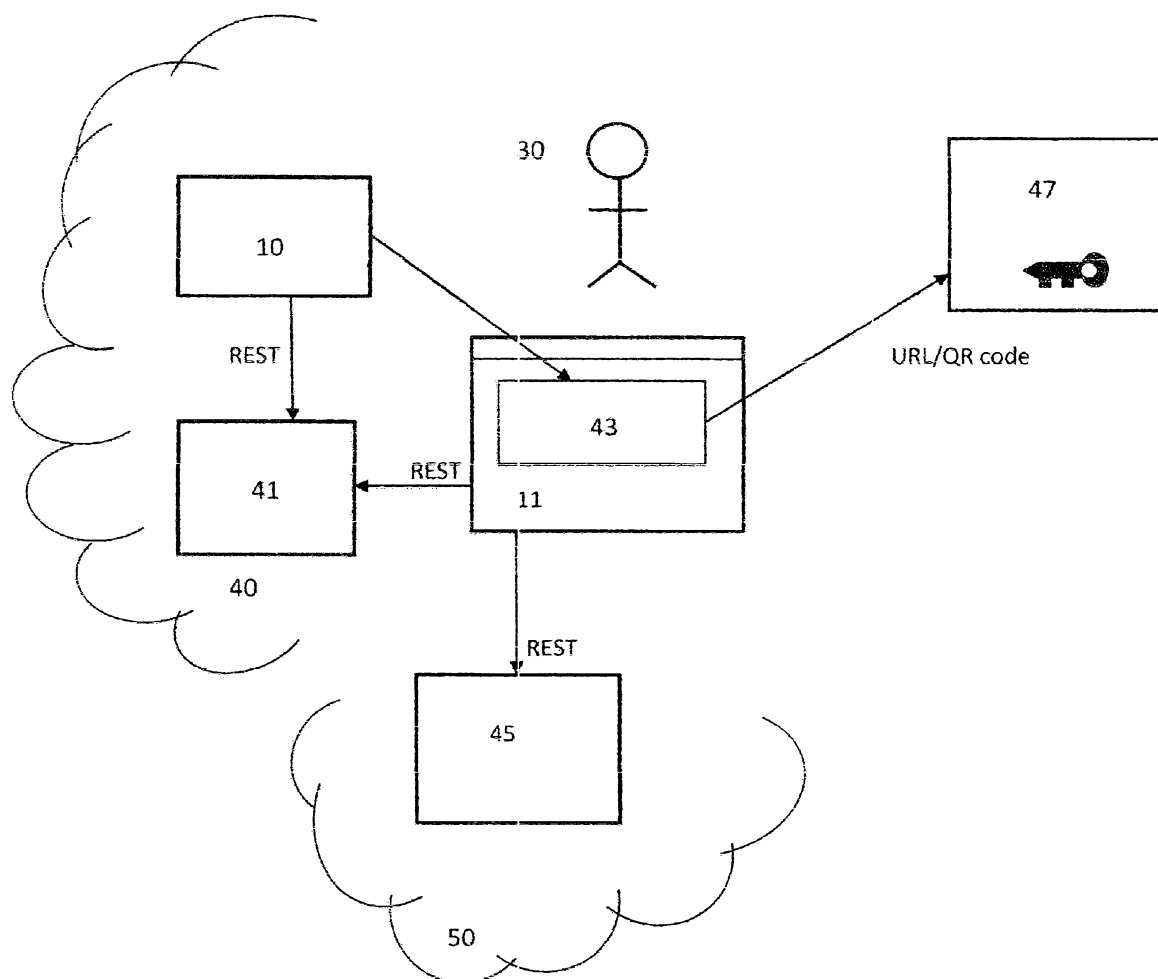


FIG.3

[Fig.4]

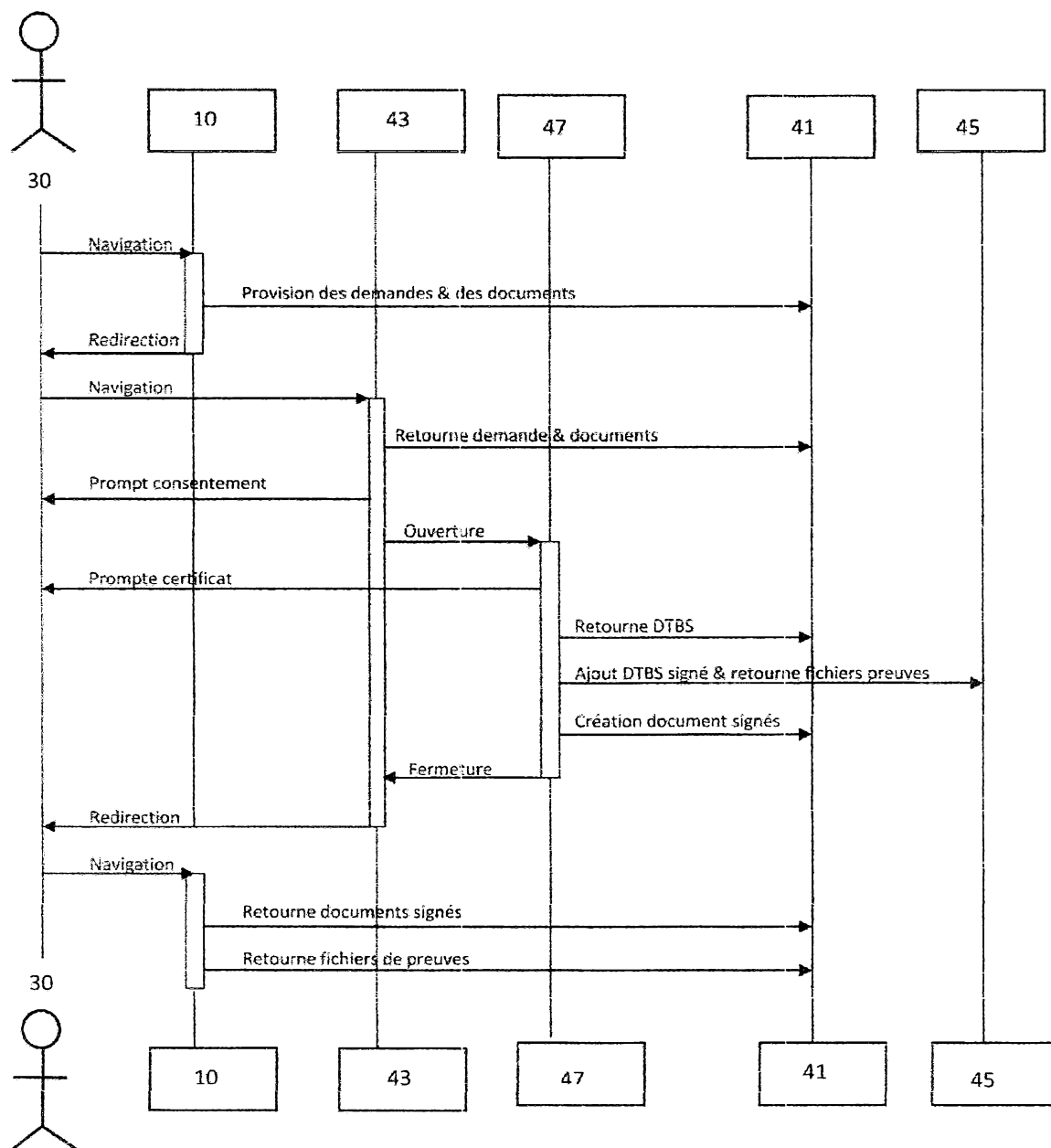


FIG.4