Ubiquitous Computing Security:

Authenticating Spontaneous Interactions

René Mayrhofer

A collection of publications

universität
wien

Submitted as Habilitation Thesis
to the Faculty of Computer Science,
University of Vienna

by

René Mayrhofer
Faculty for Computer Science
University of Vienna
Universitätsstraße 10, A-1090 Wien, Austria
Email: rene@mayrhofer.eu.org

September 15, 2008

Preface

This habilitation thesis ("Sammelhabilitation") collects and summarises original research by the author, primarily in the area of security for spontaneous interaction. Spontaneous interaction is one of the key aspects of ubiquitous computing, and securing such spontaneous interactions between devices that typically communicate over wireless and therefore invisible channels requires human-verifiable authentication. Sub-topics discussed in this thesis include interaction methods, cryptographic protocols, and sensor data analysis.

The thesis consists of two parts: Part I defines the focus of this specific research area, methodically reviews the current state of the field, puts the collected publications into perspective, and summarises the author's contributions. Part II contains reproductions of the twelve publications collected in this thesis.

Dedication

To my wife Petra Annemarie Wagner-Mayrhofer.
Writing this thesis would not have been possible without your continual support. Thank you.

I also wish to extend my sincere gratitude to all my co-authors and colleagues, who have made research on this topic highly interesting, fruitful, and fun.

Contents

Part I

Spontaneous Device Authentication: State of the Field and Own Contributions

1 Introduction

Within the last seven years, the research topic of *security for ubiquitous computing* has been tackled by an increasing number of scientific publications; papers concerned with security or privacy issues in relevant conferences on ubiquitous computing started at around 3% in 2001 and have now reached over 20% in 2007[1]. Smaller workshops and conferences focusing on the very intersection of classical computer security research and ubiquitous computing have also started to appear[2]. In this habilitation thesis, the focus lies on *security for spontaneous interaction*. This is a new topic, enabled by rapid technological progress in miniaturisation, embedded sensing, and wireless networks.

Currently, security mechanisms for client/server and desktop computing demand the user's attention, for example to enter different usernames and passwords, PIN or TAN codes, or even cryptographic key material in hexadecimal notation, or to verify hashes of cryptographic keys (which is usually neglected anyways) many times throughout the day. This approach still works (albeit badly, as demonstrated by the increasing number of security incidents) because those annoyances typically happen infrequently enough so that users can still get their actual work done. However, future computing environments are expected to be mobile, massively parallel, and highly heterogeneous. Following the vision of ubiquitous computing, users may interact with hundreds, if not thousands, of services each day, and most of these services will be used for the first and only time. Such *spontaneous interactions* between users and their environment result in ephemeral and usually wireless connections without prior knowledge of the communication partner. This is difficult both in terms of user interaction – how to select the correct service – and even more so in terms of securing these connections. To effectively prevent potential attacks on such spontaneous interactions, each one requires independent and *human-verifiable authentication*. It is an issue of trust: only users can decide which other users, devices, or services are trustworthy enough in their current situation, and in turn users need to be able to trust the wireless communication underlying their interactions. Users need to be able to perceive, comprehend, and ultimately control what their devices do. This is the core topic of the present habilitation thesis and most of the collected publications it consists of. First approaches towards securing wireless spontaneous interactions and explicitly considering users as part of the authentication protocols have only been published in 2002 [KZS02, BSSW02]. Before that, the whole area had been practically unexplored, and it is still insufficiently structured today. My own contributions to this young topic include three specific, novel spontaneous authentication methods [6, 4, 9], a general concept and specific implementation for context authentication proxies [10, 11], an open source toolkit of authentication protocols to, for the first time, support comparative studies of approaches among the research community and assist application designers [12] and more detailed work on cryptographic protocols [8], security analysis of out-of-band media [5], and prototyping [7].

[1] 1 out of 29, 1 out of 29, 3 out of 27, 1 out of 26, 0 out of 22, 1 out of 30, 7 out of 29, and 3 out of 42 at Ubicomp 2001 to 2008, respectively; 1 out of 20 at Pervasive 2002, 4 out of 26, 1 out of 20, 6 out of 24, and 4 out of 21 at Pervasive 2004 to 2007, respectively; 5 out of 64, 4 out of 37, 4 out of 39, 4 out of 38, and 5 out of 28 at PerCom 2003 to 2007, respectively.

[2] For example ESAS, the European Workshop on Security and Privacy in Ad hoc and Sensor Networks 2004 to 2007; PerSec, the International Workshop on Pervasive Computing and Communication Security 2004 to 2007; or IWSSI, the International Workshop on Security for Spontaneous Interaction 2007 and SPMU, the Workshop on Security and Privacy Issues in Mobile Phone Use 2008, both of which I co-organised with other members of the research community.

Achieving authentication of a priori unknown services or devices spontaneously and unobtrusively, yet in way that is verifiable by human senses, demands a holistic design. Issues including hardware design, sensor data handling, machine learning, embedded systems, cryptographic protocols, user interfaces, and mental models all need to be considered in the design of usable and secure authentication methods. This interdisciplinary nature of the topic makes it necessary to focus on specific aspects. The present thesis focuses on human-verifiable authentication of wireless communication between devices. Its aim is, on the one hand, to summarise part of my research work in this area, but, on the other hand, also to introduce a first taxonomy for structuring the research area from a user point of view.

In the following, I will first explicitly define the focus of this thesis (section 1.1) and introduce the associated vision of ubiquitous computing (section 1.2) along with its core aspect of spontaneous interaction (section 1.3). Section 2 will then describe security methods for securing interactions in ubiquitous computing in general, including security requirements and potential threat scenarios. Many of these threats can be prevented by authenticating wireless communication, which is the focus of this thesis and presented in detail in section 3, including an analysis of the main challenges (section 3.1), cryptographic protocols (section 3.2), auxiliary channels suitable for authentication purposes (section 3.3), and a comprehensive survey of authentication methods using these protocols and channels (section 3.4). Although my own contributions to this research area are also listed along other protocols and authentication methods, I summarise and review them in more detail in section 4. In the last section 5 of the first part, I present currently ongoing work and an outlook for future research in this highly active field.

Part II collects all publications that are submitted as part of this (collective) habilitation thesis and therefore presents the details of my own contributions as summarised in section 4.

1.1 Focus

This habilitation thesis focuses primarily on authenticating spontaneous interactions in the area of ubiquitous computing, and specifically on three aspects:

- *user interaction methods* for association between mobile devices and/or stationary services with implicit, human-verifiable authentication,
- *cryptographic protocols* to facilitate secure agreement of shared secret keys using a main wireless and an auxiliary channel with appropriate properties,
- and *sensor data analysis techniques* to support the auxiliary exchange of messages between devices in a human-verifiable manner.

Security and usability issues surrounding these aspects, such as secure communication channels making use of the shared secret that devices have agreed upon during authentication, secure storage of such keys, preventing side-channel attacks on a hardware level, power management on mobile devices, safeguarding user privacy, or social and legal implications are explicitly considered out of scope of the present thesis.

1.2 Ubiquitous Computing

Ubiquitous Computing is a vision. The famous quote from Mark Weiser's seminal 1991 article [Wei91] that

> The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.

still perfectly describes the essence of ubiquitous computing — the vision that computing technology should be integrated with daily life. Such integration fundamentally changes how users interact with technology. When aiming for continuous use of information technology services distributed over the environment, users may want to interact with hundreds or even thousands throughout their day, and many of these services will be used just once.

Part of this upcoming change is already visible in the changed usage scenarios brought by the hype around web services. By moving from "personal" services in the form of locally installed programs and data storage – the one-user-one-device case – towards lightweight clients (currently browsers, in the near future mobile devices) accessing diverse and distributed services – the one-user-many-devices case –, previous assumptions on human computer interaction are already being invalidated. One example is that required training to use a service efficiently must now be kept minimal. It will rarely be acceptable to force users to spend learning time when they only intend to use a service once or sporadically. Another example is that neither users nor devices can automatically be considered trustworthy anymore. Potentially malicious users now have access to critical services that, when previously being installed locally on the users' own devices, did not require protection against unauthorised use. On the other hand, it seems practically impossible to establish trust relationships with arbitrary public services prior to using them for the first (and most probably only) time. This leads to requiring mutual authentication between users and services, with the effect that popular public services keep (typically private) databases of millions of essentially unverified user accounts while users struggle with lists of tens of different account details for partially trusted services.

For ubiquitous computing, this trend is expected to continue, and additional assumptions will most probably be invalidated as well. A hint of upcoming changes can be taken from current driving factors for ubiquitous computing, including technological drivers like general miniaturisation, improvements in processing speed, memory, and communication bandwidth, always-on telecommunication infrastructure, and inclusion of sensing into everyday devices as well as social drivers stemming from the move towards an information society. These driving factors lead to increasingly mobile devices with sufficient local resources and sensing possibilities that are always connected to the Internet, and towards continuous use of diverse local and distributed services.

There are two additional core aspects of most work on ubiquitous computing. First, the notions of *context* and the associated *context awareness* concentrate on adapting applications to the situation or environment, that is, the context, they are used in (cf. e.g. [DA00]). Every action or interaction happens in one or multiple overlapping contexts such as "at work", "in a meeting", "in a theatre", or simply "sitting", "standing", or "sleeping". Contexts may for example be mutually exclusive, hierarchically structured, or overlapping, and they are typically recognised (semi-) automatically from different sensors (cf. my dissertation thesis [13]). The first three of the constituent papers are concerned with context recognition [1, 2, 3] and lay the basis for authentication based on context and environmental sensing.

Second, the increase in user mobility and volatility of relationships between users and services demands *spontaneous interaction* between users and their changing, context-dependent opportunities for using services in their environment.

1.3 Spontaneous Interaction

Spontaneous interaction is one of the core topics of ubiquitous computing and interwoven with its primary vision — to support the serendipitous use of (informational) services whenever and wherever it seems most appropriate to the user. Especially in the context of using a service for the first (or only) time this is difficult because no prior knowledge can be assumed on any of the involved layers. This includes the lowest layers of (wireless) communication, middle layers of networking protocols and service discovery, and upper application layers dealing with data formats, representation, and user interaction issues.

For each of these layers, various approaches to achieve the desired level of ad-hoc interoperability have been suggested. For interoperable communication, standards such as IEEE 802.11 wireless LAN, IEEE 802.15 Bluetooth or IEEE 802.15.4 ZigBee can be used. Device and service discovery may be defined by the communication stack, for example Bluetooth, or may use a protocol capable of working with multiple communication links, for example SLP, UPnP, or mDNS and DNS-SD. For interoperable data exchange, approaches reach from distributed tuple spaces to infrastructures defined especially for spontaneous interaction (e.g. the Obje interoperability framework [NES08]). However, there are (at least) two common issues to consider for all the layers: context and security. Context awareness is expected to be highly beneficial for efficient spontaneous interaction, for example if only those services are shown by the respective service discovery mechanism or only those data items displayed on the application layer that conform both to the user's situation and to their intent. Security also needs to span all the layers to prevent various threat scenarios, especially when no prior knowledge about the communication partner is available for any of the layers. Generally, when devices and services can not offer this prior knowledge because they are interacting for their first time, the only instance that *can* make informed choices on which services to use and how to interact with them is the human user.

Putting the Human in the Loop

As humans, we are unable to directly sense wireless communication, and therefore have no way of verifying which devices are communicating with each other. This makes attacks on the wireless communication channel possible and likely, allowing a malicious party to not only attack the main (wireless) communication channel, but also all the higher layers. On the other hand, we can very well decide which devices should participate in the spontaneous interaction and thus communicate wirelessly with each other, and which should not. This is an issue of trust, and is therefore specific to the interaction at hand. For example, we may place enough trust in a public but otherwise unknown display for viewing a newspaper article, but may not want to entrust it with private medical or financial records due to the risk of them being invisibly copied and forwarded. Therefore, even if there was a globally accepted trusted third party[3] for certifying devices and users, this would not solve the problem of authenticating spontaneous interactions.

Security – and in particular authentication – for spontaneous interaction therefore needs to consider users as part of the system, even more so than with traditional applications.

2 Security Methods for Ubiquitous Computing

Before going into detail about methods for authenticating spontaneous interactions, it is important to review what we actually mean by security: that is, what requirements we have for secure information technology systems and which threats we need to protect against. In this section, I provide a brief but systematic introduction of security especially for systems in the area of ubiquitous computing.

2.1 Requirements

Secure systems are often defined to fulfil three basic requirements, also called the "CIA triad" (cf. the classical paper [CW87] or many newer textbooks):

Confidentiality means that private data should only be accessible to authorised users. It is sometimes also called *secrecy*.

Integrity means that it should be impossible to undetectedly modify protected data.

Availability means that authorised users should always (or at least at clearly defined time periods) be able to access data or services. The implication is that unauthorised users should be unable to deny access for authorised users.

[3] Typically, such trusted third parties would be implemented as Public Key Infrastructures (PKIs) for scalability reasons. Because every PKI suitable for mass-production of devices needs at least one so-called root certificate as an "anchor of trust", global deployment has obvious trust issues [ES00]. Even the technical feasibility of secure global-scale PKIs is disputed [Gut].

A straight-forward way to achieve the first two requirements is to apply the well-known cryptographic methods of encryption and digital signatures. However, there are usually no solely technical means to solve the last requirement, and comprehensively addressing all three of them always involves *authentication* in some way. Additional requirements in a broader context are:

Authenticity means that it should be possible to determine the source or originator of data items or messages.

Non-repudiability extends authenticity to third parties and means that, when forwarding data items or messages from the original recipient to a third party, this third party should also be able to determine the original source and that the original sender should be unable to disclaim generating it.

Authorisation and Access Control describes the procedure of determining which – identified and authenticated – users or devices should be authorised to access specific data, services, or execute specific actions.

Auditability means that it should be possible to, after the fact, reconstruct access to data items, services, and executed actions. It is typically achieved by securely storing traces and logs of relevant actions.

The last two requirements are often more organisational than technical aspects, and are not discussed further within the scope of this thesis. However, their implementations can build upon the more basic "CIA" requirements as well as authenticity and non-repudiability.

Roughly five decades of research on cryptography have given us the tools to, on the technical level of communication channels and protocols, tackle the requirements of confidentiality, integrity, non-repudiability, authorisation and access control, and auditability once authenticity of all parties has been established. The crucial step is therefore to ascertain who users and devices are — that is, to authenticate them.

Authentication is the guarantee that a communication partner is who they claim to be. More specifically, a *subject* (or *verifier*) authenticates (demands proof for) that the *object* (or *prover*) is who or what they claim to be. The *attribute* of authentication defines which aspect of the object is authenticated.

In ubiquitous computing, subjects and objects are usually (groups of) users or stationary or mobile devices, but can also be services, single actions, or whole organisations or institutions. Most common attributes are some form of identity (such as the name of a human user or the network address of a device), absolute location, or relative position, but can also be ownership, affiliation/employment, the current behaviour/task/action of the object, and past behaviour/task/action (for example gathered via trust and reputation models). The most important distinction is, within the scope of this article, the object of authentication. We can therefore distinguish between:

User authentication is the guarantee that the correct user is interacting with a system, that is, an authentication where the object is a user. This is the most common form of authentication in today's client/server and desktop computing environments and well known from username/password methods or biometrics.

Device authentication is the guarantee to communicate with the intended device, that is, an authentication where the object is a device. It primarily prevents so-called man-in-the-middle (*MITM*) attacks on the (wireless) communication channel and is expected to become increasingly more important as users start to demand proof that they are indeed interacting with the intended service.

In this thesis, I focus on authentication from a user point of view, independently of the involved communication partners of a specific interaction.

2.2 Threat Scenarios

Complete security seems impossible. We can therefore only claim to fulfil any of the security requirements in a clearly defined scenario, somehow restricting what we assume to possibly go wrong. In practice, threat scenarios define what we assume malicious users (potential attackers or so-called adversaries) to be capable of and which of these threats a system can protect its legitimate users against. Ubiquitous computing builds upon established concepts and systems, including processing, storage, and networks, and is therefore open to all typical threats known to computer security research. The combination of established technology along with the expected changes in its use cases, however, creates new threat scenarios that are especially relevant to ubiquitous computing applications. We can classify these threats into three different levels:

1. *Physical attacks on devices* are possible when an adversary can gain physical access to a hardware device, for example the user's personal mobile device or the embedded system that runs some informational service integrated into the environment. For scenarios where users access services in their environment or communicate with others using their personal mobile devices, such physical access creates the following particularly relevant classes of threats:

    a) *Replacement* of devices allows an adversary to put their own device (or at least one controlled by them) into a context (for example a location) where a trustworthy service is expected by other users. These users may then, due to the known context and potentially good behaviour in the past, put more trust in spontaneous interactions with the adversary's device than is warranted.

    b) *Modifications* to hardware or software allow an adversary to subtly change the behaviour of otherwise still functional devices or services so that security breaches, for example capturing all user data and sending it to the adversary, can remain unnoticed by users. Especially critical are modifications to the users' personal trusted devices, for example by installing key loggers to capture passwords and cryptographic key material otherwise not stored on any device or back doors for remote access and control.

    c) *Side-channel attacks* are a powerful method for an adversary to learn something about the inner functioning of an otherwise closed system. By externally measuring some physical phenomenon, internal algorithms and data can sometimes be determined using statistical methods. The most well-known side-channel attacks rely on electromagnetic radiation (for example to read video signals transported over a cable from a distance) and power analysis (for example to determine cryptographic keys stored in dedicated embedded crypto processors like smart cards).

2. *Wireless attacks on communication* are possible when an adversary is in range of the wireless communication link between legitimate devices. Without attacking the devices themselves, they can perform different potential attacks:

    a) *Eavesdropping* allows to read all or parts of wireless communication without the legitimate communication partners being aware of this. Technically, most wireless channels can be monitored fairly easily, and we therefore have to assume all messages that are sent wirelessly to be completely public. To protect against eavesdropping, messages must be encrypted.

    b) *Denial-of-Service* (DoS) describes violations of the security requirement of availability. That is, when adversaries are able to prevent wireless communication between legitimate devices, they can deny authorised users access to specific services. Due to wireless channels intrinsically being a shared broadcast medium, most wireless communication standards can be easily disrupted. We therefore have to assume complete or selective DoS attacks to be possible and have to design authentication methods and protocols to be able to cope with it, for example using timeouts on the application level. Making DoS attacks harder is out of the scope of this work.

c) *Message injection* allows an adversary to create new messages and let them be received by a legitimate device, potentially with fake sender identities. Without some way to guarantee authenticity of messages, any recipient could be fooled into believing that another legitimate user or device sent the message while it was generated by an attacker.

d) *Modifications to messages in transit* violate the requirement of integrity of messages when adversaries are capable of modifying parts of messages during transmission. Depending on the specific wireless channel, it may require sophisticated equipment to mount the attack undetectedly.

e) *Man-in-the-Middle* (MITM) attacks describe the most powerful class of threats. As a combination of the other threats to wireless communication, an adversary is assumed to have complete control over the wireless channel between two (or multiple) legitimate devices. That is, the adversary can eavesdrop on, block, or modify any of the messages sent on the channel as well as inject new messages. The effect is that legitimate devices believe that they are communicating with each other while they are really communicating with the attacker, who acts as a "hub" for communication and may modify it as they wish. In practice, many wireless channels such as IEEE 802.11 WLAN can be easily manipulated to gain the capability of MITM attacks (for example, by creating virtual access points and letting legitimate devices communicate on different channels). Therefore, we need to assume all wireless channels to be completely open to attack and that MITM-type adversaries are present. Cryptographic protocols and mutual authentication are required to prevent this threat, which is the main focus of the present habilitation thesis.

3. *Social attacks on users* allow an adversary to manipulate spontaneous interactions without attacking hardware, software, or (wireless) communication.

a) *Confusion* is a broad class of threats in which users interact with another user, device, or service than they intended to. This can in many cases be a benign mistake caused by devices being positioned close to each other, similar names or shapes, etc. and is the responsibility of the respective user interface to prevent (cf. our work on using spatial user interfaces for secure device interaction [4, 10]). However, confusion can also be used by adversaries to maliciously cause users to interact with their devices or services instead of the original ones. Unfortunately, it is hard to protect against this class of threats in the general case — only for specific application scenarios, user interfaces, and groups of users can the potential for confusion be analysed and effective counter measures be designed. We therefore have to assume that confusion happens and try to design user interfaces, cryptographic protocols, and generally applications so that the impact is minimised.

b) A *false sense of security* can result from different mistakes, benign or caused by malicious attackers. One important threat is that users construct wrong mental models about how a system or its security measures work. This can lead to security measures not being used at all, or being used incorrectly and therefore opening possibilities for attack (for example confusion). While wrong mental models can result from non-optimal user interfaces or be encouraged by an adversary presenting fake information, they are essentially mistakes in communicating how a system works. On the other hand, so-called bidding-down attacks mean that adversaries actively manipulate authentication protocols (for example during the course of a wireless MITM attack) so as to present the user only a restricted (and typically insecure) list of choices. This kind of attack exploits that fact that authentication protocols are often designed to be extensible and interoperable among heterogeneous devices, and therefore offer multiple options. To prevent a false sense of security, both cryptographic protocols and user interfaces need to be designed to limit the options a user has to choose from but instead be secure by default, and to make internal procedures more explicit, for example by visualising a list of steps that needs to be completed.

The security methods discussed in the following mainly address threats to wireless communication by preventing MITM attacks. Social attacks on users can not be prevented on the protocol level, but protocol design can mitigate the potential for mistakes. Physical attacks, on the other hand, are out of scope of this work.

2.3 Physical Security

For all of the following discussions, we explicitly assume the legitimate devices participating in a user-initiated spontaneous interaction to be secure. That is, that the specific interaction as intended by the user will be performed by all legitimate devices. This does not mean that all devices or services a user may want to use need to be completely secure; for many tasks, it will be acceptable to use insecure devices because the action or data in question does not have any security requirements. Examples are manifold, including to read news articles on public displays, browsing the web for non-critical information, virtual black boards, etc.

When critical data or actions are part of some spontaneous interaction, then the involved devices need to be trusted (and trustworthy) for the particular transaction. The authentication methods discussed below can not prevent any physical attacks on devices or misuse of data that has been transferred in a legitimate protocol run. However, application and protocol design can mitigate physical attacks happening before or after a spontaneous interaction. In terms of applications, critical data should be stored in an encrypted form and cryptographic keys should be kept in memory only as long as strictly necessary. After it has been used, all key material should be overwritten and wiped from memory (as for example done in my OpenUAT toolkit [12]). One important element in protocol design is *forward secrecy*, sometimes also called *perfect forward secrecy* (PFS). By making cryptographic keys independent of each other, for example using public key cryptography such as the Diffie-Hellman key agreement protocol, previous session keys will not be compromised if other keys are successfully attacked at a later stage. By ensuring key independence, each spontaneous interaction needs to be attacked separately.

Although physical attacks and key compromise can not be prevented using the methods discussed in this habilitation thesis and are therefore out of scope, appropriate protocol design can at least mitigate the effects of such attacks. Wherever possible, this has been considered in the design of my cryptographic protocols and authentication methods (for example in [4, 6, 9]).

3 Authentication

Mutual authentication solves all of the threats to wireless communication except DoS. When understood and applied correctly, it also addresses social threats to users. However, this requires the authentication process to span the layers and not only include the wireless channel but also the user.

Each interaction has to be authenticated separately to ensure user-intended behaviour, a property that is sometimes referred to as being *human-verifiable,* of giving *physical evidence* [KZ03a], as *demonstrative identification* [BSSW02], or as *data origin authenticity* [WS06].

Physical evidence of the attribute used to authenticate the object is a new requirement for secure systems in ubiquitous computing, and fulfilling it requires a second communication channel called *auxiliary* (this term is often used in descriptions of cryptographic protocols) or *out-of-band* (OOB) channel (this term is often used when referring to the specific out-of-band medium that channel is based on) that is used to authenticate communication on the main (in-band), wireless channel. In the scope of this thesis, the terms auxiliary and out-of-band are used synonymously. For user-initiated spontaneous interactions, the auxiliary channel needs the human in the loop as final instance for verifying authenticity. When users are subjects of an authentication, we also speak of *human verifiability* as a subset of physical evidence.

The need for human-verifiable authentication is independent of the duration of the interaction, from permanent device *pairing* that lasts until the "end of life" of one of the (virtual) entities like in the "resurrecting duckling" model [SA99] to one-shot interactions. To abstract from the duration of an interaction, we speak of device association in contrast to device pairing, which is often used to imply a longer-lived association.

Device association is the act of creating an authentic communication channel between two or multiple devices, independent of the duration of application-level interactions.

The most common auxiliary channel used by desktop-type interactions is formed by explicit, standard user interfaces and users themselves: a service may identify itself on a display (usually without authentication), and users identify themselves (via usernames or tokens) and enter their passwords or similar secret shared data on keyboards or keypads for authentication. To reach mutual authentication, both the service and the user – or the respective devices that represent all parties – need to authenticate each other in some meaningful way.

What to authenticate?

Usually, *authentication* follows *identification*. That is, after a user or device was identified via some means like entering or transmitting usernames, device addresses, or complex compound identifiers in certificates, they need to provide proof for this identity, for example by entering or transmitting passwords, PIN codes, biometric information, or by providing proof of possessing the private key corresponding to a signed certificate. Although this two-step approach is used in the majority of cases, it is not necessary for creating secure channels — there are more properties that can be authenticated and that may be more suitable for spontaneous interactions in ubiquitous computing environments. From a user point of view, device and user identities are in fact unimportant for many applications. They usually want to interact – securely – with "that" device in front of them, that is, with a *physical entity*. Any addresses or naming schemes, that is, the *virtual entities*, are often not unique (think for example of printer names in different office buildings or mobile phone names as discoverable via Bluetooth), but only make sense in a specific context like a network realm of control. Therefore, authentication for spontaneous interactions should usually authenticate the physical instead of the virtual entity.

For communication, virtual identities are still required, for example in form of network addresses or names. However, spontaneous interaction lends itself very well to authentication without (permanent) identification; (virtual) network identities need not be permanent and need only be unique during a single interaction. In fact, virtual entities can be ephemeral and used only for a single spontaneous interaction akin to the use of nonces (random numbers used just once) in cryptographic protocols. Thus, by using physical entities for *anonymous* or *pseudonymous authentication* between ephemeral network identities and binding them to each other, secure interaction can be provided while users' privacy can be protected more easily than when relying only on virtual entities.

3.1 Challenges

Mutual authentication during spontaneous interaction poses three main challenges:

1. *Wireless communication is insecure.* As detailed in the threat scenarios in section 2.2, we have to assume wireless channels to be completely open to attack. During spontaneous interaction, communication partners have no prior knowledge about each other. On a standard wireless channel such as Wireless LAN, Bluetooth, or ZigBee, there simply is no information that would distinguish the intended communication partner from another one nearby or, worse, from a malicious attacker.

2. *Small, mobile devices often lack appropriate user interfaces.* Many devices that may be used spontaneously, such as headsets, goggles, printers, or smart cards, do not have any explicit user interfaces such as displays and keypads that could be used for traditional approaches to authentication like entering usernames and passwords.

3. *User attention does not scale.* Future small, mobile devices may be able to cope with hundreds or thousands of authentication protocol runs a day, but users will not. Entering passwords or verifying hex digits is already obtrusive enough for logins performed once or twice daily, but will be out of the question when spontaneity is desired.

These challenges necessitate new approaches to mutual authentication. Most recent authentication protocols can deal with the insecurity of wireless channels, but working without explicit user interfaces and scaling to many interactions throughout the day require fundamentally new methods. Especially the problem of scalability marks a second important difference between authentication for ubiquitous computing applications and standard desktop applications (besides the possibility for anonymous authentication based on physical entities). Users will most probably be unable to authenticate personally to each service they intend to use due to the obtrusiveness of the process, at least as long as this process is explicit. Implicit user authentication, for example using unobtrusive and automatic biometric methods such as face or voice recognition, is one option to make user authentication scalable. Another one is add a layer in between the user and the service they intend to use:

Personal device authentication shifts this burden from users to their personal devices. Users need to authenticate to personal devices that they carry with them at all times, such as mobile phones or wrist watches, only when picking them up or activating them, for example once a day. These personal devices then act as representatives for users, by keeping keys, passwords, and generally authentication information for interacting with other devices and services, and are used to transparently interact with services in a pervasive computing environment. The personal device becomes a *proxy* to replace frequent user authentication with background device authentication, and thus unifies both notions for the purpose of using services.

This option has the advantage that such a personal device can, unobtrusively and automatically in the background, run complex cryptographic protocols and represent multiple pseudonyms a user may have, therefore protecting the users' privacy much better than (implicit) biometric user authentication would allow to. First prototypes using personal devices have been suggested, for example using PIN codes for authenticating to the personal device and explicit pairing to services [CN03], and we expect mobile phones and other personal devices to be used increasingly for this purpose. In the remainder of this habilitation thesis, I therefore focus on methods for *device authentication*, for example between a personal mobile device and a service embedded in the environment or another mobile device, and leave a study of methods for implicit user authentication to future work.

3.2 Protocols

One of the main issues is how device-to-device authentication can be made spontaneous and secure at the same time. Consequently, a substantial number of protocols have recently been proposed. The “resurrecting duckling protocol” [SA99] is one of the initial proposals, and suggests a role model for devices with long-lived pairings, where a “duckling” device imprints itself with the network identity and access credentials of a “mother duck” device, and will stay paired to it until reset to its initial state. This model seems appropriate for many scenarios like pairing home devices (TV sets, DVD players, audio players, etc.) to universal remote controls, pairing headsets to mobile phones, et cetera. This static pairing relationship has been generalised by so-called “key continuity” concepts which are now in wide-spread use, for example in SSH and by caching of X.509 server certificates in web browsers, email clients, etc. The initial suggestion was to use direct electrical contact to establish the device association and transfer the required secret keys, which seems difficult in practice, not only because of the lack of standards for direct device connections. Subsequently, protocols for device association using a broader range of auxiliary channels have been suggested, most of them based on standard Diffie-Hellman (DH) key agreement with additional authentication:

- Gehrmann, Mitchell, and Nyberg suggested a family of protocols for “manual authentication” [GMN04]. The MANA I scheme allows a user to enter a short key displayed on one device on the keypad of the other; in MANA II, the user compares short keys displayed by both devices; and in MANA III, the user enters the same short key into both devices, or transfers it from one to the other like in MANA I. MANA IV [LN06] is a generalisation of these protocols with the abstract concept of some auxiliary channel and theoretical analysis of security properties. With the current state of the art, MANA IV is a secure basis for authentication protocols and we fully recommend to use it.

- Balfanz et al. were among the first to explicitly talk about using out-of-band channels such as infrared or audio for authentication, and presented two protocols making use of a human-verifiable authentic channel [BSSW02]. The first exchanges public keys over the out-of-band channel before starting a standard protocol like TLS or IKE over the wireless channel, while the second one uses only symmetric cryptography for resource limited devices, although without providing encryption.

- Hoepman also introduced pairing protocols for short-lived interactions based on manual exchange of secrets [Hoe04], which is very similar to MANA III [GMN04] and seems to have been developed independently.

- However, Vaudenay claims [Vau05b] that Hoepman’s protocol can not be implemented securely due to the lack of known hash functions with properties required by the protocol, and presents a protocol called SAS, which provides the same level of security with shorter shared secrets. SAS has later been determined to be of the MANA IV family of protocols [LN06].

- Creese et al. introduced a formal model for verifying authentication protocols that work with empirical verification [CGH$^+$05]. They presented an analysis of three related pairing protocols and show proofs of their security under their model.

- Čagalj, Čapkun, and Hubaux described three other pairing protocols with similar aims, based on short string comparison, distance bounding, and integrity codes [ČČH06]. Their second protocol is based on distance measurement, but we suggest that their scheme might be applicable to an interactive challenge-response scheme based on arbitrary sensor data.

- Wong and Stajano reviewed a few of the previously introduced protocols with regards to resistance against eavesdropping on the auxiliary channel and attacks on short key strings

[WS06]. They show an extension of the MANA III scheme that is similar to Hoepman's protocol in that it relies on the authenticity of the auxiliary channel instead of its secrecy, and propose a new protocol for asymmetric authentication with a one-directional auxiliary channel and limited input interfaces on one device.

- My own "Candidate Key Protocol" (CKP) differs from the above protocols by broadcasting candidate key parts that other devices recording similar sensor data can "tune into" [8]. In its current form, it is susceptible to brute-force offline attacks when the sensor data has low entropy, but offers a more dynamic and more scalable interaction model than the protocols based on Diffie-Hellman. CKP breaks new ground in protocol design using a different approach to key agreement, in some ways comparable to the new field of "fuzzy cryptography" (e.g. [DRS03]).

Note that, although these publications often speak of displays, keypads, and the user entering or comparing short key strings, the protocols are more widely applicable. On an abstract level, these variants of authentication protocols use multiple channels with different security properties, for example a completely open wireless channel and some low-bandwidth authentic and/or secret auxiliary (out-of-band) channel. Two notable publications have independently introduced concepts for such restricted auxiliary channels: Kindberg, Zhang, and Shankar defined the abstract concept of "constrained channels" as one-way channels that can be restricted in both sending and receiving messages, for example by location [KZS02]. Similarly, Balfanz et al. introduced "location-limited channels" as channels on which human users can control which devices are communicating with each other, and which only need to be authentic, but not secret [BSSW02]. Wong and Stajano have more recently looked at more formal ways of defining channel properties [WS06]. These concepts may be used to abstract from particular physical properties of out-of-band media that form the basis for auxiliary channels and concentrate on their use within authentication protocols.

These cryptographic authentication protocols using multiple channels are not complete solutions to device authentication — they also require an out-of-band channel with specific security properties. Different channels have already been proposed, depending on the application scenario at hand. In the following, I will systematically introduce out-of-band media that are suitable for authenticating spontaneous interactions.

3.3 Auxiliary Channels

3.3.1 Model and Methodology

To analyse and compare auxiliary channels, we need a model. Within the scope of this habilitation theses, there are two major dimensions along which to discuss such channels. First, their security properties define how auxiliary channels can be used as part of an overall authentication protocol, that is, which guarantees a protocol can assume of a specific channel. The second important dimension is the user point of view and will we discussed in more detail in the subsequent section 3.4.

In terms of security protocols, an auxiliary channel can offer the following relevant properties:

Confidentiality of a channel means that only those devices that the user intends to interact with (each other) can read messages transmitted via this channel. An example is entering a secret PIN code into a keypad and shielding it from potential reading by other users or cameras.

Complete (human-verifiable) authenticity guarantees that the recipient of a message – either a target device or a user – can without any doubt determine the sender of this message. The channel gives *physical evidence* of message transmission. An example is reading (secret of public) messages from a display that is securely embedded into a device.

medium			user sensibility	inter-action	confiden-tiality	complete auth.	partial auth.	in-tegrity	stall-freeness	ref.
		keypad input	direct	input	potentially	X		X	X	[Blu06, STU07b]
optical	visual	display	direct	verify	potentially	X		X	X	[PS99, Blu06, KZI05]
		camera	direct	input		X		X	X	[MPR05, SEKA06, BDHV07]
		laser	direct	input			X	X	X	[KZ03a], [9]
		infrared	indirect	input or implicit			X			[AH07, BDG$^+$04]
sound		audible	direct	verify		X			X	[ZJC07, GSS$^+$06, STU07a, KZI05]
		ultra-sound	indirect	verify			for angle of arrival	for dis-tance	X	[4, 10], [KZ03b]
		motion	direct	input	potentially	X		X	X	[6], [KSW07, BSHL07, PPA04]
		radio frequency	indirect	implicit			poten-tially (cf. [CM05, VSLdL07, ČČH06])			[CM05, NSHN06, VSLdL07, ČČH06]

Table 1: Out-of-band media and their properties

Partial (non-user-verifiable) authenticity guarantees that some physical property of the channel can be used to determine the sender of a message, but that this property is not directly verifiable by human users. Examples are infrared or ultrasonic messages, both of which allow a recipient to partially determine from where a transmission originated.

Integrity is a lesser guarantee than authenticity and, in the case of auxiliary channels, means that messages can not be modified by an attacker but that there is no evidence of the sender.

Stall-freeness guarantees that an attacker can not buffer and delay transmission of messages; when a sender has transmitted the message on the channel, a recipient is guaranteed to receive it instantaneously (barring physical transmission delay). This is typically the case for all human-verifiable channels.

3.3.2 Analysis of Potential Out-of-Band Media

Table 1 summarises media that have already been suggested as out-of-band channels for authentication purposes and gives their security properties according to the above model. Interaction properties from a user point of view as discussed in the model of section 3.4 are also included in the table as well as references to specific authentication methods that use the respective channels.

Keypad input Direct symbolic input is limited to both the number of buttons on the respective device and the speed a user can type on them. This creates a trade-off between the keypad size – ranging from standard-size keyboards on laptops to single, small buttons – and the channel bandwidth. With low hardware cost and/or constrained size, this means that moderate to high security levels require noticeable user effort. However, using keypad input for authentication rarely creates any additional hardware effort, as most devices are already equipped with at least one button. An advantage from a user point of view is the "physical touch" and consequently the direct mental association which devices are part of the interaction.

Display Displays can be available in a wide range of resolutions, sizes, and brightness levels. Large and bright displays enable interactivity over larger distances, while high-resolution ones offer higher channel bandwidth. Maximum display size is obviously constrained by the respective device size and additional hardware effort, that is, added cost, can be significant. From a user point of view, sufficiently capable displays can support multiple authentication methods and thus offer flexible use-cases for different application scenarios. Direct mental association of involved devices is also intuitively supported.

Camera When using the visual medium for direct data transfer between devices (e.g. [MPR05]), cameras are the counterparts to displays. They have a similar trade-off between resolution and cost. Up to moderate interaction distances where complex optical lenses are not required, camera size is less of an issue. Additional hardware effort for adding cameras to devices depends on the device type and application scenario: for mobile phones, cameras no longer add significant cost, while smaller devices would need to be equipped with digital signal processing in addition to the actual camera sensor. Note that, from a user point of view, it is important that users can see what the camera captures, and therefore devices with cameras will also need to be equipped with appropriate displays.

The bandwidth is sufficient to not only transfer authentication information as part of the cryptographic protocol, but also identity information such as network addresses to start the initial communication.

Laser Laser transmission is asymmetrical: one device requires a laser diode for transmitting, the other a suitable photo sensor for receiving. Hardware effort for the transmitter is very low, as laser diodes are cheap, small, and have low power consumption. Necessary hardware effort for the receiver mainly depends on interaction distance (larger sensors are required) and lighting conditions (at daylight, more sensitive signal processing is required). User effort is very low, as lasers only need to be pointed at a target device and the subsequent transmission is practically instantaneous. Although open to human senses, actual transmission of messages over visible lasers guarantees only partial authenticity. The reason is that, while the "dot" created by partial reflection where a laser beam strikes the sensor is visible, the actual message is transmitted in coded form and too quickly to be perceptible. Is is therefore theoretically possible for an attacker to override a message transmission with their own, potentially stronger modulation. The bandwidth of lasers is also sufficient to transmit identity information.

Infrared Infrared transmission (sending and receiving) incurs low hardware effort and is available in many mobile devices such as mobile phones or laptops. The bandwidth of infrared communication is also sufficient to transmit identity information.

Audible sound Audio transmission uses a shared medium; in the audible range, it is also shared with users, with the advantage of being directly observable and the disadvantage of distracting users in vicinity of the transmission. If users are intended recipients of audio transmissions (in the case of verification interaction), then user effort is moderate to high due to the required attention. Hardware effort is moderate for microphones and associated audio data processing, but speakers can add some size and weight to mobile devices. However, many mobile devices such as mobile phones, PDAs, and even some wrist watches are already equipped with both microphones and speakers, making it an attractive medium for off-the-shelf devices. The bandwidth of audio transmission is also sufficient to transmit identity information.

Ultrasound Ultrasound is also a shared medium, but with current technology can only be used with limited bandwidth for secure transmission. Additional hardware cost is high because of required signal processing and because no off-the-shelf device currently includes ultrasonic transducers. Such transducers also have a minimal physical size, which might increase the device size. User effort, on the other hand, can be low to nonexistent.

Motion Hardware effort for using motion as an input channel is low, because accelerometers are cheap, small, and consume little power. Signal processing requirements are also low due to slow data rates (in comparison to audio or optical transmission). The bandwidth can be moderate to high depending on signal processing effort, but as an input-only channel, motion can not be used to directly transmit identity information.

Radio frequency Radio frequency as a medium for authentication is a special case, because wireless channels are considered in-band channels. When using the same medium for authentication, some additional properties besides the data transmission capability are exploited to create the required out-of-band character. Its bandwidth depends on the specific authentication method (e.g. low for “Shake Them Up” [CM05], moderate for “Amigo” [VSLdL07], and potentially high for “LoKey” [NSHN06] but in this case requiring trust in the network operator).

A special form of an out-of-band channel with physical evidence is to rely on the human body and direct contact as a transmission medium [Zim96], although we are currently not aware of specific use of such a channel for authentication purposes.

3.4 Human-Verifiable Authentication Methods

Specific human-verifiable device authentication methods may use any of the out-of-band channels discussed above by combining it with appropriate cryptographic protocols and user interaction. For their analysis and comparison, we again need a model.

3.4.1 Model and Methodology

The following taxonomy, introduced for the first time in this habilitation thesis, approaches the recently suggested methods mainly from a user point of view, taking into account how they may be used in practice. There are two major aspects in which we can classify device authentication methods:

Interaction The style of *user interaction* defines if the user needs to *verify* the proof of or provide *input* for authentication. In the verification case, devices involved in the authentication transmit some information to the user, for example by displaying, which the user then verifies and acknowledges if it was verified correctly. The most common form of verification is to compare information transmitted by two different devices for equality, but other forms of verification are possible. In the input case, the user transmits information to involved devices in some form, for example by entering symbols or by providing input to device sensors.

User “sensibility” The medium used for information transmission can be either *directly sensible by human users*, that is, match any of the human senses, or only *indirectly*, that is, by exploiting physical properties of the medium. In the first case, users can directly monitor the property that is used for authentication, for example by verifying a displayed key hash or

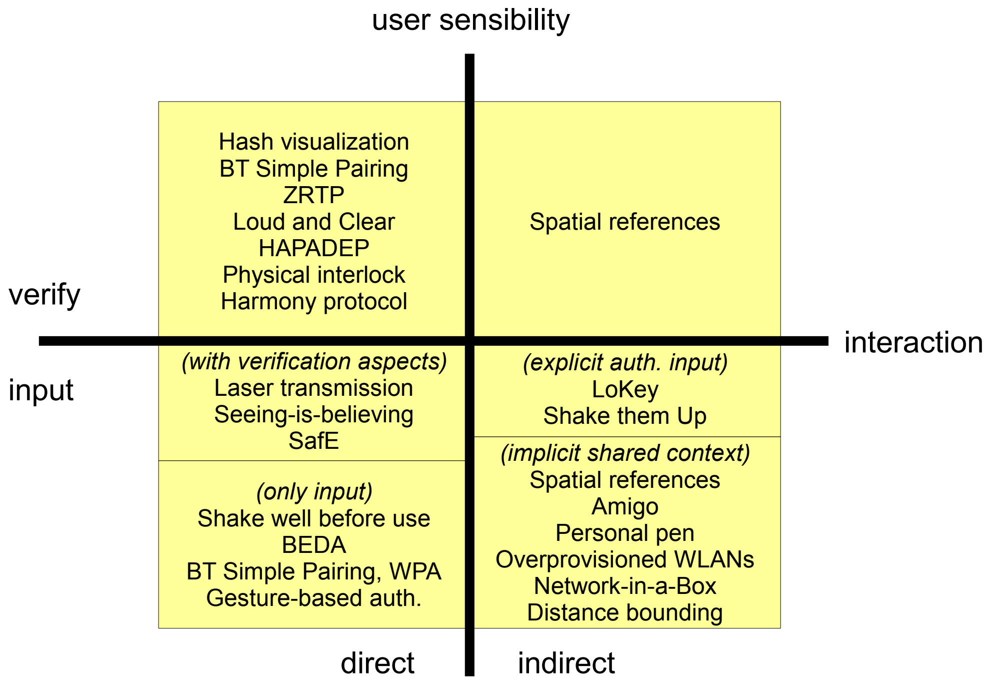


Figure 1: Taxonomy of (personal) device authentication methods from a user point of view

by providing input in the form of aiming a laser or a mobile phone camera at the physical device to interact with. In the indirect case, authentication is based on a property that can neither be directly observed nor directly influenced by the user, for example location-based authentication measured with radio frequency signals. Only the indirect effects of the used property can be verified, for example based on visualisation, or influenced, for example by bringing devices sufficiently close to each other.

Based on the combination of interaction and user sensibility, Figure 1 shows four emergent classes of device authentication methods. For input interaction, we can further distinguish into two specific sub-classes, both for direct and for indirect sensibility. Each of the four major classes defines the *information flow* between sources and sinks of information transmitted over the respective out-of-band medium, which will be discussed in more detail below. The provided authentication guarantee can, depending on this information flow, be either *one-way* or *mutual*; in the latter case we also speak of symmetric authentication.

3.4.2 Direct user verification

Information flow Devices (sources) transmit information to users (sinks), who act as the final verifying instance (in many cases by comparing). Because the out-of-band channel is directly observable, physical authenticity of all involved devices is implicitly guaranteed.

Design implications This first class of methods is the most "obtrusive" one from a user point of view, as users need to explicitly verify some property that is directly sensible and/or compare between multiple devices. On the other hand, it implicitly provides introspection in the form that users verify what their devices authenticate, and thus provides good support for building mental models. Due to the obtrusiveness of a separate verification step in the interaction, applications should (try to) embed it into the normal workflow at a stage where user attention is already focused on the association or selection process. A good example is ZRTP [ZJC07], where users can start a voice-over-IP call by reading words to each other. Such an interaction can be a natural part of starting the first call with a new communication partner, and be seen as part of the first call establishment (like searching for the number the first time it is used).

Methods On the lower end of the required effort to implement and use the method, Perrig and Song suggested "Hash Visualization" [PS99], which transforms key hashes to random art images for easier explicit comparison by the user. This is one method to allow users to compare key strings between multiple devices, e.g. required by the MANA II protocol. They also described an application to use hash visualization for user authentication by letting users choose a set of images from a larger displayed set (user authentication instead of device authentication). When all involved devices feature high-resolution displays (moderate hardware effort), then this method can offer high security (by embedding whole checksums into generated random art images) with low user effort (a quick glance will often be sufficient).

Bluetooth Simple Pairing [Blu06] offers the "Numeric Comparison" mode, which requires users to compare 6-digit numbers displayed on both devices and confirm if they match or not. This provides lower security by restricting the compared data length at the cost of higher user effort due to symbolic comparison, but lowers the hardware effort for all involved devices.

For voice-over-IP applications, Zimmerman et al. suggested ZRTP, currently under review as an RFC draft [ZJC07] and recently extended by Hlavacs et al. [HGS+08]. ZRTP provides in-band key agreement, authentication, and key continuity based on ephemeral Diffie-Hellman keys created opportunistically during each protocol run. Session keys are derived from the ephemeral DH keys and cached key material from the previous call to prevent MITM attacks on subsequent calls. For the first call between two devices, short authenticated strings encoded as words can be compared by the users reading them to each other via the established ZRTP channel (relying on user's capabilities to recognise the communication partners' voices). This method provides high security, requires low additional hardware effort for the niche area of VoIP devices, and low effort for users.

Goodrich et al. presented "Loud and Clear" [GSS+06], which uses normal audio as out-of-band channel. As part of the authentication protocol, devices speak or display non-sensical English sentences derived from key material, which users need to compare and verify that they match. Soriente, Tsudik, and Uzun later described an extension called "HAPADEP" [STU07a], which supports playing piano tunes derived from key material in addition to non-sensical sentences. Additionally, it uses the audio channel to directly transmit data which is received with the other device's microphone, and thus does not require an additional wireless communication channel. These two methods provide moderate security with low hardware effort. "Loud and Clear" requires users to spend moderate to high effort (comparing textual and/or spoken sentences), while HAPADEP might be used with slightly lower effort.

On the other end of the spectrum, Kindberg, Zhang, and Im proposed three different protocols [KZI05] that require high user effort but are limited in their security by allowing only short key strings to be compared. The first is similar to the hash visualisation method, as it requires users to compare key strings, but suggests either textual representation or to use a bit-wise multimedia pattern that is synchronised between the two devices, e.g. blinking LEDs, colour blocks on a display, or beeps. Their second method is called "physical interlock" and requires the users of devices that should be authenticated to co-operate with "sign language", e.g. to react to commands like "raise your left hand". Comparing strings bit-for-bit or symbol-for-symbol requires high user effort per bit of provided security. Their third method, the "Harmony" protocol, allows users to compare synchronised media streams, e.g. one device playing bass, the other the matching piano tune, comparable to one of the use cases for HAPADEP.

All of these support (at least optionally) mutual authentication, where both communication partners authenticate each other.

3.4.3 Direct user input

Information flow Users (sources) transmit information to involved devices (sinks), which use the provided information as direct input to authentication protocols. Because the out-of-band channel is directly controllable, physical authenticity of all involved devices is implicitly guaranteed.

Design implications This class of methods is characterised by the fact that users provide explicit input in some way that is open to human senses and which is directly used for authentication purposes. We can further distinguish methods that depend only on user input in some form, and methods where user input is coupled with aspects of verification. By providing input, users may require less attention (and thus spend less effort) than with methods using verification.

When the respective input is an integral part of the interaction, such as selecting a device to interact with, authentication may even be performed implicitly and automatically, relieving the user of any additional steps “just for security”. Such implicit authentication has the potential to make interactions secure by default, and thus to see wide usage in practical settings. Applications should try to combine selection with authentication by using the same property for both actions. Examples are selecting a printer by pointing a laser to it in our own protocol [9] or by taking a picture of its attached 2D barcode [MPR05], or selecting a Bluetooth headset by shaking it together with the mobile phone that should use it in our “Shake well before use” method [6].

Methods using input with verification At the lowest end of required user effort among the recently suggested methods combining input with verification aspects is to use visible lasers to select and implicitly authenticate devices. Kindberg and Zhang first proposed to use the laser beam as a confidential out-of-band channel to transmit key material as part of an authentication protocol [KZ03a]. This protocol assumes that the laser emits no light except onto the receiving sensor (and that no light is reflected from the latter), which may not be valid when considering attackers with free line of sight to either the sender or the receiver. Recently, we presented [9] an improved protocol which no longer requires this assumption of confidentiality, but only the more realistic and significantly weaker assumption of partial authenticity. This method provides high computational security due to the high bandwidth of lasers as out-of-band channels, but, depending on the application scenario, might only provide moderate protection against impersonation of one of the devices due to the asymmetric authentication. By adding visual feedback (for example LEDs) as “back channel”, high security can be provided in all cases. Hardware effort is low to moderate and requires a cheap laser diode on one side (typically the mobile device) and a suitably sized light sensor on the other.

McCune, Perrig, and Reiter presented “Seeing-is-Believing” [MPR05], a protocol and working implementation for using cameras in mobile phones as a human-verifiable out-of-band channel. Again, to at the same time select a device to interact with and to transmit authentication information, the target device displays a 2D barcode (or uses static, printed barcodes when no display is available), which the user can simply capture with their mobile phone camera to initiate the interaction. Saxena et al. extended this method [SEKA06] with a more secure authentication protocol and to support devices that do not have a full screen but only an LED, which is used to transmit the authentication information via blinking patterns. This approach allows users to directly verify what the sensor, that is, their camera phone, measures. In comparison to a personal device equipped with a laser diode and the service equipped with a sensor, it swaps the roles of sender and receiver on the out-of-band channel. The advantage is that it is easier for users to verify the authenticity of the other device because authentication in the protocol sense matches what the user verifies. On the other hand, it forces the user to pay closer attention than for simply pointing a laser at a target device and thus requires slightly higher (but still low) user effort. With moderate hardware effort (cameras with sufficient resolution and light sensitivity), high security can be provided.

Buhan et al. suggested SAfE [BDHV07], a proposal to also use cameras for authentication between mobile phones. However, instead of 2D bar codes, users are presumed to take facial pictures of each other, which are then used as biometric authentication data for newer cryptographic key storage techniques like e.g. the widely cited "Fuzzy Extractors" by Dodis, Reyzin, and Smith [DRS03]. This method provides moderate to high security depending on the threat model with moderate user and hardware effort. The additional advantage is that the facial picture used for authentication could also be used as future, highly intuitive reference for a communication partner.

For these methods, the distinction between verification and input interaction is slightly blurred, because users both provide input, for example by pointing a laser or a mobile phone camera towards the device or user to interact with, and are involved with further verification, for example by verifying that there is no interference in the optical out-of-band channel such as a second laser aimed at the same receiver. However, we classify them primarily as input interaction because the input ("pointing") part defines the major user interaction component. All methods provide one-way authentication but can be made symmetrical by reversing the procedure when both devices feature the required hardware (for example cameras).

Methods with user input only Among methods that rely only on user input, common motion seems to require the least user effort and at the same time provides high security with low hardware effort (devices need to be equipped with accelerometers). Our most recent work called "Shake well before use" [6] presented an interaction method to associate two or multiple devices by shaking them together for a few seconds. The common movement is also used to implicitly authenticate the device association based on accelerometer sensor data. The "Martini Sync" [KSW07] and a method by Bichler et al. [BSHL07] follow the same concept but differ in detail, with the latter providing low security but without requiring communication over a wireless channel during the authentication protocol.

Soriente, Tsudik, and Uzun presented "BEDA" [STU07b], a method to associate devices by repeatedly pressing and releasing, at the same time, one button on each device. A suggested variant of this approach is to press and release the button on one device synchronised to the other device's LED blinking. Due to the limited bandwidth of this input channel, it provides moderate security with negligible hardware effort and low to moderate user effort.

Bluetooth Simple Pairing [Blu06] and Wi-Fi Protected Setup are currently standardised authentication schemes with PIN or password entry. In Bluetooth Simple Pairing, the "Passkey Entry" mode requires a user to enter a 6-digit number displayed on one of the devices into the other. This requires higher user effort than quasi-random button presses, but can provide high security with low hardware effort.

Patel et al. presented [PPA04] a method to temporally pair a user interface device, e.g. a keyboard and a display, to a personal mobile device where the user interface authenticates to the mobile device by displaying a gesture sequence, which is then followed by moving the mobile device. Note that, although the authors talk about user authentication, the subject of the authentication is the external user interface, and not the user. It is also unclear how this scheme protects against active man-in-the-middle attacks, suggesting further research. From a user point of view, this method seems the most involved, but may be appropriate for specific usage scenarios. Hardware effort is low, but the security of the suggested protocol is currently unclear.

Bluetooth Simple Pairing, Wi-Fi Protected Setup, common motion input, and BEDA in two-button mode provide mutual authentication.

3.4.4 Indirect user verification

Information flow When the medium used for out-of-band transmission is not directly user sensible, users can not be recipients (sinks) of the information transmitted by devices (sources). Therefore, another (trusted) device must receive this transmission and relay it to the user in observable form, for example by visualisation.

Design implications The consequence for combining verification interaction with out-of-band media that are not directly sensible is that authentication can only be secure when information transmission between involved devices offers some physical guarantees that authentication can depend on (for example ultrasonic messages between devices with guarantees on the direction they were received from as used in our protocol for spatial authentication [4]). It is important that the required information transformation, for example visualisation, is accurate enough so that the probability of users confusing devices (for example by occlusion in the visualisation) is kept sufficiently low. This additional layer of indirection may be open to new threat scenarios and thus needs to be analysed as part of the whole system, both in terms of security and usability.

Methods Kindberg and Zhang proposed a method to verify device authenticity by their positions [KZ03b]. Users are supposed to perform manual ultrasound triangulations: the angle of arrival of ultrasound pulses emitted by the target device is visualised, and by performing this process multiple times, it is suggested that users can verify the physical position of devices. Security of the proposed protocol is currently unclear, and user effort is high.

We have improved this concept to include relative distance and formulated it more specifically as using "Spatial Reference" for authentication, as well as to support implicit authentication when devices are selected based on their visualised spatial position [4, 10]. In a specific implementation of authentication with relative spatial relationships we also also used ultrasound, based on our analysis of the security properties of ultrasound as an out-of-band channel [5]. In contrast to other location based authentication methods such as using infrared or WLAN signal strength, ultrasound provides higher accuracy and thus allows to visualise positions for user verification. This method can be argued to either let the user verify an implicit property (the relative spatial position of the target device) or to provide input (the same position). The classification of this method into this or the next class therefore depends on the specific application scenario it is used in. In both cases, user effort is low (as studied in our work [10]) for moderate to high security depending on the threat scenario, but hardware effort is high.

Both methods in principle provide one-way authentication, but in our implementation both devices perform spatial and thus mutual authentication.

3.4.5 Indirect user input

Information flow As with the previous class, authentication based on physical properties that are not directly user sensible depends on their effects. Coupled with input interaction, users can create the necessary pre-conditions to fulfil the authentication property, but are not directly involved with its verification. Information transmission can only happen between devices (sources and sinks). Although the difference between indirect verification and indirectly providing user input may seem subtle from a security point of view, providing common input to devices may consume significantly less user attention than verifying some property, even if embedded into the interaction.

Design implications Because users are no longer involved in the authentication process itself, it can be difficult to create correct mental models: which property is being authenticated, when has authentication been performed, and which security guarantees are provided. Applications should thus (try to) make authentication introspectable, for example using visualisation. Authentication should be transparent and unobtrusive, but not invisible. Otherwise, users may, although they might no longer feel the need to turn off obtrusive security protocols, still use them incorrectly. Particularly, they may, under a false sense of security, perform risky interactions they would not do if they had a correct understanding of the provided security guarantees.

Methods for explicit authentication with indirect user input Nicholson et al. presented "LoKey" [NSHN06], a method and implementation to use the "Short Message Service" (SMS) for authentication between mobile GSM devices. Users provide input in form of the phone number of the target device that is used both for initial connection establishment and for verification as part of the authentication protocol. If phone numbers are used as identifiers for communication partners, user effort is low. Depending on the application and threat scenario, security may be low to high and hardware effort may be low to high.

Castelluccia and Mutaf presented "Shake Them Up" [CM05], a method to authenticate devices based on their spatial proximity. By limiting the transmission range of wireless channels such as Wireless LAN, two devices can create a secret shared key. The trick in this method is that, when users shake the devices and move them around each other during the authentication, then attackers will be unable to distinguish which device sent a specific message. This property is called "source indistinguishability" and is important for the security of the proposed authentication protocol. Similarly to our "Shake well before use" method, device movement is used as necessary quasi-random user input to the authentication process. In contrast to it, however, the authentication channel is (in-band) radio frequency and only (indirectly) influenced by physical device movement. With low hardware effort, moderate to high security can be provided. However, user effort is moderate to high depending on the required security level.

Methods for implicit authentication based on user or device context The main distinction between the previous and this group of methods is that authentication happens implicitly, whenever a certain context is entered. By entering a context (for example a location) the property is fulfilled and authentication is performed. With all methods, user effort is therefore minimal.

In our work on "Spatial Reference" [4], by selecting a visualised spatial position, other devices that are not at this location are (virtually) excluded from authentication/communication (comparable to NiaB where the exclusion is done by physical walls). Therefore, the user also creates the necessary pre-condition for the devices to authenticate each other.

For peer device authentication, Varshavsky et al. presented "Amigo" [VSLdL07], a method for device association based on common radio frequency environment. When devices are co-located, they can sense similar radio environments, e.g. WLAN access points or Bluetooth devices in range. Signal strength measurements are then used as common input to the authentication. Hardware and user efforts are low (with optional "hand weaving" to improve security) while the provided security is moderate to high.

For user or device authentication to access services, several location-based authentication methods have been proposed. Bardram, Kjær, and Pedersen presented "Personal Pens" [BKP03] as contactless RFID-like smart cards coupled with a location tracking system to enable transparent and seamless login of users on changing terminals. This work was motivated by real-world problems of user authentication in hospitals [Bar05]. A similar application was presented by Aitenbichler and Heinemann [AH07], but using infrared instead of RFID-like locationing. Faria and Cheriton also used implicit authentication based on location [FC04], but with overprovisioned wireless LANs for rough location estimation. WLAN clients are authorised when inside physical boundaries, as determined by RF signal strength analysis. Due to their design, these methods can only provide low to moderate security, but require high hardware effort in terms of infrastructure support.

Balfanz et al. presented "Network-in-a-box" [BDG$^+$04], a specific implementation of their concept of location-limited channels. Using an infrared channel, users can bring their laptop into a physically secured room to authenticate wireless LAN connections. This infrared channel is assumed to be secret and authentic, and a custom application uses it to transmit WLAN setup details and key material. Hardware effort is low to moderate, and the provided security can be moderate (due to broad signal characteristics of infrared) to high (using the relatively high bandwidth when devices are restricted to a closed room).

Other approaches to location-based authentication that fall in the same group of methods are the various so-called "distance bounding" schemes, for example the specific proposal presented by Čagalj, Čapkun, and Hubaux [ČČH06] for timing-based distance bounding with ultrasound or ultra-wide band sensing. Distance bounding schemes rely on the physical propagation speed of some signal as well as on the response times of the other device, and are problematic with *wormhole attacks*, where attackers partially "accelerate" their signals by tunnelling them through faster channels. Hardware effort and security can be low to high due to the variety of out-of-band media and specific methods that fall into this category.

With the exception of "Amigo" and "Shake Them Up", which are designed to provide peer authentication and are mutual, these methods provide one-way authentication.

4 Own Contributions

4.1 Publications

My research on security for spontaneous interaction started at the intersection of my work on context awareness for ubiquitous computing and traditional security research in terms of cryptographic protocols. The constituent papers of Part II in this thesis have been selected to reflect my contributions to this field. My work on context awareness is represented by the first three collected publications, which summarise or extend parts of my PhD thesis [13]. They are included in Part II not as novel contributions to the focus of this habilitation thesis, namely spontaneous device authentication, but to introduce the basic notion of and methods for context awareness.

Context awareness is both a pre-requisite for spontaneous interaction and supports sensor-based authentication methods as presented in the previous sections. The first constituent publication [1] therefore introduces a five-step architecture for context recognition and prediction. This was the first article that considered context prediction on an abstract level, taking into account relationships between different, low-level aspects instead of predicting these separately as for example location prediction does. As such, it has already been cited in many recent publications on context prediction by other research groups. In terms of my research work on authenticating spontaneous interactions, this architecture formed the initial building block for integrating different sensors in a common data processing pipeline. Additionally, classification algorithms as studied for context recognition are also directly applicable to classifying sensor readings in terms of making authentication decisions (cf. [6]).

The second publication [2] expands on dealing with heterogeneous sensors and introduces a method to use a multi-dimensional, heterogeneous input space with standard classification algorithms. Instead of the Euclidean space assumed in the standard formulation of many such algorithms, this work shows that, by defining two operators for each dimension of the input space, most classification algorithms can be applied with minimal changes. This was an important result that is still influencing my current work on using various, highly heterogeneous sensors as sources for deciding if devices and services can be authenticated.

The third publication [3] then discusses extensions to one specific classification algorithm, namely "Lifelong Growing Neural Gas", to use a heterogeneous input space and to exploit its internal network structure for recognising high-level contexts. These extensions were initially part of my work focusing on context recognition, but later on informed some of the design decisions for sensor data processing on devices with limited resources.

The fourth constituent publication [4] is the first collected one that deals explicitly with authenticating spontaneous interactions as its main contribution. It expands on research work done mostly by colleagues from Lancaster University as part of the "Relate" EU and EPSRC projects. A peer-to-peer ultrasonic localisation system with an accuracy of up to 10 cm was developed and prototypically implemented in the form of USB "dongles" with integrated ultrasound transducers and wireless networking for synchronisation and co-ordination of ultrasonic sensing. These dongles can be attached to arbitrary mobile or stationary devices to support mutual relative localisation. In [4], we expand the prototypical system to not only provide sensing of relative locations, but to also support implicit authentication based on these relationships, which we termed “Spatial References”. The basic idea is that a user should only need to perform a single step, namely to select which device or service to interact with based on its relative spatial location. Using ultrasonic pulses, the user’s device can explicitly measure the same spatial reference that the user can see. By presenting an authentication method that couples cryptographic key material with this spatial reference, we made authentication implicit. An authentication request as the basis for further (secure) interaction can only succeed when it corresponds to the location selected by the user. In the specific implementation, we achieve this coupling by encoding cryptographic material onto single ultrasonic pulses and using this exchange as part of a key agreement and verification protocol based on standard cryptographic primitives. Specific contributions to the research field are the concept of spatial references, the secure use of ultrasonic communication, a concrete implementation, and a user study showing that a) users understand the concept and b) they can use it with sufficient accuracy.

In the fifth publication [5], we further analyse potential threats on ultrasonic communication based on different potential attacker positions. Depending on where an attacker is located, for example in the same room or directly in between the legitimate communicating devices, they have different options to interfere with ultrasonic communication. The important contribution of this paper is to, for the first time, explicitly analyse threats for ultrasound as an auxiliary channel. This analysis resulted in the development of the ultrasonic pulse encoding applied by our authentication method [4].

The sixth publication [6] represents our second approach to authenticating spontaneous interactions, in this case especially aimed at small, mobile devices. Its main idea is simple: to take those two (or even multiple) devices that should be temporarily or permanently associated with each other in one hand and shake them for a few seconds. This single user action again produces two effects: to start an interaction between those devices shaken together, and to implicitly authenticate their wireless communication channel so that only those devices that have been shaken together can participate in the interaction. In [6], we present two specific methods to use accelerometer sensor data in cryptographic authentication protocols to produce secret, shared, and authenticated keys. The first method uses a conservative cryptographic protocol structure with two phases, key agreement and key verification, and the coherence metric for comparing if the accelerometer time series are similar enough for authentication. In the second method, cryptographic key material is more directly derived from the sensor time series by distributing so-called candidate keys over the wireless channels. Devices can then “tune into” each others cryptographic key stream and produce matching keys if (and only if) their local sensor data is sufficiently similar. The main contributions to the research field are both the overall concept of using shaking for implicit authentication and the two specific methods, which were the first to be published, as well as a systematic and extensive user study that allowed comprehensive analysis of shaking motions for security purposes. In addition to being awarded best Pervasive 2007 paper, this publication has already been cited by an increasing number of other publications despite its relative youth.

In the seventh publication [7], we present a specific implementation of the first method for authentication based on accelerometer data. This implementation was done using off-the-shelf Nokia 5500 mobile phones which already come equipped with 3D accelerometers. The publication describes required changes and detail issues to perform accelerometer-based authentication over Bluetooth and with the limited resources of these standard mobile phones. A short accompanying video published online (see http://www.youtube.com/watch?v=ktJC0S4_X58 with nearly 40.000 views until the time of this writing) received noticeable attention by some large-scale online media[4].

[4] After a featured article in the New Scientist (http://technology.newscientist.com/article/dn12912), it was picked up by Slashdot (http://mobile.slashdot.org/article.pl?sid=07/11/17/1231254), Heise on-

In the eight publication [8], I present the cryptographic protocol underlying the second, more unconventional authentication method presented in [6] in more detail and analyse its properties from a security point of view based on various theoretical results. This "Candidate Key Protocol" is general and applicable to arbitrary sensor time series that two (or multiple) devices can record with some similarity. When used for other sensors than accelerometers as described in [6], only the domain-specific feature extraction parts need to be exchanged. This publication presents a new angle for cryptographic key exchange that has previously not been explored, and may therefore act as the basis for future work in this direction.

In the ninth publication [9], we explored a third option for authenticating spontaneous interactions using visible lasers. In this case, the previous assumption that lasers are confidential auxiliary channels was disputed and I presented a cryptographic authentication method using laser transmission that no longer assumes confidentiality. A prototypical implementation of both a sender based on a cheap laser diode and a receiver using simple components showed that such transmission can be realised with little cost and effort. The contribution of this publication is to show that secure authentication can be performed even under weak assumptions, namely a single, non-confidential and only partially authentic auxiliary channel. This result is also important for other channels besides lasers that may have comparable properties.

The tenth publication [10] studies authentication methods using auxiliary channels on a higher level, abstracting from specific sensing-based authentication methods. It is an extended version of my previous conference paper [14] in which I introduced the concept of *context authentication proxies* to support authentication between two devices, services, or users that can not directly share a specific context and thus not directly use any of the above methods for authenticating spontaneous interactions. A proxy can be used to authenticate one of the parties and, when pre-authenticated to the second party, be used to establish secure authentication and an appropriate trust relationship between the two parties. I analysed different options for realising context authentication proxies, including online and offline relationship with the pre-authenticated party, active and passive interactivity of the proxy itself, and different ways for establishing trust relationships. In the constituent article [10], we included a user study for a specific application scenario of using spatial authentication based on previous work [4] to grant guest devices access to a wireless LAN infrastructure. This user study shows that users tend to prefer the sensing-based authentication over the traditional, better known captive portal with username/password authentication. Additionally, this method allows to grant temporary access via more diverse trust relationships that do not necessarily demand creating user accounts in centralised databases and thus ease the administrator's task. The main contribution of this publication is the concept of context authentication proxies and an analysis of different options. Additionally, the user study shows viability of this approach for a specific application.

The eleventh publication [11] is a precursor to the work on context authentication proxies and also presents a method to use proxies for mediating spontaneous interactions, albeit with a different terminology. Its contribution to the research field was a first protocol for indirect authentication based on RFID as an out-of-band channel as well as a performance analysis of cryptographic primitives suitable for mobile devices with limited resources.

These own contributions and publications by other research groups demonstrate the availability of different methods and auxiliary channels for user-verifiable and sometimes even implicit device authentication. In Table 1, I already summarised the different out-of-band media that have been used so far for device authentication. One promising method to benefit from their different security guarantees is to combine multiple media, for example by letting users enter numbers displayed on one device and transmitted back for verification via modulated, audible sound. Such multi-modal, or multi-auxiliary-channel authentication methods can increase security, but will often do so

line (http://www.heise.de/newsticker/meldung/99142), and ORF futureZone (http://futurezone.orf.at/produkte/stories/236278/), among others.

with the downside of an increased burden for users, who will need to pay attention to multiple media. Additionally, all auxiliary channels need appropriate implementations and are therefore often being neglected to avoid the effort. Suggested approaches mostly represent specific, stand-alone examples and are not easily compare- or interchangeable. The taxonomy introduced in section 3.4 presents one way of classifying and thus comparing approaches and should therefore contribute to the research field by giving design guidelines for future work. However, practical implementations rarely use any of the presented methods because they are currently only documented in the form of scientific papers and – if at all – only available as prototype implementations. To advance the practical influence of research on securing spontaneous interactions, ready-made and easy to use implementations are required.

Consequently, the twelfth publication [12] takes first steps to tie together all the previous research efforts in a unified software toolkit. I introduced "OpenUAT", the open source ubiquitous authentication toolkit, as a project to combine out-of-band channels and interaction methods into ready-to-use building blocks for spontaneous authentication. This should not only allow to further compare methods among different aspects, but also provide implementations that can be used for real application development – taking security into account from the start but relieving designers from re-inventing the details. OpenUAT is already being used as the basis for an ongoing PhD and three Master's projects and further interest has been shown by other research groups.

4.2 Initiated Projects

Lancaster University has been widely known for its research activities in ubiquitous computing. During my post-doc research stay in Hans Gellersen's group, partially as Intra-European Marie Curie Fellow, I started involving other group members with security in ubiquitous computing. Research in this area has been ongoing ever since and will probably result in multiple project proposals within the next year.

I initiated and released the OpenUAT project [12] as open source and have been adding multiple authentication methods since (see the official project web page `http://www.openuat.org`). Starting with its first public release on SourceForge.net in January 2007, it has already been downloaded 232 times until August 2008 (see `http://sourceforge.net/projects/openuat`).

4.3 Supporting the Research Community

To support the formation of a research community in the young area of securing spontaneous interactions, I co-organised two specialised workshops together with other international experts. IWSSI 2007, the First International Workshop on Security for Spontaneous Interaction, was co-located with Ubicomp 2007 and resulted in a special issue of IJSN, the International Journal of Security and Networks, that contains extended versions of many of those publications that helped shape the research field so far. A second result was the creation of the SSI Wiki for coordinating research on security for spontaneous interaction (see `http://www.comp.lancs.ac.uk/iwssi2007/wiki/`). SPMU 2008, the Workshop on Security and Privacy Issues in Mobile Phone Use, was co-located with Pervasive 2008.

5 Future Research

Authenticating spontaneous interactions is still a young research topic, but it has been highly active in the past few years. First real-world impact is expected within a few generations of mobile device products. But being a young topic, relevant publications currently tend to come from different research areas such as context awareness, human computer interaction, sensor data analysis and machine learning, or classical cryptography.

One main task for future research is therefore to unify and standardise cryptographic protocols and auxiliary channels for spontaneous device authentication, and to develop a terminology and frameworks for describing and evaluating new approaches. As a direct extension of my work towards an open source authentication toolkit [12], personal research effort and multiple student projects are currently underway to implement previously published methods within OpenUAT. This includes generalised versions of the approaches using mobile phone cameras and 2D bar codes (cf. [MPR05]), audio transmission (cf. [GSS$^+$06, STU07a]), manual number input, transfer, or verification (cf. [GMN04]), and synchronised button presses (cf. [STU07b]). All these auxiliary channels will – for the first time – be used with the same underlying cryptographic protocol, namely an extended variant of the MA-DH respectively MANA IV protocol [LN06] termed UACAP (Unified Auxiliary Channel Authentication Protocol) that is still to be formally published. By utilising the same cryptographic protocol structure, the authentication methods will become more easily interchangeable and therefore comparable for specific applications. OpenUAT will then be able to support the selection of appropriate methods and offer ready-to-use implementations of the most common ones. An initial user study to compare some of these methods for various application scenarios is currently under preparation.

A second task is to study multi-channel authentication protocols more formally, for example based on the taxonomy presented in this habilitation thesis in section 3.4. Specific new open research questions that have resulted from my work are a direct comparison of authentication methods for the same application scenarios, a study of mental models and understandability, and the impact of different security properties of auxiliary channels on protocol design as discussed in section 3.3. Theoretical analysis should ideally yield security proofs for specific cryptographic protocols in combination with security properties of auxiliary channels. On a higher level, only few comparative user studies have so far been presented. It is well-known that too many options hinder both security and usability, as the effectiveness of any security approach depends on its user's ability to apply them [DGdlFJ04]. This has been discussed explicitly for Bluetooth Simple Pairing and Wi-Fi Protected Setup [KWP07], and studied comparatively by Uzun et al. [UKA07] as well as Valkonen, Toivonen, and Karvonen [VTK07]. Kindberg, Sellen, and Geelhoed also showed that user expectations about security and authentication protocols, that is, their perceived security, need not necessarily be aligned with their real security [KSG04]. Their study points to convenience and social issues as equally important as security and trust issues in system and interaction design. We can take these preliminary results as a strong motivation for considering the whole system when designing authentication methods in ubiquitous computing, most importantly the user interaction process. The analytical study of user mental models as well as recommendations for choosing appropriate authentication methods in various application scenarios and for different user groups, for example children or elderly, is still an open research issue.

Further open issues include the optimisation of authentication methods in terms of improving classification accuracy when sensor data analysis is involved, for example in the "Shake well before use" method, and in terms of run-time and quick user feedback. Both areas of optimisation are currently subject to personal research, for example by explicitly modelling 3D movements from acceleration time series with the aim of compensating the relative alignment of the accelerometers embedded in different devices. Another open research issue is the area of implicit user authentication without mobile devices, for example based on biometric methods and sensors distributed over the environment. One major problem for such implicit user authentication is to safeguard user privacy.

For real-world security, *co*-mobility of devices improves security with appropriately designed protocols that re-key regularly [Vau05a]; an attacker would be hard-pressed to follow the devices and therefore attack the continuous communication stream. Co-mobility, as a subset of context awareness, combined with multiple out-of-band media promises to support user-friendly and secure authentication methods. OpenUAT as one of my most recent and still ongoing contributions to the research field promises to make authentication based on auxiliary channels easy to use and therefore practical for future applications. Spontaneous interaction and security do not necessarily contradict each other — with appropriate methods, they can come together to better support users in their daily tasks.

Bibliography

[AH07] Erwin Aitenbichler and Andreas Heinemann. Proximity-based authentication for windows domains. In *Proc. IWSSI 2007*, pages 475–480, September 2007.

[Bar05] J. E. Bardram. The trouble with login: on usability and computer security in ubiquitous computing. *Personal and Ubiquitous Computing*, 9(6):357–367, November 2005.

[BDG$^+$04] D. Balfanz, G. Durfee, R. E. Grinter, D. K. Smetters, and P. Stewart. Network-in-a-box: How to set up a secure wireless network in under a minute. In *Proc. 13th USENIX Security Symp.*, pages 207–222. USENIX, August 2004.

[BDHV07] Ileana Buhan, Jeroen Doumen, Pieter Hartel, and Raymond Veldhuis. Secure ad-hoc pairing with biometric: SAfE. In *Proc. IWSSI 2007*, pages 450–456, September 2007.

[BKP03] J. E. Bardram, R. E. Kjær, and M. Ø. Pedersen. Context-aware user authentication - supporting proximity-based login in pervasive computing. In *Proc. UbiComp 2003*, pages 107–123. Springer-Verlag, October 2003.

[Blu06] Bluetooth SIG. Bluetooth Special Interest Group. Simple Pairing Whitepaper (Revision V10r00), 2006. http://www.bluetooth.com/Bluetooth/Apply/Technology/Research/Simple_Pairing.htm.

[BSHL07] Daniel Bichler, Guido Stromberg, Mario Huemer, and Manuel Löw. Key generation based on acceleration data of shaking processes. In *Proc. UbiComp 2007: Ubiquitous Computing*, volume 4717 of *LNCS*, pages 304–317. Springer-Verlag, September 2007.

[BSSW02] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proc. NDSS'02*. The Internet Society, February 2002.

[ČČH06] M. Čagalj, S. Čapkun, and J.-P. Hubaux. Key agreement in peer-to-peer wireless networks. *IEEE (Special Issue on Cryptography and Security)*, 94:467–478, 2006.

[CGH$^+$05] S. Creese, M. Goldsmith, R. Harrison, B. Roscoe, P. Whittaker, and I. Zakiuddin. Exploiting empirical engagement in authenticated protocol design. In *Proc. SPC 2005: 2nd Int. Conf. on Security in Pervasive Computing*, pages 119–133. Springer-Verlag, April 2005.

[CM05] C. Castelluccia and P. Mutaf. Shake them up! a movement-based pairing protocol for cpu-constrained devices. In *Proc. MobiSys 2005*, pages 51–64. ACM Press, June 2005.

[CN03] M. Corner and B. Noble. Protecting applications with transient authentication. In *Proc. MobiSys 2003*, pages 57–70. USENIX, May 2003.

[CW87] David D. Clark and David R. Wilson. A comparison of commercial and military computer security policies. In *Proc. 1987 IEEE Symposium on Security and Privacy*, pages 184–194. IEEE CS Press, 1987.

[DA00] A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Proceedings of the CHI 2000 Workshop on The What, Who, Where, When, and How of Context-Awareness*, April 2000.

[DGdlFJ04] Paul Dourish, Rebecca E. Grinter, Jessica Delgado de la Flor, and Melissa Joseph. Security in the wild: User strategies for managing security as an everyday, practical problem. *Personal and Ubiquitous Computing*, 8(6):391–401, 2004.

[DRS03] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. November 2003.

[ES00] C. Ellison and B. Schneier. Ten risks of PKI: What you're not being told about public key infrastructure. *Computer Security Journal*, 6(1):1–7, 2000.

[FC04] D. B. Faria and D. R. Cheriton. No long-term secrets: Location-based security in overprovisioned wireless lans. In *Proc. HotNets-III: 3rd ACM Workshop on Hot Topics in Networks*, November 2004.

[GMN04] C. Gehrmann, C. J. Mitchell, and K. Nyberg. Manual authentication for wireless devices. *RSA Cryptobytes*, 7(1):29–37, 2004.

[GSS+06] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: Human verifiable authentication based on audio. In *Proc. ICDCS 2006*, page 10. IEEE CS Press, July 2006.

[Gut] P. Gutmann. PKI: It's not dead, just resting. published at http://www.cs.auckland.ac.nz/~pgut001/pubs/notdead.pdf, shorter version appeared in IEEE Computer Magazine, August 2002.

[HGS+08] Helmut Hlavacs, Wilfried Gansterer, Hannes Schabauer, Joachim Zottl, M. Petraschek, T. Höher, and O. Jung. Enhancing ZRTP by using computational puzzles. *Journal of Universal Computer Science*, 14(5):693–716, 2008.

[Hoe04] J.-H. Hoepman. The emphemeral pairing problem. In *Proc. 8th Int. Conf. Financial Cryptography*, pages 212–226. Springer-Verlag, February 2004.

[KSG04] T. Kindberg, A. Sellen, and E. Geelhoed. Security and trust in mobile interactions: A study of users' perceptions and reasoning. Technical Report HPL-2004-113, HP Laboratories Bristol, July 2004.

[KSW07] Darko Kirovski, Mike Sinclair, and David Wilson. The Martini Synch. Technical Report MSR-TR-2007-123, Microsoft Research, September 2007.

[KWP07] C. Kuo, J. Walker, and A. Perrig. Low-cost manufacturing, usability, and security: An analysis of Bluetooth simple pairing and Wi-Fi protected setup. In *Proc. USEC 2007: Usable Security*, February 2007.

[KZ03a] T. Kindberg and K. Zhang. Secure spontaneous device association. In *Proc. UbiComp 2003*, pages 124–131. Springer-Verlag, October 2003.

[KZ03b] T. Kindberg and K. Zhang. Validating and securing spontaneous associations between wireless devices. In *Proc. ISC'03*, pages 44–53. Springer-Verlag, October 2003.

[KZI05] T. Kindberg, K. Zhang, and S. H. Im. Evidently secure device associations. Technical Report HPL-2005-40, HP Laboratories Bristol, March 2005.

[KZS02] T. Kindberg, K. Zhang, and N. Shankar. Context authentication using constrained channels. In *Proc. WMCSA 2002*, pages 14–21. IEEE CS Press, June 2002.

[LN06] S. Laur and K. Nyberg. Efficient mutual data authentication using manually authenticated strings. In *Proc. CANS 2006*, volume 4301 of *LNCS*, pages 90–107. Springer-Verlag, December 2006.

[MPR05] J. M. McCune, A. Perrig, and M. K. Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *Proc. IEEE Symp. on Security and Privacy*, pages 110–124. IEEE CS Press, May 2005.

[NES08] Mark W. Newman, Ame Elliott, and Trevor F. Smith. Providing an integrated user experience of networked media, devices, and services through end-user composition. In *Proc. Pervasive 2008*, number 5013 in LNCS, pages 213–227. Springer-Verlag, May 2008.

[NSHN06] A. J. Nicholson, I. E. Smith, J. Hughes, and B. D. Noble. LoKey: Leveraging the SMS network in decentralized, end-to-end trust establishment. In *Proc. Pervasive 2006*, pages 202–219. Springer-Verlag, May 2006.

[PPA04] S. N. Patel, J. S. Pierce, and G. D. Abowd. A gesture-based authentication scheme for untrusted public terminals. In *Proc. UIST 2004*, pages 157–160. ACM Press, October 2004.

[PS99] A. Perrig and D. Song. Hash visualization: a new technique to improve real-world security. In *Proc. CrypTEC'99: Int. Workshop on Cryptographic Techniques and E-Commerce*, pages 131–138, 1999.

[SA99] F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proc. 7th Int. Workshop on Security Protocols*, pages 172–194. Springer-Verlag, April 1999.

[SEKA06] N. Saxena, J.-E. Ekberg, K. Kostiainen, and N. Asokan. Secure device pairing based on a visual channel. Cryptology ePrint Archive, Report 2006/050, 2006.

[STU07a] C. Soriente, G. Tsudik, and E. Uzun. HAPADEP: Human asisted pure audio device pairing. Cryptology ePrint Archive, Report 2007/093, March 2007.

[STU07b] Claudio Soriente, Gene Tsudik, and Ersin Uzun. BEDA: Button-enabled device pairing. In *Proc. IWSSI 2007*, pages 443–449, September 2007.

[UKA07] E. Uzun, K. Karvonen, and N. Asokan. Usability analysis of secure pairing methods. In *Proc. USEC 2007: Usable Security*, February 2007.

[Vau05a] S. Vaudenay. On bluetooth repairing: Key agreement based on symmetric-key cryptography. In *Proc. CISC 2005*, pages 1–5, December 2005.

[Vau05b] S. Vaudenay. Secure communications over insecure channels based on short authenticated strings. In *Proc. CRYPTO 2005*. Springer-Verlag, August 2005.

[VSLdL07] A. Varshavsky, A. Scannell, A. LaMarca, and E. de Lara. Amigo: Proximity-based authentication of mobile devices. In *Proc. UbiComp 2007: Ubiquitous Computing*, volume 4717 of *LNCS*, pages 253–270. Springer-Verlag, September 2007.

[VTK07] Jukka Valkonen, Aleksi Toivonen, and Kristiina Karvonen. Usability testing for secure device pairing in home networks. In *Proc. IWSSI 2007*, pages 457–462, September 2007.

[Wei91] M. Weiser. The computer of the twenty-first century. *Scientific American*, 1496:94–100, September 1991.

[WS06] F.-L. Wong and F. Stajano. Multi-channel protocols. In *Proc. Security Protocols Workshop 2005*. Springer-Verlag, 2006.

[Zim96] T. G. Zimmerman. Personal area networks: Near-field intrabody communication. *IBM Systems Journal*, 35(3&4):609–617, April 1996.

[ZJC07] P. Zimmermann, A. Johnston, and J. Callas. draft-zimmermann-avt-zrtp-04: ZRTP: Media path key agreement for secure RTP, July 2007.

Own Publications

[1] R. Mayrhofer, H. Radi, and A. Ferscha. Recognizing and predicting context by learning from user behavior. *Radiomatics: Journal of Communication Engineering, special issue on Advances in Mobile Multimedia*, 1(1):30–42, May 2004.

[2] R. Mayrhofer, H. Radi, and A. Ferscha. Feature extraction in wireless personal and local area networks. In *Proc. MWCN 2003: 5th International Conference on Mobile and Wireless Communications Networks*, pages 195–198. World Scientific, October 2003.

[3] R. Mayrhofer. Extending the growing neural gas classifier for context recognition. In *Proc. EUROCAST 2007: 11th International Conference on Computer Aided Systems Theory*, volume 4739 of *LNCS*, pages 920–927. Springer-Verlag, February 2007.

[4] R. Mayrhofer, H. Gellersen, and M. Hazas. Security by spatial reference: Using relative positioning to authenticate devices for spontaneous interaction. In *Proc. Ubicomp 2007: 9th International Conference on Ubiquitous Computing*, volume 4717 of *LNCS*, pages 199–216. Springer-Verlag, September 2007.

[5] R. Mayrhofer and H. Gellersen. On the security of ultrasound as out-of-band channel. In *Proc. IPDPS 2007: 21st IEEE International Parallel and Distributed Processing Symposium*, page 321. IEEE CS Press, March 2007. Track SSN 2007: 3rd International Workshop on Security in Systems and Networks.

[6] R. Mayrhofer and H. Gellersen. Shake well before use: Authentication based on accelerometer data. In *Proc. Pervasive 2007: 5th International Conference on Pervasive Computing*, volume 4480 of *LNCS*, pages 144–161. Springer-Verlag, May 2007. **awarded best Pervasive 2007 paper**.

[7] R. Mayrhofer and H. Gellersen. Shake well before use: two implementations for implicit context authentication. In *Adjunct Proc. Ubicomp 2007*, pages 72–75, September 2007.

[8] R. Mayrhofer. The candidate key protocol for generating secret shared keys from similar sensor data streams. In *Proc. ESAS 2007: 4th European Workshop on Security and Privacy in Ad hoc and Sensor Networks*, volume 4572 of *LNCS*, pages 1–15. Springer-Verlag, July 2007.

[9] R. Mayrhofer and M. Welch. A human-verifiable authentication protocol using visible laser light. In *Proc. ARES 2007: 2nd International Conference on Availability, Reliability and Security*, pages 1143–1147. IEEE CS Press, April 2007. Track WAIS 2007: 1st International Workshop on Advances in Information Security.

[10] R. Mayrhofer and R. Gostner. Using a spatial context authentication proxy for establishing secure wireless connections. *Journal of Mobile Multimedia*, 3(3):198–217, March 2007.

[11] R. Mayrhofer, F. Ortner, A. Ferscha, and M. Hechinger. Securing passive objects in mobile ad-hoc peer-to-peer networks. In R. Focardi and G. Zavattaro, editors, *Electronic Notes in Theoretical Computer Science*, volume 85.3. Elsevier Science, June 2003.

[12] R. Mayrhofer. Towards an open source toolkit for ubiquitous device authentication. In *Workshops Proc. PerCom 2007: 5th IEEE International Conference on Pervasive Computing and Communications*, pages 247–252. IEEE CS Press, March 2007. Track PerSec 2007: 4th IEEE International Workshop on Pervasive Computing and Communication Security.

[13] R. Mayrhofer. *An Architecture for Context Prediction*. PhD thesis, Johannes Kepler University of Linz, Austria, October 2004.

[14] R. Mayrhofer. A context authentication proxy for IPSec using spatial reference. In *Proc. TwUC 2006: 1st International Workshop on Trustworthy Ubiquitous Computing*, pages 449–462. Austrian Computer Society (OCG), December 2006. **awarded best iiWAS/MoMM 2007 workshop paper**.

[15] R. Mayrhofer, H. Radi, and A. Ferscha. Recognizing and predicting context by learning from user behavior. In W. Schreiner G. Kotsis, A. Ferscha and K. Ibrahim, editors, *Proc. MoMM 2003: 1st International Conference On Advances in Mobile Multimedia*, volume 171, pages 25–35. Austrian Computer Society (OCG), September 2003.

Part II

Constituent Publications

1 Recognizing and Predicting Context by Learning from User Behavior

Authors: Rene Mayrhofer, Harald Radi, and Alois Ferscha

Published in *Radiomatics:* Journal of Communiation Engineering, special issue on Advances in Mobile Multimedia, 1(1):30–42, ITB Press
ISSN 1693-5152
May 2004

Recognizing and Predicting Context by Learning from User Behavior[1]

Rene Mayrhofer, Harald Radi, Alois Ferscha
Institut für Pervasive Computing
Johannes Kepler Universität Linz
Altenbergerstraße 69, 4040 Linz, Austria
email : {mayrhofer,radi,ferscha}@soft.uni-linz.ac.at

Summary
Current mobile devices like mobile phones or personal digital assistants have become more and more powerful; they already offer features that only few users are able to exploit to their whole extent. With a number of upcoming mobile multimedia applications, ease of use becomes one of the most important aspects. One way to improve usability is to make devices aware of the user's context, allowing them to adapt to the user instead of forcing the user to adapt to the device. Our work is taking this approach one step further by not only reacting to the current context, but also predicting future context, hence making the devices proactive. Mobile devices are generally suited well for this task because they are typically close to the user even when not actively in use. This allows such devices to monitor the user context and act accordingly, like automatically muting ring or signal tones when the user is in a meeting or selecting audio, video or text communication depending on the user's current occupation. This article presents an architecture that allows mobile devices to continuously recognize current and anticipate future user context. The major challenges are that context recognition and prediction should be embedded in mobile devices with limited resources, that learning and adaptation should happen on-line without explicit training phases and that user intervention should be kept to a minimum with non-obtrusive user interaction. To accomplish this, the presented architecture consists of four major parts: feature extraction, classification, labeling and prediction. The available sensors provide a multi-dimensional, highly heterogeneous input vector as input to the classification step, realized by data clustering. Labeling associates recognized context classes with meaningful names specified by the user, and prediction allows forecasting future user context for proactive behavior.

Keywords
Feature Extraction, Context Awareness, Context Prediction, Proactivity, Framework

1. Introduction

Computing environments are changing rapidly, and the pace of this change is currently increasing. Due to broad availability of computing and network infrastructure, the potential audience of computing, communication or other services of informational nature is growing steadily. As a consequence thereof, ease of use becomes a primary concern.

The purpose of our study is to enhance *information appliances* [27] to predict context and deliver proactive services to the user. An information appliance is a device designed to perform a specific function, specialized in information, with the ability to share information with other appliances. They are currently implemented as, for instance, mobile devices or within Pervasive Computing.

Many have already presented their visions of future computers, including Mark Weiser with Ubiquitous Computing [39] (which is also called Pervasive Computing), Steve Mann with Wearable Computing [22], Hiroshi Ishi with Tangible Bits [18] and Hans-Werner Gellersen with Smart-Its [13]. Common to most of them are the paradigms of Mobile Computing and Context Awareness [35]. Although these visions are radically different, they all agree that user interfaces should become less obtrusive and "smarter" with regards to adapting to the user. Today, most interfaces are explicit ones, forcing the user to adapt to the interface, to learn how to use it. If a "Personal Computer" or "Personal Digital Assistant" (*PDA*) would live up to its name, it

[1] This article is an extended version of:
R. Mayrhofer, H. Radi, and A. Ferscha. Recognizing and predicting context by learning from user behavior. W. Schreiner G. Kotsis, A. Ferscha and K. Ibrahim, editors, *The International Conference On Advances in Mobile Multimedia (MoMM2003),* volume 171, pages 25–35. Austrian Computer Society (OCG), September 2003.

should instead adapt to the user, offering implicit, intuitive and sometimes invisible interfaces.

Our work strives to add another aspect to the vision of future computers: proactivity. We postulate that a PDA, which is not bounded to being a single physical device, can only fulfill its intentions if it acts proactively – good human assistants stand out for this reason. Our idea is to provide software applications not only with information about the current user context, but also with predictions of future user context. When equipped with various sensors, an information appliance should classify current situations and, based on those classes, learn the user's behaviors and habits by deriving knowledge from historical data. Our current research focus is to forecast future user context by extrapolating the past.

It should be pointed out that the topic of proactivity in computer science is a controversial one, especially in HCI; the general concept of context awareness itself has to be handled with care [8]. Because the estimated current or predicted future context and thus implicitly the actions started by the appliance based on these assumption might be erroneous, they might need to be reverted by the user, possibly causing severe problems. Thus, proactivity in applications, when utilized for controlling actuators with impact on the real world, must be handled with care. However, the possible uses for predicted user context in applications are manifold, ranging from simple reminder messages to emergency procedures being started before an accident happens. Our work is primarily concerned with techniques that enable proactivity in embedded devices and leaves decisions about starting actions to applications built on top of it.

In the following, we present our architecture for an application framework which provides predicted user context on the basis of historical data in real-time. The remainder of this article is structured as follows: Section 2 defines our notions of proactivity and context. Section 3 lists related work and sets our work in relation to other projects. The main part of this article, section 4, explains the architecture including our contribution of adding proactivity to context awareness. In section 5, our implementation of this architecture in form of a cross-platform software framework is discussed and current results are presented in section 6. Finally, section 7 shortly summarizes our work and describes future aims.

2. Definitions

2.1 Proactivity

The term proactivity has been used in computer science mostly for software agents, where one important difference between agent oriented programming and object oriented programming is the proactivity of software agents [40]. Formally, the difference between reactivity and proactivity lies in the dependence of the current system output on the system state trajectory. If interpreted as an abstract (Moore) state machine, the internal "state" of a system at time t can be described as $q_t = d\left(q_{t-1}, a_{t-1}\right)$, where q_t is the current state, q_{t-1} is the last state and a_{t-1} is the input value at time $t-1$ (cf. [29]). In this definition, system inputs and state transitions are assumed at discrete time steps $t \in \mathrm{N}_0$. The system output depends on the state – this is how the difference between reactivity and proactivity is defined in the context of this article. In a reactive system, the output b_t at time t only depends on the current and – implicitly – on past states:

$$b_t = l\left(q_t\right)$$

In a proactive system, it can also depend on predicted future states:

$$b_t = l\left\langle q_t, \overline{q}_{t+1}, \overline{q}_{t+2}, \ldots, \overline{q}_{t+m}\right\rangle$$

The future states $\overline{q}_{t+1}, \overline{q}_{t+2}, \ldots, \overline{q}_{t+m}$ for m discrete time steps are predicted recursively by some arbitrarily complex process $\overline{q}_{t+i} = p\left\langle q_t, \overline{q}_{t+1}, \ldots, \overline{q}_{t+i-1}\right\rangle$; if only q_t is necessary for predicting any $\overline{q}_{t+i}$, then p can simply ignore the predicted states $\overline{q}_{t+1}, \ldots, \overline{q}_{t+i-1}$.

2.2 Context

Context has been defined by Dey [7] as

> *any information that can be used to characterize the situation of an entity, where an entity can be a person, place, or a physical or computational object*

which we adopt in this article. A good overview on different definitions of context can be found in [34].

Describing the situation in general, context can have many aspects, typically:

- geographical
- physical
- organizational
- social
- user
- task
- action
- technological
- time

As described in more detail in section 4.1, a single sensor does not seem to be appropriate to capture the different aspects of context.

3. Related Work

Context awareness is currently a highly active research topic [2], but most publications assume few but powerful sensors like video or infrastructure based location-tracking. Albrecht Schmidt, Hans-Werner Gellersen and Kristof van Laerhoven have presented an architecture for recognizing context from multiple simple sensors[21][34][35][36] in the TEA and Smart-Its projects. Our work takes a comparable approach to context detection by using multiple diverse sensors, but extends it to also exploit qualitative, non-numerical features [24]. Additionally, our framework introduces the prediction of future context, which has not been considered in the TEA project. The ORESTEIA project is concerned with hybrid intelligent artifacts, but depends on a priori training of artifacts by a vendor in a special training phase and explicit retraining phases for adaptation [28]; we seek to avoid the distinction of operation and training phases so that a device can be fully operational at all times.

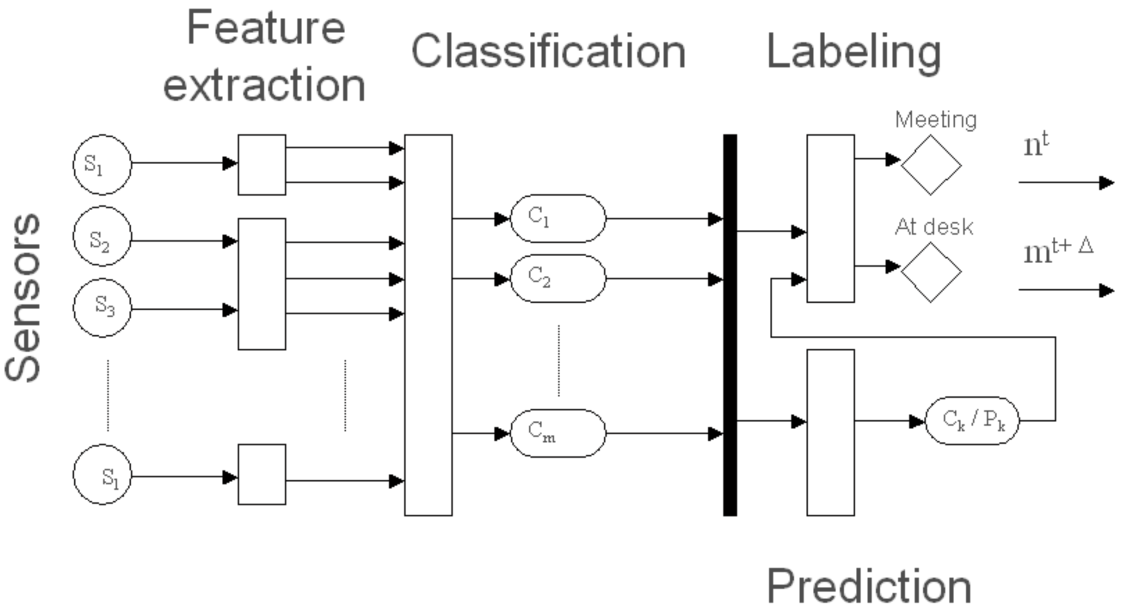
Feature extraction from different types of sensors has also been described in various publications in the field of context awareness, e.g. [4] describes the use of K-Means clustering and HMM to obtain context from a microphone, while [3] describes the use of audio and video for context detection, and others [5][17][23]. Our notion of features is equal to “cues” in the TEA project or to “Contextual Information Providers” in CIS [19]. In the field of robotics, feature extraction and sensor fusion have been studied extensively, but with a different focus. For autonomous robots, the geometrical properties of the environment (e.g. surfaces, angles, edges, color, textures, etc.) are of utter importance and need to be determined accurately to avoid potential collisions. Sensor fusion provides an appropriate means of combining multiple different sensors to resolve ambiguities, increase robustness due to redundancy and determine different properties of the same real-world objects [1]. For context awareness in information appliances, sensor fusion is at the current state of research not appropriate, because the available sensors typically capture different, mostly orthogonal aspects of the user or device context. Fusing of sensors necessitates some level of redundancy and a common model, which is currently not available for context descriptions. A possibility for exploiting multiple similar sensors, which can obviously be fused, is to exchange raw sensor or feature vectors between devices in spatial proximity. If two or more devices have similar sensors, their samples can be merged to obtain a possibly more complete view on the environment. This has been proposed independently in [23], similar to our recommendation in section 4.1.

In [7], Anind K. Dey et.al. described a software infrastructure for context awareness, which depends on a server for aggregating context and is limited to discrete-valued types of sensors. An implementation of this infrastructure is the Context Toolkit [32]. This toolkit is not directly comparable to our work; we aim to implement context recognition and prediction locally on each device, without the need for infrastructure components, while the Context Toolkit intentionally is an infrastructure approach.

Proactive adaptation of applications on the base of context has also been explored in [19]. The “Contextual Information Service” provides a lightweight interface for obtaining context information, but follows the approach of adapting the environment. We intend to autonomously adapt the device to a changing environment, which includes changing user behavior.

Learning user’s habits has previously been explored by Michael C. Mozer in The Neural Network House [25], which is able to predict occupancy of rooms, hot water usage and likelihood that a zone is entered in the next few seconds using trained feedforward neural networks. Kidd et.al. reported [20] about the Aware House, which should also learn user’s habits, but was not finished at the time of the report. The MavHome project [6] by Diana J. Cook et.al. also utilizes prediction algorithms to forecast user actions, but parts of the prediction seem to rely on database support and batch training.

Time series forecast has been explored in different areas, including distributed simulation [10], software agents [31], data value prediction in processors [33], data mining from health records [38] and theoretically for neural networks [37].



$$\langle s_1, s_2, ..., s_l \rangle_t \;\rightarrow\; \langle f_1, f_2, ..., f_n \rangle_t \;\rightarrow\; \langle c_1, c_2, ..., c_m \rangle_t \;\rightarrow\; \langle c_1, c_2, ..., c_m \rangle_{t+\Delta}$$

Input vector (Sensor vector) Feature vector Class vector Future class vector

Fig. 1 Architecure for proactivity via predicted user context

Utilizing different types of features for context recognition and the use of time series forecast methods for predicting future context on the level of aggregated context identifiers is, to the best of our knowledge, a new approach and has not been covered before by published research.

4. Architecture

Sensor readings are classified to detect common patterns in the input values. These patterns are interpreted as “states” of an abstract state machine that act as context identifiers. A user context is therefore abstracted to these states, whose internal data structures relate sensor readings to states. Although this interpretation makes it more complicated for applications to query for specific aspects of a context (e.g. location) instead of the context identifier, it allows to monitor and record the state trajectory of this abstract state machine. When a user advances from one context to another, sensor readings will change and another state will become active, reflecting the context change. Thus, interpreting the context changes as a state trajectory allows to forecast the future development of the trajectory, and therefore to predict the anticipated context. For clarity, we will only use the term context in the remainder of this article, which is similar to a state in our interpretation.

It is important to note that context classification and prediction must be performed in real-time for any practical application; it is not feasible to log data and process it offline. For the vision described in section 1, an information appliance will have to be continuously running and always be able to provide services to the user. Therefore, our architecture is targeted towards embedded systems, running without user intervention for arbitrarily long periods.

To derive knowledge about the device/user context from raw sensor data, the following steps are applied, which are depicted in Fig. 1:

1. Sensor data acquisition: Sensors, e.g. brightness, microphone or IEEE802.15 Bluetooth and IEEE802.11b Wireless LAN (*WLAN*) network interfaces, provide data streams (time series) of measurements. Usually some physical values like the incoming RF signals are the base for measurements, but more abstract sensors like the currently active application can also be utilized. In [34], sensors are classified as physical or logical, which we are not doing for a variety of reasons. Within our work, a sensor is any entity that can provide measurements of the environment a device is situated in. Values are sampled at discrete time steps $t \in \mathrm{N}_0$, whose frequency should be set to the maximum desired sample frequency of the used sensors. As this is highly application specific, no general distance between sample time steps can be determined.

A *sensor vector* $S_1 \times S_2 \times \ldots \times S_l$ defines sensor readings $\langle s_1, s_2, \ldots, s_l \rangle_t \in S_1 \times S_2 \times \ldots \times S_l$ for points in time t.

2. Feature Extraction: From raw sensor data, information can be extracted by domain-specific methods, yielding multiple data values per sensor, which are called *features* F with samples $f \in F$ as a function of time. During feature extraction, the available data is deliberately simplified, transformed or even expanded, allowing it to be interpreted better. Usually, simple statistical parameters like the mean $\bar{x}$, standard deviation s or higher moments are used as features for time series of numerical data. For nominal and ordinal data, alternative methods should be explored.

   The set of features is called a *feature vector* $F_1 \times F_2 \times \ldots \times F_n$, which defines samples $\langle f_1, f_2, \ldots, f_n \rangle_t \in F_1 \times F_2 \times \ldots \times F_n$ for points in time t in the multi-dimensional *feature space*.

   Although this definition does not allow for meta attributes on feature values, it might be advantageous to do so. In the CIS project [19], meta information like confidence or accuracy is added to feature values and can be used by applications. In the architecture presented in this article, a confidence value might be mapped to a weight in the classification step to adaptively weaken the influence of features with (currently) poor sampling quality.

3. Classification: The task of classification is to find common patterns in the feature space, which are called *classes* or *clusters*. Because a feature vector should possibly be assigned to multiple classes with certain *degrees of membership* (the "probability" or "confidence" that the feature vector belongs to a class), soft classification / soft clustering approaches are utilized. These approaches map a feature vector of n different features to degrees of membership $c \in C$ with $C := [0;1]$ of m different classes: $F_1 \times F_2 \times \ldots \times F_n \to C^m$. The classes c_i for $i = 1 \ldots m$ are regarded as the detected user context and are identified by a simple index in the class vector.

   The *class vector* C^m defines class degrees of membership $\langle c_1, c_2, \ldots, c_m \rangle_t \in C^m$ for points in time t.

4. Labeling: To ease the development of context-aware applications and for presenting detected context to the user, descriptive names should be assigned to single classes or combinations of classes (cf. [21]). Labeling maps class vectors to a set of names: $C^m \to N$ where N is a set of user-defined context names (strings).

   A *context name* or *context label* $n_t \in N$ describes the currently active context for points in time t.

5. Prediction: To enable proactivity, our approach is to forecast future context. Thus, the prediction step generates anticipated future class vectors from current ones: $C^m \times R^+ \times R^+ \to C^m$.

   A (future) class vector defines degrees of membership for each class: $\langle c_1, c_2, \ldots, c_m \rangle_s = p(\langle c_1, c_2, \ldots, c_m \rangle_t, t, s)$ for points in time t and s with $s > t$.

The following sections describe these five blocks in more detail.

4.1 Sensors

Context awareness of information appliances premises that they can rely on context-relevant information gathered by sensors. The acquired information should be as close to the user's world perception as possible [34]. Unlike sensing information in other domains (e.g. quality assurance, robotics, etc.), where the object of interest is explicitly investigated for the sake of accurate and highly reliable measurement reading, sensors for Pervasive Computing information appliances have to be less intrusive and ostensible. Furthermore, for collecting context information, varieties and events in the measured data are much more interesting than the actual sensor output; thus different techniques and methods are required [9]. Gellersen et.al. proposed the use of diverse simple sensors as an alternative to the integration of a single generic sensor. Presuming that current information appliances are already equipped with sensors that can be exploited for those means, this approach is more rational. The variety of different sensor types results in a better representation of the users context than a single generic sensor [13]. Examples of such sensors in a typical information appliance are listed in Table 1, while Table 2 lists additional sensors that might be useful for recognizing user context and can be easily added to current and future information appliances.

Table 1 Typical sensors available in a mobile device

time
microphone
brightness
Bluetooth
Wireless LAN
(un)docked
logged on(pff
application manager

Table 2 Additional sensors for a mobile device

GPS
GSM
compass
accelerometer
tilt sensor
temperature sensor
pressure sensor
various bio-medical sensors

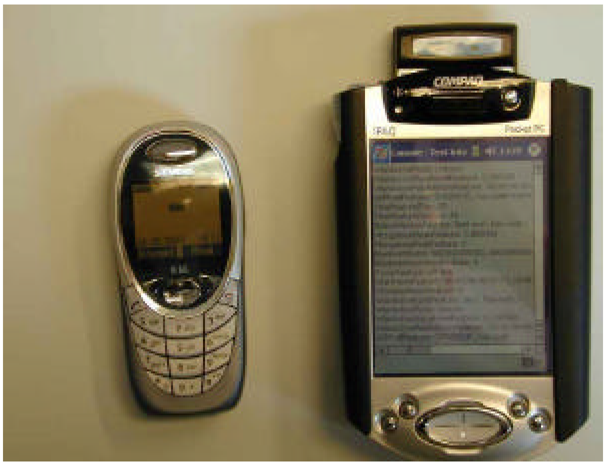
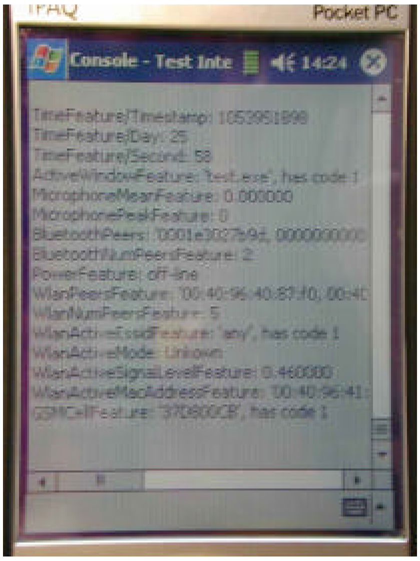


Fig. 2 Feature extraction on a typical PDA with a mobile phone as additional sensor



```
TimeFeature/Timestamp: 1053951898
TimeFeature/Day: 25
TimeFeature/Second: 58
ActiveWindowFeature: 'test.exe', has code 1
MicrophoneMeanFeature: 0.000000
MicrophonePeakFeature: 0
BluetoothPeers: '0001e3027b9d, 0000000000
BluetoothNumPeersFeature: 2
PowerFeature: off-line
WlanPeersFeature: '00:40:96:40:87:f0, 00:40
WlanNumPeersFeature: 5
WlanActiveEssidFeature: 'any', has code 1
WlanActiveMode: Unkown
WlanActiveSignalLevelFeature: 0.460000
WlanActiveMacAddressFeature: '00:40:96:41:
GSMCellFeature: '37D800CB', has code 1
```

Fig. 3 Feature extraction on a typical PDA with a mobile phone as additional sensor: example values

To improve the quality of context recognition, information appliances can share their perceptions to create a more complete model of the current context. This is accomplished by mutually granting access to the raw sensor data of devices in the neighborhood in a peer-to-peer manner. Own sensor data can be correlated with the data of sensors in range, increasing the accuracy. Nevertheless, context information can only be shared within a close range to ensure that the recognized context is still local and distinct and not a global representation of different neighboring situations. By this means, the list of sensors is easily extensible by equipping the user with smart sensors that expose their information to a close, interested (and authorized) device. E.g., a biological sensor could measure the user's pulse and transmit it to the information appliance via Bluetooth or similar ad-hoc communication methods. Fig. 2 shows an example of using a mobile phone for retrieving GSM sensor data via Bluetooth. The list of processed sensor information is only limited by processing capabilities and memory of the information appliance.

4.2 Feature Extraction

Although feature extraction and classification are well-known fields of research, most publications only cover numerical, continuous features. Recently we introduced a model for utilizing heterogeneous features (e.g. a list of Bluetooth or WLAN devices in range in combination with the time stamp) in a common classification step [24].

The feature vector $\left\langle f_1, f_2, \ldots, f_n \right\rangle$ formed by an arbitrary combination of these features is highly heterogeneous; therefore, it is necessary to find a way to cope with the different types and semantics of the feature space dimensions in the classification step. In our concept, a feature type is defined by the feature extractor that does the actual transformation of raw sensor data into the more relevant exposition of the data. Therefore, these transformations can be done independently for each feature and are completely domain specific; each feature type can implement its operators needed for classification differently. This abstraction virtually maps different kinds of sensor data and their respective feature types to a unified multidimensional, homogeneous feature space that can be classified by commonly used algorithms.

Feature types can be categorized as one of the following (a similar, albeit not identical taxonomy has been used in [26]):

- nominal (categorical, qualitative): The feature takes on values of a set on which no order relation has been or can be defined. A special case are binary features with $F = \{0,1\}$.

- ordinal (rank): The feature takes on a values of a set with a defined order relation $<: F \times F$.
- numerical (quantitative): The feature takes on values of an ordered set with defined + and • operations (an algebraic field). It can be further distinguished according to the density of values in the set:
  - discrete: $F \in Z$
  - continuous: $F \in R$
- interval: The feature takes on intervals instead of single values, e.g. $F \subseteq Pot(R)$.

A preliminary comparison of different classification methods, ranging from clustering algorithms to neural networks, showed that only two operations are necessary on an abstract feature F : a distance metric and an adaptation operator [24]. With these two operations, supervised and unsupervised classifiers like the Kohonen Self-Organizing Map (*SOM*), K-Means clustering or Lifelong GNG [16] can be easily applied to any feature which defines them. In Fig. 3, an example list of features in a typical mobile scenario is shown on the PDA screen, including lists of Bluetooth and WLAN devices in range, the current GSM cell (queried from a mobile phone via Bluetooth) and specific audio features from the microphone. Each of these features has been implemented with appropriate distance and modification operators.

4.3 Classification

The classification step is used to find similarities in and learn recurring patterns from its input data. It serves the input for the labeling and the prediction steps in form of a probability vector containing the probability of activity for each learned class.

A classification algorithm has to fulfill several requirements to be applicable for classifying sensor data and recognizing context:

- On-line learning: There is no dedicated training phase, learning has to be done unsupervised and continuously.
- Adaptivity: Learning must never stop; as user’s habits will change over time, classes must always adapt to new input data. This prevents the use of a continuously decreasing learning rate as used in many methods (e.g. some neural networks).
- Variable topology: Because the number of classes can not be determined a priori for the general case, the internal topology must be able to adapt dynamically.
- Soft classification: Context classes are not mutually exclusive, more than one context can be active at the same time (e.g. ’at home’ and ’sleeping’).
- Noise resistance: When working with real-world data, the algorithms have to cope with the intrinsic noise that is sampled with all signals.
- Limited resources: The algorithm has to work within the capability constraints of an information appliance (small RAM, little processing power, etc.).
- Simplicity: In our case, the algorithm should perform as few distinct operations on the feature vector as possible. As the feature extractors have to provide the necessary operators (see feature extraction), a multitude of operators drastically complicates the implementation.

Ideally, a classification algorithm must be on-line and thus unsupervised and must have a variable network topology to cope with changing feature vector dimensionality (changing sensor configurations). The classification algorithm must not be hard competitive to allow multiple active contexts and it has to be designed for life long learning to not forget or overwrite already learned clusters over time (which is known as the plasticity-stability dilemma [15] in neural networks and clustering literature). Table 3 shows a comparison of the most common on-line clustering algorithms and serves as a base for selecting the most appropriate one. K-Means, Leader, G K-Means and IDBSCAN segregate themselves due to their hard competitive classification strategies. SOM and RSOM tend to forget their previously learned classes very quickly due to their learning strategy and fixed network topology. Although this can be circumvented by combining the SOM with K-Means clustering [21], GNG [11] still seems to provide more flexibility in environments with changing configurations.

In [16], Hamker proposed modifications to the original GNG algorithm to cope with continuously changing environments and life-long learning (LLGNG). These modifications prevent the GNG from growing permanently by introducing a learning rate with a locality criteria. This results in a locally converging but globally still adaptive learning algorithm. New classes will always be learned but changes in already learned classes are only applied if the cluster representing this class does not match the new input vector properties to a reasonable extent. Due to these modifications, the algorithm also performs better in environments with small memory because a new cluster always represents a new context and is never redundant. Therefore, LLGNG can forget the oldest and most erroneous cluster when the memory boundaries are reached; this ensures that memory is always available for learning new classes. The basic rule behind learning and insertion in the LLGNG is that *"organisms only learn when events violate their expectations"*, previously assumed by [30]. Tests and performance evaluations are in work and will be presented in more detail in future publications.

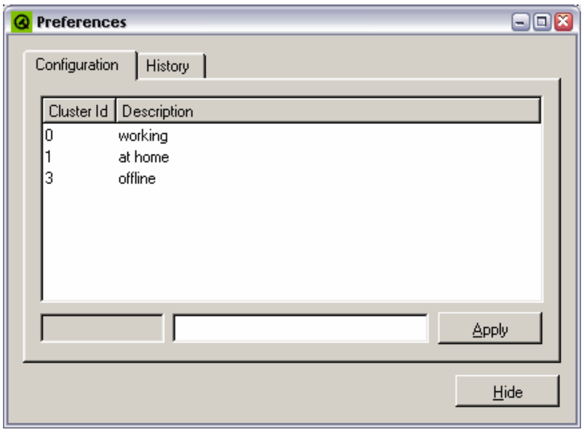
Table 3 General overview of algorithms for unsupervised clustering of sensor data, based on [21]: 1

On-line Algorithm	***Network Topology***	***Topology Preserving***	***Competitive***
SOM	fixed	yes	soft
RSOM	fixed	yes	soft
K-Means	fixed	no	hard
Leader	variable	no	hard
G K-Means	variable	no	hard
Neural Gas	variable	no	soft
NG+CHL	variable	yes	soft
GNG	variable	yes	soft
IDBSCAN	variable	no	hard

4.4 Labeling

Applications will generally be unaware of classes and their current degrees of membership. In real world scenarios, it would be virtually impossible to design applications to work with these class vectors, because they are learned in an unsupervised way and will therefore differ from one device to another; class vectors depend on the order in which those classes were detected first. Therefore, the indices of the currently active classes need to be mapped to more meaningful values. In our architecture, simple strings are used as context labels allowing users to enter them. This is the only step where user interaction is necessary and even in this case, it can be non-obtrusive. Approaches for such an interface include a discreet icon in one corner of the device screen which blinks when a still unlabeled context class has been active for some period of time, allowing a user to assign a name for the current context. Another option is to display an automatically created context log in form of a diary, which allows the user to label formerly detected context classes.

The complexity necessary for this step mostly depends on the quality of the classification step. If classes are long-term stable, i.e. previously learned classed are not overwritten by different new ones, then a simple 1-to-1 mapping of classes to labels might be enough. However, if the used classification algorithm overwrites older classes in order to learn new context, then the degrees of membership of all classes will need to mapped to labels. In [21], a simple K-Means clustering algorithm is used as a second step after clustering. For each class, represented by a winner neuron of the Kohonen SOM clustering, K-Means is applied to the input vectors of the SOM (which correspond to feature vectors in our architecture) to avoid overwriting of labels. This added complexity is necessary because of the shortcomings of the SOM, and one of the reasons for selecting GNG for our first experiments.



Cluster Id	Description
0	working
1	at home
3	offline

Fig. 4 GUI prototype: Allowing the user to assign descriptive labels

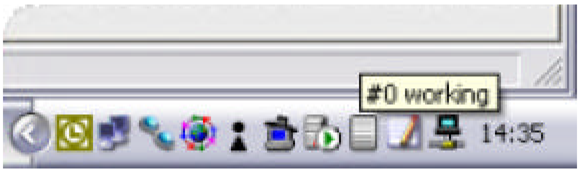


Fig. 5 GUI prototype: non-obtrusive display of current context

The first implementation is based on a direct 1-to-1 mapping and provides the user a non-obtrusive way to assign a label to a class. The prototype labelling application is currently realized independently from the framework to permit autonomous development and testing. The communication between the labelling application and the classification framework occurs via a simple SOAP protocol (a DCOM interface on windows is also available), not being restricted to a single platform. Fig. 4 shows the dialog used to assign labels to the classes found by the classification algorithm, Fig. 5 shows how the application signals the user the currently active context. It is also

[1] Kohonen Self-Organizing Map (*SOM*), the Recurrent Self-Organizing Map (*RSOM*), K-Means clustering, Hartigan's sequential leader clustering, growing K-means clustering, neural gas, neural gas with competitive Hebbian learning (*NG+CHL*), growing neural gas (*GNG*) and incremental DBSCAN (*IDBSCAN*). Unlike Van Laerhoven we rate NG+CHL and GNG as topology preserving [12].

possible to view the context history to assist the labelling of new contexts.

It is still an open issue if the classification quality will facilitate the use of a simple 1-to-1 mapping or if a more complex n-to-1 mapping from the whole class vector to labels will perform better. When an appropriate solution has been found for a wide range of application areas, the labeling will be incorporated directly into the framework. Thus, the labeling application will only be responsible for the naming of recognized contexts, but not for storing the associations or the context history.

4.5 Prediction

Prediction is the main novelty in our architecture and the focus of our current research. As prediction should again be performed without user interaction, it is not necessary to work with labeled context. Instead, the prediction step in our architecture builds upon the class vector generated by the classification step. This allows to predict more than a single future "best matching" context by exploiting the class degrees of membership (which would be impossible if the prediction would take the single "best matching", labeled context as input).

The aim is to generate class vectors for future points in time, which have the same meaning as the current class vector provided by the classification step. This allows to feed the predicted class vectors into the labeling step to provide predicted context labels for use in proactive applications (cf. Fig. 1). Before going into more detail, it is generally good to first analyze the requirements for a prediction algorithm in this sense.

- Unsupervised model estimation: Model topology and parameters need to be estimated automatically without user interaction or explicit definition of input/output behavior.

- On-line learning: For embedded devices in real-world scenarios, it is infeasible to switch between artificially separated training and prediction phases or even to store enough history for a batch training. Therefore, the algorithm has to continuously adapt its parameters during normal operation, incorporating new class vectors as soon as they arrive.

- An exception to this is to store only the recent history in detail, which could be used to optimize

- and/or evaluate the quality of the predictions (by splitting the history in a training and a test set).

- Incremental model growing: When new classes are detected in the classification step during run-time, new dimensions will be added to class vectors. The prediction algorithm must be able to incrementally increase its internal model topology without requiring a complete retraining, e.g. by initializing new dimensions with default values. It is currently unclear if shrinking of class vectors during run-time is also necessary or if "dead classes" could simply receive a minimum probability.

- Confidence estimation: The algorithm should be able to compute an estimation of the correctness of the forecasted context along with the forecast itself. This estimation can be used by the application as a confidence measure to determine if the prediction should be relied on for certain actions.

- Automatic feedback: The prediction engine should continuously estimate the next class vectors and evaluate its estimations by comparing with the real class vectors when they are available.

- Manual feedback: If some action that has been carried out automatically due to a forecast is reverted/canceled by the user, this forecast should receive a penalty to make it less likely the next time (this is known as reinforcement learning in machine learning) .

- Long-term vs. short-term: The used method should ideally be suitable (e.g. parameterizable) for different forecasting horizons, i.e. predicting context in the near future with high confidence, but also being able to predict later context, most probably with lower confidence.

We have currently not selected a specific algorithm for the prediction step because our architecture is open for arbitrary algorithms that can be adapted to suit our interface. However, after first research on possible candidates, Markov predictors seem to be generally suited well. Active LeZi [14] has already been implemented as predictor plug-in within the framework and is currently used for initial experiments on real-world data. Although prediction accuracy on simple artificial data sets looks promising, the results with real-world data suggest further research. Another special form of Markov predictors, the Variable Duration Hidden Markov Models (VDHMM) also seem to be applicable for predicting context trajectories. They explicitly model the duration distributions and are thus capable of predicting for different forecasting horizons and, as for nearly all variants of HMMs, there exist mature methods for learning model parameters. It might be necessary to use multiple different predictor plug-ins concurrently and fuse their results to generate a

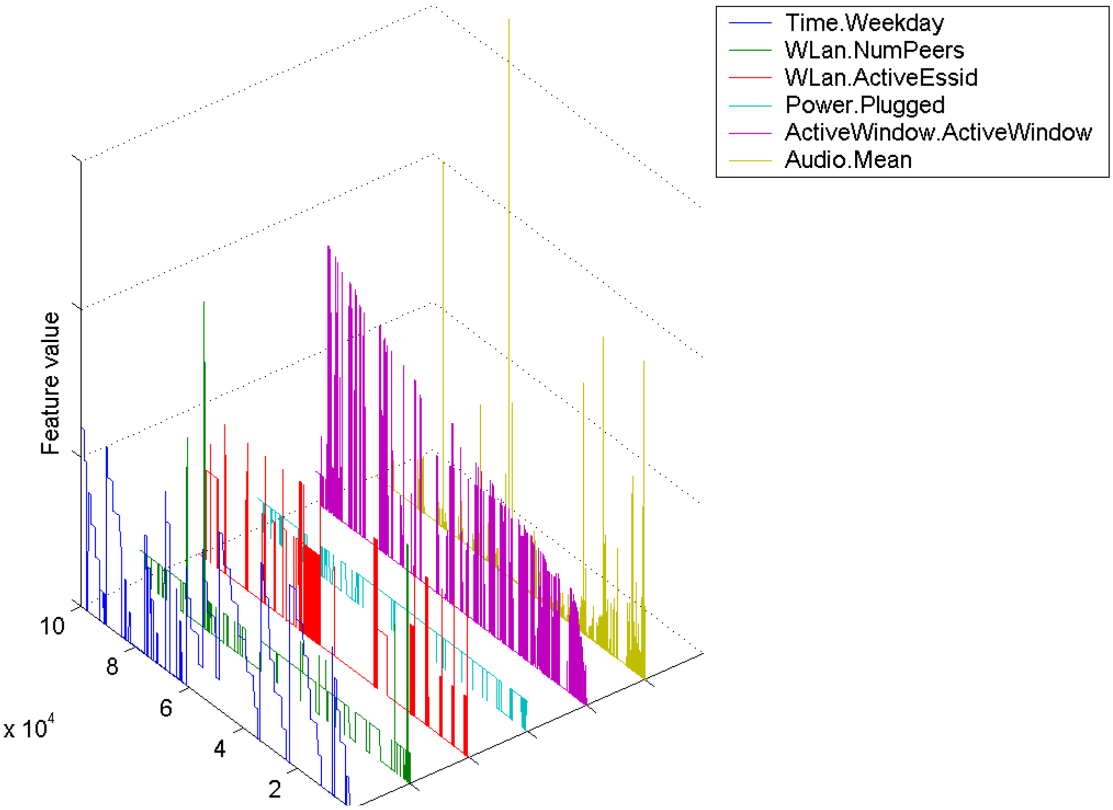


Fig. 6 Feature value trajectories of the complete data set

reasonable forecast of the context trajectory. A single prediction method is usually unable to respect trends and periodic patterns in the context history as well as performing sequential Prediction by Partial Match (*PPM*). For combining the results of different predictors, it is important that a confidence estimation is generated by the method.

5. Implementation

The described architecture has been implemented in form of a cross-platform framework which is freely available under an open source license. Interfaces provided by the framework are implemented by adhering to standardised protocols like SOAP and DCOM to guarantee most flexibility when developing context aware third party applications.

Currently it runs under Windows 2000 or XP, Linux on IA32 and ARM processor platforms, Windows CE 3.0 and, with restrictions, under Symbian OS 7.0.

6. Experiments and Evaluation

We have evaluated our framework for context recognition and prediction on two real-world data sets. The first data set has been gathered over 3 weeks on one of the servers for our smart room and includes the list and number of Bluetooth devices in the room and the list and number of Wireless LAN clients in the room. For the second data set, a broader range of sensors has been used. On a standard notebook computer, which is used for daily work, the following features were recorded over a period of about 2 months with over 100000 samples: weekday, active window (i.e. active application), mean environmental loudness, plugged into charger, WLAN ESSID, WLAN mode, WLAN signal level, WLAN access point MAC address, Bluetooth peers in range, number of Bluetooth peers in range and the GSM cell ID of the mobile phone (which was connected via Bluetooth). Fig. 6 shows the trajectory of 6 out of these 11 feature values over the whole recording period – the other features are not shown because they either yield non-atomic values or do not contribute significantly to context recognition.

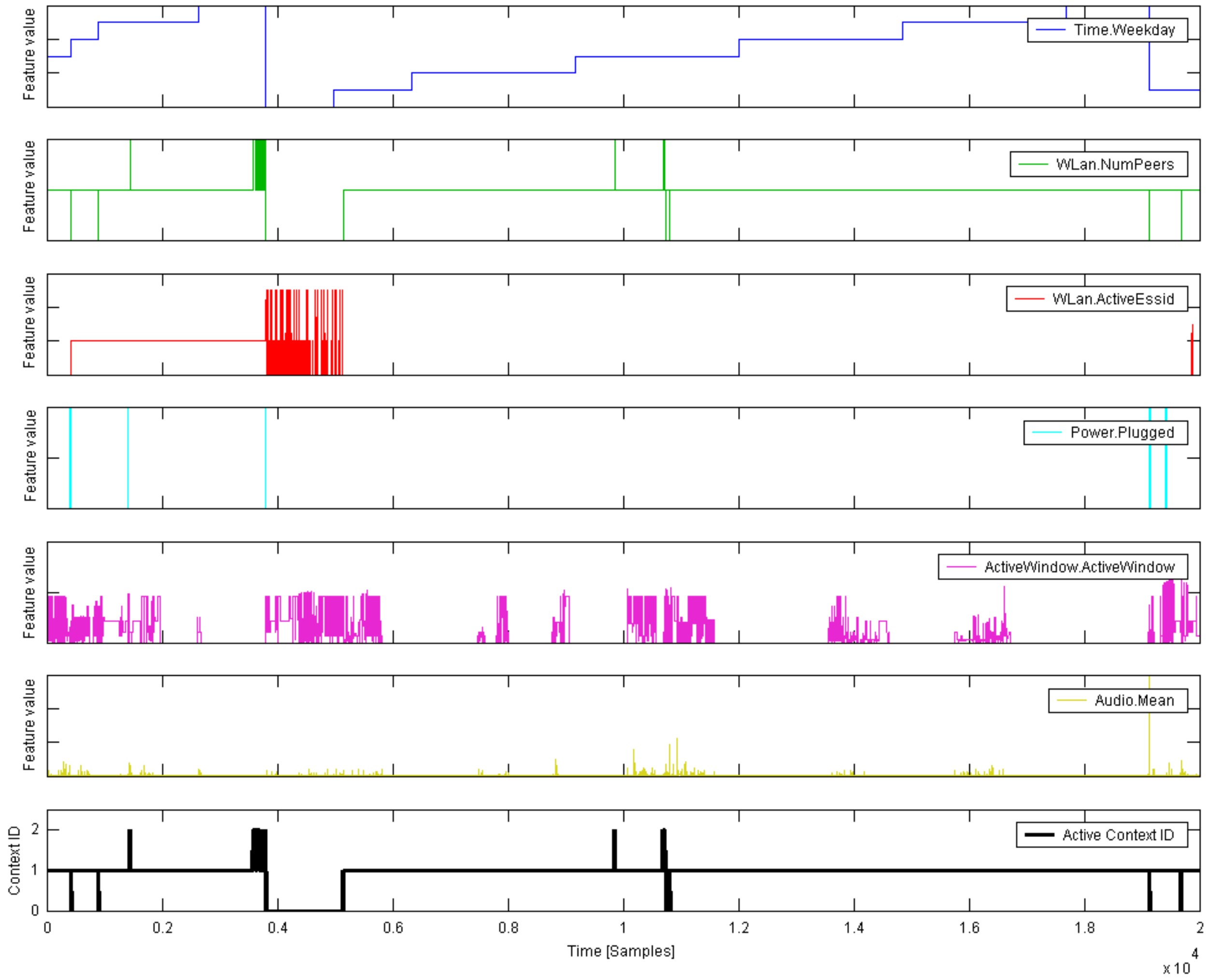


Fig. 7 Recognized context identifiers

In Fig. 7, a part of the data set over roughly 2 weeks is shown in more detail. The first 6 plots depict the feature values, while the last one shows the respective context ID with highest activation after the classification step. As can be seen, the classification algorithm selected 3 different contexts during this time frame.

7. Conclusions and Future Work

We have presented an architecture to recognize and predict user context by utilizing multiple heterogeneous sensors. This architecture consists of four steps: feature extraction (to generate a more relevant representation of sensor data, exploiting domain-specific knowledge), classification (to find similarities and common patterns in the input data), labeling (to assign simple context names to recognized classes) and prediction (to forecast future user context based on past behaviors). The novelties in this approach are the prediction of possible user actions via context forecast and the abstraction of feature types to allow heterogeneous features to be combined in a single classification step. To accomplish this, all feature types independently define the operations necessary for classification.

We have already implemented feature extraction for various sensors available on typical information appliances, including microphone, Bluetooth, Wireless LAN and additional, external sensors like a mobile phone accessible via Bluetooth. A next step in research will include gathering real-world data in an empirical study and evaluating classification and prediction algorithms based on this data.

Proactivity in applications can support users by allowing information appliances to adapt to the user instead of forcing the user to learn specifics of the interface. When equipped with multiple sensors and using those sensors to detect and predict context, an information appliance can become smarter and more intuitive to use, fostering a wider acceptance of information appliances in everyday life.

Acknowledgments

We especially thank Manfred Hechinger and Günther Blaschek for helpful discussions on the general topic of proactivity and the implications on user interfaces.

References

[1] S.B.H. Bruder. An information centric approach to heterogeneous multi-sensor integration for robotic applications. *Robotics and Autonomous Systems*, 26(4), March 1999.

[2] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.

[3] B. Clarkson, K. Mase, and A. Pentland. Recognizing user context via wearable sensors. In *ISWC*, pages 69–76, 2000.

[4] B. Clarkson, N. Sawhney, and A. Pentland. Auditory context awareness via wearable computing. In *Proceedings of the 1998 Workshop on Perceptual User Interfaces (PUI'98)*, San Francisco, CA, USA, November 1998.

[5] B. Clarkson and A. Pentland. Unsupervised clustering of ambulatory audio and video. Technical Report 471, MIT Media Lab, Perceptual Computing Group, 1998.

[6] D. J. Cook, M. Youngblood III, E. O. Heiermann, K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja. MavHome: An agent-based smart home. In *First IEEE International Conference on Pervasive Computing and Communications (PerCom'03)*, pages 521–524, IEEE Computer Society Press, March 2003.

[7] A. Dey, G. D. Abowd, and D. Salber. A context-based infrastructure for smart environments, 1999.

[8] T. Erickson. Some problems with the notion of context-aware computing. *Communications of the ACM*, 45(2):102–104, ACM Press , February 2002.

[9] D. Estrin, D. Culler, K. Pister, and G. Sukhatme. Connecting the physical world with pervasive networks. *IEEE Pervasive Computing*, 1(1):59–69, January–March 2002.

[10] A. Ferscha. Adaptive time warp simulation of timed petri nets. *IEEE Transactions on Software Engineering*, 25(2):237–257, March/April 1999.

[11] B. Fritzke. A growing neural gas network learns topologies. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 625–632. MIT Press, Cambridge MA, 1995.

[12] B. Fritzke. Some competitive learning methods. Technical report, Systems Biophysics, Inst. for Neural Comp., Ruhr-Universität Bochum, April 1997.

[13] H.W. Gellersen, A. Schmidt, and M. Beigl. Multi-sensor context-awareness in mobile devices and smart artefacts. *Mobile Networks and Applications*, 7(5):341-351. ACM Press, October 2002.

[14] K. Gopalratnam and D. J. Cook. Active LeZi: An incremental parsing algorithm for sequential prediction. *Proceedings of the Florida Artificial Intelligence Research Symposium*, 2003.

[15] S. Grossberg. Adaptive pattern classification and universal recoding: Parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23:121–134, 1976.

[16] F. H. Hamker. Life-long learning cell structures—continuously learning without catastrophic interference. *Neural Networks*, 14(4–5):551–573, May 2001.

[17] R. Headon. Movement awareness for a sentient environment. In *First IEEE International Conference on Pervasive Computing and Communications (PerCom'03)*, pages 99-106, IEEE Computer Society Press, March 2003.

[18] H. Ishii and B. Ullmer. Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proceedings of Conference on Human Factors in Computing Systems (CHI '97)*, pages 234–241. ACM, March 1997.

[19] G. Judd and P. Steenkiste. Providing contextual information to pervasive computing applications. In *First IEEE International Conference on Pervasive Computing and Communications (PerCom'03)*, pages 133-142, IEEE Computer Society Press, March 2003.

[20] C. D. Kidd, R. Orr, G. D. Abowd, C. G. Atkeson, I. A. Essa, B. MacIntyre, E. D. Mynatt, T. Starner, and W. Newstetter. The aware home: A living laboratory for ubiquitous computing research. In *Proceedings of the Cooperative Buildings, Integrating Information, Organization, and Architecture, Second International Workshop, CoBuild'99*, volume 1670 of *Lecture Notes in Computer Science*, pages 191–198. Springer, 1999.

[21] K. Van Laerhoven and S. Lowette. Real-time analysis of data from many sensors with neural networks. In *Proceedings of the fourth International Symposium on Wearable Computers (ISWC) Zurich, 7-9 October 2001*. IEEE Press, 2001.

[22] S. Mann. Wearable computing: A first step toward personal imaging. *IEEE Computer*, 30(2), February 1997.

[23] J. Mäntyjärvi, J. Himberg, and P. Huuskonen. Collaborative context recognition for handheld devices. In *First IEEE International Conference on Pervasive Computing and Communications (PerCom'03)*, pages 161-168, IEEE Computer Society Press, March 2003.

[24] R. Mayrhofer, H. Radi, and A. Ferscha. Feature extraction in wireless personal and local area networks. In *Proceedings of The Fifth IFIP TC6 International Conference on Mobile and Wireless Communications Network (MWCN 2003)*. World Scientific, October 2003.

[25] M. C. Mozer. The neural network house: An environment that adapts to its inhabitants. In *Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments*, pages 110–114. AAAI Press, 1998.

[26] S. Negri and L. Belanche. Heterogeneous kohonen networks. In José Mira and Alberto Prieto, editors, *Proceedings of the 6th International Work-Conference on Artificial and Natural Neural Networks (IWANN 2001)*, Lecture Notes in Computer Science, pages 243–252. Springer, 2001.

[27] D. A. Norman. *The invisible computer*. MIT Press, Cambridge, 1998.

[28] Deliverable D05: 1st year progress report of the Oresteia project. Technical report, January 2002.

[29] F. Pichler. *Mathematische Systemtheorie: Dynamische Konstruktionen*. Walter de Gruyter, Berlin, New York, 1975.

[30] R. A. Rescorla and A. R. Wagner. A theory of pavlovian conditioning: Variations in the effectiveness of reinforcement and non-reinforcement. In A. H. Black and W. F. Prokasy, editors, *Classical Conditioning II. Current Research and Theory*. Appleton Century Crofts, New York, 1972.

[31] M. T. Rosenstein and P. R. Cohen. Concepts from time series. In *AAAI/IAAI*, pages 739–745, 1998.

[32] D. Salber, A. K. Dey, and G. D. Abowd. The context toolkit: Aiding the development of context-enabled applications. In *Proceedings of the 1999 Conference on Human Factors in Computing Systems (CHI'99)*, pages 434–441, May 1999.

[33] Y. Sazeides and J. E. Smith. The predictability of data values. In *International Symposium on Microarchitecture*, pages 248–258, 1997.

[34] A. Schmidt. *Ubiquitous Computing – Computing in Context*. PhD thesis, Lancaster University, November 2002.

[35] A. Schmidt and M. Beigl. There is more to context than location: Environment sensing technologies for adaptive mobile user interfaces. In *Workshop on Interactive Applications of Mobile Computing IMC'98*, 1998.

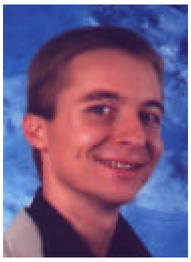
[36] A. Schmidt and K. Van Laerhoven. How to build smart appliances. *IEEE Personal Communications*, August 2001:66–71, 2001.

[37] Z. Tang and P. A. Fishwick. Feed-forward neural nets as models for time series forecasting. Technical Report 91-008, University of Florida, 1993. also published in ORSA Journal of Computing 5(4), 374–386.

[38] M. A. Orgun, W. Lin and G. J. Williams. Multilevels hidden markov models for temporal data mining. In *Proceedings of the KDD 2001 Workshop on Temporal Data Mining (held in conjunction with the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2001))*, San Francisco, CA, USA, August 2001.

[39] M. Weiser. The computer of the twenty-first century. *Scientific American*, 1496:94–100, September 1991.

[40] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. HTTP://www.doc.mmu.ac.uk/STAFF/mike/ker95/ker95-html.h (Hypertext version of Knowledge Engineering Review paper), 1994.



Rene Mayrhofer received the Dipl.-Ing. degree in Computer Science from the Johannes Kepler Universität Linz, Austria in 2002 and is currently working on his PhD on context prediction, also at Johannes Kepler Universität Linz. His research interests include context awareness, embedded systems and artificial intelligence. He is a contributing developer of the Debian GNU/Linux project.

Harald Radi is currently working on his Dipl-Ing. degree in Computer Science at Johannes Kepler Universität Linz, Austria. His research interests include component technologies, artificial intelligence and context awareness. He is a member of the PHP development team and contributor to the PEAR repository.

Alois Ferscha received his Mag. degree in 1984, and a PhD degree in business informatics in 1990, both from the University of Vienna, Austria. From 1986 through 2000 he was with the Department of Applied Computer Science at the University of Vienna at the levels of assistant and associate professor. In 2000 he joined the University of Linz as full professor where he is now head of the Institute for Pervasive Computing. He has served on the committees of several conferences like Pervasive Computing, UMBICOMP, WWW, PADS, DIS-RT, SIGMETRICS, MASCOTS, TOOLS, PNPM, ICS, etc. and is currently the program chair for the PERVASIVE 2004 conference.

2 Feature Extraction in Wireless Personal and Local Area Networks

Authors: Rene Mayrhofer, Harald Radi, and Alois Ferscha

Published in the proceedings of MWCN 2003, the 5th International Conference on Mobile and Wireless Communications Networks, pp. 195–198, World Scientific
ISBN 981-238-686-6
October 2003

FEATURE EXTRACTION IN WIRELESS PERSONAL AND LOCAL AREA NETWORKS

RENE MAYRHOFER, HARALD RADI and ALOIS FERSCHA
Institut für Praktische Informatik, Johannes Kepler University Linz, Altenberger Str. 69, 4040 Linz, Austria
Email: {mayrhofer,radi,ferscha}@soft.uni-linz.ac.at

Context awareness is currently being investigated for applications in different application areas of mobile computing. The integration of Bluetooth and Wireless LAN technologies into a vast of mobile devices — ranging from smartphones and PDAs to portable computers – has made user context sensing based on those technologies a feasible and promising approach. In this paper, we study which of the Bluetooth and Wireless LAN technology features (like radio-signal strength, device address management, etc.) can be exploited to derive user context, and develop a procedure how low level sensor data can be brought to application level context information. We introduce a method to automatically classify heterogeneous sensor data features with supervised or un-supervised classification methods. By defining two operations, a distance metric and an adaptation operator, any feature can be used as input for the classifier and can thus contribute to context detection.

1 Introduction

Common to most of the visions for next-generation computing are the paradigms of mobile computing and context awareness;[7] additionally, they all agree that user interfaces should become less obtrusive and "smarter" with regards to adapting to the user. Our approach is to add context awareness to applications, allowing them to adapt to the situation (*context*) at hand. One of the more recent definitions of context by Dey[1] is *any information that can be used to characterize the situation of an entity, where an entity can be a person, place, or a physical or computational object*, which we adopt in this paper. The goal is to derive high level context information from low level sensor data by following a three-step approach: data acquisition, feature extraction and classification. Starting from low level sensor data based on Bluetooth and Wireless LAN (*WLAN*), appropriate features (e.g. SNR, MAC addresses in proximity, access points in range, ESSID) are extracted and classified to yield high level user context information (e.g. busy, traveling, in a meeting, in the office, at lunch, at home, telephoning, etc.).

However, these sensors do not yield single, numerical values but data with a more complex structure. Some important information that can be extracted from these sensors is categorical and non-atomic, e.g. the list of MAC addresses which are currently in communication range. Nonetheless, it could provide useful information for determining the current user context via an automatic classification of all available sensor signals. If non-atomic values should be used as input to a classification algorithm which can only work with numerical inputs, they need to be coded. The standard procedure for dealing with categorical data is to code each possible sensor value as binary input to the classification algorithm. This has been applied successfully to categorical data with a bounded set of values (e.g. department), but is problematic when the set of possible values is not known in advance or too large to be coded with binary inputs (e.g. WLAN MAC addresses would need 2^{48} input dimensions to cover all possible values). Therefore, coding categorical data as numerical inputs does not seem to be the best solution and a better method should be found.

2 Related Work

In this paper, we concentrate on feature extraction from wireless network interfaces — an overview of our research on context awareness and prediction of user context has been presented elsewhere.[5,2] Although feature extraction and classification are well-known fields of research, most publications only cover numerical, continuous features. In the field of context awareness of mobile devices, Van Laerhoven described a system with custom sensors (e.g. accelerometers, light sensors), using minimum, maximum, mean and variance as features.[4] Because the restriction to numerical features severely limits the choice of sensors, we introduce a model for utilizing heterogeneous features (e.g. list of MAC addresses and WLAN ESSID) in a common classification step. For a Kohonen SOM, a method has been described to cope with heterogeneous input values.[6] However, the general case of context detection with heterogeneous features does not seem to have been covered before and is the main focus of our work.

3 Concept

To derive knowledge about the device/user context from raw sensor data, we follow the following steps:

1. Sensor data acquisition: Sensors provide data streams (time series) of measurements. Usually some physical values like RF signals are the base for measurements, but more abstract sensors like the currently active application can also be utilized. Besides wireless network interfaces, other common sensors available on typical mobile devices include a microphone, a brightness sensor (for automatic control of display brightness) and information about being connected to the docking station. Additional sensors like GPS, GSM, compass, accelerometer, tilt, temperature or pressure sensors can easily be added by connecting them via wire or Bluetooth.

2. Feature Extraction: From raw sensor data, information can be extracted by domain-specific methods, yielding multiple condensed data values, which are called *features* F with samples $f \in F$. During feature extraction, the available data is transformed, allowing it to be interpreted better. Usually, simple statistical parameters like the mean $\bar{x}$, standard deviation σ or higher moments are used as features for time series. For wireless networks, special features like the signal strength, the current WLAN ESSID or the list of access points in range are more appropriate.

3. Classification: Several features extracted in the previous step together compose a *feature vector* $F_1 \times F_2 \times ... \times F_n$, which defines points $\langle f_1, ..., f_n \rangle \in F_1 \times F_2 \times ... \times F_n$ in the multi-dimensional *feature space*. The goal of classification is to find common patterns in the feature space, which are called *classes* or *clusters*. Because a feature vector should possibly be assigned to multiple classes with certain *degrees of membership* (the "probability" that the feature vector belongs to a class), we utilize soft classification / soft clustering approaches. These approaches can be defined as mapping a feature vector of n different features to degrees of membership $u \in [0; 1]$ for m different classes: $F_1 \times F_2 \times ... \times F_n \to [0; 1]^m$

4 Feature Extraction

This paper concentrates on the second step and its interface to the third. Based on the specific feature extractor, a feature's class is, within this paper, defined as one of the following:

- nominal (categorical, qualitative): set of values on which no order relation has been defined. A special case are binary features with $F = \{0, 1\}$.
- ordinal (rank): set of values with a defined order relation.
- numerical (quantitative): ordered set of values with defined $+$ and $\cdot$ operations (an algebraic field). We can further distinguish according to the density of values in the set between discrete ($F \subseteq \mathbb{Z}$) and continuous ($F \subseteq \mathbb{R}$) features.
- interval: intervals instead of single values, e.g. $F \subseteq \operatorname{Pot} \mathbb{R}$.

When deriving context from wireless network interface data, we can identify a number of features from different categories, which seem to be relevant for mobile devices:

- Bluetooth: list of MAC addresses in range (nominal), number of MAC addresses in range (numerical/discrete)
- WLAN: list of access point MAC addresses in range (nominal), number of access point MAC addresses in range (numerical/discrete), current ESSID (nominal), associated to access point (binary), access point MAC address associated to (nominal), signal strength (numerical/continuous), transmission rate (numerical/discrete)
- GSM: list of GSM cells in range (nominal), number of GSM cells in range (numerical/discrete), current provider (nominal), current GSM cell (nominal), signal strength (numerical/continuous)

This list is not exhaustive, but should provide a meaningful view of the wireless networks context around the mobile device.

The feature vector $\langle f_1, ..., f_n \rangle \in F_1 \times F_2 \times ... \times F_n$ formed by an arbitrary combination of these features is highly heterogeneous, making it necessary to cope with the different types and semantics of the feature space dimensions in the classification step. From our comparison of different classification methods, we concluded that only two operations are necessary on an abstract feature F (this matches the results of other research^6). The first is a similarity measure or *distance metric*, which has to be defined on every feature, i.e. on every dimension of the feature space.

$$\begin{aligned} d : F \times F \to [0;1] \\ \delta = d(f_1, f_2) \end{aligned}$$

defines the distance between two samples of the feature F, normalized to $[0;1]$; the normalization is not necessary, but eases the classification step. A general distance metric has to fulfill non-negativity, identity, commutativity (symmetry) and the triangle inequality. We would like to note that the range of values in F can change (increase) during runtime, thus the distance of two given samples can also change due to the normalization. Although this is no problem with the classification methods we are investigating, others might need modifications. Additionally, the second operator adapts/modifies a point (in one dimension) and is necessary for supervised and un-supervised learning.

$$\begin{aligned} \alpha : F \times F \times [0;1] \to F \\ f' = \alpha(f, g, a) \end{aligned}$$

modifies the sample $f \in F$ towards $g \in F$ by a learning/adaptation factor $a \in [0;1]$. With these two operators, supervised and un-supervised classifiers like the Kohonen Self-Organizing Map (*SOM*) or K-Means clustering can be applied to any feature which defines them.

In the following, we will give example definitions of both operators for a selection of the features listed above.

1. signal strength: The Bluetooth, WLAN or GSM signal strength is a numerical, continuous variable; therefore, we can apply the L1 (Manhattan) metric:

   $$d(f_1, f_2) := \frac{|f_1 - f_2|}{F_{max} - F_{min}}$$

   and

   $$\alpha(f, g, a) := f + (g - f) \cdot a$$

   for samples $f_1, f_2, f, g \in F$ with maximum and minimum values F_{max} and F_{min}.

2. associated to access point: This is a binary variable with values $f \in \{0, 1\}$; the distance operator can thus be simplified to the equality relation:

   $$d(f_1, f_2) := \begin{cases} 1 \; if \; f_1 = f_2 \\ 0 \; if \; f_1 \neq f_2 \end{cases}$$

   A simple adaptation operator could be defined as

   $$\alpha(f, g, a) := \begin{cases} f \; if \; a \leq 0.5 \\ g \; if \; a > 0.5 \end{cases}$$

   which only sets it to one or the other value, depending on the current value of the adaptation factor a. A better, although more complicated variant is to implement the operator with state so that it internally sums up a and only modifies the feature value after a reaches a certain threshold.

3. number of MAC addresses in range: The number of Bluetooth peers or WLAN access points in range is a numerical, discrete variable; we define

   $$d(f_1, f_2) := \frac{|f_1 - f_2|}{F_{max}}$$

   and

   $$\alpha(f, g, a) := \begin{cases} \lceil f + (g - f) \cdot a \rceil \; if \; f \geq g \\ \lfloor f + (g - f) \cdot a \rfloor \; if \; f < g \end{cases}$$

   for a number of already detected, different MAC addresses F_{max}.

4. list of MAC addresses in range: The list of Bluetooth peers or WLAN access points in range is a nominal, non-atomic variable. Due to the non-atomicity, there are various ways for coding; but each value $f \in F$ can be seen as a set of addresses. For a list of addresses, we apply the Hamming distance (the number of different addresses)

   $$d(f_1, f_2) := |(f_1 - f_2) \cup (f_2 - f_1)|$$

   when f_1 and f_2 are sets. The adaptation operator can be arbitrarily complex; our current operator changes, according to the adaptation factor a, a randomly selected fraction of the different addresses in f and g to the addresses in g by adding and/or removing addresses in f. For performance reasons, we chose to code the list of MAC addresses as a bit vector, where already seen addresses correspond to bit positions.

5 Classification

Using these definitions, arbitrary classification methods can be used to classify feature vectors and thus derive context from wireless sensor data. Because of the resource limitations of mobile devices, it is not possible to record all sensor data; thus, batch algorithms which iterate over the whole data set at once can not be applied. On-line algorithms incorporate each input sample as soon as it arrives from the sensors and are therefore suited better. Another issue for context recognition with mobile devices is that algorithm parameters must either be constant or self-adaptive during run-time; a continuously decreasing "learning rate" as it is used in many algorithms like the SOM during its learning phase prevents an algorithm from running constantly without interruptions. For embedded systems, a distinction between learning and recognition phases does not seem to be appropriate.

Currently, we use the Growing Neural Gas[3] (*GNG*) clustering algorithm for our experiments because it offers a number of advantages over other methods, notably an unlimited number of clusters, un-supervised classification and clusters with arbitrary shapes. A short comparison of suitable clustering algorithms shows that GNG seems to be a good choice for mobile devices.[5]

All of the features listed above contribute to the classification and can therefore increase the quality of the context detection. Every feature has an unique view of the environment, yielding additional information that others can not provide. The combination of the features not only provides spatial context (in terms of qualitative localization), but also other aspects like the number of people standing nearby (if they carry Bluetooth-capable devices, which is becoming common). This can be used to determine non-spatial context, e.g. to detect a meeting situation.

6 Conclusion

We have considered to exploit Bluetooth and WLAN technologies as sensors for deriving user context. Current generations of mobile devices like smartphones or PDAs are equipped with these technologies, making them commonly available. In our approach, low level sensor data is transformed to high level context information in three steps: data acquisition, feature extraction and classification. Because some extracted features are not numerical, but categorical or even non-atomic, standard approaches to code them for the classification step do not seem appropriate. Experimental data has been collected using Bluetooth and WLAN as spatial proximity sensors over 10 days and was analyzed with K-Means and SOM classification algorithms using standard input coding. The preliminary results of our experiments suggest the use of a different input coding, e.g. the approach presented in this paper. An interface for using our feature extraction code including the different distance metric and adaptation operations in Matlab is currently being created and will allow direct, quantitative comparisons with standard algorithms. For future work, we will concentrate on the labeling of context classes by the user and on prediction of future user context, which will allow the development of proactive applications.

References

1. A. Dey, G. D. Abowd, and D. Salber. A context-based infrastructure for smart environments, 1999.
2. Alois Ferscha. Contextware: Bridging physical and virtual worlds. In *Ada-Europe*, volume 2361 of *Lecture Notes in Computer Science*, pages 51–64. Springer, 2002.
3. B. Fritzke. A growing neural gas network learns topologies. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 625–632. MIT Press, Cambridge MA, 1995.
4. K. Van Laerhoven. Combining the Self-Organizing Map and K-Means clustering for on-line classification of sensor data. In *ICANN*, pages 464–469, 2001.
5. Rene Mayrhofer, Harald Radi, and Alois Ferscha. Recognizing and predicting context by learning from user behavior. In *Proceedings of The International Conference On Advances in Mobile Multimedia (MoMM2003)*. Austrian Computer Society (OCG), September 2003. *(To appear)*.
6. S. Negri and L. Belanche. Heterogeneous kohonen networks. In José Mira and Alberto Prieto, editors, *Proceedings of the 6th International Work-Conference on Artificial and Natural Neural Networks, IWANN 2001*, Lecture Notes in Computer Science, pages 243–252. Springer, 2001.
7. A. Schmidt and M. Beigl. There is more to context than location: Environment sensing technologies for adaptive mobile user interfaces. In *Workshop on Interactive Applications of Mobile Computing IMC'98*, 1998.

3 Extending the Growing Neural Gas Classifier for Context Recognition

Authors: Rene Mayrhofer and Harald Radi

Published in the proceedings of EUROCAST 2007, the 11th International Conference on Computer Aided Systems Theory, pp. 920–927, Springer-Verlag Lecture Notes in Computer Science volume 4739
ISBN 978-3-540-75866-2
February 2007

Extending the Growing Neural Gas Classifier for Context Recognition

Rene Mayrhofer[1] and Harald Radi[2]

[1] Lancaster University, Infolab21, South Drive, Lancaster, LA1 4WA, UK
rene@comp.lancs.ac.uk
[2] Tumpenweg 528, 5084 Grossgmain, AT
harald.radi@nme.at

Abstract. Context awareness is one of the building blocks of many applications in pervasive computing. Recognizing the current context of a user or device, that is, the situation in which some action happens, often requires dealing with data from different sensors, and thus different domains. The Growing Neural Gas algorithm is a classification algorithm especially designed for un-supervised learning of unknown input distributions; a variation, the Lifelong Growing Neural Gas (LLGNG), is well suited for arbitrary long periods of learning, as its internal parameters are self-adaptive. These features are ideal for automatically classifying sensor data to recognize user or device context. However, as most classification algorithms, in its standard form it is only suitable for numerical input data. Many sensors which are available on current information appliances are nominal or ordinal in type, making their use difficult. Additionally, the automatically created clusters are usually too fine-grained to distinguish user-context on an application level. This paper presents general and heuristic extensions to the LLGNG classifier which allow its direct application for context recognition. On a real-world data set with two months of heterogeneous data from different sensors, the extended LLGNG classifier compares favorably to k-means and SOM classifiers.

1 Introduction

Context Awareness, as a research topic, is concerned with the environment a user or device is situated in. Although its roots probably lie in robotics [1], the advantages of making applications in the fields of pervasive, ubiquitous or mobile computing aware of the user or device context are obvious: user interaction and application behavior can be adapted to the current situation, making devices and their applications easier to use and making more efficient use of the device resources.

Our approach to making devices context aware is based on three steps: sensor data acquisition, feature extraction and classification [2]. In these three steps, high level context information is inferred from low level sensor data. There are two key issues in this approach: the significance of acquired sensor data, and the use of domain-specific heuristics to extract appropriate features. A broad view on

R. Moreno-Díaz et al. (Eds.): EUROCAST 2007, LNCS 4739, pp. 920–927, 2007.
© Springer-Verlag Berlin Heidelberg 2007

the current context is necessary for many applications, and can only be derived with a multitude of sensors with ideally orthogonal views of the environment. Using multiple simple sensors and merging their data at the classification step allows to use different heuristics for each sensor domain, and thus profit from well-known methods in the specific areas.

Examples for simple sensors that are relevant to typical context-aware applications are Bluetooth or wireless LAN adapters, which can capture parts of the network aspect, the GSM cell ID, which defines a qualitative location aspect, or the name of the currently active application (or application part), which captures a part of the activity aspect of a user or device context. Other, more traditional sensors include microphones and accelerometers to capture the activity aspect [3], video cameras, or simple light and temperature sensors. The types of values provided by these sensors are very different. A method to cope with this heterogeneity of features has first been presented in [4] for the general case, independent of the used classification algorithm. This paper is concerned with necessary modifications to the Growing Neural Gas (*GNG*) classifier.

GNG has been introduced by Bernd Fritzke in 1995 [5] and shares a number of properties with the conceptually similar Dynamic Cell Structure algorithm, which has been developed independently by Jörg Bruske and Gerald Sommer [6]. In 1998, Fred Hamker extended GNG to support life-long learning, addressing the Stability-Plasticity Dilemma [7]; the resulting algorithm has also been called Lifelong Growing Neural Gas (*LLGNG*). The main difference between GNG and LLGNG is that the latter uses local error counters at each node to prevent unbounded insertion of new nodes, thus allowing on-line learning for arbitrary long periods.

The basic principle of GNG and LLGNG is to insert new nodes (clusters) based on local error, i.e. it inserts nodes where the input distribution is not well represented by clusters. For classification of features to recognize context, this is an important property because it ensures independence of the usually unknown input distribution. Additionally, edges are created between the "winner" node, which is closest to the presented sample (in this case a feature vector) and the second nearest one; other edges of the winner node are aged and removed after some maximum age. The resulting data structure is a cyclic, undirected, weighted graph of nodes and edges, which is continuously updated by competitive Hebbian learning. In LLGNG, nodes and edges are created and removed based on local criteria. For details on the insertion and removal criteria, we refer to [7].

2 Extensions to LLGNG

For using LLGNG for context recognition, we extend it in two areas:

2.1 Extension 1: Coping with Heterogeneous Features

To ease the implementation, we reduce the set of heterogeneous features to a minimal subset of abstract features providing meaningful implementations of the necessary two operations *getDistance* and *moveTowards* (see also [4]).

Implementations of these methods may be general for certain types of features, but will typically benefit from domain-specific heuristics. Knowledge about the respective sensing technology should be applied in the implementation of these two methods.

For a framework for context awareness and prediction [2] on mobile device, we found that many of the typical features can be reduced to a common set of basic features. Currently, we use base classes *AbstractString*, *AbstractStringList*, *NumericalContinuous*, *NumericalDiscrete*, and *Binary* for the actual implementations. The *NumericalContinous*, *NumericalDiscrete*, and *Binary* features implement *getDistance* and *moveTowards* as the Euclidean distance metric and thus need no special considerations.

On the other hand, the *AbstractString* feature serves as a base class for features that return a single-dimensional output value that can be represented as a string (e.g. WLAN SSID, MAC address, GSM cell ID, etc.). Although not the best solution, using the string representation as similarity measure for the feature values is still more meaningful than having no metric at all. For *getDistance*, we defined the Levenshtein distance (normalized to the longest encountered string) as distance metric. For *moveTowards*, our extended version of the Levenshtein algorithm also applies these operations with a given probability and returns a string that is somewhere between (in terms of our distance metric) the compared strings and represents the actual cluster position for this feature dimension.

The *AbstractStringList* feature serves as a base class for features that return a set of output values per sample point (e.g. list of peers, list of connected devices, etc.) that cannot be easily represented by a single string. Based on the idea of using the Levenshtein distance, we assign each list element a unique index and compose a bit vector from the string list. If a given string is present in the list, its corresponding bit in the bit vector is set. The distance metric for *getDistance* is then defined as the Hamming distance between the bit vectors of two given lists. Again, the *moveTowards* function can compose an intermediate bit vector that represents the actual cluster position.

Our extended version of LLGNG simply uses *getDistance* and *moveTowards* on each dimension instead of applying Euclidean metrics.

2.2 Extension 2: Meta Clusters

In the standard formulation of GNG and LLGNG, the edges in the internal graph are only used for three purposes:

- for adapting neighbors (adjacent nodes) of the winner
- for inserting a new node between the winner and its neighbor
- for removing nodes which have lost all edges due to edge aging

Additionally, the local insertion and removal criteria in LLGNG depend on the neighbors. As can be seen, the edges are not used for analyzing the graph structure itself.

One of the problems with using standard, unsupervised clustering algorithms for context recognition is that the automatically created clusters are usually too fine-grained for mapping them to high-level context information like “in a meeting” or “at home”; for defining simple context based rules on the application level, we would like to achieve this granularity. Therefore, we introduce the concept of meta clusters by explicitly using properties of the generated graph structure. This can be seen as a heuristic that is independent of the problem domain, but that depends on the internal (LL)GNG graph.

The (LL)GNG graph, after some learning time, usually consists of multiple components distributed over the cluster space. These components consist of two or more connected nodes and are perfect candidates for high-level context information, because they cover arbitrarily shaped areas in the high-dimensional, heterogeneous cluster space instead of only RBF-type shapes that single clusters cover. In our extended version, we assign each component a unique meta cluster ID and use this ID for mapping to high level context information. For performance reasons, the meta cluster IDs can not be recalculated after each step, but have to be updated during on-line learning. When starting with two adjacent nodes in the initialization phase, we simply assign the first meta cluster ID to this component and cache the ID in each node. During on-line learning, insertion and removal of edges will lead to the following cases:

- inserting a new node: Since a new node will only be inserted between two existing ones, its meta cluster ID is set to the ID of the connected nodes.
- inserting an edge between two nodes with the same ID: No change is necessary.
- inserting an edge between two nodes with different ID: Due to *merging* two components, one of the IDs will be used for the resulting component, overwriting the other one. When both IDs have already been assigned to high level context information, the merge is prevented by not inserting the edge.
- removing an edge: If the previously directly adjacent nodes are no longer connected via another path, a meta cluster *split* has occured and a new meta cluster ID must be allocated for one of the two components.

Normal adaptation of clusters has no influence on the graph structure and can thus be ignored for the handling of meta clusters. Further (performance) optimizations for meta cluster handling in our extension include a caching of meta cluster IDs and an incremental check if two nodes are still connected after removing an edge.

2.3 Performance Optimizations

Finding the nearest and second nearest cluster to a sample vector is a common task performed by (LL)GNG. Our chosen architecture allows different types of features in every dimension of the cluster space; therefore, comparisons have to be performed separately for every dimension and cannot be optimized. One option is to limit the amount of necessary comparisons to an absolute minimum. To accomplish this, we store every cluster in a splaytree sorted by their distance from the origin. A splaytree has the advantage that recently accessed nodes are kept on the very top of the tree and infrequently used nodes move towards the bottom. Assuming that samples aren't evenly distributed in the cluster space but accumulated, the nodes of interest are always on the top of the tree and the tree only has to be traversed completely if the sample data changes spontaneously.

Additional optimizations were done by caching results of internal computations for the insertion and removal criteria as defined in [7]. These helper variables are only recomputed when the node position is changed instead of each time a neighbor is queried. More details on these optimizations are presented in [2,8].

3 Evaluation

Our extended LLGNG algorithm has been compared to the more well-known classification algorithms k-means and Kohonen Self-Organizing Map (SOM) both with artificial and real-world data sets. Details on the comparison can be found in [2].

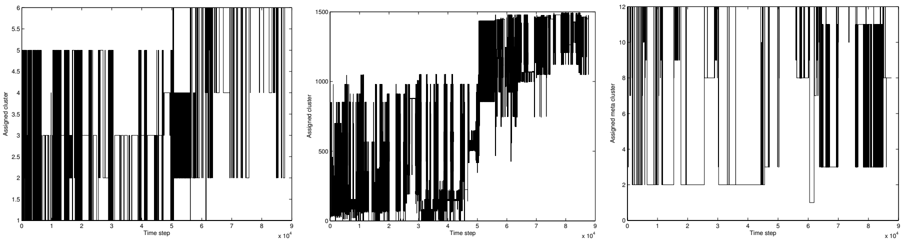
3.1 Data Set

The real-world data set used in this evaluation has been gathered continuously over a period of about two months on a standard notebook computer which was used for daily work. No special considerations were taken during the use of the notebook regarding context recognition. Therefore, the data set should provide representative sensor data for the chosen scenario. A wide range of sensors was used, including a microphone, the active/foreground window, if it was plugged into its charger, WLAN, and GSM. Domain-specific heuristics for these sensors are used to extract 28 different features. At one sample every 30 seconds, roughly 90000 samples of 28 dimensions were collected.

3.2 Pre-processing and Classification Error

Classification error was defined as the average distance between each data point and its respective best matching cluster after training has been completed, which is similar to the cost function minimized by k-means. This is a universal criterion suitable for evaluating the classification quality of arbitrary unsupervised clustering algorithms, and it is already well-defined and used in different implementations like the SOM Toolbox for Matlab; lower values for the classification error represent a better adaptation to the feature values and thus a more accurate classification.

Both k-means and SOM are used in a batch training mode and are thus not susceptible to initial transients, while LLGNG suffers from such effects due to its online mode. For k-means and SOM, the data set has been pre-processed to transform all non-numerical into numerical (binary) input dimensions by assigning one input dimension for each possible feature value, i.e. the one-of-C method. This transformation yields a 198 dimensional input space with a large number of binary inputs. All dimensions are further normalized to $[0;1]$, as recommended by standard literature on clustering. Since the implementations of the distance



(a) K-means (b) SOM (c) Extended LLGNG

Fig. 1. Cluster trajectories computed by the different classifiers

metrics specific to each feature are also normalized to $[0; 1]$, the overall classification error is assumed to be comparable even if the number of input dimensions is different.

3.3 K-Means

K-means clustering divides a given data set into k clusters by minimizing the sum of distances between the data points and cluster centers, whereas k has to be predetermined. Thus, k-means is actually not usable for live context classification, and is only used for comparison. The k-means clustering implementation in the Statistics Toolbox for Matlab was used iteratively for $k = 2 \ldots 40$ to determine the optimal number of clusters. With the optimum of 6 clusters, k-means reached a final classification error of 0.7451.

In Fig. 1a, the assigned clusters are depicted for each time step in the initial feature data set. The trajectory seems unstable and oscillates quickly between contexts. K-means clustering in this form is infeasible for embedded systems due to the enormous computational power necessary to optimize the number of clusters; determining the number of clusters for this test case took over 2 hours with 5 computers similar to our reference machine being used in parallel.

3.4 Kohonen Self-organizing Map

The Kohonen SOM is, in contrast to k-means, a soft clustering approach: each input sample is assigned to each cluster with a certain degree of membership. For the purpose of this comparison, this can easily be reduced to hard clustering by searching for the so-called "best matching unit", which is then assumed to be the context class for the respective time step. The following evaluations were performed with the specialized SOM Toolbox for Matlab developed by the Laboratory of Computer and Information Science at Helsinki University of Technology because of its flexibility and simple means of visualization.

Training a new SOM with the whole data set took 690 seconds and results in a final classification error of 0.5659. The SOM grid is automatically set to 71x21 clusters with a heuristic embedded in the toolbox. The u-matrix indicates around 4 to 8 larger areas, which seems reasonable when compared to the 6 clusters found by the k-means method and which can be seen as some form of meta clusters. Although the final classification error is lower than for k-means clustering, this is not surprising because of the significantly larger number of cluster prototypes that are available for mapping the input space. The large number of clusters formed by the SOM could not be used for directly representing high-level context, but a second classification step would be necessary.

The clusters assigned to the feature values for each time step are shown in Fig. 1b as the numbers of respective best matching units. The trajectory also shows oscillating behavior. Without a second step of clustering to exploit some form of meta clusters formed by the SOM, the resulting cluster trajectories seem unusable for context prediction on an abstract level. Additionally, the trajectory of the whole data set presented in Fig. 1b shows signs of unfavorable separation of clusters into areas in the input space around time step 50000: the apparent switch to a completely separate region of cluster prototypes is not supported by the visualization of the feature values. A change to different clusters around the time step 50000 is also visible in the k-means trajectory in Fig. 1a, but it is not as drastic as for the SOM.

3.5 Extended LLGNG

Prior to training the extended LLGNG algorithm with this test data set, a simulated annealing procedure was used to optimize some of its static parameters. However, the optimization does not significantly improve the classification error, suggesting stability against changes in the initial conditions or parameter sets and supporting the findings reported in [9].

One-pass training with the whole data set took only 474 seconds, produces 103 clusters composing 9 meta clusters and achieves a final classification error of only 0.0069. This means that the input data distribution is well approximated by the automatically found clusters, even with the inherent noise in our real-world data set and significantly less clusters than used by the SOM ($71 \cdot 21 = 1491$).

Fig. 1c shows the best matching meta clusters for each time step; the meta clusters are a higher-level concept than clusters and are thus suited better as abstract context identifiers. When comparing the trajectories computed by k-means, SOM, and the extended LLGNG variant, the latter one is more stable and shows fewer oscillations, with the notable exception of the time frame between time steps 64000 and 70000 where the LLGNG trajectory also shows oscillating behavior.

4 Conclusions

Lifelong Growing Neural Gas, a variant of the Growing Neural Gas classification algorithm, is an ideal algorithm for context recognition because it is optimized towards continuously running, un-supervised classification. In this paper, we presented necessary extensions for applying it to heterogeneous input data and for directly assigned high level context information to its output as well as performance optimizations. While k-means and the Kohonen Self-Organizing

Map (SOM) produced classification errors of 0.7451 and 0.5659, respectively, our extended LLGNG classifier achieved an error of only 0.0069.

It should be noted that, unlike SOM and k-means, the extended LLGNG is used in online mode, which is far more challenging. Without any further changes, the algorithm can immediately be used for continuous learning over arbitrary periods of time. K-means and SOM both had to be trained to produce the above results, with each sample being presented numerous times for training. The extended LLGNG only had a single pass over the whole data set and still achieves a significantly lower classification error and more stable results. One suspected reason for this success is that our extensions that enable LLGNG to deal directly with heterogeneous data indeed lead to higher classification quality due to the lower-dimensional input space and due to the preservation of the semantics of all feature dimensions.

References

1. Rosenstein, M.T., Cohen, P.R.: Continuous categories for a mobile robot. In: Proc. AAAI/IAAI: 16th National/11th Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence, pp. 634–640 (1999)
2. Mayrhofer, R.: An Architecture for Context Prediction. PhD thesis, Johannes Kepler University of Linz, Austria (October 2004)
3. Lukowicz, P., Ward, J.A., Junker, H., Stäger, M., Tröester, G., Atrash, A., Starner, T.: Recognizing workshop activity using body worn microphones and accelerometers. In: Ferscha, A., Mattern, F. (eds.) PERVASIVE 2004. LNCS, vol. 3001, pp. 18–32. Springer, Heidelberg (2004)
4. Mayrhofer, R., Radi, H., Ferscha, A.: Feature extraction in wireless personal and local area networks. In: Proc. MWCN 2003: 5th International Conference on Mobile and Wireless Communications Networks, October 2003, pp. 195–198. World Scientific, Singapore (2003)
5. Fritzke, B.: A growing neural gas network learns topologies. In: Advances in Neural Information Processing Systems, vol. 7, pp. 625–632. MIT Press, Cambridge MA (1995)
6. Bruske, J., Sommer, G.: Dynamic cell structure learns perfectly topology preserving map. Neural Computation 7, 845–865 (1995)
7. Hamker, F.H.: Life-long learning cell structures—continuously learning without catastrophic interference. Neural Networks 14(4–5), 551–573 (2001)
8. Radi, H.: Adding smartness to mobile devices - recognizing context by learning from user habits. Master's thesis, Johannes Kepler University Linz, Austria (2005)
9. Daszykowski, M., Walczak, B., Massart, D.L.: On the optimal partitioning of data with k-means, growing k-means, neural gas, and growing neural gas. Journal of Chemical Information and Computer Sciences 42(1), 1378–1389 (2002)

4 Security by Spatial Reference: Using Relative Positioning to Authenticate Devices for Spontaneous Interaction

Authors: Rene Mayrhofer, Hans Gellersen, and Mike Hazas

Published in the proceedings of Ubicomp 2007, the 9th International Conference on Ubiquitous Computing, pp. 199–216, Springer-Verlag Lecture Notes in Computer Science volume 4717
ISBN 978-3-540-74852-6
September 2007

Security by Spatial Reference: Using Relative Positioning to Authenticate Devices for Spontaneous Interaction

Rene Mayrhofer, Hans Gellersen, and Mike Hazas

Lancaster University, Computing Department, South Drive, Lancaster LA1 4WA, UK
{rene,hwg,hazas}@comp.lancs.ac.uk

Abstract. Spontaneous interaction is a desirable characteristic associated with mobile and ubiquitous computing. The aim is to enable users to connect their personal devices with devices encountered in their environment in order to take advantage of interaction opportunities in accordance with their situation. However, it is difficult to secure spontaneous interaction as this requires authentication of the encountered device, in the absence of any prior knowledge of the device. In this paper we present a method for establishing and securing spontaneous interactions on the basis of *spatial references* that capture the spatial relationship of the involved devices. Spatial references are obtained by accurate sensing of relative device positions, presented to the user for initiation of interactions, and used in a peer authentication protocol that exploits a novel mechanism for message transfer over ultrasound to ensures spatial authenticity of the sender.

1 Introduction

Spontaneous networking is of potentially great value to mobile users as it can enable them to associate their personal devices with devices encountered in their environment, and thereby to take advantage of serendipitous interaction opportunities. Spontaneous interaction in ubiquitous computing has for example been studied for applications such as social interaction and game-playing in mobile user communities. However, the potential of such interactions extends into areas that may involve more sensitive data and transactions, such as use of a vending machine over a wireless link, or direct payment transactions between two mobile devices. For such applications to be acceptable in a spontaneous network setting, a user must be able to authenticate the interaction of their personal device with the intended target device. They must be able to ascertain that the network entity their device connects to is identical with the physical device 'in front of them'. Furthermore, given the inherent vulnerability of a wireless communication channel, they must be able to rule out the presence of a third party established as 'man-in-the-middle' between their device and the target.

In a managed network environment, device-to-device authentication would be based on prior knowledge of each other or access to a trusted third party, but

J. Krumm et al. (Eds.): UbiComp 2007, LNCS 4717, pp. 199–216, 2007.
© Springer-Verlag Berlin Heidelberg 2007

in spontaneous networks neither can be assumed to be available. Instead it is necessary to provide an out-of-band mechanism alongside the wireless channel, for secure key exchange or verification of keys that have been 'speculatively' exchanged over the wireless channel. A wide range of mechanisms have been discussed in the literature, from user entry of PIN codes [1] and direct electrical contact [2] to use of communication channels with inherent physical limitations, such as infrared, audio and ultrasound [3,4].

In this paper we present a novel approach for device-to-device authentication in spontaneous networks. The main contribution is a method that uses *spatial references* to establish and authenticate interaction between a pair of devices. Spatial references capture the spatial relationship with a target device in terms of bearing and distance, and are used in an authentication protocol that couples key verification with verification of the relative position of the sender. The method and protocol are a general contribution in the sense that they can be implemented with any peer-to-peer sensing approach capable of providing accurate relative bearing and distance. However, we also contribute a concrete implementation, using a combination of radio frequency (RF) and ultrasonic (US) communication for measurement of spatial relationships.

As ultrasonic ranging is susceptible to certain attack scenarios (as we will explain in the course of the paper), we further contribute a novel coding technique for spatially-dependent message transfer over an ultrasonic channel. This technique allows a sender to transmit a message such that it can only be successfully decoded if it is received at a particular range. The technique is a key component in the protocol implementation we present, but can have wider application in ultrasonic systems independent of the particular problem we consider here.

In the subsequent section we will position our research with respect to related work, and then proceed to a description of the overall design of our method, the underlying sensing approach and the proposed user interface. This will be followed by a threat analysis, the description of a peer device authentication protocol as our core contribution, and an analysis of security and performance.

2 Related Work

Peer device authentication was first highlighted as a distinct security challenge emerging in ubiquitous computing by Stajano and Anderson, who proposed the 'Resurrecting Duckling' model for secure transient device association, bootstrapped from direct electrical contact [2]. Others have proposed channels for authentication that do not require direct contact but are 'location-limited' [3] or 'physically constrained' [4], including infrared beams [3], laser beams [5], and ultrasound [6]. Our method of spatial references effectively expands on the idea of location-limitation, using spatial measurements in addition to channel limitations, in order to further limit the position from which a device can successfully authenticate.

A variety of methods rely more on the user for device authentication, for instance for manual key entry [1], scanning of visual tags on the target and comparison with wirelessly received material [7] and verification of spoken messages generated by devices [8]. Our approach also has the user in the loop, however does not involve any user interaction *solely* targeted at security. Instead, we provide the user with a spatial technique for initiating interaction with another device; the spatial relationship is captured in this process and is then used for securing the interaction without need for further intervention of the user.

Our concrete implementation is based on the use of US as out-of-band channel. Kindberg et al. have before us proposed the use of US alongside an RF wireless channel in a protocol for validation and securing of spontaneous interaction [6]. The idea of the protocol is for devices to first exchange keys, and then to verify that the intended device is in possession of the correct key, by having the device send a nonce in plaintext over ultrasound and over RF. However, the protocol design does not consider potential attacks on the ultrasonic channel. A specific problem is the reliance on ultrasonic time-of-flight measurements for verification of device authenticity, as these involve synchronisation over the RF channel and are open to attack scenarios in which an attacker may appear nearer or further than they are [9]. As the protocol has not been implemented it is also not clear how precisely the nonce would be transmitted and what the security implications of this would be. In its general design, our protocol is similar to that of Kindberg et al., but we attend specifically to the issue of trustworthiness of ultrasonic ranging, and provide a complete implementation with security and performance analysis.

Other related work includes spatial interaction techniques. Hazas et al. [10], while not considering security, have presented an approach that uses ultrasonic peer-to-peer sensing for spatial discovery of other devices within interaction range, and work expanding on this has considered visualisation of the devices' positions in the user interface in order to ease interaction across devices (e.g. enabling transfer of a document to another device by a simple drag-and-drop operation) [11,12]. We employ the same principle in our method to let users initiate spontaneous interactions by means of spatial discovery and selection of the target device, but extend the approach by adding security in a seamless manner.

3 Security by Spatial Reference

Central to our method is the concept of *Spatial References.* A spatial reference captures the spatial relationship of a client device with a target device. A key aspect of spatial references is that they can be obtained independently by a user (seeing devices in front of them) and by their device (using sensors), and that a user can match what their device senses with what they see. Spatial references thus serve to establish shared context between a user and their device: a device can report a discovered network entity in a manner that the user can match with encountered devices, and a user can identify a target device in a way that their device can match with network entities.

3.1 Design of the Method

In our method for establishing and securing spontaneous interactions, spatial references are used for discovery of devices, for selection of a target devices, and for verification that interaction is secured between the 'right' devices. This involves the following steps:

1. The user's device uses a combination of network discovery and spatial sensing for *spatially-bounded discovery* of devices.
2. The spatially discovered devices perform spatial measurements to compute their relative positions.
3. Users are provided with a visualisation of available devices, integrated in the user interface of their personal device and laid out in correspondence with computed positions relative to the user's device.
4. Users initiate interaction and communication with a device by selection of the corresponding visual object, using direct manipulation techniques available in their user interface.
5. Selection of a device for spontaneous interaction triggers a protocol for key exchange with the target device and verification that no other devices can be present as 'man-in-the-middle' between the user's device and the target.
6. Once it has been asserted that exchanged keys are authentic, they are used for securing the communication channel between user device and target, and the initiated interaction can take place.

3.2 Spatial Discovery and Sensing

For a concrete implementation of spatial discovery and sensing we base our method on the *Relate* system for relative positioning introduced by Hazas et al. [10]. The Relate system provides wireless sensors implemented as USB dongles that can be readily used to extend host devices (such as laptops or PDAs) with spatial sensing. The Relate sensors contain three ultrasonic transducers (to cover space in front, left and right of the device) and they operate their own ad hoc network over combined radio frequency (RF) and ultrasound (US) channels (note this sensor network is separate from the wireless network that connects their host devices). Protocol functions implemented over the sensor network include network discovery and management, collaborative ultrasonic sensing, collection of measurements, and exchange of host information. The Relate sensors specifically support spatial discovery of their host devices by exchanging the hosts' network addresses over the sensor network.

The Relate sensors use RF messages to co-ordinate ultrasonic sensing. Sensing is performed by one node emitting ultrasound on its transducers, while all other nodes listen for a pulse on their transducers. The receiving sensors measure the peak signal values and the times-of-flight of the ultrasonic pulse with their three transducers. The smallest time-of-flight is used to calculate a distance estimate, and an angle-of-arrival estimate is derived from the relative spread of peak signal values measured across the transducers. The Relate sensors use

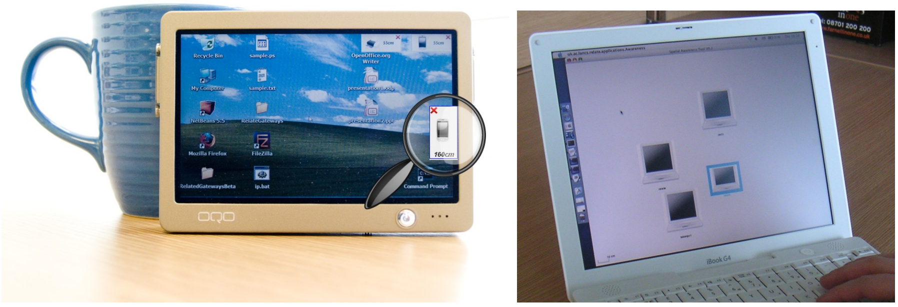


Fig. 1. Integration of spatial references to near-by devices in the mobile user interface; left: extension of Guinard et al.'s Gateways [12]; right: Kortuem et al.'s map view [11]

RF to share and collect sensor data, and each sensor provides the collected data to its host device. This then enables the host devices to compute their relative positions very accurately. Hazas et al. report a 90% precision around 8 cm in position and 25° in orientation [10]: these figures and our practical experience suggest sufficient accuracy for reliable disambiguation of devices. By collaboratively sharing US measurements over RF, partial obstruction can be dealt with in principle. However, for spatial authentication we rely on direct line of sight between the authenticating devices.

3.3 User Interface Design

Spatial discovery and sensing happen automatically and unobtrusively. Users are then provided with a visualisation of the computed relative positions of devices in the interface on their own personal device. The visualisation has to be such that a user can associate a visual screen object with a device in their environment. Figure 1 shows two possible implementation. The one on the left is based on Guinard et al.'s *Gateways* [12]: these are screen objects arranged around the edge of the user interface, representing devices in the indicated direction relative to the user's device, and here extended to also show distance information. The one on the right is adapted from [11] and shows a map view with icons spatially arranged in correspondence with the actual layout of devices discovered around the user's device. Key to our concept is that the visualisation reflects the 'real' spatial layout, so that users can make a connection between what they see and what their device sees (and visualises). This allows users to invoke interactions by spatial reference, for example simply by dragging an object onto a Gateway or icon representing a remote device. A device thus selected as targeted is associated with a particular bearing and distance as measured with on-board sensors.

4 Threat Analysis

The key idea underlying our method is to use spatial references for verification of device authenticity. In this section we consider threats in the context of the ultrasonic sensing approach we introduced above, as well as threat scenarios that arise on application level.

4.1 Attacker Capabilities

There are three channels of concern: the communication network between devices, e.g. wireless LAN with a TCP/IP stack, the radio frequency channel used for communication between spatial sensing devices (*RF*), and the ultrasound channel used for sending and receiving ultrasonic pulses (*US*). We assume an attacker ('Eve') to be capable of gaining complete control over the wireless communication channels. This allows Eve to perform a 'man-in-the-middle' (MITM) attack on the wireless channels. Assuming to devices A and B, the attacker E can pretend to A that it is B, and to B that it is A, and thus agree to a cryptographic key with A and separately with B. A and B will be unaware of this and believe to communicate securely with each other when in fact they are communicating via E (who might be partially or completely relaying their messages).

The aim of our method is to prevent that a man-in-the-middle can succeed. To this end, spatial references are used during the authentication process, and are therefore subject to potential attack. We can distinguish between three different attacker capabilities with regards to tampering with spatial references, in order of increasing complexity:

1. *RF-only*: Attacks on any of the wireless channels (RF) are the most dangerous, because they can be carried out inconspicuously (see e.g. [13]). With directed antennas, the possible range of an attacker can significantly exceed the normal range of the RF channel, as has been demonstrated by an attack on mobile phones via Bluetooth over a distance of over 1.7 km.
2. *US in room*: Control over the US channel, on the other hand, is assumed to be limited. First, for attacks on this channel, an attacker needs to be physically present in the same room (US is effectively blocked by solid materials such as walls, doors, and windows). Second, although eavesdropping is easily possible, injecting US pulses is more difficult. We assume an attacker to be capable of injecting US pulses at any time with arbitrary strength. Injection in this sense means to insert completely new messages into the US channel, while modifying, replacing, or removing other messages is not possible without detection.
3. *US in line*: An attacker in the same room can inject US pulses, but receiving devices will be able to detect the different angle of arrival. The reason is that – in contrast to distance measurements – angle of arrival is inferred from relative measurements, i.e. differences in time of arrival or signal strength. We assume it impossible to fake the angle of arrival of a US pulse, bar the capability of sound forming for US (which has not yet been shown to be possible). However, an attacker could be placed in line with A and B, and thus not be required to fake the angle.

4.2 Sensing-Level Threats

Attacking the RF channels creates three threats specific to our spatial sensing system:

(a) by removing all RF messages sent from or to a single device, an attacker Eve can prevent the device from entering the sensor network, and thus *make a specific device disappear* for all other devices – however, this can be detected by the device in question.
(b) by changing RF messages, Eve can *tamper with shared measurements*, i.e. those that are taken by remote sensors and exchanged between Relate sensors. Additionally, US ranging depends on trigger packets sent via RF.
(c) by controlling these trigger packets, Eve can *manipulate distance measurements.*

If Eve is spatially aligned line with A and B, she could also send US messages delayed or ahead of schedule to the effect that her position, from Alice's point of view, appears to be where Bob is. This creates a fourth specific threat, namely (d) to *fake the perceived distance.*

Note that, in contrast to ranging measurements, angle of arrival measurements are trustworthy in our sensing system, as they are derived from signal peak values measured on with sensors oriented in different directions, and not from time-of-flight as proposed in [6].

4.3 Application-Level Threats

The possibility to tamper with spatial references leads to three specific attack scenarios on the application level.

1. *Replacement*: The first possibility for attack is to virtually replace another device. This requires two steps: First, the original communication partner, in this case B, needs to be 'silenced' so that it will no longer be visible in terms of wireless communication and measurements. Second, Eve needs to fake her position to appear at the same place where the user ('Alice') expects B to be. In this attack, interaction happens only between Alice and Eve, and no interaction happens with B. Scenarios for this threat are thus limited to asymmetric settings where B is an infrastructure device not monitored by users.
2. *Asynchronous MITM*: When the scenario includes application-level feedback from B to Alice, there is the possibility for an asynchronous MITM attack. An example for such expected feedback is printing: Alice, when sending a document to B, expects her document to print shortly afterwards. In this case, Eve first replaces B as in the first threat, but only temporarily. After finishing authentication with Alice, she authenticates with B and forwards the intercepted messages that were originally intended for it. Eve could therefore try to avoid detection by forwarding to B and thus completing the high-level interaction. This scenario requires that B does not verify the origin of the messages, i.e. that only Alice authenticates B, but not the other way around.

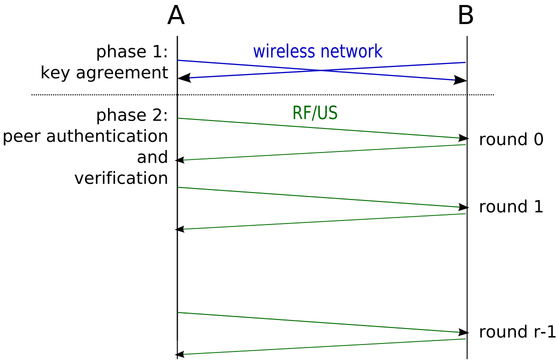


Fig. 2. Devices A and B secure their interaction by key agreement over a wireless network channel, followed by peer authentication over the RF and US channels of their spatial sensors

3. *Synchronous MITM*: For live interaction, like a chat or voice communication between two users (Alice and Bob) over the secure channel, even the slight delay of an asynchronous MITM attack would be noticeable. The most dangerous threat, because it is hard to detect when the attack is being performed, is that of a synchronous MITM. For a synchronous MITM, Eve first attacks the wireless channel as in the previous threat scenarios. But then she remains passive during spatial discovery and mutual positioning of Alice and Bob. Only during spatial authentication she tampers with the spatial measurements. Thus, she remains virtually undetectable for both Alice and Bob, while still having full access to their communication. This requires Eve to be physically between Alice and Bob, because both verify angle of arrival of spatial relationships.

5 Key Agreement and Peer Authentication

We secure spontaneous interaction between two devices A and B in two phases, *key agreement* and *peer authentication*, as shown in Fig. 2. In the first phase, we let A and B establish a shared key using a standard, unauthenticated key agreement protocol, such as Diffie-Hellman (DH) [14]. If this is successful, then A and B can use the agreed key to protect their communication against eavesdropping and tampering, with E being unable to gain sufficient knowledge of that shared key. To protect A and B against MITM attacks, we use a second phase for peer authentication (A establishing that it is really talking to B, and vice versa), and for verification that A and B are in possession of the same key (which would rule out the presence of a MITM due to the unique-key property of a protocol such as DH).

5.1 Peer Authentication

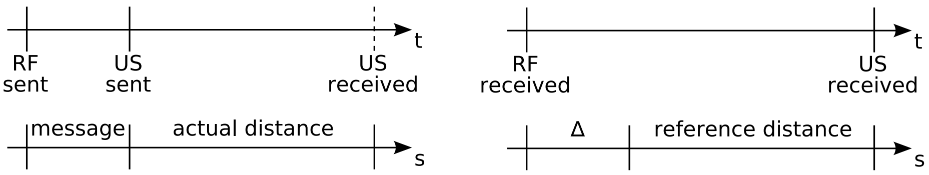
The peer authentication process is designed to be symmetric, which means that the two devices A and B authenticate each other. Even though the interaction is initiated by A in response to Alice's selection of B as target, it will often be appropriate that B can also verify the sending device and its relative position, for example to provide its user Bob with a verified visual indication in his user interface of *where* a received document has been sent from (and thus prevent replacement or asynchronous MITM attacks). As a starting point for authentication, A has a spatial reference to B as derived from the user's selection of B as her target, and B can base authentication on a corresponding spatial reference to A.

Devices A and B use the RF and US channels of their sensor nodes for peer authentication in order to tightly couple this process with spatial sensing. The devices engage in a protocol designed to establish that (i) they have agreed to the same key, and (ii) they are A and B as mutually verifiable by spatial reference. The devices approach this by generating a nonce (a random number used only once) and by transmitting the nonce encrypted over the RF channel. They also transmit the plaintext nonce over the US channel in a series of smaller parts that are coded within the actual distance measurements. When the devices receive these transmissions, they decrypt the RF message, verify that the content matches the nonce received via US, and thus establish whether their keys match. For this approach to be secure, the encoding and the transmission of these nonces need to be coordinated. In the following, we discuss these two issues and how they interact with each other.

5.2 A Spatial Coding Technique for Trustworthy Ultrasonic Ranging

When a device receives an ultrasonic pulse, it computes a distance measurement based on the time-of-flight. As explained above in section 4.2, these distances can be tampered with. We therefore introduce a method to embed information in ultrasound pulses, which (i) allows to use US as an out-of-band channel for message exchange, and (ii) makes the distances trustworthy.

During authentication, the sender delays the sending of pulses to the effect of adding a certain perceived distance to the measurement, where the added distance represents information (in our protocol, a substring of the nonce). When for instance A receives a pulse and computes a distance, this distance is the actual distance from the sender plus a distance representing the message. A proceeds with subtracting the reference distance it has of B (note the reference distance is captured when the user selects a device for interaction). This will let A retrieve the information (represented as added distance) correctly only if the received pulse has been sent from a range that corresponds with the relative position of B. That is, a correct reconstruction of the message implies that the distance is equal to the reference measurement, and therefore constitutes and implicit check of spatial integrity. Figure 3 illustrates this mechanism for message transmission over ultrasound with implicit verification of sending range. In addition to this implicit distance check, A can verify that the pulse was received from a direction corresponding with the reference held for B, thus effectively eliminating the possibility that the US transmission originates from another device but B.



(a) The sender delays an US pulse after the RF trigger to code a message, which corresponds to a distance in the spatial domain.

(b) The receiver measures time from trigger to pulse arrival, and subtracts a stored reference from the corresponding distance to retrieve the message.

Fig. 3. Message transmission embedded with ultrasonic ranging: The receiver will only be able to retrieve the message if the sender's distance matches the stored reference

5.3 Preventing MITM Relaying

A and B can thus verify that ultrasound pulses are received from the intended partner device but it is still possible that E is present as MITM on the RF channel. E would be able to infer the nonces exchanged between A and B by taking its own US measurements (note that this only requires eavesdropping on US pulses, which is simple to do as long as E is in the same room), and it could then use its keys (maliciously agreed with A and B in the key agreement phase) to re-encrypt the nonces in order to pass the key verification checks of A and B. To rule this possibility out we use an interlock protocol, which in essence commits the sender of a message to the message content before it has been transferred completely [15]. For this purpose, A and B split the encrypted nonces into multiple parts and take turns in transmitting their parts. The nonces are encrypted with a block cipher, which means that all message parts need to be reassembled before the message can be decrypted to retrieve the nonce. If E now receives a message part from A intended for B, it can not retrieve any part of the nonce. E will also not receive more message parts from A unless it passes the current one on to B, as A and B strictly adhere to turn-taking. E's only choices are then to guess the content for all message parts that will 'pass through' (before they are even transmitted by A and B, let alone decrypted by them) in order to re-encrypt these successfully (this is practically impossible), or to relay message parts unchanged in which case A and B will discover that their keys do not match (thereby detecting the presence of a MITM and aborting authentication). The interlock protocol thus rules out that a MITM attack on the RF channel can succeed during peer authentication.

5.4 Protocol Specification

An overview of the protocol phases is shown in Fig. 2. Key agreement takes place over a wireless network channel, and subsequent key verification and peer authentication over the RF/US channels of their spatial sensors. The second phase involves turn-taking of the parties in an interlock protocol over a number of rounds r. This number will be agreed between devices, in consideration of the security level, protocol duration, and US channel capacity. The US channel capacity b_u is the number of bits that can be reliably transmitted as distance offset in each round, and will depend on the characteristics of the sensors used and sensing protocol details. Assuming a nonce of 128 bits, we would need $\lceil 128/b_u \rceil$ rounds for transmission of the nonce over US. However, a smaller number of rounds may be agreed to complete the protocol faster, compromising on how many bits of the nonce are eventually compared for key verification. With r agreed, we then set the number of bits that will transmitted over the RF channel in each round to $b_m := \lceil 128/r \rceil$, splitting the encrypted nonce into equal message parts.

We will now describe our protocol more formally using the following notation: $c := E(K, m)$ describes the encryption of plaintext m under key K with a symmetric block cipher, $m := D(K, c)$ the corresponding decryption, $H(m)$ describes the hashing of the message m with a secure hash algorithm, and $m||n$ describes the concatenation of strings m and n. Additionally, the notation $M[a : b]$ is used to describe the substring of a message M starting at bit a and ending at bit b. Messages that are transmitted to the other party are printed in bold.

1. *Key agreement*, using the Diffie-Hellman key establishment protocol:
   (a) A chooses a random number $a \in \{1, ..., q-1\}$ and transmits $\mathbf{X} := g^a$,
   B chooses a random number $b \in \{1, ..., q-1\}$ and transmits $\mathbf{Y} := g^b$
   (b) A computes $K_a^{Sess} := H(\mathbf{Y}^a)$ and $K_a^{Auth} := H(\mathbf{Y}^a||PAD)$ with some secure hash algorithm,
   B generates K_b^{Sess} and K_b^{Auth} correspondingly from $\mathbf{X}^b$

   The numbers g, q and the string PAD are assumed to be publicly known. Although we envisage the use of ephemeral keys, i.e. new values for a and b for each protocol run, it might be advantageous to use long-term values for performance reasons. We use K^{Auth} $(= K_a^{Auth} = K_b^{Auth})$ for key verification in the peer authentication phase, and K^{Sess} $(= K_a^{Sess} = K_b^{Sess})$ for subsequent channel security if the verification succeeds. The additional hashing to compute two different shared keys provides forward secrecy in the case of leaked authentication key material (cf. [16, section 15.8.4]), for example by a known plaintext attack on $E(K_x^{Auth}, N_x)$ after the respective N_x is revealed in the following steps.
2. *Peer authentication*:
   (a) A chooses a nonce $N_a \in \{1, ..., 2^{128}-1\}$ and computes $M_a := E(K_a^{Auth}, N_a)$,
   B chooses N_b and computes M_b correspondingly with K_b^{Auth}
   (b) *For each round* $i := 0 \ldots r-1$:
   - A transmits a RF packet $\mathbf{M_a^i} := M_a[i \cdot b_m : (i+1) \cdot b_m - 1]$ and an US pulse $\mathbf{USP_a^i}$ delayed by $N_a[i \cdot b_u : (i+1) \cdot b_u - 1]$ units,
   - B receives message part $\mathbf{M_a^i}$ and US pulse $\mathbf{USP_a^i}$, derives a distance measurement $d_{b,a}^i$, and uses the stored reference measurement $d_{b,a}$ to reconstruct the distance-coded message $\Delta_a^i := d_{b,a}^i - d_{b,a}$. B also verifies the angle of arrival $\alpha_{b,a}^i$ and compares it with the stored reference

measurement $\alpha_{b,a}$. If the difference exceeds the typical measurement error, B aborts the authentication protocol with an error message.

- B transmits $\mathbf{M_b^i} := M_b[i \cdot b_m : (i+1) \cdot b_m - 1]$ and $\mathbf{USP_b^i}$ delayed by $N_b[i \cdot b_u : (i+1) \cdot b_u - 1]$ units, and acknowledges receipt of A's RF and US messages for round i,
- A receives $\mathbf{M_b^i}$ and $\mathbf{USP_b^i}$, verifies angle of arrival, computes $d^i_{a,b}$, uses the reference measurement $d_{a,b}$ to reconstruct $\Delta^i_b := d^i_{a,b} - d_{a,b}$, and acknowledges B's messages for round i

(c) A reassembles all received RF packets $M'_b := \mathbf{M_b^0}|| \dots ||\mathbf{M_b^{r-1}}$, decrypts the message $N'_b := D(K_a^{Auth}, M'_b)$, reassembles the nonce from the distance offsets $N''_b := \Delta^0_b|| \dots ||\Delta^{r-1}_b$, verifies that $N''_b = N'_b[0 : r \cdot b_u - 1]$, and sets $K := K_a^{Sess}$ on match or $K := null$ otherwise,
B reassembles $M'_a := M^0_a|| \dots ||M^{r-1}_a$, decrypts $N'_a := D(K_b^{Auth}, M'_a)$, reassembles $N''_a := \Delta^0_a|| \dots ||\Delta^{r-1}_a$, verifies that $N''_a = N'_a[0 : r \cdot b_u - 1]$, and sets $K := K_b^{Sess}$ on match or $K := null$ otherwise

Note, if $b_u < b_m$ (i.e. if fewer bits are transmitted via US than via RF) then step 2c) only compares $r \cdot b_u$ bits of the nonce.

If key agreement and peer authentication are completed successfully, then A and B can use the session key K to establish a secure channel. The key can be used as a shared secret for one of the standard protocols such as IPSec with PSK authentication, or one of the recently specified TLS-PSK cipher suites [17]. Other options are WPA2-PSK or EAP-FAST. K can be used directly as key material, rendering additional asymmetric cryptographical operations in the secure channel implementation unnecessary and thus speeding up channel establishment.

5.5 Implementation

We have implemented the key agreement phase of our protocol over TCP/IP. As a secure hash we use SHA$_{\mathrm{DBL}}$-256, which is a double execution of the standard SHA-256 message digest to safeguard against length extension and partial-message collision attacks [16]: SHA$_{\mathrm{DBL}}$-256 = SHA-256 ((SHA-256 (m)) $|m$).

The peer authentication phase of the protocol has been implemented over the RF/US channel of the Relate sensors, using AES (Rijndael) with a key size of 256 bits as secure block cipher for the interlock protocol. The protocol is tightly integrated with the Relate spatial sensing protocol. RF packets transmitted for authentication serve simultaneously as trigger packets for ultrasonic time-of-flight measurement. Pulses emitted on the US channel serve simultaneously for ranging and for transmission of nonce message parts.

Derived from the characteristics of the Relate sensors, we have set the number of bits transmitted in each round over US to $b_u := 3$. In each round, the 3 bit number is coded as multiples of 25.6 cm which the sender adds as offset to the receiver-perceived distance by delaying the US pulse. At the receiver end, this allows for +/-12.8 cm of measurement inaccuracy to retrieve the 3 bits correctly (note the reported precision of Relate sensors for this level of accuracy is over 95%). The duration of a round is about 200 ms (longer if other devices present are allowed to 'interrupt' the authenticating peers for spatial sensing and exchange of measurements). Transmission of the complete nonce would require 43 rounds but the number of rounds has been kept variable in our implementation to allow users to define their required level of security.

6 Security Analysis

6.1 Message Channels

In our case, information is transmitted both via RF and via US. To safeguard against *eavesdropping* all RF packets are encrypted with an authentication key, but over US the nonce will become gradually revealed as the protocol proceeds. The interlock protocol ensures that this will be of no use to an attacker, as the protocol forces commitment of encrypted nonce message parts over RF before the entire nonce can be intercepted on the US channel. The nonce is also strictly used only once which rules out *replay* attacks. Complete or selective *denial-of-service* attacks can not be protected against under our assumption of completely insecure RF channels.

As described above, the main motivation for using the interlock protocol is to protect against man-in-the-middle attacks *during* authentication. An RF-only MITM attack would be noticed, and we therefore need to analyse the possibilities for a concurrent attack on the US channel.

6.2 Ultrasonic Sensing and Message Transmission

Our approach to coding random nonces (section 5.2) and transmitting them via interlock (section 5.3) prevents all the threats outlined in section 4.2: Threat (a) constitutes a selective denial-of-service attack that can be detected by time-outs (when the selected device does not respond at all) or authentication failures (when the attacking devices responds from a different spatial position). Threat (b) does not apply to our protocol, because shared measurements are not used during authentication. Threats (c) and (d) are prevented by the random delays. As E can not know in advance when a US pulse will be sent by A or B (the delays are derived from the random nonce part that is kept secret until sending the pulse), it can not construct the encrypted RF packets to match these delays. If E injected own US pulses, A and B would also receive the original ones and thus detect that an attack is happening. E's only chance would be to cancel US pulses in-transit by generating appropriate anti-US pulses, but this is considered prohibitively difficult. Furthermore, E would need to be positioned precisely in the line-of-sight between authenticating devices in order to attempt interception and manipulation of US pulses but this presence literally in the middle between devices would be obvious to the user. Note that this MITM device can not be arbitrarily small due to a physical limits on the minimum size of ultrasound transducers.

One remaining risk is that E is positioned in line with A and B, but farther away instead of in between. If E performs a selective denial-of-service attack on B and forges distance measurements before authentication is started, it will be able to fake its perceived and subsequently visualised position as seen by A. Although for security purposes one does usually not trust other devices's measurements (they might be collaborating for an attack), we note that these measurements, shared by benign devices over the Relate RF network, may serve to reveal ongoing attacks such as this one. The shared measurements are not used for increasing trust in an authentication protocol run or providing proof of authentication, but they may still be used for decreasing trust in a protocol run, when shared measurements do not match local ones. Attacking networks of multiple Relate devices should therefore be considered significantly harder than attacking just two devices.

We should also note that attacks on the sensing level become harder in scenarios involving mobility of devices. Positioning an attacker unsuspiciously and directly in line between A and B is not trivial even in static settings. When at least one of the interacting devices is mobile, an attacker would need to be constantly re-positioned (or virtualized by sound forming, which is considered infeasible with the current state of the art in ultrasonic systems).

6.3 Applications

The application-level threats described in section 4.3 are specific to our method. With the protections of the sensing level described above, the remaining threat is the misrepresentation of E at the position of B as seen by A. *Replacement* of infrastructure devices is hard to detect, and therefore difficult to protect against. One possibility is to create an explicit application-level feedback from B that can be verified by Alice, for example to lighting an LED for a few seconds whenever authentication has succeeded. If Eve replaces B, then B will not light its LED and Alice can subsequently abort the interaction. The same protection can be used against *asynchronous MITM*, which effectively transforms these two scenarios into a *synchronous MITM* setting. However, this adds an additional step in the interaction process that may not be desirable for many applications. A more pragmatic protection against these remaining replacement and asynchronous MITM threats is to protect against E being in line with A and B by physical means, e.g. simply placing B directly in front of a wall and thus making it impossible for E to be hiding 'behind' it.

Synchronous MITM seems prohibitively difficult to perform under the above analysis of the sensing level protection, because of the necessary in-transit attacks on US pulses.

6.4 User Interaction

The overall security of our method depends on the correct selection of the target device, and the correct association of the target with a spatial reference. We need to consider two possible sources of error or incorrect association. One is that the network communication in the initial steps of our method is not secure. A user can trust the relative position information it has of other devices as this is measured with on-board sensors but any additional information exchanged may be interfered with by an attacker. For example, the Gateway interface shown in Fig. 1 is based on locally measured spatial information but in addition visualises type of discovered device based on information received over the wireless network. An attacker might tamper with this to the effect that a different device type is indicated, which might mislead the user.

The second risk at the level of user interaction is that the user selects the 'wrong' device in their user interface, in the worst case an attacker positioned near the actual target. i.e. E instead of B, in their user interface. The visual design of the UI and the accuracy of the spatial layout in correspondence with the 'real world' arrangement of devices will be key factors in reducing the risk of faulty selection, which of course will also be dependent on number and arrangement of devices discovered and visualised.

7 Performance Evaluation

The authentication protocol involves evaluation of sensor data with inherent limitations in accuracy and precision. It is therefore critical to assess impact of sensor limitations on practical performance.

7.1 Robustness Against 'False Negatives'

Sensors are inherently imprecise. Our authentication protocol is designed to account for the resulting variance in sensor readings, but only within limits that are consistent with secure authentication of devices by spatial reference (i.e. there must be no possibility that devices become confused due to allowances made for sensor error). As a consequence, the protocol can fail to authenticate legitimate peers when sensor errors occur that exceed built-in tolerance.

Figure 4 shows the success rate of authenticating legitimate peers dependent on the number of rounds of the interlock protocol and the distance between the devices. For this experiment, two devices were positioned facing each other in direct line of sight at distances of 50cm and 100cm. For each number of rounds, 250 protocol runs were performed. As shown in Fig. 4, success rates are at least 85% and typically above 95%.

Authentication only succeeds if every single US measurement taken during the protocol rounds is sufficiently accurate. In our experiment, the success rate did not decrease significantly with the number of rounds. However under less controlled conditions (e.g. slight movement of devices during the protocol run) a more notable decrease might be expected, as the probability of an erroneous measurement increases with the number of rounds. Note that the impact of distance on success rate is not very pronounced and appears to be within error of measurement (success rates for the larger distance are on average lower, but not consistently).

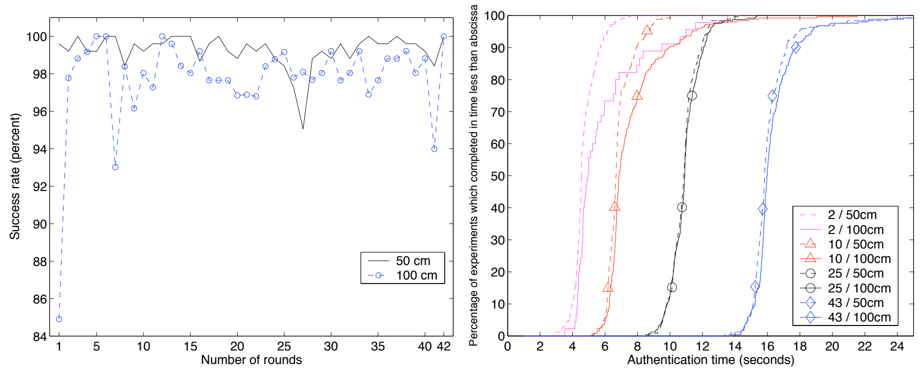


Fig. 4. Authentication success rate depending on the security level and distance (left); Authentication speed depending on the security level and distance (right)

7.2 Speed Versus Security

There is an inherent trade-off in our protocol between speed and security. The resistance against attacks increases with the number of rounds used for the interlock protocol, because each round transmits 3 bits of entropy for verifying the nonce. Therefore, an attacker's chance of guessing a nonce equals $1/2^{3r}$. To put this into perspective, after only 5 rounds a nonce would already be harder to guess than a randomly generated 4-digit PIN number. Also note that our protocol is symmetric, which means that an attacker would need to guess two nonces correctly in order to deceive the authenticating devices as MITM.

Figure 4 (right) shows this trade-off with measurements taken for 2, 10, 25, and 43 rounds, obtained with the same experimental setup of devices as described above. The variations in the time necessary for authentication over a certain number of rounds stem from the specifics of the underlying RF/US sensing protocol which can require message retransmits. The dependency on the distance between the devices is again marginal. As can be seen, a complete authentication takes around 12 s for 25 rounds.

It is important to understand that a compromise on the number of rounds in our protocol only impacts on an attacker's one-off chance to guess the correct nonce to stage an undetected MITM attack. It does not impact on the security level of 128 bits that will be provided after successful authentication. This difference is even more pronounced than in the usual online vs. offline attack discussion, because of the tight coupling with interaction at the user level. An attacker can not repeatedly attack the authentication protocol with an online attack, because it is only triggered by an explicit user action. Therefore, there is only a one-off chance for an attack, and any computational attacks are therefore matched with a security level of 128 bits. Nonetheless, our protocol allows the user or application to choose the best compromise between speed and security and scales up to a 128 bit level even for the single attack possibility.

8 Conclusion

We have contributed and discussed a method for establishing secure spontaneous interaction on the basis of spatial references. Spatial references are a type of context that allows users to match what they see with what their device sees. At the core of our method is a peer authentication protocol that uses relative bearing and distance between devices. We have presented an implementation using ultrasound for spatial measurements; however, the method can also be realised with other sensors. For example, one could consider use of cameras (which are becoming ubiquitous in mobile phones and handhelds) and vision techniques to obtain spatial references between devices.

The concrete implementation we have presented uses ultrasound, for peer-to-peer spatial sensing, and for out-of-band message transfer as part of a key verification protocol. We have provided a comprehensive threat analysis for ultrasonic ranging and contributed a novel coding technique that allows a sender to guarantee that a message was sent from a particular range. This technique can thus be used to to construct a spatially-authentic channel from sender to receiver.

Our protocol implementation is embedded in a spatial sensing scheme that more generally provides devices with accurate relative positions of peers discovered within interaction range. The method further involves a user interaction model based on visualisation of relative device positions, integrated in the user interface for direct manipulation. The method as presented relies on spatial sensors, however, the sensors are not specific to the purpose of providing security but have broader use for support of spatial interaction and services. Cameras and various other sensors are already ubiquitous in mobile devices, and given the general utility of ultrasonic transducers for ranging tasks it is easily perceivable that these will become commonplace as well.

As a final note it has to be stressed that the presented approach fundamentally differs from proximity-based methods such as near-field communication (NFC). Any proximity-based method that relies on a *quantitative* property of the out-of-band channel such as radio signal strength is open to attack from further afield — for example to attack NFC by increasing communication range with more powerful senders and/or more sensitive receivers. In contrast, our method exploits the *qualitative* out-of-band properties of ultrasound: that it is blocked by solid materials and that angle of arrival can not be faked.

The complete source code is available under an open source license at `http://ubicomp.lancs.ac.uk/relate/`.

Acknowledgements

We acknowledge support for the presented research by the Commission of the European Union under contracts 013790 "RELATE" and the FP6 Marie Curie Intra-European Fellowship program contract MEIF-CT-2006-042194 “CAPER”, and by the Engineering and Physical Sciences Research Council in the UK under grant GR/S77097/01.

References

1. Gehrmann, C., Mitchell, C.J., Nyberg, K.: Manual authentication for wireless devices. RSA Cryptobytes 7(1), 29–37 (2004)
2. Stajano, F., Anderson, R.: The resurrecting duckling: Security issues for ad-hoc wireless networks. In: Proc. 7th Int. Workshop on Security Protocols, pp. 172–194. Springer, Heidelberg (1999)
3. Balfanz, D., Smetters, D.K., Stewart, P., Wong, H.C.: Talking to strangers: Authentication in ad-hoc wireless networks. In: Proc. NDSS'02, The Internet Society (2002)
4. Kindberg, T., Zhang, K., Shankar, N.: Context authentication using constrained channels. In: Proc. WMCSA 2002, pp. 14–21. IEEE Computer Society Press, Los Alamitos (2002)
5. Kindberg, T., Zhang, K.: Secure spontaneous devices association. In: Proc. UbiComp 2003, pp. 124–131. Springer, Heidelberg (2003)
6. Kindberg, T., Zhang, K.: Validating and securing spontaneous associations between wireless devices. In: Boyd, C., Mao, W. (eds.) ISC 2003. LNCS, vol. 2851, pp. 44–53. Springer, Heidelberg (2003)
7. McCune, J.M., Perrig, A., Reiter, M.K.: Seeing-is-believing: Using camera phones for human-verifiable authentication. In: Proc. IEEE Symp. on Security and Privacy, pp. 110–124. IEEE Computer Society Press, Los Alamitos (2005)
8. Goodrich, M.T., Sirivianos, M., Solis, J., Tsudik, G., Uzun, E.: Loud and clear: Human verifiable authentication based on audio. In: Proc. ICDCS 2006, p. 10. IEEE Computer Society Press, Los Alamitos (2006)
9. Clulow, J., Hancke, G.P., Kuhn, M.G., Moore, T.: So near and yet so far: Distance-bounding attacks in wireless networks. In: Buttyán, L., Gligor, V., Westhoff, D. (eds.) ESAS 2006. LNCS, vol. 4357, pp. 83–97. Springer, Heidelberg (2006)
10. Hazas, M., Kray, C., Gellersen, H., Agbota, H., Kortuem, G., Krohn, A.: A relative positioning system for co-located mobile devices. In: Proc. MobiSys 2005, pp. 177–190. ACM Press, New York (2005)
11. Kortuem, G., Kray, C., Gellersen, H.: Sensing and visualizing spatial relations of mobile devices. In: Proc. UIST 2005, pp. 93–102. ACM Press, New York (2005)
12. Guinard, D., Streng, S., Gellersen, H.: Relategateways: A user interface for spontaneous mobile interaction with pervasive services. In: CHI 2007 Workshop on Mobile Spatial Interaction (2007)
13. Shaked, Y., Wool, A.: Cracking the Bluetooth PIN. In: Proc. MobiSys 2005, pp. 39–50. ACM Press, New York (2005)
14. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. on Information Theory IT-22(6), 644–654 (1976)
15. Rivest, R.L., Shamir, A.: How to expose an eavesdropper. Commununications of ACM 27(4), 393–394 (1984)
16. Ferguson, N., Schneier, B.: Practical Cryptography. Wiley Publishing, Chichester (2003)
17. Eronen, P., Tschofenig, H.: RFC4279: Pre-shared key ciphersuites for transport layer security (TLS) (December 2005)

5 On the Security of Ultrasound as Out-of-band Channel

Authors: Rene Mayrhofer and Hans Gellersen

Published in the proceedings of SSN 2007, the 3rd International Workshop on Security in Systems and Networks, part of IPDPS 2007, the 21st IEEE International Parallel and Distributed Processing Symposium, p. 321, IEEE Computer Science Press
ISBN 1-4244-0909-8
March 2007

On the Security of Ultrasound as Out-of-band Channel

Rene Mayrhofer and Hans Gellersen
Computing Department, Lancaster University
{rene,hwg}@comp.lancs.ac.uk

Abstract

Ultrasound has been proposed as out-of-band channel for authentication of peer devices in wireless ad hoc networks. Ultrasound can implicitly contribute to secure communication based on inherent limitations in signal propagation, and can additionally be used explicitly by peers to measure and verify their relative positions. In this paper we analyse potential attacks on an ultrasonic communication channel and peer-to-peer ultrasonic sensing, and investigate how potential attacks translate to application-level threats for peers seeking to establish a secure wireless link. Based on our analysis we propose a novel method for authentic communication of short messages over an ultrasonic channel.

1. Introduction

Spontaneous interaction in wireless ad hoc networks is especially vulnerable to attacks on the wireless communication channels. Such attacks include eavesdropping, injecting and modifying packets, replay, or denial of service. We generally have to assume that attackers are able to mount 'man-in-the-middle' (MITM) attacks, where they agree to two different keys with the communicating devices and which subsumes the other attack types. The major problem of purely wireless communication is therefore key management: to securely exchange keys with the intended communication partner. Using (supposedly) computationally secure key agreement protocols such as Diffie-Hellman [4], this problem is further shifted to that of authentication: to securely verify that a key belongs to the intended communication partner. In pervasive computing, and more generally in peer-to-peer networking, we can not currently assume the availability of a globally trusted third party. For spontaneous interaction, the only option is therefore ad hoc verification of keys, which requires some extra channel with additional security properties. This is a channel other than the main wireless channel and often called 'out-of-band' channel.

Balfanz et al. introduced the notion of *location-limited channels* for out-of-band communication channels that require devices to be in a certain, user-verifiable spatial relationship in order to establish communication [1]. Kindberg et al. discussed *constrained channels* along similar lines [11]. A variety of communication technologies have been considered for implementation of out-of-band channels with the desired characteristic of limiting communication to a user-controlled context. This includes ultrasound, on which we focus our analysis in this paper.

Ultrasound (US) is an interesting candidate technology for out-of-band communication alongside wireless radio (RF), for two reasons. First, ultrasound has inherent limitations in signal propagation (unlike RF, US signals are contained in rooms). Secondly, ultrasonic communication can be used by peer devices to estimate their relative positions (from US time-flight measurements, with RF communication for synchronisation [9]), and thus to obtain information that can be useful for verification of device authenticity. Ultrasound has been noted as a possible technology for authentication of peers within a room, exploiting its broadcast and propagation characteristics [1]. A concrete protocol design with ultrasound as out-of-band channel for authentication of spontaneous device associations has been discussed in [10]. In this protocol, ultrasound is proposed for out-of-band communication of nonces, and for verification of the spatial direction from which a transmission has been received. However, the protocol has not been implemented, and assumptions made concerning the use of ultrasound have neither been tested nor analysed in more depth.

In this paper we contribute an analysis of ultrasound as out-of-band channel for secure authentication of devices in wireless ad hoc networks. We assume a device A seeking to establish a secure wireless link to a device B without prior knowledge of B, or access to a shared trusted third party. The principal threat in this scenario is that a man-in-the-middle E can establish itself between A and B. A and B may be mobile devices, but they are assumed to be static in relation to each other during the initial establishment of the link (but may move freely after successful channel establishment). The protocol proposed earlier [10] has the disadvantage that users are expected to move deliberately to different locations and verify spatial measurements during the authentication phase, which can be cumbersome for ad hoc interaction. We do not assume any explicit actions by users solely for authentication purposes, but analyse the security properties of an ultrasonic channel by itself.

In the following sections we first look into properties of ultrasonic systems that can be exploited for peer authentication. We then analyse attack scenarios on the ultrasonic communication channel, and further analyse how these translate to threats at application level. We conclude the paper describing a novel method for authentic communication of short messages over an ultrasonic channel.

2. Properties of Ultrasonic Systems

Devices that use ultrasound as out-of-band channel can exploit properties of the medium both implicitly and explicitly. Ultrasound has propagation characteristics that implicitly contribute toward location-limited communication, in particular by containing signals in rooms which provides users with a distinct level of control. Devices can use ultrasound also explicitly, to estimate their spatial relationship for purposes of verifying that the device they are talking to is indeed in the assumed position, for instance 'in front of the user'.

Ultrasound signals are, due to the large differences in acoustic impedances between air and solid materials, (almost) completely reflected or absorbed by walls, doors and windows. Bending around doors or other openings causes chaotic influences on signal propagation and is practically unpredictable from an attacker's point of view. Consequently, we can assume signals or messages transmitted over an ultrasonic channel not to leave a room, and we can also assume that it is not possible to inject ultrasonic messages into a room from the outside.

Ultrasound signals travel at comparatively low speed which makes it possible for a pair of devices to measure time-of-flight of a pulse or message transmitted over an ultrasonic channel, provided they have access to an RF channel for synchronisation. Time-of-flight measurements allow for very accurate ranging (i.e. distance estimation) between peers, with errors reported well below 10cm [13, 9]. Even better accuracy can be achieved if either multiple emitters or receivers are used to take measurements from different angles (as in ultrasonic positioning infrastructures, e.g. [7, 14]). However, for our target use, verification of A and B's authenticity in a spontaneous encounter, we assume that devices will not trust other sensors but their own.

Estimation of the direction from which an ultrasound signal has arrived is possible if a device has multiple receivers on board. If these receivers are placed sufficiently far apart then it can be possible to estimate angle-of-arrival from differences in time-of-flight (this method was suggested though not tried in Kindberg & Zhang's peer authentication protocol [10]). Another possibility, better suited for devices of small dimension as typical in mobile scenarios, is to use an arrangement of receivers facing in different directions and to derive angle-of-arrival from analysis of peak signal values (an incoming pulse or message will register the highest peak with the receiver oriented most closely to the emitting device). This method has been used for instance in the RELATE system with 3 transceivers covering 180 degrees, with reported raw measurement error of 33 degrees [9].

3. Threat analysis for ultrasonic communication and sensing

For our threat analysis we assume devices A and B seeking to secure communication over a wireless radio network. We further assume possible use of ultrasound as out-of-band channel over which messages can be exchanged, and use of ultrasound synchronised over RF for estimation of distance and possible relative orientation. Note that transmitting messages over US as part of an authentication protocol is different from using US for distance-bounding protocols, introduced as a method for determining the maximum distance between devices [2]. As shown recently [3], direct use of US for distance bounding is open to relaying attacks and thus not considered secure for authentication of peer devices. For our analysis, we do not assume US to provide a *secure* upper bound on the distance.

As for attacker capabilities, we make two principal assumptions:

1. An attacker can stage attacks on the RF channel from anywhere within the range of the wireless network, to eavesdrop, to cause Denial-of-Service (DoS), or to impose itself as man-in-the-middle (MITM) between A and B (to the effect that A and B believe to be talking to each other, while actually talking to E).

2. Attacker capabilities on the US channel depend on how the attacker is located relative to the attacked devices[1]; in other words, we assume that an attacker is not able to 'virtualize' their position by using groups or whole arrays of coordinated ultrasound emitters. Note that speaker arrays can be used for spatialised audio [5] but due to the shorter wavelength of ultrasound and its more complex propagation characteristics, it would appear prohibitively difficult to achieve accurate ultrasound spatialisation.

[1] With location or position of an attacker we actually refer to the position of the communication device used to mount the attack.

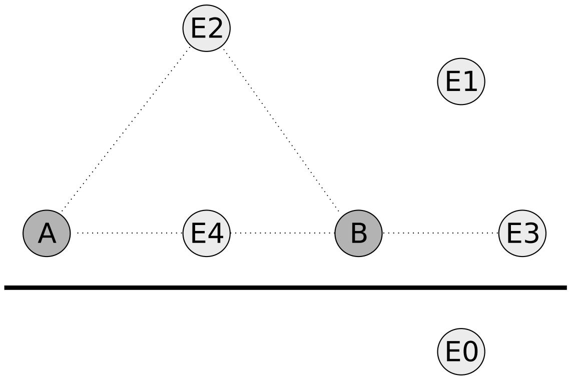


Figure 1. The capability of E to stage an attack on ultrasonic communication and sensing between A and B depends on how E is positioned with respect to A and B.

Figure 1 illustrates our scenario with devices A and B, and an attacker E that can be in different positions (E0, E1, etc.) with respect to A and B. In the following we analyse potential threats to ultrasonic communication and sensing between A and B for each of these positions:

1. *E0 – outside room*: It is an inherent property of ultrasound that signals are blocked by walls, doors and windows. Therefore we can assume that it is not possible to eavesdrop on ultrasound communication from outside a room, and that it is also not possible to interfere with the ultrasound channel from the outside, i.e to insert or modify messages that would compromise ultrasonic communication between A and B. While US signals can bend around doors or corners, this subjects them to distortions that in practice also exclude meaningful attacks. In such situations, accurate location information for both attacker and victim would be needed to produce proper time-delay-of-arrival at the receiver; creating an arbitrary angle-of-arrival is prohibitively difficult [8].

   However, an attacker E would still be able to mount an attack on the RF channel, for general denial of service, targeted prevention of devices from discovery on the network, or tampering with messages that may be used for synchronisation of ultrasonic sensing. The concrete threats with respect to ultrasonic communication and sensing are: a) to *prevent a device from participation in ultrasonic communication*, for example to the effect that a target device B selected by a user with device A becomes barred from authenticating itself; b) to *cause erroneous distance estimates*, i.e. having A estimate B to be closer or further away then they actually are and thus compromising any distance- or position-based verification procedure; and c) to *modify any distance, orientation, or position estimates exchanged over RF* (note it is common in ultrasonic positioning system to exchange measurements over RF, but it is not required and can be avoided). Threat b) also invalidates any assumptions that may be made for distance-bounding methods, even without more complex relay attacks as described earlier [3].

2. *E1 – in room*: If E is in the same room with A and B, or more precisely in the same propagation range, then E will have the additional capability to listen to all ultrasound communication, and to insert pulses or messages on the ultrasound channel, at any time with arbitrary signal strength. However, E will not generally be able modify or replace other messages exchanged over the channel, e.g. between A and B, unless E is positioned "more strategically" in cases we discuss further below. Concrete threats arising are thus: d) to *eavesdrop on the ultrasound channel*; e) to *insert ultrasonic pulses or messages* with a potential of confusing or compromising ultrasonic sensing between other devices, and to present itself at a certain distance (cf. case E2); and f) to *block ultrasound transmission*, which, depending on the specific implementation of ultrasonic sensing and the sophistication of the attack, may or may not be distinguishable from signal noise. The distinction between blocking of transmission, i.e. denial of service, and selectively removing messages sent by other devices is blurred.

3. *E2 – equidistant positions*: If E is positioned at the same distance from a receiving device (e.g. A) as the intended target device (e.g. B) then E might achieve to be verified as B, if verification were based on distance only. Obviously if E is positioned equidistant from both devices, it may achieve positive verification by both and A and B as a man-in-the middle. Note that E can easily make itself appear at a particular distance from a single receiver, from anywhere in its ultrasonic range, by emitting an ultrasonic pulse or message ahead of the synchronisation schedule (to appear nearer) or delayed after a synchronisation point (to be appear farther). The threat in either case is: g) to *appear at the same distance as the target device*, and it highlights the value of angle-of-arrival estimates in addition to range measurements for device verification purposes.

4. *E3 – in line*: When E manages to position itself in line with A and B, its US messages will be received at an angle-of-arrival that corresponds with the angle at which the device 'in the middle' is positioned from the perspective of the receiving device. For example,

in the scenario shown in Fig. 1, E will appear to be B from A's point of view, but not to be A from B's point of view. The threat is thus: h) to *appear from the same angle to a single device*. Note that peer-to-peer angle-of-arrival estimates tend not to be very accurate in practice and thus it may suffice for E to approximate a position in line with A and B, in order to produce this threat.

5. *E4 – in between*: A position at some point on the line directly in between A and B offers E most capabilities for an attack on US communication and sensing between A and B. US messages produced by E will be received from the same angle at which A and B are positioned respectively, creating threat i) to *appear from the same angle to both devices*. Additionally, by being directly in the line of US signalling between A and B, the attacker E may be able j) to *cancel or modify specific US messages in transit*, by means of generating anti-ultrasound (similar to noise-cancellation in audio).

Table 1 summarises threats to ultrasonic communication and sensing:

Case	Threats	Safeguards
E0	Attack on RF	US safe as out-of-band channel for authentication
E1, E2	Attack on US ranging	Check for duplicate pulses, verify angle of arrival
E3	Attack from direction of target	Mutual verification of positions
E4	Attack from direction of peer	Requires additional measures

Table 1. Summary of threats and safeguards

The main conclusions from this analysis are:

- Ultrasound can be effective as out-of-band channel for authentication of peer devices if the presence of an attacker in the same room can be ruled out by other means.
- If an attacker has access to the same room as the peer devices, then US ranging as such is not safe for further limiting the communication channel.
- Angle-of-arrival can be used to further constrain the communication channel, and to limit the possibility of an attack to attacker positions approximately in line with the peer devices; only cases E3 and E4 remain to be addressed. Verifying angle-of-arrival also prevents attacks from outside the room that may be relying on reflections e.g. at half-open doors.

4. Application-level threats

In this section we extend our analysis to review application-level threats for devices that use RF in combination with ultrasonic communication and sensing for authentication of peers. For our discussion we assume A to be a device operated by a user, and B to be a target device selected by the user for association with their device. B may be a device in the environment, or the device of another user.

1. *Replacement*: The first threat on application-level is for the attacker E to virtually replace the intended target device (say B), to the effect that A authenticates E instead of the actual target. For E to achieve this attack, they need to first 'silence' B so that B does not emit ultrasound and remains undetected by A (see threats a and f as discussed in the previous section). E further needs to pass potential verification of its position, which it can achieve by manipulating the distance at which it would be sensed by A (threats b and e, or g), and by positioning itself in line with A and B (cases E3 and E4, threat h). In this attack, interaction occurs only between A and E, and no interaction occurs with B. This scenario is limited to situations where the user does not expect a human-verifiable response from B in the process of interaction.

2. *Asynchronous MITM*: An asynchronous MITM attack occurs when the attacker E first achieves authentication with A (replacing B as described above), and in a second step with B (without the need to pretend to be A, and thus without positional constraint). E can then intercept messages from A, and forward them to B, to ensure a response of B as expected by the user. An example for such a situation would be printing: A, when sending a document to a printer B, expects it to print shortly afterwards. In this scenario, the attacker avoids detection by forwarding intercepted messages. However, the scenario requires that B does not verify the sender of the messages (only A authenticates B, but not the other way around).

3. *Synchronous MITM*: For a synchronous MITM attack E must achieve to establish itself between A and B on both the RF and the US channel, to appear to A as B and vice versa. If A and B use angle-of-arrival in the authentication process, E will only be able to achieve this attack if positioned literally in the middle between A and B (case E4).

Spontaneous interactions commonly occur in unknown and open environments, and generally it will not be possible to rule out the presence of an attacker in close enough proximity, and not blocked by walls, in order to threaten peer authentication. For peers to guard further against attacks, sensing of angle-of-arrival can be used to significantly constrain the positions from which attacks remain possible (leaving cases E3 and E4).

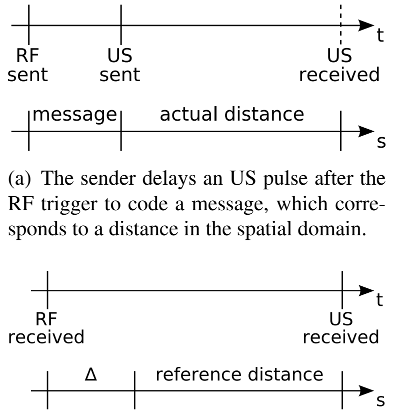
If we assume A as a user's device to be mobile but B to be a stationary device, such as a printer, then it might be plausible that an attacker positions itself strategically, to be in line between the stationary device and a likely user position. A scenario E3 might be excluded when B is mounted to a wall, or placed against wall. However, if B is also a mobile device, carried by another user, then it will generally be more difficult and less likely that an attacker achieves to position themselves between A and B. It could be argued that a user would naturally detect any device positioned between its own device A, and a target device, but it has to be noted that attacks would be possible with very small wireless sensor nodes.

If we accept the possibility of a malicious sensor node E directly between A and B then we need to consider more closely the node's attacking capability, in particular for cancellation and replacement of US messages in transit (threat j). If A sends a US message triggered over RF, E will need to cancel the message with anti-ultrasound, and generate its own US message directed at B. The smaller E is, the less time E has to replace a message to reach B within the expected time-of-flight, because of the time it takes the US messages to pass by E. For example, if E measures only a few centimetres so not to conspicuous, then it will only have a few hundred microseconds for the computations required for modifying the pulse, which, in current wireless sensor hardware, will not be sufficient. Moreover, given the propagation characteristics of ultrasound, it would not seem plausible that pulses can be cancelled without noise effects that would allow to uncover the attack.

5. Authentic communication of short messages over an ultrasonic channel

In this section we introduce a method for communication of short messages over ultrasound, effectively coding bits as distance quantities, in a way that ensures decoding only to be possible by a receiver who is positioned at an expected distance from the sender. This method is designed to overcome the problem that ultrasonic ranging as such is open to attack, and allows effective use of distance estimates to constrain ultrasonic communication. As for the application-level threat discussed, it can be used in conjunction with angle-of-arrival verification, in order to safeguard against attack in cases E3 and E4 (providing the 'additional measure' referred to in table 1, under the assumption that the attacker is not able to modify messages in transit).

Our method requires in a first step, that the authenticating devices take a reference measurement of their distance, and that this measurement is verified by a user. This can be elegantly achieved if all devices discovered in ultrasonic sensing range are displayed in a map corresponding with their real-world placement, as proposed in [9] for seamless interaction with across devices.



(a) The sender delays an US pulse after the RF trigger to code a message, which corresponds to a distance in the spatial domain.

(b) The receiver measures time from trigger to pulse arrival, and subtracts a stored reference from the corresponding distance to retrieve the message.

Figure 2. Message transmission embedded with ultrasonic ranging.

As illustrated in Figure 2(a), a sender transmits information by delaying an ultrasonic pulse in relation to an RF synchronisation message. By delaying the US pulse, the time-of-flight and thus distance will appear larger than it is, and the virtually added distance represents the transmitted information. A receiver retrieves the virtually added distance and thus the message by subtracting the reference measurement from their actual measurement, see Figure 2(b). The receiver will only be able to retrieve the message content, if the sender's distance matches the stored reference. By retrieving the random nonce this way, it can be used in higher-level protocols as an authentic message from the remote host. Note that the transmitted information is not private; any receiver in the same room will see the same virtually added distance, in comparison to previous measurements.

The authenticity of the proposed channel is created by delaying US pulses, but only provided E does not know the transmitted information beforehand. We propose that the channel can be used for transmitting nonces as part of an authentication protocol such as the MANA I protocol [6]. For authenticity of messages, both the distance and the angle must match the expectations of the receiver. We have already argued that the attacker will not be able to manipulate angle-of-arrival measurements, but E could still be positioned in between A and B (case E4). E could also create US pulses so to appear to come from A's or B's position, but only if it knows when the pulse would be sent. This though depends on the message content, and in the case of nonces would be random. When E introduces its own pulses, the received message will be different from the nonce that the sender transmitted, and authentication protocols can be constructed to detect this. The random element and distance-based coding make the US channel authentic. We have implemented a concrete authentication protocol using this property in conjunction with an interlock protocol and based on an existing peer-to-peer US sensing platform [12].

6. Conclusions

In this paper we have analysed and discussed security properties of ultrasound as out-of-band channel in the context of peer device authentication. We identified potential threats to ultrasonic communication and sensing in dependence of attacker position, and analysed how these translate to application-level threats. A particular observation is the vulnerability of ultrasonic ranging to manipulation. To address this problem, and to provide an authentic out-of-band channel, we proposed a new method for distance-coded communication over ultrasound.

Our proposed method of piggy-backing information on single ultrasound pulses makes the US channel authentic, and thus protects against synchronous MITM attacks even when assuming far-reaching attacker capabilities. Protecting against asynchronous MITM attacks requires changes to the application, e.g. to light an LED when the infrastructure device is engaged in an interaction. Then a user could notice the delay between the two interactions and abort the transaction.

7. Acknowledgements

This work is partly supported by the EC under contract 013790 "RELATE", and by EPSRC under grant GR/S77097/01. We thank Mike Hazas for discussions on details of ultrasound propagation characteristics.

References

[1] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proc. of the 2002 Network and Distributed Systems Security Symposium (NDSS'02)*. The Internet Society, Feb. 2002.

[2] S. Brands and D. Chaum. Distance-bounding protocols (extended abstract). In *Theory and Application of Cryptographic Techniques*, pages 344–359, 1993.

[3] J. Clulow, G. P. Hancke, M. G. Kuhn, and T. Moore. So near and yet so far: Distance-bounding attacks in wireless networks. In *Proc. ESAS 2006: 3rd European Workshop on Security and Privacy in Ad hoc and Sensor Networks*, pages 83–97. Springer, 2006.

[4] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[5] T. Funkhouser, N. Tsingos, and J.-M. Jot. Survey of methods for modeling sound propagation in interactive virtual environment systems. *Presense and Teleoperation*, 2003.

[6] C. Gehrmann, C. J. Mitchell, and K. Nyberg. Manual authentication for wireless devices. *RSA Cryptobytes*, 7(1):29–37, 2004.

[7] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *Proc. ACM/IEEE MobiCom '99*, pages 59–68, New York, NY, USA, 1999. ACM Press.

[8] M. Hazas. Personal communication, 2006.

[9] M. Hazas, C. Kray, H. Gellersen, H. Agbota, G. Kortuem, and A. Krohn. A relative positioning system for co-located mobile devices. In *Proc. ACM MobiSys 2005*, pages 177–190, New York, NY, USA, June 2005. ACM Press.

[10] T. Kindberg and K. Zhang. Validating and securing spontaneous associations between wireless devices. In *Proc. 6th Information Security Conference (ISC'03)*, pages 44–53. Springer, October 2003.

[11] T. Kindberg, K. Zhang, and N. Shankar. Context authentication using constrained channels. In *Proc. IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 14–21. IEEE Computer Society, June 2002.

[12] R. Mayrhofer, H. Gellersen, and M. Hazas. An authentication protocol using ultrasonic ranging. Technical Report COMP-002-2006, Lancaster University, October 2006.

[13] M. Minami, Y. Fukuju, K. Hirasawa, S. Yokoyama, M. Mizumachi, H. Morikawa, and T. Aoyama. DOLPHIN: a practical approach for implementing a fully distributed indoor ultrasonic positioning system. In *Proc. of Ubicomp 2004*, pages 347–365, Nottingham, UK, Sept. 2004. Springer.

[14] N. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proc. ACM MobiCom '00*, August 2000.

6 Shake well before use: Authentication based on Accelerometer Data

Authors: Rene Mayrhofer and Hans Gellersen

Published in the proceedings of Pervasive 2007, the 5th International Conference on Pervasive Computing, pp. 144–161, Springer-Verlag Lecture Notes in Computer Science volume 4480
ISBN 978-3-540-72036-2
May 2007, **awarded best Pervasive 2007 paper**

Shake Well Before Use: Authentication Based on Accelerometer Data

Rene Mayrhofer and Hans Gellersen

Lancaster University, Computing Department, South Drive, Lancaster LA1 4WA, UK
{rene,hwg}@comp.lancs.ac.uk

Abstract. Small, mobile devices without user interfaces, such as Bluetooth headsets, often need to communicate securely over wireless networks. Active attacks can only be prevented by authenticating wireless communication, which is problematic when devices do not have any a priori information about each other. We introduce a new method for device-to-device authentication by shaking devices together. This paper describes two protocols for combining cryptographic authentication techniques with known methods of accelerometer data analysis to the effect of generating authenticated, secret keys. The protocols differ in their design, one being more conservative from a security point of view, while the other allows more dynamic interactions. Three experiments are used to optimize and validate our proposed authentication method.

1 Introduction

Applications envisioned for ubiquitous computing build upon spontaneous interaction of devices, such that a device can make serendipitous use of the services provided by peer devices that may not be known a priori. In many scenarios, it will be desirable to verify and secure spontaneous interactions in order to ascertain that devices become paired as intended and protected against attacks on their wireless link. In a managed network environment, device-to-device authentication would be based on prior knowledge of each other or access to a trusted third party, but neither can be assumed to be available in wireless ad hoc networks for ubiquitous computing. As a consequence, secure device pairing requires the user to be in the loop, for example to enter a shared secret such as a PIN code into both devices. A challenge is to find mechanisms for users to pair devices that are not only secure but also scale well for use in ubiquitous computing. Specific challenges are that devices will, in many cases, be too small to reasonably include key pads and displays, and that required user attention must be minimal to be acceptable for spontaneous and short-lived interactions.

Pairing of a mobile phone with a headset for interaction over a wireless channel is a familiar example: we would like to achieve such interaction in a spontaneous manner (i.e. not requiring pre-configuration of phone and headset for each other) but also ensure that it is secure. The wireless communication channel between the devices is susceptible to attacks ranging from eavesdropping to

A. LaMarca et al. (Eds.): Pervasive 2007, LNCS 4480, pp. 144–161, 2007.
© Springer-Verlag Berlin Heidelberg 2007

man-in-the-middle (MITM). If an attacker were successful in establishing themselves between, in this case, phone and headset, during the pairing process, then they would obtain complete control over all phone calls. To safeguard against such attacks, a so-called *out-of-band channel* is used during pairing in order to authenticate communication over the primary channel. The out-of-band channel must be limited such that it is user-controllable that only the intended devices can communicate over it for the purposes of authentication. Note that authentication and the subsequent pairing can be anonymous or "ephemeral" [1], i.e. based on information only shared over the out-of-band-channel rather than actual device identities.

In this paper we contribute a method for device-to-device authentication that is based on shared movement patterns which a user can simply generate by shaking devices together. Using embedded accelerometers, devices can recognize correlation of their movement and use movement patterns for authentication. From a user perspective, jointly shaking is a simple technique for associating devices [2]. In our method, it simultaneously serves as out-of-band mechanism. Shaking has a number of characteristics on which we can build for our purposes:

- It is *intuitive*. People are familiar with shaking objects as manual interaction that does not require learning, for instance from shaking of medicine, or musical instruments. This means that shaking is unobtrusive in the sense that it does not require the user's full attention while being performed.
- It is *vigorous*. While there are many motion patterns that could be performed with two devices, shaking tends to produce the highest continuous acceleration values. While bouncing will produce larger accelerations, they only occur as short spikes. Shaking provides acceleration larger than most activities – and can thus be detected by simple thresholding – for as long as necessary to pair devices (and as long as the user will not get tired).
- It is *varying*. As we will show below in our first experiment (in section 7.1), the activity of shaking can be surprisingly different for different people. We do not use shaking patterns as identification, but still benefit from large differences in acceleration values, because this generates high entropy from an attacker's point of view.

It is important to note that users do not have to follow a particular pattern of shaking but that they can shake as they like; we do not attempt to identify people by their shaking patterns, but use it as a source of shared device movement.

We contribute two protocols that combine cryptographic primitives with accelerometer data analysis to establish secure wireless channels by creating authenticated secret keys. The two protocols achieve this aim differently: the first is based on Diffie-Hellman key agreement and authentication of this key, uses a conservative and better known design, provides better security and allows more flexibility in comparing accelerometer time series; the second generates cryptographic key material directly out of accelerometer data streams, is computationally less expensive and thus easier to implement on resource limited devices, and allows more dynamic interactions and group authentication.

Both protocols use standard techniques of sensor data processing and time series analysis: sampling, alignment, and feature extraction. After extracting appropriate features, our cryptographic protocols ensure that authentication is only possible if both devices have access to the same feature values. Specifically, they protect against MITM attacks on the wireless communication channel by using additional information gathered from the extracted features. This approach is general, so that other sensors than accelerometers can be used with similar methods, apart from changes in domain-specific heuristics. Sensor-based authentication offers potential benefits to small, mobile devices that communicate wirelessly and do not have traditional user interfaces. Examples are mobile phones, smarts cards, key fobs, and generally accessories like headsets, watches, or glasses.

2 Related Work

First concepts on secure device pairing suggested direct electrical contact [3], while other suggestions to implement an out-of-band channel include a "physical interlock" and the "Harmony" protocol [4], ultrasound [5], visual markers and cameras [6], audio messages [7], the GSM short message service (SMS) [8], key comparison, distance bounding and integrity codes [9], or manual input [10,1]. The DH-DB protocol proposed in [9] might also be applicable to an interactive challenge-response scheme based on sensor data such as accelerometer data. These approaches, with the exception of using camera phones, have in common that they scale poorly from a user point of view. That is, they tend to be obtrusive and require the user's attention. In our approach, we implement a low bandwidth private channel over similar accelerometer readings, and use it for authenticating a device pairing.

The idea of shaking two (or multiple) devices together to pair them has first been described as "Smart-Its Friends" [2]. We use the same interaction technique but extend it to include secure authentication. Castelluccia and Mutaf presented a protocol for pairing CPU-constrained wireless devices under the assumption of anonymous broadcast channels [11]. To achieve this property of source indistinguishability, they argue that devices engaging in this authentication protocol should be shaken and rotated randomly around each other. This shaking serves to prevent signal strength analysis, but is, in contrast to our work, not used directly as input to the authentication protocol. Hinckley presented an implementation of "synchronous gestures" [12] as a means of user interaction. By correlating accelerometer time series on devices connected via WLAN, bumping them together or tilting them can be detected and used as user input. Bumping is one possible user interaction for starting the pairing process, i.e. a trigger for our authentication method. Another closely related work was presented by Lester et al. [13] and describes how to determine if two devices are carried by the same person.

3 Design of the Acceleration-Based Pairing Method

Figure 1 shows our architecture for authenticating device pairings with shaking patterns. Both protocols make use of the same three pre-processing tasks 1 to 3. They are executed locally on each device and result in "active" time series segments of equidistant samples. Our two protocols differ in tasks 4 and 5, which can both be interactive, i.e. communicate with the remote device to which the pairing is in process.

For protocol 1, tasks 4.1 and 5.1 are actually executed in parallel: after generating a secret key with standard Diffie-Hellman (DH) key agreement (which is the first phase of task 5.1), the devices exchange their time series segments via an interlock protocol. Then they compare their locally generated segment with the one received from the remote device to check if they are similar enough. If they pass this check, the second phase of task 5.1 derives the secret session key that will be used for consecutive secure communication. This design is conservative from a security point of view and, due to the non-interactive feature extraction and comparison, allows the devices to use different means of verification. The disadvantage of splitting task 5.1 into two phases is potentially a larger delay for authentication, and the disadvantage of using DH is higher computational load.

Protocol 2 executes its tasks 4.2 and 5.2 in order: discrete (in contrast to the real-valued samples) feature vectors are extracted in task 4.2, which act as input to the interactive key agreement in task 5.2. This is an iterative process. In each time step, feature vectors generated by 4.2 are checked for matches in task 5.2. After sufficient iterations, a secret shared key can be generated out of the collected matching feature vectors in task 5.2. This design has the advantages of more dynamic key agreement, with devices being able to "tune into" other device's key streams, and of being less computationally expensive. On the other hand, it does not provide forward secrecy and protection against offline attacks as protocol 1 does, and is more unconventional and thus less well studied from a security point of view.

For both protocols, there is a trade-off between usability and security that can be exploited by applications and users depending on their requirements. Tasks 4 and 5 are described in more detail in sections 5 and 6, respectively.

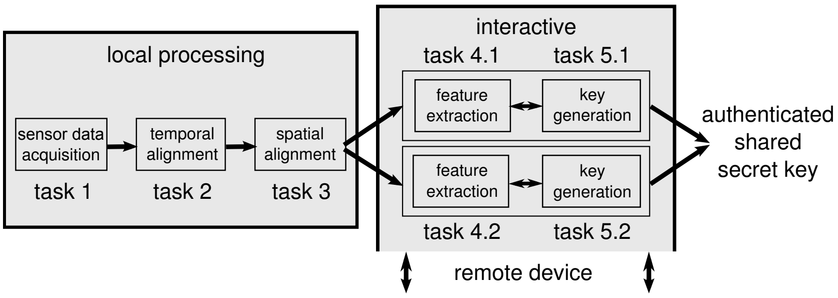


Fig. 1. Architecture for both authentication protocols

4 Pre-processing of Accelerometer Data

The three pre-processing tasks, executed as consecutive steps, are used to sample and segment the sensor data so that feature extraction can build on normalized time series.

Task 1: Sensor data acquisition. This first task is conceptually straight forward, but requires careful implementation. Sensor data is assumed to be available in the form of time series of acceleration values in all three dimensions, sampled at equidistant time steps. These must be taken locally and not be communicated wirelessly — for security purposes, it is critical not to leak any of this raw data, which can be difficult considering the possibility of powerful side-channel attacks (see e.g. [14]). Our practical experience shows a sample rate between 100 and 600 Hz to be appropriate.

Task 2: Temporal alignment. As the two devices sample accelerometer time series independently in task 1, we require temporal synchronization for comparison. We assume that devices are equipped with sufficiently accurate real-time clocks, so that differences in sampling rates and drift will not be issues. This reduces temporal alignment from an arbitrarily complex problem to *triggering* the authentication procedure and to *synchronizing* the starting points for time series comparison.

Triggering can be *explicit* by direct user input, e.g. pressing an "authenticate now" button on both devices within a short time frame or bumping both devices against the table or each other, or *implicit*, simply by starting to shake both devices. We prefer the second protocol due to its ease of use, although it is more difficult to implement. Synchronization can be at a *sample level*, i.e. within less than half the sample width, or at an *event level*, i.e. based on the onset of detected (explicit or implicit) events with the respective device. We use the latter, because it does not require time synchronization between the devices — shaking events can be detected locally at each device without communication, which is beneficial from a security point of view.

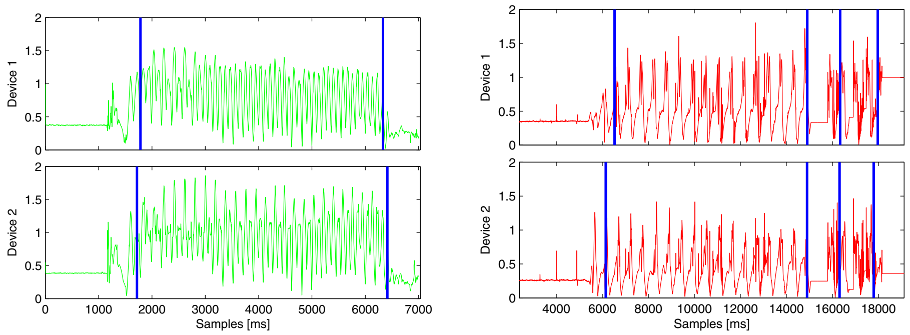
For both triggering and synchronization, we detect motion and align those parts of the time series where shaking is detected, which we call *active segments*, by their start times. Segments are considered active when the variance of a sliding window exceeds a threshold. Practical experiments show good results at a sample rate between $f = [128; 512]$ Hz with a sliding window of $v = f/2$ samples, i.e. 1/2 second, and a variance threshold around $T_\sigma = 750$.

Task 3: Spatial alignment. Shaking is inherently a three-dimensional movement. In addition to the need to capture all three dimensions, the alignment between the two devices is unknown. This means that the three dimensions recorded by the two devices will not be aligned, which is a hard problem in itself. Lukowicz et al. describe how to calibrate three-dimensional accelerometers without user interaction during stable periods [15]. However, since we are interested in the active phases and have to assume that the alignment of the devices changes during the transition,[1] we can not directly apply this result. Instead, we reduce the three dimensions to a single: by taking only the magnitude over all normalized dimensions, i.e. the length of the vector, we solve the alignment problem. This approach requires considerably less resources than other methods such as principal component analysis (PCA) or modeling using domain-specific knowledge.

The result of these steps is that, when shaken together, both devices will extract active segments of one-dimensional acceleration magnitude vectors. Even without synchronized clocks, the start times of these independent time series are typically synchronized within a few samples (on the event level).

5 Feature Extraction for Authentication Purposes

Two devices that are shaken together will experience similar, but not exactly the same movement patterns. Even assuming noise-free sampling of accelerations, the two accelerometers must have physically separate centers. Whenever rotation is part of the movement, these separate centers will necessarily experience different accelerations, thus causing different sensor time series even if the devices remain fixed in relation to each other. The problem of verifying that two devices are shaken, or more generally, moved together therefore becomes a classification problem. Figure 2 shows examples of spatially aligned sensor time series used as input to feature extraction with detected borders of active segments.



(a) Two devices shaken by one person in the same hand

(b) Two devices shaken by two people, one each

Fig. 2. Example time series after spatial alignment with detected active segments

In deciding if time series are similar enough for authentication, the aim of the feature extraction task is twofold: a) to extract feature values that are robust to small variations in the shaking patterns and to sampling noise and b) to extract

[1] When a user picks up the two devices to shake them, they will most probably be aligned differently in their hand than they were before picking them up.

a sufficiently large feature vector for use in the authentication protocol. In our approach, the feature vector will be used to authenticate a key or to directly generate a key, and thus it needs to be of high entropy from an attacker's point of view, i.e. involve a large amount of uncertainty.[2] As indicated in section 1, we argue that shaking is an appropriate movement for creating entropy: it creates varying sensor readings, because it is one of the human movement patterns that includes the highest frequency components. Slower movements will intuitively not generate as much entropy.

There is an extensive body of literature on feature extraction from accelerometer data. Particularly relevant to our problem are the described uses of the coherence measure by Lester et al. [13] and cross-covariance by Aylward et al. [16]. Both suggest sliding, windowed variances on each device for activity detection, as used in our current implementation for task 2. Huynh and Schiele compare different features for activity recognition and suggest the use of quantized FFT coefficients [17]. For task 4, we select the most promising of the recently suggested features: coherence and quantized FFT coefficients.

5.1 Coherence

We adopt the approach that was previously used by Lester et al. to distinguish between two devices worn by the same person (on different parts of the body) and two devices worn by two people walking in-step. They used coherence averages and showed that simple, non-calibrated, cheap accelerometers are suitable for analyzing human motion. Coherence is approximated by the magnitude squared coherence (MSC) as

$$C_{xy}\left(f\right) = \frac{P_{xy}\left(f\right)}{P_{xx}\left(f\right) \cdot P_{yy}\left(f\right)}$$

with (cross-) power spectra

$$P_{xy}\left(f\right) = \frac{1}{n}\sum_{k=0}^{n-1} x_k\left(f\right) \cdot \bar{y_k}\left(f\right)$$

computed over FFT coefficients $x_k\left(f\right) = FFT\left(a_k\left(t\right) \cdot h\left(t\right)\right)$ and $y_k\left(f\right) = FFT\left(b_k\left(t\right) \cdot h\left(t\right)\right)$ using the standard von-Hann window $h\left(t\right) = \frac{1-\cos(2\pi t/w)}{2}$ That is, it is computed as the power spectrum correlation between two signals split into n (optionally overlapping) averaged slices a_k and b_k of the signals a and b, respectively, normalized by the signal power spectra. Note that, although the signals a and b in time domain are real, their FFT coefficients x and y are complex. By using squared magnitudes, C_{xy} is also real-valued. By $\bar{x}$ we refer to the conjugate complex of x. Because the significance of coherence values depends on the number of averaged slices n – the more slices, the lower the coherence

[2] The authentication protocol is said to be computationally secure if an attacker's entropy of the key approaches the key length, which is typically 128 bits.

values are for the same signals –, we reduce longer time series to a maximum length of 3 seconds. This is a compromise between sufficient variability for robust classification and quick user interaction. The final value is computed simply by averaging up to a cut-off frequency f_{max}

$$C_{xy} = \frac{1}{f_{max}} \int_{0}^{f_{max}} C_{xy}(f)\, df$$

With this heuristic, we threshold C_{xy} to create a binary decision of similarity for our authentication protocol. As explained below, our experiments have shown that, with a sampling rate of $r = 256\,\mathrm{Hz}$ and windows of $w = 256$ samples with an overlap of 7/8 and a cut-off frequency of $f_{max} = 40\,\mathrm{Hz}$, coherence provides good distinction between two devices being shaken by one person from two devices being shaken by two people, one each.

5.2 Quantized FFT Coefficients

Coherence is a powerful measure of similarity, but, due to its use of continuous values, does not lend itself to directly creating cryptographic key material out of its results. Keys must be bit-for-bit equal, and thus be based on discrete instead of continuous values. By retaining basic features of the coherence measure and condensing them into discrete feature vectors, we can use those for a different way of comparing two accelerometer time series. Coherence is based on FFT coefficients, so it seems logical to quantize them into discrete values.

Huynh and Schiele compared different features with different window sizes and found that pairwise adding of neighboring FFT coefficients and grouping into exponential bands performed best in recognizing activities with moderate to high intensity levels, while other features like pairwise correlation or spectral energy were worse [17]. They also reported that the highest FFT peaks could generally be found up to the tenth coefficient, which backs our own findings that coefficients above 20 Hz do not contribute significantly.

We compared four variants of FFT-based feature vectors: linearly or exponentially quantized coefficients used either directly of added pairwise. Our experiments have shown that pairwise added, exponentially quantized FFT coefficients performed best, as also suggested in [17]. When aiming for equivalence of feature vectors, there is however an additional complication: small differences of values near the boundaries of quantization bands can lead to different feature values, although the FFT coefficients are only marginally different. Our solution is to quantize each FFT vector into multiple *candidate* feature vectors with different offsets. These offsets range from 0 to the value of the smallest quantization band. The similarity criteria in this case is simply the percentage of matching candidate feature vectors out of all vectors sent to another device. Thresholding this percentage produces a binary decision for the authentication protocol. We achieved best results for distinguishing shaking by one person from shaking by two people, one device each, with $b = 6$ exponentially scaled bands for quantization, $k = 4$ candidates, and a cut-off frequency of $f_{max} = 20\,\mathrm{Hz}$ at a sampling rate of $r = 512\,\mathrm{Hz}$ with FFT windows of $w = 512$ samples, overlapping by 50%.

6 Authentication Protocols

The two feature vectors generated in task 4 constitute, if equivalent, a shared secret password. This shared string is not directly suitable to act as a secret key for cryptographic primitives, because it is neither of defined length (e.g. 128 bits) nor distributed uniformly. But it is possible to create a cryptographically secure secret key via interactive protocols, authenticated by the feature vectors.

The choice of features directly influences requirements on the cryptographic protocols. To compute the coherence measure, both vectors need to be available completely to both devices.[3] Therefore, the time series must be exchanged during the interactive protocol — in a way that does not reveal them to an attacker. Our first authentication protocol uses asymmetric cryptography to achieve this.

Feature vectors composed of quantized FFT coefficients, on the other hand, do not allow for additional differences — authentication should only proceed if both vectors are bit-for-bit equal. The advantage is that cryptographic key material can be created using only symmetric cryptography, which is more suitable for embedded devices.

For the formal descriptions of our protocols, we use the following notation: $c = E(K, m)$ describes the encryption of plain text m under key K with a symmetric cipher, $m = D(K, c)$ the corresponding decryption, $H(m)$ describes the hashing of message m with some secure hash, and $m|n$ the concatenation of strings m and n. The notation $M[a : b]$ is used to describe the substring of a message M starting at bit a and ending at bit b. The symbol $\oplus$ describes bit-wise XOR and $|S|$ the number of elements in a set S. If a message M is transmitted over an insecure channel, we denote the received message $\widetilde{M}$ to point out that it may have been modified in transit, by noise or attack. C refers to some publicly known constant. We use AES as a block cipher for E and D and SHA_{DBL}-256 as a secure hash for H, which is a double execution of the standard SHA-256 message digest to safeguard against length extension and partial-message collision attacks [18] and is defined as $\text{SHA}_{\text{DBL}}\text{-256} = \text{SHA-256}\left((\text{SHA-256}(m))\,|m\right)$.

6.1 Protocol 1: Diffie-Hellman and Interlock*

Fig. 3 shows our first authentication protocol, which is based on a standard Diffie-Hellman (DH) key agreement (introduced in their seminal article [19]) followed by an exchange of the condensed time series and comparison locally at each device.

Using DH key agreement, devices A and B generate two – supposedly – shared keys K^{Auth} and K^{Sess}, where it is impossible to infer one from the other (under the assumption that the hash function does not allow to find a pre-image). Creating two keys, one for authentication, one as session key, provides forward secrecy. Because DH is susceptible to MITM, the devices need to verify that their keys are equivalent. The unique key property of DH guarantees with a

[3] For security reasons, both devices should independently decide if authentication was successful, and thus both need to compute the coherence.

A	Message	**B**
	Choose DH key	
$x \in \{1 \ldots q-1\}$ $K_a^{Sess} :=$ $H(Y^x)$ $K_a^{Auth} :=$ $H(Y^x \vert C)$	$\xrightarrow{X := g^x}$ $\xleftarrow{Y := g^y}$	$y \in \{1 \ldots q-1\}$ $K_b^{Sess} :=$ $H(X^y)$ $K_b^{Auth} :=$ $H(X^y \vert C)$
	Split and encrypt signal	
$(i = 0 : n-1, \quad j = 1 : n)$: $a_i := a[128i : 128i + 127]$ $c_0 \in \{0, \ldots, 2^{128} - 1\}$ $c_j := E(K_a^{Auth}, c_{j-1} \oplus a_{j-1})$	$\xrightarrow{A_1 = c_0[0:63] \Vert \ldots \vert c_n[0:63]}$ $\xleftarrow{B_1 = d_0[0:63] \Vert \ldots \vert d_m[0:63]}$	$(i = 0 : m-1, \quad j = 1 : m)$: $b_i := b[128i : 128i + 127]$ $d_0 \in \{0, \ldots, 2^{128} - 1\}$ $d_j := E(K_b^{Auth}, d_{j-1} \oplus b_{j-1})$
	Interlock and reassemable	
$(i = 0 : m-1)$: $\tilde{d}_i := \tilde{B}_1[128i : 128i + 63] \Vert$ $\tilde{B}_2[128i : 128i + 63]$ $\tilde{b}_i := D(K_a^{Auth}, \tilde{d}_{i+1}) \oplus \tilde{d}_i$ $\tilde{b} := \tilde{b}_0 \vert \ldots \vert \tilde{b}_m$	$\xrightarrow{A_2 = c_0[64:127] \Vert \ldots \vert c_n[64:127]}$ $\xleftarrow{B_2 = d_0[64:127] \Vert \ldots \vert d_n[64:127]}$	$(i = 0 : n-1)$: $\tilde{c}_i := \tilde{A}_1[128i : 128i + 63] \Vert$ $\tilde{A}_2[128i : 128i + 63]$ $\tilde{a}_i := D(K_b^{Auth}, \tilde{c}_{i+1}) \oplus \tilde{c}_i$ $\tilde{a} := \tilde{a}_0 \vert \ldots \vert \tilde{a}_n$
	Compare	

if $a \sim \tilde{b}$
then $K := K_a^{Sess}$
else $K := null$
		if $b \sim \tilde{a}$
		then $K := K_b^{Sess}$
		else $K := null$

Fig. 3. Protocol 1: Diffie-Hellman key agreement followed by exchange of the complete time series via interlock*

very high probability, that, if $K_a^{Auth} = K_b^{Auth}$, there can be no attacker E with $K_{e1}^{Auth} = K_a^{Auth}$ and $K_{e2}^{Auth} = K_b^{Auth}$, and subsequently, no $K_{e1}^{Sess} = K_a^{Sess}$ and $K_{e2}^{Sess} = K_b^{Sess}$.

This verification is done with an extended *interlock* protocol. Interlock [20] is not used widely, but is an efficient (in terms of message length) method to verify that two parties share the same key. By using this key as an input to a block cipher and splitting packets in halves, a MITM can only decrypt these packets after having received both halves. The interlock protocol then demands that A and B will only send their second halves after they have received the first halves from the respective other side. This has the effect that both sides must commit themselves to their values, by sending the first halves of the encrypted blocks, before they can receive, and subsequently decrypt, the other side's message. Thus, interlock can be seen as a commitment scheme (see e.g. [21] for a definition) based on block ciphers. An attacker E is now left with only two options: either to forward the original packets, or to create packets on its own. In the former case, A and B will be unable to decrypt the messages properly, because they do not share the same key. In the latter case, E must guess the contents of the messages, and encrypt them with the appropriate keys, before it has access to the actual messages. When the messages sent by A and B have an entropy of e bits. this leaves E with a single 2^{-e} chance of remaining undetected.

The original version of interlock is suitable for messages the size of the cipher block length. Because in our case the vectors of the accelerometer sensor data, condensed into a time series of magnitudes, have arbitrary length, we introduce a slightly extended protocol that we call *interlock**. In this variant, A and B encrypt their complete messages, i.e. the (zero-padded) vectors a and b with lengths of n and m blocks, respectively, with any of the well-known block cipher modes. For our motion authentication protocol, we simply use the cipher block chaining (CBC) mode with a random initialization vector (IV). The resulting cipher texts c and d with lengths of $n+1$ and $m+1$ blocks are then split into two messages by concatenating the first halves of all cipher blocks into the first messages A_1 and B_1 and the second halves of all cipher blocks into the second messages A_2 and B_2. This ensures that E can not decrypt any of the blocks, and can therefore not even learn parts of the plain text messages.

After exchanging their messages a and b, A and B verify that $a \sim b$, that is, that they are similar enough under their chosen criteria. We use coherence as described in section 5, but other suitable features can be used without changes to the protocol. Because of this possibility, we do not try to minimize the message lengths as e.g. suggested in [13]. In fact, A and B could use completely different similarity criteria, and could still authenticate using the same protocol. This is important for practical implementations, because different generations of devices will need to be compatible with each other.

The MANA III scheme described in [10] serves a similar purpose as this protocol, but using different cryptographic primitives. While we employ a block cipher, the MANA III scheme uses a MAC. Both constructions build on a mutual commitment to an authenticator string before transmitting parts of it.

6.2 Protocol 2: Candidate Key Protocol

In our second protocol, which we call the *candidate key protocol* (CKP), the shared secret key is generated from sensor data instead of by DH. As depicted in Fig. 4, feature vectors v are hashed to generate *candidate key parts* h. If the feature extraction task produces multiple "parallel" feature vectors v^i for each time window, as suggested above in section 5, then these yield multiple candidate

A	Message	**B**
$s_a^{r_a} \in \{0, \dots, 2^{128}-1\}$	$C_a^{r_a} :=< r_a, s_a^{r_a}, h_a^1, h_a^2, \dots >$ $\longrightarrow$	$s_b^{r_b} \in \{0, \dots, 2^{128}-1\}$
For each vector v_a^i:		**For each vector** v_b^j:
$h_a^i := H(s_a \| v_a^i)$	$C_b^{r_b} :=< r_b, s_b^{r_b}, h_b^1, h_b^2, \dots >$ $\longleftarrow$	$h_b^j := H(s_b \| v_b^j)$
add $< r_a, v_a^i >$ to LH_a		add $< r_b, v_b^j >$ to LH_b
$r_a := r_a + 1$		$r_b := r_b + 1$
For each hash $\tilde{h}_b^j$:		**For each hash** $\tilde{h}_a^i$:
if $\exists < \hat{r}_a, \hat{v}_a^i > \in LH_a$		if $\exists < \hat{r}_b, \hat{v}_b^j > \in LH_b$
s.t. $\tilde{h}_b^j = H(\tilde{s}_b^{r_b} \| \hat{v}_a^i)$		s.t. $\tilde{h}_a^i = H(\tilde{s}_a^{r_a} \| \hat{v}_b^j)$
then add $< \hat{r}_a, \hat{v}_a^i >$ to $MC_{a,b}$		then add $< \hat{r}_b, \hat{v}_b^j >$ to $MC_{b,a}$
if $\|MC_{a,b}\| >= N_a$	$K_a :=< N_a, s_a^{r_a}, kh_a >$ $\longrightarrow$	if $\|MC_{b,a}\| >= N_b$
then $kh_a := H(s_a^{r_a} \| last_{N_a}(MC_{a,b}))$		then $kh_b := H(s_b^{r_b} \| last_{N_b}(MC_{b,a}))$
and $k_a := H(s_a^{r_a} \| last_{N_a}(MC_{a,b}) \| C)$	$K_b :=< N_b, s_b^{r_b}, kh_b >$ $\longleftarrow$	and $k_b := H(s_b^{r_b} \| last_{N_b}(MC_{b,a}) \| C)$
if $\exists \hat{c}_a \subseteq MC_{a,b}$	ACK kh_b $\longrightarrow$	if $\exists \hat{c}_b \subseteq MC_{b,a}$
s.t. $H(\tilde{s}_b^{r_b} \| \hat{c}_a) = \widetilde{kh}_b$		s.t. $H(\tilde{s}_a^{r_a} \| \hat{c}_b) = \widetilde{kh}_a$
then $k'_b := H(\tilde{s}_b^{r_b} \| \hat{c}_a \| C)$	ACK kh_a $\longleftarrow$	then $k'_a := H(\tilde{s}_a^{r_a} \| \hat{c}_b \| C)$
if k_a is not set $k := k'_b$		if k_b is not set $k := k'_a$
if k'_b is not set $k := k_a$		if k'_a is not set $k := k_b$
if $kh_a = \widetilde{kh}_b$		if $kh_b = \widetilde{kh}_a$
then $k := k_a$		then $k := k_b$
else $k := k_a \oplus k'_b$		else $k := k_b \oplus k'_a$

Fig. 4. Protocol 2: candidate key protocol for directly creating a secret key from common feature vector hashes

key parts h^i. The one-way hashes are a simple way to communicate that a device has generated a certain feature vector without revealing it. To make dictionary attacks harder, we use the standard method of prepending random salt values s before hashing. When B receives such candidate key parts from A, it can check its own history of recently generated feature vectors LH to check for equals. When B has generated the same feature vector, it is stored in a list of *matching key parts* MC specific to each communication partner. As soon as enough entropy has been collected in this list, B concatenates all feature vectors, appends C, hashes the resulting string, and sends a *candidate key* K to A. If no messages have been lost in transit, A should be able to generate a key with the same hash, and thus the same secret key, which it acknowledges to B. If messages have been lost, A can simply ignore a candidate key and create its own later on.

CKP is again a general protocol and can be used with any feature vectors. Here we apply it to quantized FFT coefficients, which work well for accelerometer data. A more thorough analysis of CKP itself will be provided separately.

7 Experimental Evaluation

We conducted three experiments, two to optimize parameters for the feature extraction tasks described in section 5, and one to validate our assumption of ease of use. All three experiments used four simple ADXL202JE accelerometers, two on each device, mounted at an angle of 90° so that all three dimensions could be measured with a maximum acceleration of 2 g. The accelerometers are fixed with compressed foam inside ping-pong balls (see Fig 5), and sampled at roughly 600 Hz. By choosing balls as "device" shapes and orienting the accelerometers randomly inside the balls, each data set has different orientations. The subjects were also asked to pick the devices up at the start of each sample, so that orientations change between samples. Although the accelerometers were wired to enable higher sampling rates, the attached cables were lightweight, flexible, and long enough so as not to disturb movements of subjects.

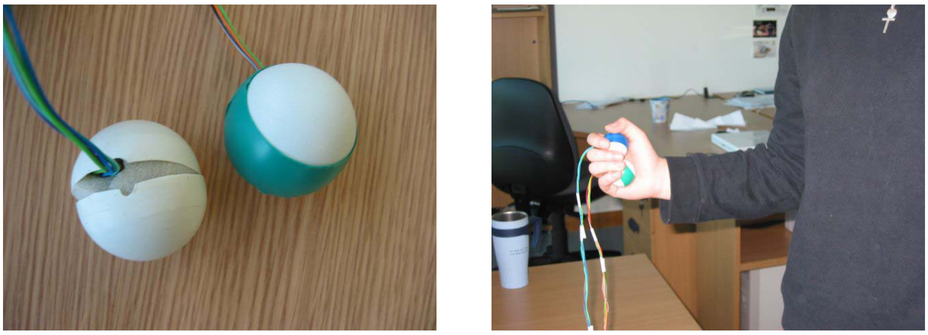


Fig. 5. Experimental setup: devices with accelerometers and subject during data collection

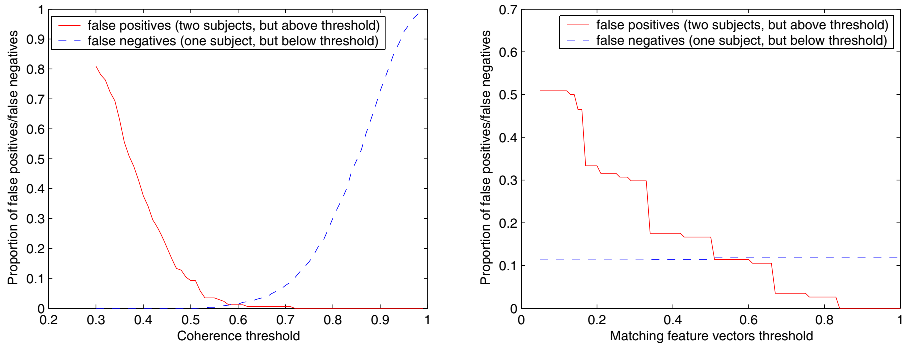
7.1 Experiment 1: Single Subjects Data Collection

The first experiment was explorative and aimed to discover how people typically shake small, lightweight objects. 51 people, 19 female aged between 20 and 55, 32 male aged between 20 and 58, of different professions, including cafeteria staff and other non-office workers, were asked to shake both ping-pong balls, explicitly without further instructions. For each subject, 30 samples of roughly 5 seconds were taken: 5 each with both balls in the left, both in the right hand, one ball in each hand, and while either standing or sitting. This extensive data set of 1530 samples shows surprisingly large differences in style, frequency, and vigor of the shaking patterns. Samples with both balls in one hand serve as our "positive" data set where authentication should be successful. The cases where one ball was shaken in each hand are "neutral": because a single person is performing the motion, authentication could, but does not have to succeed.

7.2 Experiment 2: Pairs Trying to "hack" Authentication

The second experiment served to establish our "negative" data set of cases where authentication should not be successful. It was organized as a competition with a small prize to motivate participants to try harder. The goal was for a pair of subjects to produce shaking patterns as similar as possible to each other. 8 different pairs contributed 8 complete data sets of 20 samples each and 4 incomplete sets with less samples: 5 samples each for both subjects using their left hands, both their right, one subject left, the other right, and vice versa. Each sample has roughly 15 seconds, because some time was allowed for starting the motion and synchronization. For more flexibility in moving together, the pairs were only standing but not sitting. Immediate feedback after each sample was provided to the pairs in the form of the similarity values for both protocols, so that they could adapt their shaking patterns appropriately for highest values.

Data from these two experiments was used to find parameters for detecting active segments for the temporal alignment task, and to optimize parameter combinations for the feature extraction task. The parameters for feature extraction reported in section 5 have been found by a full parameter search using this extensive data set. For coherence, we use the parameter combination that generates the maximum difference of coherence averages between all positive and negative samples. Due to the larger parameter search space with higher dimensionality, for the second protocol we use the combination that minimizes $4e_P + e_N$. e_P is the percentage of false positives, i.e. the number of successful authentications for pairs, and e_N is the percentage of false negatives, i.e. the number of authentication errors for both balls shaken in one hand. That is, false positives were weighted higher than false negatives. The values listed above in the respective sections produced optimal results on this data set. An explorative analysis of the results depending on these parameters shows that most of them are robust w.r.t. the difference in coherence averages. This suggests that even with suboptimal parameter combinations, which may be the case when using these values with different data sets, results should not deteriorate significantly.



(a) Protocol 1

(b) Protocol 2

Fig. 6. Thresholds for coherence and the number of matching FFT slices control the trade-off between false positives and false negatives w.r.t. all positive/negative samples

Figure 6 shows the trade-off between false positives and false negatives, depending on the thresholds. Unsurprisingly, the percentage of false positives decreases with increasing thresholds for both protocols. For protocol 1, shown in Fig. 6a, false negatives begin to increase noticeably at a threshold of around 0.6, while for protocol 2, shown in Fig. 6b, they remain nearly constant. The threshold, either for coherence or for the percentage of matching candidate feature vectors, can be set by the application, or possibly even by the user. From a security point of view, we obviously prefer to restrict the number of false positives to zero. With a coherence threshold of 0.72 and a threshold of 84% matching parts, we achieve false negatives rates of 10.24% and 11.96%, respectively, with no false positives. These false negatives are sufficiently low to provide user friendly interaction, as also shown by our third experiment. The feedback of a failed authentication is immediate, and users just need to shake the devices again.

There is room for improving the results for our first protocol using coherence. As explained in section 5, we only use 3 seconds for comparing the time series. If active segments are longer than this, we can choose freely which parts to use. Figure 7 shows the average coherence values for our "negative" data sets, depending on the offset of the compared time series parts. Number 1 corresponds to the first 3 seconds, number 2 to the time series between 3 and 6 seconds, etc. The graph shows that two people tend to loose synchronization the longer the common movement needs to be sustained. We could exploit this fact by skipping the first few seconds and comparing later parts, at the expense of forcing users to shake devices longer. The results given above were generated by taking the beginning of the active segments, and thus with the most difficult parts.

Data from the first experiment was also used to estimate the entropy of feature vectors used for our second protocol. Using the parameters found with the first two data sets but 256 Hz instead of 512 Hz sample rate, quantized FFT coefficient vectors were computed over all 1530 samples. This parameter combination generates feature vectors of 21 discrete values from 0 to 5. Each subject

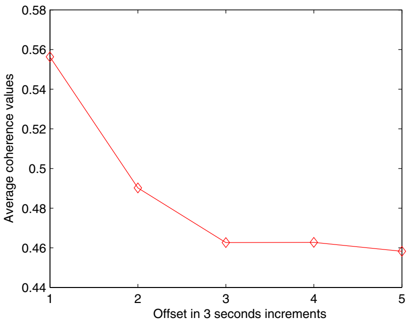


Fig. 7. Average coherence values depending on the segment offset show that people tend to synchronize their movements better at the beginning of their coordinated motions

generated on average 526.86 different feature vectors, with a minimum of 140 and a maximum of 1037. Aggregated over all subjects, there were 5595 different vectors for the left hands, 4883 for the right, and 7988 and 7770 for device 1 and 2, respectively. Overall, 12220 different feature vectors were generated during the first experiment, corresponding to an entropy of 13.58 bits per feature vector. If we assume an attacker to know which device, person, and hand are involved in a protocol run, this entropy decreases to around 7 to 10 bits, depending on the person. Overlapping feature vectors will have even less entropy, but we can still assume to generate at least 7 bits entropy per second using our second protocol.

7.3 Experiment 3: Single Subjects Live Usability Validation

The third experiment was run in "live" mode instead of data collection with batch processing, and used the same parameters. 30 subjects were asked to shake both devices in their dominant hand, with the aim of achieving successful authentication for both protocols. A simple GUI showed the status of both devices (active/quiescent) and the similarity values for both protocols, with green background if it was higher than the respective threshold and red for lower values. Subjects were asked to read a short list of tips for improving the similarity values (to align the devices roughly along the movement axis, to keep the wrist stiff, to shake quickly and vigorously, and to keep the elbow steady) and then to use interactive trial&error for achieving successful authentication. 8 of the subjects could immediately and reproducibly achieve this for both protocols starting with their first try, 8 subjects after at most 5, and 2 subjects after at most 10 tries. The remaining 12 subjects had more difficulty, but 7 could reproducibly achieve authentication after being shown once how others did it, and then within at most 3 further tries. 5 subjects only achieved authentication with either of the protocols, but not with both at the same time. This experiment shows that, even though the average rate of false negatives is low for the extensive data set from the first experiment, some people have more trouble to generate strong but similar movement patterns than others. Nonetheless, it also shows that the method is easy enough to learn within a few minutes from printed instructions and trial&error, and that it can be used intuitively after this brief learning period. The fact that a few subjects performed significantly better after being shown suggests that the printed instructions need to be improved.

8 Conclusions

We have proposed two protocols for authentication based on accelerometer data that generate secret keys between two devices when they are shaken together. Although using similar techniques for accelerometer data analysis, it is evident that the protocols achieve their aim very differently, from a security as well as from a protocol point of view. We consider the first protocol more secure, but the second to be more scalable. That is, if a large number of devices are in range of the wireless network, a device using protocol 1 may need to run Diffie-Hellman key agreement with a considerable number of other devices to find that which it is shaken together with. For the second protocol, it only needs to broadcast its candidate key parts stream, and the matching device can "tune in", i.e. synchronize, to this key stream. On the other hand, the security level of our CKP-based protocol 2 is limited to the entropy of the feature vectors, and is susceptible to offline attacks. When (pessimistically) estimating the entropy rate at around 7 bits per second, 20 seconds of shaking should be sufficient to achieve a security level of 128 bits. Users or applications may choose lower security levels.

Another potential issue in terms of security of protocol 2 is that secure hash functions, the cornerstone of our design, have been subjected to considerably less theoretical analysis than the DH construction or block ciphers which are used in protocol 1. New attacks on hash functions are being discovered [22], although the SHA-256 family of hashes, including the even more conservative $\mathrm{SHA}_{\mathrm{DBL}}$-256, is still considered secure. Additionally, protocol 1 utilizes these well-studied cryptographic primitives within a conservative design. An attacker has a one-off chance for an online attack – to guess the whole time series – and is thus significantly less likely to be successful than an offline attack on protocol 2. Although we can not currently quantify the security level against such unlikely online attacks, the security level of protocol 1 against offline attacks is 128 bits even after only 3 seconds of shaking (assuming DH to be secure). By introducing two protocols with different design, application developers can decide on this well-known trade-off between security and performance according to their requirements. Protocol 2 offers benefits for devices with limited resources, large wireless networks, and quick interaction, while we recommend using protocol 1 for higher security demands.

Feature extraction and cryptographic protocols are mostly independent of each other. Improvements in feature extraction to generate higher entropy and/or be more robust against off-center rotational effects in the movements can be used without modifying the cryptographic protocols, with the potential to significantly increase the entropy rate and thus decrease shaking time. For protocol 1, such improvements can even be distributed independently while remaining compatible to older devices. We note that our cryptographic protocols are also suitable for use with other types of sensors, while pre-processing and feature extraction tasks would most likely need to be modified.

Potential applications for our pairing protocols are manifold; coupling a mobile phone with a Bluetooth headset, establishing a transient secure connection between two smart cards for exchanging digital money, or passing access rights between key chains are prominent examples. 3D accelerometers are now being embedded into off-the-shelf mobile devices like the "Nokia 5550 Sport" and can immediately be used for authentication with our protocols. In our experiments described in section 7, we intentionally used simple, cheap accelerometers that are suitable for mass deployment.

The user interaction for authenticating devices is limited to just shaking them together for a few seconds, and is thus unobtrusive. By combining the explicit user interaction – taking two devices into one hand and shaking them as an indication that they should pair – with implicit authentication, we limit the burden placed on users. Connections are secured by default, not only as an option.

Full source code of our implementation including a demonstration application as well as our data sets are available as open source at `http://www.openuat.org`.

Acknowledgments

We gratefully acknowledge support by the Commission of the European Union under the FP6 Marie Curie Intra-European Fellowship program contract MEIF-CT-2006-042194 "CAPER", and especially thank Kristof van Laerhoven for insightful discussions on analyzing accelerometer data and providing sensor boards.

References

1. Hoepman, J.H.: The emphemeral pairing problem. In: Proc. 8th Int. Conf. Financial Cryptography, Springer-Verlag (2004) 212–226
2. Holmquist, L.E., Mattern, F., Schiele, B., A., P., Beigl, M., Gellersen, H.W.: Smart-its friends: A technique for users to easily establish connections between smart artefacts. In: Proc. UbiComp 2001, Springer-Verlag (2001) 116–122
3. Stajano, F., Anderson, R.: The resurrecting duckling: Security issues for ad-hoc wireless networks. In: Proc. 7th Int. Workshop on Security Protocols, Springer-Verlag (1999) 172–194
4. Kindberg, T., Zhang, K., Im, S.H.: Evidently secure device associations. Technical Report HPL-2005-40, HP Laboratories Bristol (2005)
5. Kindberg, T., Zhang, K.: Validating and securing spontaneous associations between wireless devices. In: Proc. ISC'03: 6th Information Security Conf., Springer-Verlag (2003) 44–53
6. McCune, J.M., Perrig, A., Reiter, M.K.: Seeing-is-believing: Using camera phones for human-verifiable authentication. In: Proc. IEEE Symp. on Security and Privacy, IEEE CS Press (2005) 110–124
7. Goodrich, M.T., Sirivianos, M., Solis, J., Tsudik, G., Uzun, E.: Loud and clear: Human verifiable authentication based on audio. In: Proc. ICDCS 2006: 26th Conf. on Distributed Computing Systems, IEEE CS Press (2006) 10

8. Nicholson, A.J., Smith, I.E., Hughes, J., Noble, B.D.: LoKey: Leveraging the sms network in decentralized, end-to-end trust establishment. In: Proc. Pervasive 2006, Springer-Verlag (2006) 202–219
9. Čagalj, M., Čapkun, S., Hubaux, J.P.: Key agreement in peer-to-peer wireless networks. IEEE (Special Issue on Cryptography and Security) **94** (2006) 467–478
10. Gehrmann, C., Mitchell, C.J., Nyberg, K.: Manual authentication for wireless devices. RSA Cryptobytes **7** (2004) 29–37
11. Castelluccia, C., Mutaf, P.: Shake them up! In: Proc. MobiSys 2005: 3rd Int. Conf. on Mobile Systems, Applications, and Services, ACM Press (2005) 51–64
12. Hinckley, K.: Synchronous gestures for multiple persons and computers. In: Proc. UIST '03: 16th ACM Symp. on User Interface Software and Technology, ACM Press (2003) 149–158
13. Lester, J., Hannaford, B., Borriello, G.: "Are you with me?" – Using accelerometers to determine if two devices are carried by the same person. In: Proc. Pervasive 2004, Springer-Verlag (2004) 33–50
14. Batina, L., Mentens, N., Verbauwhede, I.: Side-channel issues for designing secure hardware implementations. In: Proc. IOLTS: IEEE Online Testing Symp. (2005)
15. Lukowicz, P., Junker, H., Tröster, G.: Automatic calibration of body worn acceleration sensors. In: Proc. Pervasive 2004, Springer-Verlag (2004) 176–181
16. Aylward, R., Lovell, S.D., Paradiso, J.A.: A compact, wireless, wearable sensor network for interactive dance ensembles. In: Proc. BSN 2006: Int. Workshop on Wearable and Implantable Body Sensor Networks, IEEE CS Press (2006) 65–68
17. Huynh, T., Schiele, B.: Analyzing features for activity recognition. In: Proc. SocEUSAI 2005. ACM Int. Conf. Proceeding Series, ACM Press (2005) 159–163
18. Ferguson, N., Schneier, B.: Practical Cryptography. Wiley Publishing (2003)
19. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. on Information Theory **IT-22** (1976) 644–654
20. Rivest, R.L., Shamir, A.: How to expose an eavesdropper. Commununications of ACM **27** (1984) 393–394
21. Vaudenay, S.: Secure communications over insecure channels based on short authenticated strings. In: Proc. CRYPTO 2005, Springer-Verlag (2005)
22. Wang, X., Yin, Y., Yu, H.: Finding collisions in the full SHA-1. In: Proc. CRYPTO 2005, Springer-Verlag (2005)

7 Shake well before use: Two implementations for implicit context authentication

Authors: Rene Mayrhofer and Hans Gellersen

Published in the adjunct proceedings of Ubicomp 2007, the 9th International Conference
on Ubiquitous Computing, pp. 72–75
ISBN 978-3-00-022600-7
September 2007

Shake well before use: two implementations for implicit context authentication

Rene Mayrhofer and Hans Gellersen

Lancaster University, Computing Department, South Drive, Lancaster LA1 4WA, UK
{rene,hwg}@comp.lancs.ac.uk

Abstract. Secure device pairing is especially difficult for spontaneous interaction in ubiquitous computing environments because of wireless communication, lack of powerful user interfaces, and scalability issues. We demonstrate a method to address this problem for small, mobile devices that does not require explicit user interfaces like displays or key pads. By shaking devices together in one hand for a few seconds, they are securely paired. Device authentication happens implicitly as part of the pairing process without the need for explicit user interaction "just for security". Our method has been implemented in two variants: first, for high-quality data collection using wired accelerometers; second, using built-in accelerometers in standard Nokia 5500 mobile phones.

1 Introduction

Device pairing over wireless channels is insecure because of the possibility of man-in-the-middle and impersonation attacks. To safeguard against such threats, device-to-device communication needs to be authenticated. However, in the case of spontaneous interaction, only the user who initiates the interaction can distinguish between the intended target and other similar devices or malicious attackers. Such an authentication is difficult because of two reasons: many devices lack explicit user interfaces like displays and keypads that could be used to verify the device pairing, and explicit authentication does not scale when considering hundreds of spontaneous interactions a day.

Context-based authentication is one approach to address both issues, by using implicit sensory input and thus making authentication unobtrusive. Shaking two (or multiple) devices together is one option for device pairing. It has first been suggested by Holmquist et al. [1] as a possible user interaction method and subsequently been studied by others [2,3].

When devices are being shaken together, they will experience similar acceleration values. In addition to the user interaction for selecting devices to pair with, these sensor data streams can be used as input for secure authentication. Our demonstration applications build upon the authentication method introduced previously [4]. We present two specific implementations of this method. The first is used for high-quality data collection, interactive experimentation, and optimization of parameters and algorithms, and is based on wired accelerometers and laptop or desktop PCs. The second runs on off-the-shelf Nokia 5500 mobile phones and uses their embedded 3D accelerometers.

2 Authentication based on shaking

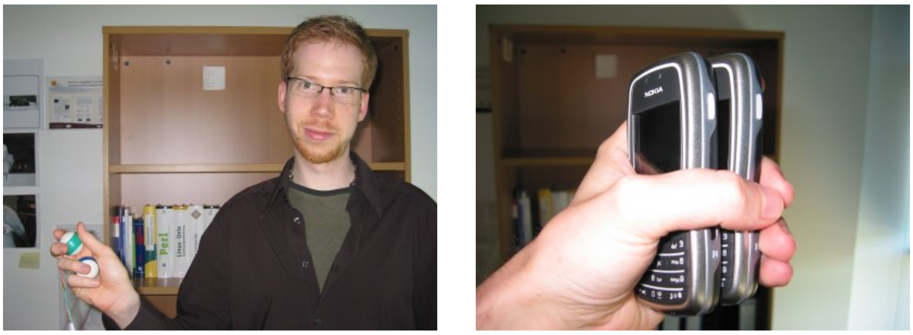
Our authentication method as described in detail in [4] consists of five tasks. The first three pre-processing tasks *sensor data acquisition*, *temporal alignment* and *spatial alignment* are executed locally and independently on each device and serve to extract and normalize *active segments*, which represent the accelerometer time series during shaking. Based on these active segments, the final two tasks *feature extraction* and *key generation* may interactively communicate with the remote device(s) over insecure wireless channels such as IEEE 802.11 wireless LAN or IEEE 802.15 Bluetooth to generate authenticated, secret shared keys on all devices shaken together.

We have proposed two different protocols for implementing the final two tasks. Both have the same aim of generating an authenticated, secret shared key from sensor time series, but achieve this with very different designs. The first protocol uses a conservative design and well-understood cryptographic primitives in two phases, key agreement and key verification. In the first phase, based on unauthenticated Diffie-Hellman key agreement, the devices agree to secret shared keys over the wireless channel. Using these keys in the second phase, they run an extended variant of the interlock protocol to exchange their recorded active segments in a way that detects man-in-the-middle attacks. Finally, the devices can locally and independently compare their local with the respective remote sensor data and verify if the pairing process should succeed or not. Our current implementation uses the coherence metric, but different devices can use different means of comparing active segments.

The second protocol is more unconventional and generates the secret shared key directly from sensor time series. To this end, the feature extraction task computes exponentially quantized, pairwise added FFT coefficient vectors over sliding windows of the active segments. These feature vectors are used as input to a *Candidate Key Protocol* (CKP) [5], which broadcasts one-way hashes of the vectors as so-called candidate key parts. If a receiving device can compute the same one-way hash, it has verified that its sensor input matches that of the sender without actually revealing it to eavesdroppers. After a sufficient number of such matching key parts have been collected, they are concatenated and hashed again to create a so-called candidate key, which is again broadcast. If one or multiple remote devices can generate the same key, it is acknowledged and can be used for subsequent secure communication. The second protocol is more dynamic and scalable, as it allows remote devices to “tune into” the key stream of another. On the other hand, the first protocol is more flexible in terms of using different methods of comparison and is considered more secure against offline attacks.

3 Implementation for data collection and experimentation

The sensor data acquisition task of our first implementation uses four ADXL202JE accelerometers, two per device mounted at an angle of 90° and set to output pulse-width modulation at about 600 Hz sample rate with a maximum acceleration of 2 g. The primitive sensor boards are fixed within ping-pong balls using



(a) Implementation 1: wired accelerometers for high-quality data collection

(b) Implementation 2: off-the-shelf Nokia 5500 mobile phones

Fig. 1. Two implementations of the "Shake well before use" authentication method

compressed foam so that they will remain constant inside the balls, but with arbitrary (and from the outside unknown) orientation (see Fig. 1a). All eight pulse-width modulated output channels are connected directly to a standard parallel port and polled at around 1 MHz, resulting in a resolution of around 10 bits per sample.

A simple ASCII coding of this sensor data stream acts as the interface to a Java-based implementation, where it is down-sampled to either 128 Hz or 256 Hz for the remaining two pre-processing tasks and implementations of both cryptographic authentication protocols. The first protocol uses TCP channels for communication, while the second one uses UDP multicast packets. Their respective results are displayed in a simple GUI to give immediate user feedback during interactive experimentation.

4 Implementation on off-the-shelf mobile phones

Our second implementation runs on off-the-shelf Nokia 5500 mobile phones, which feature an integrated 3D accelerometer (see Fig. 1b). A background Symbian application is started automatically and uses the Nokia Sensor API to access the accelerometer data at its pre-set sample rate of around 30 Hz [6]. It opens a TCP socket for streaming a binary coding upon request.

The remaining pre-processing tasks as well as an implementation of the first protocol using Bluetooth RFCOMM communication are contained within a Java MIDlet that connects to the TCP socket provided by the Symbian part. Bluetooth as a wireless communication channel poses two challenges for the implementation: there is no broad- or multicast, and inquiry as well as service discovery are slow for user interaction. Our implementation addresses this by performing the first phase of the protocol, namely Diffie-Hellman key agreement, opportunistically. Whenever a compatible device is found by the regular background inquiry process, an unauthenticated secret shared key is established with it. The second phase is started as soon as a valid active segment segment has been collected and exchanges it with all remote devices for which a shared key is already known. Those devices that have been shaken together will then verify that their respective active segments are similar enough and thereby authenticate the secret shared key.

5 Conclusions

We present a method for secure device pairing based on shaking devices together. Two different implementations show different aspects; while the first, wired implementation allows easier experimentation, rapid prototyping of new algorithms for comparing similarity of active segments, and high-quality data collection, the second one demonstrates that the method can be used on resource limited devices like mobile phones. Using Bluetooth as a communication channel poses new challenges, which are partially addressed by implementing opportunistic key agreement.

Our complete, open source implementations are available at `http://www.openuat.org`.

Acknowledgments

We gratefully acknowledge support by the Commission of the European Union under the FP6 Marie Curie Intra-European Fellowship program contract MEIF-CT-2006-042194 “CAPER”.

References

1. Holmquist, L.E., Mattern, F., Schiele, B., A., P., Beigl, M., Gellersen, H.W.: Smart-its friends: A technique for users to easily establish connections between smart artefacts. In: Proc. UbiComp 2001, Springer-Verlag (September 2001) 116–122
2. Lester, J., Hannaford, B., Borriello, G.: “Are you with me?” – Using accelerometers to determine if two devices are carried by the same person. In: Proc. Pervasive 2004, Springer-Verlag (April 2004) 33–50
3. Marin-Perianu, R., Marin-Perianu, M., Havinga, P., Scholten, H.: Movement-based group awareness with wireless sensor networks. In: Proc. Pervasive 2007, Springer-Verlag (May 2007) 298–315
4. Mayrhofer, R., Gellersen, H.: Shake well before use: Authentication based on accelerometer data. In: Proc. Pervasive 2007, Springer-Verlag (May 2007) 144–161
5. Mayrhofer, R.: The candidate key protocol for generating secret shared keys from similar sensor data streams. In: Proc. ESAS 2007, Springer-Verlag (July 2007) 1–15
6. Vajk, T., Bamford, W., Coulton, P., Edwards, R.: Using a mobile phone as a 'Wii like' controller. In: Proc. CyberGames 2007. (September 2007) *to appear*.

8 The Candidate Key Protocol for Generating Secret Shared Keys from Similar Sensor Data Streams

Authors: Rene Mayrhofer

Published in the proceedings of ESAS 2007, the 4th European Workshop on Security and Privacy in Ad hoc and Sensor Networks, pp. 1–15, Springer-Verlag Lecture Notes in Computer Science volume 4572
ISBN 978-3-540-73274-7
July 2007

The Candidate Key Protocol for Generating Secret Shared Keys from Similar Sensor Data Streams

Rene Mayrhofer

Lancaster University, Computing Department, South Drive, Lancaster LA1 4WA, UK
rene@comp.lancs.ac.uk
http://www.comp.lancs.ac.uk/

Abstract. Secure communication over wireless channels necessitates authentication of communication partners to prevent man-in-the-middle attacks. For spontaneous interaction between independent, mobile devices, no a priori information is available for authentication purposes. However, traditional approaches based on manual password input or verification of key fingerprints do not scale to tens to hundreds of interactions a day, as envisioned by future ubiquitous computing environments. One possibility to solve this problem is authentication based on similar sensor data: when two (or multiple) devices are in the same situation, and thus experience the same sensor readings, this constitutes shared, (weakly) secret information. This paper introduces the *Candidate Key Protocol* (CKP) to interactively generate secret shared keys from similar sensor data streams. It is suitable for two-party and multi-party authentication, and supports opportunistic authentication.

Keywords: context authentication, sensor data, cryptographic hash.

1 Introduction

Secure communication over a wireless channel is a difficult problem, especially for spontaneous interaction. Spontaneous interaction in the sense of ad-hoc communication between devices is often aimed for in ubiquitous computing [1], following its vision of seamlessly interacting with whatever services are currently available and useful. Moreover, many of these proposed devices are small, need to cope with limited resources such as memory, computational power and battery life, and do not have any conventional user interfaces such as key pads or displays. Communication is assumed to happen over shared wireless channels that are open to any device, which is necessary to enable transparent interoperability.

It is difficult to secure such interactions because we can not assume the involved devices to have any a priori information about each other. Creating a secure channel depends on an authentication step. If Alice (A) wants to interact

F. Stajano et al. (Eds.): ESAS 2007, LNCS 4572, pp. 1–15, 2007.
© Springer-Verlag Berlin Heidelberg 2007

with Bob (B)[1] and does not know anything about Bob a priori, then she will be unable to distinguish a legitimate interaction with Bob from malicious behavior by Eve (E) — Eve can simply perform a valid protocol run with Alice. Currently, there is no globally trusted public key infrastructure (PKI), and it is doubtful if there will be any. Even if there was one that would be able to sign trusted devices, it would not solve the problem of authenticating spontaneous interaction: Eve could just set up a trusted device E of her own and intercept the communication by getting A to communicate with her device instead of B. We therefore need to individually authenticate the interaction between each communicating pair of devices. Such authentication essentially aims at secret key agreement between A and B.

This problem is amplified as ubiquitous computing is expected to generate far more frequent spontaneous interactions. When using hundreds of different devices each day, conventional authentication methods like passwords or PINs fail to scale. Examples of devices that communicate wirelessly with each other are mobile phones, Bluetooth headsets, networked cameras, printers, in the near future goggles with integrated displays, and many more. We use the practical example of establishing a secure channel between a mobile phone and a Bluetooth headset without loss of generality.

Our approach is to authenticate devices based on shared context, which is manifested by similar sensor readings. Whenever two devices are in the same situation, e.g. being worn by the same person, capturing the same audio environment, or just being close to the same object, their sensors will experience similar time series. These time series can be used to implicitly authenticate a secure channel between the devices. There are multiple possibilities for authentication based on similar time series. The more conventional approach is to perform an unauthenticated (anonymous) key agreement like Diffie-Hellman [2], exchange the time series using the secret shared key via some commitment scheme, and compare if they are similar enough with an appropriate metric to prevent man-in-the-middle (MITM) attacks. However, this approach is computationally expensive and consists of two phases, which introduces an additional delay. We present an authentication protocol, the *Candidate Key Protocol* (*CKP*), which derives cryptographic key material directly from sensor data streams and utilizes only hash functions as cryptographic primitives.

In Section 2, we discuss related work and motivate the need for an authentication protocol based on conventional primitives in spite of more recent research on information theoretic security. After defining the threat scenarios that CKP is designed to deal with in section 3, we explain the approach and detailed specification of CKP in section 4. A first practical implementation using UDP multicast and initial experimental results are described in sections 5 and 6, respectively.

[1] In the context of this paper, we use A, B, and E for describing the devices that interact with each other interchangeably with the established names Alice, Bob, and Eve of the respective users. The reason is that one of the devices might be an infrastructure device, such as a printer or a display, that does not belong to any single user.

We finish with discussing the security properties and possibilities for extending the protocol in section 7.

2 Related Work

Results from two research areas are relevant to the present paper: information theoretical work in cryptography with influences from quantum cryptography, and authentication protocols inspired by practical issues, mostly from ubiquitous computing research.

Generating keys from noisy channels, or more generally, from (random) correlated information, received some attention in theoretical cryptography research, e.g. [3][4][5][6]. For a good introduction into the topic and for results for public, non-authenticated channels, we refer to [7,8,9]. These publications give interesting information theoretical results on key agreement, which no longer assume the intractability of some computational problem like the discrete logarithm problem, but provide what is often called "unconditional security". The basic concept is that, when two legitimate communication partners either have a noisy communication channel or when they have access to correlated information, then it is possible for them to agree to a secret key even when an adversary has access to their noisy channel or partial knowledge of their shared information. There are two classes of such authentication protocols: interactive, e.g. [7,8,9], and non-interactive, e.g. [6]. Non-interactive protocols have the obvious advantage that they can be used to establish a shared secret when only one-way communication is available. This has additional practical consequences. Even when two-way communication is possible, issues like time delays, packet loss, etc. can be handled more easily with non-interactive protocols. On the other hand, interactive protocols are necessary under the assumption of active adversaries (see e.g. [7, section III.D]). Our proposed protocol is interactive.

Other results [10] seem particularly promising because they describe an authentication protocol based on a weak secret key, which closely matches our real world problem of using sensor time series as a weak secret key.

However, these theoretical results do not yet seem to have been implemented, and practical applicability is therefore still limited. Another problem is that, although the shared secrets may be weak, large secrets are required to guarantee the security properties of these protocols. For small and embedded devices, it is difficult to process large strings of secret data, and it is difficult to find good sources of large secret strings in the first place.

In contrast, we use conventional, i.e. computational, cryptographic primitives based on intractability assumptions which are still assumed to hold. With possible future availability of quantum computers, these assumption may need to be revised. In this paper, we use the terminology of information theoretical cryptography as far as appropriate because of the similar aims and assumptions. When adding the assumption of non-reversibility of cryptographic hash functions, then our proposed Candidate Key Protocol can be seen as an instance of a secret key agreement based on correlated random variables.

It is not obvious how the calculus introduced in [8] for noisy channels could be applied to the case of similar sensor time series that A and B have access to and which E can get some knowledge about. Future work may use this or a similar calculus to analyze the security of CKP more analytically.

A large number of interactive protocols based on authenticated Diffie-Hellman (*DH*) key exchange [11] have recently been suggested, mostly inspired by practical problems of authentication in real world applications. This is assumed to be computationally, instead of unconditionally secure. The classical interlock protocol [12] can be seen as a predecessor of these, but it already used the notion of committing to values before revealing them. Newer protocols are mostly based on commitment schemes, e.g. the MANA family of protocols for manual string input or verification [13], optimized in [14].

While the "resurrecting duckling protocol" [15] aims at long-lived pairings, Hoepman introduced pairing protocols for short-lived interactions based on manual exchange of secrets [16][17], which scales poorly from a user point of view. The protocol proposed in [16] is very similar to MANA III [13] and seems to have been developed independently. Vaudenay claims [18] that Hoepman's protocol can not be implemented securely due to the lack of known hash functions with properties required by the protocol, and presents a protocol called SAS, which provides the same level of security with shorter shared secrets.

Creese et al. introduce a formal model for verifying authentication protocols that work with empirical verification [19]. They present the analysis of three related pairing protocols and show proofs of their security under their model.

Čagalj et al. describe three other pairing protocols with similar aims, based on short string comparison, distance bounding, and integrity codes [20]. Their second protocol is based on distance measurement, but we suggest that their scheme might be applicable to an interactive challenge-response scheme based on sensor data.

CKP is related to all these protocols because it shares similar aims, but differs in the approach. Instead of authenticating ephemeral session keys or long-term pairings created with DH, CKP creates shared keys by using sensor streams as input.

3 Threat Scenarios

In this section, we briefly outline the threat scenarios that are relevant to a device authentication protocol and to CKP in particular. Typical threats for a communication channel are *eavesdropping*, *replaying* of messages, and *deletion*, *insertion* and *modification* of messages. All of these threats are subsumed in the so-called man-in-the-middle (*MITM*) attack, where E is assumed to be "in between" A and B and have complete control over their communication channel. When an unauthenticated key agreement like Diffie-Hellman is used between A and B, E can delete all messages between A and B and instead perform two independent key agreements, one with A and one with B. In this paper, we explicitly assume an active adversary, and CKP is designed to detect when a

MITM attack is being performed and fail to authenticate in this case. However, in the general case, it is not possible to distinguish between a *benign* authentication failure when the sensor values experienced by A and B are not similar enough and a *malign* authentication failure caused by an attack.

Another typical threat is *denial of service* (DoS). This refers to E making communication, and in the scope of this paper, authentication impossible between A and B. When assuming an active adversary, DoS is easily possible and will therefore not be discussed further. However, the protocol should provide indication to the user when it can not complete, either due to benign communication error or due to a DoS attack. Distinguishing between these two cases is, again, not possible in the general case and we therefore treat them equally.

We also point out that attacks on the involved devices themselves are out of the scope of this paper and assume that the two devices A and B are trusted for the purpose of the interaction. If A trusts B with some document, but B (intentionally or due to an attack) forwards it to E, then authentication between A and B can not prevent this.

To summarize, our main threat scenario is an active attack on the (wireless) channel including full MITM capabilities. We assume that there is some sensor data which both A and B can get with better accuracy than E. Here we use the same argument as applied in [7, Theorem 5]: if Alice and Bob do not share any correlated information, then "from Bob's point of view, Alice has no advantage compared to Eve. If Eve performs the same protocol as Alice would, pretending to be Alice, Bob accepts with the same probability as he would accept a protocol execution with Alice". Assuming an experiment where Alice, Bob, and Eve can receive the same bit string over independent noisy channels, [7] concludes that "secret-key agreement against *active* adversaries is only possible if Alice's and Bob's channels are both less noisy than Eve's channel". This is to be intuitively expected, but in contrast to the results for passive adversaries [3].

We argue that this assumption is justified because, when A and B are in a similar context, their sensor time series should be more similar to each other than to the sensor time series perceived by E, even if only slightly. This can be achieved by measuring *local* physical phenomena which an adversary can not reasonably influence to obtain measurements with higher accuracy than A and B. Examples for appropriate phenomena are acceleration, sound, light, or radio frequency signal strength.

4 The Candidate Key Protocol

The candidate key protocol interactively generates secret shared keys from sensor streams between two (or multiple) devices. Figure 1 shows the relations between A, B and E. All devices are assumed to have full access to a wireless communication channel, and we explicitly assume E to be capable of deleting, inserting, and modifying messages between A and B without them being able to notice at this level. Additionally, A and B are assumed to share aspects of their context

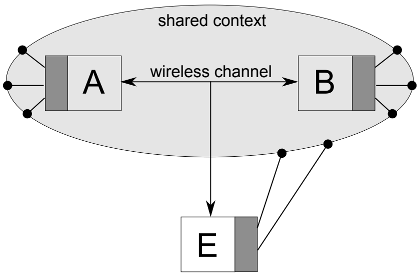


Fig. 1. Assumptions of CKP: A, B, and E share complete access to a wireless channel, and restricted access to the context in which interaction is taking place

and have sensors that can capture these aspects. E is assumed not to share the same context, but be able to access it with (similar or different) sensors with inferior accuracy.

4.1 Approach

Our approach to generating secret shared keys from similar, but not equal, sensor time series is based on the concept of *candidates*. When sampling sensor time series, raw samples are typically not used directly, but more meaningful *features* are extracted based on domain specific knowledge. We note that this step is critical for any use of sensor data, although the respective requirements depend on the application. For authentication, i.e. generating cryptographic key material, it is important for the extracted features to have high entropy from an adversary's point of view. In the terms typically used for feature extraction and context-aware systems, high entropy implies that the chosen features must clearly distinguish devices being in the same context from devices being in different contexts. The reverse, however, does not necessarily hold. Good separation in this sense does not imply high entropy, because an adversary is free to choose different features. Therefore, features should be chosen such that an adversary can learn the least amount of information about their specific values. We can not give generally valid recommendations because features are highly application specific. For the scope of this paper, we assume feature vectors to be available as input for authentication.

Even with features that perform well for a given application, there is still room for errors. To generate key material from feature vectors, we need to convert to integer values at some point; simple quantization errors can then lead to different keys even when the feature vectors are very similar. That is, quantization can increase noise. Generally, extracting and comparing feature vectors is a classification problem with the usual trade-off between separation and recall. Making features more distinctive to generate higher entropy will generate more errors in comparing feature vectors from devices in the same context, i.e. *false negatives*. Tuning the features for more robustness will make it easier for an

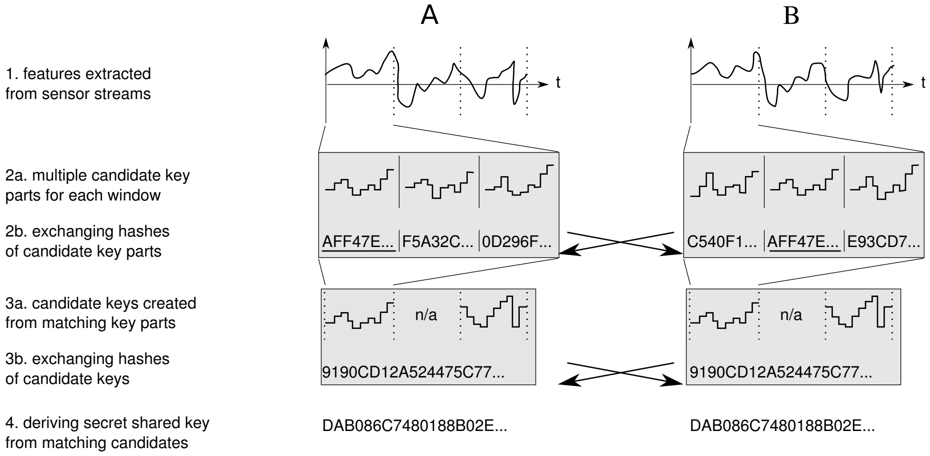

A
B
1. features extracted from sensor streams
t
t
2a. multiple candidate key parts for each window
2b. exchanging hashes of candidate key parts
AFF47E... F5A32C... 0D296F...
C540F1... AFF47E... E93CD7...
3a. candidate keys created from matching key parts
n/a
n/a
3b. exchanging hashes of candidate keys
9190CD12A524475C77...
9190CD12A524475C77...
4. deriving secret shared key from matching candidates
DAB086C7480188B02E...
DAB086C7480188B02E...

Fig. 2. Approach to generating a secret shared key: candidate key parts are time windows over extracted features from sensor time series and are concatenated to candidate keys

adversary to estimate their values, i.e. generate *false positives*. For the purpose of authentication, false positives must be strictly avoided, but when the false negatives rate is too high, the authentication method may become unusable in practice. Therefore, our approach is to allow the feature extraction step to yield multiple (parallel) feature vectors in each time step. We then use these multiple different candidates in a way that does not leak additional information to an adversary, and thereby provide a partial solution to this trade-off.

In Fig. 2 we show an overview of CKP, starting with the extracted features. The generation of multiple candidates for each feature vector is again application specific, but there exist general methods. One example is that different offsets for quantization can be used to alleviate the problem of quantization errors and thus solve a large class of false negatives. This method is depicted in Fig. 2. Every feature vector becomes a *candidate key part.* That is, it is a candidate for inclusion in the shared secret, subject to matching with the remote device. We then compute hashes of all candidate key parts for the current time step, which we abstract to a strictly monotonically increasing round number. These hashes are exchanged between A and B to verify which of the candidate key parts, if any, match. Note that transmitting their cryptographic hash values does not reveal any useful information about the candidate key parts themselves, because secure hash functions are assumed to be one-way functions. After a sufficient number of matching key parts, i.e. after accumulating enough entropy, the matching key parts are concatenated to a *candidate key.* With the possibility of multiple matches in each round, there are different ways to concatenate this key. Therefore, hashes of the candidate keys are again exchanged between A and B. When they match, A and B have successfully agreed to a shared secret.

4.2 Specification

After introducing the concepts of candidate key parts and candidate keys, we now present the detailed specification of CKP. Figure 3 defines the steps of the protocol.

Alice	Message	**Bob**
	I.1. New local feature vectors for rounds r_a, r_b	
For each vector v_a^i: $s_a^{ra} \in 0, \ldots, 2^{128} - 1$ $h_a^i := H(s_a \mid v_a^i)$ add $< r_a, v_a^i >$ to LH_a		For each vector v_b^j: $s_b^{rb} \in 0, \ldots, 2^{128} - 1$ $h_b^j := H(s_b \mid v_b^j)$ add $< r_b, v_b^j >$ to LH_b
	I.2. Exchange candidates	
	CAND $C_a^{ra} :=< r_a, s_a, h_a^1, h_a^2, \ldots >$ $\longrightarrow$ CAND $C_b^{rb} :=< r_b, s_b, h_b^1, h_b^2, \ldots >$ $\longleftarrow$	
$r_a := r_a + 1$		$r_b := r_b + 1$
	I.3. New remote candidates	
For each hash $\widetilde{h}_b^j$: if $\exists <\widehat{r}_a, \widehat{v}_a^i> \in LH_a$ s.t. $\widetilde{h}_b^j = H(\widetilde{s}_b \mid \widehat{v}_a^i)$ then add j to $ML_a^{\widetilde{r}_b}$ and add $< \widehat{r}_a, \widehat{v}_a^i >$ to $MC_{a,b}$		For each hash $\widetilde{h}_a^i$: $\exists <\widehat{r}_b, \widehat{v}_b^j> \in LH_b$ s.t. $\widetilde{h}_a^i = H(\widetilde{s}_a \mid \widehat{v}_b^j)$ then add i to $ML_b^{\widetilde{r}_a}$ and add $< \widehat{r}_b, \widehat{v}_b^j >$ to $MC_{b,a}$
	I.4. (optional) Exchange matches	
	MATCH $M_a^{ra} :=< \widetilde{r}_b, ML_a^{\widetilde{r}_b} >$ $\longrightarrow$ MATCH $M_b^{rb} :=< \widetilde{r}_a, ML_b^{\widetilde{r}_a} >$ $\longleftarrow$	
add numbers $\widetilde{ML}_b^{ra}$ of round $\widetilde{r}_a$ from LH_a to $MC_{a,b}$		add numbers $\widetilde{ML}_a^{rb}$ of round $\widetilde{r}_b$ from LH_b to $MC_{b,a}$
	II.1. Generate candidate key	
if $\lvert MC_{a,b} \rvert >= N_a$ then $kh_a :=$ $H(s_a^{ra} \mid last_{N_a}(MC_{a,b}))$ and $k_a :=$ $H(s_a^{ra} \mid last_{N_a}(MC_{a,b}) \mid C)$		if $\lvert MC_{b,a} \rvert >= N_b$ then $kh_b :=$ $H(s_b^{rb} \mid last_{N_b}(MC_{b,a}))$ and $k_b :=$ $H(s_b^{rb} \mid last_{N_b}(MC_{b,a}) \mid C)$
	II.2. Exchange candidate keys	
	KEY $K_a :=< N_a, s_a^{ra}, kh_a >$ $\longrightarrow$ KEY $K_b :=< N_b, s_b^{rb}, kh_b >$ $\longleftarrow$	
	II.3. Search for candidate key	
if $\exists \widehat{c}_a \subseteq MC_{a,b}$ s.t. $H(\widetilde{s}_b^{rb} \mid c_a) = \widetilde{kh}_b$ then $k'_b := H(\widetilde{s}_b^{rb} \mid c_a \mid C)$		if $\exists \widehat{c}_b \subseteq MC_{b,a}$ s.t. $H(\widetilde{s}_a^{ra} \mid c_b) = \widetilde{kh}_a$ then $k'_a := H(\widetilde{s}_a^{ra} \mid c_b \mid C)$
	II.4. Acknowledge key	
	ACK $\widetilde{kh}_b$ $\longrightarrow$ ACK $\widetilde{kh}_a$ $\longleftarrow$	
	II.5. Set key	

if k_a is not set $k := k'_b$
if k'_b is not set $k := k_a$
if $kh_a = \widetilde{kh}_b$
then $k := k_a$
else $k := k_a \oplus k'_b$
		if k_b is not set $k := k'_a$
		if k'_a is not set $k := k_b$
		if $kh_b = \widetilde{kh}_a$
		then $k := k_b$
		$k := k_b \oplus k'_a$

Fig. 3. Specification of CKP

For the formal description, we use the following notation: $H(m)$ describes the hashing of message m with some secure hash algorithm, and $m|n$ describes the concatenation of strings m and n. The symbol $\oplus$ describes bit-wise XOR and $|S|$ the number of elements in a set S. When a message M is transmitted over an insecure channel, we denote the received message $\widetilde{M}$ to point out that it may have been modified in transit, by noise or attack. Subscripts denote the different sides (a or b for an authentication between A and B), while superscripts denote specific vectors in a set of vectors. The syntax $\widehat{x}$ denotes the (open) result of a search for matches in a set. When a hash is computed from a set of vectors, we mean the concatenation of all vectors in some pre-defined order, typically by their round number.

v denotes raw feature vectors without cryptographic key properties, i.e. they do not need to be distributed uniformly. h denotes cryptographic hashes of feature vectors and r denotes round numbers. Each host keeps a set LH as a history of recently added local feature vectors and one set MC for each remote host to store the matching candidates as reported by this host. Any of the SHA family of hashes seems appropriate to implement H, and we currently use SHA-256 as a secure hash.

CKP consists of two phases:

I *Collecting entropy from feature vectors and determining matching candidate key parts*: In step I.1, locally generated feature vectors are stored in a local history for future reference. This history LH may be implemented as a circular buffer, overwriting oldest feature vectors. By computing the secure hash, *candidate key parts* are created from these feature vectors and sent to the remote device in step I.2. Note that each round r uses a unique salt value s^r that is prepended before hashing to make attacks with lookup tables more expensive.

In step I.3, received candidate key parts are compared with feature vectors in the local history LH. All matching vectors are advanced to the status of *matching key parts* by adding them to the set of matching candidates MC, which is specific to each remote host that CKP is run with.

Step I.4 is optional, and should only be used in asymmetrical settings. In an asymmetrical setting, only one host broadcasts candidate key parts. Any host receiving the candidate key parts and recognizing matching key parts acknowledges these matches, which enables the broadcasting host to keep track of matching key parts.

II *Generating the secret shared key*: Each host can check locally if enough matching key parts have been collected, and/or if the associated feature vectors accumulate enough entropy for a secret shared key. When the local criteria are fulfilled, a *candidate key k* and an associated *candidate key identifier kh* are generated in step II.1 by concatenating the feature vectors that belong to the last N matching key parts and, again, computing secure hashes over the concatenated string with prepended salt values. To decouple the actual key and its identifier, a public padding string C is appended before hashing for the generation of k. The candidate key identifiers are exchanged in step II.2.

In step II.3, the hosts then try to locally generate a key that matches a received candidate key identifier. This may be computationally expensive, depending on the number of matching key parts in MC, the number of matching key parts N used for the generation of kh, and the number of duplicate matches in each round. The reason is that, for example, host B has no knowledge about the exact set of matching key parts chosen by host A to generate its kh_a. Because hosts A and B may be out of sync with their round counters, it is unknown which rounds contributed matching key parts. And because A and B most probably generate candidate key parts in different order even within the same round, it is unknown which of the matches in a specific round was chosen when there were multiple. B therefore needs to try all possible combinations of N_a elements of $MC_{b,a}$, which has potentially a run time complexity of $O(A^H)$ where A is the maximum number of different candidate feature vectors generated in each round, and $H > N$ is the maximum size of the history MC. However, in practice we expect only few duplicates, and the search can be further optimized by starting with the most likely, i.e. the most recent, round numbers recorded in MC. Another possible optimization is to transmit the round and vector numbers with candidate key messages to uniquely identify the set of parts. This trade-off between message size and computational cost depends on application-specific cost models, but does not influence the security level of the protocol.

If a matching key could be generated, it is acknowledged in step II.4. After receipt of a *key acknowledge*, the hosts can start to use the generated key k that matches the acknowledged key identifier kh.

Note that at this stage, there is the possibility that the generated keys at hosts A and B are different. This can happen when hosts A and B independently generate and exchange candidate keys in steps II.1 and II.2 and the respective KEY messages overlap during transmission. Then, in steps II.3 and II.4, both hosts may find and acknowledge the respective remote host's key, again with overlapping ACK messages. That is, when host A generated a key k_1 and B a key k_2 in step II.1, then after step II.4, host A may have found and acknowledged k_2 while B may have found and acknowledged k_1. By concurrently reacting to overlapping messages, A and B have effectively swapped their keys, but are still left with different k_1 and k_2. To solve this synchronization problem locally, the hosts remember the originally generated keys and check if the received key acknowledge is different. If yes, they can simply compute the final secret shared key as the XOR of the two different keys.

In this form, CKP does not assume the communication channel to have any specific properties, because our basic assumption is a MITM with full control over this channel.

5 Implementing CKP with Lossy Channels

In a practical implementation, the communication channel may be lossy. That is, packet delivery is not guaranteed even when no MITM attack is taking place. This is the case for most broadcast radio frequency (RF) channels such as IEEE 802.11 WLAN or IEEE 802.15.4 ZigBee.

Our first implementation of CKP uses UDP as a lossy communication protocol. This has three advantages: a) UDP can be used directly between any hosts connected via an IP based network. b) UDP allows to broad- or multicast packets and can therefore be used for group authentication or spontaneous authentication as described in more detail below. c) UDP offers guarantees comparable to many low-level broadcast RF channels, thus porting our implementation e.g. to TinyOS [21], should be straightforward.

The protocol specification presented in section 4 lends itself to implementation on lossy channels, because it is robust against packet loss. When candidate key parts get lost, there will simply be no matches for the respective round. When candidate keys get lost, they can not be used to generate secret shared keys, but new candidate keys will be generated in subsequent rounds. However, issues arising from asynchronism and overlapping messages need to be dealt with at the implementation level.

There are various possibilities for asynchronism in CKP. Here we concentrate on the case where a remote message arrives for a round before the respective local action has been processed. This includes many special cases like a disparity between the system clocks or delayed processing due to multi-tasking. To cope with such asynchronism, we introduce message buffers to keep a history of recently received messages. Then, when local operations such as adding new feature vectors are processed, this history is replayed to simulate a new arrival of messages using the updated local state. This method allows to cope with asynchronism while considering limited resources in terms of memory and CPU capabilities.

Note that, among others, [7, Theorem 8] states that "perfect synchronization is impossible", i.e. that there are always some cases in which the decisions of Alice and Bob about generating a common key are different. Our implementation of CKP using UDP can only safeguard against Alice and Bob agreeing to a different shared key (i.e. a MITM attack). But, under the assumption of a completely insecure communication channel without any guarantees, it will always be possible for one host to finish the protocol with success, while the other finishes with failure. In this case, further secure communication is not possible, and the hosts can use time-outs to detect it.

6 First Experimental Results

CKP has already been applied to one specific device pairing method: implicit authentication by shaking devices together for a few seconds [22]. This method uses 3D accelerometers as input to two alternative authentication protocols, one of them being CKP. When shaking two devices with integrated accelerometers together, their sensor time series are similar enough to create a secret shared key, but an adversary can not obtain these time series with sufficient accuracy. The lower bound of the entropy rate has been estimated at about 7 bits per second [22], which means that around 20 s of shaking are sufficient to generate over 128 bits of entropy. Experiments on "human man-in-the-middle" attacks, where adversaries try to duplicate the shaking patterns of victims to produce similar sensor time series, show that people are unable to reproduce these patterns even when we allow for cooperation between adversary and victim (which would not be possible during a real attack). It remains to be shown if high-speed cameras could be used to estimate the local acceleration values and thus lower the candidate key parts entropy from an adversary's point of view.

7 Security Analysis and Discussion

When generating cryptographic key material, the most important point is to achieve high entropy with regards to a possible adversary's knowledge. A key can only remain secret if Eve is sufficiently uncertain about it. It is important to note that, principally, CKP can not increase the entropy of a secret key compared to the total entropy of all feature vectors it has been created from. Instead, any public communication between Alice and Bob must reveal something about the key — CKP can only try to make this additional information useless to Eve. Note that feature extraction and estimation of entropy are entirely application specific. We can only assume the locally added feature vectors to carry sufficient entropy, and leave it to the specific implementation to guarantee this.

Hashing the sensor time series to generate candidate key parts and candidate keys serves to reduce an adversary's usable information about them. This is often termed "privacy amplification". When we assume the SHA family of hash functions to be universal as defined in [23] and reproduced in [24], then an upper bound for the information that Eve can obtain about the secret key has been shown in [24, Corollary 5]: if Eve has access to t bits of the (weak) secret W with n bits, then her expected knowledge about a key $K = H(W)$ with a length of $r = n - t - s$ bits for some safety parameter $s < n - t$ is restricted to a maximum of $2^{-s} / \ln 2$. This assumes that W is uniformly distributed. For our application to sensor time series, W is not uniformly distributed, and a significant part of its distribution function may be known to Eve. We can only conjecture that the above corollary may be applicable to those components of the sensor time series that are completely unknown to Eve and thus uniformly distributed from her point of view, but can not currently provide a proof. This conjecture suggests that, if we intend to extract a secret shared key with a size of $r = 128$ bits, then the difference between the length of the sensor time series W and Eve's information about it, i.e. $n - t$ bits, must be larger then 128. Intuitively, this requirement is trivial. But the theoretical analysis indicates that by hashing the input, all the entropy of the weakly secret sensor time series should be retained in K. This means that transmitting the candidate key parts, which are hashes over the sensor time series, should not reveal more about them than an adversary already knew. It is currently unclear if more information about the final secret key is revealed when MATCH messages are transmitted to acknowledge matching candidate key parts, but we do not expect this to be the case. Nonetheless, the normal mutual authentication mode seems more conservative, because only the candidate key part hashes and the candidate key hashes are transmitted, but no more information about which of the candidate key parts have been used to construct the secret key.

It is important to note that there is a possibility for brute-force attack. The problem arises when parts that are extracted from sensor time series only have a small entropy from Eve's point of view. In this situation, even when reversing the hash function is impossible, she could just generate lookup tables of all possible time series parts and compare their hashes with the CAND messages. This is slightly mitigated by our use of salting, but only makes the attack more computationally expensive, and not less likely to be successful. Eve only needs to keep a small amount of possibly matching key parts in a history to try and create keys that match the transmitted KEY messages, in much the same way that is also used in the legitimate protocol run. For this reason, it is better to use less candidate key parts to construct a key. When the sensor time series parts that can be extracted naturally using domain specific knowledge only have a small entropy, then multiple such parts should be buffered and bundled into one candidate key part. Guessing a candidate key part and verifying that it matches its received hash value has an average complexity of $O\left(2^{e-1}\right)$ when the feature vector has e bits of entropy from Eve's point of view. Thus, two concatenated feature vectors would need $O\left(2^{2e-1}\right)$ steps to guess. This entropy level directly defines the security level of the whole CKP run. It has been shown in [24] that adding random material can in principle increase the length of K that can be extracted from the weak secret W, but we currently do not see a method to apply this to CKP.

Finally, there are two additional advantages of CKP over more traditional authentication protocols, e.g. ones based on public key infrastructures. First, the continuous broadcasting of candidate key parts and, after detecting matches, of candidate keys, allows remote hosts to "tune in to" another host's authentication stream. This allows to easily construct applications with *opportunistic authentication*, where hosts automatically authenticate with each other as soon as they enter a shared context: when a host picks up broadcasts from another and is able to generate matching key parts, they are guaranteed to record similar sensor readings.

Second, CKP can be trivially generalized to group authentication. In the specification in Fig. 3, only steps II.4 and II.5 need to be adapted. All hosts can continue to generate candidate key parts and candidate keys, and to search for candidate keys as for two-host authentication. However, keys can only be acknowledged and used in steps II.4 and II.5, respectively, after all hosts that should be members of the authenticated group were able to generate matching keys. A possible solution is to split step II.4 into a *tentative acknowledge* and a *group acknowledge* message, where the latter is only sent after the tentative acknowledge has been received from all group members.

8 Conclusions

Our proposed Candidate Key Protocol (CKP) is one approach to solving the problem of device-to-device authentication for spontaneous interactions. Replacing *explicit* means of authentication like manual password input or string verification with an *implicit* authentication based on similar sensor data streams offers significant advantages from a user point of view: wireless communication can be made secure by default instead of relying on a separate authentication step.

Context-based authentication is in fact a classification problem, with the known problems of false positives, which need to be strictly avoided for security reasons, and false negatives, which hinder seamless and unobtrusive user interaction. One main novelty of CKP is that multiple candidate key parts in each step can be used to address the problem of false negatives. Its advantages over other proposed approaches to the same problem and based on Diffie-Hellman key agreement authenticated by short, or weak, shared secrets are threefold: it is less computationally expensive and thus well suited for implementation with limited resources, it provides opportunistic authentication, and it is trivially extensible to group authentication. The major disadvantage is that the generated secret shared key is only as secure as the entropy of the candidate key parts and that it does not provide forward secrecy. Newer results on information theoretically secure key agreement are very promising for authentication based on sensor data streams, but have not yet been implemented in practice. Relying on conventional secure hashes allows us to implement and test CKP in real-world settings like the authentication method based on shaking devices together.

Complete source code of our current Java implementation is available at `http://www.openuat.org`, as part of the open source ubiquitous authentication toolkit.

Acknowledgments

We gratefully acknowledge support by the Commission of the European Union under the FP6 Marie Curie Intra-European Fellowship program contract MEIF-CT-2006-042194 “CAPER”.

References

1. Weiser, M.: The computer of the twenty-first century. Scientific American 1496, 94–100 (1991)
2. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. on Information Theory IT-22(6), 644–654 (1976)
3. Maurer, U.M.: Perfect cryptographic security from partially independent channels. In: Proc. STOC ’91: 23rd ACM Symp. on Theory of Computing, May 1991, pp. 561–571. ACM Press, New York (1991)
4. Juels, A., Wattenberg, M.: A fuzzy commitment scheme. In: Proc. 6th ACM Conf. on Computer and Communications Security, pp. 28–36. ACM Press, New York (1999)
5. Juels, A., Sudan, M.: A fuzzy vault scheme. Cryptology ePrint Archive, Report 2002/093 (July 2002)
6. Dodis, Y., Smith, A.: Correcting errors without leaking partial information. In: Proc. STOC ’05: 37th ACM Symp. on Theory of Computing, May 2005, pp. 654–663. ACM Press, New York (2005)

7. Maurer, U., Wolf, S.: Secret-key agreement over unauthenticated public channels — part i: Definitions and a completeness result. IEEE Trans. on Information Theory 49(4), 822–831 (2003)
8. Maurer, U., Wolf, S.: Secret-key agreement over unauthenticated public channels — part ii: The simulatability condition. IEEE Trans. on Information Theory 49(4), 832–838 (2003)
9. Maurer, U., Wolf, S.: Secret-key agreement over unauthenticated public channels — part iii: Privacy amplification. IEEE Trans. on Information Theory 49(4), 839–851 (2003)
10. Renner, R., Wolf, S.: Unconditional authenticity and privacy from an arbitrarily weak secret. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 78–95. Springer, Heidelberg (2003)
11. Boyko, V.M.P., Patel, S.: Provably secure password-authenticated key exchange using Diffie-Hellman. Cryptology ePrint Archive, Report 2000/044 (2000)
12. Rivest, R.L., Shamir, A.: How to expose an eavesdropper. Commununications of ACM 27(4), 393–394 (1984)
13. Gehrmann, C., Mitchell, C.J., Nyberg, K.: Manual authentication for wireless devices. RSA Cryptobytes 7(1), 29–37 (2004)
14. Pasini, S., Vaudenay, S.: An optimal non-interactive message authentication protocol. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 280–294. Springer, Heidelberg (2006)
15. Stajano, F., Anderson, R.: The resurrecting duckling: Security issues for ad-hoc wireless networks. In: Proc. 7th Int. Workshop on Security Protocols, April 1999, pp. 172–194. Springer, Heidelberg (1999)
16. Hoepman, J.H.: The emphemeral pairing problem. In: Proc. 8th Int. Conf. Financial Cryptography, February 2004, pp. 212–226. Springer, Heidelberg (2004)
17. Hoepman, J.H.: Ephemeral pairing on anonymous networks. In: Hutter, D., Ullmann, M. (eds.) SPC 2005. LNCS, vol. 3450, pp. 101–116. Springer, Heidelberg (2005)
18. Vaudenay, S.: Secure communications over insecure channels based on short authenticated strings. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, Springer, Heidelberg (2005)
19. Creese, S., Goldsmith, M., Harrison, R., Roscoe, B., Whittaker, P., Zakiuddin, I.: Exploiting empirical engagement in authenticated protocol design. In: Hutter, D., Ullmann, M. (eds.) SPC 2005. LNCS, vol. 3450, pp. 119–133. Springer, Heidelberg (2005)
20. Čagalj, M., Čapkun, S., Hubaux, J.P.: Key agreement in peer-to-peer wireless networks. IEEE (Special Issue on Cryptography and Security) 94, 467–478 (2006)
21. TinyOS Alliance: TinyOS web page (2006) `http://www.tinyos.net`
22. Mayrhofer, R., Gellersen, H.: Shake well before use: Authentication based on accelerometer data. In: Proc. Pervasive 2007: 5th International Conference on Pervasive Computing, May 2007, Springer, Heidelberg (to appear, 2007)
23. Carter, L., Wegman, M.: Universal classes of hash functions. Journal of Computer and System Science 18, 143–154 (1979)
24. Bennett, C.H., Brassard, G., Crépeau, C., Maurer, U.: Generalized privacy amplification. IEEE Transaction on Information Theory 41(6), 1915–1923 (1995)

9 A Human-Verifiable Authentication Protocol Using Visible Laser Light

Authors: Rene Mayrhofer and Martyn Welch

Published in the proceedings of WAIS 2007, the 1st International Workshop on Advances in Information Security, part of ARES 2007, the 2nd International Conference on Availability, Reliability and Security, pp. 1143–1147
ISBN 0-7695-2775-2
April 2007

A Human-Verifiable Authentication Protocol Using Visible Laser Light

Rene Mayrhofer and Martyn Welch
Computing Department, Lancaster University
Infolab21, Lancaster LA1 4WA, UK
rene@comp.lancs.ac.uk, martyn@warpedlogic.co.uk

Abstract

Securing wireless channels necessitates authenticating communication partners. For spontaneous interaction, authentication must be efficient and intuitive. One approach to create interaction and authentication methods that scale to using hundreds of services throughout the day is to rely on personal, trusted, mobile devices to interact with the environment. Authenticating the resulting device-to-device interactions requires an out-of-band channel that is verifiable by the user. We present a protocol for creating such an out-of-band channel with visible laser light that is secure against man-in-the-middle attacks even when the laser transmission is not confidential. A prototype implementation shows that an appropriate laser channel can be constructed with simple off-the-shelf components.

1. Introduction

Authentication is one of the key issues for secure wireless communication in ubiquitous computing applications. Realising the vision of ubiquitous computing, i.e. of services being integrated into our daily environment, is inherently dependent on intuitive, efficient, and secure methods for spontaneous interaction. When users start interacting with hundreds of services throughout the day, they can neither afford to pay close attention nor invest noticeable effort into these interactions. Securing wireless communication during the interaction must therefore be unobtrusive and implicit; additional steps required "just for security" will most likely be unacceptable. Nonetheless, intuitive interaction demands that the authenticity of communication partners must be easily verifiable by humans.

One approach to solving this issue is to rely on personal, trusted, mobile devices to interact with the environment. These are only used by one user at a time and act as representatives for interactions with other devices, utilising wireless communication in the process. To protect against man-in-the-middle (MITM) attacks on the wireless channel, an out-of-band channel is required for authentication. Various out-of-band channels have already been suggested, most of which provide "physical evidence" for the communication peers in the sense that humans can verify either or both sides of the wireless channel. Examples for such human-verifiable out-of-band channels are relative location measured via ultrasound [9], visual markers photographed with camera phones [10], audio [4], or common motion [8].

In this paper, we present a protocol for creating an out-of-band channel for authentication with visible laser light. In contrast to an earlier protocol suggested by Kindberg and Zhang [6], we do not assume the laser transmission to be confidential. Instead, we assume an attacker to be able to either violate the confidentiality of data transmitted via laser, i.e. to read it, or to violate its authenticity, i.e. to inject own data into the receiver, but not both at the same time. Our contribution is a protocol to establish a secret, authenticated shared key between the personal trusted device and a remote device under these assumptions. The personal device incorporates a laser diode and thus acts as the transmitter on the out-of-band channel, while the remote device is capable of detecting the light from the laser.

In section 2 we briefly analyse related work before discussing our threat model in more detail in section 3. Our protocol is presented in section 4 and analysed from a security point of view in section 5. Finally, section 6 describes an initial prototype implementation that we are currently working on.

2. Related work

Ringwald was among the first to present a working prototype for device-to-device interaction using lasers [12], followed by Patel and Abowd [11]. Both used relatively simple ways of modulating a laser diode and reconstructing the signal at the receiver end, whilst using the laser as an out-of-band method for initiating wireless communication by transmitting the device network address, although without considering security of the interaction.

Kindberg and Zhang previously suggested the transmission of secret keys via modulated laser light [6], under the assumption that the laser emits no light except onto the receiving sensor. However, as explained in section 3, this assumption may not be valid when considering attackers with free line of sight.

Seeing-is-Believing uses 2D barcodes and camera phones as a visual channel [10]. This approach allows users to directly verify what the sensor, i.e. their camera phone, measures. In comparison to a personal device equipped with a laser diode and the service equipped with a sensor, this approach swaps the roles of sender and receiver on the out-of-band channel. The advantage is that it is easier for users to verify the authenticity of the communication peer because authentication in the protocol sense matches what the user verifies. On the other hand, it forces the user to pay closer attention than for simply pointing a laser at a target device.

3. Threat model

Previous work assumed a modulated laser beam to be confidential from attackers [6]. However, this assumption does not seem valid considering two practical experiences:

- Laser diodes do not produce perfectly focused beams of light. This can be observed for example on laser pointers; parts of the light emitted by the laser diode can be seen from almost any angle within its front hemisphere (even if the majority is emitted along the primary axis).
- The laser light is reflected as scattered light from most surfaces, including photovoltaic elements suitable for use as receivers.

That is, the laser light can be seen both at the sender and at the receiver from almost any other point with direct line of sight. With high-speed cameras, it seems possible to capture the modulated signals with reasonable accuracy. We therefore do not assume a modulated laser channel to be confidential.

It is also questionable whether this channel can be assumed to be authentic, because most photovoltaic elements suitable for receivers can not distinguish angle of arrival and thus not between different senders. It is possible for an attacker to point their laser beam on the receiver and therefore inject their own messages into the out-of-band channel. Detecting such message injection will depend on the relative pulse strengths and the sophistication of the receiver. Users may also be unable to spot a “second dot” on the receiver if for example infrared lasers are used by the attacker. However, any such message injection will modify the original messages sent by the user’s personal device.

Therefore, we can only assume that an attacker can not easily block or completely change the information transmitted via a modulated laser beam without previous knowledge of the message contents. We also need to assume the remote device to be secure and trustworthy. Cases where information sent to it by the user is forwarded after successful authentication are out of the scope of this paper.

All wireless communication is generally assumed to be completely open to attack and possibly controlled by a MITM. The aim of our protocol is to prevent MITM attacks on the wireless channel.

In comparison with related protocols for constructing out-of-band channels like MANA I [3], SAS [13], and proposals by Balfanz et al. [2] and Hoepman [5], our assumptions are slightly less constrained. In contrast to MANA I, we do not assume the channel to be confidential. In contrast to the proposal by Balfanz et al., the direction of the channel is reversed. In contrast to Hoepman’s proposal, we do not assume the channel to be confidential and authentic at the same time. Our protocol is most closely related to SAS [13]. However, a further important difference to these protocols is that, for light sensors capable of detecting laser light, we can not assume the user’s laser beam to be the only input. This necessitates some additional precautions in our protocol, and makes it generally difficult to construct completely secure authentication schemes.

4. Protocol

Our proposed authentication protocol combines a wireless channel (**RF**) with a modulated laser (**L**) to create an authenticated secret key, similar to previous work [6]. The difference is that we can not use **L** for transmitting secret keys due to our assumption of **L** not providing confidentiality. Instead, **L** is used to transmit random numbers used only once (*nonces*) as part of a commitment scheme, comparable to e.g. the MANA III protocol [3] and a more recent proposal by Wong and Stajano [14, section 7]. Our protocol is designed so that an attacker would need to violate both the confidentiality and the integrity properties of the laser channel at the same time, i.e. to read what the user’s personal device sends and to inject their own messages into the receiver.

From a user interaction point of view, we combine two steps into one: device selection and implicit authentication. Nonetheless, this combined selection and authentication requires two user actions to prevent accidental selection of a “wrong” device. First the laser needs to be turned on to allow aiming, then the selection and implicit authentication needs to be performed. This can be implemented e.g. with two buttons or with one two-action button.

In the following description, the notation $m|n$ is used to describe string concatenation and $HMAC_K$ refers to an

HMAC [7] with key K. A message M sent over a noisy channel is received as M', to point to possible changes during transmission.

The protocol consists of the following steps between the user's personal device P and the remote device R:

1. The user presses the first button on P to turn on the laser and modulate it with a continuous stream of "ping" messages.

2. When the laser hits the receiver and the "ping" messages are detected, R switches to the "authentication in progress" state and broadcasts a "found" message over $\mathbf{RF}$. In this state, R will only interact with a single personal device (the first to contact it in the next step).

3. By receiving the broadcast, P learns the network address of R. P and R agree to a secret key K via standard Diffie-Hellman key agreement (DH) over $\mathbf{RF}$ and R turns on its first LED (e.g. yellow).

4. When satisfied with the selection of R, the user presses the second button and the devices loop through the following steps until authentication is successful or the user stops the process by releasing the button:

   (a) P generates a fresh nonce N.

   (b) P computes $M_1 := HMAC_K(N|1)$ and sends it to R over $\mathbf{RF}$.

   (c) R acknowledges the receipt by sending $M_2 := HMAC_K(M_1)$ to P over $\mathbf{RF}$.

   (d) P verifies M_2 and transmits $M_3 := N$ over $\mathbf{L}$ by modulating the laser.

   (e) R receives N', computes $HMAC_K(N'|1)$ and verifies that it matches M_1. It then sends $M_4 := HMAC_K(N|2)$ over $\mathbf{RF}$ and turns on its second LED (e.g. green).

   (f) P verifies M_4 and notifies the user of successful verification, e.g. by turning on an LED (green).

The loop is necessary due to the possibility of transmission errors over $\mathbf{L}$; it is important not re-use nonces but to generate fresh nonces in each iteration. Only when both R and P signal success (e.g. with green LEDs) should the user continue with the interaction.

Note that the authentication part of the protocol does not rely on asymmetric primitives and is thus suitable for implementation on resource limited devices such as sensor nodes. However, when not assuming the laser channel to be confidential, asymmetric cryptography like DH or its Elliptic curve variant (ECDH) is necessary for creating a secret shared key (see step 2 in the protocol).

5. Analysis

Our protocol uses both the (weak) confidentiality and integrity properties of the modulated laser channel:

- Integrity of $\mathbf{L}$ is exploited in steps 4b) to 4e): a MITM can only pass the check in 4e) when it can inject its own nonce $\widetilde{N}$ so that the $HMAC_K(\widetilde{N}|1)$ matches. Without such an injection on $\mathbf{L}$, there are only two options: When the MITM simply relays M_1, the HMAC will not match because of the different shared key. On the other hand, the MITM can not generate a valid HMAC message because N has not yet been transmitted and is therefore unknown. Step 4b) thus serves to commit the sender P to the content that will be sent over $\mathbf{L}$ and to bind this commitment to the shared key K.

- Confidentiality of $\mathbf{L}$ is exploited in steps 4d) to 4f): a MITM can only pass the checks in 4e) and 4f) when they can eavesdrop on the laser, because only then will N be revealed.

Each of the steps is necessary under our assumptions:

- M_1 needs to be sent in 4b) and acknowledged in 4c) before transmitting N over $\mathbf{L}$ in 4d), otherwise the attacker could just postpone sending the message from 4b) until N has been sent in plain text (i.e. assuming authenticity but not confidentiality of $\mathbf{L}$) .

- M_4, generated in 4e) and verified in 4f), is necessary otherwise the attacker could inject their own nonce $\widetilde{N}$ in steps 4b) to 4d) and pass the check in 4e) (i.e. assuming confidentiality but not authenticity of $\mathbf{L}$).

- The LEDs on P and R are necessary so that the user can check synchronicity. Without these the attacker could just generate message M_4 in step 4e) without verifying that $HMAC_K(N'|1)$ matches M_1 (i.e. assuming authenticity but not confidentiality of $\mathbf{L}$).

Due to using long (i.e. $\geq$ 128 bits) nonces, this protocol is not susceptible to attacks against short codes on the out-of-band channel [14, section 3]. Only when an attacker can perfectly overhear the original nonce N (sent by P over $\mathbf{L}$) and inject an own nonce $\widetilde{N}$ over $\mathbf{L}$ (as received by R) will a MITM attack on $\mathbf{RF}$ go undetected. As outlined in section 3, a laser channel is neither strictly confidential nor authentic. An attacker close to the target device R can observe the "red dot" at the sender and can shine a (possibly stronger and/or invisible IR) laser beam on the receiver, thus violating both the channel's confidentiality and authenticity. It remains to be shown how practical such attacks on both the confidentiality and the integrity are, taking the mobility of P and short interaction times into account.

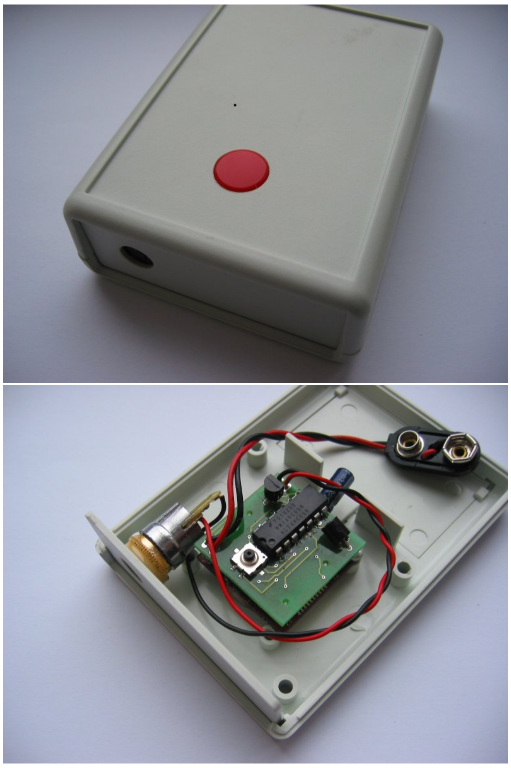


Figure 1. Prototype implementation of a personal trusted device with an Intel Mote and off-the-shelf components

Denial-of-service attacks on the laser receiver can not be avoided, but would be even easier to perform on the wireless channel.

6. Prototype implementation

Figure 1 shows our prototype personal trusted device with a laser diode. It is based on an Intel Mote ISN100-BA (with an ARM7 core at 12 MHz and integrated Bluetooth radio); a laser diode stripped from a £ 1 laser pointer; a two-action button and a few additional off-the-shelf components (NAND gates, transistor, etc.). The Intel Mote runs TinyOS [1] and is used to implement the P side of our protocol using Bluetooth as the $\mathbf{RF}$ channel and the UART for modulating the laser channel $\mathbf{L}$. Our first receiver prototype, shown in Fig. 2, uses a photo resistor (covered with the lens from a PIR) and a simple high-pass and thresholding circuit for reconstructing the signal. The signal is fed (via a RS232 level converter) into a serial port of a PC which also has a Bluetooth dongle. Not considering the Intel Mote, the overall cost of building both the prototype sender and the receiver was below £ 10.

User interaction is designed to be as simple as possible.

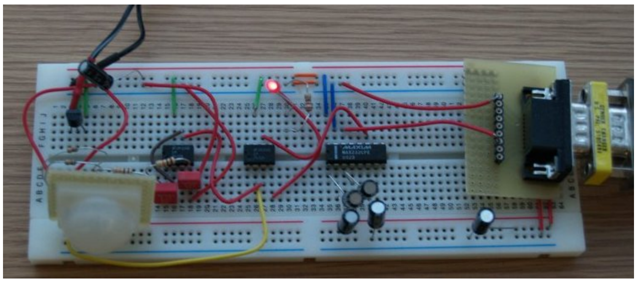


Figure 2. Prototype implementation of the receiver part.

We use a two-action button, similar to the buttons commonly used in digital cameras, to implement the two levels of action. By pressing the single button half-way, the laser lights up and allows proper aiming. By depressing the button fully, the target is selected and authenticated. Patel and Abowd also suggested to use a two-action button, but did not report a practical implementation of such an interaction [11].

Our prototype is still under development, and first results suggest improvements to the receiver are required. In practise it seems difficult to focus the laser beam on a target area that is about 2 cm in diameter, we thus intend to experiment with solar cells as receivers, as suggested e.g. by Ringwald [12]. The prototype is currently a proof of concept for modulated laser transmission and simple user interaction, but does not currently implement our complete protocol under TinyOS. We have not yet considered higher-level error correction methods for recovering from transmission errors on the laser channel $\mathbf{L}$. Instead, our protocol loops at sending nonces until successful authentication. This is not only robust against actual transmission errors, but also against "wiggle" when aiming the laser.

Another practical issue we have not yet considered is network discovery. In our protocol step 2, we assume some method for R to announce on the $\mathbf{RF}$ channel to P that it has been found by the laser beam. Bluetooth is a promising protocol for supporting wide interoperability, but does not provide broadcasts, and inquiry times are also particularly slow. One possibility to overcome this issue is to opportunistically run a DH key agreement (protocol step 3) with every Bluetooth device in range and to send the "found" message via multiple unicast packets to all previously discovered devices.

7. Conclusions

We have presented a protocol for creating a shared secret key over a wireless channel and authenticating it with a modulated laser channel. Under the assumption that an attacker can not both eavesdrop on the laser transmission and inject their own laser messages at the same time, our protocol is secure against man-in-the-middle attacks, eavesdropping, and message alterations.

Our prototype implementation is work in progress, but first results confirm that it is possible to transmit short modulated messages with laser diodes and simple off-the-shelf components. This low-cost solution makes a laser channel a viable option for wide-spread implementation in consumer devices such as mobile phones. We suggest that a laser channel can be used as an intuitive and secure out-of-band channel for spontaneous device pairing.

8. Acknowledgements

We gratefully acknowledge support by the Commission of the European Union under FP6 Marie Curie Intra-European Fellowship contract MEIF-CT-2006-042194 "CAPER" and thank Matt Oppenheim for support in building the prototype sender.

References

[1] TinyOS web page. `http://www.tinyos.net`, 2006.

[2] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proc. NDSS'02: 2002 Network and Distributed Systems Security Symposium*. The Internet Society, February 2002.

[3] C. Gehrmann, C. J. Mitchell, and K. Nyberg. Manual authentication for wireless devices. *RSA Cryptobytes*, 7(1):29–37, 2004.

[4] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: Human verifiable authentication based on audio. In *Proc. ICDCS 2006: 26th Conf. on Distributed Computing Systems*, page 10. IEEE, July 2006.

[5] J.-H. Hoepman. The emphemeral pairing problem. In *Proc. 8th Int. Conf. Financial Cryptography*, pages 212–226. Springer, February 2004.

[6] T. Kindberg and K. Zhang. Secure spontaneous devices association. In *Proc. UbiComp 2003: 5th Int. Conf.*, pages 124–131. Springer, October 2003.

[7] H. Krawczyk, M. Bellare, and R. Canetti. RFC2104: HMAC: Keyed-hashing for message authentication, February 1997.

[8] R. Mayrhofer and H. Gellersen. Shake well before use: Authentication based on accelerometer data. In *Proc. Pervasive 2007: 5th International Conference on Pervasive Computing*. Springer, May 2007. *to appear*.

[9] R. Mayrhofer, H. Gellersen, and M. Hazas. An authentication protocol using ultrasonic ranging. Technical Report COMP-002-2006, Lancaster University, October 2006.

[10] J. M. McCune, A. Perrig, and M. K. Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *Proc. IEEE Symposium on Security and Privacy*, pages 110–124. IEEE, May 2005.

[11] S. N. Patel and G. D. Abowd. A 2-way laser-assisted selection scheme for handhelds in a physical environment. In *Proc. UbiComp 2003: 5th Int. Conf.*, pages 200–207. Springer, October 2003.

[12] M. Ringwald. Spontaneous interaction with everyday devices using a PDA. In *Proc. Workshop on Supporting Spontaneous Interaction in Ubiquitous Computing Settings*, September 2002.

[13] S. Vaudenay. Secure communications over insecure channels based on short authenticated strings. In *Advances in Cryptology - Proc. CRYPTO 2005: 25th Int. Cryptology Conf.* Springer, August 2005.

[14] F.-L. Wong and F. Stajano. Multi-channel protocols. In *Proc. Security Protocols Workshop 2005*. Springer, 2006.

10 Using a Spatial Context Authentication Proxy for Establishing Secure Wireless Connections

Authors: Rene Mayrhofer and Roswitha Gostner

Published in Journal of Mobile Multimedia, 3(3):198–217, Rinton Press
ISSN 1550-4646
March 2007

Journal of Mobile Multimedia, Vol. 3, No. 3 (2007) 198–217
© Rinton Press

USING A SPATIAL CONTEXT AUTHENTICATION PROXY FOR ESTABLISHING SECURE WIRELESS CONNECTIONS[a]

RENE MAYRHOFER   ROSWITHA GOSTNER
Computing Department, Lancaster University, South Drive
Lancaster LA1 4WA, United Kingdom
rene@comp.lancs.ac.uk   gostner@comp.lancs.ac.uk

Received March 13, 2007
Revised May 6, 2007

Spontaneous interaction in wireless ad-hoc networks is often desirable not only between users or devices in direct contact, but also with devices that are accessible only via a wireless network. Secure communication with such devices is difficult because of the required authentication, which is often either password- or certificate-based. An intuitive alternative is context-based authentication, where device authenticity is verified by shared context, and often by direct physical evidence. Devices that are physically separated cannot experience the same context and thus cannot benefit directly from context authentication. We introduce a context authentication proxy that is pre-authenticated with one of the devices and can authenticate with the other by shared context. This concept is applicable to a wide range of application scenarios, context sensing technologies, and trust models. We show its practicality in an implementation for setting up IPSec connections based on spatial reference. Our specific scenario is ad-hoc access of mobile devices to secure 802.11 WLANs using a mobile device as authentication proxy. A user study shows that our method and implementation are intuitive to use and compare favourably to a standard, password-based approach.

Keywords: spontaneous interaction, wireless and mobile security, authentication, location awareness

1 Introduction

Spontaneous interaction is a desirable feature for many ubiquitous computing scenarios. It is typically seen as a process between users or devices that are in direct contact with each other, and often implies spatial proximity. However, spontaneous interaction can also be important between users or devices that are physically or virtually separated, but can communicate over some common channel like a wireless network. A similar situation arises when interacting with devices that do not feature any user interface, but only communicate wirelessly. One prominent example is IEEE 802.11 WLAN itself: users, represented by their client devices, engage in spontaneous interaction with access points that usually neither have a user interface nor are physically accessible (they might be built into building infrastructure).

The problem with such settings is to authenticate users or devices. Wireless networks are particularly vulnerable to attacks, ranging from simple eavesdropping to more sophisti-

[a] This is an extended version of "Rene Mayrhofer: *A Context Authentication Proxy for IPSec using Spatial Reference*, Proc. TwUC 2006, Austrian Computer Society (OCG), 449–462, December 2006".

cated man-in-the-middle (MITM) attacks. Although there are well-known protocols to secure communication over wireless networks, they all depend on some form of authentication. Only after authenticating the communication partner, further steps to create a secure channel make sense. More specifically, the problem is to authenticate intuitively and efficiently.

From a user point of view, secure channel setup should be as transparent as possible and should cause minimal, if any, overhead to the desired spontaneous interaction. Any additional burden that is caused by authentication is not part of the intended interaction, and thus collides with spontaneity. Obvious and often deployed solutions for authentication are typically either secure or convenient. Password-based authentication like Bluetooth-style PINs, WEP, and WPA-PSK is one example, which is unfortunately neither particularly secure nor user friendly; another well-known solution is certificate-based authentication like X.509 public key infrastructures (PKIs).

An example of a secure channel implementation is IPSec. It is currently considered one of the most secure communication protocols, supports both password- and certificate-based authentication, and has been designed for cross-platform interoperability, but is daunting to set up even for technically skilled users. Although it has desirable properties from a security point of view, many users may choose not to use it for spontaneous and ad-hoc interactions. Giving credit to its wide-spread use and practical problems, also investigated by others [1, 3], we therefore use the setup of IPSec connections as our motivating example. More specifically, our demonstration application is to grant secure access to a WLAN access point – and consequently the network it manages – to new clients such as laptops via IPSec connections.

Context based authentication, or *context authentication*, allows secure and intuitive authentication without introducing unreasonable overhead that would be incompatible with spontaneous interaction. It uses shared context between devices to create shared secrets. These shared secrets can consequently be used as cryptographic tokens for creating secure channels. However, devices such as WLAN access points that are physically separated from user devices or that have no sensors or user interfaces are unable to experience the same context.

Our approach to allow such devices to authenticate via shared context is to introduce a *context authentication proxy*. The proxy is pre-authenticated to the device that does not have sensors or a user interface itself, and authenticates to other devices on behalf of it. This concept is independent of the underlying infrastructure for expressing trust, and can work in online and offline settings and with existing password- or certificate-based authentication mechanisms. Our example application uses a mobile context authentication proxy in the form of a personal digital assistant (PDA) for better ease of use.

The contribution of this work is twofold: we examine the general concept of a context authentication proxy in more detail, discuss different options of implementing it, and we show a specific application for a widely used protocol. A user study shows that authentication based on relative location — one aspect of shared context — is a viable alternative to standard, password-based authentication. Our implementation also confirms a user study presented in related work [1], anecdotally showing a significant improvement of ease of use in setting up IPSec connections due to use of context authentication. We also argue that, although demonstrated by an application for securing wireless networks, context authentication proxies are of wider applicability.

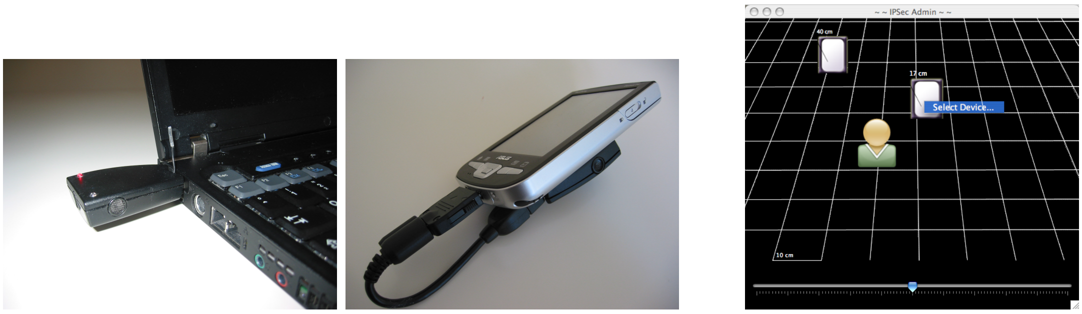
In the following, we first discuss related work in Section 2 and the previously introduced concept of context authentication in Section 3. Our main contribution is the general notion of an authentication proxy presented in Section 4 and our specific implementation for authenticating IPSec connections shown in Section 5 which we investigate in Section 6. Finally, we discuss further alternatives for implementing context authentication and for using the established shared secrets in secure communication protocols in Section 7.

2 Related work

Our chosen example of securing IEEE 802.11 WLAN using context authentication has also been discussed by Balfanz et al. [1]. They present a system called "Network-in-a-box" (NiaB) that uses an infrared channel to transmit authentic cryptographic tokens and automates the set-up of secure wireless communication in much the same way as our example application does. This infrared connection is established between the client device and either the WLAN access point itself, or, in case of a distributed infrastructure, an "enrollment station", which can be regarded as a stationary instance of a context authentication proxy. Furthermore, they show in a user study that context authentication can, for end-users, significantly lower the time required to set up a secure wireless network. The major difference to our work is the role of the authentication proxy. In NiaB, the authentication proxy is described as a permanent station that authenticates all devices that are able to establish infrared connections to it. On the other hand, we specifically assign the authentication proxy an active role, in which it triggers the authentication process to a selected client, as described in more detail in Section 4. A specific advantage of our approach is that the authentication proxy can be mobile — and for our demonstration application, it explicitly is. Instead of forcing users to bring their devices to fixed stations, administrators can authenticate devices wherever it is necessary and appropriate. This can include authentication of new fixed stations, which is not possible with the less flexible enrollment station described by NiaB.

Kindberg et.al. [8] describe "channel proxies", which may be seen as a low-level implementation of a context authentication proxy. These channel proxies selectively forward messages depending on some constraints, like location of the sender or the receiver. In contrast, our concept of context authentication proxies explicitly includes high-level processing of messages. In our example, this allows the complete authentication protocol to be performed between the proxy and the WLAN client, while the WLAN access point will typically be unaware of the whole process.

Godber and Dasgupta [3] describe another implementation that is closely related to the demonstrative application we discuss in Section 5. Their system called "Secure Wireless Gateway" (SWG) uses IPSec to secure IEEE 802.11b WLAN, and also provides a captive portal to redirect unauthenticated users to a web page with instructions on how to authenticate. They suggest to use a common shared key for guest users, which is to be considered insecure against MITM attacks, and individual shared keys for registered users. However, they explicitly do not investigate generation and distribution of these individual shared keys or the use of certificate-based authentication and define it as out of scope of their work. In the present article, we focus on this key distribution problem and present an implementation similar to SWG as an example application making use of easy key distribution.



(a) USB dongles for sensing relative spatial positions can be added to typical off-the-shelf laptops, PDAs, or even mobile phones

(b) User interface for spatial selection

Fig. 1 Current implementation of context authentication by spatial reference

3 Context authentication

Context authentication tries to provide intuitive means of authenticating users or devices by verifying that they are in some specific context, e.g. at some specific location. The possibilities for sharing context are obviously constrained by the sensors available to the involved devices. These sensors are used to verify some properties of the device to authenticate, i.e. to verify that the other device is in the same context. Context authentication then aims to create shared secrets for setting up secure communication, usually in the form of cryptographic key material.

In earlier work we reported on using *spatial reference* for authenticating spontaneous interaction [13] and on the security properties of our underlying localization method [12]. Selecting devices based on direct line of sight has also been explored with the "gesturePen" [18]. The gesturePen has the intention of selecting devices by pointing at them, in much the same way as we select devices based on their relative spatial position in this work. Location is just one option for context authentication, and for the description of additional options, we refer to others [8, 2]. In the present article, we investigate connections that are initiated in an ad-hoc manner but that might yield longer-lived security associations. Specifically, we establish IPSec connections on first contact, but continue to use these connections once established.

Building upon our current implementation [13], we assume devices to be equipped with sensors in the form of USB dongles. These dongles provide accurate sensing of relative spatial positions using ultrasound. Figure 1a shows two of them attached to a laptop and attached to a PDA — both can sense each others position with an accuracy of better than 10 centimeters in distance and 25° in angle [6].

In an office space with many laptops, PDAs, and other devices communicating over the same wireless network within a small area, this fine-grained sensing of shared context offers distinct advantages in selecting specific devices. If an administrator wants to allow "that device over there" to access the wireless network,[b] then other devices in the same room should not automatically be allowed too. Solutions based on infrared connections can not easily provide such a fine-grained selection because infrared beams often span the whole area.

[b] The same method can be used to allow access to private parts of the network, or, more generally speaking, to specific resources. We use access to the wireless network only as an example.

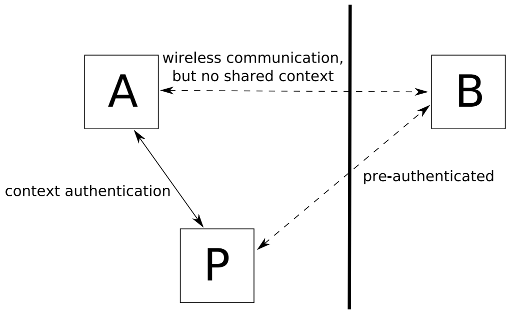


Fig. 2 Using a context authentication proxy P allows physically separated devices A and B to benefit from context authentication even when they can not experience the same context

Our context authentication protocol integrates secure authentication transparently and seamlessly with device selection, as shown in Figure 1b. Simply by selecting a visualized device position, corresponding to the physical device as visible to the user, the authentication process is triggered. User interaction is thus changed from selecting devices from a network-discovered list to a spatially-discovered environment; authenticating selected devices happens automatically without further user interaction. This seamless integration makes the protocol well suited for spontaneous interaction. Security properties of our authentication protocol [13] and ultrasound as an out-of-band channel [12] have been discussed previously. Here we simply assume the protocol to provide a shared secret to both devices that perform the spatial authentication.

Although we build upon this specific authentication protocol for our demonstrative application, the concept of authentication proxies is independent of the underlying sensing platform for context authentication.

4 Authentication proxy

Previous work on context authentication assumes that those devices that authenticate each other can experience the same context, but this is not always possible. Figure 2 shows a device A, e.g. owned by Alice, trying to interact securely with a device B, e.g. a WLAN access point. Because the access point is physically inaccessible, Alice can not benefit from direct context authentication with it to secure her communication. By introducing a *context authentication proxy* P, we give her this option. The authentication proxy experiences the same context as one of the devices, i.e. it shares some aspect of the context. With the other device, it is pre-authenticated. It will usually be desirable that context be shared with the more volatile side, i.e. with mobile devices, changing environments, or, generally speaking, with transient connections. Since we assume a more permanent relationship with the other end of the authentication, in this example between P and the access point, the necessary pre-authentication only needs to occur once during set-up of these devices. Any standard authentication protocol, e.g. password- or certificate-based ones or any means of conveying trust of B in P can be used. Due to this trust relationship, the possibly mobile authentication proxy P is assumed to be used or maintained by a trusted person, such as a system administrator.

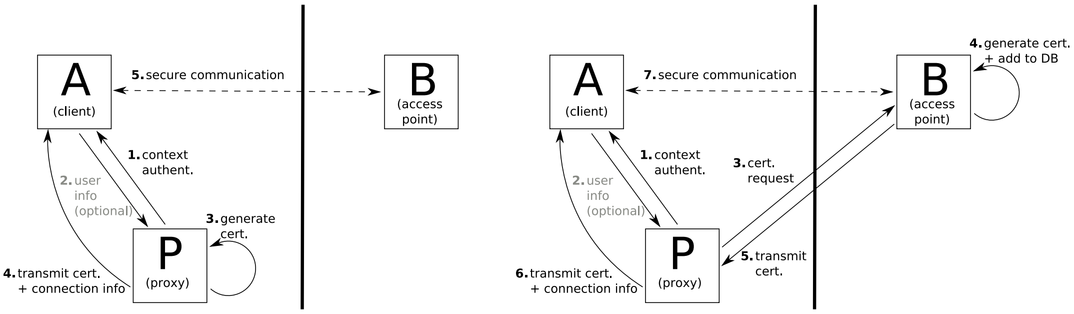
The main task of the authentication proxy is to create a shared secret between A and B, to enable secure communication between them over a wireless network. Depending on the initiator of the authentication, we can distinguish between two different approaches for user interaction with the proxy:

- We speak of a *passive authentication proxy* when P acts as an authentication service and simply waits for clients to initiate an interaction. The *client* takes the active role, starts context authentication with P to obtain a shared secret for communicating securely with B, and may need to engage in another authentication procedure with B over the now-authenticated wireless network. Instances of this approach are the closely related NiaB [1], and one of our previous works [14], which describes the use of RFID tags to secure communication over wireless ad-hoc peer-to-peer networks. The former requires a further offline authentication step performed by the in-house certificate authority when used for "enterprise" WLAN access, or relies on the infrared channel authentication for the simpler "home" WLAN setting. In the latter, we store public keys of network peers on associated RFID tags that can be read for secure spontaneous interaction. Note that in this previous work, we termed the RFID tags "objects" and the associated devices with which the interaction takes place the "proxies" because of a slightly different focus on the interaction.

- For an *active authentication proxy*, the roles of waiting for and of initiating the context authentication are swapped between A and P. That is, the *proxy* takes the active role, starts context authentication with A to generate a shared secret for letting A communicate securely with B, and may take additional steps to register A with authorization databases. In this case, A only waits to be authenticated and does not need to take any additional steps. This requires even less user interaction by offloading some steps to the proxy and can thus further decrease the burden placed on the user for setting up secure communication. We point out that the interaction between A and P, and subsequently between A and B, is still spontaneous. However, the change in roles relieves the client from going through additional steps after the initial context authentication and shifts this task to the proxy. P is in a better position to perform them, because it is part of the existing network and is thus assumed to know more about it than the new client.

Choosing between a passive and an active authentication proxy also depends on the respective trust model. If the trust model can express transitive trust, i.e. delegating trust from one entity to another, then B can delegate authorization decisions to P. Without the ability to delegate trust, an active authentication proxy can still initiate the context authentication, but a subsequent authorization step might be necessary before A can access resources on B. In this case, the choice of authentication proxy should match the interaction style of the application, i.e. who initiates the spontaneous interaction. Note that arbitrary trust models can be used, including the sharing of passwords — which is clearly not recommended from a security point of view — and that most can be used to delegate trust in some way. A concept for delegating restricted trust over potentially multiple hops is described e.g. by Steffen and Knorr [17] and could be used in combination with context authentication proxies.

One secure and standardized option to delegate trust is to use X.509 certificates signed by a certificate authority (CA) managed by P and trusted by B. Every certificate that P



(a) The proxy implements the CA — no connection between the proxy and the access point is necessary

(b) The access point (or infrastructure) implements the CA — the proxy needs an online connection to request certificates for and forward them to the client

Fig. 3 Two options for delegating trust with a context authentication proxy

creates and signs will be trusted by B, allowing P to make decisions about authorizing clients to use B's services. In this sense, our approach of a context authentication proxy is an implementation of the plug-and-play PKI [5]: a client device is automatically provided with an X.509 certificate that allows it to use some services. But instead of initially authenticating with the suggested username/password combination, we authenticate client devices based on context, specifically based on their relative spatial position. This makes the approach more usable for spontaneous interaction.

5 Application for establishing IPSec connections

In this section we present *IPSecME* (IPSec made easy), an application to delegate trust for authorizing IPSec connections that uses an active authentication proxy and standard X.509 certificates. It uses our secure spatial authentication protocol described in earlier work [13] and does not depend on software being pre-installed on the client like NiaB.

5.1 Concept

Our IPSecME application can be used for setting up arbitrary IPSec connections by providing appropriate connection details in the form of an XML configuration file to the authentication proxy. IPSec tunnels over an otherwise open 802.11 WLAN are a practical example without loss of generality. For simplifying the discussion, we also assume the access point to act as an IPSec gateway, but it could be easily split into different devices without any change to our work.

IPSecME consists of two parts, one running on the client and one on the proxy device. Figure 3 shows two options for implementing this application using an active context authentication proxy P: the CA can either run directly on P, or it can run on the access point B (or any other infrastructure device). In the former case, B delegates trust about authorization to P by allowing all clients A that present a certificate signed by the proxy's CA to establish IPSec tunnels. As illustrated in Fig. 3a:

1. P authenticates A via shared context, in this application via spatial reference.
2. A can optionally send information about the logged in user, the machine name, etc., if this should be encoded in the X.509 certificate.
3. P generates a new X.509 certificate with the information provided by A and/or locally entered data and signs it with its CA key. Note that the certificate is bundled with the matching private key.
4. P forwards the new certificate, the private key, its CA certificate, and details about the IPSec connection, i.e. the IP address of the gateway, the remote subnet, etc. to A. The private key is encrypted with the shared key generated in step 1.
5. A uses its new certificate and the IPSec connection description to establish a secure connection to B.

This option has the advantage that no online connection between the P and B is required. The trust between them is formed by B importing P's CA certificate. After this, no further communication between B and P is necessary for authenticating arbitrary clients.[c]

In the latter case, P requests certificates from the CA running on B using an online connection. As illustrated in Fig. 3b:

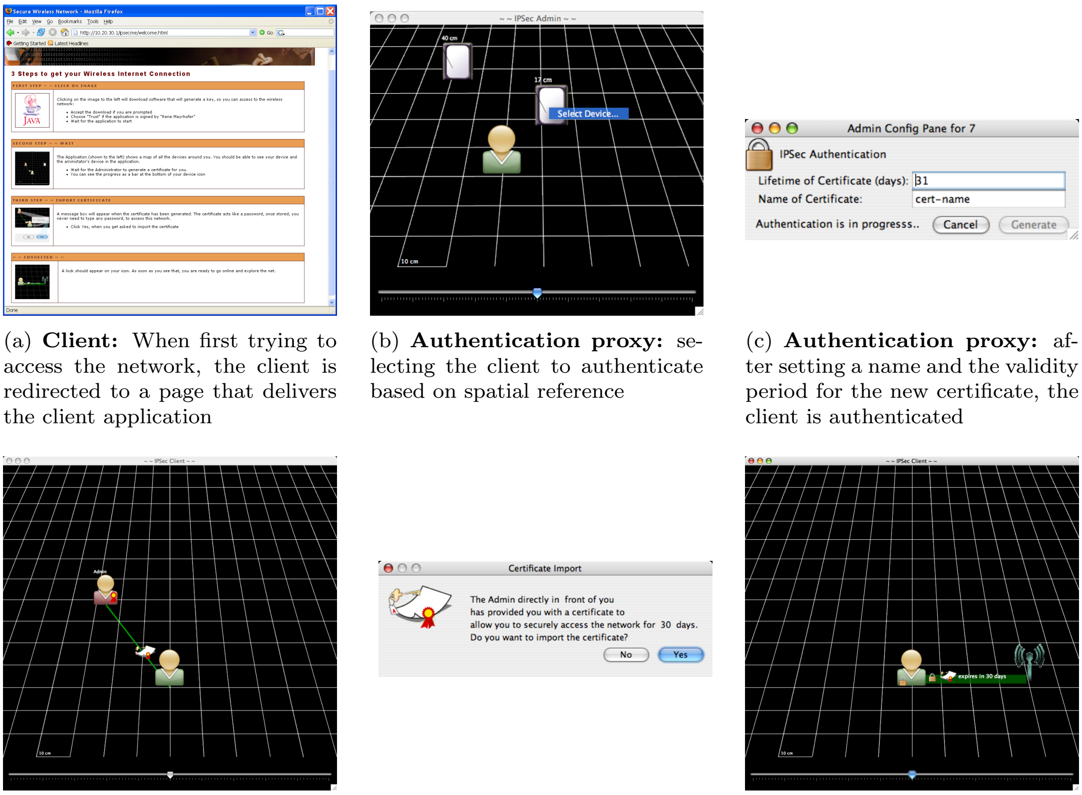
1. equal to step 1 in the former case
2. equal to step 2 in the former case
3. P generates a certificate request with the information provided by A and/or locally entered data and sends it to B.
4. B decides if A should be authorized and, if yes, signs the certificate request with its CA key and adds the new certificate to its authorization database.
5. B sends the new certificate to P.
6. equal to step 4 in the former case
7. equal to step 5 in the former case

The necessarily secure connection between B and P forms the pre-authentication between them with a slightly different trust model. B trusts P to authenticate A based on shared context and to forward machine information and certificates, but keeps decisions about authorization local. For spontaneous interaction, the first option has the advantage that no online connection between B and P is necessary, and we therefore implement this one.

5.2 Implementation

The implementation currently runs on a standard Laptop running Windows XP SP2 or Linux with any of the available IPSec implementations as the client A and a PDA running Pocket PC 2003 (or a laptop running any supported operating system) as the authentication proxy P. Because our context authentication protocol using spatial reference and the IPSecME application have been implemented in Java, other platforms can be supported fairly easily. All platform-specific parts, i.e. managing certificates, establishing IPSec connections, and access to the ultrasound sensing devices, have been implemented for Windows XP, Linux, and Mac OS/X. The combination of access point and IPSec gateway, depicted as B in the above concept, has been implemented in two different versions. A standard access point connected to a PC running Gibraltar firewall [11] represents an enterprise scenario where the

[c] Note that revoking a certificate that P generated will require an update of its associated certificate revocation list (CRL) on B, and consequently communication between B and P. However, for spontaneous interactions, short-lived certificates can be used to alleviate the need for CRL updates.



Secure Wireless Network - Mozilla Firefox

http://10.20.30.1/ipsecme/welcome.html

3 Steps to get your Wireless Internet Connection

FIRST STEP ~ ~ CLICK ON IMAGE

Clicking on the image to the left will download software that will generate a key, so you can access to the wireless network:

- Accept the download if you are prompted
- Choose "Trust" if the application is signed by "Rene Mayrhofer"
- Wait for the application to start

SECOND STEP ~ ~ WAIT

The Application (shown to the left) shows a map of all the devices around you. You should be able to see your device and the aministator's device in the application.

- Wait for the Administrator to generate a certificate for you.
- You can see the progress as a bar at the bottom of your device icon

THIRD STEP ~ ~ IMPORT CERTIFICATE

A message box will appear when the certificate has been generated. The certificate acts like a password, once stored, you never need to type any password, to access this network.

- Click Yes, when you get asked to import the certificate

~ ~ CONNECTED ~ ~

A lock should appear on your icon. As soon as you see that, you are ready to go online and explore the net.

Done

Admin Config Pane for 7

IPSec Authentication

Lifetime of Certificate (days): 31

Name of Certificate: cert-name

Authentication is in progresss..

Certificate Import

The Admin directly in front of you
has provided you with a certificate to
allow you to securely access the network for 30 days.
Do you want to import the certificate?

(a) **Client:** When first trying to access the network, the client is redirected to a page that delivers the client application

(b) **Authentication proxy:** selecting the client to authenticate based on spatial reference

(c) **Authentication proxy:** after setting a name and the validity period for the new certificate, the client is authenticated

(d) **Client:** authentication in progress

(e) **Client:** accepting the new certificate and the CA of the IPSec gateway to establish the connection

(f) **Client:** IPSec connection successfully established

Fig. 4 Screen shots of the IPSecME application

functionality of B is distributed in the infrastructure. An embedded implementation using the OpenWrt distribution [16] on an Asus WL-500GP access point represents the home/small office scenario with a single, combined device. Both implementations use Openswan [19] as IPSec implementation and ChilliSpot [7] to provide the captive portal. These two scenarios show that our approach can be used with arbitrary implementations of WLANs and IPSec gateways as long as they support external X.509 CAs.

Figure 4 shows how users experience the whole process. The client does not need to have any special software pre-installed and does not need any a priori information about the environment. When it first connects to the WLAN, which is publicly accessible, its web browser gets redirected to a local web page in the same way as it is used by the currently popular WLAN hot spots (see Fig. 4a). From this web page, the user can start the client part of the application via Java Webstart and then simply waits for the proxy to initiate authentication. We assume that devices are either equipped with ultrasound sensing or that the USB dongles are attached at this stage. For ease of use, we skip the optional step 2 and omit to use client-provided information for generating the certificate. With spontaneous interaction, any such information tends to be meaningless anyway due to the lack of a globally accepted naming scheme. An administrator using the context authentication proxy can then select the client based on spatial reference (see Fig. 4b) and specify the validity period of the certificate and optionally a name describing the client for later use (see Fig. 4c). This name only needs to be meaningful within this environment, e.g. to the administrator. After initiating the context authentication protocol, the certificate is generated and signed automatically, and the IPSec connection details along with the certificate are sent to the client (see Fig. 4d). Note that the private key contained in the PKCS#12 format used for transmitting the certificate is encrypted with the shared secret that has been established between the client and the proxy during context authentication. Thus, they can communicate over the public, insecure WLAN without worrying about attacks. Finally, after receiving the certificate and connection details, the client can, when accepting them, immediately import its new certificate and establish the IPSec connection (see Fig. 4e and 4f). Further communication is automatically secured by the IPSec tunnel, which in our case includes all traffic to and from the client.

6 Experimental Evaluation

Although our spatial authentication method in general and our authentication proxy application for establishing IPSec connections in particular have been designed to make user interaction as intuitive as possible, they are new and to this time unknown to potential users. In contrast, users have already been trained to use existing, typically password-based methods to get access to WLANs. We therefore conducted two user studies to evaluate how end-users react to our method and to discover potential issues, and one informal study from the administrator point of view.

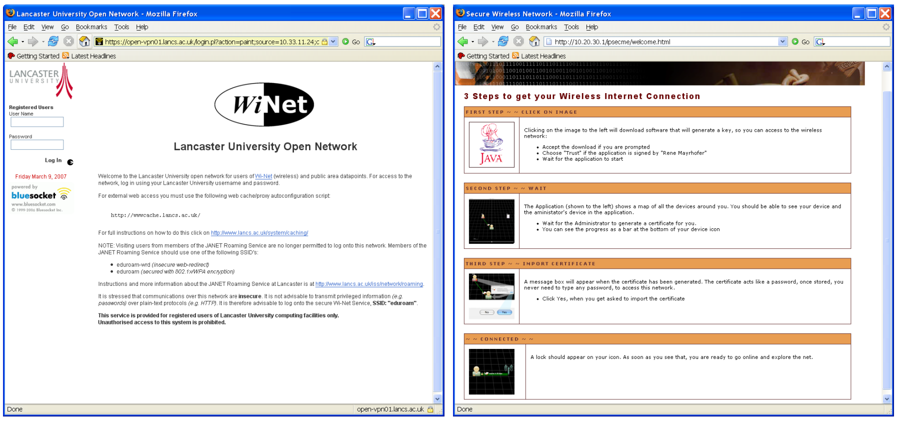
All subjects were office workers, either researchers from various fields or administrative staff in an academic environment. They generally had extensive experience with typical desktop applications and Internet usage, some also from a more technical point of view.

6.1 Study 1: Comparison to WLAN with captive portal and password authentication

Experimental design Our first study directly compared the end-user experience and satisfaction between a currently deployed solution and our IPSecME research prototype. Subjects were asked to get access to the respective WLAN with a standard laptop running Windows XP. In our study, neither of the variants assumed any a priori information to be shared between the subjects, who acted as guests in a new environment, and the WLAN environment itself. For simplifying the study, two laptops were used, one set to the WLAN ESSID of the first, the other to the ESSID of the second network. When opening the web browser (Mozilla Firefox) in the respective unauthenticated states, both displayed a captive web page with instructions on how to gain access to the network.

The aim of this study was to compare usability and end-user experience, and therefore subjects were not explicitly educated about the underlying principles and differences between the methods. Specifically, they were not told that the existing method only authorizes their laptop to access the network, while our IPSecME method additionally provides a secure IPSec channel for all IP connections.

Figure 5a shows the WLAN captive portal web page used at Lancaster University. Users can enter their normal network account details in the form of username and password to gain access to the network. Guests new to this environment would not have such an account, and therefore our subjects were asked to use one specific account and given the username and password (12 randomized characters, mixed upper and lower case letters and digits).


Lancaster University Open Network - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
https://open-vpn01.lancs.ac.uk/login.pl?action=paint;source=10.33.11.24;c
Go
Getting Started Latest Headlines
LANCASTER UNIVERSITY
Log In
powered by bluesocket www.bluesocket.com © 1999-2006 Bluesocket Inc.
WiNet
Done
open-vpn01.lancs.ac.uk
Secure Wireless Network - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://10.20.30.1/ipsecme/welcome.html
Go
Getting Started Latest Headlines
JAVA
No
Yes
Done

Registered Users
User Name
Password

Friday March 9, 2007

Lancaster University Open Network

Welcome to the Lancaster University open network for users of Wi-Net (wireless) and public area datapoints. For access to the network, log in using your Lancaster University username and password.

For external web access you must use the following web cache/proxy autoconfiguration script:

```
http://wwwcache.lancs.ac.uk/
```

For full instructions on how to do this click on http://www.lancs.ac.uk/system/caching/

NOTE: Visiting users from members of the JANET Roaming Service are no longer permitted to log onto this network. Members of the JANET Roaming Service should use one of the following SSID's:

- eduroam-wrd *(insecure web-redirect)*
- eduroam *(secured with 802.1x/WPA encryption)*

Instructions and more information about the JANET Roaming Service at Lancaster is at http://www.lancs.ac.uk/iss/network/roaming.

It is stressed that communications over this network are **insecure**. It is not advisable to transmit privileged information *(e.g. passwords)* over plain-text protocols *(e.g. HTTP)*. It is therefore advisable to log onto the secure Wi-Net Service, **SSID: "eduroam"**.

This service is provided for registered users of Lancaster University computing facilities only.
Unauthorised access to this system is prohibited.

3 Steps to get your Wireless Internet Connection

FIRST STEP ~ ~ CLICK ON IMAGE

Clicking on the image to the left will download software that will generate a key, so you can access to the wireless network:

- Accept the download if you are prompted
- Choose "Trust" if the application is signed by "Rene Mayrhofer"
- Wait for the application to start

SECOND STEP ~ ~ WAIT

The Application (shown to the left) shows a map of all the devices around you. You should be able to see your device and the aministator's device in the application.

- Wait for the Administrator to generate a certificate for you.
- You can see the progress as a bar at the bottom of your device icon

THIRD STEP ~ ~ IMPORT CERTIFICATE

A message box will appear when the certificate has been generated. The certificate acts like a password, once stored, you never need to type any password, to access this network.

- Click Yes, when you get asked to import the certificate

~ ~ CONNECTED ~ ~

A lock should appear on your icon. As soon as you see that, you are ready to go online and explore the net.

(a) Method 1: authentication with standard network accounts

(b) Method 2: authentication with IPSecME

Fig. 5 Screen shots of respective captive web pages

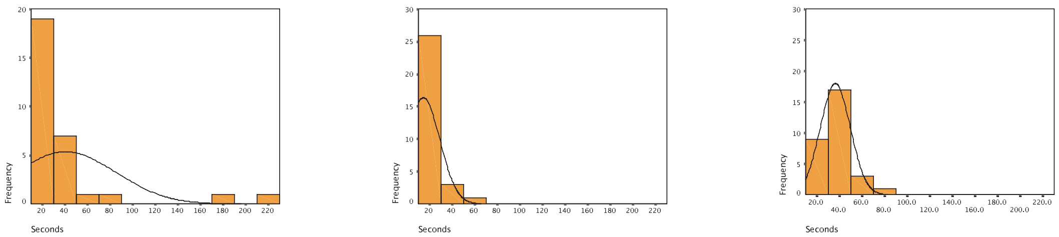
Figure 5b shows the captive portal web page as displayed to unauthenticated guests by our IPSecME WLAN gateway. This page simply allows to download the Java Webstart client application as shown earlier in Fig. 4. The Relate dongles were already plugged into the laptop, and subjects were told to follow the instructions on the web page. These instructions proved to be sufficient for subjects to use IPSecME for gaining network access.

For finishing the defined task of accessing the Google web page using both methods, the subjects required around 5 minutes on average. Measured variables were the number of errors, required time to read the captive web page, and time from starting the respective first step of authentication until successfully finishing it. For both methods, the investigator then closed the browsers and asked the subjects to perform the same task a second time, simulating subsequent network access, e.g. the next day of a visit. Variables were only recorded for the first access, the second aimed at examining user satisfaction. Finally, subjects were asked the following questions for both methods:

- I found the method easy to use for one-time access.
- I found the method fast to use for one-time access.
- I found the method easy to use for subsequent access.
- I found the method fast to use for subsequent access.

Answers to the above questions were a seven-point Likert scale with ratings from 1 ("strongly agree") to 7 ("strongly disagree"). Additionally, users were asked which method they liked more and which method they felt was more secure. On the next page, i.e. only after answering these questions, subjects were asked if:

- Were you aware that the IPSecME method provides encrypted connections? (Yes/No)
- Are you concerned about someone recording your Wireless Network usage (web sites, email)? (Yes/No)



(a) Password-based method: authentication

(b) IPSecME method: reading captive web page

(c) IPSecME method: authentication

Fig. 6 Results: required times for gaining network access

Results Due to the complexity of the whole process of gaining network access and the large underlying differences between the methods, study 1 was split into two phases.

A preliminary study with 15 subjects, 26.7% female, 73.3% male, was used to exploratively discover issues in user interaction and understanding, and to refine the exact study procedure and questionnaire so as to reduce any study bias towards either of the methods as far as possible. As a result, we were able to improve the user interface for making the underlying steps of IPSecME clearer. Especially the involvement and usage of the certificate is now visualized in multiple places, as this turned out to be an unknown concept to many users.

The main study was conducted with 30 different subjects (non-overlapping with the preliminary study), 40% female, 60% male. 80% had used the existing Lancaster WLAN before. For the existing Lancaster password-based method, 2 subjects made 1 error during entering the password and 3 subjects made 3 errors. For the IPSecME method, there were no user errors at all, but for 12 subjects the Relate authentication protocol failed due to distance measurement errors on the ultrasound channel and had to be repeated (we refer to [13] for a more detailed description of false negatives in the protocol). In this case, users (on the client) needed to acknowledge a dialog box stating that the protocol failed and the investigator (on the authentication proxy) restarted the device authentication.

Figure 6 shows the times people took for reading the IPSecME captive web page (with a mean of $\mu = 14.76$ seconds and a standard deviation of $\sigma = 14.58$ seconds) and to complete the respective authentication methods ($\mu = 40.95$ and $\sigma = 44.39$ for the existing, $\mu = 36.63$ and $\sigma = 13.27$ for the IPSecME method). For the existing password-based method, reading times were negligable due to the high familarity of most subjects. In the direct comparison, 15 subjects preferred IPSecME, 4 preferred the existing method, and the remaining 11 had no clear preference, but acknowledged advantages and disadvantages of both methods. Although common tests for significance can not be applied in this case, it is interesting to note that all 5 subjects who made errors during typing in the password for the existing method preferred IPSecME. 15 subjects felt that IPSecME was more secure (10 of which also stated that they preferred IPSecME), 9 felt that the existing method was more secure, and 6 could not decide. Without explicitly pointing it out, 14 subjects were aware of the fact that IPSecME provided encrypted connections after authentication, and 16 were not. 24 were generally concerned about anybody recording their wireless network usage when using insecure access methods.

For first time access, a Wilcoxon Signed Rank test does not show statistically significant differences for either of the questions (see Table 1: a rating of 1 means “strongly agree”, 2 means “agree”, and 3 means “slightly agree”). However, there are statistically significant

Question	Access	Median existing	Median IPSecME	z	$p <$
"easy to use"	one-time	2	3	-1.21	0.226
"fast to use"	one-time	2	2.5	-0.81	0.42
"easy to use"	subsequent	2	1	-3.85	**0.0001**
"fast to use"	subsequent	3	1	-3.98	**0.0001**

Table 1 Results: rated answers to questionnaire and Wilcoxon Signed Rank test results

differences for subsequent accesses, indicating that our subjects rated IPSecME higher than the existing password-based method for subsequent accesses both in terms of ease of use and of speed.

Discussion The general impression of study 1 is that, even though IPSecME is a new and unknown method and sometimes produces authentication errors that require a retry, our subjects were comfortable using it for the first time and rated it similarly to the existing method. For subsequent access, IPSecME is rated significantly better, which is unsurprising due to the automatic reconnects within the lifetime of the certificate, compared to the need for re-authentication on each access using the existing method.

When accumulating reading and authentication times, IPSecME takes on average about 10 seconds longer, but it can be argued that reading the web page is a one-time task, while the authentication process itself is quicker. Our subjects seem to implicitly have taken this into account, as IPSecME was rated only slightly lower in terms of one-time authentication speed.

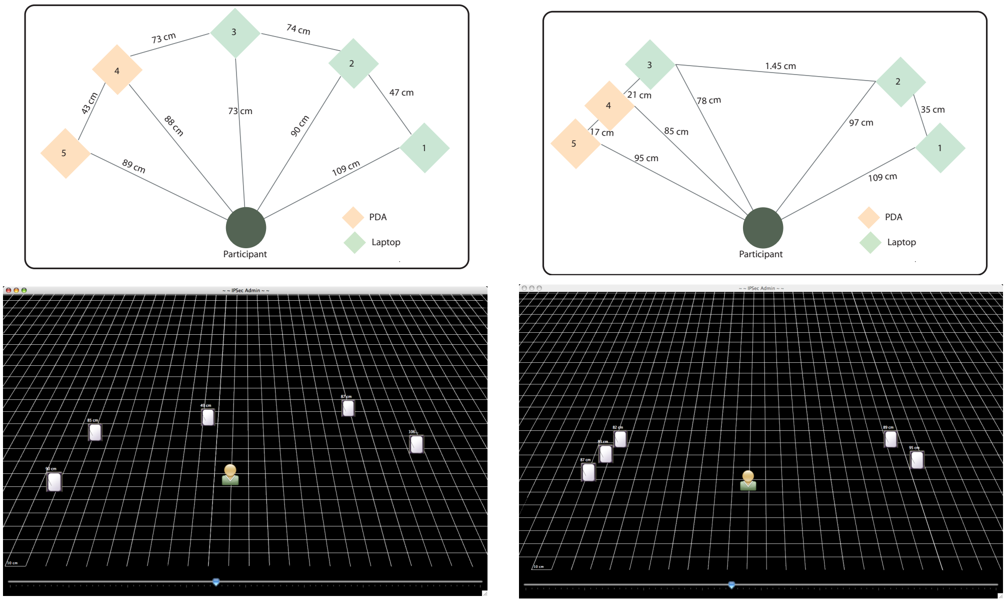
From additional, informal answers given by the subjects, we found that this convenience provided by installing a certificate on the client machine is seen as a major advantage. In the preliminary study, a few users expressed concerns about running additional software (the IPSecME client application) on their machines, while this did not appear as an issue in the main study. This is presumably due to improvements in the user interface to more clearly indicate what the client application does and how the certificate is being used.

6.2 Study 2: Selecting real-world devices with a spatial GUI

Experimental design The second study examines our method from the authentication proxy point of view and investigates how well people deal with our spatial selection method and user interface. 30 subjects were seated at a specific place in front of a meeting table and asked to use a laptop with 15" display, mouse, and attached Relate dongle for selecting different devices using our spatial user interface.

Figure 7 shows the two investigated settings with slightly different placement of the other 5 devices that were equipped with Relate dongles. Every device had its number printed on the case and clearly visible to the subject. To alleviate the influence of a training bias, an initial task used setting 1 for training purposes. With only devices 2 and 4 present, the procedure for the following tasks was explained: after the investigator mentioned a device number, the subject should select the corresponding device icon in the spatial user interface by right-clicking on it and then clicking on the pop-up menu item. Actions and errors were not recorded for the training task of selecting device number 4. Figure 8 shows the simplified placement of devices 2 and 4 and the spatial user interface as it was seen by the subjects.

To focus on the spatial selection, our user interface uses the same icon for all devices and does not show any identification information; the only textual information shown is the



(a) Setting 1: equidistant spacing

(b) Setting 2: clusters

Fig. 7 Placement of devices on the table and in the spatial user interface

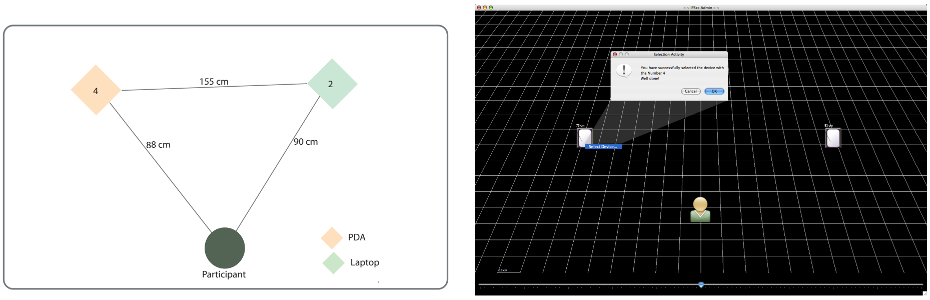
respective measured distance to the device. Therefore, different positions are the only distinguishing criteria. Informally, we discovered that subjects did not use the printed distances as an aid, but mostly the placement of the other devices relative to each other.

The task was for subjects to select 4 different devices, 2 in each setting. First they were asked to select device 2 followed by device 5 in setting 1, then device 1 followed by device 3 in setting 2. While changing the setup from setting 1 to setting 2 by slightly moving the devices, subjects were able to watch the laptop screen and follow the movement in the spatial user interface. Figure 7 also shows the respective spatial user interfaces.

An additional task was used to investigate correlations between the ability to estimate distances to and between real-world objects and perceived difficulties in mapping spatial relationships with our user interface. Subjects were asked to estimate the distances:

- between themselves and the door (2.60 m)
- the width of the door (1 m)
- between devices 2 and 3 (1.45 m)
- the width of the table (2.8 m)

The absolute distances between the estimates given by the subjects and the real distances were accumulated for each subject. In addition, the investigator asked how easy the subjects found the mapping task (rating 1 to 7). The whole study took around 7 minutes per subject on average. Measured variables were the time from when the investigator named the real-world device until the subject right-clicked on the correct item in the spatial user interface and the number of errors until the correct device was selected.

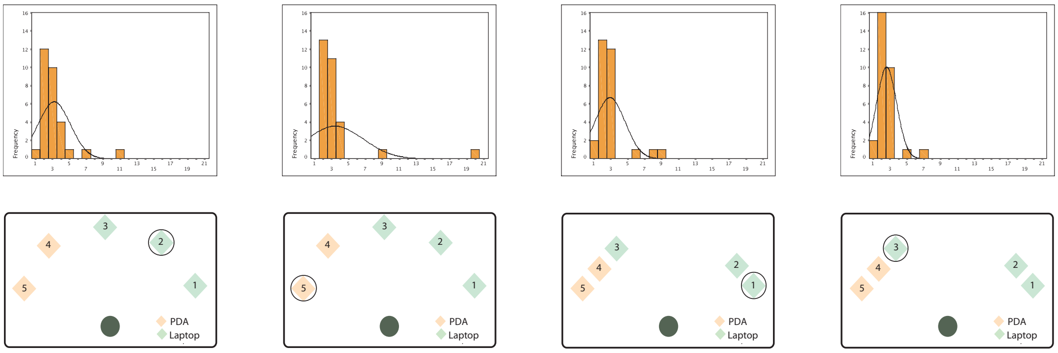


Selection Activity

You have successfully selected the device with the Number 4
Well done!

(a) Simplified setting 1 for training (b) Spatial user interface during training task

Fig. 8 Training task



(a) Task 1 (b) Task 2 (c) Task 3 (d) Task 4

Fig. 9 Results: required times for device selection

Results 30 subjects, 23.3% female, 76.7% male, participated in the second study. 25 of these subjects are researchers in computer science, 1 is a researcher from a different area, and 4 belong to the University administrative staff.

Figure 9 shows the measured times the subjects needed to select the correct devices. Mean times for task 1 were $\mu = 3.2$ with $\sigma = 1.91$, for task 2 $\mu = 3.39$ with $\sigma = 3.34$, for task 3 $\mu = 2.89$ with $\sigma = 1.77$, and for task 4 $\mu = 2.58$ with $\sigma = 1.19$ seconds. The numbers of subjects making errors were, for each of the tasks, 2 (with 1 error per subject), 1 (the subject made 2 errors at this task), 1 (only 1 error), and 0, respectively, and the errors are fully disjoint, i.e. made by 4 different subjects. All 4 subjects who made errors answered that the tasks were easy to perform.

We performed three tests to examine statistical correlations:

- The hypothesis of a correlation between the subjects making any error during the mapping tasks and their answer to the question on ease of use was neither accepted not rejected with statistical significance. This is an expected results considering the generally low error rate.

- The hypothesis of a correlation between the accumulated absolute error of distance estimates and the the subjects making any error during the mapping tasks was also neither accepted not rejected with statistical significance.

- For the third test, the accumulated absolute errors were classified into two groups: $[0; 1]$ m and $]1; \inf[$ m. A Mann-Whitney test shows a correlation with the answer to the question on ease of use with $U = 49.5$ and $p < 0.08$. Therefore, people who were better at estimating real-world distances found the task easier to perform, which matches intuition.

Discussion Due to the small error rate, we can not quantitatively characterize the errors. One likely influence seems to be a training effect, because two subjects made an error during task 1, one subject each in tasks 2 and 3, but there were no errors during task 4. There were two surprising findings: First, that task 2 took on average longer than task 1, which is contrary to any learning effect. One possible explanation is that the target device in task 2 was to the left of the subjects and most probably outside their primary field of sight when facing straight; they had to turn slightly to find it. Additionally, the device was smaller (a PDA instead of a laptop). Second, that setting 2 seemed to be generally easier for subjects than setting 1, although it contained partial occlusions of devices from the subject point of view. The most probable explanation is that people can more easily deal with clusters of small numbers than with a homogeneous group of a large number of devices. In setting 2, there are two clusters, 3 devices to the left and 2 to the right, and the numbers of devices are small enough so that people did not need to count for selecting the target device.

The single outlier with a large time in task 2 was caused by the subject who made 2 errors in the same task.

6.3 Study 3: Comparison from an administrative point of view

Another important aspect of wireless network access is its administration and management. Our third study examines IPSecME from an administrator's point of view. With informal demo and interview sessions, we explained our approach to the two network administrators responsible for the Computing Department at Lancaster University.

In an interactive questionnaire, both administrators stated that the currently deployed system was problematic for spontaneous guest access; although the creation of guest accounts, e.g. for meetings hosted at the department, was supported, it was a cumbersome and slow process that was unsuitable for spontaneous access. Standard practice is therefore for the hosts to either share their password or enter it at the guest's mobile device to grant access, both of which is questionable in terms of network security. An additional issue is that MacOS/X and Linux users are not well supported by the policy of periodic password changes. When not logging on to the Windows domain but only to the wireless network, passwords can not be changed and thus expire, forcing users to find a Windows client to re-gain wireless network access.

Although no clear preference for permanent, registered users has been mentioned, the administrators would prefer IPSecME over the current system for managing guest access. The major two reasons for this preference are security and spontaneous access. Security is improved by creating client-specific X.509 certificates on the fly for network access, and guests no longer need to use access credentials of registered users, which significantly improves accountability. Spontaneous access for clients is made easier by allowing local administrators, hosts of meetings and events, or secretarial staff to quickly grant network access while restricting it to specific guest devices, instead of having to interact with centralized authorization databases.

7 Discussion

The concept of an authentication proxy is generally applicable to arbitrary ways of authentication via shared context, and NiaB has already shown that the use of a special instance of an authentication proxy with infrared works well. It has yet to be investigated how well this concept integrates with other options such as cameras or microphones for sensing shared context. Our software [10] has been designed to make the context authentication protocol exchangeable. It is a simple task to change our application to use IrDA like in NiaB, for example, or to use something different like authentication over an audio channel [4] or with mobile phone cameras [15]. Even though practical applications have not yet made use of authentication proxies in those cases, we do not anticipate any major obstacles.

There are also other options for implementing the secure channel after successful authentication. In this work, we use the well-known IPSec protocol, but the different TLS suites, IEEE 802.1x, or IEEE 802.11i are also considered to be secure protocols and may be more appropriate for different application scenarios. Securing WLANs has been chosen as a scenario due to its practicality and wide applicability. By leaving the WLAN itself open and publicly accessible, we can provide public services usable without authentication, and additional access to authenticated users. We already use this possibility to deliver the authentication application to new clients, thus making it unnecessary to require any pre-installed software. This combination of two (or multiple) levels of service is more difficult to achieve with IEEE 802.1x. For purely spontaneous interaction, IPSec transport connections can be used between just two hosts instead of tunnel connections for securing all traffic a host generates.

For trust delegation, there are again multiple possibilities. In our application, we rely on standard PKI techniques, but shared passwords, OpenPGP keys, or even hardware tokens are other examples that can be used with the same concept. The decision of using online or offline relationships between the service and the authentication proxy is also highly dependent on both the application and the trust model. If the trust model allows delegation of trust, then an active authentication proxy can have distinct advantages, especially when a wireless connection to the actual service is not available ubiquitously. The trust relationship then allows pre-authentication of a client to the service, via the authentication proxy, even before any wireless contact to the actual service is possible. This gives more freedom in performing the authentication, because it can be done at any time for later use. Our application demonstrates this by pre-authenticating IPSec connections for accessing a private network securely over an otherwise public WLAN or from the Internet. This use of IPSec connections is often termed “road warrior” support, because the home network can be accessed from anywhere.

The security of our approach builds upon three parts: First, our context authentication protocol is considered secure against known attack scenarios; it uses multiple rounds of an interlock protocol to verify that only a device at a specific relative position can successfully authenticate. Ultrasound sensing is used as a side channel for transmitting information, in a way that is tightly interwoven with the spatial relationship between devices and that prevents man-in-the-middle attacks on the wireless channel (see [13] and [12] for a more detailed analysis). Second, IPSec as a protocol for secure channels is currently considered as one of the most secure standards. Third, well-known PKI techniques delegate trust to the context authentication proxy. We explicitly point out that the security of our proposed use of authentication proxies relies on the physical security of the proxy devices; when attackers can access these proxies physically, they can access resources as defined by the respective trust model. This is not a new restriction — the security of most protocols relies on physical security of some of its components. An active authentication proxy, like the PDA in our example application, might be small and mobile, and thus even more care needs to be taken to protect it.

When comparing our method with others from a user point of view, we need to distinguish two different aspects. One is the ease of use for gaining access to a protected WLAN. Our user study presented in Section 6.1 shows that spatial authentication compares favourably even to a method already known to and used by the subjects, mostly because of forming a longer-lived security association that can be re-used for subsequent network access.

The second aspect is to establish secure IPSec connections, which is not supported by the standard password-based approach, and thus not currently used by most subjects. Due to this lack of real-world comparability, we only have anecdotal evidence that IPSec connection set-up is significantly eased by our use of an active authentication proxy: For comparable security using e.g. the web administration interface of Gibraltar firewall, an administrator first needs to log in and navigate to the certificate management module (4 steps), create a new certificate for the client (10 fields in a web form), and download it. Then this certificate needs to be imported on the client machine (manual transfer of the file, e.g. with a USB storage device, followed by 14 steps under Windows XP) and an IPSec connection needs to be created (8 steps with the Windows XP wizard). In contrast, using our demonstration application, a new client needs to start the application (1 step, Fig. 4a), an administrator needs to spatially select the client device (1 step, Fig. 4b) and enter the certificate details (2 fields, Fig. 4c). After automatically transmitting the new certificate to the client and importing it, the user only needs to start the IPSec connection (1 step, Fig. 4e). Intuitively it seems clear that it is a considerable improvement over manual configuration. By explicitly assigning the authentication proxy an active role, the end user is relieved from dealing with the connection set-up details at all. This combines into a single step two tasks that are usually separate: the selection, often called *identification*, of a device followed by a proper authentication, and the *authorization* to use some service. We argue that only one step, namely deciding about authorization, is necessary from an administrator point of view and that the authentication step should be made implicit for spontaneous interaction to become viable.

It might become difficult to distinguish devices on the visualized map when too many are presented at once. However, in our user study this did not appear as a problem, and the issue would be implementation specific and is not inherent to the concept of an authentication proxy. We point out that the use of spatial reference for context authentication assumes the availability of appropriate sensors, either built into a device, or attached to it. For example in a meeting scenario, spatial reference is a generally useful tool [9] and using it for granting temporary access to resources – with the approach described in this article – thus integrates seamlessly. In other scenarios, ultrasound sensing might not be readily available for current mobile devices. Although our USB dongles make it easy to attach them, it is an additional step that needs to be done. But, as mobile devices begin to include more sensors, context authentication will be more easily possible in the near future.

8 Conclusions

In this article, we argue that context authentication is more intuitive then typical password- or certificate-based methods, especially for spontaneous interaction. The example of setting up secure WLAN connections shows clearly that these often-used methods do not scale with regards to the number of wireless connections used by a single person. A direct comparison between the number of steps that need to be executed by a user and an administrator for creating such a secure connection between a password-, a certificate-, and a context-based authentication procedure is obviously biased; our demonstration application has been designed specifically to make this as easy as possible, while other methods are usually not aimed at supporting spontaneous interaction. Nonetheless, practical experience shows that those WLANs where simple, spontaneous interaction is desired, such as WLAN hot spots in hotels or airports, either do not use any authentication at all or tend to be seen as awkward by most users. Context authentication allows to provide secure wireless connections without demanding user attention "just for security". Our main contribution is the general concept of a context authentication proxy, which allows devices to use context authentication when they can not actually experience the same sensor values for any suitable aspect of context. A first demonstration application implements this concept for a prominent example, namely WLAN access. The fact that other projects have also approached this scenario shows the practical importance of the problem.

Compared to SWG, we benefit from the use of certificates to provide better security for larger scenarios, where re-keying of the whole system to disable access for a single client is not reasonable. We extend the results of the NiaB project in three areas: First, by making the context authentication proxy active, we give both the clients and the administrator more flexibility in the authentication process. By running a CA on the proxy, the decisions about authentication and authorization can be condensed into only one spatial device selection step to improve ease of use. Second, the proxy is made mobile and supports offline authentication where connectivity to the target network is not available. Third, ultrasound sensing provides more fine-grained selection of devices, and the same granularity is used in the spatial authentication protocol. This allows multiple devices in the same area to be distinguished better, e.g. to grant temporary network access in a meeting scenario with multiple laptops and PDAs on one desk. With an infrared channel like the one used in NiaB, there is no protection against active man-in-the-middle attacks. Therefore, the context authentication needs to be run in a secure environment where such attacks are prevented by organizational restrictions (e.g. that only one device is allowed to enter the authentication room at any time). With our proposed spatial authentication protocol, context authentication is secure even in public and untrusted environments.

Complete source code of our client and proxy implementations is available at `http://www.openuat.org/spatial-ipsec-proxy`, including configuration files for the gateway using Gibraltar firewall and using OpenWrt.

Acknowledgments

We gratefully acknowledge support for the presented research by the Commission of the European Union under contracts 013790 "RELATE" and the FP6 Marie Curie Intra-European Fellowship program contract MEIF-CT-2006-042194 "CAPER", and by the Engineering and Physical Sciences Research Council in the UK under grant GR/S77097/01.

References

1. D. Balfanz, G. Durfee, R. E. Grinter, D. K. Smetters, and P Stewart. Network-in-a-box: How to set up a secure wireless network in under a minute. In *Proc. 13th USENIX Security Symp.*, pages 207–222. USENIX, August 2004.
2. D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proc. NDSS'02: 2002 Network and Distributed Systems Security Symp.* The Internet Society, February 2002.
3. A. Godber and P. Dasgupta. Secure wireless gateway. In *Proc. WiSE'02: 3rd ACM workshop on Wireless security*, pages 41–46. ACM Press, 2002.

4. M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: Human verifiable authentication based on audio. In *Proc. ICDCS 2006: 26th Conf. on Distributed Computing Systems*, page 10. IEEE CS Press, July 2006.
5. P. Gutmann. Plug-and-play PKI: A PKI your mother can use. In *Proc. 12th USENIX Security Symp.*, pages 45–58, August 2003. published at `http://www.cs.auckland.ac.nz/~pgut001/pubs/usenix03.pdf`, shorter version appeared in IEEE Computer Magazine, August 2002.
6. M. Hazas, C. Kray, H. Gellersen, H. Agbota, G. Kortuem, and A. Krohn. A relative positioning system for co-located mobile devices. In *Proc. MobiSys 2005: 3rd Int. Conf. on Mobile Systems, Applications, and Services*, pages 177–190. ACM Press, June 2005.
7. Jens Jakobsen. Chillispot web page. `http://www.chillispot.org`, 2006.
8. T. Kindberg, K. Zhang, and N. Shankar. Context authentication using constrained channels. In *Proc. WMCSA: 4th IEEE Workshop on Mobile Computing Systems and Applications*, pages 14–21. IEEE CS Press, June 2002.
9. G. Kortuem, C. Kray, and H. Gellersen. Sensing and visualizing spatial relations of mobile devices. In *Proc. UIST 2005: 18th ACM Symp. on User Interface Software and Technology*, pages 93–102. ACM Press, October 2005.
10. R. Mayrhofer. Towards an open source toolkit for ubiquitous device authentication. In *Workshops Proc. PerCom 2007: 5th IEEE International Conference on Pervasive Computing and Communications*, pages 247–252. IEEE CS Press, March 2007. Track PerSec 2007: 4th IEEE International Workshop on Pervasive Computing and Communication Security.
11. R. Mayrhofer and Esys GmbH. Gibraltar firewall web page. `http://www.gibraltar.at`, 2006.
12. R. Mayrhofer and H. Gellersen. On the security of ultrasound as out-of-band channel. In *Proc. IPDPS 2007: 21st IEEE International Parallel and Distributed Processing Symposium*, page 321. IEEE CS Press, March 2007. Track SSN 2007: 3rd International Workshop on Security in Systems and Networks.
13. R. Mayrhofer, H. Gellersen, and M. Hazas. Security by spatial reference: Using relative positioning to authenticate devices for spontaneous interaction. In *Proc. Ubicomp 2007: 9th International Conference on Ubiquitous Computing*, LNCS. Springer-Verlag, September 2007. *to appear.*
14. R. Mayrhofer, F. Ortner, A. Ferscha, and M. Hechinger. Securing passive objects in mobile ad-hoc peer-to-peer networks. In R. Focardi and G. Zavattaro, editors, *Electronic Notes in Theoretical Computer Science*, volume 85.3. Elsevier Science, June 2003.
15. J. M. McCune, A. Perrig, and M. K. Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *Proc. IEEE Symp. on Security and Privacy*, pages 110–124. IEEE CS Press, May 2005.
16. OpenWrt. OpenWrt web page. `http://openwrt.org`, 2006.
17. R. Steffen and R. Knorr. A trust based delegation system for managing access control. In *Advances in Pervasive Computing: Adjunct Proc. Pervasive 2005*, volume 191, pages 1–5. Austrian Computer Society (OCG), April 2005.
18. C. Swindells, K. M. Inkpen, J. C. Dill, and M. Tory. That one there! pointing to establish device identity. In *Proc. UIST '02: 15th ACM Symp. on User interface software and technology*, pages 151–160. ACM Press, 2002.
19. Xelerance Corporation. Openswan web page. `http://www.openswan.org`, 2006.

11 Securing Passive Objects in Mobile Ad-Hoc Peer-to-Peer Networks

Authors: Rene Mayrhofer, Florian Ortner, Manfred Hechinger, and Alois Ferscha

Published in Electronic Notes in Theoretical Computer Science volume 85.3, Elsevier Science
ISSN 1571-0661
June 2003

Electronic Notes in Theoretical Computer Science 85 No. 3 (2003)
URL: http://www.elsevier.nl/locate/entcs/volume85.html 17 pages

Securing Passive Objects in Mobile Ad-Hoc Peer-to-Peer Networks

Rene Mayrhofer [a] Florian Ortner [a] Alois Ferscha [a]
Manfred Hechinger [a]

[a] *Institut für Praktische Informatik*
Johannes Kepler Universität Linz
Altenberger Str. 69
A-4040 Linz
Austria

Abstract

Security and privacy in mobile ad-hoc peer-to-peer environments are hard to attain, especially when working with passive objects (without own processing power, e.g. RFID tags). This paper introduces a method for integrating such objects into a peer-to-peer environment without infrastructure components while providing a high level of privacy and security for peers interacting with objects. The integration is done by associating public keys to passive objects, which can be used by peers to validate proxies (peers additionally acting on behalf of objects). To overcome the problem of limited storage capacity on small embedded objects, ECC keys are used.

1 Introduction

Currently, mobile ad-hoc networks (MANETs[3]) are a highly active research topic with many publications covering different aspects of this inter-disciplinary field (e.g. [17]). These aspects include, but are certainly not limited to, hardware (e.g. size, rugged design, power consumption, communication), software (e.g. operating system/platform, communication protocols, memory usage), interaction (e.g. interaction models, HCI aspects), security and application issues. In this paper, we will focus on privacy and security aspects of ad-hoc, peer-to-peer networks.

The SmartInteraction project is an approach to interact with persons, things and places in a natural and non-obtrusive way. As for example people meet each other, their “interaction profile” is mutually compared in analogy

[1] This work has been developed in cooperation with Siemens CT SE2, Munich

©2003 Published by Elsevier Science B. V.

to their natural, automatic choice of sympathy. Following the vastly successful way of human interaction, the Peer-to-Peer (*P2P*) paradigm is used for direct communication among all participating devices. This offers complete device autonomy, independence of central authorities and reliability due to redundancy. Within the SmartInteraction project, this principle is even taken one step further by also being independent of any common communication infrastructure: we utilize solely ad-hoc wireless networks, currently either IEEE802.11b Wireless LAN (*WLAN*) or IEEE802.15.1 Bluetooth (*BT*). To match the flexibility of the P2P approach, local profiles describing the device capabilities, user attributes and preferences are kept on every peer. Upon spatial contact with other peers, these profiles provide the base for matching user interests and determining further coordination. Additionally, context constraints defined in profiles provide the necessary context awareness for ubiquitous applications; different situations, identified by context parameters, demand different behavior. As in any ubiquitous system, privacy and security are major concerns and are taken seriously by utilizing active and passive privacy control backed by strong cryptography. This paper does not aim to develop new cryptographic algorithms or novel security protocols, but instead utilize and combine well-known and secure techniques. However, we were unable to find protocols or methods for securely integrating passive objects without own processing capabilities into a P2P infrastructure. As this is an issue in our project, it was necessary to develop a method to secure remote proxies that act on behalf of passive objects; this is the main contribution in the present work.

This paper is organized as follows: In section 2, we start by shortly explaining the hard- and software environment the SmartInteraction project is situated in, including our definition of (passive) objects. Section 3 then gives an overview of related work, while section 4 presents our approach to P2P privacy and security between powerful peers. An improvement to this approach to securely integrate (passive) objects with powerful peers – the main contribution – is presented in section 5. After that, we give a short conclusion and an outlook on our planned future research in section 6.

2 Environment

The SmartInteraction project aims to provide a flexible framework for ad-hoc, mobile P2P interaction of multiple, heterogeneous devices. A software framework has been developed which handles many aspects of ad-hoc, P2P interaction and therefore allows the efficient construction of applications in this fast-growing domain. This paper focuses on security aspects of the framework, which is able to run on a wide range of platforms (*peers*); the only requirement is a Java 1.1 compatible JVM and arbitrary communication technology. However, we also want to integrate devices without any processing capabilities (*objects*) into the P2P interactions.

2.1 Peers

For fully distributed, instantaneous, ad-hoc P2P interaction, processing capabilities are required on each interaction partner. These so-called *peers* can run our software framework, which allows them to discover and communicate with each other. Possible platforms for peers are standard servers (especially for remote proxy peers, described in the next section), notebooks, sub-notebooks, handhelds, PDAs or even mobile phones. Small, mobile devices will normally have limited resources such as processing power, RAM or storage capacity, but they are nonetheless capable of securing their own communication with strong cryptography (see appendix A).

2.2 Objects

As already mentioned, we do not want to restrict ourselves to only integrating peers in the P2P interactions, but we also want to have *objects* participating. In our environment, we define an *object* in the following way:

> An object is passive with regards to to executing custom code, i.e. it does not have processing power that could be exploited to run parts of a custom software. Objects are required to have a unique identification number (*ID*).

This definition does not exclude objects having a CPU and carrying out computations. Thus, the following devices are examples for objects in our definition:

- RFID (*Radio Frequency Identification tags*[7]): These are either passive (powered by the interrogator) or active (with own power supply), via RF (Radio Frequency) accessible small memory devices, which are available for a broad range of applications from multiple vendors like Identec Solutions, Inside or Texas Instruments. An Identec iD-2 tag and the i-CARD PCMCIA reader are shown in Fig. 1.
- IrDA (*Infrared Data Association*) beacons: These are active devices, periodically sending infrared packets that can be received by any device with a standard IrDA port (e.g. notebooks, PDAs, mobile phones). IrDA beacons have already been deployed on larger scale for various applications (e.g. [19]).
- Bluetooth (IEEE 802.15) devices: There already exist many Bluetooth devices with own processing power which could be peers in our definition. But even Bluetooth devices without own processing power can be used as objects by utilizing their MAC address as unique object ID.

As those devices, especially the RFID tag technology, become smaller with each generation, embedding them into real-world objects (e.g. food packaging, clothes[2], books, posters, doors or any other tangibles[3]) allows those

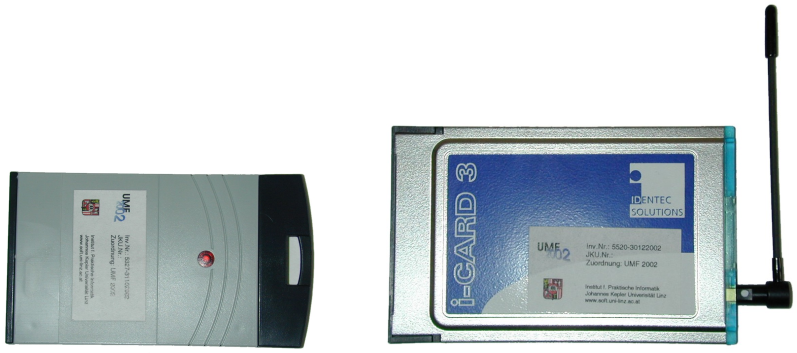
[2] Benetton recently adopted Philips RFID technology for 'smart' labels

[3] Tangible interfaces try to give physical form to digital information[10].

real-world objects to take part in interactions within the SmartInteraction project, creating a digital representation of the real-world object. This digital representation can accomplish any appropriate task to support the real-world interaction, e.g. offering detailed information on food nutrition or allowing to place reviews on a book borrowed from a public library. There are two possibilities for integrating objects into a P2P interaction environment that allow peers to interact with objects:

- Local proxies: One possibility is to keep the actual data on the objects themselves (e.g. on a RFID tag's custom data storage area) and process it on the peers that wish to interact with the objects. On the peers, a wrapper acts as local proxy for the object, performing all computations and possibly also modifying the data on the object.
  The obvious disadvantage is that a wrapper for each type of application and type of object must be installed on each peer that wishes to interact with those objects. Furthermore, Securing objects is virtually impossible when peers are allowed to modify the object's data (e.g. posting reviews on a book), because the objects themselves, having no processing capabilities, are unable to control access to that data.
- Remote proxies: The other possibility is to only keep a unique ID on the object itself and set up remote *proxy peers* that act on behalf of the objects. When detecting an object, an *ordinary peer* will store its ID and try to find a peer which represents that object, i.e. which is a remote *proxy peer* for this ID (*synchronous proxy interaction*). When no peer currently in range claims to be responsible for the object ID, it will start the interaction as soon as one becomes available (*asynchronous proxy interaction*). This allows very flexible interaction patterns between peers and objects (represented by proxy peers) because the proxy can be arbitrarily complex. More importantly, we are able to guarantee object privacy and security with this scheme, which is our main contribution in this paper. We would like to point out that the distinction between an ordinary peer and a proxy peer only stems from the applications running on them; the underlying framework is equivalent on both, the proxy peer just declares to be responsible for certain object IDs. The different terms are only used to distinguish both sides of a P2P communication in protocol explanations.
  However, the disadvantage is that an additional peer is needed for the interaction. For some applications, a hybrid scheme may be appropriate: read-only data could be stored directly on the object to communicate first information and a remote proxy could be available for further, more flexible interaction.

For the SmartInteraction project, we decided that the flexibility and security of the remote proxy approach outweighs the disadvantage. Therefore, in the remainder of the paper we will only talk about this method.



(a) iD-2

(b) i-CARD

Figure 1. Identec RFID hardware

3 Related Work

To the best of our knowledge, this is the first attempt to bring up the topic of securely integrating passive objects in an ad-hoc, P2P environment.

Like Frank Stajano pointed out, it is not possible to provide a certificate authority (an online server for all peers) for authentication in a highly dynamical, ad-hoc P2P environment[18, pages 85ff]. Additionally, he suggests to exchange all information which is needed for security measures (certificates, keys) during the bootstrap phase like our system does[18, pages 91ff].

The Freenet project[2] was probably one of the first projects to integrate high-standard privacy and security in a completely distributed P2P architecture.

Marc Langheinrich described a privacy awareness system integrated into an ubiquitous computing environment[13], building on the P3P standard by the W3C (World Wide Web Consortium). The difference to our system is that it depends on infrastructure components whereas our approach is completely based on the P2P paradigm.

The W3C published another recommendation for implementing XML signatures[5] that provide integrity, message authentication and signer authentication for data of any type.

Additionally, there are numerous papers about security measures for RFID technology, which focus on the physical layers[16] (e.g. preventing denial-of-service attacks). Security in sensor networks is currently also a very active research topic (e.g. [15]).

This paper builds upon well-known methods, but integrates passive objects into a fully distributed P2P security concept, which differentiates it from previous publications.

4 Privacy and Security in mobile ad-hoc networks

4.1 Motivation

For pervasive computing environments in general, security is an important issue because the possibilities for attacks are enormous. Mobile devices like notebooks, PDAs or mobile phones usually carry important data like the user's phone numbers, calendar, notes and other private data (cf. [18]). Following the Code of Fair Information Practices (*FIPs*)[8], any information processing system (mobile ad-hoc P2P systems are in essence only information processing systems) must assure the reliability of data and prevent misuse (principle 5: security). In addition to ensuring this required data security and reliability, our privacy control addresses principle 1 (openness) and partially principle 3 (secondary usage) with our concept of active privacy. Principles 2 (disclosure) and 4 (correction) require organizational precautions in our environment and are thus not covered by this technical solution.

The core objective in this approach is to provide a high level of privacy and data security to the users of mobile, ad-hoc P2P systems, not the secure authentication of users to infrastructure components or any other application that is not focused on the user's own privacy. Therefore, all decisions and policies concerning privacy and security should be local to the respective peers that participate in some secure environments.

Basically, there are two different aspects of privacy from the user's point of view, to be tackled with two different privacy policies:

- "Passive" privacy: The goal is to shield the user from incoming information and only present desired messages. As we are all inundated with information, protection has become a necessity. By means of profile matching (to match interests and preferences), the SmartInteraction Framework already provides good shielding to the user, but it can be enhanced by using authentication information in the shielding process.
- "Active" privacy: The goal is to filter outgoing information and only allow non-private information to leave the peer. One possibility to implement it is to allow a fine-grained definition of "access control" in the local peer's profiles and to use authentication information to determine the level of trust in other peers.

In this paper, by the term "privacy control", we describe the active software component that actually makes the decisions on active and passive privacy - it determines which messages are allowed to be sent or received. With the term "privacy policy", we describe the set of rules and preferences a user defined for the mobile device - they will be enforced by the privacy control.

4.2 Peer-to-Peer Security

Communication between ordinary peers is based on XML-messages. Consequently, it is necessary to secure those. This is accomplished with a hybrid system similar in design to PGP which uses both symmetrical (with session keys) and asymmetrical encryption (with private/public key pairs) and digital signatures for authentication.

Our current architecture uses well-known techniques, featuring a very high privacy and security level while operating in an ad-hoc P2P environment and retaining maximum autonomy of peers:

- Use of hybrid encryption.

  Symmetric encryption: To comply with the current best practices, it is advisable to use Rijndael, the AES winning cipher, as the symmetric cipher: it is secure, well-analyzed and fast (the speed penalty compared to RC6 is tolerable, cf. table A.1). In this document, Rijndael will simply be named AES. However, there are some doubts on the security of Rijndael[4], which are currently only theoretical. AES ciphers have a block size of 128 Bit and possible key lengths of 128, 192 and 256 Bit, but some (including Rijndael) are capable to use keys with a higher length (and can therefore be adapted to a higher security level).

  Asymmetric key management:

  · RSA: RSA has the main advantage that it can be used for creating digital signatures as well as for asymmetric encryption, requiring only one keypair for each role.

  · EC ElGamal/ECDSA: Elliptic curve cryptography (ECC) variant of the ElGamal key exchange algorithm. ECC keys offer the same level of security as other methods with significantly smaller key sizes[14] (e.g. 163-bit ECC in contrast to 1024-bit RSA). However, it seems to be generally slower.

  Digest generation: For computing digests of messages, the standard SHA256 algorithm is used. Digests are generally created over the whole XML message and then signed with the private key associated to the respective role that sent the message.

- Use of the X.509v3 standard for issuing and validating certificates.

  For proving the authenticity of (public) keys and their association to roles, certificates are needed. These certificates are issued by certificate authorities (*CAs*) and bind the public key to the role description, digitally signed by the private key of the CA. For mutually authenticating peers that do not know each other directly, certificates seem the best option. However, since a single, hierarchical PKI poses many security risks[6], we opt to not depend on one. Instead, we will utilize multiple, independent CAs that are completely autonomous (there will be no hierarchical structure between CAs) as well as webs of trust between users.

  X.509v3[9] is a well-established standard for certificate formats and is, among others, used for SSL/TLS and S/MIME. Therefore, it is used far more often than OpenPGP as a mere certification standard (outside the domain of email and Usenet net news). There are various free implementations for handling X.509v3 certificates (e.g. OpenSSL, SUN Java JSSE), including Java libraries. The main advantage of X.509v3 is the possibility to define arbitrary fields in the certificate, which can be used to add meta-data (e.g. adding the department in addition to the company name). Although this is also possible with OpenPGP, it would need to be done application-specific – X.509v3 offers this in its standard form. Additionally, X.509 supports Certificate Revocation Lists

  - Certificates can be issued by multiple CAs and each device can store multiple independent certificates for the different roles of its user.
  - Key pairs, role certificates and CA certificates are transferred to the device in a bootstrap phase.
  - Certificates are exchanged between peers before actual data is transferred. When an ordinary peer wants to use authentication or encryption to communicate with another peer, both have to exchange their profiles. Therefore, certificates are automatically embedded in the first message (cf. Fig. 4, step 5).

- All verification, validation and authentication decisions are made locally and autonomously by each peer. In a mobile P2P environment we can not rely on an infrastructure with central servers that are constantly available; thus the devices are forced to be completely autonomous. This not only enhances the privacy of users by keeping important decisions local, but also allows to produce a detailed log of which personal data was sent to whom.

Although our current work concentrates mostly on mutual authentication of peers via certificates, signed and encrypted messages, the privacy control should be able to intervene with all parts of the framework. One example would be to completely turn off the radios of all wireless communication channels on the hardware layer, becoming fully invisible to other peers.

5 Integration of objects

5.1 Problem description

When trying to integrate passive objects (according to our definition in section 2.2) into a security infrastructure, a number of problems arise. The main cause is that an object is, due to not having any processing power, unable to perform any authentication – neither authenticating itself nor verifying the authenticity of other peers. In Fig. 2, the interaction with an object is depicted. After detecting an object O in range, the ordinary peer A will search for a proxy peer B which claims to be responsible for O. Because no direct interaction between O and A is possible (A can only detect O and read it's ID and possibly some custom data), all authentication will need to be performed

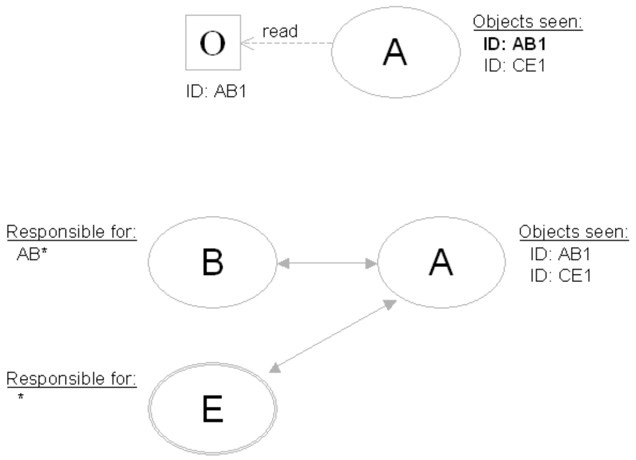


Figure 2. Communication between an ordinary peer and an object via a remote proxy

between A and B. As already mentioned in the architecture description, we can not depend on a single trusted third party like a CA to certify A and B and therefore the validity of B's responsibility for O. In real-world scenarios it will be virtually impossible to pre-authenticate A and B (without a single trusted third party) via a web of trust. Thus, it would be possible for an attacker E to claim responsibility for objects (by setting up a remote proxy for the respective IDs) she/he doesn't own, opening up a number of security threats such as:

- Interception of data: When A tries to send private data O, E could easily intercept this data by pretending to be a valid proxy for the respective object.
- Forgery of data: E could send forged data to A, pretending to be legitimately representing O and thus exploiting A's possible trust in O.
- Tracking of users: E could construct a proxy peer claiming to be responsible for all objects in some geographical area and capturing and logging all communication requests. Since ordinary peers will try to contact proxies when interacting with objects, they will also contact E as pretended proxy. E can then track the movement of peers in the geographical area, which can be seen as a severe threat to the privacy of ordinary peers (cf. [16, section 4.1]).

A has, in this situation, no possibility to distinguish between the claims of B and E as depicted in Fig. 2.

In the following, we present a solution to these privacy threats. However, due to the nature of our project environment, it is inherently impossible (on this level) to prevent against one additional threat, which we also want to describe: An attacker could place an object P (or multiple objects) and a cor-

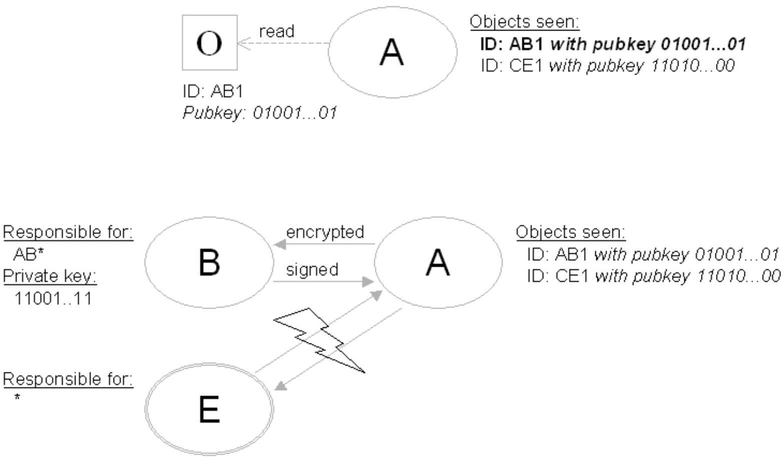


Figure 3. Secured communication between an ordinary peer and an object via a remote proxy

rectly associated proxy peer E in spatial proximity to the real object, gaining a reasonable chance that ordinary peers will find P instead of O and thus contact E instead of B. This "physical attack" can not be overcome with software tools (cf. [18]). However, this sort of attack can successfully be prevented by using certificates to authenticate B, as described in section 4.2. A can then validate the authenticity of B and thus refuse to connect to E when it does not provide a valid, accepted certificate.

Finally, we want to note that it is not necessary to protect against malicious ordinary peers on this level. When the proxy peer is trusted, attacks by ordinary peers can also be inhibited using the techniques described in section 4.2.

5.2 Solution

Our proposed solution is to store a public key on the object itself, and the associated private key on the proxy peer that is responsible for interacting on behalf of the object. As depicted in Fig. 3, O stores a public key in addition to its ID. This key is either an EC (elliptic curve) or RSA key, depending on the available storage area for custom data. The reason for choosing EC ElGamal/ECDSA[12,1] as asymmetric algorithms is that keys with a significantly smaller size offer the same level of security, compared with RSA. On the Identec i-D2 active RFID tags, there are only 64 Bytes available for storing the public key. Table 1 lists the sizes (in Bytes) of public keys in (binary) DER[11] encoding, generated with the openssl library [4]. When using 160 Bit

[4] The openssl cryptographic library offers a wide range of symmetrical and asymmetrical algorithms, including certificate authority functionality, and a command line interface for accessing them. It is freely available at http://www.openssl.org/.

Algorithm	Parameters	Key size
RSA	1024 Bit modulus	162
DSA	1024 Bit prime	442
EC	secp160r2 curve	64

Table 1
Public key sizes

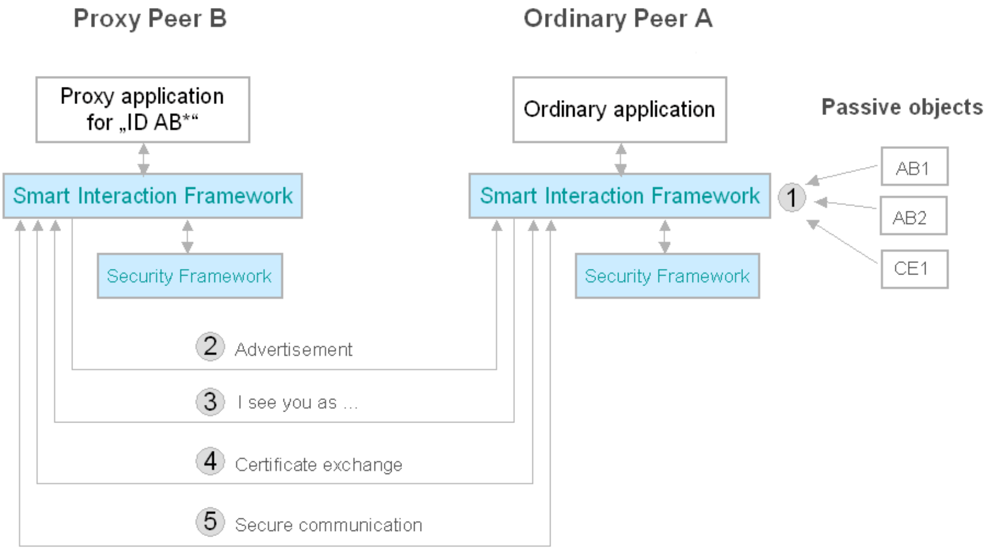


Figure 4. Initiating an interaction between a peer and an object via a remote proxy

prime fields for EC, which is considered to be comparably secure as using a 1024 Bit long modulus for RSA (e.g. [14]), the public key will fit perfectly into the objects's custom storage area, even in a standard encoding format. If more storage is available on the used object technology, standard RSA public keys can be used for better run-time performance.

The respective private key (regarding the public key stored on O) belongs to B and is kept there. When reading objects in range, A stores all public keys in an internal table, associating them with the object IDs from which the public keys were read. This table then allows A to decide locally and autonomously if a proxy peer that claims to be responsible for O is valid.

Protocol description

- All messages from an ordinary peer to a proxy peer (from A to B) are AES-encrypted with a temporary session key. Using the public key from the internal table (the public key stored on O), A can send the session key to B utilizing either EC ElGamal or RSA.

- All messages from a proxy peer to an ordinary peer (from B to A) are signed, either with ECDSA or RSA using the private key stored on B. A can then verify all messages received from B for integrity and validity using the public key from its internal table. This includes the announcement messages (cf. Fig. 4, step 2) sent by B. Thus, E can not send messages that A considers as valid, not even the announcement message where E claims to be responsible for O.

- Further security measures to obviate other attacks can be applied one protocol level higher. On this level, the communication is transparent to both involved peers and can thus be handled as it would be between two ordinary peers, including the use of certificates for authentication. Our protocol on this level is shown in Fig. 4:

Step 1 An ordinary peer A finds passive objects (AB1, AB2 and CE1) and stores their IDs and public keys (EC ElGamal/ECDSA or RSA) in a local table.

Step 2 Proxy peer B sends a signed advertisement to A where it claims responsibility for a set of passive objects (AB*).

Step 3 A compares the advertisement with the entries in the local table and reports successful matches (AB1 and AB2) to B, thus notifying the proxy application of the actual object IDs it should act for.

Step 4 Exchange of all needed certificates between A and B (with RSA keys).

Step 5 Finally, secure communication between A and B on behalf of AB1 and AB2 is possible. Messages are fully signed and encrypted in both directions, using the RSA keys from the exchanged certificates (equivalent to normal interaction between ordinary peers).

It is important to note that B does never send a cryptographic key (neither a temporary, symmetric session key nor an asymmetric public key) to A until the certificate exchange; A either generates the key (the session key for AES encryption) or uses the public key that was read from the object, fulfilling our requirement of autonomy. Thus, it is impossible for E to spoof messages from O and to read messages destined for O. However, this holds true only if two assumptions are fulfilled:

- The private key belonging to the public key stored on O is only stored on B and kept safe. As this is a standard requirement for the usage of private keys, it is a matter of physical security of B. In a typical scenario, the remote proxy peers will either run on trusted embedded devices or on tightly controlled servers, allowing to secure the key.

- The public key stored on O can not be changed by E. This is usually guaranteed with read-only objects that can be written only once (e.g. fuses in RFID tags) or by password-protected write access to the objects. Practically, this places no restriction on the possible scenarios because only the *physical* object needs to be read-only, not its virtual counterpart (represented by the remote proxy). With the virtual representation in the P2P environment, any interaction is possible, including operations that modify data on the remote proxy, associated with the physical object.

We want to point out that this solution will not be able to provide complete privacy and security on its own, it has to be seen as an addition to the standard methods described in section 4.2. However, without an addition, a secure integration of objects as defined in section 2.2 does not seem to be possible. When both layers (as laid out in this paper) are used, we are now able to provide a high level of privacy and security, even in this difficult environment.

6 Conclusions and Future Work

We have introduced a method to securely integrate passive objects (basically re-sourceless peers with a unique identification number) into an ad-hoc P2P environment. After introducing the SmartInteraction project and defining the term "object" in the context of this project, a method for transparently interacting with objects which do not have their own processing power has been given; the most flexible approach for such an interaction is via remote proxies that act on behalf of the objects and are themselves peers. The problem with this kind of interaction via a remote proxy peer (responsible for a list of objects identified by their ID) is that three parties are involved in the authentication process: the object, the ordinary peer (which seeks to interact with the object) and the proxy peer (which differentiates itself from an ordinary peer only by claiming responsibility for objects). Because the object does not have the ability to actively participate in the authentication process, the ordinary peer must use locally available information to verify the authenticity of all proxy peers that claim to be responsible for the object. Our solution in the SmartInteraction project is to put public keys directly on the objects, which must be read-only to ordinary peers. With this technique, an attacker can no longer spoof responsibility for an object.

We have also shown by empirical performance evaluation that strong encryption is possible even on thin client technology (like PDAs), giving rise for using standard cryptographic algorithms. A proof-of-concept implementation of our methods and protocols is available, formal verification of the protocol still has to be done.

As a next step, we will work on another important aspect of privacy and security on mobile devices: configuration by end-users. Because the privacy and security component is very flexible, there are also many aspects that can be configured (e.g. details on certificate validity checks). End-users will generally be unaware of those aspects and unable to set them properly, not knowing about the consequences of each. We will perform a field study to measure the actual usage of certain privacy and security features; this will be possible after the privacy and security module has been completely integrated into a new demonstration application, which will be presented in more detail in future work.

References

[1] *Elliptic curve digital signature algorithm (ECDSA)*, American Bankers Association (1999).

[2] Clarke, I., T. W. Hong, S. G. Miller, O. Sandberg and B. Wiley, *Protecting free expression online with Freenet*, IEEE Internet Computing 6 (2002), pp. 40–49.

[3] Corson, S. and J. Macker, *RFC2501: Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations* (1999).
URL `http://RFC.net/rfc2501.html`

[4] Courtois, N. and J. Pieprzyk, *Cryptoanalysis of block ciphers with overdefined systems of equations.*
URL `http://eprint.iacr.org/2002/044/`

[5] Eastlake, D. E., J. M. Reagle and D. Solo, *XML-signature syntax and processing* (2002), W3C Recommendation, HTML version at http://www.w3.org/TR/xmldsig-core/.

[6] Ellison, C. and B. Schneier, *Ten risks of PKI: What you're not being told about public key infrastructure*, Computer Security Journal 6 (2000), pp. 1–7.

[7] Finkenzeller, K., "RFID-Handbuch: Grundlagen und praktische Anwendungen induktiver Funkanlagen, Transponder und kontaktloser Chipkarten," Carl Hanser Verlag München, 2002, third edition.

[8] *Records, computers, and the rights of citizens viii*, U.S. Dep't. of Health, Education and Welfare, Secretary's Advisory Committee on Automated Personal Data Systems (1973).

[9] Housley, R., W. Polk, W. Ford and D. Solo, *RFC3280: Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile* (2002).
URL `http://RFC.net/rfc3280.html`

[10] Ishii, H. and B. Ullmer, *Tangible bits: Towards seamless interfaces between people, bits and atoms*, in: *Proceedings of Conference on Human Factors in Computing Systems (CHI '97)* (1997), pp. 234–241.

[11] *ITU-T recommendation X.690: Information technology - ASN.1 encoding rules: Specification of basic encoding rules (BER), canonical encoding rules (CER) and distinguished encoding rules (DER)* (1997), also ISO/IEC 8825-1:1998.

[12] **Koblitz, N., A. Menezes and S. Vanstone, *The state of elliptic curve cryptography*, Designs, Codes and Cryptography 9 (1994), pp. 173–193.**

[13] **Langheinrich, M., *A privacy awareness system for ubiquitous computing environments*, in: *UbiComp 2002: Ubiquitous Computing*, Lecture Notes in Computer Science 2498, 2002, p. 237ff.**

[14] **Lenstra, A. K. and E. R. Verheul, *Selecting cryptographic key sizes*, Journal of Cryptology: the journal of the International Association for Cryptologic Research 14 (2001), pp. 255–293.**

[15] **Perrig, A., R. Szewczyk, V. Wen, D. E. Culler and J. D. Tygar, *SPINS: security protocols for sensor netowrks*, in: *Mobile Computing and Networking*, 2001, pp. 189–199.**

[16] **Sarma, S. E., S. A. Weis and D. W. Engels, *Radio frequency identification: Risks and challenges*, CryptoBytes (RSA Laboratories) 6 (2003).**

[17] **Satyanarayanan, M., *Privacy: The achilles heel of pervasive computing?*, IEEE Pervasive Computing 2 (2003), pp. 4–5.**

[18] **Stajano, F., “Security for Ubiquitous Computing,” Wiley Series in Communications Networking & Distributed Systems, John Wiley & Sons, 2002.**

[19] **Want, R. and A. Hopper, *Active badges and personal interactive computing objects*, IEEE Transactions on Consumer Electronics 38 (1992), pp. 10–20, first published as ORL technical report 92.1: ”The Active Badge Location System”.**

A Performance evaluation

Before defining an architecture that fulfills our requirements, we have to study which techniques are feasible on the described devices. The performance data was obtained on an Athlon 1,8 GHz PC, on a Fujitsu-Siemens Pocket Loox (with a 400 MHz XScale ARM processor and 64 MB RAM) and on a Compaq Ipaq 3870 (with a 206 MHz StrongARM processor and 64 MB RAM) with a small test program utilizing the freely available BouncyCastle Java cryptography library[5]. Table A.1 should only give an overview as the current implementation is not optimized for performance. Additionally, these values also include some processor time for console output, which is slow on Pocket-PCs (without performance output, the test program should be faster). On the PC, 5 test runs were done to obtain the average values while on the PocketPCs 10 test runs were performed. All values are in milliseconds (ms).

[5] The library, including API documentation, can be downloaded freely from http://www.bouncycastle.org/

Symmetric encryption and decryption as well as digest generation were performed on a typical XML message, which had a size of 1768 Byte. Asymmetric signatures and RSA encryption were performed on the 256 Bit digest generated from that message. Key lengths were 128 Bit for symmetric and 1024 Bit respectively 163 Bit for asymmetric encryption. These tests should reflect the typical operations. For EC, we could only test the ECDSA (signature generation and verification) part, because there seems to be no implementation of EC ElGamal available at the moment. We will implement EC ElGamal with BouncyCastly and make it publically available as the SmartInteraction project progresses. The higher variance in the PC test runs can be explained by concurrently running programs.

As can be seen from table A.1, high security encryption and signatures are possible on current PDAs – a typical message can be encrypted with AES/RSA and signed in less than 1300 ms and decrypted and verified in less than 1200 ms. The high values for generation of RSA key pairs do not influence the intended security architecture because keys would be generated on more powerful external systems and transferred to the mobile devices in a bootstrap phase. However, on the PocketPC platform, EC operations (signature generation and verification) take significantly longer than their RSA counterparts, most probably because of a weak floating point unit; on the PC, EC signature generation and verification have a run-time comparable to RSA. Currently, we will use RSA as asymmetric algorithm whenever possible and only employ EC when key size is the limiting factor.

	PC mean	PC std. dev.	Loox mean	Loox std. dev.	Ipaq mean	Ipaq std. dev.
AES: init. for encr.	3	0,00	172	60,48	200,5	70,73
AES: encryption	20	15,53	402,7	115,21	479,6	86,50
AES: init. for decr.	2,2	0,98	72,3	39,03	167,9	97,12
AES: decryption	24,2	16,17	493,3	151,47	373,9	116,43
RC6: init. for encr.	3,8	3,60	90	36,52	218,4	146,83
RC6: encryption	6,6	0,80	215,8	87,14	447,5	163,96
RC6: init. for decr.	2,4	0,80	48,2	8,85	161,4	48,42
RC6: decryption	13,8	16,12	258,8	68,48	377,1	62,92
SHA256: digest	16,4	4,84	306,9	101,76	770,3	507,15
RSA: param. gen.	2,4	0,49	85,3	123,31	45,5	49,47
RSA: keypair gen.	3057,8	2375,87	10328,5	5043,77	15886,8	7164,11
RSA: signature gen.	73,8	16,23	360,2	19,84	421,6	115,54
RSA: signature verify	2,2	0,40	142,5	27,49	187,8	71,58
RSA: init. for encr.	4,4	4,32	4,8	1,17	3,5	0,50
RSA: encryption	6	8,00	34,6	4,80	151,8	39,21
RSA: init. for decr.	2,6	0,80	4,6	0,49	392,6	85,01
RSA: decryption	48	16,98	123,5	2,46	254,3	30,51
EC param. gen.	9,4	4,59	748,6	154,57	514	60,37
EC keypair gen.	61,2	22,48	9194,2	453,19	5998,2	492,87
EC signature gen.	35,8	8,61	8959,3	444,01	6622	378,30
EC signature verify	51,6	4,22	17138	769,20	11389,6	667,37

Table A.1
Performance measurements

12 Towards an Open Source Toolkit for Ubiquitous Device Authentication

Authors: Rene Mayrhofer

Published in the proceedings of PerSec 2007: the 4th IEEE International Workshop on Pervasive Computing and Communication Security, part of PerCom 2007, the 5th IEEE International Conference on Pervasive Computing and Communications, pp. 247–252, IEEE Computer Science Press
ISBN 0-7695-2788-4
March 2007

Towards an Open Source Toolkit for Ubiquitous Device Authentication

Rene Mayrhofer
Lancaster University
Computing Department
Infolab21, Lancaster LA1 4WA, UK
rene@comp.lancs.ac.uk

Abstract

Most authentication protocols designed for ubiquitous computing environments try to solve the problem of intuitive, scalable, secure authentication of wireless communication. Due to the diversity of requirements, protocols tend to be implemented within specific research prototypes and can not be used easily in other applications. We propose to develop a common toolkit for ubiquitous device authentication to foster wide usability of research results. This paper outlines design goals and presents a first, freely available implementation.

1. Introduction

Needs for authentication in ubiquitous computing environments are as diverse as the applications. They may need to authenticate users or other devices, either after an identification within some system-wide naming scheme, pseudonomously, or sometimes even anonymously, with the communication partners being identified by no more than their network addresses. Anticipating the growing importance of personal devices like mobile phones in interacting with the environment, we focus on *device-to-device* communication during spontaneous interaction. This kind of authentication, either with identified subjects or anonymously, provides a good compromise between security, scalability, and privacy. A user can authenticate to her/his personal device the moment it is activated, e.g. with passwords or biometric schemes, and then use this trusted device to interact with others. As such ad-hoc encounters are expected to be numerous throughout the day, authentication needs to be handled in an efficient manner, and be as unobtrusive as possible.

Context based authentication, that is, authentication based on certain properties of the user or device context, promises efficient, secure, and most importantly intuitive authentication methods (see e.g. [4, 5, 6, 9, 10, 11, 12, 13, 14]). The last point is especially important, because security measures are frequently disabled if they are not usable enough but get in the way of user's daily jobs. One example for using context is authentication based on spatial reference: by using a positioning system, devices can measure their spatial positions relative to each other. A visualised map can be used select devices to interact with. This is intuitive, and users can directly verify the spatial relationship, in contrast to purely wireless communication. We can construct secure device-to-device authentication out of this intuitive user interaction by coupling wireless communication with spatial sensing, using well-known cryptographic primitives.

However, context based authentication is a new research topic, and implementations of such protocols thus tend to be very application-specific. Currently, authenticating wireless communication in ubiquitous computing applications needs to be designed and implemented for each application. More often than not, it is therefore left out of research prototypes, to be "added later", because security is hardly the research focus of many of these projects. But practical experience shows, and handbooks insistently suggest [7], that security needs to be a requirement from the start; fitting it later on top of an existing system does not work in most instances. Re-usable hard- and software components are required that can be treated as black boxes from the developer's point of view to make it easier for applications to benefit from the advantages of context based authentication.

In this paper, we define our design goals for a ubiquitous authentication software toolkit and present a first implementation. The main contribution of this toolkit is the combination of cryptographic protocols with sensor data to create context based authentication. It provides lower-level primitives and higher-level context authentication protocols in the same way as the OpenSSL toolkit [1] provides cryptographic algorithms and an SSL/TLS implementation. Our approach is to rely on simple, standardised, off-the-shelf and therefore cheap sensors and provide software components to use them for authentication purposes.

This toolkit is part of ongoing research and will be extended to include new developments. Three projects for device-to-device authentication with different sensors already make use of it, and we expect more projects to follow.

Section 2 analyses requirements for such a toolkit, followed by a brief comparison of different software platforms for its implementation in section 3. Section 4 gives an overview of the current implementation and its availability, and in section 5 we provide a brief overview of projects making use of it.

2. Design goals

The main functional requirement for the authentication toolkit is to provide methods for creating shared secrets between two (or multiple) devices. These secrets should be authenticated to prevent man-in-the-middle (MITM) attacks. Authentication between personal devices and the environment is difficult, because such devices are small, mobile, and typically have limited resources. The absence of large screens and efficient input devices makes authentication based on sensor information even more attractive. Different applications require different sensor modalities for interaction as well as different levels of security. Therefore, a *toolkit*, i.e. a collection of loosely coupled components, is better suited to fulfil those different needs than a *framework* that implements the complete program flow and offers only defined hooks for application behaviour. For adding authentication to applications, it is easier to select and combine provided components than to make the application fit a pre-defined structure. Following this general design choice, we identify more detailed non-functional requirements. The toolkit should be:

- *lightweight*: Resources on mobile, battery-powered devices are generally sparse. This includes storage and run-time memory, CPU, communication bandwidth, but also battery lifetime, input/output devices, and user attention. A toolkit should be as small as reasonably possible, use static memory buffers when possible, and minimise communication. These aims are conflicting, and when no generally acceptable compromise can be found in some case, the respective components should be parameterisable for application developers.

- *self-contained*: Devices and platforms where the toolkit might be used are expected to be extremely diverse. Therefore, we can not depend on specific libraries to be available. Any dependencies that are not included in the default platforms should be included in the toolkit.

- *simple to use*: An authentication toolkit is most useful if it can be used without great care on the side of application developers. This has two reasons: if it is too complex to learn, developers will not use it for simple applications, and if it is complex to use, it is likely that it will be used erroneously and thus insecurely. Ideally, the various components of the toolkit can be used as black boxes with simple interfaces, and can be combined with each other and with application-specific hooks to build secure context authentication protocols without knowing about the internals. We therefore explicitly minimise the number of exported options, and focus on reasonable default values and automatic parameterisation whenever possible.

- *extensible*: It is obvious that a toolkit should be easily extensible by additional components. When designing it as a collection of related and compatible, but separable components, this goal should be automatically fulfilled, in contrast to framework-type design structures where extensibility must be explicitly considered.

- *vertical*: As context based authentication concerns all layers from sensing hardware, input/output devices, networking, application context, up to user interaction, a toolkit should provide components that span the layers. High-level components that relate to complete use-cases can make use of primitives from various lower-level layers.

- *interoperable*: Ubiquitous computing environments are inherently heterogeneous. Authentication protocols therefore need to be interoperable between different platforms. Thus, network communication should either be based on standardised protocols (e.g. IETF RFCs) or use simple ASCII line based protocols in the spirit of SMTP, HTTP, and others.

Implementation issues are mainly that the toolkit must be secure and that it needs to work asynchronously. Making a system secure is hard to achieve in the general case, because the security of a system depends on all of its components. The weakest parts will most likely be outside the toolkit. Nonetheless, the toolkit itself should be written carefully to protect against known and mitigate future attacks. This includes systematic protection against overflow attacks, pre- and post-condition checks for all methods, sanity checks for internal consistency, and wiping cryptographic key material from memory as soon as it is no longer required (see e.g. [7] for a more detailed introduction into the topic of secure programming). Especially the last point can be tricky to implement with run-time platforms like a JVM (Java virtual machine) or a .NET CLR (common language runtime). In addition to these standard best practises, “defensive” programming techniques demand checking every input value syntactically, semantically, and for the context/state in which it is read. This includes validating received network packets and direct user input, but, in contrast to typical desktop applications, also sensor values, which might be tampered with as well.

The second issue is to deal with asynchronous program flows. An event based structure has two advantages over the standard blocking procedure calls: First, authentication methods are likely to cause noticeable delays while engaging in wireless communication or waiting for user or sensor input. Instead of forcing all applications to deal with this issue, it is simpler to provide asynchronous callbacks for all actions that delay a program flow. Second, reacting to events and switching hardware components to standby or sleeping modes in between can be used as an effective way of saving battery power (see e.g. the design of TinyOS [3]).

Finally, a toolkit is most useful when it is released under a license liberal enough to allow linking in all cases, both for other open source and for proprietary, closed source applications. The GNU General Public License (GPL) is problematic in this respect, because it does not allow linking with components that are not released under the same license.

3. Platform choices

As already mentioned, we expect a very heterogeneous range of devices to make use of context authentication. Consequently, no single software platform can be the base for supporting all of these systems. We briefly evaluate the most common platforms that promise to be supported on a large range of devices.

Java The Java platform, i.e. the programming language and the virtual machine (JVM), is becoming increasingly well supported by off-the-shelf devices. This includes laptop/desktop systems, PDAs, many newer mobile phones, industrial and embedded devices, and some consumer devices like upcoming Blue-ray players. Most of the resource limited devices do not provide the complete runtime library and language features, but only a subset called J2ME. From a security point of view, there are advantages and disadvantages: on the one hand, memory management embedded into the runtime makes buffer overflows harder to trigger, but at the same time it allows less control in terms of managing key material in memory.

.NET With similar goals to Java, .NET provides its own equivalent to the virtual machine (the CLR) and runtime components with memory management, but with the benefit of supporting multiple languages. It is supported on laptop/desktop systems and less commonly on PDAs and mobile phones. The .NET runtime seems to have neither particular advantages nor disadvantages over Java in terms of security.

C++ This is not a platform in the same sense as Java and .NET, but libraries like Boost allow cross-platform development. The main use of C++ in our target range of devices is Symbian OS, which is currently used on the majority of reasonably powerful mobile phones. C++ allows in-depth control of memory management and supports automatic wiping of key material by destructors and stack unwinding. It is also regarded as more lightweight than both Java and .NET, especially in terms of run-time memory consumption. However, experience shows that buffer overflows are a major issue with C and C++.

TinyOS Finally, many sensor nodes used in research projects run TinyOS [3] as their platform. It uses a dialect of C as its language, and compiles the whole firmware into one statically linked binary. Owing to the restricted CPUs that most sensor nodes use, TinyOS does not support dynamic memory management, and is thus expected to have slightly fewer vectors for buffer overflow attacks. On the other hand, cryptographic operations are expensive in terms of CPU consumption and battery lifetime, so programming on TinyOS poses different challenges than the other platforms.

The toolkit as a collection of algorithms is necessarily dependent on the target platform. Ideally, the toolkit should be ported to all above platforms, because all are widely used for devices we envisage to make use of context authentication. Authentication protocols should be interoperable between the different platforms, even if implementations differ. The key point is that the components should look the same to application developers, abstracting from possibly significant platform differences.

At the moment, we concentrate on Java with J2ME compatibility to support many mobile platforms in the first release. A port to .NET should include the complete functionality, while implementations for Symbian OS and TinyOS might use only selected components due to resource constraints.

4. Current implementation

Our current implementation contains components on the layers of cryptographic primitives, key agreement and authentication protocols, secure channels, and dealing with sensor data. Fig. 1 gives an overview of the dependencies between the components in different layers. These layers include the following specific components at the time of this writing:

Cryptographic primitives Implementations of ciphers, secure hashes, etc. are widely available, even as part

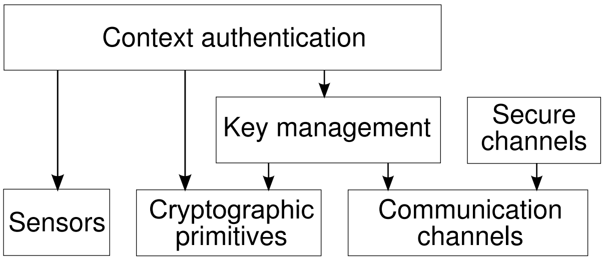


Figure 1. Interactions between components of different layers of the toolkit

of newer Java 2 runtimes (generally starting with version 1.4). However, this so-called Java cryptography extension (JCE) has not yet been included in the J2ME standard that is supported by most Java implementations on small, mobile devices. To fulfil the goal of being self-contained, the toolkit therefore uses wrappers around the JCE algorithms where necessary, and includes alternative implementations from the Bouncycastle cryptographic library [2]. When JCE is not supported on a target platform, the toolkit can use these as a fallback, albeit typically with slightly worse performance due to missing native implementations. Classes from Bouncycastle are also used to augment JCE primitives where they lack higher-level support.

On top of these primitives, we add small wrapper and utility classes that make the underlying algorithms as simple to use as possible. One notable example is a class to create X.509 certificates on-the-fly, for the purpose of using standard protocols like TLS.

Communication channels Java already offers good support for working with TCP or UDP connections. The toolkit again adds utility classes for simplifying setup and use of these protocols for standard cases, e.g. a threaded TCP server that listens in the background and starts key agreement protocols upon connection, or UDP multicast sockets for point-to-multipoint authentication protocols. These classes generally take care of low level details like binding to every network interface that has been found on the device without showing this complexity, to fulfil the goal of being simple to use.

Key management protocols This layer uses cryptographic primitives and communication channels to provide simple key agreement protocols. One example is a standard, unauthenticated Diffie-Hellman (DH) key agreement over TCP connections with an ASCII based protocol. Another protocol is just being created, and will interactively create cryptographic key material from sensor data streams, with either UDP multicast or Bluetooth communication. Additional candidates for this layer that have recently been proposed are the MANA family of protocols [8], a variant proposed by Wong and Stajano [20], and SAS [18]. Key management protocols for multi-party settings like the one proposed by Wacker et al. [19] and trust delegation protocols like the token based approach by Steffen and Knorr [17] also belong to this layer of components. We expect some of these and other protocols to be added to the toolkit in the future, with a common basic structure to make them simple to combine or exchange.

Sensors and feature extractors Dealing with sensor data is an important part for context authentication; this includes interfacing to the hardware sensors for data acquisition, handling of time series, and extracting appropriate features. The toolkit focuses on ease of use and automatic parametrisation, but exposes the parameters to applications when reasonable defaults can not be set. Currently, we provide base classes for reading data from ASCII based sources, simple Bluetooth RFCOMM channel access via the JSR82 API, computing time series statistics, time series aggregation, and segmenting time series based on a simple activity detection. An FFT implementation and a quantizer support feature extraction. All of these classes are optimised for real-time processing on resource limited devices.

Context authentication protocols Components on this layer tie together key agreement and authentication based on sensor data to create context authentication protocols. The result of a successful execution of one of these protocols is an authenticated secret shared key that can be used by applications. Protocols for authentication based on spatial reference and on common motion patterns are already available in the toolkit.

Secure channels The last layer implements secure communication channels, preferably based on standard protocols. These protocols generally depend on either trusted third parties, which can not be realistically assumed for ubiquitous computing environments, or shared secrets, as e.g. generated by components of the context authentication protocols layer. Currently, the toolkit provides wrappers for managing IPSec channels with operating system support. This has been implemented for Linux (with FreeS/WAN, Openswan, strongSwan, or racoon), Windows 2000/XP, and Mac OS X (with racoon), either using X.509 certificates or PSKs (pre shared keys). We also intend to provide a secure channel implementation based on TLS-PSK as an alternative to IPSec.

All key agreement and authentication protocols are event based and executed in the background. Three types of events can be generated: success (with the resulting shared secret key embedded into the event), failure (with a message giving reason for the failure), and progress (optionally with indication of how many steps have been finished and how many are left). These events can be used to provide user feedback in applications.

The Log4j framework is used for run-time configurable logging and JUnit for an extensive set of unit tests. Test cases cover single components as well as combinations spanning multiple layers, and special tests including real-world data samples for the context authentication protocols. Additional utility classes are used for defensive and fail-safe programming, like a ”safety belt” timer used to terminate authentication protocols after timeouts.

Developers can use components simply by adding the single JAR file (or only the required components if program size is an issue) and using the provided classes. Applications only need to implement a single interface to process standard authentication events as described above. Although applications are free to use components from all layers, the two top layers are expected to be the most useful: *context authentication protocols* provide secret, authenticated key material which can then be used by *secure channels* components for securing the actual communication between devices.

Extending the toolkit with new components is similarly simple: there is no central structure that needs to be followed for every part, nor are there main interfaces that must be implemented. The toolkit provides some basic infrastructure, and extensions are free to use it. Its design loosely follows the principle of UNIX command line tools: to combine components with simple interfaces into more complex parts. When no suitable context authentication protocol is available for a specific applications, then the more basic layers should help in constructing it, ideally also adding it as a new component to the open source toolkit.

5. Initial projects using the toolkit

We currently use the toolkit in three applications that make use of context authentication:

- Our first application authenticates WLAN clients by spatial reference to set up IPSec channels [14]. The protocol for authenticating relative device positions with ultrasonic pulses [16] was the first complete context authentication protocol to be implemented within the toolkit. Additionally, this application creates X.509 certificates on the fly and configures the operating system IPSec support by using components of the lower layers of the toolkit.

- The second application authenticates devices based on common movement by comparing accelerometer time series [15]. It uses the same building blocks of the key management layer as the first application, namely a standard DH key agreement over TCP with an interlock protocol to prevent MITM attacks while exchanging the time series. This application triggered improvements in the sensor layer, as it requires real-time computation of features on the accelerometer data streams in time and frequency domain. As a more efficient alternative to the DH/interlock protocol combination, we are currently working on a novel protocol to create key material from sensor data streams using only symmetric cryptographic primitives.

- In the third project, visible light pulses are used to transmit authentic messages between devices with direct line of sight. This application again uses the DH and interlock primitives, but due to one-way “transmission” over this out-of-band channel, we are currently designing an alternative combination of these key management components. This is easily possible due to the loose coupling of the components.

These applications are mostly concerned with the “user interface” parts, while the protocols, hardware access, and internal management functions are provided by the toolkit. The first application is finished and available as an example with the current release, and new components developed for the other two applications will be added after they have received sufficient testing.

All components used in the first application (with the exception of the operating system IPSec channels, which rely on native libraries) run and are tested on an Asus MyPAL PocketPC with an IBM J9 JVM in addition to the standard desktop JVMs. In fact, the application scenario explicitly includes PDAs as devices that participate in authentication by spatial reference. First experiments show that most primitives and protocols also run on mobile phones with J2ME implementations. One of the current challenges is to consistently support sensor data access and wireless communication on different mobile phone platforms in the respective layers.

Our current experiences show that implementation of the cryptographic protocols was among the easier parts and, implemented for one application, it was in practise simple to reuse for others. It was more difficult and time consuming to interface with and use sensor data, e.g. to find appropriate feature extraction algorithms. This shows an important difference to the typical (and better understood) usage of sensor data: for context authentication, we do not need good separation between different classes, but high entropy from an attacker’s point of view. Therefore, we expect the toolkit to be of particular value in this area, as well as in upper layers that tie together sensor data with cryptographic protocols.

6. Conclusions and future outlook

The toolkit is a collection of cryptographic primitives, key management protocols, wrappers for dealing with sensor data, and high-level context authentication protocols. Because of its design, it is easy to use and extend. Work on this toolkit has been ongoing for over a year. Some of its components have initially been developed for specific applications that make use of context authentication. After splitting them out and generalising them, they were integrated into this toolkit. The version available at the time of this writing is an alpha release that lays groundwork for implementing a growing set of protocols. Its lightweight, event based structure has proven useful in the projects that triggered the development of this toolkit as well as in further research projects currently under development.

Our choice of Java as the first implementation platform and a liberal open source license should allow the toolkit to be used on a large range of devices. We invite researchers working on security in ubiquitous computing to contribute their proposals to a common collection, so that application developers can easily evaluate and use them.

A first alpha release of the toolkit is available at `http://www.openuat.org` under the terms of the GNU Lesser General Public License (LGPL). This allows linking with proprietary and closed source applications, but guarantees that changes and additions to the toolkit itself will remain open source.

7. Acknowledgements

We gratefully acknowledge support by the Commission of the European Union under contract 013790 "RELATE" and the FP6 Marie Curie Intra-European Fellowship program contract MEIF-CT-2006-042194 "CAPER", and by the Engineering and Physical Sciences Research Council in the UK under grant GR/S77097/01. We thank Gerd Kortuem and two anonymous reviewers for comments on an earlier draft.

References

[1] OpenSSL: The Open Source toolkit for SSL/TLS web page. `http://www.openssl.org`, 2006.

[2] The Legion of the Bouncy Castle web page. `http://www.bouncycastle.org`, 2006.

[3] TinyOS web page. `http://www.tinyos.net`, 2006.

[4] D. Balfanz, G. Durfee, R. E. Grinter, D. K. Smetters, and P. Stewart. Network-in-a-box: How to set up a secure wireless network in under a minute. In *Proc. 13th USENIX Security Symposium*, pages 207–222. USENIX, August 2004.

[5] J. E. Bardram, R. E. Kjær, and M. Ø. Pedersen. Context-aware user authentication - supporting proximity-based login in pervasive computing. In *Proc. UbiComp 2003: 5th Int. Conf.*, pages 107–123. Springer, October 2003.

[6] S. Creese, M. Goldsmith, R. Harrison, B. Roscoe, P. Whittaker, and I. Zakiuddin. Exploiting empirical engagement in authenticated protocol design. In *Proc. SPC 2005: 2nd Int. Conf. on Security in Pervasive Computing*, pages 119–133. Springer, April 2005.

[7] N. Ferguson and B. Schneier. *Practical Cryptography*. Wiley Publishing, 2003.

[8] C. Gehrmann, C. J. Mitchell, and K. Nyberg. Manual authentication for wireless devices. *RSA Cryptobytes*, 7(1):29–37, 2004.

[9] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: Human verifiable authentication based on audio. In *Proc. ICDCS 2006: 26th Conf. on Distributed Computing Systems*, page 10. IEEE, July 2006.

[10] M. K. R. Jonathan M. McCune, Adrian Perrig. Seeing-is-believing: Using camera phones for human verifiable authentication. Technical report, CMU, November 2004.

[11] T. Kindberg and K. Zhang. Validating and securing spontaneous associations between wireless devices. In *Proc. ISC'03: 6th Information Security Conf.*, pages 44–53. Springer, October 2003.

[12] T. Kindberg, K. Zhang, and S. H. Im. Evidently secure device associations. Technical Report HPL-2005-40, HP Laboratories Bristol, March 2005.

[13] T. Kindberg, K. Zhang, and N. Shankar. Context authentication using constrained channels. In *Proc. WMCSA: 4th IEEE Workshop on Mobile Computing Systems and Applications*, pages 14–21. IEEE Computer Society, June 2002.

[14] R. Mayrhofer. A context authentication proxy for IPSec using spatial reference. In *Proc. TwUC 2006: 1st Int. Workshop on Trustworthy Ubiquitous Computing*, pages 449–462. Austrian Computer Society (OCG), December 2006.

[15] R. Mayrhofer and H. Gellersen. Shake well before use: Authentication based on accelerometer data. accepted for publication at Pervasive 2007, *to appear*.

[16] R. Mayrhofer, H. Gellersen, and M. Hazas. An authentication protocol using ultrasonic ranging. Technical Report COMP-002-2006, Lancaster University, October 2006.

[17] R. Steffen and R. Knorr. A trust based delegation system for managing access control. In *Advances in Pervasive Computing: Adjunct Proc. Pervasive 2005*, volume 191, pages 1–5. Austrian Computer Society (OCG), April 2005.

[18] S. Vaudenay. Secure communications over insecure channels based on short authenticated strings. In *Advances in Cryptology - Proc. CRYPTO 2005: 25th Int. Cryptology Conf.* Springer, August 2005.

[19] A. Wacker, T. Heiber, H. Cermann, and P. Marron. A fault-tolerant key-distribution scheme for securing wireless adhoc networks. In *Proc. Pervasive 2004: 2nd Int. Conf. on Pervasive Computing*, pages 194–212. Springer, April 2004.

[20] F.-L. Wong and F. Stajano. Multi-channel protocols. In *Proc. Security Protocols Workshop 2005*. Springer, 2006.