MAKI: Une infrastructure à clefs publiques pour systèmes muti-agents

Arthur Baudet[1,2], Oum-El-Kheir Aktouf[1], Annabelle Mercier[1], Philippe Elbaz-Vincent[2]

[1] Univ. Grenoble Alpes, Grenoble INP, LCIS, 26000 Valence, France
[2] Univ. Grenoble Alpes, CNRS, IF, 38000 Grenoble, France

{arthur.baudet,oum-el-kheir.aktouf,annabelle.mercier}@lcis.grenoble-inp.fr
philippe.elbaz-vincent@math.cnrs.fr

Résumé

Ce papier présente une infrastructure à clefs publiques pour les systèmes multi-agents-embarqués ouverts. Ces derniers sont particulièrement vulnérables car ils permettent à des agents inconnus, et possiblement malicieux, d'entrer et d'interagir avec les agents existants. Nous nous intéressons notamment aux attaques portant sur les communications afin de permettre aux agents de communiquer sans risques de voir leurs communications manipulées. Pour ce faire, nous proposons une infrastructure mettant à profit l'autonomie des agents et permettant de s'assurer de la sécurité de leurs interactions.

Mots-clés

Déploiement de systèmes multi-agents, système multi-agent ouvert, agent embarqué, infrastructure à clefs publiques

Abstract

This paper presents a public key infrastructure for open multi-agent systems of embedded agents. These systems are very prone to attacks as they are confronted with unknown systems with unknown goals. We aim at securing the communications between agents to provide foundations for more advanced security solutions as well as allowing agents to communicate without the risk of their messages being tampered with. To do so, we deploy a key infrastructure taking advantage of the agents autonomy to allow authenticity and integrity checks and accountability of all interactions.

Keywords

Multi-agent system deployment, open multi-agent system, embedded agent, public key infrastructure

1 Introduction

Le paradigme multi-agent permet de construire des logiciels et systèmes plus résilients, pouvant plus facilement passer à l'échelle, et pouvant se confronter aux challenges actuels dont la complexités ne cesse de croitre. Notamment, avec l'augmentation constante du nombre d'objets connectés, le développement des systèmes autonomes, ce paradigme semble être de plus en plus pertinent pour réaliser le contrôle et la coordination de ces systèmes embarqués hétérogènes. Dans ce cadre, nous nous intéressons plus particulièrement aux systèmes ouverts, lesquels permettent à de nouvelles entités de les rejoindre ou de le quitter. Nous les nommons Systèmes Multi-Agent-Embarqués (SMAE) ouverts. De tels systèmes peuvent exister sous la forme de réseaux de drones coopérants, pour surveiller des feux de forêts par exemple, entrant et sortant de systèmes en fonction de leurs ressources. Un système pareil pourrait profiter d'une collaboration de constructeurs de drones, chacun déployant des drones au fur et à mesure que nécessaire.

Des études récentes [4, 7] montrent que les SMAE, et les systèmes similaires sont particulièrement vulnérables aux menaces venant de l'intérieur ainsi qu'aux attaques sur les communications. Ils montrent également que les Systèmes de Gestion de la Confiance (SGC) sont un moyen courant de mitiger ces menaces. Cependant, ces SGC prennent souvent des hypothèses fortes concernant les couches inférieures, notamment la couche cryptographique [11, 13]. Par exemple, ces hypothèses peuvent nécessiter la présence d'une tierce partie pour fournir une racine de confiance ou pour précharger des certificats dans les agents. Cela rend ces solutions inapplicables dans un contexte ouvert et décentralisé. Ainsi, une solution spécifique pour fournir aux agents des clefs cryptographiques leur permettant de communiquer de manière sécurisée est requise ici.

Le reste du papier est structuré de la sorte : la prochaine section présente les hypothèses et le modèle d'attaquant que nous considérons dans ce travail. Puis, on fonction de ce modèle, nous discutons des travaux connexes en Section 3. Ensuite, nous introduisons notre contribution dans la Section 4 et en proposons une validation dans la Section 5 sous deux aspects différents : d'une part à l'aide d'un outil de vérification de modèles et d'autre parts à travers le développement en simulation d'une preuve de concept. Enfin, nous concluons en Section 6.

2 Hypothèses et modèle de menaces

Nous cherchons à sécuriser les communications dans les systèmes multi-agents d'agents embarqués pour permettre l'utilisation des SGC sans risque de falsification des échanges. Notre proposition repose sur les hypothèses sui-

vantes :

— Les primitives cryptographiques utilisées, le matériel sur lequel elles sont exécutées et leurs implémentations sont sécurisées.
— Un SGC approprié est exécuté et prend en compte les attaques dont il peut faire l'objet.
— Un protocole de routage est utilisé pour permettre aux agents de communiquer entre eux.

Suivant ces hypothèses, nous définissons le modèle d'attaquant comme étant un attaquant disposant de ressources similaires à celles des agents du système et qui aurait le contrôle du canal de communication. Il serait capable d'écouter, de bloquer et d'altérer tout message. De plus, nous ne faisons aucune supposition sur les intentions des autres agents.

3 Travaux connexes

Les travaux présentés dans [14] et [9] reposent sur l'utilisation de cryptographie à seuil ou sur une infrastructure sur mesure pour fournir une infrastructure à clefs publiques (ICP) décentralisée. Cependant, ces deux solutions nécessitent toujours une vérification initiale ou le préchargement de certificats pour fournir une authentification.
Dans [5], les auteurs proposent une ICP distribuée pour le contrôle de systèmes industriels via l'utilisation d'un framework d'agents qui nécessite un opérateur pour ajouter ou supprimer les agents. Ceci n'est pas compatible avec la caractéristique d'ouverture des systèmes que nous étudions.
Dans les travaux [1, 6], l'ICP décentralisée repose sur une table de hachage distribuée pour permettre la signature, le stockage et la certification des certificats. Bien que ces dernières approches résolvent le problème du consensus dans la gestion des certificats, elles ne fournissent pas de moyens de filtrer les nœuds indignes de confiance.
Dans l'ensemble, toutes les approches ci-dessus fournissent un moyen de décentraliser ou de distribuer une ICP, mais elles ne peuvent pas s'appliquer à notre problématique, car elles reposent trop sur des tiers ou sur un contrôle externe.
Plus récemment, la plupart des efforts visant à concevoir une ICP décentralisée impliquent l'utilisation d'une blockchain [17, 18, 19]. La blockchain a été conçue pour fournir un consensus sur les informations dans des systèmes décentralisés sans confiance préexistante, ce qui en fait une solution idéale pour délivrer, stocker, et révoquer les certificats, comme pour deux travaux précédemment cités. Pourtant, les blockchains actuellement déployées ne sont pas adaptées à notre problématique. La plupart utilisent la preuve de travail pour leur algorithme de consensus et ce dernier n'est pas du tout adapté aux systèmes embarqués. Quant aux solutions utilisant des algorithmes alternatifs, comme la preuve d'enjeu, elles se basent sur des blockchains publiques accessibles sur un réseau externe (Internet dans la majorité des cas) et non pas en local entre les nœuds du système.
La cryptographie basée identité ou attributs semble aussi être adaptable aux systèmes décentralisés [16, 8]. Cependant, cela implique une connaissance préalable sur les agents, ce qui est une hypothèse difficile à satisfaire dans des systèmes hétérogènes ouverts.
Par conséquent, aucune approche de notre état de l'art ne nous fournit une infrastructure existante répondant aux exigences de décentralisation, d'ouverture et d'autonomie du type de systèmes étudiés. C'est pourquoi nous fournissons dans notre travail les fondations d'une telle ICP à travers Multi-Agent Key Infrastructure (MAKI). MAKI a pour but l'établissement et le partage de clefs asymétriques pour permettre aux agents de communiquer de façon sécurisée. Elle s'appuie également sur le SGC pour déployer une auto-organisation résiliente contre les attaques internes et le renforce en sécurisant les communications ainsi qu'en rendant possible l'exclusion d'intrus.

4 Multi-Agent Key Infrastructure

L'utilisation de signatures cryptographiques permet d'assurer les trois caractéristiques de sécurité nécessaires à des communications sécurisées :

L'intégrité d'un message doit être maintenue pour éviter qu'un attaquant modifie les communications ;

L'authenticité d'un message est nécessaire pour éviter qu'un attaquant prenne l'identité d'un autre agent ;

La non-répudiation d'un message est nécessaire pour qu'un attaquant ne puisse pas mentir et revenir sur une de ses interactions.

Cela nécessite simplement que les agents génèrent une paire de clefs asymétriques et qu'ils les utilisent pour signer chaque communication. Ce mécanisme répond à lui seul à l'exigence de cryptographie décentralisée que nous avons fixée. Cependant, cela permet des comportements abusifs tels que des agents utilisant plusieurs paires de clefs en même temps ou changeant leurs clefs au fil du temps. Nous empêchons ces comportements en liant l'identité d'un agent à la clef publique qu'il utilise et en mettant à profit le SGC pour rendre inefficace le changement d'identité (voir Section 5.2). Nous complémentons l'action du SGC en ajoutant la possibilité de révoquer un agent, à travers son certificat, au lieu de simplement l'ignorer.

4.1 Architecture

Puisque nous nous concentrons sur les systèmes hétérogènes ouverts, nous ne nous attendons pas à ce que les agents possèdent des certificats avant même de rejoindre MAKI. Et nous ne voulons pas non plus appliquer des protocoles d'authentification spécifiques. Au lieu de cela, nous avons conçu MAKI pour ne pas nécessiter d'authentification. Les agents sont anonymes et ne sont classés bienveillants ou malveillants que sur la base de leurs actions. Ceci est possible grâce à la non-répudiation apportée par l'utilisation de signatures cryptographiques. Cela signifie que l'identité d'un agent est liée aux clefs qu'il utilise pour interagir.
Nous avons conçu MAKI comme une ICP légère et décentralisée. Des principes de l'ICP, nous n'avons conservé que la fonction d'autorité de certification (AC), l'utilisation obligatoire des certificats et la révocation des certificats.

Pour imposer l'utilisation de la cryptographie, les messages non signés ou dont la signature n'est pas valide doivent être ignorés. De plus, les agents doivent s'assurer que la source des messages qu'ils reçoivent détient bien un certificat valide pour la clef utilisée pour signer le message. Les requêtes d'existence d'ACs ou les demandes de certification peuvent être signées avec des clefs non certifiées.

Ces certificats sont délivrés et révoqués par des ACs définies par l'utilisation d'un algorithme d'auto-organisation, capitalisant ainsi sur l'autonomie des agents. Un agent peut librement demander un certificat et l'AC est autonome pour y répondre ou non. Elles sont également autonomes dans le choix de révoquer un agent, mais il est attendu qu'elles le fassent lorsque la majorité des agents en qui elles ont confiance le demande. La révocation est effectuée à l'aide de deux mécanismes. D'une part, chaque AC maintient et diffuse une Liste de Révocation de Certificats (LRC). Cette méthode est directe et instantanée, mais, en fonction des capacités du réseau, les mises à jour des LRC peuvent prendre du temps. Ainsi, pour atténuer ce risque, nous utilisons d'autre part des certificats à courte durée de vie, qui ne seront pas renouvelés par les ACs ayant connaissance de la révocation.

4.2 Organisation

Le bon fonctionnement de l'auto-organisation dépend des règles que MAKI ajoute au SGC. Par exemple, la façon dont nous évitons le problème des AC auto-signées, la probabilité qu'un agent malveillant devienne AC ou les récompenses d'une certification croisée sont toutes expliquées dans la Section 4.3.

Les agents peuvent ont un rôle parmi deux : None ou AC. None est le rôle par défaut et n'a aucune responsabilité envers l'ICP. Les ACs sont chargées de délivrer des certificats aux agents None et aux autres AC, de révoquer les certificats, de stocker et distribuer une liste des certificats délivrés et une LRC. Comme MAKI ne s'appuie pas sur des tiers pour établir une racine de confiance, les ACs sont toutes initialement auto-signées et peuvent ensuite utiliser la certification croisée pour créer un réseau d'ACs dignes de confiance.

Les ACs sont auto-élues. Les agents capables d'être AC (en fonction de leur capacité de calcul, de stockage, d'énergie...) décident eux-mêmes s'ils vont devenir AC. L'algorithme 1 décrit comment le choix peut être fait. Cet algorithme a été conçu avec deux objectifs : (i) chaque agent est proche d'une AC et (ii) les ACs ne doivent pas devenir des points de défaillance uniques. Il conduit à une distribution uniforme des ACs avec une ou plusieurs (en fonction de `T`, la probabilité qu'un agent décide de devenir une AC redondante) ACs par groupe d'agents. Il est possible d'adapter la définition de « proche » pour réduire le nombre d'ACs. Il y aura plus d'ACs si « proche » signifie être à portée de communication que si cela signifie être à trois fois la portée de communication. Il est également possible d'adapter la valeur de `T` pour augmenter ou diminuer le nombre de ACs redondants. Si `T` est très élevé, presque tous les agents qui peuvent être AC le seront, mais si `T` est très faible, seuls

Algorithme 1 Algorithme décrivant comment la décision de devenir un CA est prise.

```
T ∈ [0, 1) ▷ La probabilité qu'un agent décide de devenir une
AC même si d'autres ACs dignes de confiance sont proches.
Role ← None
CAs ← BROADCAST(CAListingRequest)
TrustedCAs ← FILTER(CAs, TrustLevel.Moderate)
if can become CA and (TrustedCAs is empty or RANDOM(0,
1) < T) then
    Role ← CA
```

les agents éloignés d'une AC le deviendront. La définition de « proche » et la valeur de `T` doivent être adaptées à l'application, à la densité et aux capacités de l'application sur laquelle fonctionne MAKI.

Le choix d'une AC pour un agent None est similaire à la décision de devenir une AC. Des situations de point de défaillance unique peuvent survenir si les tous agents choisissent l'AC la plus fiable et que cette AC est considérée comme la plus fiable pour la plupart d'entre eux. Pour éviter de tels cas, un agent choisira l'une de celles auxquelles il fait le plus confiance, mais pas spécialement la plus fiable. Cela se traduit par un choix aléatoire pondéré par les valeurs de confiance. Cette façon de choisir garantit que la plupart du temps, une AC hautement fiable sera choisie, sans que ce soit toujours la même. Et cela mène aussi à choisir quelque fois des ACs considérées comme moins fiables, leur laissant l'occasion de faire leurs preuves.

L'ajout d'un nouvel agent dans MAKI est simple. L'agent déterminera d'abord s'il doit devenir une AC et, si non, demandera un certificat à une AC. Une fois fait, il peut décider de chercher une AC plus fiable en demandant à ses voisins ou garder l'AC qu'il a choisie. Dans tous les cas, il diffusera son certificat pour s'assurer que ses voisins en prennent connaissance.

Les ACs auto-signées ne sont pas sensibles aux mécanismes de révocation. La seule façon de les exclure est de les ignorer et de suggérer aux nouveaux agents de les éviter. Une façon de supprimer cet avantage est d'encourager la certification croisée. Cela signifie que les AC pourraient demander à d'autres AC de signer leurs certificats, ce qui les rendrait susceptibles à la révocation. Cela ne présente aucun avantage intrinsèque pour l'AC recourant à ce mode de certification, car cela ne fait que rendre plus difficile l'obtention et le maintien d'un certificat valide, mais peut être considéré comme une preuve de bonne foi et être récompensé au niveau du SGC.

Globalement, MAKI ne réduit pas la sécurité apportée par le SGC puisque, même si elles ne peuvent pas être révoquées, les AC auto-signées peuvent toujours être ignorées et leur mauvaise réputation partagée avec les nouveaux agents. Cela signifie que le nombre d'agents malveillants que MAKI peut gérer ne dépend que du SGC et de la complexité des attaques exécutées contre lui. Nous montrons dans la Section 5.2 comment, avec un SGC simple, MAKI gère les agents malveillants une fois que le SGC détecte leurs comportements malveillants.

TABLE 1 – Compromis entre les risques et les avantages pour chaque interaction entre les agents, en fonction de leurs rôles

Interaction	Risque	Bénéfice	Confiance requise
	Autorité de Certification		
Délivrer un certificat	Modéré. Permettre à un agent malveillant de participer.	Haut. Augmenter sa légitimité.	Modérée ou aucune†
Révoquer un certificat	Haut. Perdre la confiance des agents en désaccord avec la révocation. Exclure un agent bienveillant.	Haut. Exclure un agent malveillant.	Modéré.
Demander une certification croisée	Haut. Voir sa réputation* diminuer si le certificateur n'est pas de confiance. Donner plus de légitimité à une AC malveillante.	Modéré. Voir sa réputation augmenter.	Haute
Accepter une demande de certification croisée	Haut. Donner plus de légitimité à une AC malveillante.	Haut. Voir sa réputation augmenter.	Haute
	None		
Demander un certificat	Modéré. Donner plus de légitimité à une AC malveillante. Avoir besoin de changer d'AC.	Haut. Posséder un certificat est obligatoire pour participer au système.	Modérée ou aucune‡

* Le terme « réputation » est utilisé ici pour décrire la confiance moyenne globale envers un agent.
† Les ACs n'ont pas de moyen de s'assurer qu'un agent nouvellement entrant est digne de confiance, elles laissent une chance à ces agents dans un premier temps.
‡ Un agent nouvellement entrant n'a pas de moyen de déterminer l'AC la plus digne de confiance, il fait donc un choix non éclairé dans un premier temps.

4.3 Gestion de confiance

MAKI n'est pas conçu pour un SGC spécifique. De plus, définir un modèle de confiance pour chaque cas d'utilisation spécifique de MAKI n'est pas possible. Au lieu de cela, nous spécifions ici comment MAKI s'appuie sur le SGC pour déployer son auto-organisation.

Nous présentons dans le Tableau 1 une évaluation des risques des interactions dans MAKI et les seuils de confiance recommandés à atteindre pour les réaliser. Ces seuils de confiance représentent le niveau de confiance qu'un agent doit avoir dans les autres agents pour interagir avec eux.

En complément des interactions présentées, tout agent peut demander et partager un certificat sans risque ni confiance requise, cela permet de vérifier ou de prouver que l'exigence de détention d'un certificat valide est satisfaite. Il en va de même pour la requête et le partage de la LCR qui, elle aussi, ne contient que des informations publiques.

MAKI exploite également le SGC pour atténuer la prolifération des ACs malveillantes en ajoutant un coût au rôle d'AC. Une AC ne peut être légitime que si elle répond continuellement aux demandes de certification, de révocation, de partage de la liste des certificats valides en cours qu'elle a distribués et de sa LRC De cette façon, même une AC malveillante doit contribuer au système pour garder son rôle. Cela peut traduit notamment par une légère augmentation de la confiance une AC chaque fois qu'elle répond à une demande de certification. De plus, la confiance accordée à un agent certifié est pondérée par la confiance accordée à l'AC qui a signé son certificat. Cela a pour but d'encourager les agents à choisir des ACs en lesquelles ils ont confiance, mais en lesquelles les autres agents font également confiance. Cela dans le but de faire en sorte que les ACs se comportent correctement avec chaque agent et pas seulement avec certains d'entre eux.

Si nous avons expliqué comment atténuer le risque que des agents malveillants deviennent ACs, et donc des ACs auto-signées, nous pouvons également proposer un moyen de réduire le nombre d'ACs auto-signées en ajoutant une récompense pour les ACs à certification croisée. Cette récompense de confiance encouragera les ACs se certifier entre elles, prenant donc le risque de d'avoir leurs certificats révoqués et d'être exclues, pour garder, voire augmenter, leur légitimité.

4.4 Gestion des certificats

Les représentations ASN.1 [10] du certificat et de la LRC données en Figure 1. Le format du certificat est une version simplifiée du format X.509 dont les principales différences sont l'inclusion de la clef publique de l'émetteur, puisqu'elle fait partie de son identité, l'inclusion le champ supplémentaire, `subjectInfo`, qui est laissé à la discrétion des concepteurs du système, et la suppression de plusieurs champs non utiles pour notre ICP. En utilisant ce for-

```
Identity ::= SEQUENCE { name INTEGER, publicKey BIT STRING }
```

(a) Représentation ASN.1 du champ Identity utilisé dans le certificat et la LRC de MAKI.

```
Certificate ::= SEQUENCE {
  version [0]  INTEGER,
  serialNumber INTEGER,
  signature    BIT STRING,
  issuer       Identity,
  validity     SEQUENCE {
                 notBefore UTCTime,
                 notAfter  UTCTime
               },
  subject      Identity,
  subjectRole  Role,
  subjectInfo  SubjectInfo
}
Role ::= INTEGER { NONE(0), CA(1) }
```

(b) Représentation ASN.1 du certificat de MAKI.

```
CertificateRevocationList ::= SEQUENCE {
  version [0]         INTEGER,
  signature           BIT STRING,
  holder              Identity,
  thisUpdate          UTCTime,
  revokedCertificates SEQUENCE OF SEQUENCE {
    serialNumber        INTEGER,
    issuer              Identity,
    subject             Identity,
    revocationDate      UTCTime,
    reasonCode          ReasonCode
  }
}
ReasonCode ::= ENUMERATED {
   idCompromise(0),
   cessationOfOperation(1)
}
```

(c) Représentation ASN.1 de la LRC de MAKI.

FIGURE 1 – Représentation ASN.1 du certificat et de la LRC de MAKI.

mat, avec un champ `subjectInfo` vide, un `time_t` de 4 octets pour `UTCTime`, 193 octets pour la clef publique au format OpenSSH, 105 octets pour la signature brute, 1 octet pour `Version`, `Role` et `ReasonCode`, et 2 octets pour `SerialNumber` et `Name`, la taille d'un certificat est de 507 octets.

Comme MAKI ne s'appuie pas sur les autorités d'enregistrement pour délivrer et distribuer les certificats, la distribution des certificats incombe aux agents eux-mêmes. Les agents peuvent diffuser leurs certificats périodiquement et doivent les joindre aux premiers messages de chaque échange. Un agent peut également demander le certificat d'un autre agent. Ces méthodes de distribution sont moins efficaces que la collecte et le partage des certificats par un tiers, mais elles permettent une meilleure évolutivité et décentralisation tout en éliminant tout risque provenant de l'utilisation de cette tierce partie ainsi que tout risque de création de points de défaillance unique.

En ce qui concerne le format de la LRC, nous avons intégré le champ `reasonCode`, jusque-là optionnel, dans les champs obligatoires afin que les AC puissent être tenues responsables de chaque révocation. Nous réservons l'utilisation de la LRC à l'exclusion des agents, ainsi, parmi les dix valeurs possibles, nous n'avons conservé que les champs `KeyCompromise` (renommé `IdCompromise`) et `CessationOfOperation`. D'autres codes pourraient être ajoutés pour indiquer des comportements malveillants spécifiques à l'application. Avec les mêmes choix de format que pour le format de certificat, une LRC contenant $n \in \mathbb{N}$ certificats est de $305 + n \times 397$ octets. La distribution des LRC sont font, avec ou sans demande auprès des ACs.

5 Validation

5.1 Vérification de modèles

Nous avons employé une technique de vérification de modèles pour nous assurer que l'auto-organisation décrite en Section 4.2, partie centrale de MAKI, est valide. Pour cela, nous avons utilisé l'outil Model Checking for Multi-Agent Systems (MCMAS) [15] pour vérifier trois propriétés :

(A) Si l'organisation comprend au moins une AC, chaque agent finira par posséder un certificat valide.

(B) À partir d'un ensemble de None, où au moins un agent possède la capacité de devenir AC, une organisation comportant au moins une AC émerge.

(C) À partir d'une organisation comportant au moins une AC, si tous les ACs disparaissent, le système se réorganise pour retrouver une organisation à un moins une AC.

En utilisant l'enchaînement de (B) puis (A) ou (C) puis (A) nous pouvons être assurés que l'auto-organisation permet bien à chaque agent d'avoir l'opportunité d'obtenir un certificat, un prérequis à la participation dans le système.

La Figure 2 donne un aperçu du modèle d'un agent (Agent1) appliquant l'algorithme d'auto-organisation. Notamment, dans le cas où il reste None, il s'appuie sur un autre agent (Agent2) pour lui délivrer un certificat. L'ensemble des modèles sont disponibles sur le dépôt [3].

Néanmoins, MCMAS ne permet pas de modéliser les évolutions de confiances entre les agents, les résultats que nous obtenons avec ne concernent que des situations où les agents sont bienveillants. De plus, des problématiques, par exemple liées à l'exécution asynchrone des agents, ne sont pas prises en compte.

Ainsi, bien que certaines propriétés peuvent être validées avec une forte confiance en la preuve, ici on fait confiance en MCMAS pour produire des résultats justes. Tester lors d'exécutions est aussi nécessaire pour valider les aspects que nous n'avons pas pu modéliser.

5.2 Preuve de concept

Pour valider le principe général et nous assurer de la faisabilité de MAKI, nous avons développé une preuve de concept,

```
Agent Agent1
 ...
 Protocol:
  -- CA behavior
  role = CA and cert = none : { self_sign };
  role = CA and cert = self_signed and advertised = false : { advertise };
  role = CA and cert = self_signed and advertised = true and cert_asked = true : { deliver_cert };
  -- None behavior
  role = None and cert = none and cert_asked = false and knows_ca = true : { ask_cert };
  Other : { wait };
 end Protocol
 Evolution:
  -- CA evolution
  cert = self_signed if Action = self_sign;
  advertised = true if Action = advertise;
  cert_asked = true if Agent2.Action = ask_cert;
  cert_asked = false if Action = deliver_cert;
  -- None evolution
  cert_asked = true if role = None and Action = ask_cert;
  knows_ca = true if role = None and  Agent2.Action = advertise;
  cert = signed if role = None and Agent2.Action = deliver_cert;
 end Evolution
end Agent
```

FIGURE 2 – Aperçu synthétique d'un modèle MCMAS d'un agent MAKI.

disponible à [3], à l'aide de l'environnement de développement Mesa [12].

Paramétrage Concernant les choix cryptographiques, nous avons suivi les recommandations du NIST [2]. Ainsi, nous avons choisi l'algorithme de signature Elliptic Curve Digital Signature Algorithm, employé avec des clefs de taille 256-bits et basé sur la courbe P-256.

Nous avons aussi eu à choisir un modèle de confiance. Pour cela, nous avons un modèle simple, suffisant pour satisfaire les prérequis de MAKI. Ce modèle comprend une valeur de confiance initiale faible, une contremesure standard pour limiter les changements d'identité fréquents et le maintient simultané de plusieurs identités, ainsi que trois seuils de confiance : *Low*, *Moderate* et *High*. La confiance croit en suivant la fonction donnée en Équation 1 et décroit instantanément à 0 à la moindre détection de malveillance.

$$f : x \mapsto \frac{x}{x + 10} \tag{1}$$

La valeur 10 a été déterminée expérimentalement en fonction des autres paramètres de la simulation, notamment sa durée.

Les seuils *Low*, *Moderate* and *High* ont respectivement été fixés à 0.3, 0.7 et 0.9 et la valeur initiale a été fixée à *Low*. Dans ce modèle, une valeur de confiance inférieure à *Low* implique que l'agent est ignoré.

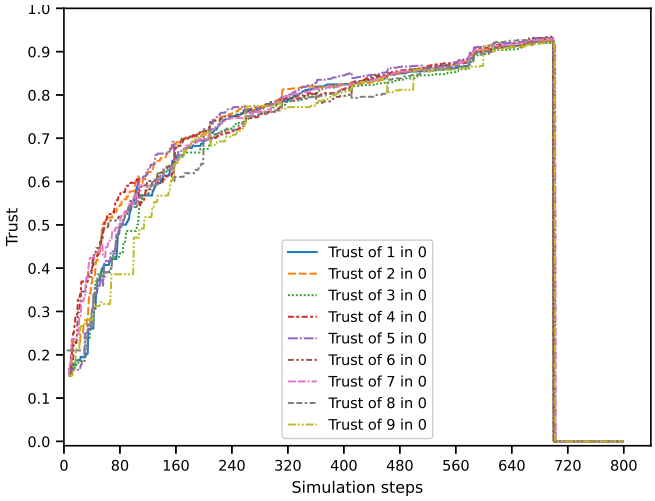
Le modèle inclut aussi les informations de confiance indirectes : un agent peut demander aux autres participants ce qu'ils pensent d'un certain agent. De plus, il est attendu que les agents écoutent toutes les communications pour s'assurer de la cohérence entre les dires et les actions de ses voisins, notamment concernant les distributions de certificats. Cela reste possible car nous ne cherchons pas à protéger la confidentialité des échanges propres à MAKI, ceux-ci restent publiques.

Puisque nous faisons l'hypothèse de la présence d'un protocole de routage correct, nous avons placé les agents de façon à ce qu'ils soient tous à portée de communication les uns des autres et avons développé un protocole très minimaliste dans lequel tous les messages sont diffusés, sans risque de perte et aucun accusé de réception n'est attendu. Néanmoins, les agents ne supposent pas que chaque demande recevra une réponse en temps voulu, voire reçue tout court. Enfin, nous faisons augmenter la confiance des agents de façon aléatoire au fil du temps pour émuler des interactions réussies et alimenter le SGC. Les interactions malveillantes sont, elles aussi, émulées : la détection d'un comportement malveillant est arbitraire et ne sert qu'à déclencher les processus de protection de MAKI, la révocation de certificats et la réorganisation.

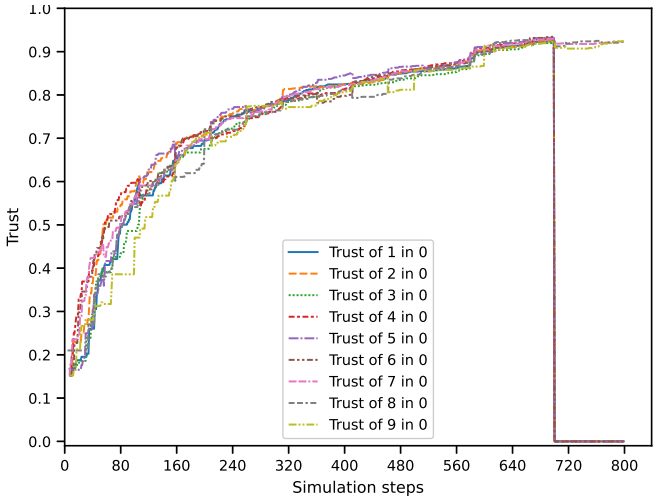
Exécutions Nous avons réalisé plusieurs simulations sous plusieurs scénarios d'attaque et en présentons deux ici. Chaque simulation comprend dix agents, $\mathcal{A}_0$ à $\mathcal{A}_9$, avec seulement $\mathcal{A}_6$, $\mathcal{A}_7$, $\mathcal{A}_8$ et $\mathcal{A}_9$ en capacité d'endosser le rôle d'AC.

Le premier est un scénario simple : l'agent $\mathcal{A}_0$ est détecté comme malveillant à l'étape 700 par $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$, $\mathcal{A}_4$, $\mathcal{A}_5$, et $\mathcal{A}_6$. Ces agents demandent et obtiennent ensuite la révocation du certificat d'$\mathcal{A}_0$, ce mène ce dernier à perdre la confiance du reste du système. La courbe de la confiance en $\mathcal{A}_0$ est donnée en Figure 3a. On y voit bien que la révocation d'un certificat mène à l'exclusion de l'agent certifié. En comparaison, la Figure 3b montre l'évolution de la confiance en désactivant la révocation des certificats. On y voit que, même si la confiance en $\mathcal{A}_0$ diminue un peu de par l'utilisation d'informations indirectes, seule la révocation permet une réelle exclusion de l'agent malveillant.

Le second scénario présenté correspond à une attaque coordonnée de l'ensemble des ACs à un instant donné : à l'étape 700 $\mathcal{A}_6$, $\mathcal{A}_8$ et $\mathcal{A}_9$, les ACs, sont tous détectés comme malveillants par le reste des agents. Ces trois agents sont donc



(a) Évolution de la confiance en $\mathcal{A}_0$ en fonction du temps avec révocation de son certificat.



(b) Évolution de la confiance en $\mathcal{A}_0$ en fonction du temps sans révocation de son certificat.

FIGURE 3 – Graphe montrant l'évolution de la confiance en $\mathcal{A}_0$ en fonction du temps avec et sans révocation de son certificat.

```
1098 main:step:701
1099 agent-7:<>:CertAdvert(Certificate(serial_number: 0, issuer: Identity(7, <public_key>), subject:
         Identity(7, <public_key>), subject_role: Role.CA, not_before: 702, not_after: 4702, version:
         1))
1100 DEBUG:agent:0:net:=>:Data(dest: Identity(7, <public_key>), payload: CertReq())
1101 DEBUG:agent:1:net:=>:Data(dest: Identity(7, <public_key>), payload: CertReq())
1102 DEBUG:agent:3:net:=>:Data(dest: Identity(7, <public_key>), payload: CertReq())
1103 DEBUG:agent:2:net:=>:Data(dest: Identity(7, <public_key>), payload: CertReq())
1104 DEBUG:agent:4:net:=>:Data(dest: Identity(7, <public_key>), payload: CertReq())
1105 DEBUG:agent:5:net:=>:Data(dest: Identity(7, <public_key>), payload: CertReq())
```

FIGURE 4 – Extrait simplifié de la trace d'exécution montrant la réorganisation du système après la perte de confiance dans les ACs.

ignorés, mais le système se retrouve sans AC de confiance. $\mathcal{A}_7$ change donc de rôle et devient AC. Voyant qu'un nouveau AC de confiance est disponible, le reste des agents font appelle à lui. Une partie de la trace d'exécution montrant cette réorganisation est donnée en Figure 4. On y voit $\mathcal{A}_7$ diffuser son nouveau certificat et $\mathcal{A}_0$, $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$, $\mathcal{A}_4$, $\mathcal{A}_5$ leur faire une requête de certificat. Les règles d'auto-organisation de MAKI permettent bien au système de réparer même si tous ses ACs disparaissent ou s'avèrent non dignes de confiance.

6 Conclusion

Dans cet article, nous avons introduit une infrastructure à clefs publiques décentralisée, MAKI, adaptée aux systèmes multi-agents ouverts d'agents embarqués. Cette infrastructure permet de l'utilisation de primitives cryptographiques pour sécuriser les communications entre agents ainsi que d'exclure de possibles intrus. L'exclusion se fait par l'utilisation de certificats délivrés et révoqués par des agents autorité de certification. Ces agents maintiennent un réseau d'autorités dignes de confiance sans requérir à de tierces parties grâce à un système de gestion de confiance. Une preuve de concept de MAKI montrant l'intérêt de l'utilisation de certificats ainsi que sa capacité d'adaptation lors d'attaques est présentée. Une description du processus de validation de l'auto-organisation par vérification de modèles est aussi présentée. Le code source de la preuve de concept et les modèles utilisés pour la validation sont disponibles en ligne [3].
Nous comptons finaliser la validation de MAKI par des simulations basées sur la preuve de concept. Nous explorons également une solution basée sur la blockchain pour fournir un moyen de, plus facilement, partager leurs certificats et d'auditer les autorités de certification.

Remerciements

Ce travail a bénéficié d'une aide de l'État gérée par l'Agence Nationale de la Recherche au titre du programme « Investissements d'avenir » portant la référence ANR-15-IDEX-02.

Références

[1] Agapios Avramidis, Panayiotis Kotzanikolaou, Christos Douligeris, and Mike Burmester. Chord-PKI : A distributed trust infrastructure based on P2P networks. *Computer Networks*, 56(1) :378–398, 2012.

[2] Elaine Barker and Quynh Dang. Recommendation for Key Management Part 3 : Application-Specific Key Management Guidance, 2015.

[3] Arthur Baudet. Code and data presented in jfsma 2023, 2023. https ://doi.org/10.5281/zenodo.7689499.

[4] Arthur Baudet, Oum-El-Kheir Aktouf, Annabelle Mercier, and Philippe Elbaz-Vincent. Systematic Mapping Study of Security in Multi-Embedded-Agent Systems. *IEEE Access*, 9 :154902–154913, 2021.

[5] Sergi Blanch-Torné, Fernando Cores, and Ramiro Moreno Chiral. Agent-based PKI for Distributed Control System. In *2015 World Congress on Industrial Control Systems Security (WCICSS)*, pages 28–35, 2015.

[6] Xavier Bonnaire, Rudyar Cortés, Fabrice Kordon, and Olivier Marin. A Scalable Architecture for Highly Reliable Certification. In *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pages 328–335, 2013.

[7] Djallel Eddine Boubiche, Samir Athmani, Sabrina Boubiche, and Homero Toral-Cruz. Cybersecurity Issues in Wireless Sensor Networks : Current Challenges and Solutions. *Wireless Personal Communications*, 117(1) :177–213, 2021.

[8] Hui Cui and Robert H. Deng. Revocable and Decentralized Attribute-Based Encryption. *The Computer Journal*, 59(8) :1220–1235, 2016.

[9] Nicholas C. Goffee, Sung Hoon Kim, Sean Smith, Punch Taylor, Meiyuan Zhao, and John Marchesini. Greenpass : Decentralized, PKI-based Authorization for Wireless LANs. In *In 3rd Annual PKI Research and Development Workshop*, pages 26–41, 2004.

[10] ITU-T. X.680-X.693 : Information Technology - Abstract Syntax Notation One (ASN.1) & ASN.1 encoding rules, 2021.

[11] Rutvij H. Jhaveri and Narendra M. Patel. Attack-pattern discovery based enhanced trust model for secure routing in mobile ad-hoc networks. *International Journal of Communication Systems*, 30(7) :e3148, 2017.

[12] Jackie Kazil, David Masad, and Andrew Crooks. Utilizing Python for Agent-Based Modeling : The Mesa Framework. In *Social, Cultural, and Behavioral Modeling*, volume 12268, pages 308–317, 2020.

[13] Deepika Kukreja, S. K. Dhurandher, and B. V. R. Reddy. Power aware malicious nodes detection for securing MANETs against packet forwarding misbehavior attack. *Journal of Ambient Intelligence and Humanized Computing*, 9(4) :941–956, 2018.

[14] Francois Lesueur, Ludovic Me, and Valerie Viet Triem Tong. An efficient distributed PKI for structured P2P networks. In *2009 IEEE Ninth International Conference on Peer-to-Peer Computing*, pages 1–10, 2009.

[15] Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. MCMAS : an open-source model checker for the verification of multi-agent systems. *International Journal on Software Tools for Technology Transfer*, 19(1) :9–30, 2017.

[16] Tatsuaki Okamoto and Katsuyuki Takashima. Decentralized Attribute-Based Encryption and Signatures. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E103.A(1) :41–73, 2020.

[17] Bo Qin, Jikun Huang, Qin Wang, Xizhao Luo, Bin Liang, and Wenchang Shi. Cecoin : A decentralized PKI mitigating MitM attacks. *Future Generation Computer Systems*, 107 :805–815, 2020.

[18] Ankush Singla and Elisa Bertino. Blockchain-Based PKI Solutions for IoT. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pages 9–15, 2018.

[19] Alexander Yakubov, Wazen M. Shbair, Anders Wallbom, David Sanda, and Radu State. A Blockchain-Based PKI Management Framework. In *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–6, 2018.