

# Protocole de sécurisation de gestion à distance d'un radar automatique de route

Pierre-Marie JUNGES, Florent NOSARI  
<https://github.com/p1ma/SSI>

30 janvier 2018

## 1 Présentation du protocole

### 1.1 But

Le but de ce protocole est de permettre la gestion à distance d'un radar automatique de route par les autorités compétentes en ayant la certitude que les informations soient authentiques (i.e. qu'elles proviennent bien des relevés fait par le radar/gestionnaire ), que seul le gestionnaire puisse accéder au données du radar, que seul le gestionnaire puisse contrôler le radar et que toutes les informations qui transitent entre les deux soient confidentielles (i.e. qu'une tierce personne ne puisse pas y avoir accès).

En résumé :

- Les informations qui transitent doivent être confidentielles et authentiques
- Seul le gestionnaire peut contrôler le radar et accéder au données du radar

### 1.2 Contraintes

Voici le contraintes que nous avons décidé de prendre en compte pour rendre le protocole plus proche des conditions réelles.

- Les acteurs ne partagent pas de clé secrète avant l'initiation du protocole.
- Le gestionnaire ne connaît pas nécessairement la clé publique du radar.

### 1.3 Déroulement

Le protocole est initié par le gestionnaire qui demande à un serveur d'authentification le droit de d'accéder au radar. Les échanges se font à l'aide de cryptographie à clé publique jusqu'à obtenir une clé secrète commune entre le gestionnaire et le radar.

L'algorithme d'échange est décrit ci-dessous :

Soient  $G$  le gestionnaire avec  $PK_g$  sa clé publique et  $SK_g$  sa clé privé  
 $R$  le radar avec  $PK_r$  sa clé publique et  $SK_r$  sa clé privé  
 $S$  le serveur avec  $PK_s$  sa clé publique et  $SK_s$  sa clé privé  
 $K_{Session}$  la clé secrète partagé lors d'une session  
 $N_g$  et  $N_r$  des nonces  
*Commande* une commande quelconque  
*Resultat* une réponse quelconque

Les connaissances initiales sont les suivantes :

$G : \{G, R, S, PK_g, PK_s, Commande\}$   
 $R : \{R, S, PK_r, PK_s\}$   
 $S : \{G, R, S, PK_s, PK_r\}$

Le gestionnaire initie le protocole en envoyant une demande de connexion au serveur avec l'objet de la demande signé.

$G \rightarrow \{R_{inv(PK_g)}.G\}_{PK_s} \rightarrow S$

Si l'identité du gestionnaire et du radar sont vérifiées alors le serveur envoie les informations de connexion ( $K_{Session}$  étant générée à ce moment là et signé par le serveur) au gestionnaire.

$G \leftarrow \{\{K_{Session}\}_{inv(PK_s)}.R.PK_r\}_{PK_g} \leftarrow S$

Une fois que le gestionnaire a reçu les informations de connexions, il envoie au radar la clé secrète  $K_{Session}$  signé par le serveur.

$G \rightarrow \{G.\{K_{Session}\}_{inv(PK_s)}\}_{PK_r} \rightarrow R$

Le radar confirme la réception de la clé en envoyant  $G$  au gestionnaire suivit  $N_r$  pour vérifier l'authenticité de la commande qui va suivre le tout chiffré avec  $K_{Session}$ .

$G \leftarrow \{G.N_r\}_{K_{Session}} \leftarrow R$

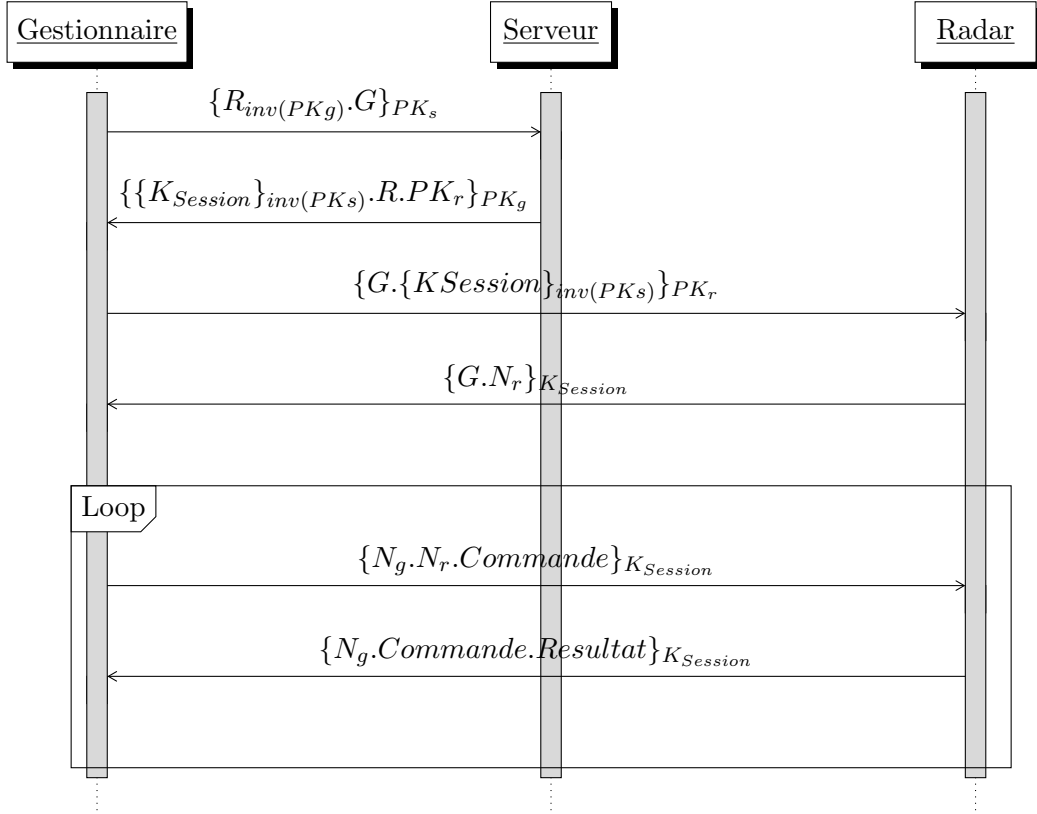
Le gestionnaire peut ainsi envoyer la commande et l'authentifier avec le nonce, il envoie aussi un nonce pour authentifier le résultat de la commande.

$G \rightarrow \{N_g.N_r.Commande\}_{K_{Session}} \rightarrow R$

Le radar envoie par la suite la commande suivit du résultat de la commande et du nonce qui l'identifie.

$G \leftarrow \{N_g.Commande.Resultat\}_{K_{Session}} \leftarrow R$

FIGURE 1 – Échanges effectués lors du déroulement du protocole



## 2 Analyse à l'aide du logiciel AVISPA

### 2.1 Les goals

Les propriétés que nous avons tester avec AVISPA sont les suivante :

- Confidentialité de  $K_{Session}$
- Confidentialité de  $Commande$
- Confidentialité de  $Resultat$
- Confidentialité de  $N_g$
- Confidentialité de  $N_r$
- Authentification sur  $N_g$
- Authentification sur  $Resultat$
- Authentification sur  $N_r$

## 2.2 Résultat

### 2.2.1 Conditions normales

En conditions normales, c'est à dire de  $G$  à  $R$  uniquement, le protocole est sécurisé. Dans le fichier HTPSL on crée les sessions de la façon suivante :

- $A$  communique avec  $B$

Avec  $A$  représentant le gestionnaire et  $B$  le radar.

```
1 SUMMARY
2 SAFE
3
4 DETAILS
5 BOUNDED NUMBER OF SESSIONS
6 TYPED MODEL
7 BOUNDED SPEC. READING DEPTH
8
9 PROTOCOL
10 protocol.if
11
12 GOAL
13 As specified
14
15 BACKEND
16 CL-AtSe
17
18 STATISTICS
19 Analysed      : 203 states
20 Reachable    : 83 states
21 Translation: 0.03 seconds
22 Computation: 0.02 seconds
```

Listing 1 – Résultat d'exécution - Conditions normales

### 2.2.2 Ajout d'un intrus au milieu

Ici on teste dans les conditions où un intrus  $I$  tenterait une attaque type *man-in-the-middle*. Dans le fichier HTPSL on crée les sessions de la façon suivante :

- $A$  communique avec  $I$
- $I$  communique avec  $B$
- $A$  communique avec  $B$

Avec  $A$  représentant le gestionnaire et  $B$  le radar.

```
1 SUMMARY
2 SAFE
3
4 DETAILS
5 BOUNDED NUMBER OF SESSIONS
6 TYPED MODEL
7 BOUNDED SPEC. READING DEPTH
8
9 PROTOCOL
```

```

10 protocol.if
11
12 GOAL
13 As specified
14
15 BACKEND
16 CL-AtSe
17
18 STATISTICS
19 Analysed      : 3206 states
20 Reachable     : 855 states
21 Translation: 0.05 seconds
22 Computation: 0.14 seconds

```

Listing 2 – Résultat d’exécution - Man-in-the-middle

### 2.2.3 Rejeu

Ici on teste dans les conditions où l’on fait deux communications équivalent. Dans le fichier HTPSL on crée les sessions de la façon suivante :

- $A$  communique avec  $B$
- $A$  communique avec  $B$

Avec  $A$  représentant le gestionnaire et  $B$  le radar.

```

1 SUMMARY
2 SAFE
3
4 DETAILS
5 BOUNDED NUMBER OF SESSIONS
6 TYPED MODEL
7 BOUNDED SPEC. READING DEPTH
8
9 PROTOCOL
10 ../protocol.if
11
12 GOAL
13 As specified
14
15 BACKEND
16 CL-AtSe
17
18 STATISTICS
19 Analysed      : 419280 states
20 Reachable     : 217455 states
21 Translation: 0.05 seconds
22 Computation: 36.64 seconds

```

Listing 3 – Résultat d’exécution - Rejeu

### **3 Démarche**

Tout notre démarche est visible sur notre dépôt github <https://github.com/p1ma/SSI> ainsi que le code source.