Optimisation d'un Protocole d'Authentification dans les Réseaux de Capteurs Sans Fil

Youssou FAYE
Université de Franche-Comté
Laboratoire LIFC
Besancon, France
yfaye@lifc.univ-fcomte.fr

Hervé GUYENNET
Université de Franche-Comté
Laboratoire LIFC
Besancon, France
herve.guyennet@lifc.univ-fcomte.fr

Ibrahima NIANG
Université Cheikh Anta Diop
Departement Maths-Informatique
Dakar, Senegal
iniang@ucad.sn

Yanbo SHOU
Université de Franche-Comté
Laboratoire LIFC
Besancon, France
yanbo.shou@lifc.univ-fcomte.fr

***Résumé*—Les domaines d'applications des Réseaux de Capteurs sans Fil (RCSFs) ne cessent de s'étendre. Dans certaines de ces applications, afin d'empêcher des entités non autorisées de joindre le réseau, un mécanisme de contrôle d'accès doit être mis en place. Récemment, Wong et al., puis Vaidya et al. ont proposé un protocole qui permet aux utilisateurs d'un RCSF de s'authentifier avant d'accéder aux services fournis. Dans ce papier, après avoir montré les vulnérabilités lors d'attaque par Déni de Service (DoS) et par falsification des estampilles temporaires sur le protocole proposé par Vaidya et al., Nous proposons une solution efficace dans l'environnement des capteurs, qui non seulement conserve tous les avantages du protocole proposé par Vaidya et al., mais améliore sa sécurité. Comparée à cette dernière proposition, notre solution est sécurisée contre les attaques mentionnées, et dans certains cas, consomme moins d'énergie.**

Mots clés : Réseaux de Capteurs Sans Fil, contrôle d'accès, authentification, mot de passe.

I. Introduction

Constitué d'un grand nombre de capteurs, le RCSF est souvent destiné au contrôle d'espaces géographiquement limités. En général, les données récoltées par les capteurs comme par exemple la température ne sont pas confidentielles. Dans certaines applications, la plupart des requêtes sont envoyées à une station de base ou à la passerelle du réseau. Cependant, pour les applications temps réel ou critiques, comme par exemple les applications militaires, les données critiques, doivent être protégées contre toute utilisation frauduleuse, et accessibles en temps réel non seulement depuis la station de base ou la passerelle du réseau, mais parfois aussi de n'importe où dans le réseau à travers les capteurs en mode ad hoc.

Dans ce contexte, il est essentiel de limiter l'accès au réseau seulement aux entités éligibles (capteurs ou utilisateurs), tandis que les requêtes provenant des entités non autorisées ne doivent ni être traitées ni transmises par les capteurs du réseau [1]. L'authentification à distance des utilisateurs a été depuis lontemps la solution de base la plus utilisée dans les réseaux traditionnels. Cependant, dans les réseaux de capteurs, elle reste moins étudiée à cause des contraintes énergétiques, de mémorisation, de calcul et de transmission. La plupart des solutions existantes [2,3,4,5,6] sont généralement concentrées sur l'environnement des cartes à puce. Certaines d'entre elles utilisent une approche basée sur un mot de passe faible, et d'autre (les plus récentes) sur un mot de passe fort. L'authentification par mot de passe-faible utilise la cryptographie à clé publique, plus précisemment le cryptosystème El Gamal [7] qui utilise une signature basée sur le logarithme discret le rendant impraticable dans les RCSFs. Les solutions d'authentification par mot de passe fort utilisent uniquement les fonctions de hachage et l'opérateur ou-exclusif, ce qui facilite leur implémentation dans les RCSFs.

Vaidya et al.[8] est une des dernières solutions basée sur un mot de passe fort. Leur protocole est divisé en trois phases : une phase d'enregistrement, une phase de login, et une phase d'authentification. Cet article montre que, cette solution, de par son manque de vérification du mot de passe lors de la phase de login, est vulnérable lors d'attaque par DoS. De même, elle ne fournit pas une protection complète contre la falsification des estampilles temporelles. Ainsi nous proposons une nouvelle solution qui maintient tous les avantages de leur soulution, tout en améliorant sa sécurité.

Le reste du papier est organisé comme suit. La section II présente l'état de l'art. Une description de la solution de Vaidya et al. [8] et une analyse de sa sécurité sont respectivement fournies dans les sections III et IV. La solution proposée est présentée à la section V, puis sa sécurité est analysée à la section VI. La section VII décrit une implémentation de notre solution et celle de Vaidya et al.[8]. Enfin une brève conclusion est donnée à la section VIII.

II. Etat de l'art

Le contrôle d'accès a toujours été un problème classique dans beaucoup d'applications et systèmes informatiques existants. L'authentification à distance des utilisateurs a été depuis longtemps la solution de base la plus utilisée. Initialement, les solutions d'authentification d'utilisateurs [2,3,4,5,6] proposées dans l'environnement des cartes à puce étaient inspirées de Lamport(1981) [9], à la différence qu'aucune table de vérification n'était pas stockée dans le système distant pour la validité du login de l'utilisateur. Ces solutions utilisent une approche par mot de passe avec un login statique. Certaines d'entre elles [3] utilisent la technique d'un mot de passe faible. Elles présentent l'avantage d'une mémorisation facile du mot de passe. Cependant la cryptographie à clé publique utilisée reste le principal inconvénient pour une application dans l'environnement des capteurs. Par contre, d'autres solutions [2,4,5,6] utilisent la technique d'un mot de passe fort et sont basées uniquement sur les fonctions de hachage

et l'opérateur ou-exclusif, ce qui facilite leur implémentation dans les RCSFs. Leur inconvénient réside de la difficulté de mémorisation du mot de passe. Des solutions comme [10] introduisent la technique du mot de passe fort avec un login dynamique afin de se protéger contre l'usurpation de login. Ce qui permet le libre changement de login et de mot de passe. Cependant, Lee et al.[11] ont montré que ces protocoles basés sur un login dynamique sont toujours vulnérables aux attaques telles que la répétition, la contrefaçon de login et la fabrication. Leur solution proposée pour les cartes à puce utilise la même technique des fonctions de hachage et du ou-exclusif. Pour une adaptation dans les RCSFs, Wong et al. [12] proposent une solution basée sur Lee et al.[11] moins coûteuse en calcul. La solution de Tseng et al. [13] montre les insufisances de Wong et al. [12] et essaie d'augmenter sa sécurité.

Récemment, Vaidya et al., dans [8], améliorent leur version [14] basée sur Wong et al. [12]. Leur solution est sécurisée contre plusieurs attaques comme l'attaque sur le login, par répétition, et par fabrication, mais reste toujours vulnérable à d'autre types d'attaques que nous détaillerons plus loin.

III. Passage en revue du protocole de Vaidya et al.

La solution de Vaidya et al. [8] est composée de quatre phases : une phase d'enregistrement, une phase de login, une phase d'authentification et une phase de changement de mot de passe. Ces phases seront brièvement décrites dans les prochaines sections. Nous utiliserons pour la suite les notations du Tableau I.

Table I
Notations

Symboles	Description
UD	dispositif de l'utilisateur (PDA, PC etc..)
GW	le noeud passerelle
LN	un Noeud capteur Login
H( )	une fonction de hachage à sens unique
$\oplus$	l'opérateur ou-exclusif (XOR)
$\|\|$	l'opérateur de concatenation
Succ_Reg	message d'enregistrement avec succès
Acc_login	message d'accepation du login
Succ_Change	message de changement avec succès
x	la clé de la passerelle
UID	l'identité de l'utilisateur
PW	mot de passe choisi par l'utilisateur
TS	temps d'enregistrement d'un noeud
t, T , T_0	temps actuels enregistrés par un des noeuds
$\triangle$T	délai de transmission
$\longrightarrow$	transmission de message

A. Phase d'Enregistrement

Durant cette phase, l'utilisateur (UD) est au voisinage de la passerelle GW, il choisit librement un mot de passe PW et calcule le haché vpw = H(PW). Puis, au temps TS, il envoie son identité UID et le hachage vpw à la passerelle GW en mode sécurisé. Après réception du message, la passerelle GW calcule X = H(UID$\|$x) puis répond à l'utilisateur le message Succ_Reg (X) avec X pour lui notifier que l'enregistrement est effectué avec succès après avoir stocké (UID, vpw, X, TS). L'utilisateur stocke X pour une utilisation future. La GW distribue ensuite les paramètres (UID, X, TS) aux noeuds capteurs de login (LN) capables de fournir des interfaces aux utilisateurs pour se loguer.

Algorithm 1 *: Etapes Phase d'Enregistrement (PE)*

PE1- UD : Calcule vpw = H(PW) ;
PE2- UD $\longrightarrow$ GW : UID, vpw ;
PE3- GW : Calcule X = H(UID $\|$ x) ;
Stocke UID, vpw, X, TS ;
PE4- GW $\longrightarrow$ UD : Succ_Reg(X) ;
PE5- UD : Stocke X ;
PE5- GW $\longrightarrow$LNs : UID, X, TS ;
PE6- LN : Stocke UID, X, TS ;

B. Phase de Login

Durant la phase de login, l'utilisateur est au voisinage du LN, il calcule A = H(vpw$\|$t), puis soumet (UID, A, t) au LN le plus proche. Après réception de la requête à T_0, le LN vérifie si le UID soumis est non valide et T_0 - t$\geq\triangle$T. Si au moins une condition est vérifiée, le message de login est rejeté, sinon le LN récupère le paramètre A correspondant et calcule C_K = (X $\oplus$A $\oplus$ T_0), puis il envoie (UID, C_K, T_0, t) à la passerelle GW.

Algorithm 2 *: Etapes de la Phase de Login (PL)*

PL1- UD : Calcule A = H(vpw$\|$t) ;
PL2- UD $\longrightarrow$ LN : UID, A, t ;
PL3- LN : **Si** UID est valide
Alors le X correspondant est connu ;
Sinon rejette le message de login ;
Si T_0 - t$\geq\triangle$T
Alors rejette le message de login ;
Sinon récupère A, calcule C_K = (X $\oplus$A $\oplus$ T_0) ;
PL4- LN $\longrightarrow$ GW :UID,C_K,T_0,t ;

C. Phase d' Authentification

Durant cette phase, la passerelle GW vérifie la validité de UID et t, la requête de login est rejetée s'ils ne sont pas valides. S'ils sont valides, alors la passerelle GW vérifie si T_1-T_0 $\geq$ $\triangle$T et T_0-t $\geq$ $\triangle$T. Si au moins une des conditions est satisfaite, la requête de login est considérée comme un message répété et est rejetée. Sinon la passerelle récupère le vpw et le paramètre A correspondants, puis calcule A'= H(vpw$\|$ t) et C_K' = (X $\oplus$ A'$\oplus$ T_0). Si $C_K \neq C_K$', le message de login est rejeté, sinon, la GW calcule V_M = H(X$\|$A'$\|$ T_1) et envoie un message d'acceptation (Acc_login, V_M, T_1) au LN. Le LN calcule V'_M, et si V_M =V'_M , il calcule également Y_K = H($V'_M\|$ T_2) et envoie (Acc_login, Y_K, T_1, T_2) à l'utilisteur UD. Après réception du message au temps T_3, l'utilisateur UD vérifie si T_1-T_0 $\geq$ $\triangle$T et T_0-t $\geq$ $\triangle$T. Si au moins une des conditions est vraie, alors le message Acc_login est rejeté, sinon, l'utilisateur calcule $V"_M$ =H(X$\|$A$\|T_1$) et Y'_K =

H($V"_M \| T_2$), puis vérifie si $Y_K = Y'_K$. Si cette condition est vraie, l'utilisateur UD commence à obtenir les données, sinon le message d'acceptation de login Acc_login est rejeté.

Algorithm 3 *: Etapes de la Phase d'Authentification (PA)*

PA1- GW : **Si** UID et t sont valides
Alors récupère les paramètres (UID, X, TS) ;
Sinon rejette le message de login ;
Si T_1 - $T_0 \geq \triangle$T et T_0 - t$\geq \triangle$T
Alors message de login supposé répété et rejeté ;
Sinon calcule A' = H(vpw$\|$ t) ;
calcule C'_K = (X$\oplus$A'$\oplus$ T_0) ;
Si $C'_K \neq C_K$
Alors rejette le message de login ;
Sinon calcule V_M = H(X$\|$A'$\|T_1$) ;
Stocke t ;
PA2- GW $\longrightarrow$ LN : Acc_login,V_M,T_1 ;
PA3- LN : **Si** T_2 - $T_1 \geq \triangle$T
Alors rejette le message de login ;
Sinon Calcule V'_M = H(X$\|$A$\|T_1$) ;
Si $V_M \neq V'_M$
Alors LN rejette le message de login ;
Sinon Calcule Y_K=H($V'_M \| T_2$) ;
PA4- LN $\longrightarrow$UD : Acc_login,Y_K,T_1,T_2 ;
PA5- UD : **Si** T_1 - $T_0 \geq \triangle$T et T_0 - t$\geq \triangle$T
Alors rejette le message Acc_login ;
Sinon calcule $V"_M$ = H(X$\|$A$\|T_1$) ;
calcule Y'_K = H($V"_M \| T_2$) ;
Si $Y_K \neq Y'_K$
Alors rejette le message Acc_login ;
Sinon commence à obtenir les données ;

D. Phase de changement de mot de Passe

Durant la phase de changement de mot de passe, l'utilisateur UD change son mot de passe PW en PW1. Ainsi il calcule vpw1 = H(PW1) et envoie le triplet (UID, vpw, vpw1) à la passerelle GW en mode sécurisé. La passerelle GW vérifie son UID et vpw. Si les deux sont corrects, elle met à jour sa base de données. La passerelle envoie ensuite un message de changement effectif avec succès (Succ_Change) à UD en même temps elle distribue les informations de mise à jour à tous les LNs. Après réception, les LNs vérifient la validité du UID avant de mettre à jour leurs données.

Algorithm 4 *: Etapes de la Phase changement mot de Passe (PP)*

PP1- UD : Calcule vpw1 = H(PW1) ;
PP2- UD $\longrightarrow$ GW : UID, vpw, vpw1 ;
PP3- GW : **Si** UID et vpw existe sur sa liste
Alors mets à jour vpw avec vpw1, TS avec TS1 ;
PP4- GW $\longrightarrow$ UD : envoie Succ_Change ;
PP5- GW $\longrightarrow$LNs : envoie UID, TS1 ;
LN : **Si** UID existe dans sa liste ;
Alors mettre à jour TS avec TS1 ;

IV. Vulnérabilités du protocole de Vaidya et al.

Dans cette section, nous nous intéressons à la sécurité du protocole de Vaidya et al. [8]. Ainsi nous allons montrer que ce protocole est vulnérable à l'attaque de DoS durant sa phase de login, et à l'attaque par falsification d'estampilles temporaires durant sa phase d'authentification.

Notons que, durant la phase de login, l'utilisateur est dans le voisinage du LN, le scénario de communication entre le LN et la passerelle GW est de plusieurs sauts. Puisque le UID a circulé plusieurs fois en clair dans le réseau, et que durant la phase de login, seuls le UID et le temps t sont vérifiés par le LN, le DoS peut survenir de deux façons différentes. Premièrement, l'intrus peut intercepter ou écouter un UID valide puis le soumet avec un faux mot de passe. Puisque le LN vérifie seulement le UID, il va transmettre ce message à la passerelle située à plusieurs sauts. Du coup, tous les capteurs intermédiaires entre le LN et la GW vont assurer la propagation d'une fausse requête à travers le réseau. Ce qui va entrainer une perte d'énergie considérable au niveau de tous ces capteurs assurant le relai. Cette transmission d'un saut de l'intrus entrainera plusieurs retransmissions dans le réseau, ce qui à la longue épuise l'énergie des capteurs. Deuxièmement, ce scénario peut arriver d'une autre façon. Puisque les mots de passe n'apparaissent jamais en clair lors de leur saisie, si un utilisateur se trompe de saisie de mot de passe, le même effet se reproduit. La transmission étant généralement l'opération la plus coûteuse en énergie, la propagation de fausses requêtes doit être évitée.

Le protocole est aussi vunérable lors d'attaque par falsification d'estampilles temporaires échangées durant sa phase d'authentification. Vaidya et al.[8] ont supposé qu'un attaquant ayant capturé un LN, obtient UID, X, TS, et qu'il lui est possible d'écouter UID, A, t afin de monter une attaque par falsification sur Wong et al. [12]. Partant de ces suppositions, leur protocole, de façon différente, sera vulnérable à l'attaque par falsification des temps transmis entre les différentes entitées impliquées. Puisque seul le délai de transmission est vérifié, l'attaquant peut créer deux faux temps T'_0 et t' dont la différence respecte le délai de propagation en y ajoutant un petit nombre ξ partout sur les deux temps T_0 et t déjà transmis en clair dans le réseau. Ainsi il effectue T'_0=T_0+ξ ,t'=t+ξ, ensuite il calcule C'_K = H(X$\oplus$A$\oplus T'_0$) puis envoie le message (UID, $C'_K T'_0$,t'. Puisque (T_1 - T'_0)$\leq \triangle$T et (T'_0-t')$\leq \triangle$T le message passe.

V. Solution Proposée

Dans cette partie, nous proposons une nouvelle solution afin de résoudre les faiblesses notées dans Vaidya et al.[8]. Cette nouvelle solution comporte les mêmes phases : une phase d'enregistrement, une phase de login, une phase d'authentification, et une phase de changement de mot de passe qui est la seule à ne pas avoir subi de changements dans cette nouvelle solution proposée.

A. Phase d'Enregistrement

Dans Vaidya et al., l'utilisateur choisit un mot de passe PW, puis calcule vpw=H(PW) et envoie vpw avec son identité pour se loguer. Et pour le reste du protocole, le mot de passe PW n'est plus utilisé. Il n'est pas nécessaire de calculer vpw, qui est autant vulnérable que PW. L'utilisateur peut choisir tout simplement un mot de passe puis le soumettre, et c'est le H(PW) qui sera stocké par la passerelle GW et les noeuds de login LNs.

Ainsi dans cette phase d'enregistrement, l'utlisateur UD choisit librement son mot de passe PW qu'il va soumettre avec son identité à la passerelle GW en mode sécurisé. La passerelle GW calcule X = H(UID‖x) puis répond à l'utilisateur le message Succ_Reg(X) avec X pour lui notifier que l'enregistrement est effectué avec succès. Elle stocke les paramètres (UID, H(PW), X, TS), et distribue (UID, X, H(PW),TS) aux noeuds login LNs capables de fournir des interfaces aux utilisateurs pour se loguer.

Algorithm 5 *: Etapes de la Phase d'Enregistrement(PP)*

PE1- UD : choisit son mot de passe PW ;
PE2- UD $\longrightarrow$ GW : UID, PW ;
PE3- GW : calcule X = H(UID ‖ x)
Store UID, H(PW), X, TS ;
PE4- GW $\longrightarrow$ UD : Succ_Reg (X) ;
PE5- UD : Stocke X ;
PE6- GW $\longrightarrow$LNs : UID, X, H(PW), TS
PE7- LN : Stocke UID, X, H(PW),TS ;

B. Phase de Login

L'utilisateur calcule A= H(H(PW)‖t) et soumet (UID, A, t) au LN. Après réception de la requête au temps T_0, le LN vérifie : si l'identité UID est non valide ou A $\neq$ H(H(PW)‖t), ou T_0 - t $\geq\triangle$, le message de login est rejeté, sinon, le LN calcule C_K = (X$\oplus$A$\oplus T_0$) et t'=H^2(PW)$\oplus$t, puis envoie (UID,C_K,T_0,t') à la passerelle GW.

Algorithm 6 *: Etapes de la Phase de Login (PL)*

PL1- UD : calcule A = H(H(PW)‖t) ;
PL2- UD : $\longrightarrow$ LN : UID, A, t ;
PL3- LN : **Si** UID est non valide
Alors rejette le message de login ;
Sinon calcule A' = H(H(PW)‖t) ;
Si A$\neq$A'
Alors rejette le message de login ;
SinonSi T_0 - t$\geq\triangle$T
Alors rejette le message de login ;
Sinon récupère le paramtre A correspondant ;
calcule C_K = (X $\oplus$A $\oplus$ T_0) ;
calcule t'=H^2(PW)$\oplus$t ;
PL4- LN $\longrightarrow$ GW :UID,C_K,T_0,t'

C. Phase d'Authentification

Durant cette phase, la passerelle vérifie si le UID et le temps t sont valides. Le message de login est rejeté s'ils ne le sont pas. Ensuite il calcule t=t'$\oplus H^2$(PW), puis vérifie si T_1 - $T_0\geq\triangle$T et T_0 - t$\geq\triangle$T. Si au moins une condition est satisfaite, alors le message de login est considéré comme répété et est rejeté. Sinon la passerelle GW récupère les paramètres H(PW) et A correspondants puis calcule A' =H(H(PW)‖t) et C'_K = (X$\oplus$A'$\oplus$ T_0). Le message de login est rejeté si $C_K\neq C'_K$, sinon la GW calcule V_M = H(X‖A'‖T_1) puis envoie le message (Acc_login, V_M, T_1) au LN et stocke t. Le LN calcule V'_M, et après vérification que V_M = V'_M, il calcule Y_K = H(V'_M‖T_2). Ensuite il envoie le message (Acc_login, Y_K, T_1, T_2) à l'utilisateur UD. Après réception du message au temps T_3, l'utilisateur UD vérifie si T_1 - $T_0\geq\triangle$T et T_0 - t$\geq\triangle$T. Si au moins une condition est vérifiée, le message d'acceptation de login est rejeté. Sinon, l'utilisateur calcule, $V"_M$ = H(X‖A‖T_1) et Y'_K = H($V"_M$‖T_2), puis vérifie si Y_K =Y'_K. Si la condition est vraie, l'utilisateur commence à obtenir les données, sinon il rejette le message d'acceptation de login.

Algorithm 7 *: Etapes de la Phase d'Authentification (PA)*

PA1- GW : **Si** UID et t sont valides
Alors récupère (UID, X, TS, H(PW))
calcule t=t'$\oplus H^2$(PW) ;
Sinon rejette le message de login ;
Si T_1 - $T_0\geq\triangle$T et T_0 - t$\geq\triangle$T
Alors message de login supposé répété et rejeté ;
Sinon calcule A' = H(H(PW)‖ t ;
calcule C'_K = (X$\oplus$A'$\oplus$ T_0) ;
Si $C'_K\neq C_K$
Alors rejette le message de login ;
Sinon calcule V_M = H(X‖A'‖T_1) ;
Stocke t ;
PA2- GW $\longrightarrow$ LN : Access_login,V_M,T_1 ;
PA3- LN : **Si** T_2 - $T_1\geq\triangle$T
Alors rejette le message de login supposé répété ;
Sinon calcule V'_M = H(X‖A‖T_1) ;
Si $V_M\neq V'_M$
Alors rejette le message de login ;
Sinon calcule Y_K=H(V'_M‖T_2) ;
PA4- LN $\longrightarrow$UD : Acces_login,Y_K,T_1,T_2 ;
PA5- UD : **Si** T_1 - T_0t$\geq\triangle$T et T_0 - t$\geq\triangle$T
Alors rejette le message (Acces_login) ;
Sinon calcule $V"_M$ = H(X‖A‖T_1) ;
calculeY'_K = H($V"_M$‖T_2) ;
Si $Y_K\neq Y'_K$
Alors rejette le message (Acces_login) ;
Sinon UD commence à obtenir les données ;

D. Phase de changement de mot de Passe

Cette phase n'a pas changé, elle est la même que celle de Vaidya et al.

VI. Analyse de sécurité de notre solution

Dans cette section, nous allons analyser notre solution proposée afin de montrer qu'elle est résistante à plusieurs types d'attaques. Nous terminons la section par une étude comparative avec d'autres solutions existantes.

A. Securité

L'attaque par Deni de Service : en donnant aux LNs la possibilité de vérifier le mot de passe lors de la phase de login, notre solution se protège contre le DoS. Puisque le LN stocke H(PW), après avoir reçu le message de login (UID, A,t), il peut calculer A'=H(H(PW)||t). Si A'=A, alors le mot de passe est correct, sinon le message de login est rejeté.

L'attaque par falsifcation : la solution proposée protège également contre la falsification des estampilles temporaires envoyées en clair entre le LN et la GW dans Vaidya et al. Nous proposons de les envoyer en mode sécurisé avec une simple utilisation du ou-exclusif. C'est ainsi que le LN, après avoir reçu le message de login, calcule une fausse estampille t'=H^2(PW)$\oplus$t puis envoie le message (UID,C_K,T_0,t') au lieu de (UID,C_K,T_0,t) à la passerelle GW. Ce qui rend l'estampille t confidentiel. Puisque H^2(PW)$\oplus$t$\oplus H^2$(PW)=t, la passerelle GW calcule t=t'$\oplus H^2$(PW) pour retrouver l'estampille t.

B. Comparaison avec d'autres solutions

Le Tableau II, donne une comparaison en termes d'opérations de hachage, de ou-exclusif ainsi que le nombre de communications multi-sauts de quelques solutions utilisant la même approche.

Table II
Comparaison du nombre d'opérations effectuées

Protocoles	Nombre total d'opérations
Wong et al. [12]	$7T_H$+$4T_{XOR}$+$3C_{MH}$
Tseng et al.[13]	$5T_H$+$4T_{XOR}$+$3C_{MH}$
Vaidya et al. [14]	$8T_H$+$4T_{XOR}$+$3C_{MH}$
Vaidya et al. [8]	$11T_H$+$4T_{XOR}$+$3C_{MH}$
Solution Proposée	$15T_H$+$7T_{XOR}$+$3C_{MH}$

T_H : temps pour exécuter la fonction de hachage H().
T_{XOR} : temps pour exécuter l'opération XOR .
C_{MH} : Délai de communication multi-sauts entre le LN et la GW.
NB : Ces opérations peuvent aussi être traduites en énergie.

D'après le Tableau II, on peut remarquer que la solution proposée a un coût supérieur de quatre opérations de hachage (T_H) et de trois opérations ou-exclusif (T_{XOR}) à celle de Vaidya and al.[8]. Notons que le coût énergétique de l'opération ou-exclusif est largement inférieur à celui du hachage. Ainsi notre solution se retrouve avec un léger surplus de consommation d'énergie tout en produisant une meilleure sécurité.

Considérons une requête avec une UID valide mais avec un mot de passe erroné qui proviendrait de l'intrus ou d'une erreur de saisie de la part d'un utilisateur légitime. Pour notre solution, cette requête de login sera freinée au niveau du LN, et pour les autres solutions, elle sera propagée jusqu'à la passerelle GW. Pour ce cas, le Tableau III montre une comparaison entre notre solution et celles de Vaidya et al. On voit ici que

Table III
Overhead cost Comparison.

Protocoles	Nombre total d'opérations
Vaidya et al.[14]	$4T_H$+$2T_{XOR}$+$2C_{MH}$
Vaidya et al. [8]	$4T_H$+$2T_{XOR}$+$2C_{MH}$
Solution proposée	$3T_H$

notre solution présente moins d'opérations, par conséquent sa consommation d'énergie sera meilleure. Et en plus, notons que, cet écart d'énergie va augmenter considérablement dans les protocoles de Vaidya et al. en fonction du nombre de sauts entre le LN et la GW, alors que pour notre solution qui ne fait pas intervenir une communication multi-saut (C_{MH}) pour les fausses requêtes, l'energie sera constante.

VII. IMPLEMENTATION

Nous rapellons qu'il y a deux protocoles de Vaidya et al.[8, 14]. Le but de notre implémentation est d'estimer la consommation énergétique en fonction du nombre de sauts entre le LN et la GW en s'appuyant sur les comparaisons effectuées dans les deux tableaux de la précédente section, . Nous avons fait une implémentation de notre solution et celle de Vaidya et al. [8] avec TinyOS. Le programme est testé sur la plateforme MicaZ, et le simulateur Avrora est utilisé pour mesurer la consommation d'énergie.

A la première étape de la simulation, nous avons évalué la consommation d'énergie en fonction du nombre de sauts entre le LN et la GW en se basant sur le Tableau II où les deux protocoles sont considérés sans attaques. Pour chaque paramètre de données comme (UID,A,PW, C_k etc.) on a utilisé des données de 16 bits. Pour le hachage, nous avons utilisé une implémentation de la fonction de hachage universelle PolyR décrite dans le papier de Ted and al.[15], comme une interface TinyOS.

La **Figure 1**, montre la consommation d'énergie de chaque solution.

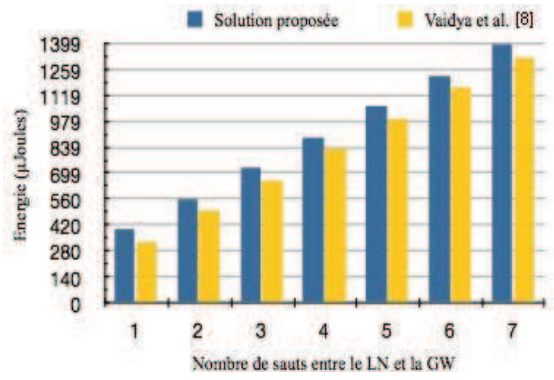

Figure 1. Consommation énergétique basée sur le Tableau II.

A la deuxième étape, nous nous intéressons à l'effet de propagation d'une fausse requête sur la consommation

énergétique. Ainsi nous avons utilisé le Tableau III pour évaluer l'éneregie consommée en fonction du nombre de sauts entre le LN et la GW. La **Figure 2**, montre la consommation d'énergie de chacune des solutions. Ainsi nous pouvons voir que l'énergie reste constante pour notre solution car la fausse requête ne se propage pas. Elle augmente en fonction du nombre de sauts dû aux transmissions des capteurs assurant la propagation de la fausse requête.

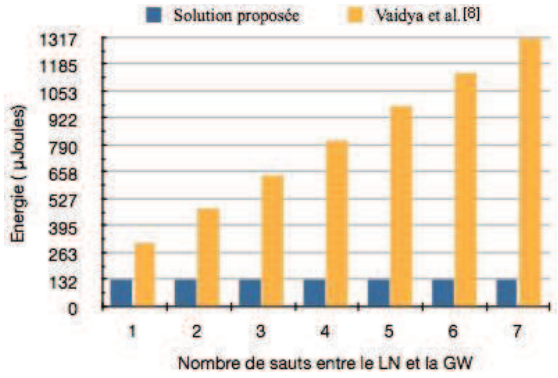

FIGURE 2. Consommation énergétique basée sur le Tableau III.

VIII. CONCLUSION

Dans ce papier, nous avons proposé d'optimiser le protocole de Vaiya et al.[8] afin d'avoir un protocole beaucoup plus sûr dans l'environnement des capteurs. La solution proposée conserve tous les avantages de celle de Vaidya et al. et améliore sa sécurité par la protection contre le DoS et la falsification. Dans certains cas, elle justifie une meilleure sécurité avec seulement un coût énergétique additionnel de quatre opérations de hachage (T_H) et de trois opérations ou-exclusif (T_{XOR}). Et dans d'autres cas, nous avons aussi montré dans l'implémentation, qu'elle est bien meilleure en consommation énergétique car elle justifie au moins une opération de hachage (T_H), de deux opérations ou-exclusif (T_{XOR}) et deux opérations (C_{MH}) de moins que celle de Vaidya et al.[8]. Cette différence augmente avec le nombre de sauts entre le LN et la GW. Nos travaux futurs vont dans le sens d'introduire la probabilité de risque à partir de laquelle le comportement de chaque solution sera étudié. Ce qui permettra de déterminer énergétiquement la meilleure solution pour une architecture donnée d'un RCSFs.

RÉFÉRENCES

[1] Y. Faye, I. Niang and T. Noêl. A Survey of Access Control Schemes in Wireless Sensor Networks. World Academy of Science, Engineering and Technology, Issue 59 : 2011, Paris, France, Pages 814-823, November 2011.

[2] A. K. Awasthi, and S. Lal, A remote user authentication scheme using smart cards with Forward Secrecy, IEEE Transactions on Consumer Electronics, vol.49, no.4, pp.1246-1248, Nov. 2003.

[3] M. S. Hwang, C. C. Chang, and K. F. Hwang, An E1Gamal-like cryptosystem for enciphering large messages, IEEE Trans .on Knowledge and Data Engineering, vol.14, no.2, pp.445-446, 2002.

[4] C. C. Lee, L. H. Li, and M. S. Hwang, A remote user authentication scheme using hash functions, ACM Operating Systems Review, vol.36, no.4, pp.23-29, 2002.

[5] J. J. Shen, C. W. Lin, and M. S. Hwang, A modified remote user authentication scheme using smart cards, IEEE Trans. on Consumer Electron., vol.49, no.2, pp.414-416, May 2003.

[6] H. M. Sun, An Efficient remote user authentication scheme using smart cards, IEEE Trans. on Consumer Electron., vol. 46, no. 4, pp. 958-961, Nov. 2000.

[7] B. Schneier, Applied cryptography, John Wiley & Sons Inc., New York, 2nd edition, 1996.

[8] Binod Vaidya1, Min Chen2 and Joel J. P. C. ,Rodrigues3, Improved Robust User Authentication Scheme for Wireless Sensor Networks December 2009

[9] L. Lamport, Password authentication with insecure communication, Communications of the ACM, vol.24, no.11, pp.770-772, 1981.

[10] M.L. Das, A. Saxena, and V.P. Gulati, A Dynamic ID-based Remote User Authentication Scheme, IEEE Transactions on Consumer Electronics, Vol. 50, No. 2, 2004.

[11] C.Y. Lee, C.H. Lin, and C.C. Chang, An Improved Low Communication Cost User Authentication Scheme for Mobile Communication, Proceedings of the IEEE 19th International Conference on Advanced Information Networking and Applications (AINA 2005), Taiwan, March 2005.

[12] K. H. M. Wong, Y. Zheng, J. Cao, and S. Wang, A dynamic user authentication scheme for wireless sensor networks, In Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 06), vol. 1, Jun. 2006, pp. 244-251.

[13] Tseng, H. R., Jan, R. H., and Yang, W. 2007. An improved dynamic user authentication scheme for wireless sensor networks. In Proceedings of the IEEE Global Communications Conference (GLOBECOM07), Nov. 2007 ;986-990.

[14] B Vaidya, J.S. Silva, J.J. Rodrigues, Robust Dynamic User Authentication Scheme for Wireless Sensor Networks, In Proc. of the 5th ACM Symposium on QoS and Security for wireless and mobile networks (Q2SWinet 2009), Tenerife, Spain, Oct. 2009, pp 88-91.

[15] Ted Krovetz, Phillip Rogaway,Fast Universal Hashing with Small Keys and No Preprocessing : The PolyR Construction, D. Won (Ed.) : ICISC 2000, LNCS 2015, pp. 73-89, 2001. Springer-Verlag Berlin Heidelberg 2001