UNIVERSITEIT AMSTERDAM

VU Research Portal

Are mobile agents outlawed processes?

Brazier, F.M.; Oskamp, A.; Schellekens, M.H.M.; Wijngaards, N.J.E.

document version
Publisher's PDF, also known as Version of record

Link to publication in VU Research Portal

citation for published version (APA)
Brazier, F. M., Oskamp, A., Schellekens, M. H. M., & Wijngaards, N. J. E. (2003). Are mobile agents outlawed
processes? In A. Oskamp, & E. Weitzenbroeck (Eds.), *Proceedings of LEA 2003: The law and electronic agents*
Unipub.

ARE MOBILE AGENTS OUTLAWED PROCESSES?

Frances Brazier[1], Anja Oskamp[2], Maurice Schellekens[3] and Niek Wijngaards[1]

[1] Intelligent Interactive Distributed Systems, Faculty of Sciences
Vrije Universiteit Amsterdam, de Boelelaan 1081a, 1081 HV, Amsterdam, The Netherlands
Email: {frances,kubbe,niek}@cs.vu.nl
Phone: +31 - 20 - 444 7737, 7756; Fax: +31 - 20 - 444 7653

[2] Computer and Law Institute, Faculty of Law
Vrije Universiteit Amsterdam, de Boelelaan 1105, 1081 HV, Amsterdam, The Netherlands
Email: a.oskamp@rechten.vu.nl
Phone: +31 - 20 - 444 6215; Fax: +31 - 20 - 444 6230

[3] Center for Law, Public Administration and Informatization, Faculty of Law
Tilburg University, P.O. Box 90153, 5000 LE, Tilburg, The Netherlands
Email: M.H.M.Schellekens@uvt.nl
Phone: +31 - 13 - 466 - 8044; Fax: +31 - 13 - 466 8149

Abstract. An important characteristic of mobile agents is that they often do not run on their user's platform, but on the platform of someone else. There often is no pre-existing relation between the 'owner' of the running agent's process and the owner of the platform on which the agent process runs. When there are conflicts the position of the owner is not clear: is he allowed to slowdown the process or even remove it from the system? And how can the interests of the owner of the agent be protected? This paper explores the legal and technical perspectives to protecting the integrity of agent processes.

1. Introduction

Mobility and relative autonomy are important characteristics of agents. Combined with a virtually borderless Internet, these characteristics make it possible that software agents are capable of migrating once, or more often, to run on different platforms to perform their tasks. A platform may host multiple agents and usually allocates a process for each running agent. It is becoming less evident that the owner of a software process and the owner of the platform are the same or are even mutual related, either institutionally or contractually. Nevertheless, the 'owner' of the running agent's process and the owner of the platform on which the agent runs both run the risk that their integrity may be violated by the other. When it comes to a confrontation, the owner of the platform has the strongest position: he can slow down or even terminate the running agent's process. This raises the question whether a running agent's process is at the mercy of the platform owner.

This paper investigates whether the law protects the integrity of platforms and running agents' processes, and if so how. It also considers the circumstances when the platform owner may have the upper hand. Technically, it is investigated what measures can be taken to protect the integrity of both the agent and the platform and which counter measures are possible. Questions of law are dealt with according to Dutch law. The words 'computer system' and 'platform' are used interchangeably in this paper.

2. Legal Issues

2.1 Availability and Integrity

Legally, the integrity of a computer system is protected in both criminal law and civil

law. Civil ownership provides a basis to act against infrictions of integrity. Criminally, a computer system is considered as a physical object, the damaging of which is a criminal offence if it is done intentionally and without right (see art. 350 Dutch Criminal Code). For an object to be damaged, it is sufficient that it has ceased to function (HR 19 October 1971, NJ 1972, 33); it need not be physically damaged. E.g. erasing relevant parts of a computer's operating system damages the system, provided it does no longer function. Erasing a single Word-document from a computer's hard disk leaves the functioning of the computer intact and thus does not lead up to a damaged computer.

From the foregoing it appears that it is possible to affect the integrity of data without affecting the integrity of the physical carrier of the data. This raises the question how the integrity of data as such is protected in law. The integrity of data is protected in both criminal and civil law. In civil law the affection of the integrity of data may constitute an unlawful act. In criminal law the integrity of data as such is the subject of specific criminal offenses (see art. 350a and 350b Dutch Criminal Code). The convention on cybercrime gives an example of the statutory provision that is representative of the national legal provisions protecting availability and integrity of data.

Article 4.1 Convention of Cybercrime (CoC) deals with data interference::

Article 4 – Data interference
   1   Each Party shall adopt such legislative and other measures as may be necessary to establish as criminal offences under its domestic law, when committed intentionally, the damaging, deletion, deterioration, alteration or suppression of computer data without right.

We now focus on the question: may a system administrator remove or modify agent processes without the consent of their owners? Note that when in this paper the 'owner' of an running agent's process is mentioned, the word 'owner' denotes the person who other than a system administrator has control over the running agent. Often this may be the person in whose interest the agent runs. The word 'owner' is thus not used in its legal meaning when used with respect to agents or agent processes. For stylistic reasons, the word 'user' is also used as an alternative for 'owner'.

In analyzing the issue of removal or modification of agent processes, two situations must be discerned. On the one hand, it is possible that the system administrator and the user of the software agent have a contractual agreement about the use of the platform by the agent. In this case, their mutual relation is governed by their contract and its terms and conditions. We do not pursue this topic in this paper. On the other hand, it is possible that no contractual agreement exists. In this case, the law must provide the answer. As we saw, statutory law protects the availability and integrity of data. Data are however not processes, but it seems reasonable to assume that processes cannot be terminated without removing or altering data and that comparable considerations apply with respect to processes.

From the provision of art. 4 CoC, it becomes clear that an affection of the integrity of data is only a criminal offense if it is done 'without right'. It is thus relevant to determine whether a system administrator acts without right (or not) if he removes an agent's process from his platform or modifies it. On the one hand, the system administrator is the owner of his platform, or he acts on behalf of the owner. An owner may in general remove or alter data and processes that reside or run on his platform. On the other hand, the system administrator has allowed the software agent to commence running on his platform at some point in the past. Therewith he may or may not have raised certain expectations. Furthermore, it is the question whether he may exercise his rights as an owner without regard for the interests of other people, viz. the 'owner' of an agent's

process running on his platform. In order to identify the circumstances that are relevant two judgments, one from Dutch and one from American case law are analyzed below. American case law is of course not directly of interest for Dutch law, and is solely used for identifying relevant circumstances in dealing with integrity problems of software agents.

2.2 Relevant caselaw: Xs4all/ABFAB

The judgement of Hof Amsterdam 18 July 2002 LJN-number AE5514 (ABFAB/Xs4all) focuses on the question whether an ISP (Xs4all) can require a sender of commercial e-mail to refrain from sending mail to the subscribers of the ISP. Xs4all based her contention that an ISP may indeed require so, on her ownership of computer capacity, transmission capacity and subscriber collection. The court held however that the nature of the service that Xs4all offered, implied that the general public must have the possibility to send e-mail to her subscribers. This and the fact that e-mail is of increasing importance in society, made that the sending of e-mail cannot easily be said to be a violation of Xs4all's right in the computer capacity etc. It fits in better with the public character of the service-provision to accept that the freedom that Xs4all claims for herself as an owner is bound to certain restrictions.

For a system administrator (modifying an agent) this means that a reference to his (employer's) ownership of the system may not be enough to establish that he did not act 'without right'. This is especially so if the service is public in nature and is relevant for society. The system administrator will have to bring by additional circumstances in order to construe his authority to touch the integrity of the mobile agent. Such additional circumstances may be found in the reason the system administrator has to modify the agent, e.g. to maintain system integrity.

2.3 Relevant caselaw: Bidder's Edge/eBay

eBay is an Internet-based, person-to-person trading site (eBay, Inc. v. Bidder's Edge, Inc. 100 F. Supp. 2d 1058, N.D. Cal. 2000). eBay offers sellers the ability to list items for sale and prospective buyers the ability to search those listings and bid on items. Bidder's Edge (BE) is an auction aggregation site designed to offer on-line auction buyers the ability to search for items across numerous on-line auctions without having to search each auction individually. The information available on the BE site is contained in a database of information that BE compiles through access to auction sites such as eBay.

eBay demanded that BE ceased accessing the eBay site, because BE's queries for filling her database consumed capacity of eBay's system. The court found that BE's access to the site constituted a trespass to chattels. A trespass to chattels is defined by the court as an intentional interference with the possession of personal property has proximately caused injury. See Thrifty-Tel v. Beznik, 46 Cal. App. 4th 1559, 1566 (1996). If this trespass consists of accessing a computer system, the plaintiff must establish that: (1) defendant intentionally and without authorization interfered with plaintiff's possessory interest in the computer system; and (2) that defendant's unauthorized use proximately resulted in damage to plaintiff. See Thrifty-Tel, 46 Cal. App. 4th at 1566; see also Itano v. Colonial Yacht Anchorage, 267 Cal. App. 2d 84, 90 (1968).

With respect to the first requirement, the court concluded that BE's activity is sufficiently outside of the scope of the use permitted by eBay that it is unauthorized for the purposes of establishing a trespass. Although the website is publicly accessible, eBay does not grant the public unconditional access and does not generally permit the type of automated access made by BE. In fact, eBay explicitly notifies automated visitors that their access is not permitted. BE continued however to "crawl" eBay's web

site even after eBay demanded BE to terminate this activity. The court decided that even if BE's web crawlers were authorized to make individual queries of eBay's system, BE's web crawlers exceeded the scope of any such consent when they begin acting like robots by making repeated queries. With respect to the second requirement the court held that eBay suffered damage by the mere fact that BE's use is appropriating eBay's personal property by using valuable bandwidth and capacity, and necessarily compromising eBay's ability to use that capacity for its own purposes.

This judgment identifies issues in the case of a system administrator removing unwanted agent processes In the first place, it is relevant under what conditions the system administrator has allowed a software agent to use his system. In this respect, it is relevant that the software agent or its user can recognize that the system administrator imposes conditions and the nature of these conditions. In the second place, it is relevant to what extent the software agent uses the capacity of the system administrator's system. In the above case, it wasn't necessary that alternative uses were frustrated by the fact that BE used the system. It was sufficient that capacity was consumed that could have been used by eBay or its customers.

2.4 Analysis

The system administrator, as the owner, allows software agents to use his agent platform. The system administrator may or may not bind the use by the software agents to certain conditions. At a later point in time the system administrator may want to disallow the use of his platform. What is he to do with running agent processes? Can these processes only be prematurely removed from the platform with the consent of the agent's user or can the system administrator remove agent processes from his platform without consent of the agent's user or even against their will? A preliminary analysis of case law has been performed in order to find the arguments pro and contra interference by the system administrator. This analysis made clear that the following circumstances are relevant:

- Has the owner of the platform stated conditions to the use of his system? Has the agent process or its 'owner' violated these conditions? Does the absence of conditions raise expectations about the usability of the platform for use by software agents? With respect to the conditions other issues may be relevant, such as: is it recognisable that conditions are being imposed? Is the nature of these conditions recognisable for the software agent or its user?
- What interest has the owner of the platform to disallow further use of his system? Does he suffer damage? Is the platform's integrity affected? Are – real or potential – other uses of system capacity precluded by the software agent?
- What is the nature and seriousness of the measures taken against the software agent? Requirements of proportionality and subsidiarity diminish the system administrator's leeway.
- What is the nature of the service provided to the users of software agents? Is it a public service? Is the service important for society, such as an e-mail service? To what extent must the interests of the 'owner' of the agent process in its unencumbered functioning be protected?

For practical purposes it is of course wise to log the pertinent situations and actions in order to (be able to) proof the facts that underpin ones position in court.

3. Technical Issues

In computer systems, integrity is an important feature: improper alteration of a system (and its assets) should be detectable and recoverable (Anderson, 2001; Tanenbaum and

Steen, 2002). The integrity of both the agent and the agent platform is important in the context of this paper. Both an agent's and an agent platform's integrity need to be protected from other agents and agent platforms. The integrity of users and system administrators is not taken into account in this paper.

3.1 Agent Integrity

An agent needs to be protected from unwanted alterations, caused by system malfunctions or malicious entities, e.g. other agents or agent platforms. An altered agent may act differently (potentially maliciously), thereby risking its usefulness and potentially also the reputation of the agent itself and its users. Altering agents which carry confidential information (e.g., their human counterpart's credit card information) may cause more damage than altering an agent which doesn't. Note that some agents are more crucial to the correct functioning of an application than others, when, e.g., an agent in a swarm application becomes corrupted, the effect may be negligible. To protect the integrity of an agent it is necessary that the agent has a unique identity, which distinguishes the agent from all other agents.

Techniques to detect improper modifications are mainly based on tracing alterations to an agent. A mobile agent may need to carry these traces during its life span if agent platforms 'en route' cannot be trusted. One technique is to use an agent container (Karnik and Tripathi, 2001; Noordende, Brazier and Tanenbaum, 2002) as the unit for transportation of an agent and its associated information, as alterations are securely logged. Alternatively, the agent may regularly visit a trusted third party for an integrity check.

Techniques to recover from improper alterations are usually based on restoring information from a copy (a 'backup') at a secure location. Agents with built-in fault tolerance may recover more easily, even if one process of the agent becomes corrupted or is terminated (Marin, Sens, Briot and Guessoum, 2001). Error correction techniques (e.g. as used for data communication over a telephone line) are rarely used as their effectiveness is limited. If no backup is available, then perhaps information can be restored by replaying past behaviour (and consulting relevant agents and agent platforms).

An agent also needs protection from an agent platform. Unfortunately, an agent platform has enormous power over an agent which cannot easily be restrained. Some solutions advocate special purpose hardware, others develop cryptographic techniques (e.g. Sander and Tschudin, 1998). Protocols requiring agent platforms to provide a rationale for their modifications to agents may also be needed. More research is clearly required.

An agent may need to provide 'a reason' for its actions, e.g. to determine liability of users. For this purpose, an agent may be obliged to keep a trace of information (including orders) received from users and possibly from other agents, together with its responses.

In sum, standards are required for protocols and auditing logs concerning agent integrity; this area requires more research, especially with respect to heterogeneous large-scale open systems.

3.2 Agent Platform Integrity

An agent platform hosts agents, possibly from different users and involved in different applications. Agents residing at an agent platform compete for its resources (such as processor cycles, bandwidth, disk space) and services (such as (reliable and secure) communication, mobility, automated updates, streaming connections). Breaches of the integrity of an agent platform may result in malfunctioning of agents being hosted, and possible lead to loss of reputation of the agent platform's holder. Note that an agent

platform needs not only be protected from agents it hosts, but also from other agent platforms. To preserve integrity of an agent platform, the agent platform needs to have policies for access control, prevention, monitoring and corrective action; usually provided by a system administrator. Existing agent platforms currently address these and other security issues (including FIPA-OS, e.g. Poslad and Calisti, 2000).

Access control

Policies for access control, i.e. deciding whether an agent is to be hosted by the agent platform, are usually based on the agent's identity, and possible on information of its user, or other characteristics. In addition, the decision may be influenced by information on reputation or gossip, and observations of the (expected) load of the agent platform itself.

In case of updating the Bidder's Edge database software robots were used to search, amongst others, eBay's database. eBays's system administrator could identify agents such as the Bidder's Edge agents. Their agents could, for example, be identified by the IP-addresses from which they originated.

When deciding to host an agent, constraints may be imposed, including permissions (read/write to specific information, execution of other programs) and resource consumption constraints (e.g., Jansen, 2001). In case of eBay for example, it was technically possible for Bidder's Edge agents to extensively access eBay.

Prevention

An agent platform may assign different levels of 'trust' to agents it hosts. To limit possible damage, preventive measures may be taken. One measure is to check an agent's code for viruses and other malignant behaviour; this is, alas, impossible, as formal verification does not provide a fast answer if any, and software which scans for, e.g., viruses can only detect known or similar problems. A safer approach is to have an agent only use code which is known to be safe, e.g. by regenerating an agent's code using an agent factory (Brazier, Overeinder, Steen and Wijngaards, 2002). Another measure is to place the agent in a safer, restricted, execution environment, known as a 'sandbox', e.g. the Java Virtual Machine, with which unwanted code commands can be intercepted. In the case of Ebay, preventive measures were not of relevance, as far as can be deduced from the judgement.

Monitoring

To provide a reliable environment for each of its agents, an agent platform needs to monitor itself to detect unwanted behaviour. Often, unwanted behaviour is caused by unexpected resource consumption of correctly functioning agents: e.g. when all agents access specific databases in the system congestion and overloading may occur. Detection of unwanted behaviour is complicated by the level of detail which may be needed, and the overhead of information transfer and processing generated by the monitoring system: a policy is required. It is, in general, not possible to monitor all actions of all agents in minute detail; but it is possible to monitor resource consumption on computers hosting these agents and use this information for management purposes (Mobach, Overeinder, Wijngaards and Brazier, 2003). For example, an agent which increases its use of the processor and sends thousands of messages per time unit, may be involved in an unwanted activity. Monitoring message content may only be feasible in specific cases, and has an impact on the privacy of agents and their users.

Another source of information is from outside entities such as system administrators, agents, trusted third parties, and other agent platforms. This information needs to be

assigned a level of trust, before a subsequent action is taken.

Corrective actions

An agent platform requires guidelines for which automated corrective action to take in which situation, as too many potential situations may emerge to involve a human system administrator. A misbehaving system administrator's agent may be treated differently than someone else's misbehaving untrusted agent. Possible corrective actions include:

- slowing down an agent by allocating fewer processor-cycles,
- adjusting an agent's permissions and resource allocations, e.g. to limit message sending to once per large-time-unit,
- moving an agent into a sandbox,
- forcing an agent to migrate to another agent platform (possibly with the agent's latest internal state, otherwise causing the agent to revert to its last saved state),
- returning the agent, by stopping the agent and sending the agent (together with last saved state) to human counterpart(s),
- killing an agent (including deleting its identity from location services),
- updating and distributing reputation and trust information with regard to this agent,
- consulting a system administrator.

In the eBay case the agent platform could consider isolating the suspicious Bidder's Edge agents in such a way that is not harmful to eBay's system, possibly while storing enough data in order to allow the agents to resume on another system (this was not an option for eBay). A more drastic option which eBay's system administrator could consider is killing Bidder's Edge agents. In all cases the system administrator could consider notifying the owner of the agents (Bidder's Edge) of their illegal presence and their potential termination.

The agent platform is required to maintain a trace of its corrective actions possibly with a rationale, and may require a protocol for notifying agents or their users related to the subjects of corrective actions. A rationale should specify the (potential) damages to be caused by the agents involved and the appropriateness of the corrective actions taken.

4. Discussion

The conditions under which access to a computer system is granted are relevant. Nevertheless, a number of outstanding questions with respect to these conditions remain. The conditions for use of the system must, e.g., be clearly communicated to the software agent and its user by the system administrator. A straightforward way to 'communicate' the conditions is perhaps a system that is able to 'physically' prohibit any use of the system that violates the conditions. If a platform can actively prohibit usage which violates the conditions, then a question is whether a platform can distinguish between usage satisfying or violating the conditions.

Technical measures may have an all or nothing character: allowing either too much or too less. In the eBay v. BE case, the former blocked at one point the latter's IP-addresses in an attempt to prohibit the uses that violated the conditions. This measure did however not differentiate between uses that satisfied or violated the conditions. This case also highlights a second drawback of physical measures: BE easily robbed the measure of its effectiveness by using a proxy-server, thus hiding her IP-address and nullifying the effect of eBay's IP-blocking. In addition, technical measures may, on their own, make the precise conditions for use insufficiently explicit: was e.g. BE's access through the proxy-server allowed since it was not blocked or must IP-blocking be understood to be a general denial of access?

It is indispensable to have a protocol that facilitates communication of the conditions of use much more clearly and explicitly than is the case with 'coercive technical measures'. A distinction needs to be made with respect to the recipient of the conditions of use. If they are to be communicated to the software agent, a protocol is required that can express that conditions are being communicated and specify these conditions, in such a way that this is meaningful for the software agent. In addition, it is desirable for communication of conditions to guarantee non-repudiation of receipt, i.e. it must be provable that the software agent has undeniably received and understood the conditions. An appropriate protocol could, e.g., require a software agent to acknowledge receipt of the conditions. If the conditions are to be communicated to the owner of the software agent, the owner must be identifiable and have a contact address. Again, a protocol is necessary, including non-repudiation of receipt.

For the time being, system administrators may still be confronted with the presence of 'suspicious' agent's processes on their systems. The technical analysis has shown that there is a whole range of measures a system administrator can take against software agents violating conditions for use. These measures range from reducing resources for the agent to removing the agent from the system. The legal demands of proportionality and subsidiarity imply that the least far-reaching measures that are effective must be chosen. The question is, however, the effects of these measures. The answer to this question may be learned by experience as it largely depends on dependencies between large numbers of co-operating agents and does not lend itself very well for imposition by a legislator. However, learning by experience does not come by of its own. Experiences have to be collected and discussed, a subject that lends itself very well for self-regulation by organisations of system administrators. The outcome of such collection and discussion can be laid down in directives that can guide system administrators in their approach to agent processes that go beyond the latter's allowed use.

Acknowledgements

References

Anderson, R. (2001), *Security Engineering: A Guide to Building Dependable Distributed Systems*, Wiley Computer Publishing, New York.

Brazier, F.M.T., Kubbe, O., Oskamp, A., and Wijngaards, N.J.E. (2002), Are Law-Abiding Agents Realistic? *in* Sartor, G. and Cevenini, C. (Eds.), *Proceedings of the workshop on the Law of Electronic Agents (LEA02)*, pp. 151-155.

Brazier, F.M.T., Overeinder, B.J., van Steen, M., and Wijngaards, N.J.E. (2002), Agent Factory: Generative Migration of Mobile Agents in Heterogeneous Environments, *Proceedings of the 2002 ACM Symposium on Applied Computing (SAC 2002)*. pp. 101-106.

Jansen, W. (2001), A Privilege Management Scheme for Mobile Agent Systems, *First International Workshop on Security of Mobile Multiagent Systems, Autonomous Agents Conference*, ACM Press.

Karnik, N. and Tripathi, A. (2001), Security in the Ajanta Mobile Agent System, *Software - practice and experience*, pp. 301-329.

Marin, O., Sens, P., Briot, J-P. and Guessoum, Z. (2001), Towards Adaptive Fault-Tolerance For Distributed Multi-Agent Systems, in *Proceedings of ERSADS'2001*, Bertinoro, Italy, pp. 195-201.

Mobach, D.G.A., Overeinder, B.J., Wijngaards, N.J.E. and Brazier, F.M.T. (2003), Managing Agent Life Cycles in Open Distributed Systems, *Proceedings of the 18th ACM Symposium on Applied*

Computing, ACM Press, pp. 61-65.

Noordende, G. van 't, Brazier, F.M.T. and Tanenbaum, A.S. (2002), A Security Framework for a Mobile Agent System, *in* K. Fischer and D. Hutter (Eds), *Proceedings of the 2nd International Workshop on Security in Mobile Multiagent Systems (SEMAS 2002), associated with AAMAS-2002, Bologna, Italy* , DFKI Research Report RR-02-03, Deutsches Forschungszentrum fur Künstliche Intelligenz, pp. 43-50.

Poslad, S and Calisti, M (2000), Towards Improved Trust and Security in FIPA Agent Platforms, *Proceedings of the Autonomous Agents 2000 Workshop on Deception, Fraud and Trust in Agent Societies*, Barcelona, Spain, pp. 87-90.

Sander, T. and Tschudin, C.F. (1998), Protecting Mobile Agents Against Malicious Hosts, *in* G. Vigna (Ed), *Mobile Agents and Security*, Lecture Notes in Computer Science, **1419**, Springer-Verlag, pp. 44-60.

Tanenbaum, A.S. and Van Steen, M. (2002), *Distributed Systems: Principles and Paradigms*, Prentice Hall, New Jersey.

Wijngaards, N.J.E., Overeinder, B.J., Steen, M. van, Brazier, F.M.T. (2002), Supporting Internet-Scale Multi-Agent Systems, *Data and Knowledge Engineering*, **41**(2-3), pp. 229-245.

Wong, H.C. and Sycara, K. (1999), Adding Security and Trust to Multi-Agent Systems, *Proceedings of Autonomous Agents '99 Workshop on Deception, Fraud, and Trust in Agent Societies*, pp. 149-161.