Artif Intell Rev (2014) 42:607–636
DOI 10.1007/s10462-012-9343-1

A review of attacks and security approaches in open multi-agent systems

Shahriar Bijani · David Robertson

Published online: 16 May 2012

Abstract Open multi-agent systems (MASs) have growing popularity in the Multi-agent Systems community and are predicted to have many applications in future, as large scale distributed systems become more widespread. A major practical limitation to open MASs is security because the openness of such systems negates many traditional security solutions. In this paper we introduce and classify main attacks on open MASs. We then survey and analyse various security techniques in the literature and categorise them under prevention and detection approaches. Finally, we suggest which security technique is an appropriate countermeasure for which classes of attack.

Keywords Security · Multi-agent system (MAS) · Open MAS · Attack taxonomy · Attack detection · Attack prevention · Lightweight Coordination Calculus (LCC)

1 Introduction

An *open system* is a system that allows new components, which may have been created by different parties or for different objectives, not known at design time, to interact at run-time (Poslad and Calisti 2000). An open *multi-agent system* (MAS) is an *open system* in which agents can join and leave freely (Demazeau and Rocha Costa 1996). Open MASs have growing popularity in the Multi-agent Systems community and are predicted to have many applications in the future. Artikis et al. (2009) have argued that there is a growing interest in open MASs.

S. Bijani (✉) · D. Robertson
Informatics School, Edinburgh University, 10 Crichton St., Edinburgh EH8 9AB, UK
e-mail: s.bijani@ed.ac.uk

D. Robertson
e-mail: dr@inf.ed.ac.uk

S. Bijani
Computer Science Department, Shahed University, Persian Gulf Highway, Tehran, Iran

Although openness in open MASs makes them attractive for different new applications, new problems emerge, among which security is a key. The more these systems are used in the real world, the more the necessity of their security will be obvious to users and system designers. Unfortunately there remain many potential gaps in the security of open multi-agent systems and little research has been done in this area, leaving systems vulnerable.

A MAS could be defined as a subcategory of software systems, a high level application on top of the OSI networking model; therefore the security of MASs is not a completely different and new concept; it is a sub-category of computing security. Also some traditional security mechanisms resist use in MAS directly, because of the social nature of MASs and the consequent special security needs (Robles 2008). In open MASs the autonomy, openness and independence of agents from human users, brings about new threats, undetectable using traditional security mechanisms that consider the lower network layers. Open MASs are particularly difficult to protect, because we can make only minimum guarantees about the identity and behaviour of agents and conventional security mechanisms, like authentication and encryption, are at best a small (though necessary) part of the solution. Focussing on MAS security as a whole should not undermine the necessity of applying conventional security solutions for the lower network layers.

Considering the agents' interaction model, open MASs can be divided into three types with different security issues. In the first category it is presumed that agents are completely autonomous and there is no assumption about protocols except that messages are passed (maximum autonomy and flexibility). In this case, since neither is information available about protocols nor about agents, we cannot ensure the security of interactions beyond assuming that agents individually may manage to guess or just rely on the security of lower layers. In the second type, there is one fixed protocol (an electronic institution) for agent communication. This is closer to traditional security models than the first type and is more likely to standardise by means of conventional security mechanisms and reasoning about protocol security properties but is too rigid for many applications of knowledge sharing in MASs.

The third class of MASs utilises multiple protocols for different applications so that agents may play roles in various electronic institutions. In the most general case, agents may invent the protocols themselves and share them with others or use other (unknown) agents' protocols; e.g. the *OpenKnowledge* system (Robertson et al. 2008).We have focused on the issue of security for this form of dynamic interactions, but in the present review, we have emphasised common aspects where we are able to generalise our security analysis to other types of Open MAS.

In this review, we mainly address the security of open MASs, however we do not focus on security issues that are specific to the mobility of mobile agents i.e. attacks from hosts on agents and vice versa. Many papers on the security of MASs have been presented in the literature, although only a few have focused on open MASs. In addition, most research has dealt with mobile agents security issues either directly or indirectly and many of the security solutions have been proposed for threats from agents to hosts or from hosts to agents.

There have been many attempts to protect mobile agents from the host platform in the literature (Bierman and Cloete 2002; Jansen and Karygiannis 2000; Oey et al. 2010); some are based on cryptography while others are not; e.g. code obfuscation (Majumdar and Thomborson 2005), function encryption (Lee et al. 2004), environmental key generation (Riordan and Schneier 1998), execution tracing (Tan and Moreau 2002), and agent monitoring (Page et al. 2005). Another important security issue in mobile agent systems is protecting the agent platform from mobile agents. Some example techniques are: Proof Carrying Code (Necula and Lee 1998), sandboxing (Wahbe et al. 1993) and code signing (Jansen and Karygiannis 2000). However, the importance of the security issues come from mobility of agents

should not diminish the importance of many other security threats in open MASs and we will not concentrate on them any further.

In the following sections, we first suggest an attack taxonomy and a risk assessment of threats to open MASs (Sect. 2), then we review and discuss security approaches in MASs (Sect. 3) and finally summarise security mechanisms that provide countermeasures to each attack on open MASs (Sect. 4).

2 Attacks on open MAS

Studying all possible attacks on open MASs is too broad an area. We assume that some confidential and important information exists in the open MAS (in order that attacks are worthwhile) and the interface between the agent platform and the agent is secure (so as to limit scope of the topic and to focus on attacks by agents on other agents rather than on the infrastructure). In this section, we first introduce the LCC language, which is used to formulate some attacks in the following section, and then describe important attacks on open MAS.

2.1 Lightweight coordination calculus (LCC)[1]

In our security analysis, Lightweight Coordination Calculus (LCC) is used to implement interaction models and formulate attacks. An interaction model in LCC is defined as a set of clauses, each of which specifies a role and its process of execution and message passing. The LCC syntax is shown in Fig. 1.

For different applications, agents may use their own interaction model or download an existing one. When an agent selects an interaction model and a role in it, it should behave as the interaction model dictates.

Figure 2 illustrates an example of an interaction model for a simple communication in LCC. There are two roles (clauses) in this interaction model: *requester* and *informer*. In the first clause, a requester asks about something from an informer, then gets an answer from it and then continues as a requester. In the second clause, an informer is asked by a requester and then it should tell the requester if it knows the answer.

2.2 Attack taxonomy

As a part of our security analysis, we categorised various attacks on open MAS. Attack or vulnerability taxonomies are designed for different purposes (Igure and Williams 2008): (1) to develop automated tools for performing security assessment, (2) to provide a way to explore unknown attacks and (3) to understand the attacks' implications and the defence mechanism against them. The latter is the main goal of our attack classification, which is valuable for the prevention, detection and response to potential attacks.

Attacks on (open) MASs can be categorised in different ways, however we suggest a three-layer taxonomy illustrated in Fig. 3. The first level of our classification is based on violations against the four main security properties i.e. confidentiality, integrity, availability and accountability. In the second level we categorise attacks based on the novelty of the technique used to run an attack, namely, traditional and modern attack techniques. The third

[1] LCC (Robertson 2005) is a compact executable specification to describe the notion of social norms and it is based on logic programming.

```
Interaction Model := {Clause,...}

Clause := Role::Def

Role := a(Type, Id)

Def := Role | Message | Def then Def| Def or Def | null← C

Message:= M ⇒ Role | M ⇒ Role ← C | M ⇐ Role | M ⇐ Role ← C

C:= Constant | P(Term,...)| ¬ C | C∧C | C∨C

Type := Term

Id := Constant | Variable

M:= Term

Term:= Constant | Variable | P(Term,...)

Constant:= lower case character sequence or number

Variable := upper case character sequence or number
```

Fig. 1 LCC language syntax; principal operators are: messages (⇒ and ⇐), conditional (←), sequence (then) and committed choice (or)

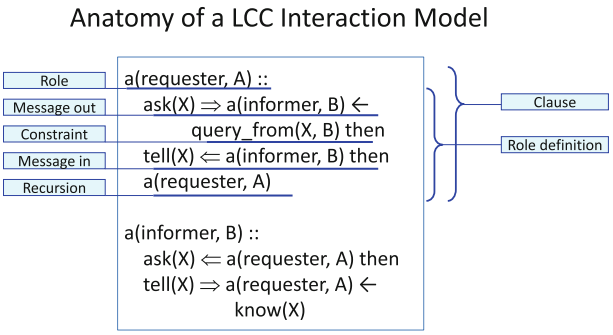

Fig. 2 An example of a LCC interaction model

level of attack classification is grounded on the attack target, which could be sender agents, recipient agents and transferring information.

Some dimensions to attack classification might be to some extent fuzzy, for example, in the second level, some modern methods could be categorised as traditional. However it does not affect our classification objective. We intend not to have mutually exclusive attack classes, so there might be some attacks that could be categorised in more than one class. This taxonomy might be tailored for each open MAS with regards to its implementation and application.

The four classes in the first level of taxonomy are: (1) *disclosure attack*, (2) *modification attack*, (3) *denial of service attack* and (4) *fake identity attack*. We describe the main attacks introduced in the taxonomy in the following sub-sections.

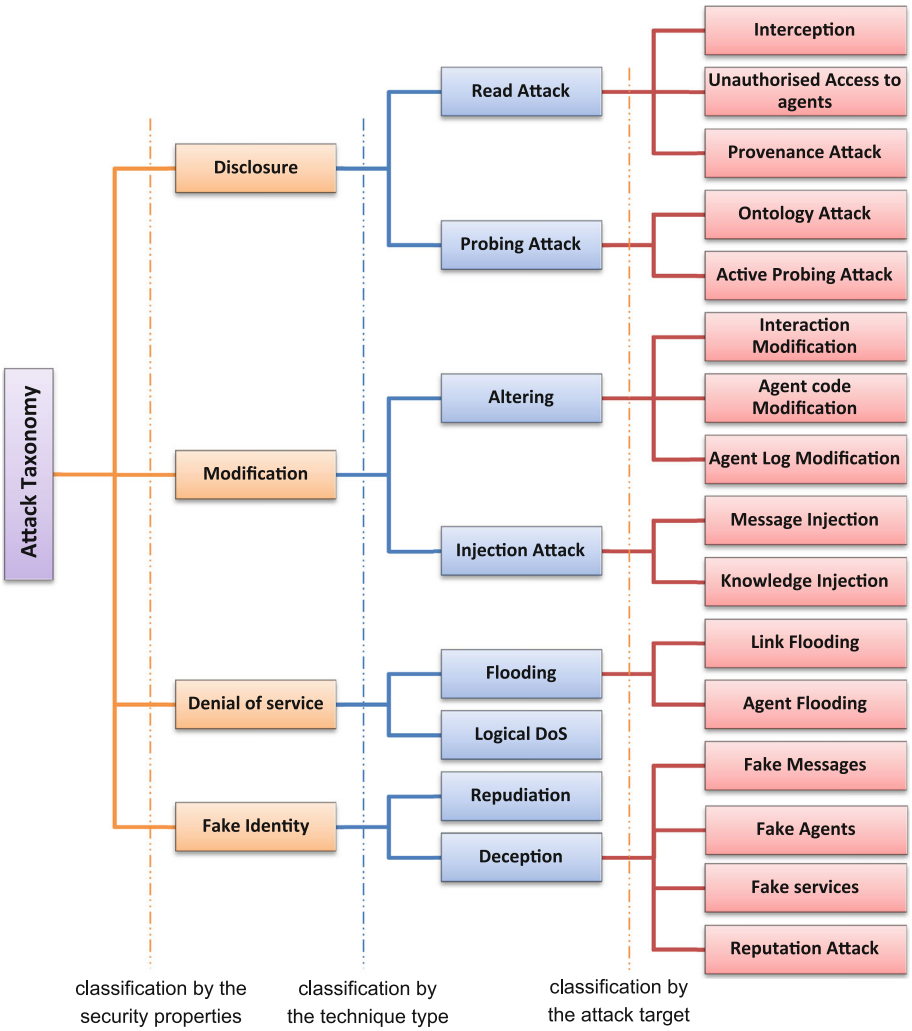

Fig. 3 Taxonomy of important attacks on open MASs

2.2.1 Disclosure

Although open MASs are usually open to all, this does not mean that there is no confidential information in which malevolent people are interested. In the *disclosure* attack, the attacker tries to access confidential information. These attacks are of two types: Firstly, conventional *read attacks* similar to those in traditional computer security literature. Based on the attacker target we have various read attacks:

(a) Interception: unauthorised access to confidential interaction information, e.g. interception of communicating messages between agents.
(b) Unauthorised access to agent: the agent's information (including its state and code) and the agent's local knowledge could be disclosed to an adversary. In mobile agents

```
a(ontologyAttacker(S, Os, Of), A) ::
( ask(subclasses(C)) ⇒ a(ontologyService, O) ← S = [C|Sr] then
tell(subclasses(C,Sc)) ⇐ a(ontologyService, B) then
a(ontologyAttacker(Sn,On,Of),A) ← merge(Sr, Sc, Sn)and merge(Os, Sc, On)
or
null← S = [] and Of = Os

a(ontologyService, B) ::
ask(subclasses(C)) ⇐ a(ontologyAttacker(_,_,_), A) then

tell(subclasses(C,Sc)) ⇒ a(ontologyAttacker(_,_,_), A) ←
known_subclasses(C, Sc) then
a(ontologyService, B)
```

Fig. 4 A sample interaction model of an ontology attack: an attacker 'A' starts from the ontology root (S) and rebuilds the ontology (Of) by asking for subclasses of each node from the ontology service 'B'. Initially 'S' = 'Os' = the ontology root. The merge function merges the two first arguments and into the third argument. Finally, the copied ontology 'Os' will save into 'Of'

systems there are threats from the host like reverse engineering and theft of mobile agent's information (Bierman and Cloete 2002).

(c) Provenance attack: attack on provenance information of agents and their communications. An example of attack on provenance information is when agent A sends message $M1$ to receiver B in an open interaction, maybe A, B and $M1$ were not private to a malicious agent, who is monitoring the interaction, but probably the history of the sent messages from A or the agents' names who has interacted with A would be confidential information. Disclosure of a mobile agents' itinerary information to an adversary is another example of threat against open MAS.

Secondly, *probing attacks*, in which an adversary accesses confidential information by analysing the results of the sent queries. Whether the attacker target is an ontology of a MAS or not, we can classify the probing attack into two types: (a) ontology attack and (b) active probing attack.

Use of ontologies in MASs is currently popular especially in knowledge-intensive applications. In an *ontology attack*, the attacker accesses the whole ontology by sending many queries to a semi-open ontology-based system. In the latter attack, it is assumed that the whole ontology is confidential but it is also open to questions about small parts of itself. The attacker may find the entire information and relations by asking intelligent and complementary queries based on its knowledge of the semantics of the ontology representation language being used by the attached agent (Fig. 4).

Another type of probing attack is active probing attack, in which an adversary accesses the private local knowledge (e.g. decision rules and policies) of the victim agent by injecting facts to the agent's knowledge-base, asking queries and analysing results. Bijani et al. (2011) redefined the concept of probing attack in conventional network security to be applicable in MASs and suggested four type of probing attacks. In probing attack sometimes the attacker injects information to a MAS and analyses the reaction of the system.

2.2.2 Modification

This class of attack is against the integrity of the system. An integrated system guarantees that information has not been tampered with during handling (Mitchell 2003). A *modification* attack on an open MAS happens when a malevolent agent modifies a piece of information in the system. Based on the novelty of the attack technique, the *modification* attack on open MASs could be classified into *altering* and *injection* attack.

We classified *altering* attack based on the attack target, which could be the interaction, the agent itself, or the agent's log files, into: (a) modification of the agents' interactions by altering the transferring information, (b) altering the agent code, data and configuration and (c) altering the event logging system of a MAS. A malicious agent can exploit general vulnerabilities like *buffer overflow* to perform the attack.

An example of *interaction modification* attack is similar to the one in peer to peer systems called the *colluded truncation attack* (Yue et al. 2009) in which two or more malicious agents, who surround a sender agent, collude to modify a sent message. Silei et al. (2008) explained use the same idea in an attack from host to mobile agent's code. *Modification* attacks are usually dependent upon intruding on low level network communication layers.

Another type of *modification* attack is *injection* attack, which is similar to *active probing attack*. The objective of the adversary in injection attack is twofold: to infuse forged information into MAS and to inject some facts into the system to be able to infer confidential information. *Injection* attack is classified into *message injection* and *knowledge injection*. In the *message injection* attack, an adversary injects fake messages or malicious interaction models into an open MAS to change or control the interaction between agents while in *knowledge injection* untruthful facts are added into an agent's knowledge base to affect the agent decisions (Bijani et al. 2011).

There also could be an active probing attack on an open MAS that use ontology merging. An intelligent adversary can build a malicious ontology and add (inject) it to the ontology merging procedure. As a result of merging, a new ontology emerges that leaks confidential information.

2.2.3 Denial of service (DoS)

In a *denial of service* attack on MASs the attacker attempts to prevent the system to provide the intended services to its legitimate users. The goal of *DoS* may be wasting other agents' resources, delaying the service, making real users forsake the system or ruining the system's reputation. A malicious agent may attack just one agent or a group of agents. When more than one attacker collaborates in a *DoS attack*, this is called a *distributed denial of service (DDoS)*. As in computer networks (but here, with a different meaning) a *DoS* attack can be divided into two types (Traynor et al. 2008): *flooding* and *logical* attacks.

Flooding DoS happens when too many messages are sent to one or more agents to overwhelm the victim agent or the connection to an agent by consuming the agent or network resources (e.g.: the agent's message buffer or the communication bandwidth).

While *flooding* is almost a blind attack, a *logical DoS* uses more sophisticated methods to exploit software or system bugs. In open MASs with dynamic interaction models, any user can publicise an interaction model, a malevolent agent may publish an interaction model to make other agents send many messages to each other (Fig. 5), do worthless tasks (Fig. 6) or remain in infinite loops. Another kind of *logical DoS* is that one hostile agent does not play its role correctly to terminate or interrupt a popular interaction. In (Sit and Morris 2002) a similar attack called *store and retrieve* is introduced in peer to peer systems. *Inconsistent*

```
a (attacker(PeerList), A) ::
       null ←PeerList = []
  or (
        m(PeerList) ⇒  a(zombie,Z) ←PeerList=[H|T]
         then  a(attacker(T), A)
    )
a(zombie, Z)::
       m(PeerList)⇐ a(attacker(_), A) then a(attacker(PeerList),Z)
```

Fig. 5 An example of a simple interaction model for *logical DoS*: an attacker sends a message 'm' including the list of gents to all agents. Each agent who receives this message changes its role to an attacker and starts sending messages to others

```
a (attacker(PeerList, K), A) ::
       null ←PeerList = []
  or (
        trigger(K) ⇒  a(zombie, Z) ←PeerList=[H|T]
        then  a(attacker(T,K), A)
    )

a(zombie, Z)::
      trigger(K) ⇐ a(attacker(_), A) then  a(in-loop(K),Z)

a(in_loop(K),Z)::

null← K = []       or

a(in_loop(K1),Z) ← K1 =[a|L]
```

Fig. 6 Another example of a simple interaction model for *logical DoS*: an attacker sends a trigger message including a number 'K' to all agents. Each agent who receives this message, starts looping for 'K' times ('K' may be a very large number)

behaviour (Sit and Morris 2002) of a malicious peer in behaving with other peers also can be categorised as a *logical DoS* attack.

These malicious behaviours could be camouflaged as part of a real interaction model in a way that zombie peers cannot be guaranteed to detect them via static analysis of the interaction (even when the specification of this interaction is explicit, as in our examples)

2.2.4 Fake identity

Fake identity is a reinterpretation of a *Sybil* attack (Douceur 2002) in open MAS. In peer to peer networks, a *Sybil* attack is an attack in which an attacker subverts the reputation system by generating a huge number of pseudonymous peers to abuse the resources or to affect the system in a way that he/she wants. *Fake identity* attack is divided into two classes: *deception* and *repudiation* attack.

Different versions of the *deception* attack are as follows:

(a) Fake messages: Having a fake identity makes it easy for an attacker to send deceptive information to others. An example of the sort of system exposed to this vulnerability is the OKOmics system (Sierra et al. 2008), by which proteomic researchers can send queries to different laboratories and compare the results. If an attacker could pretend to be a legitimate proteomic laboratory, then it could deceive researches by wrong answers to their queries about incomplete sequence satisfying the process of identification.
(b) Fake agents: An attacker plays many roles in a genuine interaction. e.g.: in an auction, many fake bidder agents are created by an attacker to demolish the auction.
(c) Fake services: An attacker creates an entirely fake interaction with fake agents for deception or fraudulence. e.g.: a completely fake auction interaction with pseudo auctioneer and bidders to attract real agents and deceive them.
(d) Reputation attack: A group of attackers combine to deceive the reputation mechanism; e.g. many malicious agents may collude to increase their own rank and thus deceive a trust service into believing that they are trusted agents.

Another class of *fake identity* attack is *repudiation*, which may be a result of an attack on the trust service in an open MAS. When benign agents believe in the reliability of a malicious agent by relying on a deceived trust service, they will probably interact with it. Then the malicious agent may send requests (e.g.: buying) to others (e.g.: resellers), but then repudiate its requests at the final stages to reach its goal (e.g.: to bring the resellers into disrepute amongst sellers or to defame the whole system to all benign users). An attacker may also deny its pervious actions by repudiation of its identity.

It is mainly the responsibility of the trust service of the MAS to prevent the *fake identity* attacks, but most trust services are based on a *symmetric reputation* algorithm, which is proved that is not Sybil-proof (Cheng and Friedman 2005), so cannot guarantees against these attacks.

2.3 Risk assessment

Attacks bring about risks that have to be studied and modelled to determine and prioritise effective countermeasures against them. To have a better understanding of potential attack risks in an open MAS we have utilised the DREAD risk assessment methodology (Microsoft 2010). In DREAD the threat (attack) is approached and weighed from the different aspects of Damage, Reproducibility, Exploitability, Affected users and Discoverability. The weights indicating the risk level of each category are then averaged to calculate an overall risk assessment of the threat. The categories of risk in DREAD are:

- *Damage potential* indicates the damage caused if an attack occurs,
- *Reproducibility* is the easiness of attack reproduction,
- *Exploitability* shows the simplicity of the attack exploitation,
- *Affected users* are a rough estimate of the number of affected agents,
- *Discoverability* points out the easiness of discovering the vulnerability exploited in the attack.

It has to be noted that each open MAS has a different set of risks with regards to its implementation and application, so the weighing mechanism has to be customised accordingly to achieve more precise results. To give a sense of this risk assessment technique an example is presented and has utilised the following risk levels: high (3), medium (2), and low (1) (Table 1).

In our risk analysis, each DREAD category is evaluated as low, medium or high with regards to the following explanation. In *Damage potential*, *high* may indicate the risk of an

Table 1 A sample DREAD risk assessment for an open MAS

Attack	DREAD risk					
	Damage potential	Reproducibility	Exploitability	Affected users	Discoverability	Risk (Max = 3)
Read attack						
Interception	3	3	2	2	3	2.6
Access to agents	2	2	2	1	2	1.8
Provenance	1	2	2	1	2	1.6
Probing						
Ontology attack	1	2	2	1	1	1.4
Active probing	2	2	1	1	1	1.4
Altering						
Interaction modification	3	2	2	2	2	2.2
Agent code modification	3	2	2	1	2	2
Agent log modification	1	2	1	1	3	1.6
Injection						
Message injection	1	2	1	1	1	1.2
Knowledge injection	3	2	1	2	1	1.8
Flooding						
Link flooding	2	3	3	3	2	2.6
Agent flooding	2	3	2	2	2	2.2
DoS						
Logical DoS	2	2	2	3	2	2.2
Repudiation	2	2	2	2	2	2
Deception						
Fake agents	1	3	2	1	2	1.8
Fake services	3	3	2	2	2	2.4
Reputation attack	2	3	3	3	3	2.8

3: High risk, *2*: medium risk, *1*: low risk

adversary getting full trust authorisation (e.g. become an administrator), *medium* could mean the leak of sensitive information and low risk might be in revealing unimportant information. In *Reproducibility*, *high* risk is where attacks are reproduced easily, *medium* reproducibility is when the attack is confined to specific conditions and *low* reproducibility means the attack is very hard to replicate. The *Exploitability* risk would be rated *high* if an amateur can take advantage of the vulnerability of the system, *low* if it takes a highly skilled and experienced adversary to carry out the attack. The percentage of affected agents in a MAS would denote the risk level of the *Affected users* category, accordingly. The *Discoverability* of an attack would consider the information available on the vulnerability; a highly discoverable attack has published information about it, while an obscured vulnerability would claim a low risk assessment.

According to the risk analysis in Table 1, reputation attack, interception and flooding are the highest risked threats, while message injection, ontology attack and active probing have the lowest risks.

3 Security of open MAS

Security approaches in the multi-agent security domain can be divided into two parts: the first approach is *prevention*, in which usually encryption-based techniques and authentication methods (e.g.: certificates and PKI[2]) are used. Most research on secure MASs follows this approach. Poslad and Calisti (2000), Finin et al. (2002), Wang et al. (1999), Sun and Chen (2011), Thirunavukkarasu et al. (1995) and Botelho et al. (2009) are some examples of using encryption to prevent MASs from malicious attacks. For instance, Poslad and Calisti (2000), Wang et al. (1999) and Odubiyi and Choudhary (2007) suggest security architectures for the IEEE FIPA agent standard by means of authentication, PKI and VPN.[3] Other prevention methods for secure MASs are: policy driven and secure development methodologies, e.g. Mouratidis et al. (2003b). The first one is based on applying security policies, which may be used for access control, e.g. Quillinan et al. (2008), definition of acceptable behaviour, e.g. Vazquez-Salceda et al. (2003) or policy randomisation to prevent adversaries guess the next agent action, e.g. Tan et al. (2004).

The second approach is *detection*, which tries to detect attacks on MASs and then *respond* to them. Little research has been done in this area and the focus of the work has been on attacks and countermeasures in mobile agents, e.g., Endsuleit and Wagner (2004), Page et al. (2005), Jansen and Karygiannis (2000) and Bierman and Cloete (2002). The main problems in mobile agent systems, which are not in the scope of our review, are threats from agents to hosts and vice versa.

In the following, we describe the contributions of related work to security of open MAS. Some of the systems described, although appropriate to open MAS, were developed with other architectures in mind. Therefore, the shortcomings we identify for them might not be flaws in those other applications.

3.1 Prevention approach

"*Prevention is better than cure*", so the first step against security threats is trying to avoid attacks using prevention methods. Various prevention methods with different attitudes, at different levels of abstraction and for different goals have been proposed and implemented

[2] Public Key Infrastructure.

[3] Virtual Private Network.

Table 2 Examples of encryption and certificate based methods to securing MASs

References	Description	Agent platform
Foner (1996)	PGP-based solution using symmetric and public key encryption	Yenta
He et al. (1998)	Security agents as CA and public key cryptography	KQML
Wang et al. (1999)	A lightweight asymmetric encryption scheme	–
Wong and Sycara (1999)	Unique agent IDs and SSL to provide agent communication security. Agent Certification Authority (ACA) to certify the binding of agents's ids to their public keys and DCA for deployer keys (for authentication)	RETSINA
Poslad and Calisti (2000)	Symmetric and public key encryption and Simple Public Key Infrastructure for authentication	FIPA model
Karnik and Tripathi (2001)	El-Gamal public key for encryption (and a DSA public key for digital signatures) Signed certificates using the agent owner's private key	Ajanta
Novak et al. (2003)	Symmetric and public key encryption and Security Certification Authority (SCA) for authentication	FIPA model
Borselius and Mitchell (2003)	XML encryption service to secure ACL messages using Open PGP	FIPA model
van't Noordende et al. (2004)	SSL encryption for agent communication and an encoded SHA-1 hash of a public RSA key called Self-certifying Identi*er (ScID)*	Mansion
Park et al. (2006)	ID-based cryptography and Local PKI	–
Vila et al. (2007)	TLS/SSL for agent communication security. IMTPoverSSL certificates to provide confidentiality, data integrity and mutual authentication	JADE-S
van't Noordende et al. (2009)	Public key encryption and Self-certifying Identifiers (ScIDs) for end-to-end authentication	Agent operating system (AOS)

in the literature of MAS security. As one possible way of classification, we have categorised these prevention methods as follows: encryption and certificate driven systems, policy-based methods and secure agent development techniques.

3.1.1 Encryption and certificates

Most existing security solutions for MASs suggest the use of encryption to fulfil confidentiality, integrity and non-repudiation in these systems. Symmetric encryption,[4] public key (or asymmetric) encryption,[5] digital signature and certificate management are the popular methods in communication security (Table 2). We discuss specific elements of Table 2 below.

[4] In symmetric cryptography, a secret key, shared between both parties, is used for encryption and decryption. Some examples of common symmetric encryption algorithms are AES, DES, Triple DES, RC6 and Blowfish.

[5] In asymmetric cryptography a pair of public key and private key are used and everything that encrypted by a public key can be decrypted by private key and vice versa. To send a secret message, sender codes it by the

Wang et al. (1999) have suggested *asymmetric encryption* scheme, which is simple and lightweight compared to well-known encryption algorithms such as DES and RSA. The encryption algorithm consists of compressing the message, N-bit grouping, subtracting from the secret key saved in the secret code file and ungrouping. The authors have argued that to guess the secret codes, hackers face a combinational explosion problem and also the compression adds another security layer. The disadvantages of this method, as the authors indicate, are the weakness of the algorithm for short messages and *secret key* management difficulties, which includes producing, transferring and saving large secret files in agents.

Broadly speaking, it can be said that two important problems of using symmetric cryptographic algorithms in open MASs are the need of a separate *secret key* for each pair (or group) of agents and sharing *secret keys*. The first problem may lead to a scaling hurdle for a large number of agents and the second problem particularly affects symmetric cryptosystems in some open MAS, in which unknown agents appear and re-appear frequently. In other words, in open MASs with large (possibly unbounded) number of agents it seems impractical to allocate a separate encryption key for each pair of agents and manage them. The secret key is the security basis of symmetric encryption methods, because if it is discovered, all messages can be decrypted, so it should be protected securely. A common solution for these problems is combining *public key cryptography* schema with the symmetric encryption which many such as Foner (1996), Wong and Sycara (1999), and Borselius and Mitchell (2003) have done.

Wong and Sycara (1999) have proposed a security infrastructure to address the security and trust of the RETSINA framework (Sycara et al. 2003), a reusable multi-agent infrastructure, and provide solutions for secure communication, integrity of system level services (such as naming and matchmaking services) and accountability. They have used unique agent IDs and Secure Socket Layer (SSL)[6] protocol, beneath their agent communication layer, to provide agent communication security. Use of SSL encryption is based on the assumption that agents' deployers have public and private keys binding their physical identities and they should be made responsible for the actions of their agents. The authors also suppose that ANSs[7] and Matchmakers are trusted. In a different research with a similar approach, Vila et al. (2007) have introduced various security services for the JADE framework by integrating existing JADE-S security features and their own mechanism, IMTPoverSSL. IMTPoverSSL provides confidentiality, data integrity and mutual authentication using a certificate-based container-to-container structure. In this framework, each container (group of agents) securely stores other containers' certificates and the above security features are provided using TLS/SSL protocol. In both Methods agents communication security relies on the security of TLS and SSL.

Broadly speaking, relying on widely-used existing security mechanisms has two sides. The disadvantage is that various hackers from different communities (network, web, etc.) try to find its vulnerabilities. In the case of TLS/SSL, there is a reported TLS *Renegotiation* attack (Ray 2009), in which an attacker may be able to tamper messages,[8] and even in some situations might break the encryption, which is a *man-in-the-middle* attack. On the other hand, the positive points are: (1) implementation of security protocols is not the responsibility of MAS

Footnote 5 continued
public key of receiver and the receiver can decode the message by their private key. Diffie-Hellman, RSA are ElGamal some examples of well-regarded asymmetric encryption schema.

[6] SSL and its successor, TLS (Transport Layer Security), run on layers beneath application protocols such as HTTP and SMTP and above the TCP transport protocol.

[7] Agent Name Servers.

[8] A plaintext injection attack against SSL and all current versions of TLS.

developers meaning that implementation security flaws will be avoided; (2) it is a feasible resolution in some open MAS that agents platforms are created by different authorities; (3) these mechanisms are widely used and tested, so they are more reliable; (4) corresponding organisations will eliminate security vulnerabilities e.g. for the TLS *Renegotiation* attack, IETF[9] suggested a solution (Rescorla et al. 2010).

Although using public key cryptography solves the problem of secret key management, authenticity and integrity of the other agents' public keys are still a question. The question is how an agent X can be confident that the claimed public key of agent Y belongs to Y and the key has not been tampered with or replaced by a malevolent agent. Generally, applying a public key cryptosystem without using an authentication mechanism may redound to man-in-the-middle attacks. Using certificate-based schema i.e. Public Key Infrastructure (PKI) and Web of Trust in Open-PGP (a defacto public key encryption system) are common solutions to this. Douceur (2002) proves that trusted certification is the only approach to prevent Sybil attack (or *fake identity attack* in case of MASs) and without a logically centralised authority, there is no solution to Sybil attacks.

The following are some examples of approaches using certificate-based encryption.

- In Foner (1996), in which the main concern is privacy, security issues of Yenta (a decentralised, p2p matchmaking agent) such as gathering other agents' private data and discovering users' profiles from their agents have been addressed. The author has implemented a security system to protect the integrity of MASs and has suggested a PGP-based solution using symmetric and asymmetric encryption for the agents' communications.
- He et al. (1998), have nominated autonomous agents, called security agents, as certificate authorities (CA) in PKI (instead of static hierarchy) to scheme an authentication foundation for MAS security, design scalable authentication systems and make certificate management customisation possible. They have suggested adding new speech acts to KQML to publish various certificates including apply-certificate, issue certificate, renew-certificate, update-certificate and revoke-certificate.
- Wong and Sycara (1999) have proposed a PKI-based solution for RETSINA model of MAS. There are two certification authorities in their security infrastructure: *Agent Certification Authority* (ACA) for agent keys as a part of their security infrastructure and *Deployer Key Certificates* (DCA), a certification authority for deployer keys which lies outside their infrastructure.
- Borselius and Mitchell (2003) have developed an approach for agent secure communication based on using Open PGP to encrypt and sign ACL[10] messages. They have also recommended using the XML encryption service to secure ACL messages.
- Novak et al. (2003) proposed X-Security package to secure agent communication layer of FIPA based agent systems. X-Security supplies a secure model for inter-platform communication and agent activities even in case of inaccessibility of a CA. Security Certification Authority (SCA) is introduced as an independent agent which can renew, suspend and withdraw agents' digital certificates.
- Park et al. (2006) have proposed an algorithm to enhance the security of MASs in distributed computing environments through using *ID-based Cryptography* (ID-C) to solve the scalability problem of PKI in a way that there is no infrastructure necessary to authenticate public keys and manage directories to store certificates. They have suggested an

[9] Internet Engineering Task Force.

[10] Agent Communication Language.

ID-based Threshold decryption method without *key escrow*[11] providing key recovery scheme and key update strategy for a dynamic group membership.

Unfortunately, using public key cryptography infrastructure, like PKI and Open PGP, does not completely solve the underlying problem of public key authenticity in open systems. In other words, correctly identifying the public key of the user (agent owner) and ensuring that this key belongs to the real one still is a problem. In an open system, in which there is no previously established contact to legally commit the agent owners, repudiation of the signature and keys are possible. A malicious user may also pretend to be a famous organisation so as to gain confidence of users in accepting its certificate. Furthermore, some general security challenges of using PKI are mentioned in Ellison and Schneier (2000). Additionally, in some applications of open MASs (for example in emergency response or crisis management), applying certificates, which requires pre-negotiations or physical contacts between agents' owners and the certificate authorities, may not be very helpful. Furthermore, almost all the proposed solutions using encryption and certificate-based methods have focused on securing agent communications in low level transactions, securing agent messages only at platform level, but the issue of security in MASs extends to higher levels of system architecture.

Generally cryptography and certificate based solutions are suggested to prevent *read* and *altering attacks*, although openness in MASs might cause some difficulties in practise. Encryption of sensitive data and authentication are the first and most effective steps toward countering *interception*, *unauthorised access to agents*, *provenance attack* and *agent log modification*. To avoid *interaction modification* attack, besides encryption, SSL, digital signature (which is not content-aware), integrity checking (e.g. based on SHA-1 and MD5) is also necessary. There is still more needed to impede *interaction modification* attack by avoiding a single point of responsibility; not relying on just one (even trusted) agent information, which is the responsibility of the trust service in a MAS. To prevent agent modification attack, employing *encrypted function* and other code security mechanisms are recommended. As an additional level of protection, proof carrying code is a promising technique to detect any modification in agent's code.

3.1.2 Policy-based methods

Policy-driven mechanisms have been applied widely in variety of security applications. Security policies, as a prevention approach to secure MAS, can be used for access control, definition of acceptable behaviour or confidentiality in adversary environments. Some examples of these methods are shown in Table 3.

Access control using security policy is quite common in computer systems and MASs are no exception. Wagner (1997) has shown how the database concept of multi-level security can be applied to inter-agent communication in order to protect confidential information. A knowledge system of *MSL*[12] databases and some basic inter-agent communication rules have been defined. The communication rules implement security classifications and the *MSL* database assigns a security classification (e.g.: unclassified, confidential, secret and top secret) to all information items are and allocates an authorisation to all users. The whole system then enables agents to comply with a defined security policy.

In Quillinan et al. (2008), it is argued that well defined and easily configurable security policies address security of MAS. Several agent middleware systems provide access control

[11] In the *key escrow* schema, private keys are stored in an escrow for key recovery under certain circumstances by an authorized third party.

[12] *Multi-level secure*.

Table 3 Examples of policy-based methods

References	Type	Description
Wagner (1997)	Access control	Use of multi-level security concept
Quillinan et al. (2008)	Access control	Role based access control
Tekbacak et al. (2009)	Access control	Decentralised ontology and XACML
Tan et al. (2004)	Definition of acceptable behaviour	Profile-based reasoning model (for dynamic security reconfiguration)
Vazquez-Salceda et al. (2003)	Definition of acceptable behaviour	Electronic institutions as police norms
Paruchuri et al. (2006)	Policy randomisation	Linear and non-linear programming algorithms to randomise single-agent policies to avoid an agent's action being easily predictable
Paruchuri et al. (2009)	Policy randomisation	A non-linear program with non-convex constraints ensuring the communication constraints are met
Tekbacak et al. (2011)	Role-based policy	A policy model for agents when ontologies in the environment can change

architecture such as *JADE-S, SeMoA* and *AgentScape*. *JADE-S* supplies a style of decentralised access control that is not so flexible todefine default security policies. *SeMoA* has a centralised access control and does not allow users to define their own policies. *AgentScape* provides a hybrid access control, a combination of centralised and decentralised policy enforcement. The architecture used in *AgentScape* middleware, uses Role Based Access Control[13] (RBAC) and while having a set of default security policies, allowing users to customise them (Quillinan et al. 2008). Vitabile et al. (2008) extended JADE-S framework with a strong user authentication, a reputation-based trust system and an access control mechanism based on policy files. In the proposed framework FPGA biometric sensors provide secure and fast authentication of the agent owner in a MAS.

Although access control is necessary in many systems, defining a suitable security policy, which does not cramp users and is not very open, is non-trivial, especially when there are heterogeneous agent systems with different ontologies. Malicious agents may also exploit vulnerabilities in access governance systems by masquerading. A common issue in many access control mechanisms for MASs is that they are usually applied at the level of agent middleware and concern access to low-level objects (e.g. files and IP address). This could be considered as a weakness, because they might not be able to detect high-level access violations (such as probing attack and fake identity attacks).

In (Tekbacak, Tuglular, and Dikenelli, An Architecture for Verification of Access Control Policies with Multi Agent System Ontologies, 2009), a decentralised ontology and XACML[14]

[13] *Role-based Access Control* or *role-based security* (Ferraiolo & Kuhn) is a predominant access control mechanism, in which all access is through roles (collections of permissions).

[14] XACML is a declarative access control policy language implemented in XML.

based access control architecture for MASs has been proposed. The agent domain ontology and access control parameters in agent security ontology have been combined within a common XACML policy document that is used through different MAS applications through translation of XACML and OWL to description logic (DL) concepts. This formalization under DL concept allows defining and effectively implementing an array of policy analysis services and helps the verification of policies under a common point. One limitation of this approach is that it is not completely compatible with open MASs without further upgrades in their implementation and it is mainly because of the centralised architecture of this model.

The second type of policy driven methods is more general than just controlling access to information and defines acceptable behaviour of the system. An example is Tan et al. (2004), in which Tan et al. have proposed a policy-based infrastructure for dynamic security reconfiguration in open and heterogeneous systems to address end-to-end security interoperability problem. They have developed a profile-based reasoning model for a dynamic security reconfiguration system, which detects and analyses policy conflicts and the need for security reconfiguration and then resolve them at a meta-level without changing underlying systems' implementation. A similar approach may be used to detect attacks at run-time and automatically reconfigure the system for a suitable response.

Definition of acceptable behaviour for an agent by means of electronic institution is another way of attack prevention in MAS. An electronic institution provides a set of rules that define what agents are permitted and forbidden to do and the consequences of their actions (Esteva et al. 2004). An electronic institution, as a collection of norms implemented by security policies, can be devised as a framework to define police norms that guide, control and regulate behaviour of other agents in an open MAS (Vazquez-Salceda et al. 2003).

The third approach in policy-based methods utilises policy randomisation to prevent adversaries guessing the agents' next actions (Paruchuri et al. 2006, 2009). The worst-case assumptions here are; first, agents act in adversarial environment which cannot be modelled (because there may be unseen adversaries whose actions and capabilities are unknown); second, the adversary can observe the agent's state; third, the adversary knows the agent policy; forth, there is no or limited communication among agents. The authors have introduced a randomised policy making to avoid an agent's action being easily predictable, using a decision-theoretic model based on the Multi-agent Constrained Markov Decision Problem (MCMDP).

Paruchuri et al. (2006) have provided three linear and non-linear programming algorithms, to randomise single-agent policies. They also have implemented a new algorithm, Rolling Down Randomisation (RDR), which efficiently generates randomised policies via the single-agent linear programming method. With similar assumptions in their prior work, the authors in Paruchuri et al. (2009), have developed a non-linear program with non-convex constraints that randomises agent team policies while ensuring that the communication constraints of the team are met and the miscoordination arising due to randomised policies is countered.

Loulou et al. (2007), have proposed a formal approach to prevent attacks on MASs by verifying security policies. They proved theorems describing in which circumstances security policies are successful and could overcome a given kind of attack on mobile agents. They used Z-Eves tools for type checking and theorem proving. In a different study on credential-based authorisation policies, Becker (2010) has introduced a formal framework to analyse secrecy of policy languages. Becker redefined two information flow properties (non-interference and detectability) in credential systems and proposed an inference system that informs us what an adversary can detect from our system. Although his work is not directly related to MASs, but his formal approach can be used to protect MASs against probing attacks.

Security policy approach, especially using low-level and high-level access control methods, can help the prevention of read attacks (*interception, unauthorised access to agents,*

Table 4 Examples of adding security to agent-oriented software engineering

References	Phase	Description
Liu et al. (2002)	Requirements engineering	Modelling of relationships among strategic actors in order to extract, identify andanalyse security requirements
Yu and Cysneiros (2002)	Requirements engineering	A framework to model the way agents interact to achieve privacy
Mouratidis et al. (2003b)	Throughout the development process	Adds security concepts into the design methodology in *Tropos*
Bresciani et al. (2004a)	Requirements engineering	An algorithm to reduce the complexity or criticality of security requirement analysis
Mouratidis and Giorgini (2009)	Throughout the development process	A methodology for integrating functional and security requirements in *Tropos*
Xiao (2009)	Throughout the development process	A model-driven architecture by merging the concept of agent role in AOSE and Role Based Access Control
Massacci et al. (2010)	Requirements engineering	A try to bridge the gap between policy specification and requirements analysis by deriving privacy policies from requirements
Rojas and Mahdy (2011)	Throughout the development process	An extension to *Tropos* methodology by integrating threat modelling in software application development

provenance attack). Defining acceptable behaviour (e.g. limit the number of queries from one agent) can assist counter *denial of service* and *ontology attacks*.

3.1.3 Secure agent development

Considering security concerns during the development and implementation of MASs is another method to prevent security threats. To fulfil this goal, we consider two approaches: agent-oriented software engineering and language-based security for agents.

3.1.3.1 Agent-oriented software engineering Our objective is to bridge the gap between requirements analysis and policy specification by deriving privacy policies from requirements.

Adding security to agent-oriented software engineering (AOSE) has become an important area within the agent research community (Mouratidis et al. 2003a). Liu et al. (2002) and Yu and Cysneiros (2002) are examples of addressing security issues within the requirements engineering process and Mouratidis and Giorgini (2009), Xiao (2009) and Rojas and Mahdy (2011) are examples of integrating security throughout the agent development process (Table 4). We can also benefit from UML extensions as generic tools for secure software

development e.g. UMLsec (Jurjens 2002) that incorporate security requirement analysis to software design.

Many efforts have been made to provide appropriate methods for integrating functional requirements and non-functional security requirements during the whole software development stages using *Tropos*, an agent-oriented software development methodology (Bresciani et al. 2004b); e.g. Mouratidis et al. (2003a), Mouratidis (2007), Beydoun et al. (2009), Massacci et al. (2010). Mouratidis et al. (2003a) has extended *Tropos* by adding security concepts into the design methodology, with concepts such as Security Diagrams, Security Constraints, Secure Entities and Secure Capabilities, that enable MAS developers to describe security requirements. They illustrate their new methodology using a case study from health care sector. In Bresciani et al. (2004a) the degree of criticality and complexity of the parts of the agent system has been analysed to identify possible security bottlenecks of the system. This analysis facilitates decision making of agent system developers, about probable trade-offs between functional requirements and security. The authors also have suggested an algorithm to reduce the complexity or criticality of security requirement analysis process. Other efforts to suggest a methodology for integrating functional and security requirements in *Tropos* are Mouratidis et al. (2003b), Mouratidis and Giorgini (2009) and Rojas and Mahdy (2011). Massacci et al. (2010) have proposed the use of *Secure Tropos* methodology with SI*, a modelling language to deal with security and trust. Their goal was to derive privacy policies from requirements to bridge the gap between policy specification and requirements analysis.

Xiao (2009) has merged the concept of agent role in AOSE and in Role Based Access Control to produce a model-driven architecture for building adaptive and secure MAS. The idea is that interaction models, containing the agent's role, obligations and security policy rules, which define constraints derived from agent social roles, identify rights of agent. Therefore, functional requirements and security constraints are connected by the notion of role. This method aims to deal with a complete MAS development process from requirements analysis and the early design phase to the implementation phase. The proposed methodology is compatible with open multi-agent systems.

Attack modelling, during the design phase, allows software engineers to clearly understand and analyse design flaws and mitigate security vulnerabilities. As a result, various security attack modelling techniques and tools have been introduced in computer security literature. We could name attack trees (Schneier 1999), attack graphs (Lippmann and Ingols 2005), statecharts (El Ariss and Xu 2011) and petri nets-based attack nets (McDermott 2000) as a few of these techniques. These techniques can also be customised and applied to agent-oriented software engineering to prevent known attacks on open MASs. For example in Rojas and Mahdy (2011), the Tropos methodology is extended by integrating threat modelling in software application development.

3.1.3.2 Language-based security Access control is insufficient for the secrecy of computational systems because it does not prevent the propagation of confidential information and is not suitable for encrypted information. Encryption and digital signature also cannot prevent attacks that exploit information flow (e.g. *injection* attack). Information flow analysis is a complement to those approaches to ensure some security properties hold.

Using information flow theory for software security analysis is an old idea e.g. by tagging confidential data and analysing tagged data propagation (Denning 1976). Sabelfeld and Myers (2003) argue, in their survey of language-based information flow security, that sound type systems could be a promising language-based technique to specify and enforce an information flow policy.

Information flow techniques can also be applied to MAS. Halpern and ONeill (2008) provided a general framework for analysing the secrecy of MASs that can handle probability and non-determinism in synchronous or asynchronous systems. In their approach, a logic that includes modal operators for reasoning about knowledge and probability is used to syntactically characterise secrecy. They worked on I/O systems but their solution is applicable to agent protocol analysis and semantics for agent programming languages.

There are two approaches to analysing information flow security: static and dynamic (runtime) analysis. Static techniques prove program correctness with reasonable computation cost, conservatively detect implicit and explicit information flows and provide stronger security assurance than the dynamic techniques (Sabelfeld and Myers 2003).

To avoid a*ctive probing* and *injection* attacks in an open MAS, we advocate the implementation of static language-based security analysis that uses security types to prevent information leakage. For this purpose the agent could be implemented either by languages such as JIF or by enhanced versions of current agent development languages.

3.2 Detection approach

Although prevention methods usually take precedence over detection methods, it is not always possible or feasible to impede all kind of attacks. In practice, there is a continual battle between defenders and attackers struggling to break the security defences. Hence, the design and implementation of powerful detection and response mechanisms against possible attacks as a second stage of defence are essential. In the detection approach, the sooner an attack is detected, the less the impact of the attack on the victim agents would be. In case of open MAS, in which there is a minimum guarantee about the agent's identity and behaviour, detection seems to be at least as important of prevention.

3.2.1 Monitoring

Monitoring is a general method of attack detection in hostile environment. The goal of monitoring-based techniques in an open MAS is to detect the misbehaviour of agents or to find anomalies in the system. For misbehaviour detection, we need to define the specifications of agent communication protocol in order to detect interactions that exploit it. As agent communications do not follow a common consented standard, misbehaviour should be defined separately for each type of MAS.

In the anomaly detection approach, patterns in data that do not conform to the expected behaviour are detected (Chandola et al. 2009). Classification, clustering or statistical methods are a few examples of anomaly detection techniques that can be applied in open MASs for attack detection. Anomalies are detected when the current MAS state differs from the trained model (classifier). The openness in open MASs might be a hinder to successful anomaly detection. Openness in the sense that new agents can freely join the system is not a problem, but when the open MAS allows new component to be added to the system at run-time, e.g. the OpenKnowledge system (Robertson et al. 2008), this leads to a dynamic behaviour that resists anomaly detection. In these more dynamic systems, anomaly detection techniques generate many false positives rendering the techniques ineffective.

We will now discuss *peer monitoring*, *information monitoring* and *policy monitoring* as three misbehaviour detection monitoring techniques and *activity monitoring* as an anomaly detection technique.

3.2.1.1 Peer monitoring Zaslavsky and Indrawan (2004) have introduced the *Buddy* model and Page et al. (2005) have extended it to a more general security mechanism for mobile agent systems. This model adds a separate security layer to the agents' business functionality. In the proposed centralised system, agents who monitor others are called *Buddy* and the others who are monitored are called *protected agents* (PA). A PA might also be Buddy of another PA. There are three basic rules ensuring the uniformity of the model: first, each PA has two *Buddy* agents; second, PA and*Buddy* agents must never be the same; and three, an agent can be a *Buddy* of only one *protected* agent. The home base host manage security of the whole MAS and in case of any malicious activity; it receives alerts from the monitoring system.

While the *Buddy* model has been designed for mobile agents, with minor changes it can be applied to non-mobile agent systems as well. The centralised nature of this model may be a limitation for scalability of open multi-agent systems, although use of local monitoring subsystems may overcome this shortcoming.

3.2.1.2 Information monitoring Monitoring published information in agent communications and keeping track of transferred messages facilitate the detection of some information-centred attacks: *provenance, active probing and ontology attacks*. For example, in the case of the *ontology attack*, monitoring users' queries to the ontology and defining some thresholds for the number and the scope of the queries may impede the attack. The information monitoring technique helps us to at least to find what other agents (adversaries) have already found out about the agents and be more cautious in future communication with the suspected agents.

3.2.1.3 Policy monitoring Implementing agent on top of web services has been introduced as a potential market and a way of more popularity for agent systems (Petrie and Bussler 2003; Xiao 2009), so compatible security solutions from Web Service community may be helpful in open MAS. Clark et al. (2010) have introduced a framework for secure monitoring of a specific Service Level Agreement (SLA) and have implemented it in *AgentScape* system. The authors have modified WS-Agreement, an SLA specification for establishing agreement between parties in Web Services, for effective monitoring. They have designed a model for secure and reliable violation monitoring of SLAs and a method for specifying violation policies in a hostile environment. A centralised and a decentralised monitoring of contraventions were tested and the results showed slightly better performance in the decentralised version. As an intrusion can be considered a kind of violation, this system might also be applied to detect attacks, at least for those attacks for which we could specify their features.

3.2.1.4 Activity monitoring *Activity monitoring* is similar to the concept of *activity profiling* (Carl et al. 2006) to detect the *denial of service* attacks in computer network literature. It is based on the calculation of the average traffic rate for the whole interaction between two agents. Each agent can measure the average traffic rate and whenever any counterpart agent behaves very differently, it can react by raising an alarm, decreasing the rank of that agent or filtering possible malicious agent. In a MAS with centralised management approach, the total MAS communication activities and the average rate of all inbound and outbound flows can be measured and be used to detect suspicions communications. To avoid the high-dimensionality problem, a MAS can be clustered into different classes, each of which has a monitoring agent.

To detect the DoS and DDoS attacks, especially flooding attacks, *change-point detection* methods and *wavelet analysis* of agent traffics are recommended (Carl et al. 2006). We can also employ existing anomaly detection techniques; i.e. Classification Based, Clustering

Based, Nearest Neighbour Based, Statistical, Information Theoretic and Spectral (Chandola et al. 2009).

3.2.2 Attack modelling

Modelling attacks or attackers is a useful technique to detect some attacks on MASs. In security literature you can find several formal/informal attack(er) modelling strategies for different purposes. Security modelling, in general, is an approach to analyse various aspects of security (e.g. confidentiality and integrity) in a system. While it is more likely to use security modelling as a prevention method (such as a number of attack modelling techniques in Sect. 3.1.3.1), online modelling could also be recognised as a detection approach.

In the following sub-sections, two attack modelling approaches to attack detection in open MASs are introduced: coordination graph and statistical modelling.

3.2.2.1 Coordination graph Given that some attacks on agent systems are taking advantage of social behaviour between attackers, Braynov and Jadliwala (2004) have studied detection of coordinated attacks on MAS. A group of attacker agents may collude to stage a large attack on the victim agent system by dividing the tasks into separate subtasks among themselves. Conventional intrusion detection systems just detect the last section of the attack chain, not the attacker assistants who prepare attack prerequisites. Assuming that there is a security mechanism to detect a single malicious action, the authors have defined formal metrics on the coordination graph to discover main and peripheral attacker agents. They have introduced a formal model of distributed monitoring and a formal method and an algorithm to detect maximal malicious group of attackers using a coordination graph (nodes are states and arcs are attacks) of all users.

The proposed methods can be applied to detect attacks at an early stage and is capable of being used online (for attack detection) or offline (for forensic analysis). The suggested method is a useful defence against many coordinated attacks, but the assumption that every single malicious action can be detected may not be valid in some attacks, such as an *ontology attack*, so it may not be appropriate for them.

Dove (2009) has proposed a platform for unmanned autonomous system testing (UAST) based on a socially aware team of autonomous agents. His platform can be merged with the attack detection method of Braynov and Jadliwala (2004) to employ socially aware security agents as a community watch in an open MAS. Both methods can be used to detect coordinated attacks such as *distributed denial of service*.

Khan et al. (2009) have taken a different approach and have suggested a technique that models malicious hosts' behaviour against mobile agents. They have also proposed Mobile Agent Graph Head Sealing (MAGHS) method to grantee the integrity of mobile agent system. MAGHS uses symmetric encryption algorithms with a shared secret key between every host and the home platform. It is assumed that a dynamic data structure like graph can represent the resultant of mobile agents' execution on a host. The attacker behaviour is modelled as a time function, in which long time means unsuccessful attack, and In the case of any possible modification in the agent state, it can be detected by the agent owner. MAGHS is a solution to truncation and repudiation attacks in mobile agent systems.

3.2.2.2 Statistical modelling Statistical modelling is another approach, in which anomaly of an open MAS can be detected. In statistical anomaly detection techniques, it is assumed that anomalies occur in the low probability regions of the stochastic model (Chandola et al. 2009). There are two types of statistical anomaly modelling: (1) parametric techniques such

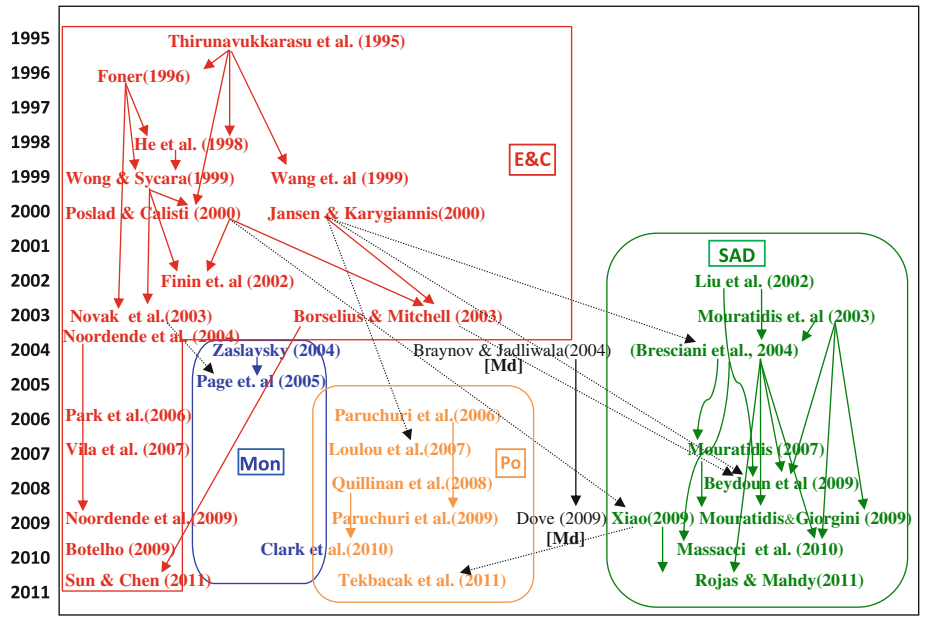

Fig. 7 Chronological order of proposed security approaches to secure MASs. 'E&C' means encryption and certificate-based methods, 'Po' is policy-based methods, 'SAD' stands for secure agent development, 'Mon' is for monitoring approach and 'Md' means modelling technique

as the Gaussian model (Aggarwal and Yu 2008) or regression model methods (Kadota et al. 2003) and (2) non-parametric techniques such as histogram-based methods e.g. (Dasgupta and Majumdar 2002).

In both modelling techniques, a statistical model of the training data is generated and a statistical test is applied to the new data set. Anomalies are the resultant instances with low probability. Parametric techniques assume an underlying distribution in the statistical population, while non-parametric techniques do not assume any distribution and the number and nature of the parameters are not fixed in advance. These anomaly detection methods can be used to detect some *probing attacks*, *ontology attacks* and *denial of service attacks* on open MAS. A disadvantage of statistical techniques is the assumption that the data is generated from a particular distribution and it often does not hold for high dimensional real data sets (Chandola et al. 2009).

4 Summary and conclusion

In this paper, we have introduced and categorised the main forms of attack on open MASs and various approaches to secure them. A three-layer attack taxonomy has been proposed to help understand the implications of attacks and the defence mechanism against them. In the first layer of our taxonomy we have classified attacks based on violations against confidentiality, integrity, availability and accountability. In the second and third layer we have categorised attacks based on the novelty of the attack technique and the attack target. In total, we have introduced 16 attack classes and described a number of them in LCC code. In addition we have performed a risk assessment using the DREAD technique on the attacks, in order to facilitate prioritising response or preventive measures against them.

Table 5 Summary of security mechanisms to countermeasure different attacks on open MASs

Attacks	Countermeasures									
	Prevention					Detection				
	Encryption and certificates	Policy-based (access control)	Policy-based (behaviour defin.)	Agent-oriented software eng.	Language-based security	Peer monitoring	Information monitoring	Policy monitoring	Activity monitoring	Attack modelling
Read attack										
Interception	●	●	○	●	●	○	○	◉	○	○
Access to agents	●	●	○	●	◉	○	○	◉	○	○
Provenance	●	●	○	●	◉	○	●	◉	◉	○
Probing										
Ontology attack	◉	◉	◉	◉	◉	○	●	◉	◉	◉
Active probing	◉	◉	◉	○	●	○	◉	◉	◉	◉
Altering										
Interaction modification	●	●	○	●	●	●	○	◉	○	○
Agent code modification	●	●	○	●	●	●	○	◉	○	○
Agent log modification	●	●	○	◉	○	●	○	◉	○	○
Injection										
Message injection	◉	◉	○	○	●	○	◉	◉	◉	◉
Knowledge injection	◉	◉	○	○	●	○	◉	◉	◉	◉
Flooding										
Link flooding	○	○	◉	●	○	◉	○	○	●	●
Agent flooding	○	○	◉	●	○	◉	○	○	●	●

Table 5 continued

Attacks	Countermeasures									
	Prevention					Detection				
	Encryption and certificates	Policy-based (access control)	Policy-based (behaviour defin.)	Agent-oriented software eng.	Language-based security	Peer monitoring	Information monitoring	Policy monitoring	Activity monitoring	Attack modelling
DoS										
Logical DoS	○	○	◉	●	○	◉	○	○	●	●
Repudiation	●	○	○	◉	○	○	○	◉	○	●
Deception										
Fake agents	●	◉	◉	◉	○	○	○	○	●	◉
Fake services	●	◉	◉	◉	○	◉	○	○	●	◉
Reputation attack	●	○	◉	◉	○	○	○	○	●	●

We have reviewed and categorised security solutions intended to provide security in MASs and have suggested some solutions from network security literature as countermeasures to attacks against open MASs. The presented security solutions can be divided into prevention and detection approaches; *Encryption and certification, policy-based methods* and *secure agent development* are three prevention mechanisms, and *monitoring* and *modelling* are two detection approaches. Some of the presented security techniques did not aim at securing open MASs or even MASs, but they were found to be applicable to open MASs.

To illustrate links between various techniques a visual summary of security approaches in the literature proposed to secure MASs is provided as a chronological tree in Fig. 7. This might not be a comprehensive phylogenetic tree but it shows key relations amongst different approaches from 1995 to 2011. This does not include approaches to security issues that are specific to the mobility of mobile agents; i.e. threats from mobile agent to platforms and from platforms to agents.

Proposed appropriate security techniques for each attack category are summarised in Table 5. In this table • indicates an effective and feasible security technique to counter an attack, which is strongly recommended, although it may not completely solve the problem. ○ means that the attack cannot be prevented or detected with the specified security countermeasure and ◉shows that the security technique does not solve the problem, but helps us make the attack harder or discourage attackers. The feasibility and effectiveness of suggested solutions are imprecise because of variations in security sub-techniques and their dependency on the application and implementation of the open MAS. For example, variety access-control mechanisms have different capabilities in the prevention of ontology attacks, however in Table 5, it is assumed that the most effective mechanism will be used to countermeasure each attack.

Open MASs are growing in popularity in the Multi-agent Systems community, while there still remain many potential gaps in their security. The limitations in protecting open MASs like minimal information on the identity and behaviour of agents make them particularly difficult to protect, but their openness attracts new applications, making new problems emerge. These vulnerabilities have many affects rendering security issues indispensible and further research necessary. This review on the possible attacks and security approaches in open MASs could pave the path for future studies.

References

Aggarwal CC, Yu PS (2008) Outlier detection with uncertain data. In: SIAM international conference on data mining (SDM), pp 483–493

Artikis A, Sergot M, Pitt J (2009) Specifying norm-governed computational societies. ACM Trans Comput Logic 10:1–42

Becker MY (2010) Information flow in credential systems. IEEE Comput Secur Found Symp 0:171–185

Beydoun G, Low G, Mouratidis H, Henderson-Sellers B (2009) A security-aware metamodel for multi-agent systems (MAS). Inf Softw Technol 51(5):832–845

Bierman E, Cloete E (2002) Classification of malicious host threats in mobile agent computing. In: SAICSIT'02: Proceedings of the 2002 annual research conference of the South African institute of computer scientists and information technologists on enablement through technology. South African Institute for Computer Scientists and Information Technologists, South Africa, pp 141–148

Bijani S, Robertson D, Aspinall D (2011) Probing attacks on multi-agent systems using electronic institutions. In: Declarative Agent Languages and Technologies Workshop (DALT), AAMAS 2011

Borselius N, Mitchell C (2003) Securing FIPA agent communication. In: Proceedings of the 2003 International conference on security and management (SAM'03), vol 1, USA, pp 135–141

Botelho V, Enembreck F, Avila B, de Azevedo H, Scalabrin E (2009) Encrypted certified trust in multi-agent system. In: The 13th international conference on computer supported cooperative work in design, pp 227–232

Braynov S, Jadliwala M (2004) Detecting malicious groups of agents. In: Proceedings of the 1st IEEE symposium on multi-agent security and survivability (MAS&S) 2004. IEEE Computer Society, Philadelphia, pp 90–99

Bresciani P, Giorgini P, Manson G, Mouratidis H (2004a) Multi-agent systems and security requirements analysis. In: Lecture Notes in Computer Science. Springer, Berlin

Bresciani P, Perini A, Giorgini P, Giunchiglia F, Mylopoulos J (2004b) TROPOS: an agent-oriented software development methodology. Auton Agents Multi Agent Syst 8:203–236

Carl G, Kesidis G, Brooks RR, Rai S (2006) Denial-of-service attack- detection techniques. IEEE Internet Comput 10(1):82–89

Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv 41:15:1–15:58

Cheng A, Friedman E (2005) Sybilproof reputation mechanisms. In: P2PECON'05: Proceedings of the 2005 ACM SIGCOMM workshop on economics of peer-to-peer systems. ACM, Philadelphia, pp 128–132

Clark KP, Warnier M, Quillinan TB, Brazier FM (2010) Secure monitoring of service level agreements. In: Proceedings of the 2nd international workshop on organizational security aspects (OSA 2010). IEEE

Dasgupta D, Majumdar N (2002) Anomaly detection in multidimensional data using negative selection algorithm. In: The IEEE conference on evolutionary computation. Hawaii, pp 1039–1044

Demazeau Y, Rocha Costa A. (1996) Populations and organizations in open multi-agent systems. In: Proceedings of the I national symposium on parallel and distributed AI (PDAI'96), Hyderabad

Denning DE (1976) A lattice model of secure information flow. Commun. ACM 19-5:236–243

Douceur JR (2002) The sybil attack. In: IPTPS '01: Revised papers from the 1st international workshop on peer-to-peer systems. Springer, pp 251–260

Dove R (2009) On detecting and classifying aberrant behavior in unmanned autonomous systems under test and on mission. In: Live virtual constructive conference. International Test and Evaluation Association

El Ariss O, Xu D (2011) Modeling security attacks with statecharts. In: The joint ACM SIGSOFT conference—QoSA and ACM SIGSOFT symposium. ACM, pp 123–132

Ellison C, Schneier B (2000) Ten risks of PKI: what you're not being told about public key infrastructure. Comput Secur J 16(2):1–7

Endsuleit R, Wagner A (2004) Possible attacks on and countermeasures for secure multi-agent computation. In: Proceedings of the international conference on security and management (SAM'04), Las Vegas, pp 221–227

Esteva M, de la Cruz D, Rosell B, Arcos JL, Rodriguez-Aguilar JA, Cuni G (2004) Engineering open multi-agent systems as electronic institutions. In: 19th national conference on artifical Intelligence (AAAI 04). AAAI Press, pp 1010–1011

Finin T, Joshi A, Joshi A (2002) Developing secure agent systems using delegation based trust management. In: Security of mobile multiAgent systems (SEMAS 02) held at autonomous agents and multiAgent systems (AAMAS), pp 200–202

Foner LN (1996) A security architecture for multi-agent matchmaking. In: Proceedings of the 2nd international conference on multi-agent systems, pp 80–86

Halpern JY, ONeill KR (2008) Secrecy in multiagent systems. ACM Trans Inf Syst Secur 12:5:1–5:47

He Q, Sycara KP, Finin TW (1998) Personal security agent: KQML-based PKI. In: The 2nd international conference on autonomous agents

Igure V, Williams R (2008) Taxonomies of attacks and vulnerabilities in computer systems. Commun Surv Tutor 10(1):6–19

Jansen W, Karygiannis T (2000) Mobile agent security. National Institute of Standards and Technology (NIST) Special Publication 800-19

Jurjens J (2002) Using UMLsec and goal trees for secure systems development. In: The 2002 ACM symposium on applied computing. ACM, Madrid, pp 1026–1030

Kadota K, Tominaga D, Akiyama Y, Takahashi K (2003) Detecting outlying samples in microarray data: a critical assessment of the effect of outliers on sample classification. Chem-Bio Inform 3:30–45

Karnik NM, Tripathi AR (2001) Security in the Ajanta mobile agent system. Softw Pract Experience 31(4):301–329

Khan A, Arshad Q, Niu X, Yong Z, Anwar MW (2009) On the security properties and attacks against mobile agent graph head sealing (MAGHS). In: The 3rd international conference and workshops on advances in information security and assurance (ISA 09). Springer, Seoul, pp 223–228

Lee H, Alves-Foss J, Harrison S (2004) The use of encrypted functions for mobile agent security. In: The 37th annual Hawaii international conference on system sciences (HICSS'04). IEEE Computer Society, p 10

Lippmann RP, Ingols KW (2005) An annotated review of past papers on attack graphs. Linoln Lab, MIT, Cambridge
Liu L, Yu E, Mylopoulos J (2002) Analyzing security requirements as relationships among strategic actors. In: 2nd Symposium on requirements engineering for information security (SREIS 2002)
Loulou M, Tounsi M, Kacem AH, Jmaiel M, Mosbah M (2007) A formal approach to prevent attacks on mobile agent systems. In: SECUREWARE'07: Proceedings of the the international conference on emerging security information, systems, and technologies. IEEE Computer Society, Washington, pp 42–47
Majumdar A, Thomborson C (2005) On the use of opaque predicates in mobile agent code obfuscation. In: Intelligence and security informatics. Springer, Berlin, pp 255–236
Massacci F, Mylopoulos J, Zannone N (2010) Security requirements engineering: the SI* modeling language and the secure tropos methodology. Adv Intell Inf Syst 265:147–174
McDermott JP (2000) Attack net penetration testing. In: The 2000 workshop on new security paradigms (NSPW'00), Cork, pp 15–21
Microsoft (2010) Threat risk modeling. Retrieved from The Open Web Application Security Project: http://www.owasp.org/index.php/Threat_Risk_Modeling
Mitchell C (2003) Security for Mobility. Institution of Electrical Engineers, Piscataway
Mouratidis H (2007) Secure tropos: a security-oriented extension of the tropos methodology. Int J Softw Eng Knowl Eng (IJSEKE) 17(2):285–309
Mouratidis H, Giorgini P (2009) Enhancing secure tropos to effectively deal with security requirements in the development of multiagent systems In: Safety and security in multiagent systems. Springer-Verlag, pp 8–26
Mouratidis H, Giorgini P, Manson G (2003a) Modelling secure multiagent systems. In: AAMAS 03: Proceedings of the 2nd international joint conference on autonomous agents and multiagent systems. ACM, New York, pp 859–866
Mouratidis H, Giorgini P, Weiss M (2003b) Integrating patterns and agent-oriented methodologies to provide better solutions for the development of secure agent systems. In: Workshop on expressiveness of pattern languages 2003, at ChiliPLoP
Necula G, Lee P (1998) Safe, untrusted agents using proof-carrying code. In: Vigna G (ed) Mobile agents and security. Springer, Berlin pp 61–91
Novak P, Rollo M, Hodik J, Vlcek T (2003) Communication security in multi-agent systems. In: The 3rd central and eastern European conference on multi-agent systems (CEEMAS'03). Springer, pp 454–463
Odubiyi JB, Choudhary AR (2007) Building security into an IEEE FIPA compliant multiagent system. In: Proceedings of the 2007 IEEE workshop on information assurance, IAW. IEEE Computer Society, West Point, pp 49–55
Oey MA, Warnier M, Brazier FM (2010) Security in large-scale open distributed multi-agent systems. In: Kordic V (ed) Autonomous agents. IN-TECH, pp 107–130
Page JP, Zaslavsky AB, Indrawan MT (2005) Extending the buddy model to secure variable sized multi agent communities. In: Proceedings of the 2nd international workshop on safety and security in multiagent systems, Utrecht, pp 59–75
Park H, Ju H, Chun K, Lee J, Ahn S, Noh B (2006) The algorithm to enhance the security of multi-agent in distributed computing environment. In: ICPADS'06: Proceedings of the 12th international conference on parallel and distributed systems. IEEE Computer Society, Washington, pp 55–60
Paruchuri P, Tambe M, Ordonez F, Kraus S (2006) Security in multiagent systems by policy randomization. In: Proceedings of the 5th international joint conference on autonomous agents and multiagent systems (AAMAS 06). ACM, Hakodate, pp 273–280
Paruchuri P, Pearce JP, Marecki J, Tambe M, Ordonez F, Kraus S (2009) Coordinating randomized policies for increasing security of agent systems. Inf Technol Manag 10:67–79
Petrie C, Bussler C (2003) Service agents and virtual enterprises: a survey. IEEE Internet Comput 7:68–78
Poslad S, Calisti M (2000) Towards improved trust and security in FIPA agent platforms. In: Workshop on deception, fraud and trust in agent Societies, Spain
Poslad S, Charlton P, Calisti M (2002) Specifying standard security mechanisms in Multi-agent systems. In: Trust, reputation, and security: theories and Practice, AAMAS 2002 international workshop. Springer, Berlin, pp 122–127
Quillinan TB, Warnier M, Oey MA, Timmer RJ, Brazier FM (2008) Enforcing security in the agentScape middleware. In: Proceedings of the 1st international workshop on middleware security (MidSec). ACM
Ray M (2009) Authentication gap in TLS renegotiation. http://extendedsubset.com/?p=8
Rescorla E, Ray M, Dispensa S, Oskov N (2010, Feb) Transport layer security (TLS) renegotiation indication extension. Internet Engineering Task Force (IETF)
Riordan J, Schneier B (1998) Environmental key generation towards clueless agents. Mobile agents and security. Springer, Berlin 15–24

Robertson D (2005) A lightweight coordination calculus for agent systems. In: Declarative agent languages and technologies II, vol 3476/2005. Springer, Berlin, pp 183–197
Robertson D, Giunchiglia F, Harmelen Fv, Marchese M, Sabou M, Schorlemmer M et al (2008) Open knowledge—coordinating knowledge sharing through peer-to-peer interaction. In: Languages, methodologies and development tools for multi-agent systems. 1st International workshop, LADS 2007. Revised Selected and Invited Papers, vol 5118, pp 1–18
Robles S (2008) Trust and security. In: Moreno A., Pavn J. (eds) Issues in multi-agent systems: the agentCities. ES experience (Vol. Chapter 4). Birkhäuser, Basel pp 87–115
Rojas DM, Mahdy AM (2011) Integrating threat modeling in secure agent-oriented software development. Int J Softw Eng (IJSE) 2:23–36
Sabelfeld A, Myers A (2003) Language-based information-flow security. IEEE J Sel Areas Commun 21(1): 5–19
Schneier B (1999) Attack trees. Dr. Dobb's J Softw Tools 24:21–29
Sierra C, Walton C, Robertson D, Gerloff EJ, Li JS, Abian J et al (2008) Report on bioinformatics case studies. Techreport
Silei L, Rui Z, Jun L, Junmo X (2008) A novel security protocol to protect mobile agent against colluded truncation attack by cooperation. In: International conference on cyberworlds, pp 186–191
Sit E, Morris R (2002) Security considerations for peer-to-peer distributed hash tables. In: IPTPS'01: revised papers from the 1st international workshop on peer-to-peer systems. Springer, pp 261–269
Sun B, Chen H (2011) Communication security in MAS with XML security specifications. Appl Mech Mater 65:251–254
Sycara K, Paolucci M, Van Velsen M, Giampapa J (2003) The RETSINA MAS infrastructure. Auton Agents Multi Agent Syst 7:29–48
Tan H, Moreau L (2002) Extending execution tracing for mobile code security. In: 2nd International workshop on security of mobile multiAgent systems (SEMAS 2002), Bologna, pp 51–59
Tan JJ, Poslad S, Xi Y (2004) Policy driven systems for dynamic security reconfiguration. In: Proceedings of the 3rd international joint conference on autonomous agents and multiagent systems (AAMAS), vol 3. IEEE Computer Society, pp 1274–1275
Tekbacak F, Tuglular T, Dikenelli O (2009) An architecture for verification of access control policies with multi agent system ontologies. In: COMPSAC'09: Proceedings of the 2009 33rd annual IEEE international computer software and applications conference. IEEE Computer Society, pp 52–55
Tekbacak F, Tuglular T, Dikenelli O (2011) Policies for role based agents in environments with changing ontologies. In: The 10th international conference on autonomous agents and multiagent systems (AAMAS 11), Taipei, pp 1335–1336
Thirunavukkarasu C, Finin T, Mayfield J (1995) Secret agents—a security architecture for the KQML agent communication language. In: Intelligent information agents workshop (CIKM'95)
Traynor P, McDaniel P, Porta TL (2008) Security for telecommunications networks: future directions and challenges. Springer, US
van't Noordende G, Brazier FM, Tanenbaum AS (2004) Security in a mobile agent system. In: The 1st IEEE symposium on multi-agent security and survivability, pp 35–45
van't Noordende GJ, Overeinder BJ, Timmer RJ, Brazier FM, Tanenbaum AS (2009) Constructing secure mobile agent systems using the agent operating system. Int J Intell Inf Database Syst (IJIIDS) 3:363–381
Vazquez-Salceda J, Padget JA, Cortes U, Lopez-Navidad A, Caballero F (2003) Formalizing an electronic institution for the distribution of human tissues. Artif Intell Med 27:233–258
Vila X, Schuster A, Riera A (2007) Security for a multi-agent system based on JADE. Comput Secur 26:391–400
Vitabile S, Conti V, Militello C, Sorbello F (2008) An extended JADE-S based framework for developing secure multi-agent systems. Comput Stand Interfaces 31:913–930
Wagner G (1997) Multi-level security in multiagent systems. In: Proceedings of the 1st international workshop on cooperative information agents. Springer, London, pp 272–285
Wahbe R, Lucco S, Anderson T (1993) Efficient software-based fault isolation. In: The 14th ACM symposium on operating systems principles, pp 203–216
Wang H, Varadharajan V, Zhang Y (1999) A secure communication scheme for multiagent systems. In: PRIMA'98: selected papers from the 1st Pacific Rim international workshop on multi-agents, multiagent platforms, vol 1599. Springer, London, pp 174–185
Wong HC, Sycara K (1999) Adding security and trust to multi-agent systems. In: Proceedings of autonomous agents'99 workshop on deception, fraud, and trust in agent societies, pp 149–161
Xiao L (2009) An adaptive security model using agent-oriented MDA. Inf Softw Technol 51:933–955
Yu E, Cysneiros LM (2002) Designing for privacy and other competing requirements. In: 2nd Symposium on requirements engineering for information security (SREIS^{TM}02), Raleigh

Yue X, Qiu X, Ji Y, Zhang C (2009) P2P attack taxonomy and relationship analysis. In: ICACT'09: Proceedings of the 11th international conference on advanced communication technology. IEEE Press, pp 1207–1210
Zaslavsky A, Indrawan M (2004) A buddy model of security for mobile agent communities operating in pervasive scenarios. Proc. Australas Inf Secur Data Mining Web Intell Softw Int 32:17–25