Security issues for pervasive personalized communication systems

Bertin Klein, Tristan Miller, and Sandra Zilles

DFKI GmbH, Erwin-Schrödinger-Straße 57, 67663 Kaiserslautern, Germany
e-mail: {Bertin.Klein, Tristan.Miller, Sandra.Zilles}@dfki.de

Abstract. Technological progress allows us to equip any mobile phone with new functionalities, such as storing personalized information about its owner and using the corresponding personal profile for enabling communication to persons whose mobile phones represent similar profiles. However, this raises very specific security issues, in particular relating to the use of Bluetooth technology. Herein we consider such scenarios and related problems in privacy and security matters. We analyze in which respect certain design approaches may fail or succeed at solving these problems. We concentrate on methods for designing the user-related part of the communication service appropriately in order to enhance confidentiality.

1 Introduction

Ubiquitous or pervasive computing is so commonplace today that few are aware of its origins—namely, Weiser and his seminal work [11]. Discount chains like Aldi sell ever more portable and smaller computers, which will soon be nothing but "mobile phone computers". Eventually everybody will carry their computer around in their pocket and use it also to do phone calls. Recently, we envisioned that the content of people's writing desks will also be transferred to the virtual world and manipulated with such devices [6]. Already today has much of this vision been realized.

We are currently designing prototypes for TINKERBELL, a social networking application run on Bluetooth-enabled mobile phone computers (also known as mobile digital assistants, or MDAs). The original idea behind TINKERBELL was to obviate the need of some pretext for initiating and enjoying contact with other people. However, we soon discovered that, through the incorporation of knowledge-based technology, the potential applications for such a system would be legion. By intelligently and efficiently observing, managing, and sharing information about their owners, TINKERBELLs could become something like mirrors of or proxies for their owners. It is obvious, then, that privacy and security concerns will play an important role in the development of such systems.

This paper introduces TINKERBELL's communication model and discusses the privacy and security issues arising therefrom. Researchers and developers have long recognized privacy and security as central concerns in the development and

deployment of ubiquitous systems [10, 8]. We believe that these concerns will be at the forefront of consumers' purchasing decisions once they are in a position to choose between competing systems. Seemingly small details could then tip the scales and have great consequences.

2 Tinkerbell's communication model

Since people socialize based mainly on their similarities to one another—i.e., shared properties or shared opinions—a basic TINKERBELL can be rather simple. It need only host some information that is representative of its owner and check with other TINKERBELLs for similarities with the information about their owners. Our current prototypes encode and exchange personal information using an information scheme (an ontology) which represents concepts like the town where one lives and hobby descriptions. However, TINKERBELL also makes use of information from other users in the vicinity, creating an interesting flavor of context and locality.

The information that a TINKERBELL carries is initially a simple set of preconfigured information items. However, it is constantly provided with new data from its "small-talk" with other TINKERBELLs or from interactions with its owner—perhaps in a Tamagotchi-game-like manner, or by new addressbook entries, or by learning semantic fingerprints of recent papers read by its owner, or perhaps by qualitative feedback about the matches it has made. And, knowing the current state of knowledge technology, it is not so far-fetched to conceive of TINKERBELL as becoming more and more a personalized counterpart to its owner (though certainly not further than the owner wishes).

In general TINKERBELL is designed to require neither central services nor prior knowledge of access/address information. (This is probably why we call it pervasive.) TINKERBELLs which come into proximity with each other will automatically swap information (some of which may be encoded with one-way hashing functions). When a TINKERBELL learns that its owner has a certain number of contacts, hobbies, or other characteristics in common with someone else, it will immediately notify its owner. The owner can then take the appropriate action, which might range from a simple hello to engaging in personal or business-related dialogue with the second party.

With this architecture, TINKERBELL is near being able to offer to its owner all the six categories for pervasive context-aware services [1]:

1. proactive triggering of events (if it records past events)
2. streamlining interaction (because it is involved in its owners interaction)
3. memory for past events (it can at least record its interactions)
4. reminders for future events (this is already a feature of current MDAs)
5. optimizing patterns of behavior (e.g., when passers-by are identified as not interesting)
6. sharing experience (that is one of the ideas behind the notion of small-talk)

3 Security and privacy issues

For applications like TINKERBELL, in order to survive on a market or in a society, it is crucial to invest in some thought on system vulnerability and possible countermeasures to misuse or attack. Of course no system can be "secure" in an absolute sense; it can be secure only to some degree or level of confidence. However, human culture is by now, for the most part, well acquainted with this state of affairs. Security is rightly seen as a trade-off between risk and reward [10].

Security has a highly social dimension. That is, we should always be cognizant of the sometimes-overlooked fact that the security of any system involving humans crucially depends on human behaviour. Consequently our philosophy of security and privacy for TINKERBELL is based on how to minimize the security risks associated with human oversight. Our first task is therefore to systematically generate a plan to identify such risks. The last step to be taken is to ensure that these risks are addressed with the appropriate technology—technology which people will actually use. Unfortunately, however, establishing the security of a system is almost impossible to fully accomplish during the design stages. Absence of security in unlikely places is often discovered only later, after people have already experienced losses.

According to Robinson & Beigl [9], most security issues in pervasive computing arise from the difficulty of coordinating the symbiosis of physical and virtual entities. We can therefore identify two main sources of security issues of pervasive applications like TINKERBELL:

1. **Transfer of real to virtual.** The transfer of processes from the real to the virtual blurs human understanding and attention. Security issues arise where new technologies create new virtual spaces which are not covered by state-of-the-art security procedures and conventions.
2. **Access of computational power to data.** When information is stored on electronic devices and is accessible to computational processes, there are new possibilities created to use the data, and thus also to do security-compromising things.

Note that when considering an application like TINKERBELL which is thought of as a service to be sold to private users, it is important not only to provide actual security, but also to provide the user with a *feeling* of security. The user's perception of security often includes that he himself is involved in the security mechanisms and thus has a certain amount of control. But this may also make a system susceptible to certain threats. For example, the user can be confused or persuaded into corruptive actions by hackers manipulating information. This subject, known as social engineering, is studied in detail by Cybenko et al. [2].

3.1 Specific security issues

Security in the specific context of TINKERBELL is considered with respect to the following four criteria:

- *Confidentiality* is the guarantee that information is shared only between a user and the entities the user is willing to communicate the information to.
- *Authenticity* is the assurance that the mobile phone in a Bluetooth connection has the claimed identity and has subscribed to the TINKERBELL service.
- *Integrity* means the correctness of stored and communicated personal (and annotation) data in the sense that only the corresponding person (the author or a responsible moderator) can alter them.
- *Availability* means that the TINKERBELL service is accessible and usable for subscribed persons using appropriate mobile devices.

Confidentiality First of all, it is worth noting that the design of the TINKERBELL service should allow the user to prioritize the confidentiality of his data. That is, he should be able to declare which of his data are absolutely or first-grade confidential (i.e., no one should be allowed to read them), second-grade confidential (only members of certain TINKERBELL user groups should be allowed to read them), third-grade confidential, and so on down to the nth-grade confidential, where access in granted to every other TINKERBELL user regardless of identity.

This somehow reflects a quite discrete notion of the idea of a so-called *digital aura*, a thought model for spontaneous interaction between moving objects defined by Ferscha et al. [3]. A digital aura reflects the "aura" of an object (person or thing) in the context of wireless communication. As soon as two objects are located near each other, they start comparing profile data (which would be nth-grade confidential data in our scenario); further communication and data exchange is enabled only in case the profiles of the two objects are sufficiently similar. (Here the notion of "similarity" has to be defined appropriately on a context-by-context basis.) This concept of digital auras has also successfully been implemented into a software framework enhancing the development of context-aware peer-to-peer applications [4].

Given our discrete notion of digital auras, the central question is how the user can declare which grade of his personal data may be provided to which persons/mobile devices.

A first and simple possibility might be to introduce blacklists with names or device IDs of persons for which access should be denied. Similarly, whitelists ("friend lists") can name people or mobile devices for which access is permitted. This of course in turn requires methods for assuring a certain grade of confidentiality and integrity of the blacklist (and whitelist) data. A disadvantage of blacklists is that the user can define them only if the contact to some inconvenient discussion partner has been established at least once. (Otherwise the user wouldn't know the partner is undesirable.)

A second possibility, aiming at denying contact to inconvenient persons in advance, would be to annotate persons/device IDs. The registration process alone is a first step in this direction. Of course, here an additional security aspect concerns the annotation data, which need to be stored on a server. Not only the usual considerations apply here, but we also see a difference in trust

into a system when it hosts such personal data, with the potential for police and comparable agencies to be empowered to access these data immediately on short notice.

Privacy aspects are of particular importance when annotating data on persons. In case someone knows one of the user's secrets, he can publish this secret in the user's annotations—a massive privacy attack! Even if the policy of the annotation service allows for removing annotations of a particular personal kind, there is the risk that third parties will become aware of the annotations (and thus of the user's secret) before the user himself notices and removes them. Of course, in the first step, third parties will not be able to assign any person to the negative annotations, because they stick to a device ID. But in case contact is built up, the secret can be associated to the corresponding person. Even more simple, the attacker can place its victim's name into the annotation.

Because of these risks, there is a moral barrier which may get in the way of acceptance of annotating persons. A method preventing too big a misuse of the annotation idea is to trace back the attacker via his or her device ID, which could be stored for the author of each annotation.

A third possibility would be to preselect possible contact partners by their profiles. Here some part of the user's nth-grade confidential data could be used to form a profile which is matched with the corresponding profiles of other TINKERBELL users. A conceivable policy might be to establish a contact only to those whose profiles have a similarity with the concerned user's profile above some certain threshold. In order to keep even these profile data confidential in case of a mismatch, a good policy might be to use one-way functions to encode the profile data and then execute the matching algorithm on the encoded data, which cannot be decoded (cf. [7]). In case the threshold is not reached, further access is denied and both parties don't even know each other's profiles used for matching.

Authenticity Regarding authentication, one small remark can be made in advance: even if authentication is provided, the user can never be sure that the authenticated partner behaves as desired. In other words, authentication does not establish trust. But this is of course a problem in our scenario which cannot be solved, because no one can prevent the owner of the authenticated device from misusing the data exchanged.

Perhaps TINKERBELL will provide a chance to systematically use the idea of disclosing differently "watermarked" information to other parties. People sometimes do this when filling out forms, such introducing intentional misspellings into their name, like "Jonn" instead of "John". When John later receives unwanted advertisements addressed to "Jonn", he has feedback on his information disclosure behavior and can learn to adapt and/or report the unscrupulous communication partner to a consumer protection agency. In many ways TINKERBELL could easily support such strategies.

Keeping the social aspect of TINKERBELL in mind, one guideline for authentication may be to approximate the human way of judging and talking to unknown

or newly encountered people. People can make up their mind in a conversation with somebody rather unknown to disclose rather private information or they can decide very quickly that they don't want to. TINKERBELL extends the possibilities here. In case your TINKERBELL rings you up and tells you that a partner you encounter matches a profile you are looking for, but the real-life counterpart does not give you the same impression, you might become suspicious.

TINKERBELL, viewing it as a mirror of its owner, can store or be used as a substitute for biometric information of its owner. This could be used to identify the user to his TINKERBELL or to other users. It is further conceivable that a TINKERBELL, like government identification and credit cards, might host a picture of its owner, which in many cases can ease authentication.

Integrity We are aware of the notion of "Bluetooth snarfing"—the access and deletion of the stored data on mobile phones via Bluetooth. However, we realize that on the deeper levels of hacking systems, to a large degree we need to rely on third party efforts. TINKERBELL will probably allow users to institute a policy of completely denying acceptance of any executable code. Further, encryption and signing of information stored on TINKERBELLs could be feasible to some degree (cf. [7]).

Availability To achieve availability, several points have to be taken into consideration. First, and trivially, the server used must be prevented from breaking down. Moreover, it might be helpful to have as little data and as few modules as possible within the responsibility of the server. Second, also trivially, one has to take care of enabling Bluetooth communication whenever it is demanded. Additionally, denial of service attacks have to be prevented. These may include, for instance, spamming.

4 Conclusions

We have introduced TINKERBELL, an application for MDAs (mobile digital assistants, state-of-the-art mobile phone computers) which we consider to be a pervasive computing application. TINKERBELLs contact other TINKERBELLs in the vicinity (distance of Bluetooth reach), engage in small-talks, and give away a configurable amount of information about their owners, in order to instigate or facilitate social interaction.

A crucial area of concern with this technology is that the transfer of small-talk to the virtual world invites mismatches of the human feeling/assessment for the situation. In the real world, humans do give away their phone numbers, even to people they haven't known for long, but not to everybody. It needs to be ensured that (users instruct) their TINKERBELLs (to) behave in a comparable way so that unwanted surprises do not happen.

TINKERBELLs disclose and send information to others; they act in the name of a person and host information about their owners. Thus, it is clear that

they are vulnerable in principle and that the security issues of how to assure confidentiality, authenticity, and integrity need to be taken care of.

TINKERBELLs and their owners will typically have a one-to-one relationship, with reach into the real world. (Mainly they appear together, even if TINKERBELLs can be given away or be stolen.) They are perhaps comparable to the ownership of cars—to a degree they open up the possibility to trace the activity of a person and/or his TINKERBELL. With a limited disclosure of information about all the individuals, this is enough to open up the chance for new (socially motivated and socially grounded) security applications.

Acknowledgments. The authors would like to thank Steffen Lange for helpful discussions.

References

1. Peter Brown, Winslow Burleson, Mik Lamming, Odd-Wiking Rahlff, Guy Romano, Jean Scholtz, and Dave Snowdon.
2. George Cybenko, Annarita Giani, and Paul Thompson. Cognitive hacking: A battle for the mind. *IEEE Computer*, 35(8):50–56, 2002.
3. Alois Ferscha, Manfred Hechinger, Rene Mayrhofer, Marcos dos Santos Rocha, Marquart Franz, and Roy Oberhauser. Digital aura. In *Proceedings of the 2nd International Conference on Pervasive Computing*. Springer-Verlag, 2004.
4. Alois Ferscha, Manfred Hechinger, Rene Mayrhofer, and Roy Oberhauser. A light-weight component model for peer-to-peer applications. In *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops*, pages 520–527. IEEE Computer Society, 2004.
5. Dieter Hutter, Günter Müller, Werner Stephan, and Markus Ullmann, editors. *Proceedings of the 1st International Conference on Security in Pervasive Computing (revised papers)*, volume 2802 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.
6. Bertin Klein, Stevan Agne, Achim Ebert, and Michael Bender. Enabling flow: A paradigm for document-centered personal information spaces. In *Proceedings of the 8th IASTED International Conference on Artificial Intelligence and Soft Computing*, 2004.
7. Håkan Kvarnström, Hans Hedbom, and Erland Jonsson. Protecting security policies in ubiquitous environments using one-way functions. In Hutter et al. [5], pages 71–85.
8. PAMPAS consortium and constituency. Deliverable D04: Final roadmap (extended version). Technical Report IST-201-37763, PAMPAS consortium, 2004.
9. Philip Robinson and Michael Beigl. Trust context spaces: An infrastructure for pervasive security in context-aware environments. In Hutter et al. [5], pages 157–172.
10. Ian Smith, Anthony LaMarca, Sunny Consolvo, and Paul Dourish. A social approach to privacy in location-enhanced computing. In *Proceedings of the Workshop on Security and Privacy in Pervasive Computing*, 2004.
11. Mark Weiser, Rich Gold, and John Seely Brown. The origins of ubiquitous computing research at PARC in the late 1980s. *IBM Systems Journal*, 38(4):693–696, 1999.