Usability meets Security – The Identity-Manager as your Personal Security Assistant for the Internet

Uwe Jendricke and Daniela Gerd tom Markotten
Albert-Ludwigs-University of Freiburg
Institute for Computer Science and Social Studies
Department of Telematics
Friedrichstrasse 50, D-79098 Freiburg, Germany
{ujendric, dany}@iig.uni-freiburg.de

Abstract

In today's applications, most users disregard the security functionality. They do not have the knowledge and/or the motivation to configure or to use the existing security functions correctly. In this paper, we present a new concept to improve the usability of security mechanisms, introducing an extended classification of protection goals. As a result, the everyday use of security functionality can be reduced to selecting the user's identity, which is the basis of the Identity-Manager, *a new security tool presented in this paper. It offers a user interface for security functionality compatible to all internet applications. So even the inexperienced users are able to configure and negotiate their security needs in a convenient way.*

1. Introduction

The results from [12] and our own research show that wrong system configuration by the user is one of the main reasons why effective computer security often fails. Based on our experiences with former projects [10], we started to examine why many users are not able to configure their security settings by themselves. As an example of internet based applications, we studied the security functions of the Netscape Navigator, the Internet Explorer, and PGP relating to usability aspects.

To investigate further, we regarded both the everyday interaction between the security system and the user, and the existing solutions for setting security policies. In our definition, a *security policy* is a group of all system settings that configure the security-relevant functions (e.g. how to handle cookies or which crypto algorithms are used).

Configuring security is usually not a primary activity for computer users. "People do not generally sit down at their computers wanting to manage their security ... they want security in place to protect them..." [12]. Thus, security configuration work should preferably be avoided during the user's everyday activities. This can be realized by a mostly all-inclusive configuration once during the installation of the system. This has the advantage that the users are only concerned once with the unpopular security configuration. To prevent the user from opening security leaks, a strictly guided and user-friendly installation procedure is necessary. The users have to be informed in a comprehensible way of the risks they are automatically protected against. Perhaps the assistance of experts can be helpful for this one-time configuration. Generally, the aim of our concept is to bother the user as little as possible with security matters.

In section 2, the usability of today's security user interfaces is examined and deficiencies are highlighted. In section 3, we present the new concept for security configuration, which allows a simplified everyday configuration for the user. The Identity-Manager is the application that is based on these results. It is presented in section 4. In section 5, an outlook on our further research is given.

2. Usability of Today's Security User Interfaces

With the increasing awareness of security risks, more and more software applications are used, where security functions have to be configured. The main reason for the failure of secure systems seems to be not the malfunction of these systems but the wrong configuration and use. This leads to the conclusion that the existing user interfaces are not appropriate for most users. It can be seen that some basic rules for user interface design were disregarded in most cases. As early as 1975, the psychological acceptability was discussed by [11]. The user's mental image of the protection goals should match the used mechanisms. In the existing user interfaces, the mental image of developers seems

1063-9527/00 $10.00 © 2000 IEEE

to match, but not the mental image of most of the standard users. This is demonstrated in the following sections.

2.1. Configuration

Today's applications offer very different user interfaces for configuring and altering the security policy. Most existing interfaces fall into two categories, according to the configuration of the security policy. As an example, we show the MS Internet Explorer 5 which offers both types of configuration. The first category just shows a few buttons or a slider to set the security level to 'low', 'medium', and 'high', shown in figure 1. We call this the 'low-medium-high' approach. This approach has several disadvantages. If the user selects 'medium security', the IE proposes 'Safe browsing and still functional'. However, some unsafe mechanisms are still active, e.g. cookies are accepted without prompt. Furthermore, the user is not informed about possible disadvantages of higher security levels in a comprehensible way. In addition, security configuration requires a higher granularity to fulfill the user's needs in different situations, which makes IE with its four situations (internet, local intranet, trusted sites, restricted sites) insufficient.

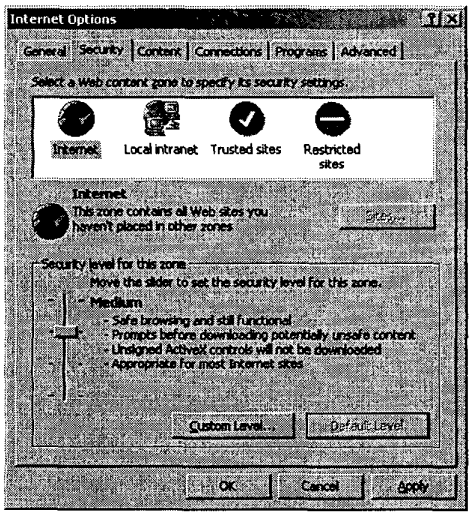

Figure 1. 'low-medium-high' settings of IE

The other category of security configuration offers the possibility of a customized and detailed setting of security mechanisms (figure 2). The correct setting takes a lot of time and often requires (expert-)knowledge. Most users are not motivated to invest the amount of time required for the configuration. In addition, many users do not have the knowledge to understand the consequences of their configuration activities. There is a high risk of tearing open security

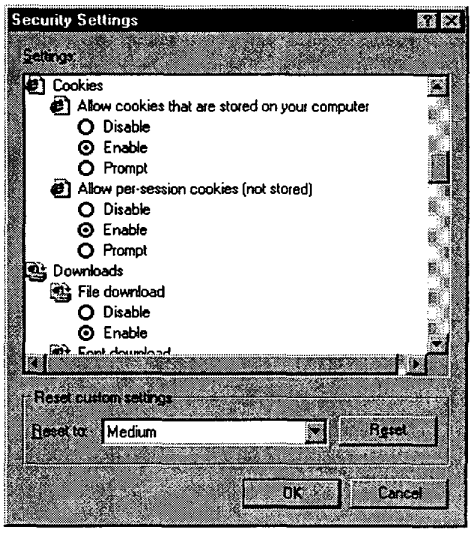

Figure 2. Custom security settings of IE

leaks by accidentally using the wrong settings. The usability tests of [12] and [6] show that some users believed to communicate in a secure way even though their communication was not secure at all.

Summarizing, the settings are either too simple and do not offer the necessary granularity, or they are too hard and too time consuming to set.

2.2. Warnings and Information Windows

If a violation of the actual security policy happens, most applications show a warning window (see figure 3). The text in the window explains the security problem and asks the user if the critical action should be executed. In most cases this text is not appropriate for all users. Non-experts often do not understand the security problem that is explained and just click 'ok' because this is the only way to perform the desired action. By clicking 'ok', an insecure action will be allowed (and performed) and the user does not know exactly what happens. Neither Netscape nor IE offer suggestions for a secure alternative.

In addition, warnings appear when the system offers a higher security level than the level defined in the actual security policy. At first sight, the user does not know if the pop-up window is a severe warning or just an information about the security policy. These warning windows should be avoided, especially for non-experienced users.

With a very detailed security configuration it should be possible to prevent those warnings. However, these detailed settings implicate the problem of setting them correctly, as shown before with the Internet Explorer.

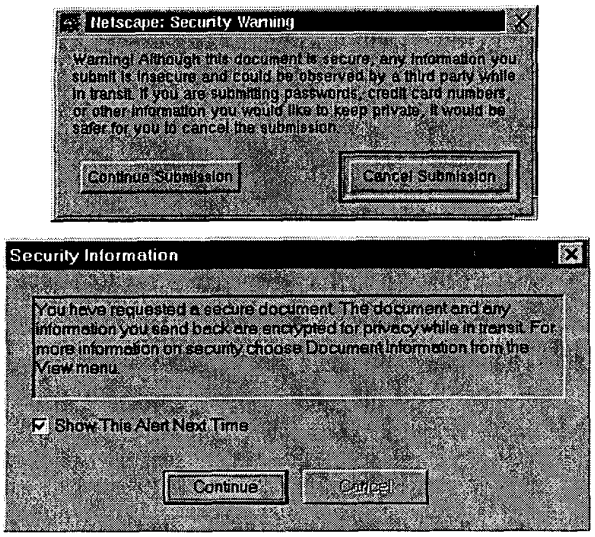

Figure 3. Each warning window points out an inaccurate security configuration

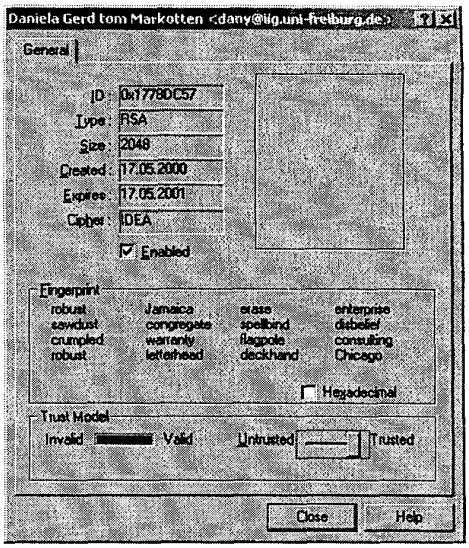

Figure 4. PGP keys: the user needs to set the trust model for the key

2.3. Everyday Use

In everyday use, most applications with security functionality demand higher knowledge about security issues from the user. Most users do not have that knowledge and this is one of the reasons for their failure when using security mechanisms. One example is the PGP software. When using pgp-keys, users need to have knowledge about trust models as shown in figure 4. In [12], some more problems are described.

A more user-friendly example is the command line version of secure shell SSH. Using SSH instead of the insecure telnet makes hardly any difference to the user. Security details that may only be used correctly by experts (e.g. key exchange) are hidden from the user without a reduction of the software functionality.

3. The User – an Accountable Identity?

As shown in section 2, the security settings of most applications are inappropriate. Using security functionality implies a high amount of complexity, e.g. handling keys, signatures or random numbers. A mechanism that sits in-between this complexity and a non-complex user interface needs to be found. Generally, a user-centered security [14] has to be realized: usability should be one of the primary goals in software design.

We divide the security relevant mechanisms into two parts. One part is completely system-controlled without user interaction[1] during everyday use. The other part consists of the few remaining functions where the users are empowered (and required) to set their individual security policy.

Protection of Protection goals	Content	Circumstances
Confidentiality Goals	Confidentiality Hiding	Unobservability Anonymity
Integrity Goals	Integrity	Accountability

Table 1. Different aims of protection goals

3.1. The Attacker Model

During the examination of existing definitions of protection goals [4, 13], we found an inadequate differentiation of user perspectives.

In these attacker models, protection goals are divided into two groups, *content* and *circumstances* of the communication, referring to their aim of protection (see table 1, taken from [13]). Based on this differentiation, they use two different user perspectives. For protection goals concerning the content, the communication partners protect themselves together against third parties[2]. In the second case, protecting the communication circumstances, the user wants to

[1] However, experts must have the opportunity to gain control over all functionality.

[2] The third party consists of all parties not participating in the communication, i.e. the "rest of the world".

Unobservability	Definition
CC 2.1	ensures that a user may use a resource or service without others, especially third parties, being able to observe that the resource or service is being used.
I. against all	ensures that a user may use a resource or service without *communication partners and others* being able to observe that the resource or service is being used.
II. against 3^{rd} parties	ensures that a user may use a resource or service without *third parties* being able to observe that the resource or service is being used.
Anonymity	
CC 2.1	ensures that a user may use a resource or service without disclosing the user's identity.
I. against all	ensures that a user may use a resource or service without disclosing the user's identity to *anybody else*.
II. against 3^{rd} parties	ensures that a user may use a resource or service without disclosing the user's identity to *third parties* but possibly to the communication partners.

Table 2. Definitions of protection goals

protect himself against everybody else.

We introduce a further differentiation for the protection goals concerning the circumstances of the communication. The confidentiality goals *anonymity* and *unobservability* cannot merely be understood as protection goals to secure the user from the rest of the world. In some situations, the user just needs anonymity and unobservability against third parties and not against his communication partners.

Based on our differentiation of the user perspectives, the existing definitions of protection goals should be divided into two new groups:

Group I: protection against everybody else (including communication partners).

Group II: protection against third parties.

In table 2, we show our definitions of the protection goals unobservability and anonymity in relation to the definitions of the Common Criteria [4].

3.2. Implications for Protection Goals

Based on the definitions of section 3.1, it is now possible to set up a system of implications between the protection goals (see figure 5). This forms the theoretical background of the simplifications and the Identity-Manager that is presented in the next sections.

If a protection goal x implies another protection goal y and x is guaranteed, then y is guaranteed, too. Implications are transitive. The implications for unobservability II $\Rightarrow$ hiding $\Rightarrow$ confidentiality are described in [13]. The other implications can be proven by the following mathematical model.

We only regard point-to-point communications, as point-to-multipoint can be expressed by several point-to-point communications. When a communication has taken place,

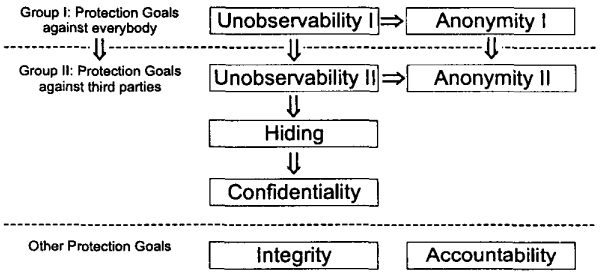

Figure 5. Implications

we consider the knowledge of all parties about that communication. For the mathematical description, here are first some definitions. Data that is sent through the network belongs to a communication c_i. Communications are numbered sequentially, so the communication c_{i+1} happens later than c_i. During each communication, data is transmitted. This data consists of three parts: the address arrays of the two communication partners and the reference data.

$c_i = (a_i, b_i, d_i)$ communication i
a_i address array of partner A of communication c_i
b_i address array of partner B of communication c_i
d_i transmitted reference data of communication c_i

The knowledge of each party about the contents and the circumstances of the communication is represented by sets. An element in the set represents that the party knows that element.

$A = \{a_i, b_i, d_i \mid 0 < i < \infty\}$	data known by partner A
$B = \{a_i, b_i, d_i \mid 0 < i < \infty\}$	data known by partner B
$TP = \{a_i, b_i, d_i \mid 0 < i < \infty\}$	data known by third parties

In addition, we need the following definitions:

m_i	message (without information about a_i or b_i)
$e(m_i)$	encrypted message
$d_i \in \{m_i; e(m_i)\}$	transmitted reference data: message or encrypted message

Now, the knowledge about a communication that took place can be expressed. As an example, consider a communication c_i between the partners A and B. This communication can be expressed by $c_i = (a_i, b_i, d_i)$. The receiver B knows the communication contents, so $d_i \in B$. If the communication contents are encrypted ($d_i = e(m_i)$), third parties are still able to get the transmitted data d_i, but because of the encryption they do not know the message content that was transmitted: $m_i \notin TP$. In spite of the encryption, third parties may still observe the circumstances of the communication: $a_i, b_i \in TP$. If a message was sent anonymously, the receiver and third parties have no possibility of obtaining any information about the sender's identity (and the contents of the address array of the sender) of a message: $a_i \notin B; a_i \notin TP$. Here, we do not consider the encryption of address arrays. If somebody knows the contents of the address array, he knows the addressed partner.

The process of acquiring knowledge is represented by sets with an increasing number of elements. If the third party does not observe a communication c_i, then $a_i, b_i, d_i \notin TP$. If the third party observes the communication c_i, it acquires knowledge about the communication circumstances and contents: $a_i, b_i, d_i \in TP$. If the third party has received this information once, this cannot be revoked. This is represented in our model by the fact that the number of elements in a set cannot decrease.

After a communication c_i has taken place, the protection goals unobservability (I and II) and anonymity (I and II) can be formalized and the vertical implications from group I to group II and the horizontal implications of figure 5 can be concluded:

$$\begin{array}{ccc} \text{Unobservability I} & & \text{Anonymity I} \\ a_i, b_i, d_i \notin B \wedge a_i, b_i, d_i \notin TP & \Rightarrow & a_i \notin B \wedge a_i \notin TP \\ \Downarrow & & \Downarrow \\ \text{Unobservability II} & & \text{Anonymity II} \\ a_i, b_i, d_i \notin TP & \Rightarrow & a_i \notin TP \end{array}$$

Unobservability implies anonymity: If nobody is able to observe the actor while using a resource or service, then no information about the actor's identity can be extracted. The sender is implicitly anonymous. In addition, there are implications from group I to group II (everybody else to third parties), which indicate that a protection goal against everybody else implies the same protection goal just against third parties. The third parties are a subset of everybody else, so the implication is obvious.

In addition to the protection goals linked by implications there are two more protection goals without implications: integrity and accountability. These are discussed in detail in section 3.3 and section 3.4.

3.3. System-controlled Protection Goals

For our new simplified user interface we need a reduction of complexity between the security functionality and the user interface. The results of section 3.1 lead to a new fundamental simplification of the user interface because some protection goals can be completely system-controlled without user interaction during everyday use.

Protection Goals Against Third Parties. As a result of our system of protection goals and its implications, we claim that all protection goals against third parties can be enforced almost anytime. In order to achieve this, it suffices to enforce unobservability II (figure 5). There is no imaginable case where the user needs to be observable by third parties: unknown and unauthorized actors never have to be informed about a communication. The technical realization of unobservability II is possible with reasonable effort in most cases. An overview of available technologies can be found in [2, 3].

Our concept guarantees a simplified everyday configuration and use with the trade-off of an extended installation procedure. It has the advantage for the non-experienced user that he does not have to configure the group II of protection goals anymore during everyday use.

The remaining protection goals are integrity, accountability, and those of group I.

Integrity. Integrity ensures that modifications of the communicated contents are detectable by the communication partners. In telecommunication systems, transmission errors and manipulation of transmitted data can be easily detected, e.g. by cyclic redundancy check (CRC). The detection and correction of integrity violations can be achieved transparently without bothering the user. For instance, this is realized during an ftp connection by the security layer in the TCP/IP stack. So the integrity needs no configuration by the user.

Unobservability I. The ideal case for the user would be the continuous enforcement of the protection goals against everybody else, which means that only the continuous enforcement of unobservability I would be needed. However, there are some arguments against unobservability I. First, the technical realization is extremely difficult, expensive, and in some cases impossible. Secondly, there are hardly any situations where unobservability I is required and furthermore there are situations where unobservability I is not allowed (e.g. for billing, preventing fraud, etc.) or not desired. For these reasons we do not consider this protection

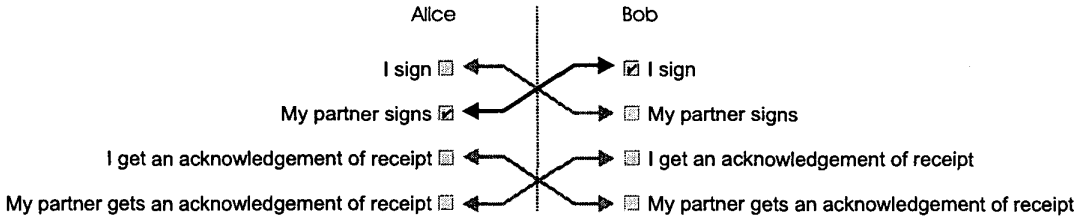

Figure 6. Relations of accountability between communication partners Alice and Bob, Alice demands a signed document from Bob.

goal in our concept and suggest special applications for the cases where unobservability I is needed [12].

3.4. User-controlled Protection Goals: the User's Identity

The remaining protection goals anonymity I and accountability have to be handled by the user because their configuration depends on the situation and the user's attitude to his personal security needs. Our investigations led us to an interesting result: the two remaining protection goals can be reduced to the user's *identity*, which in this case is the user's appearance in the network. Each user has a lot of identities of different kinds, e.g. IP-addresses, e-mail addresses, nicknames, credit card numbers, real names, postal addresses, etc. A survey on personal data and its relevance to the user is given in [5]. Generally, the difficulty in relating the identities to the user himself varies by degrees. Some identities can be easily linked to the user, e.g. a certified digital signature, while others are harder to relate, e.g. IRC nicknames.

The identity determines the degree of linkability of the user's actions. A constantly changing identity (e.g. one-time pseudonyms) reduces the possibility to link actions of a user. However, if this user always uses the same IP-address, others have the ability to link his actions.

In most cases, users want to disclose as little of their identity [5] as possible. On the other hand, the communication partners often want to get as much information as possible about their partners. In the worst case, this problem has to be negotiated individually for each situation. Several strategies are possible:

- The type of identity is negotiated, e.g. 'if you tell me your address, I'll tell you mine'. It is possible that one party (e.g. an online-shop) offers a rebate if the other one exposes more details of its identity. This model is already used, e.g. with client-cards. However, negotiation with an automatic shop system is limited.
- A system (e.g. a shopping system) offers several identities the user can select. If the user selects the more detailed identity, he may get a rebate.
- The user selects the identity out of a set of predefined identities that might be described in common speech or with a few icons.
- The user builds the identity with a toolbox if a new identity is needed. This procedure must be quick and easy because users may create a lot of identities. Zero-Knowledge Inc. already provides a service where this is used, called 'Freedom'[3].

Accountability. As mentioned earlier, each point-to-multipoint communication can be expressed by several point-to-point communications. So, accountability in multipoint communications can be reduced in the same way too. Accountability has different perspectives for the sender and for the recipient. In addition, accountability has to be differentiated for the actions *sending* and *receiving*. Therefore, accountability must be divided into four cases which must be configurable by the user, e.g. by checkboxes. This configuration is not independent from the configuration of the communication partner. If Alice demands a signed document from her partner Bob, this demand must become visible in Bob's user interface. Bob is asked if he wants to sign the document. This is illustrated in figure 6.

Some of these settings need to be negotiated. It cannot be allowed for Alice to configure Bob's system to send an acknowledgement of receipt. Alice must have the opportunity of asking Bob to configure the system to generate an acknowledgement of receipt for Alice. The same problem must be solved when Alice asks Bob to sign a document. Alice is not allowed to decide whether Bob signs a document or not.

4. The Identity-Manager

To realize a secure but easy-to-use system in the way described in section 3 and to get one single user interface

[3] http://www.freedom.net/

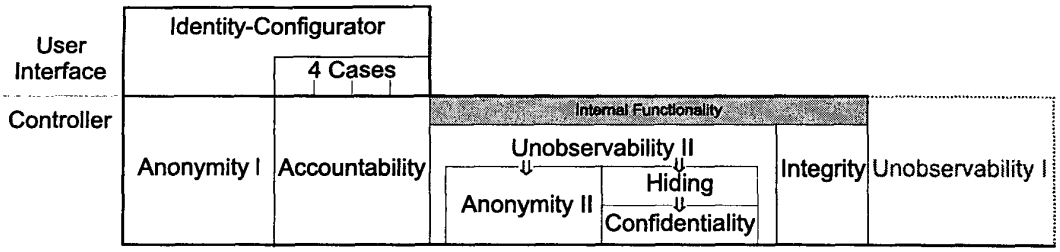

Figure 7. Architecture of the iManager

for security, we have designed an application that is called *Identity-Manager* (iManager). It exists in-between the internet applications and the network and helps the users to manage their identities. Therefore, it controls the data flows to and from the network and offers a user-friendly interface for the required security functionality.

4.1. The Architecture of the iManager

With the results of section 3, we developed an architecture for the iManager, which is shown in figure 7. The architecture consists of two layers: the controller layer and the user interface.

Some of the protection goals can be handled completely by the controller layer with the internal functionality that controls their enforcement without user interaction. The other protection goals still need the attention of the user. However, their use and configuration can be simplified. Anonymity I and accountability can be described with the identity of the user (see section 3.4) which is selected for each situation. Accountability has to be set by the user, as described in section 3.4. Unobservability I is not considered further, as described in section 3.3.

As far as we know, nothing comparable to this architecture is realized in any operating system nor in any of the existing internet applications, such as Netscape Communicator, Internet Explorer, or Eudora. In addition, these applications show very different user interfaces and configuration possibilities for security functionality. With the iManager, all applications can be used with only one security user interface. This eases the handling of the security functionality.

Some applications, such as Norton Internet Security 2000 or Pegasus Mail, or internet services, such as Freedom or digitalme[4] have a few similarities with our concept, but all of them are very limited to special applications or services and do not cover the whole problem.

[4]http://www.digitalme.com/

4.2. Details of the Identity-Manager

The iManager operates like a firewall in-between the network and the applications[5]. This proxy-like architecture offers several advantages. The iManager is compatible with all existing applications, which is a very important factor because the existing applications e.g. for e-mail or web access need not be adjusted. Just one iManager is needed for all applications. The structure of the iManager and its integration is shown in figure 8.

The iManager consists of several modules. Databases are used to store the data of the user and network specific data. Application modules for encryption, digital signatures, or mix network access are realized as plug-ins. Finally, the iManager has to communicate with the applications, the network, and the user, so it needs system interfaces for the applications and to the sockets of the TCP/IP network and a user interface.

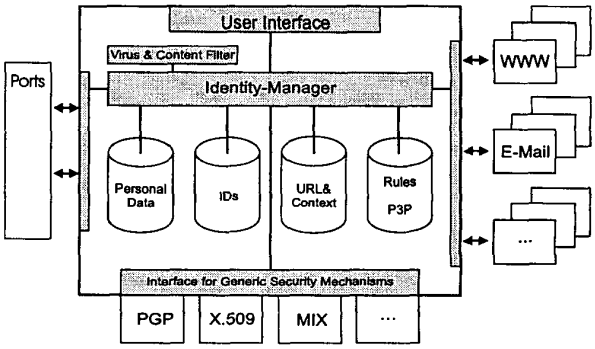

Figure 8. Structure of the iManager

Databases. There are four databases for the storage of data: personal data, identities, URLs, and rules. The databases are used for the creation, management, and the use of identities.

[5]In future, parts of the iManager should be integrated into the operating system.

Personal Data: The system controls the access to the personal data of the user. For this, the user has to enter his personal data only once, such as name, address, telephone numbers, e-mail address, credit card number, bank information, and the private key. This is a critical database that has to be protected accordingly.

Identities: The user may operate in the network with a lot of different identities (IDs). IDs, as described in section 3.4, consist of parts of the personal data of the user, combined with additional information such as nicknames. For the generation of IDs, access to the personal data is needed. A set of predefined IDs is imaginable, perhaps certified by experts.

URL: The system stores each URL[6] the user has visited or communicated with. For each URL, an ID is stored, which may be a predefined default ID or an ID that is selected by the user. By URL-tracking, the same ID can be used as default, when the user returns to an address. The advantage here is that no more security configuration activities are required when a URL is visited several times. However, contents or owner of an URL might change, so the user must have the ability to change the ID. Perhaps it is useful to give these URL-ID pairs a limited lifetime that may even be user-defined. Collections of URL-ID pairs may be distributed (e.g. company-wide) and certified by experts.

Rules: Another tool for the simplification of the ID-configuration is the use of rules, which can be used to describe a fundamental policy. It may be defined, for example, that no telephone numbers are transmitted to .com-domains. With a set of such rules, the ID configuration can be simplified enormously. However, an incorrectly defined rule may cause a lot of damage, too. Therefore, rules have to be used with care. In addition to the rules, support of the P3P functionality [1] is planned. The user can define P3P preferences that are used by the iManager.

System Interfaces. The iManager needs access to several algorithms for security functionality. In addition, it must be extendable for new mechanisms, which results in an integrated plug-in system. Two different types of modules can be distinguished. The iManager consists of a number of modules to realize the internal functionality for the group of protection goals that need no everyday configuration by the users. Other modules are controlled by the user interface for identity configuration and usage. In addition, the iManager has interfaces for connections to the TCP sockets and it offers sockets for the applications to connect to.

[6]Uniform Resource Locator, including e-mail-addresses, irc, news,...

Filter Functionality. Like a firewall, the iManager is able to scan all incoming and outgoing data. This enables the system to prevent unwanted publication of personal data.

The incoming data can be scanned and changed. The iManager has the ability to recognize code and to insert the needed data based on the actual identity. This feature needs a wrapper [7] that extracts the information (e.g. HTML forms) out of the web page. For example, it may fill in the name and address of the user's identity directly in the HTML/XML-code of a dialog-field. This functionality may produce errors, if the information is misinterpreted by the wrapper. Therefore, the user has the ability to verify the information in the browser window. In the incoming data filter, other functionalities, such as virus check, may also be integrated.

More important is the scanning of the outgoing data, because reclaiming data once sent through the internet is very difficult [8]. The system can delete personal data and/or warn the user if data was entered that is not allowed by the actual identity settings and the P3P policy. So the system acts like a firewall and it prevents security violations in the outgoing data.

4.3. The User Interface

The user interface is an essential part of the iManager because the acceptance of the whole system strongly depends on its user-friendliness [9]. Two different kinds of user interaction exist: a strictly guided configuration of the system during installation [12] and the everyday use. The user must be allowed to change and to browse the initial configuration at any time.

In everyday use, the user must select identities or generate new ones. Additionally, the iManager can offer a small status-window where the user has the ability to watch all activities. The identity-window always shows the actual identity, which changes when the application is switched or a new URL is used.

Displaying Identities. The identity must be displayed in a user-friendly way. The first idea was the representation on a slider with anonymous identities on the left and accountable identities on the right. This would grant a fast and easy setting of the identity required. However, suppose a linear ranking of the identities e.g. from anonymous to less anonymous – this ranking is subjective. As an example, consider one identity that consists of the user name and the postal address, the other consists of the name and the telephone number. It depends on the user, which identity is more critical for him to publish, so a general ranking is impossible.

So far, a representation with checkboxes seems to be the best approach for identity configuration. A prototype of this interface is shown in figure 9.

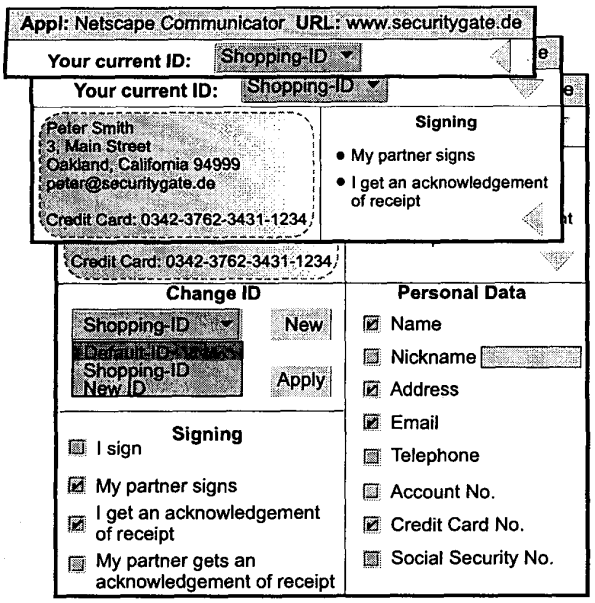

Figure 9. User Interface Prototype of the iManager

During everyday use the interface just shows the active application, the actual URL, and the actually selected identity (upper window). The user has the opportunity to view more details at any time: the user just clicks the triangle on the right and the window expands (middle window). The expanded part of the window shows the contents of the selected identity. The personal data is shown as on a business card on the left and on the right part the user obtains the information as to who signs the document and sends acknowledgements. The user has the opportunity to expand the window again by clicking the new triangle. The window then expands again and the user can change the identity or to define new ones. This is shown in the lowest window of figure 9.

4.4. Perspectives and Limits of the iManager

The iManager sits in-between the applications and the network. This has the advantage that all existing standard internet applications can be used. However, this concept also has a few disadvantages. The iManager cannot control the access of several users to their e-mail archives on the local machine. All the local security issues are not in the scope of the iManager. However, this should be the duty of a secure OS. An unsolved problem is the use of secure http (https) – here the iManager must have the ability to take over the encryption.

In its position between network and applications, the iManager offers a lot of additional possibilities. The iManager scans all incoming and outgoing data, which allows content filtering and the scanning for viruses. An extraordinary behavior of the internet applications such as network traffic of mass mailing caused by virus or trojan horses, can also be detected.

5. Summary and Outlook

Today's user interfaces for security are insufficient. This problem, in combination with a non-awareness of the security problems by most users, is a severe hindrance in the world of internet and e-commerce. New ideas and concepts are needed to solve this problem.

With the concept of reducing the everyday configuration effort for the users, we may have found a solution to this problem. The reduction of the remaining configurable items to the user's identity offers the user a single and understandable object to handle. There is no more confrontation with abstract terms that are mostly unknown to the user. In addition, the user does not need a lot of different security configuration tools. With the Identity-Manager we offer one comprehensible interface for all internet applications.

However, there are some unsolved problems. Right now, the system is not evaluated in user tests. Furthermore, the system needs a lot of mechanisms to realize the full functionality, e.g. a mix system or something comparable, but our research project is integrated in a joint research project[7] where a complete security architecture will be developed that performs the needed functionality.

References

[1] The Platform for Privacy Preferences 1.0 (P3P1.0) Specification, May 2000.

[2] O. Berthold, H. Federrath, and M. Köhntopp. Project 'Anonymity and Unobservability in the Internet'. In *Workshop on Freedom and Privacy by Design / Conference on Freedom and Privacy 2000*, pages 57–65, Toronto/Canada, April 2000.

[3] D. L. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24, 2:84–88, 1981.

[4] Common Criteria for Information Technology Security Evaluation, Part 2: Security functional requirements, August 1999. Version 2.1/ISO IS 15408.

[5] L. F. Cranor, J. Reagle, and M. S. Ackerman. Beyond Concern: Understanding Net Users' Attitudes About Online Privacy. Technical report, AT&T Labs Research, April 1999.

[6] V.-P. Kröger. Security of User Interfaces – A Usability Evaluation of F-Secure SSH, December 1999.

[7] The project is financed by the DFG, the central public funding organization for academic research in Germany. Project pages http://www.iig.uni-freiburg.de/telematik/spps/

[7] N. Kushmerick, D. S. Weld, and R. Doorenbos. Wrapper Induction for Information Extraction. In *Proceedings of IJCAI-97 (Nagoya)*, pages 729–735, August 1997.

[8] M. McGinity. Surfing your Turf. *Communications of the ACM*, 43(4):19–21, April 2000.

[9] J. Nielsen. *Usability Engineering*. Academic Press Inc, 1994.

[10] M. Reichenbach, H. Damker, H. Federrath, and K. Rannenberg. Individual Management of Personal Reachability in Mobile Communication. In L. Yngström and J. Carlsen, editors, *Information Security in Research and Business; Proceedings of the IFIP TC11 13th International Information Security Conference (SEC '97): 14-16 May 1997, Copenhagen, Denmark*, pages 163–174, London, 1997. Chapman & Hall.

[11] J. H. Saltzer and M. D. Schroeder. The Protection of Information in Computer Systems. In *Proceedings of the IEEE*, volume 63, pages 1278–1308, September 1975.

[12] A. Whitten and J. Tygar. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *Proceedings of the 8th USENIX Security Symposium*, August 1999.

[13] G. Wolf and A. Pfitzmann. Empowering Users to Set Their Security Goals. In G. Müller and K. Rannenberg, editors, *Technology, Infrastructure, Economy*, volume 3 of *Multilateral Security in Communications*, pages 113–135. Addison Wesley Longman Verlag GmbH, 1999.

[14] M. E. Zurko and R. T. Simon. User-centered security. In *Proceedings of the UCLA conference on new security paradigms workshops*, pages 27–33, Lake Arrowhead, CA USA, September 1996.