Privacy-Invasive Software and Preventive Mechanisms

Martin Boldt and Bengt Carlsson
Blekinge Institute of Technology, School of Engineering
SE-372 25 Ronneby, Sweden
{martin.boldt, bengt.carlsson}@bth.se

***Abstract*-Computers are increasingly more integrated into peoples' daily lives. In this development, user privacy is affected by the occurrence of privacy-invasive software (PIS), sometimes loosely labelled as spyware. The border between legitimate software and PIS is vague and context dependent, at best specified through End User License Agreements (EULA). This lack of spyware definition result in that current countermeasures are bound to noticeable misclassification rates. In this work we present a classification of PIS from which we come to the conclusion that additional mechanisms that safeguard users' consent during software installation is needed, to effectively counteract PIS. We further present techniques that counteract PIS by increasing user awareness about software behaviour, which allow users to base their software installation consent on more informed decisions.**

I. INTRODUCTION

A powerful component in any business strategy is user/customer information gained either with or without violating the privacy and security of users. In general, the company with the most information about its customers and potential customers is usually the most successful one [25]. This situation has given raise to a parasitic market, where questionable actors focus on short time benefits when stealing personal information for faster financial gain [6][9][18]. In addition, this situation is further fueled by money from advertisers who want their online ads distributed; often ending up in advertising software (*adware*) on users' trespassed computers [8][29]. Throughout this paper we describe such software as *privacy-invasive software* (PIS); ranging from legitimate software and spyware to truly malicious software (*malware*). Some of the most common technical problems associated with PIS include [1][9][17][21][32]:

- Unauthorised resource utilisation causing deteriorate system stability
- Third party software installation without user consent
- Displaying of unsolicited advertising content at varying frequency and substance
- Settings and properties of other software are being changed
- Personally identifiable information is covertly transmitted to third parties
- Poor or non existing removal mechanisms
- Time invested in recovering systems from such unsolictited programs

Various sources claim that up to 80% of all Internet connected computers have one or more spyware infections on their computer systems [3][14][35]. One general problem concerning these investigations is the lack of a proper definition of what is being measured and investigated, i.e. what spyware actually is [17]. There exist numerous terms that are used as synonyms to spyware, e.g. evilware, scumware, snoopware, thiefware, or trackware. These terms are all used to group software together that is somehow being disliked by users, regardless of being illegal or not [6][27]. The border between legitimate and illegitimate software is non existing, or at least very vague and context dependent. Wrongly classified software render in that legitimate software vendors get their products labelled as spyware, or some of its synonyms. To protect their business, these vendors take legal actions against the responsible developers of such inaccurate spyware countermeasures [30]. This means that the absence of a proper spyware definition result in legal disputes between software vendors and developers of spyware countermeasures. In the end, users have no alternative but to put their confidence in inaccurate countermeasure tools that leave them with trespassed systems [19].

Today, much software that incorporates advertisements is regarded as spyware by the users, which in

turn render in distrust of advertised financed software in general. This is very unfortunate since advertise financed software development is a powerful tool, that allow vendors to provide a product "free of charge". This way users do not have to pay since the software developer get revenues from the advertising agency for delivering ads to the users [8]. But as have been shown over recent years, there must exist regulations and rules of conduct that control how these techniques interface with the users [6][11].

II. Spyware and User Consent

In major operating systems today, e.g. Microsoft Windows, software installations are carried out in a rather ad-hoc manner. Basically the user retrieves a software package from a source, such as a non trusted Web site, and then executes it on the system. Even though some operating systems offer more standardized installation methods, e.g. FreeBSD's ports system, this method is still available in parallel [15]. The disadvantage with this ad-hoc software installation method is the lack of control over what software that enters the system [4][26]. Also, the instrumentation that allow users to evaluate the software prior to the actual installation is inaccurate, or non-existing. Without such instrumentation it is troublesome for the users to give an *informed consent* for the software to enter their system. Our use of the term informed consent is based on the work by Friedman, Felten, and Millett, in which they divide the word *informed* into: *disclosure* and *comprehension* [16]. The word *consent* is divided into: *voluntariness, competence,* and *agreement*.

Today, users give their consent to software installations by accepting the terms stated in End User License Agreements (EULA). Unfortunately many computer users today are not capable of comprehending these EULAs, since they are disclosed in a very legal, formal, and lengthy manner [30]. A license agreement that includes well over 6000 words (compared to US constitution which includes 4616 words) is not unusual, which users need a degree in contract law to understand [6]. Even users that have the prerequisite knowledge do not have the time to read through lengthy EULAs each time they install new software, resulting in that most users simply accept the license agreements without reading through them first [27]. Due to this it is next to impossible for users to reach sustainable trust-related decisions during software installation since today's computing systems doesn't provide sufficient support for making such decisions. By not being able to evaluate software entering their system, users unknowingly allow illegitimate software to enter.

In addition to the threat from infection of inferior software, the lack of informed consent during software installations also impose very vague user awareness on what they have agreed upon; e.g. users can't deduce the pop-up ads on the computer screen with an earlier approval of the corresponding EULA [6]. The underlying problem is how software vendors should disclose information to their customers about their product's implications on the user and the user's system. More importantly, how to present it in a clear and comprehensive manner towards the users. Vendors of PIS target this problem when using deceptive methods for deploying their software on users' machines. Since these attacks target the human/computer interface it should be understood as a semantic problem, not a syntactic one, or as Bruce Schneier put it "any solutions will have to target the people problem, not the math problem" [24].

In the end, it is of paramount importance, for both users and legitimate software vendors, that a clear separation between acceptable and unacceptable software behaviour is established [6][27]. However, we believe that an acceptable behaviour is context dependent, i.e. what one party regard as acceptable software behaviour is regarded as unacceptable by others. Therefore, users need to know what they install and, with the help of aiding mechanisms, learn to distinguish between what they believe is acceptable and unacceptable software, prior to any actual software installation on their system.

III. Software Classifications

In the introduction section we defined PIS as any software violating user's privacy, ranging from legitimate software to malware, with spyware in-between. As a matter of fact we intend to exclude spyware from the list of PIS by either classifying them as legitimate software or malware, from a preventive point of view. Before taking this initiative we first look at current spyware classifications followed by the classification of PIS before, in the discussion area, coming back to the prevention view.

A. Spyware Classification

The term spyware is used at two different abstraction levels [1][17]. In the more precise version, the term is defined as "any software that monitors user behaviour, or gathers information about the user without adequate notice, consent, or control from the user" [1]. On the other hand, the more abstract use of the term has showed itself hard to define [1][6]. This notion of the term is often used to describe any software that is disliked by users, even though properly introduced toward the users and with their "uninformed" consent. As a group of software, spyware is located in between legitimate and malicious software, but unfortunately the exact border has not been unveiled [6]. If we could identify these borders and thereby both differentiate between spyware and malware, and secondly between spyware and legitimate software we would have come close to encapsulating spyware.

One difference between spyware and malware is that spyware, to a large extent, target sensitive but not critical information, while malware do. However, the main difference between spyware and malware is that spyware present users with some kind of choice during the installation or entrance into their system [17]. This means that any software that installs itself without asking for the user's permission should no longer be treated as spyware, but instead as malware. Do note that there exists

malware with the same properties and behaviour as spyware, but these are no longer situated in between legitimate software and malware [28]. The difference between legitimate software and spyware is based on the degree of user consent associated with certain software, i.e. informed consent means legitimate software and otherwise spyware. Hence, the problem really boils down to inaccurate mechanisms for users to evaluate the software's degree of appropriateness to enter their system (see Section 2). Even though there exist no accurate definition of the wider use of the term spyware, legislations against these threats is being implemented in a number of nations. Specifying legislative actions against something that is not properly defined and constrained impose risks on items located close to the target [34].

TABLE I: CLASSIFICATION OF SPYWARE WITH RESPECT TO USER AWARENESS AND PERMISSION (HIGH OR LOW) AND USER CONSEQUENCE (POSITIVE OR NEGATIVE)

	Positive Consequences	**Negative Consequences**
High Consent	Overt providers	Double agents
Low Consent	Covert supporters	Parasites

Warkentin et al. present a classification of spyware as the combination of *user consequences* and the *level of consent* from the user; see Table I [34]. User consequences are divided into either positive or negative, and the consent level is represented as a continua spanning between high and low. They define spyware in a two by two matrix, with the following four different types: *overt provider*, *covert supporter*, *double agent*, and *parasite*. The overt provider is synonymous to any legitimate software, while a covert supporter have less, or none, user consent. Both types provide the user with some useful service, i.e. the existence of the software is in some sense beneficial for the user. Double agents act as trojan horses, which obtain user consent for providing one task, but really execute another task causing unexpected negative consequences. Finally, the parasite has no user consent what so ever and it impairs negative consequences on the user, and his system

B. PIS Classification

In this paper we further improve the classification provided by Warkentin et al. in three ways. First, we introduce an intermediate value on both the consent and consequence axis, so that the matrix size is increased to three by three. This results in that the consent level is classified as *high*, *medium*, or *low*. By including this middle value we prepare a base for further discussions on an important group of PIS that get user consent, but not as a result of an informed decision.

Secondly, we remove all positive consequences from the grading on the consequence axis. Since we focus on spyware we exclusively focus on the negative consequences towards users. The grading consists of *negligible*, *moderate*, and *severe* negative user consequence. Software with negligible negative consequences include legitimate software which always have some degree of negative impact, e.g. with regard to resource utilisation.

Thirdly, we split user consequence property into both *direct negative consequences* and *indirect negative consequences*. This means that user consequences consists of both the direct consequences of the software, i.e. what it is designed to do, combined with the indirect consequences, i.e. how the mere existence of the software affect the whole computer system. Note that the indirect consequences are not visible in our matrix, but is used when describing the various entries inside it.

Since the wider use of the term spyware includes so much more than only information gathering software, any classification trying to capture all these various types of behaviour must basically contain any software that includes questionable user consent and negative user consequences [17]. Therefore we enumerate every software permutation that exist as a combination of user consent and negative software consequences. By doing so we capture every form of "active" legitimate software, spyware and malware. Our three by three matrix then consists of nine different types of PIS, ranging from legitimate software to full fetched malware, such as worms or viruses. Anyone of these nine types of PIS collect information about the user and his/her system, or negatively affects the user's computer experience.

TABLE II: CLASSIFICATION OF PRIVACY-INVASIVE SOFTWARE WITH RESPECT TO USER'S INFORMED CONSENT (HIGH, MEDIUM AND LOW) AND NEGATIVE USER CONSEQUENCE (NEGLIGIBLE, MODERATE AND SEVERE).

	Negligible Negative Consequences	**Moderate Negative Concequences**	**Severe Negative Conceuences**
High Consent	1) Legitimate software	2) Adverse software	3) Double agents
Medium Consent	4) Semi-transparent software	5) Unsolicited software	6) Semi-parasites
Low Consent	7) Covert software	8) Trojans	9) Parasites

Our classification of PIS in Table II presents three different groups of software, the first has high user consent to provide its service (top row), the second group include software that has some kind of user consent but it does not correspond to the software's full behaviour (middle row), and the last group include software that does not have any consent from the user at all (bottom row). By inspecting these three groups as software with PIS behaviour, we can narrow down what types of software that should be regarded as spyware, and which ones that should not. The top row includes software that has

received full permission from the user, and where the behaviour is fully transparent towards the user, i.e. they should not be regarded as spyware.

1. *Legitimate software* has the user's full consent and provides some beneficial service to the user. Information collection and other potential spyware behaviour should be treated as a legitimate functionality, as long these are fully transparent to the user.
2. *Adverse software* has the same properties as any legitimate software, but with the exception that it renders in increased negative consequences. However, this software does not include any covert behaviour from the users and should therefore not be labelled as spyware.
3. *Double agent* has been designed to cause a number of negative effect on the user's system, but where all these consequences are fully transparent to the user, i.e. user has given his/her consent based on an informed decision. One example includes installation of file-sharing tools that are bundled with questionable software. Users chose to install these tools despite being fully aware of the consequences, since the gained benefit of the tool motivates it [17].

The middle row in Table II includes software that has gained some sort of consent from the user, but it was not based on an informed decision and should therefore be regarded as spyware.

4. *Semi-transparent software* includes any software that provides a requested and beneficial service toward the user, but where some of the functionality is not communicated to the user. Even if the covert functionality does not impair any negative consequences, the software could introduce vulnerabilities to the user's system. This is especially alarming since the user is unaware of it and therefore unable to address the threat. Based on this reasoning this type of software should be regarded as spyware.
5. *Unsolicited software* is installed on users systems without their explicit consent, i.e. requested software that covertly installs further software. Commonly, users give their permission to the first software, but are usually, to various degrees, unaware of the existence and behaviour of any third party bundled software. This group of software should be labelled as spyware.
6. *Semi-parasite* is pushed on users when, for instance visiting Web pages. These software often deceive users into thinking they are needed to access for example a Web page. Since no information about the consequences are presented to the user they are left unaware of the covert functionality that cause major negative consequences.

Software that installs and executes without any user consent at all is represented in the bottom row of Table II. By covertly sneaking inside users' systems such software has clearly crossed the line of what should be regarded as acceptable behaviour, and should therefore be labelled as malware. This group of malware is further divided into three types depending on the degree of negative user consequences they are causing.

7. *Covert software* is software that secretly installs themselves on systems without causing any direct negative consequences. See reasoning for semi-transparent software.
8. *Trojan* is software that deceives users into installation in the belief that they provide some beneficial service, but which include convert functionality that impose negative consequences on the user.
9. *Parasite* includes software with pure negative consequences that gains entrance to the user's system without his/her awareness or consent. Once inside, this software does whatever the attacker has designed it to do.

This PIS classification emphasis on user's informed consent, i.e. the user competence and comprehension of software behaviour is essential for the classification of PIS. High comprehension means legitimate software, medium means spyware, and no comprehension means malware, if all users really are able to make such a decision. If not, which is the case today, most of the first row will belong to the spyware section. This classification is user dependent, i.e. more knowledgeable users may change the awareness in a dynamic way, which is further discussed in the next section. The PIS classification also emphasis on negligible, moderate or severe negative consequences when the system is misused. This is essential when more accurate mechanisms to inform users about software behaviours are deployed, allowing users to individually decide what consequences are acceptable (compared to the software's positive effects), and which is not.

IV. PIS Countermeasures

The inner workings of PIS does not necessarily include any malicious behaviours, but rather benign behaviours such as showing content on the screen (advertisements) or sending non-critical information (visited Web sites) through the network. Vendors of countermeasures against PIS do not target "dangerous" behaviours, as is the case with for example antivirus tools; making it harder to separate PIS from legititmate software. In addition, the only property that distinguishes PIS from legitimate software is the lack of user consent. Without proper techniques that safeguard user consent during the installation process, it is impossible for any PIS countermeasure to accurately decide on what software to target.

Current PIS countermeasures are based on centrally governed classifications of what software that should be regarded as spyware and which should not. This model provides a too coarse mechanism to accurately distinguish between the various types of PIS that exists since this relation is based on individual users' perspectives. That is, the degree of user consent needs to be regarded when distinguishing spyware from products that are beneficially tailored toward the users' needs. In the following subsections we present mechanisms that support users when making difficult trust decisions about whether to

allow certain software to enter their system. The goal is *not* to make these trust decisions for the users, but instead to develop mechanisms that support them. In the end it's up to the users to make the decision [13].

A. Software Deeds

In Section 2 we describe the problems that users face when trying to give an informed consent based on the corresponding EULA content. The purpose of using EULAs is to establish a juridical agreement between the software user and vendor, and not to enlighten users about potential implications. However, since no additional standardized methods exists that aid users in comprehending software behaviour, users need to put their faith in third party reviews (e.g. from friends or magazines) [13].

Clearware.org try to increase software transparency towards users based on *software deeds*, which refine the EULA content and present it in a more human readable format [10]. These deeds exist at different comprehension levels and include pictograms (small pictures) that denote various behaviours of software, e.g. if the software display advertisements or if it includes fully functional removal routines. At the easiest level users are able to get a basic understanding about the behaviour of software they are about to install, by simply skimming through a set of pictograms.

In addition to these human readable deeds there also exists a corresponding machine readable deed in XML-format, making it possible for the operating system to automatically filter software in its perimeter based on local user preferences.

B. Software Preferences

By individually configuring software preferences, e.g. with respect to security, privacy, or performance, it is possible for users to define their own level of acceptance for new software. Exporting these preferences into a machine readable format, e.g. XML, allows the operating system to access and enforce them. This would result in that only specific software that matches the user's preferences is allowed to enter their system, e.g. with respect to pop-up advertisements. As soon as a user starts installing new software, including an XML-deed described in previous section, the values extracted from this are automatically matched against the user's preference list. If a match is found, the software is allowed to enter. Otherwise, the operating system would take some secondary action, e.g. notify the user about the mismatch or ask for permission to proceed.

This technique is used by the *Platform for Privacy Preferences* (P3P) to allow Web sites to specify what information they gather from users when visiting their site [7][12]. Users define their own local P3P preferences in their Web browser, which then automatically compares these against the remote preferences distributed by the requested Web server. As a result users' browsers only accept interaction with Web servers whose privacy-preferences correspond to their own local preferences [23].

C. Third Party Software Certification

Software *white-listing* is a technique that uses trusted third parties to certify acceptable software. One example is TRUSTe's *Trusted Download Program* which certifies "privacy-friendly" software [33]. A widespread service using such techniques provides market incentives for vendors of PIS to clearly and unavoidably communicate key functionalities of their software.

Problems concerned with third party certification include what software the third party regard as acceptable and which is not. A single organization that reach wide spread use with their certification will gain powerful influence in deciding what software to certify. Some sort of verification must also be carried out to check so that the software behaviour really corresponds to the certification requirements. Because of the vast amount of software that needs to be certified, some sort of automatic verification of software behaviour is probably needed which harden this approach [22].

The opposite of white-listing is *black-listing* where a trusted third party specifies software that is unacceptable. To some extent anti-malware tools, such as anti-virus software, function as a black-listing mechanism that identify and remove unacceptable software.

D. Collaborative Reputation Systems

Reputation systems include an algorithm which allows members of a community, such as eBay.com, to estimate other members' behaviour before an interaction. A collaborative reputation system presents an interesting method to address PIS, by collecting the experience from previous users' trust decisions regarding installation of software. While techniques such as third party certification or software deeds aim at increasing user awareness, reputation systems instead collect and refine user experiences. This experience is then used in a collaborate manner to inform (novel) users about the general opinion that exist for a specific software. The fundamental idea is that users make more accurate trust decision when incorporating such information, i.e. accompanying rumours from friends.

A first, modest, version of such a reputation system should include simple information about whether users decide to install certain software or not, i.e. if they choose "install" or "cancel". Based on this information, subsequent users are presented with statistics about previous users' installation decisions for a specific software. A more useful system needs to include evaluations of software that previously has been installed, e.g. when users decide to uninstall software they are asked to evaluate it by specifying a grade and a comment. This information is then processed by the reputation system, together with for how long the software were installed on the user's system. When subsequent users wish to install the same

software, they are presented with the collective view on this software, i.e. what previous users' has thought about the software and for how long they decided to use it. Since such a system utilize sensitive information (e.g. what software users install) makes it crucial that privacy and anonymity concerns are properly addressed.

Sandra Steinbrecher presents a design for a "privacy-respecting reputation system", which is based on continual changes between several user pseudonyms inside the reputation system [31]. This approach allows the system to protect user privacy, anonymity, and unlinkability between former actions in the system.

V. Discussion

Since user's informed consent distinguish legitimate software from spyware, it is important to safeguard and support users' authority to make informed decisions about software, a priori to the installation. Without such measures it is insuperable to correctly define and mitigate spyware, based on the software functionality, since this is depending on every single user's relation to specific software, e.g. what one user regard as a spyware another user see as a beneficial tool. This relation is impossible to capture at a later stage by any countermeasure tool, i.e. such tools are condemned to vast classification failures of spyware. From this follows that removal of malware "only" is a technical problem, while removal of spyware is both a technical and juridical problem. In previous investigations we have observed that leading anti-spyware tools are quite inaccurate in their classification of spyware, rendering in notable false alarms and false accusations rates [5].

Current spyware countermeasures are reactive, i.e. are designed to remove known spyware. Protecting users' systems with such techniques often target the threats once already inside systems. In an attempt to improve this situation countermeasure vendors try to broadening their defences, e.g. by relocating into network routers and servers [23]. However, these countermeasures could not properly capture all software labelled as spyware by the users, and at the same time protect legitimate software, due to the individual nature of these classifications. In this paper, we present a number of mechanisms that aid users in their evaluation of software before actually installing them. Mechanisms that improve user awareness fundamentally change the classification of spyware, since fully transparent software never can be labelled as spyware, by definition. Introducing such mechanisms result in a transformation of Table II intoTable III, i.e. the middle row is removed. Any software that presents complete and correct information to the user during the installation is represented as one of the three legitimate software types on the top row. Exactly which one depends on their behaviour and consequences on the system, and the user. Users can with the help of local preferences differentiate between what software, on the top row of Table III, they regard as acceptable and which they don't, i.e. there exist a possibility of individual adjustment for the user. On the other hand, software that does not play by the rules, not presenting complete and correct information, are defined as one of the three software types on the malware row.

Any software not playing by the rules, in terms of properly announcing their intent prior to the installation, should rightfully be targeted and handled by anti-malware tools. This imply that anti-malware tools should not only target software that use exploitable system vulnerabilities to gain entrance to systems, but also software that deceive users about their business by using inferior user disclosure. From this follows that anti-malware mechanisms handles any software not subordinate to the rules of complete prior disclosure and consent from the users. Once this group of software has been excluded, users can depend on the information found in deeds, EULAs and other documentation to be correct. Any "dishonest" software slipping through the anti-malware tools' detection would impact a few initial systems. However, the affected users will surely downgrade the responsible software via the reputation system, so that subsequent users are more restrictive. Over time the reputation will catch up on these questionable software vendors, forming a future deterrent effect.

Table III: Difference between legitimate software and malware with respect to user's informed consent and negative user consequences.

	Negligible Negative Consequences	**Moderate Negative Consequences**	**Severe Negative Consequences**
High Consent	Legitimate software	Adverse software	Double agent
Low Consent	Covert software	Semi-parasites	Parasites

Software certification and deeds described in Section 4, further imply that users' uncertainty about what software that is installed on their system decrease significantly. This render in that any indirect negative consequences associated with unsolicited software are removed, i.e. software with exploitable vulnerabilities that execute on users systems without their awareness. Introducing preventive mechanisms against PIS offer the following three benefits towards the users:

1. A lowest level with regard to software behaviour, deed, and EULA correctness that is accepted for software in general.
2. Basis on which software behaviour and consequence can be evaluated prior to any installation, blocking unacceptable software before entering the system.
3. Possibility for users to define individual software preferences, which allow only a subset of all legitimate software to enter their system.

It is notable that all types of software that currently is targeted by traditional anti-spyware mechanisms are either removed by the introduction of the preventive mechanisms, or are fully covered by anti-malware tools. We therefore believe that, after an initial transitional period, all anti-spyware tools will be outmaneuvered by, or integrated in, already existing

anti-malware tools. These anti-malware tools will act as regulators that safeguard both users' systems from illegal infections, and indirectly protect the correctness of information about software, e.g. in EULAs.

VI. Conclusions

Users need to know what they install, and learn how to distinguish between acceptable and intolerable software, a priori to any software installation. Everyone needs to be presented with complete, accurate and condensed information about the software's functionality during the installation process. We argue that additional mechanisms that safeguard user's informed consent are required to fight PIS effectively.

Our classification of PIS put emphasis on user consent, where high consent means legitimate software, medium consent means spyware and low consent means malware. To exclude spyware from legitimate software and malware, the classification emphasis on negligible, moderate or severe user consequences in an environment of either high or low user consent.

As future work we will develop and evaluate a proof-of-concept PIS preventing reputation system, and a Microsoft Windows environment client.

References

[1] W. Ames, "Understanding Spyware: Risk and Response", in the *IEEE Computer Society*, Volume 6, Issue 5, 2004.

[2] Anti-Spyware Coalition, http://www.antispywarecoalition.org, 2006-05-31.

[3] AOL/NCSA Online Safety Study, http://www.staysafeonline.org, 2006-05-31.

[4] K. P. Arnett and M. B. Schmidt, "Busting the Ghost in the Machine", in *Communications of the ACM*, Volume 48, Issue 8, 2005.

[5] M. Boldt and B. Carlsson, "Analysing Countermeasures against Privacy-Invasive Software", in the *Proceedings of ICSEA'06*, Papeete Tahiti, 2006.

[6] J. Bruce, "Defining Rules for Acceptable Adware", in the *Proceedings of the Fifteenth Virus Bulletin Conference*, Dublin Ireland, 2005.

[7] S. Byers, L.F. Cranor, and D. Kormann, "Automated Analysis of P3P-Enabled Web Sites", in the *Proceedings of the Fifth International Conference on Electronic Commerce (ICEC2003)*, Pittsburgh USA, 2003.

[8] Center for Democracy &Technology, "Following the Money", http://www.cdt.org, 2006-05-31.

[9] E. Chien, "Techniques of Adware and Spyware", in the *Proceedings of the Fifteenth Virus Bulletin Conference*, Dublin Ireland, 2005.

[10] Clearware.org, http://www.clearware.org, 2006-05-31

[11] L. F. Cranor, "Giving notice: why privacy policies and security breach notifications aren't enough", in *IEEE Communications Magazine*, Vol. 43, Issue 8, 2005.

[12] L. F. Cranor, "P3P: Making Privacy Policies More Useful", in the *IEEE Security & Privacy*, Volume 1, Issue 6, 2003.

[13] L. F. Cranor, *"Security and Usability"*, O'Reilly, Sebastopol, 2005.

[14] Earthlink Spy Audit, http://www.earthlink.net/spyaudit/press/, 2006-05-31.

[15] FreeBSD Ports, http://www.freebsd.org/ports/, 2006-05-31.

[16] B. Friedman, E. Felten, and L. I. Millett, "Informed Consent Online: A Conceptual Model and Design Principles", *CSE Technical Report*, University of Washington, 2000.

[17] N. Good, et al., "Stopping Spyware at the Gate: A User Study of Privacy, Notice and Spyware", in the *Proceedings of the Symposium on Usable Privacy and Security,* Pittsburgh USA, 2005.

[18] S. Görling, "An Introduction to the Parasite Economy", in EICAR 2004, Luxemburg, 2004.

[19] A. Jacobsson, M. Boldt, and B. Carlsson, "Privacy-Invasive Software in File-Sharing Tools", in *Proceedings of the 18th IFIP World Computer Congress,* Toulouse France, 2004.

[20] Lavasoft, http://www.lavasoftusa.com, 2006-05-31.

[21] D. M. Martin Jr, R. M. Smith, M. Brittain, I. F. Fetch, and H. Wu, "The privacy practices of Web browser extensions", in *Communications of the ACM*, Volume 44, Issue 2, 2001.

[22] A. Moshchuk, T. Bragin, S. D. Gribble, and H. M. Levy, "A Crawler-based Study of Spyware on the Web", in the *Proceedings of the 13th Annual Network and Distributed System Security Symposium (NDSS 2006)*, San Diego CA, 2006.

[23] Proxy Appliances Blue Coat Systems, Inc., http://www.bluecoat.com, 2006-05-31.

[24] B. Schneier, "Inside risks: semantic network attacks", in *Communications of the ACM*, Volume 43, Issue 12, 2000.

[25] C. Shapiro and H. R. Varian, *"Information Rules - A Strategic Guide to the Network Economy"*, Harvard Business School Press, Boston Massachusetts, 1999.

[26] S. Shukla and F. F. Nah, "Web Browsing and Spyware Intrusion", in *Communications of the ACM*, Volume 48, Issue 8, 2005.

[27] J. C. Sipior, "A United States Perspective on the Ethical and Legal Issues of Spyware", in *Proceedings of Seventh International Conference on Electronic Commerce*, Xi'an China, 2005.

[28] E. Skoudis, *"Malware - Fighting Malicious Code"*, Prentice Hall PTR, Upper Saddle River NJ, 2004.

[29] Spyware Developers Net Huge Profits, Outrage - With Annual Revenues of $2 Billion, Pop-up Ads are a High-Stakes Game, http://www.msnbc.msn.com/id/13757388/, 2006-07-15.

[30] "Spyware": Research, Testing, Legislation, and Suits, http://www.benedelman.org/spyware/, 2006-03-10.

[31] S. Steinbrecher, "Design Options for Privacy-Respecting Reputation Systems within Centralised Internet Communities", in *Proceedings of the 21st IFIP SEC 2006*, Karlstad Sweden, 2006.

[32] StopBadware.org, http://www.stopbadware.org, 2006-05-31.

[33] TRUSTe - The Trusted Download Program (Beta), http://www.truste.org/trusteddownload.php, 2006-05-31.

[34] M. Warkentin, et. al, "A Framework For Spyware Assessment", in *Communications of the ACM*, Volume 48, Issue 8, 2005.

[35] Webroot Software, *"State of Spyware - Q3 2005*", http://www.webroot.com/resources/, 2006-05-31.