Infrastructure for Trusted Content

Editors:	Dr. Stephan Spitz (GD)
	Dr. Walter Hinz (GD)
	Dr. Marc-Michael Bergfeld (GD)
Reviewers:	Karel Wouters (KUL)
	Aleksandra Kuczerawy (KUL)
Identifier:	D6.2.1
Type:	Deliverable
Class:	Public
Date:	28 August 2008

Abstract

As the usage of information and communication technology increases in private and professional life, personal data is widely stored. While service providers need to rely on identifying their customers, conscious identity management and privacy develops into a new value for the service user, especially in the electronic service context. Here, modern technology infrastructures can assist in providing security to both sides by assuring identification and privacy at the same time.

This paper introduces the 'Security of Service' concept and presents different technical environments through which it can be established. Privacy in service composition is elaborated conceptually and from a technological standpoint. Potential authentication scenarios and emerging use cases are introduced. In conclusion, implications for the needed technological infrastructures are derived and necessary future research directions are indicated.

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 216483.

SEVENTH FRAMEWORK PROGRAMME

Members of the PrimeLife Consortium

1.	IBM Research GmbH	IBM	Switzerland
2.	Unabhängiges Landeszentrum für Datenschutz	ULD	Germany
3.	Technische Universität Dresden	TUD	Germany
4.	Karlstads Universitet	KAU	Sweden
5.	Università degli Studi di Milano	UNIMI	Italy
6.	Johann Wolfgang Goethe – Universität Frankfurt am Main	GUF	Germany
7.	Stichting Katholieke Universiteit Brabant	TILT	Netherlands
8.	GEIE ERCIM	W3C	France
9.	Katholieke Universiteit Leuven	K.U.Leuven	Belgium
10.	Università degli Studi di Bergamo	UNIBG	Italy
11.	Giesecke & Devrient GmbH	GD	Germany
12.	Center for Usability Research & Engineering	CURE	Austria
13.	Europäisches Microsoft Innovations Center GmbH	EMIC	Germany
14.	SAP AG	SAP	Germany
15.	Brown University	UBR	USA

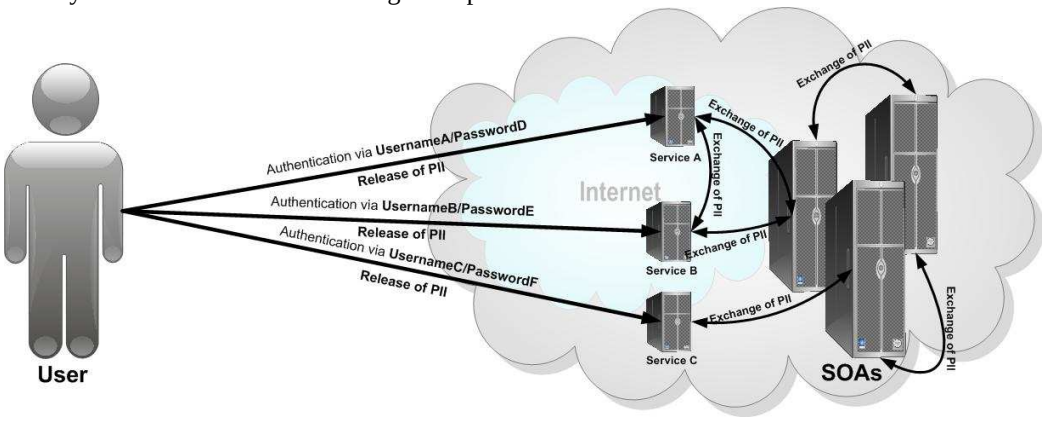
List of Contributors

Researchers from multiple PrimeLife partners have contributed to this deliverable. The following list represents the contributions per chapter.

Chapter	Author(s)
Executive Summary	Stephan Spitz (GD), Marc-Michael Bergfeld (GD)
Introduction	Stephan Spitz (GD), Walter Hinz (GD), Marc-Michael Bergfeld (GD)
Variants of Secure Environments	Stephan Spitz (GD), Jan Zibuschka (GUF), Thorsten Urhahn (GD), Hans Borgs (GD), Marc-Michael Bergfeld (GD)
Privacy in Service Composition	Laurent Bussard (EMIC), Anna Nano (EMIC), Uli Pinsdorf (EMIC), Stuart Short (SAP), Slim Trabelsi (SAP)
Potential Authentication Scenarios and Emerging Use Cases	Lars Janssen (GUF), Stuart Short (SAP), Slim Trabelsi (SAP), Laurent Bussard (EMIC), Anna Nano (EMIC), Uli Pinsdorf (EMIC)
Outlook	Jan Schallaböck (ULD)

Executive Summary

Mobile devices have developed into omnipresent companions of modern life. They fuel an increasingly IT-based and virtual mode of human interaction in the business and private world as they enable services such as payment, access and multimedia-based communication. They are becoming important enablers in the identification of individuals. Further, internet-based services are increasingly provided through mobile devices. Hence, the security and privacy issues that already exist in the world wide web gain importance for the mobile world too.



In this context, the two parties of a service (i.e. the provider on the one side and the customer on the other) increasingly perceive service security as a crucial selling proposition. At the same time, service quality is becoming a non-differentiating basic requirement.

From a service provider perspective, the security of a service is dependent upon the identification of the respective individual in the specific service encounter. From a customer perspective, security roots in the minimum possible transmission of the specifically needed personal data and the highest possible level of privacy for the data that is transferred.

The challenge of secure service provision is intensified by the trend that customers tend to present themselves under different identities when interacting with various service providers. This calls for a process which identifies the individual user in the moment of a service request, secures his privacy herein, and enables him to flexibly adapt various identity profiles.

The trend towards more flexible and customized services, for example in mash-up approaches and service structures that span across trust-domains and applications, impacts the identification of individuals. Such composite services span client and service side applications and require new mechanisms for data handling and trust evaluation.

The here presented Work Package (i.e. WP 6.2.1) strives to explain which infrastructure may be applied to establish an environment of secure services. In this environment, flexible identification shall be possible whilst privacy is assured.

Particular attention is given to web-related technologies and trusted computing (i.e. Trusted Platform Modules (TPMs)) as future infrastructures as well as Smart Cards (including GSM/UMTS/SIM and signature cards) and PKI solutions as present infrastructures.

Hence, a technological infrastructure is defined in which trusted mobile devices can be applied to enable secure services. As a result, presenting and authenticating various identities in service encounters will be possible while highest privacy can still be guaranteed.

Table of Contents

List of Figures

Chapter 1

Introduction: Identity Management, Mobile Devices and the Internet

1.1 Overview of the Total System

The definition of a scalable privacy model in conjunction with different levels of security is a cornerstone of identity management. It is common to use different identities or roles when acting as, for instance, an employee, a customer, or a parent. Extreme cases can even be observed in a tribe in New Guinea where people change their name (identity) depending on whom they are talking to. Thus, each person has several identities. This behaviour is even more common in the electronic and virtual world: For example, our bank has different information about us than a mobile operator, a business partner or the government.

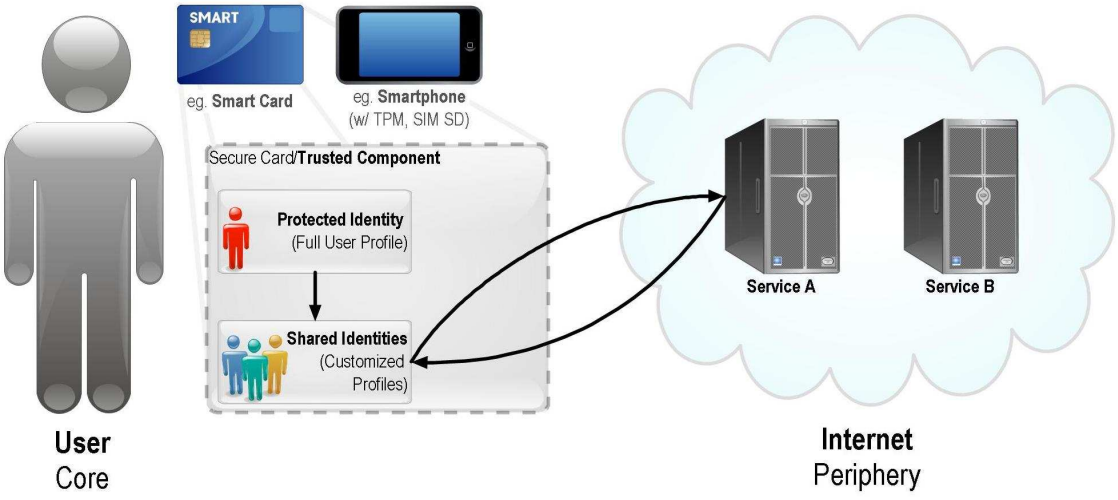


Figure 1: Managing different identities from one protected core data set

Hence, the bank's perception of our identity is different to the way we are seen by other institutions. This is because different privacy settings are applied. Each setting or level of privacy corresponds to a different

data subset. The various data bundles provide different sets of knowledge and presentation schemes of our selves. Therefore, there is a direct connection between security levels and identity.[1]
In consequence, managing your identity means managing various privacy settings with different security features as enablers. From a technical standpoint, the overall concept of identity management in the wider service context can be established by using a trusted component which provides different levels of privacy and identity information to its surrounding context. In sum, this establishes a secure service environment.
As sketched in Figure 1, the secure client / trusted component will represent the core entity and user-perspective of a secure service. The internet will serve as service periphery. Taken together, these two perspectives will establish an environment in which secure services can be provided when the user interacts with the internet. Typical secure environment functionalities required in conjunction with identity management are:

- All kinds of authentication mechanisms (mutual/external/internal)
- Security policy enforcement (as far as possible in a client component)
- Secure Communication Endpoint, e.g. SSL-Endpoint
- Secure Display and Keypad (only in high-end devices)

Overall, the PrimeLife project addresses a wide variety of architectures for identity management. The here presented deliverable focuses on managing different levels of privacy through the support of a mobile device.
Using identity management in combination with such a mobile device in order to establish a secure service environment faces three different challenges:

1. Challenge from the user perspective: During the past years, mobile devices have become ubiquitous companions and IT services are providing technologies that match information with a user's interests and annotate usage profiles. This optimization of annotations and profiles in connection with security make a portable secure device a suitable participant of a semantic secure environment. Mobile devices are no longer mere enabling tools for decentralised service provision. Today, the devices themselves increasingly play important roles in establishing secure service environments and enabling a participation in these. In this context, privacy for the individual user is especially ensured only if the respectively needed parts of the whole user-related data (profile) are transferred from the personal mobile device. This assures continued privacy whilst providing a clear identification and authentication of the user for the service provider.

2. Challenge from the security perspective: At present, systems and technology architectures are still complex because many technology modules are developed by different companies and no dominant design has been established in the market of privacy which may use security modules such as Smart Cards. Hence, it remains a reality that most of the components follow their own security concepts.
In such a fragmented setting, achieving coherence of security (i.e. a solid and seamless security that stays pervasive through the whole architecture) continues to be a big challenge.
The here presented deliverable addresses this challenge in chapter 3 ("Privacy in Service Composition").

3. Challenge from the infrastructural perspective: To work properly, a secure service may depend upon the cooperation of different companies and/or the integration of different sub-services. To achieve this, each service provider first needs to identify what kind of subscription is asking for which service.[2] To do so, each provider initially requires a genuine identity of the individual user. This genuine identity is presented before the service is provided. In a generally fragmented market and in cross-industry service provision, each service provider can be expected to have his own mechanism of matching identities to services. Hence, several ways to validate and manage identities can be expected.
The infrastructural challenge is to integrate different identity systems and the way how the subscription/identity can be recognized in and through the infrastructure composed by different service

[1] According to the International Organization for Standardization (ISO), security is defined as the minimization of "the vulnerability of the value and other resources" [106].
[2] Subscription refers to a legally binding interaction between the service provider and the service user. In a subscription, the service user requests the provision of a pre-defined set of services for a set timeframe. If service provision is sub-contracted to another service provider, this 3rd party will have to initally check whether the actual subscription covers the requested service.

providers. Integrating this infrastructure perspective is closely linked to solving the above mentioned security challenge.
Therefore, this challenge is also addressed in chapter 3 ("Privacy in Service Composition").

1.2 Security Considerations

In the services industry – and particularly in services which are based on electronic devices – a new paradigm seems to be emerging: The conscious management of identity in a secure service context. Quality of Service (QoS) has already become an indispensable attribute as a foundation to this understanding. Hence, QoS is already defined in the deployment of any service architecture.
Treating Security of Service (SoS) as a similarly crucial attribute of a service calls for an exploration and definition of the term.
In the PrimeLife project, the SoS concept will first be conceptualised from a general point of view and then adapted to the whole service architecture in terms of:

- Size
- Mobility
- Complexity
- Distribution of knowledge in the system, of decision points in the software, and in the services provided.

It is obvious that security needs a kind of trust anchor. Such a security anchor enables the user:

- To define a secure domain, which also is context-dependent (home/family, friends, work, travel).
- To manage the user's personal environment so that the management of the user's device base is both secure and easy.
- To define which of his devices can be publicly or restrictedly accessed and how interactions occur.

A secure client in combination with a trusted component in a mobile device can serve as a security anchor in the overall SoS concept and system architecture.

1.3 Environments for the 'Security of Service' Concept

The Security of Service (SoS) concept can be expected to develop along three phases. These phases are not only important to anticipate the characteristics of future security requirements, but can also assist to pin-point the definition of the concept alongside its anticipated path of development. Therefore, defining the SoS-concept could be accomplished in 3 steps.[3] These match the respectively needed capability tiers:

Step 1. **Security requirements in a static environment:** This will consider fixed and concrete client and server components, actors and scenarios. The definition of SoS will result in fixed *security requirements* for the given scenarios. At this level, SoS behaviour can be developed and modified in the client and server component.

Step 2. **Security rules in a dynamic environment:** This will consider the heterogeneity of scenarios. The definition of SoS will result in *security rules*. The equipment, context-aware, will know rules of behaviour under different situations. The SoS will be changed in a situation-specific manner according to the rules defined during the pre-development. Hence, the SoS is dynamic and co-evolves with the isotropic and steadily changing context into which it is embedded. This approach is similar to the way in which Europay, Mastercard, and Visa (EMV[4]) consider security in different and continuously changing scenarios.

Step 3. **Security policies in an adaptive environment:** This will consider unknown equipment, actors, and heterogeneity of space. The definition of SoS will result in *security policies*. The client and the server will know the policies. Both will consider whether a given service is to be continued or blocked for a dedicated actor in a specific situation. In an adaptive environment, QoS and SoS

[3] The here presented steps of defining an SoS-concept have been developed for application in future workpackages, when the various scenarios become more detailed and preferred use cases are elaborated more profoundly in conceptual and technical terms.
[4] Note: EMV is a standard for interoperation of IC chip cards and IC capable Point of Service terminals and Automatic Teller Machines for authenticating credit and debit card payments.

take a “flexible and secure”, “pervasive and secure”, “resilient and secure”, “recoverable and secure” character, depending on the situation. The client and the server components will deploy an adaptive SoS and QoS ad hoc by negotiating the SoS according to the agreed security policies.

Chapter 2

Variants of Secure Environments

The objective of this chapter is to sketch the different possibilities for the protection of security-critical data in conjunction with identity management infrastructures.
As a starting point, the mobile secure environment will require a root user profile that can be used to derive a profile compatible for different dedicated services. From a user perspective there are three essential requirements for these profiles:

- Integrity: No manipulation of the content (tamper-proof), e.g. by viruses
- Availability: No denial of service caused by potential attacks
- Confidentiality: Value reading (i.e. extracting the respectively needed information) and no leakage of profile information to a non-trustworthy third party

It is common practice to achieve a protection of personal data by isolation in a tamper-proof/resistant environment. The applied technologies vary from highly tamper-proof or tamper-resistant environments like Smart Cards to software-based isolation mechanisms with hardware support, e.g. virtualization or ARM Trust Zone. To enable a cross-domain composition, it is necessary to integrate such security environments in identity management infrastructures. Moreover, it is necessary to have standardized, highly accepted interfaces to security components because the identity management infrastructures are composed across different systems and domains. Therefore, it is the aim of PrimeLife to investigate in Work Package 6.2:

- Security environments to protect personal data
- The integration of security environments in identity management infrastructures
- Mechanisms for the life cycle management of secure environments and personal data storage devices.

2.1 Overview of Secure Environments

First ideas about tamper-proof and tamper-resistant cards with an embedded chip already appeared in 1970. Since then, the Smart Card market has grown amazingly. A lot of different kinds of Smart Cards currently ensure the security of authorization processes, including and thereby enabling identity management.
Since a few years, Smart Cards also appear in new form factors, e.g. with a USB connector or as a built-in device in a PC, i.e. the so-called TPM (Trusted Platform Module). Especially different applications of TPMs [38] are an example for the increasing need to ensure the integrity of data and the identity of persons within global networks.
The intention of the following diagram is to give an overview of the different varieties of secure environments. They range from software-based solutions in embedded systems to Smart Cards and tokens. It is not the intention of Figure 2 to comprehensively mention all possible kinds of secure environments and also not to be exclusive, which means that there are combinations possible. For example, a SIM can also be

a Java Card or a virtualization can also appear in conjunction with a tamper-resistant device such as a Smart Card.

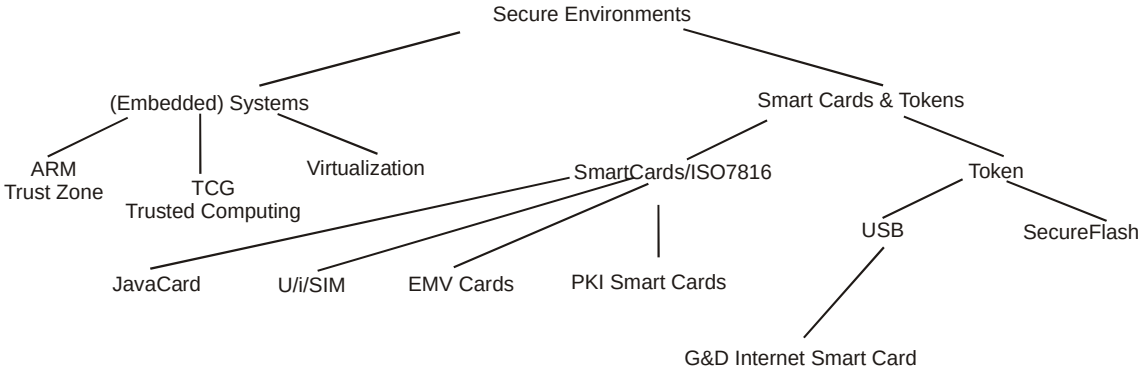


Figure 2: Overview of different secure environments

Within the here presented Work Package, focus is laid on the secure environments sketched in Figure 2. This corresponds to the target set in the project proposal.[5]

2.2 Capabilities of Smart Cards and Token Technology

The main usage of Smart Cards is the storage of highly confidential information, such as cryptographic keys, and the execution of security critical processes, such as an authentication to prove the identity of a person or device. Classical Smart Card use cases are [31], [33]:

- Authorization in communication networks, such as mobile phone networks or the Internet,
- Execution of security-critical processes with banking and payment applications, e.g. credit and debit operations on an electronic purse,
- Storage of sensitive personal information, e.g. on health and identity cards,
- Physical and logical access control.

Major Smart Card milestones in the past were the introduction of the SIM (Subscriber Identity Module) as the security device in mobile networks and the invention of Java on Smart Cards, i.e. the JavaCard™ Standard. With JavaCard™ the flexibility of Smart Cards increased, because it was the first time possible to develop Smart Card applications in an interoperable format. The so-called JavaCard™ Applets can be executed in an almost interoperable manner on different JavaCards™ from different Smart Card vendors.
With the increasing computing capabilities of µProcessor Smart Cards new opportunities appear. Newer Smart Cards, connected to a host over the USB interface, offer a full TCP/IP stack in the operating system. These Internet-Smart Cards no longer depend on a PC to be able to communicate because they can act independently as a network node in a global network like the Internet. So, they may serve as a good protection mechanism for personal data in combination with information exchange in a cross-domain network. A person can determine, which information about her/him/is published by help of the communication gateway Internet-Smart Card that supports standard Web technologies like HTML (hypertext markup language) pages and HTTP (hypertext transfer protocol). Therefore, the Internet-Smart Card ([28], [29]) hosts a Smart Card Web Server (SCWS) which acts as graphical user interface for the personal token. The Internet-Smart Card technology and SCWS also appear in the future USIMs[6] in mobile networks. On the user side, a web-like look and feel simplifies information exchange with a Smart Card. For example, users browse a phone book or FAQ list based on HTML pages stored directly on the Smart Card Web Server (SCWS) hosted on the (U)SIM. On the provider side, an HTTP-based update mechanism

[5] Note: The proposal specifically calls for web-related technologies, web-service architectures, trusted computing, GSM/UMTS-SIM, signature cards and citizen cards, PKI, and large portal accounts.
[6] Note: A Universal Subscriber Identity Module is an application for UMTS mobile telephony running on a Smart Card which is inserted into a 3G mobile phone. The USIM is not the Smart Card itself but merely a logical entity on the physical card. It stores user subscriber information, authentication information and provides storage space for text messages and phone book contacts. For authentication purposes, the USIM stores a long-term preshared secret key, which is shared with the Authentication Center in the network. The USIM also verifies a sequence number that must be within a range using a window mechanism to avoid replay attacks, and is in charge of generating the session keys to be used in the confidentiality and integrity algorithms.

simplifies the exchange of content with previously issued (U)SIMs. In conjunction with the Internet technology on Smart Cards, the variety of different data types stored on the Smart Card and delivered by the SCWS is considerably increasing. Moreover, the Smart Card can be responsible for protecting countless data such as music, video clips, purchased ring tones, personal data, and access information for various mobile services.[7]

2.3 Selected Standardization Bodies for Secure Environments

2.3.1 ISO / IEC7816

The International Organization for Standardization (ISO) is an international standard-setting body composed of representatives from various national standards organisations. Founded on February 23, 1947, the organisation promulgates worldwide industrial and commercial standards. ISO also has a strong position in the field of IT Security and Smart Cards. Here, ISO's Sub-Committee 17 (SC17) has responsibility for developing standards for Identification Cards and personal identification, including a lot of Smart Card-related aspects.
One key factor for the success of Smart Cards was the ISO7816 series, which has been issued by the SC17 group and several national groups. The ISO7816 series describes many Smart Card-related aspects from physical and electrical characteristics to the communication protocols and data structures. Therefore, the ISO7816 is the basis for other standards, which build new functionalities on top of the ISO7816 series. Examples for standards using ISO7816 as basis are the specifications issued by ETSI SCP[8] or the JavaCard standard.
In a greater granularity, SC17 has different Work Groups (WGs). Here, WG1, for example, focuses its attention on physical characteristics and test methods for identification cards [107]. Within WG1, Task Force 2 (TF2) dedicates its attention to durability of such cards.

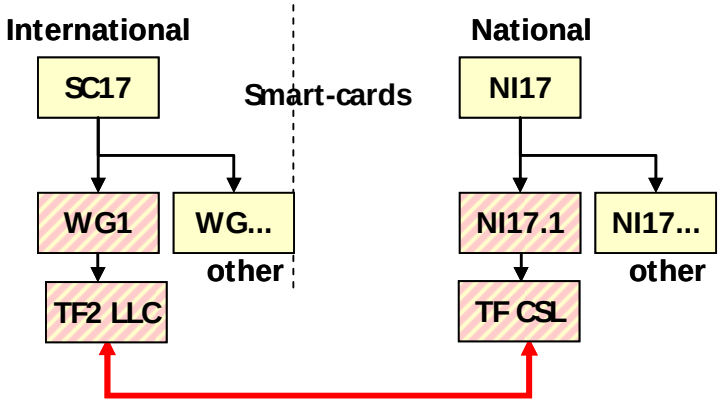


Figure 3: Correspondence of ISO SC17 standardization with NI17 group

Each international dimension of ISO's Sub-Commities and the respective work groups is complemented by national initiatives (NIs). Taken together, these different groups represent the overall standardization body.

[7] In addition to Smart Cards, technologies such as Trusted Platform Modules (TPMs) and ARM Trust Zones (TZs) have recently been conceptualised and presented to the market place. In difference to the Smart Cards and Tokens, these technologies are physically mounted into mobile devices. As they mainly serve the same target as Smart Cards and can in fact be understood as physically integrated Smart Cards, the here presented argumentation focuses on Smart Cards and Tokens only. By doing so, it inherently includes aspectes of TPM and TZ technologies.
[8] ETSI SCP stands for European Telecomunications Standards Institute Project for Smart Cards.

2.3.2 Java Card Forum

The Java Card Forum (JCF) enabled Java for the usage as Smart Card programming language. The JCF comprises a technical and a business committee. As previously mentioned, Java has certain advantages in conjunction with Smart Cards, especially because Java applications can be developed, validated and interoperability-tested rapidly across suppliers of compliant Smart Cards. This can significantly reduce the time-to-market for new Smart Card programs, applets, and allow rapid changes to in-field schemes. To date and over the last 8 years, the JCF has worked upon 6 iterations of the Java Card API[9] specifications and associated test and compatibility kits. The group is currently working on enhancements to the 2.2 Java Card API and, in parallel, the newest generation of Java Card specification requirements (3.0).

2.3.3 Trusted Computing Group

The Trusted Computing Group (TCG) [38] is a standardization body, which defines open standards for hardware-enabled trusted computing and security technologies, including hardware building blocks and software interfaces, across multiple platforms, peripherals, and devices. TCG specifications will enable more secure computing environments without compromising functional integrity, privacy, or individual rights. The primary goal is to help users protect their information assets (data, passwords, keys, etc.) from compromise due to external software attack and physical theft. The TCG, a successor to the Trusted Computing Platform Alliance (TCPA), is an initiative started by AMD, Hewlett-Packard, IBM, Infineon, Intel, Microsoft, and Sun Microsystems to implement Trusted Computing. TCG's original major goal was the development of a Trusted Platform Module (TPM), a semiconductor integrated processor core or integrated circuit that conforms to the trusted platform module specification put forward by the Trusted Computing Group and is to be included with computers to enable trusted computing features. TCG-compliant functionality has since been integrated directly into certain mass-market chipsets. Based on the TPM concept, TCG has also introduced a specification for so called Mobile Trusted Module (MTMs) which extends the aspect of security to mobile phone devices:

> *"The TCG has created a core set of security specifications to be used as standardized building blocks. Both the TPM and Mobile Trusted Module specification are based on this same core set of security functions providing the same essential TCG roots of trust. The MTM is tailored to the verified mobile phone framework and offers enhancements suitable for that framework. […These MTM specifications provide] the core framework, commands and control specifications needed to provide a TCG based security building block solution in mobile phones."* [108]

The key element of trustable computing in TPMs and MTMs is a secure boot process. Therefore, one major working topic of the TCG is the definition of such a secure boot sequence in conjunction with different computing environments (e.g. PCs and embedded systems). As the TPM/MTM is a passive device, it has to interact with a trustworthy bootloader to place a trust anchor in the system. The TCG boot process relies on the storage of trust measurement values and a cryptographic key hierarchy.

2.4 Security in Embedded Systems and Virtualization

Security in embedded computing environments becomes more and more important since the usage of embedded systems, such as PDAs and cell phones, in business transactions increases. Different use cases, in conjunction with Near Field Communication (NFC), require a secure environment. A security component, which is embedded in a mobile device, is necessary to store and handle business and personal information in a secure manner. This goes much beyond the functionality of a classical SIM, which has been the first secure element in a mobile device.

Tamper-resistant devices like Smart Cards alone are highly secure, but offer only limited computing capabilities and memory capacity. Combining them within a secured mobile device and the corresponding secure environment eliminates such restrictions. Thereby, greater flexibility can be offered and addition

[9] Note: API = Application Programming Interface. An API is a set of declarations of the functions (or procedures) that an operating system, library or service provides to support requests made by other computer programs.

spheres of service provision can be enabled. Therefore, PrimeLife will also work on secure computing environments in WP 6.2.
Virtualization is a broad term that refers to the separation of computing resources between different operating systems, processes, or other runtime environments. The technology behind virtualization is the abstraction of computer resources. One definition of virtualization taken from Wikipedia is:

> "*Virtualization is a technique for hiding the physical characteristics of computing resources from the way in which other systems, applications, or end users interact with those resources. This includes making a single physical resource (such as a server, an operating system, an application, or storage device) appear to function as multiple logical resources; or it can include making multiple physical resources."* [39]

Throughout WP 6.2, PrimeLife wants to focus on the security aspects of embedded devices and also virtualization [39]. Virtualization offers the possibility to isolate security-sensitive (personal) data and processes dealing with this data. The separation of the computing resources offers the possibility to split computing hardware into two process environments. One process environment is dedicated to security-sensitive information and the other one to standard information processing. To realize multiple process environments on a single hardware platform, a hypervisor[10] is required. Part of each process environment is a virtual representation of the underlying hardware that makes it possible to host a standard operating system and a secure operating system.

2.5 Usability Analysis of Secure Environments

It is obvious that the security, which is required by identity management systems, can be on different levels. Therefore, it is a goal of PrimeLife in WP 6.2 to sketch a scalable security environment for identity management systems and a general proposal to apply the previously described security environments according to the desired level of security that has to be guaranteed.
Looking at the requirements of an identity management system, it is important

- to have a portable device to store personal information in a highly secure manner,
- to have a secure environment to process this personal information and to give information away on an authorized request.

It is therefore advisable to have a combined solution of a security token or Smart Card, which exchanges data with a secure execution environment on an embedded system or PC.
A Smart Card or token can then offer a maximum of portability and flexibility whilst still assuring a high level of security. The secured embedded system as complementing partner in the overall security solution will in turn offer much more memory capacity, computing power and also flexibility in usage.
Moreover, there are approaches to use the computing power of embedded systems also in conjunction with security critical processes (e.g. on the Smart Card) through the definition of different isolation mechanisms.
The following two sections give a brief overview of the different strength and weaknesses of Smart Cards and Tokens versus Secure Embedded Systems:

2.5.1 Strengths & Weaknesses of Smart Cards and Tokens

Smart Cards and Tokens show the following application strengths:

⊕ High Security

⊕ Highly standardized interfaces e.g. ISO7816, ETSI SCP…

⊕ Centralized personalization and individualization processes

⊕ High portability of the personal secure environment

In turn, they also show the following application weaknesses:

[10] Note: A hypervisor (also: virtual machine monitor) is defined as a software-based virtualization platform that allows multiple operating systems to run on a host (e.g. PCs or mobile devices) at the same time. A hypervisor establishes a seperation of various open and / or secure software modules within the host's software.

⊖ Limited memory capacity and computing power

⊖ Not always under control of the device owner, e.g. SIM, Company ID

⊖ No user interface

2.5.2 Strengths & Weaknesses of Embedded Security Systems and Virtualization

Embedded Security Systems / Virtualization show the following application strengths:

⊕ Easy to integrate, because no hardware interfaces

⊕ Large memory and computing power in comparison to Smart Cards

⊕ The device user has control over the device

⊕ Availability of a display and keypad

In turn, they also show the following application weaknesses:

⊖ Midrange Security

⊖ Limited portability of the personal secure environment

In essence, Smart Cards and Tokens can provide high security in a mobile and flexible manner. Embedded Security Mechanisms and Virtualization may provide significant processing power for security relevant applications. Hence, a combination of these two dimensions could provide a system which can cover all levels of security, be static as well as flexible and highly performing.
As an example, such systems could provide Smart Cards or Tokens for mobile devices which can store different identities and assist in using selected ones of these for different services such as payments, bookings or participation in online communities. The Embedded Security Mechanism would assist in decoding and processing the data stored on the Smart Card or Token, thus making the overall system secure and highly performing.

2.6 Interoperability Layer

2.6.1 Introduction

This chapter mainly investigates authentication and authorization infrastructures as the prevalent method used to establish trust in content distributed online today. While the prior sections dealt with trusted computing infrastructures and the next sections will mainly deal with the mechanisms deployed to make and keep content trusted on the server side, this section is concerned with finding flexible ways to integrate those two layers.

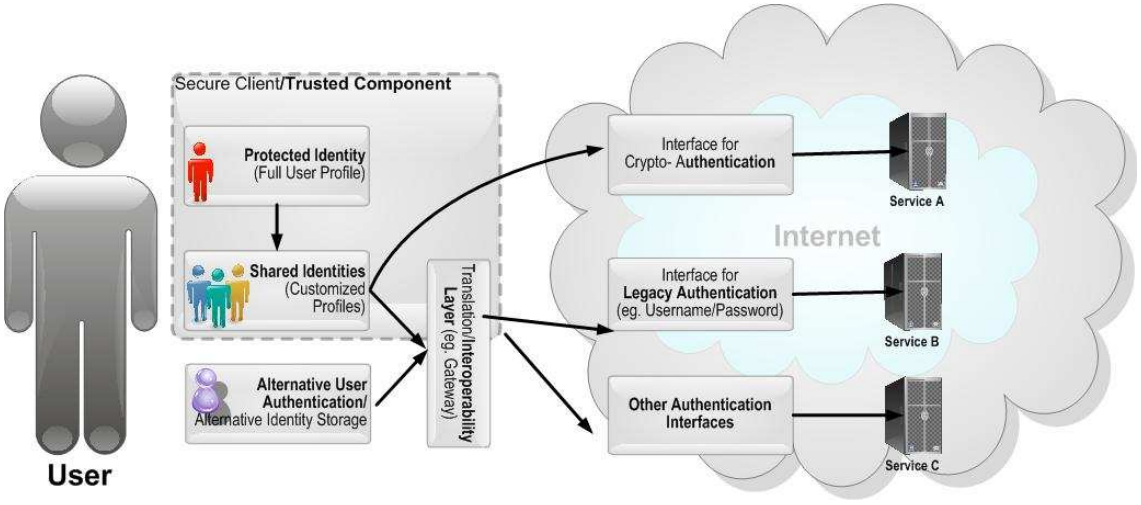


Figure 4: The Interoperability Layer within a Generalized Overall Architecture.

Current authentication deployments exhibit several shortcomings. In the web environment, users need to come up with passwords for a lot of different services. Examples are web based mail, e-commerce sites and discussion forums. Passwords are also widely used for authentication in email, operating system login, remote shells, databases, and instant messaging. This dominance of isolated password systems leads to a large number of passwords that a user has to generate, memorize, and remember[11]. However, remembering a lot of randomly selected, independent passwords is quite straining for users, especially if some passwords are used only occasionally. Users tend to either choose weak passwords [4], enabling dictionary or even brute force attacks, or choose related passwords for several or even all accounts [2], which makes the authentication system vulnerable to cross-service attacks [11].
This behaviour occurs in addition to general insecurities attributed to password systems, well as interception of passwords [42]. Even projections of the password authentication infrastructure's collapse as a whole exist [41]. These projections are based on limited cognitive capacities of the users in comparison to the exponentially growing power of automated hacker attacks.
So, there are three main ways in which current authentication infrastructure developments try to advance the state of the art:

- Claim-based authentication proposes to use credentials (e.g. SAML Tokens[12]) to authenticate by showing claims asserted by trusted third parties.
- Protocols like IDEMIX[13] improve claim-based authentication by ensuring anonymity and unlinkability.
- Specific deployments often advertise offering a unified interface for identity management, including single sign-on (SSO)[14] support for web sites, to improve the user experience.

However, those improvements, and specifically the SSO aspect, depend on a broad adoption of identity federation protocols by services, which has not been achieved by any proposed protocol as of yet. OpenID[15] is quite broadly adopted at the moment by blogs and similar services and MS Passport[16] was during its heyday even adopted by big players like eBay and monster.com, but a perceptible adoption of sign-on in everyday internet use for a broad group of users would still require an even broader adoption of these technologies. Service providers implementing such protocols obviously have to make an investment

[11] Note: Memorizing relates to the ad hoc ‚storing' a password in your mind. Remembering refers to ‚retrieving' the password from your memory after a longer period of time in which you might not used the password.
[12] SAML = Security Assertions Markup Language. SAML tokens are XML representations of claims. [109]
[13] IDEMIX = Identity Mixer. IDEMIX is an anonymous credential or pseudonym system. [110]
[14] Supporting each and every service is often not feasible for SSO systems spanning large networks [22]. The scenario where only partial integration is reached will be referred to as "reduced sign-on".
[15] OpenID is a an open and decentralized identity system. It eliminates the need for multiple usernames across different websites, simplifying your online experience. [111]
[16] Microsoft Passport Network was originally named.NET Passport and is now administered under the Windows Live ID brand. It is a single sign-on service developed and provided by Microsoft that allows users to log in to many websites using one account (see [45] and [112]).

that may be jeopardized by protocols emerging in the field afterwards. Also, replacing passwords as an authentication mechanism may add complexity and thus contribute to the loss of users. Additionally, it may not be in a service provider's best interest to standardize in the area of authentication. He may be putting his lock-in of existing customers at risk by allowing them to easily apply their stored identity in other contexts.
So, while reaping the benefit of such solutions often requires the implementation of specific interfaces by the individual service providers, missing incentives for services to become relying parties often lead to a limited adoption by services which in turn limits the benefit for the user and by this hinders the spread of the identity management infrastructure.
Employing an extra interoperability layer to translate authentication tickets (which are defined in this context as the secrets sent to service providers, e.g. passwords or credentials) originating from the card into a format that is digestible by non-cooperating services (see Figure 7), may alleviate this problem by enabling the system to deliver a meaningful reduced sign-on experience to the user without requiring either the implementation of specific protocols or the cooperation of services. Thus, interoperability with legacy authentication systems at services – such as the username/password authentication interfaces dominating today's web – may be a valid strategy to increase the user-side adoption of identity management infrastructures.
This reflects similar experiences from enterprise scenarios, where integration with legacy systems not supporting the protocols employed by the newly deployed Electronic Identity Management systems is often a business requirement [22]. However, in addition to it being applicable to the services he or she wants to use, a user may also have security requirements towards an envisioned new authentication infrastructure.

2.6.2 Requirements for Interoperable Authentication

Several requirements can be derived from the discussion in section 2.5.1 and will be listed in this section, together with a short summary of the rationale behind them. To allow for a structured presentation, the requirements will be categorized as either security requirements, necessary to ensure the security of the basic operations of the system, interoperability requirements, necessary to ensure its capability to integrate with different identity management infrastructures or further stakeholder requirements, reflecting requirements that do not directly influence the usefulness of the infrastructure, but rather improve the ease of use for the different stakeholders.

2.6.2.1 Security and Privacy Requirements

- **Security of Generated Authentication Ticket:** The authentication tickets sent to services must be pseudorandom and independent. Furthermore, a service authentication ticket for any site must not give any information on any other services' tickets. As a corollary, generated tickets should be chosen from a suitably large set and should ideally be uniformly distributed, to avoid efficient (e.g. dictionary) attacks focusing on a specific, more probable subset.
- **Protection of Central Authentication Secret:** Given a single secret, e.g. a secret signature key or master password, the system should generate secure, distinct authentication tickets, e.g. passwords, for each web site. The central secret should be protected using the strongest available measures, as it is also a single point of failure ("Keys to the Kingdom") [14]. Preferably, the usage of a single secret should not be enforced, but it should rather be an option for the users to employ several secrets for different services, if they deem it necessary.
- **Minimization of Attack Surface against Insiders:** Unlike protocols employing an authentication proxy, we aim to realize a protocol that cannot be executed by a third party alone in order to thwart impersonation of the user by this third party. The same holds true for the individual service providers, who in the wide-spread password authentication scenario could leverage the tendency of users to reuse passwords at several services [14] for cross-service attacks (this has been referred to as the "Domino Effect", as affected services fall - in potentially great numbers - one after another [3]).
- **Minimal Release of Personal Information:** The personal information released by the interoperability layer should default to the smallest possible amount. This amount is dependent

upon the requirements of the respective service, the capabilities of the client, and the underlying identity federation protocol. Hence, the interoperability layer shall not lower the security of the overall system by releasing more information than required by the front- and the back-end.

2.6.2.2 Interoperability Requirements

- **Compliance with Legacy Server-Side Authentication Policies:** The generated authentication tickets must conform to services' expectations. E.g., in the case of passwords, each generated password needs to be accepted by the web site, i.e. it must comply with the service's password policy. A certified credential would require that the service trusts the CA to reap the benefits of certification.
- **Integration of Strong Authentication:** The architecture should be able to build on existing Smart Card or public key infrastructures (PKIs) and other existing infrastructures providing strong authentication to be applicable to high-security scenarios. To make the system easily deployable on top of e.g. an existing signature card infrastructure, we would prefer to use only algorithms present on a wide range of hardware.

2.6.2.3 Stakeholder Requirements

- **Pervasiveness:** We aim for inter-device mobility, making storage of authentication information on the device impractical. Additionally, the implemented authentication mechanism should be executable on mobile devices, in spite of their limited performance.
- **Minimal Interaction:** The system should require minimal user interaction, as each necessary interaction step significantly reduces the acceptance of security-related systems. It has been stated that "The user base for strong cryptography declines by half with every additional keystroke or mouse click required to make it work." ("Ellison's Law") [3]. However, note that this does not necessarily override other requirements. For the release of personal information, for example in banking applications, it is still necessary to collect the user's consent in some fashion and therefore have a step of interaction in the overall process. But still, reports from the field suggest that the impact of an overabundance of options on the adoption of the system will probably be negative.[17]
- **Consistency:** The user experience offered by the system across several log-in domains, potentially using several protocols, should be as uniform as possible, easing the learning curve and operation of the system.
- **Minimal Provider Costs:** We aim to minimize necessary server infrastructure, while still meeting the interoperability and usability requirements. This applies to the number of services involved, the complexity of server-side components, and costs of the integration of additional identity management infrastructures.
- **Controlled Release of Personal Information:** Control over the release of his personal information shall always be held by the user. This requirement is mandated by user concerns, legislation [23], and can also be seen as a security requirement**.**

2.6.3 Design of an Exemplary Interoperability Layer

2.6.3.1 Overview

We here propose a pluggable architecture, designed to make adding new Smart Card APIs or other authentication factors easy. Also, password encoder and transmission components have been designed to be easy to replace. An additional benefit of this modular approach is a small, portable core containing the key algorithms.

[17] Note: It canbe expected that systems which want to offer too many functionalities may cut themselves of from any meaningful adoption, as complexity lowers ease of use, and missing ease of use kills adoption.

The basic data flow can be summarized in four steps and is based on [17]:

1. Define a scheme for deriving service identifiers for the different service providers the user might want to authenticate to. This can be implemented by concatenating several attributes of the service, such as a service name, URL, user's login name, IP address, and so on.
2. Combine the identifier for the service with the user's central authentication secret using strong cryptography [1] [8].
3. Transform the resulting value into a pseudorandom service authentication ticket, for example an account password.
4. Transfer the password to the appropriate web service. This may be realized by an application that e.g. integrates with current browsers, maybe as a plug-in [9] [16], or alternatively operates as a proxy, intercepting and rewriting network traffic, e.g. HTTP requests [8]. Implementations that generate passwords for additional services, such as database access or remote login, are also possible.

Steps 1. and 2. allow for multiplexing of authentication tickets, which is useful when only a limited number of tickets can be supplied, due to restrictions of tokens holding the information or because of cognitive restrictions in the case that passwords are used directly.
Steps 3. and 4. are especially relevant in the context of interoperability. They allow for presenting a derived security ticket to a service as a password. However, both of these steps require additional meta-information, describing the authentication deployment of the individual services.

2.6.3.2 Exemplary Data Flow

In this section, a sample design for an interoperability layer connecting a signature-capable Smart Card to legacy password-based mechanisms is presented. We propose a design for a proof-of-concept demonstrator in a specific scenario, not a full-scale infrastructure yet. We propose using a multi-platform language like e.g. Java for easy porting of the core components and easy deployment on many platforms, including e.g. mobile devices and various host applications, for example web browsers.
This is chosen as an arbitrary existing SIM card infrastructure, demonstrating the adaptability of the system. It offers strong cryptographic capabilities, namely it is capable of creating RSA signatures [15] and also provides 3DES encryption.
We propose a pluggable architecture, designed to make adding new Smart Card APIs or other authentication factors easy. Also, password encoder and transmission components are designed to be easy to replace, to allow plugging in other infrastructure adapters in their place (see Figure 5). An additional benefit of this modular approach is a small, portable core containing the key algorithms.

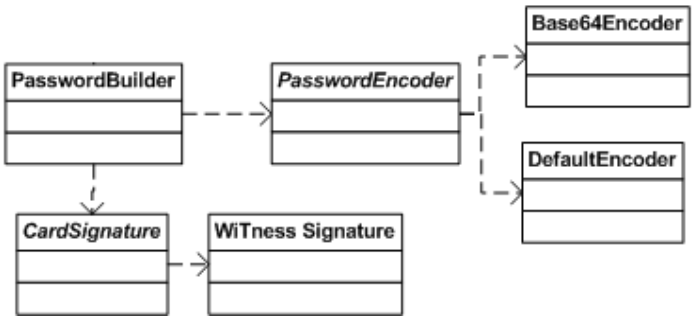


Figure 5: Interoperability Layer Design Overview.

Several cryptographic primitives, such as hash functions [1], signatures, or a combination of algorithms [8], are suitable for step 2. In this example scenario, signature enabled Smart Cards are used as an example of a widely-deployed trusted component.
Like hash functions, electronic signatures can be used to generate strong service passwords for the user. Unlike hash functions, digital signatures have the security property of unforgeability, meaning that an attacker can't produce the user's signature for any text if he does not have the secret key, even if he is given

the user's public key and several examples of signed messages. This would also translate to passwords: An attacker cannot compute any of the user's service passwords without knowing the secret key stored on the Smart Card, even if he knows the user's passwords for several other accounts. The whole process is illustrated in Figure 6:

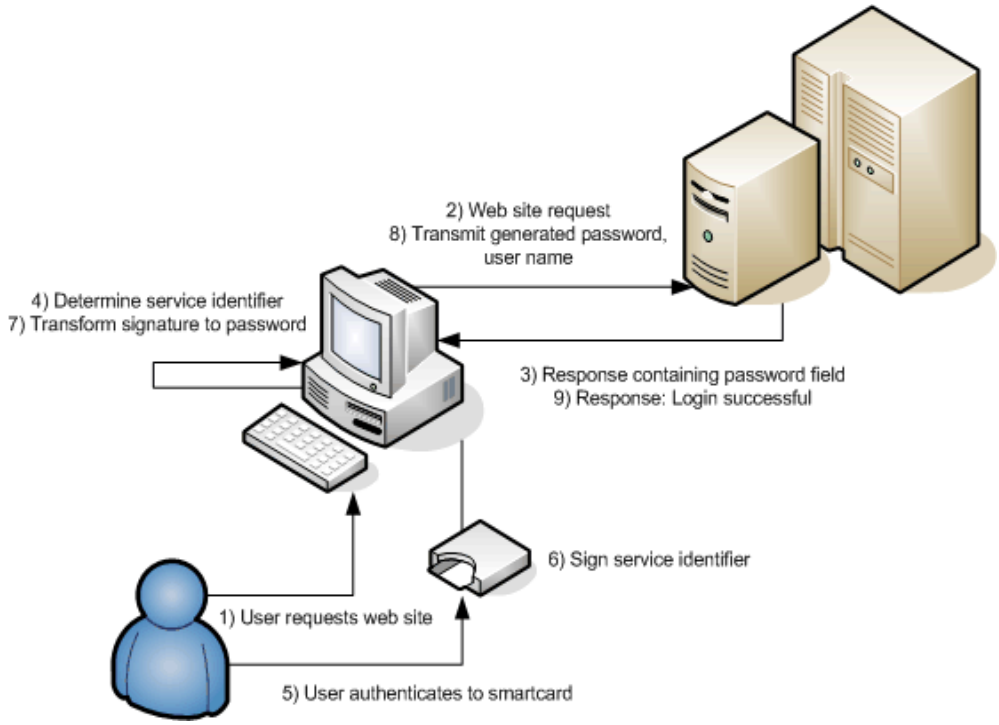


Figure 6: Smart Card Integration with Password System: Data Flow.

When the user needs to authenticate to an e-commerce site (1-3), the local system first derives the service identifier from the available context information, such as the accessed service's domain name and IP address (4). The user authenticates to the Smart Card using his PIN, thus unlocking the private signature key (5). The service identifier for the relevant account is then signed by the signature card using the private key, producing an electronic signature (6). The resulting value is encoded as a password (7). This is a critical step. While unforgeability is guaranteed due to the fact that signatures are used, the distribution and set size of generated passwords are also a factor for security of the system - it needs to output passwords chosen from a suitably large set, and may not employ an overly skewed selection algorithm. The transcoded signature is transmitted to the service provider requiring authentication, along with the user's login name (8). Access to the protected resources is then granted (9).

One advantage of this approach is that the central secret – the user's private key – is actually stored on the Smart Card and not directly dependent upon a user chosen password. Guessing the PIN will only allow access to this key if the attacker is also in possession of the token.

It has to be noted that the consistency requirement can only be achieved using deterministic signatures. This limits the theoretical strength of the system; however, it is an obvious requirement when interfacing with password authentication mechanisms.

The usage of passwords derived from signatures links the user's identity to his intent to use the service. Of course, signatures in this scenario are not linked to individual transactions. This is due to the fact that the widely deployed password systems do not perform user authentication on a transactional level.

The generated service passwords are directly dependent upon the user's cryptographic signature keys. If the user's key pair needs to be replaced, because e.g. it has been revoked, all the generated service passwords will change. While this poses a serious usability barrier in the described basic version of the system, saving account meta-information on a server can improve user experience during this and other use cases. The system is able to iterate over all the accounts, using the same architecture as the master password changing assistant in [7]. Note that, while the revoked key pair's signatures can no longer be verified, they may of course still be encoded and submitted as passwords.

2.6.4 Discussion and Outlook on the Interoperability Layer

Of course, in a SSO system, loss of the central secret – the secret key on the Smart Card token – means loss of all derived passwords. This deliverable does not discuss mechanisms for ensuring the robustness with regard to lost passwords in detail. However, most services offer a means to reset or retrieve a lost password. Additionally, conventional methods, like saving encrypted password lists to secure storage, e.g. on the token, as a backup or replacement, may be used.

As pointed out in [1] and [8], unlinkable user pseudonyms may also be generated in a similar fashion, which would be especially useful when combined with anonymous communication channels, based on e.g. TOR [8].

Apart from storing passwords in an encrypted form, it is also possible to generate them on the fly, using strong cryptography. However, such methods have to meet several requirements to guarantee their usefulness for user and service provider. Our system's main concern is building a secure, interoperable authentication infrastructure on legacy systems (Smart Card infrastructures and password authentication mechanisms), rather than e.g. aiding anonymous service usage [8].

Also, the portability of the Smart Card token, along with the pervasiveness offered by the algorithm's operability without saved passwords, suggest implementing the system on mobile terminals. There are functions on standard-issue GSM SIMs that may take up the role of the signature algorithm presented in the example scenario. However, real-life implementations of these functions are dependent on the individual mobile operator. Also, keys are often shared between subscriber and mobile operator. So, while the system may be easily implementable if a signature-capable card is implemented in a mobile terminal, employing standard-issue SIMs for a similar functionality will require additional investigation, and - at least in some cases - additional authentication secrets.

Using the SSO system does not require trust towards third parties, as opposed to systems based on an authentication proxy or similar architecture. The authentication secret is only handled by user and service, with the central authentication secret remaining on the user side – more specifically, on the token – at all times. The system offers an alternative to hash functions for the purpose of generating passwords on the fly. In addition to the capabilities of hash-function-based systems, the presented implementation makes use of the strength of Smart Card-based two-factor-authentication. It also meets the technical requirements outlined, offering a mobile and interoperable dynamic authentication infrastructure built on legacy systems.

Chapter 3

Privacy in Service Composition

3.1 Overview of Service Composition

This chapter describes the challenges of security and privacy assurance in service composition. The document describes various approaches on data handling which are only lightly covered in today's state of the art technology.

Service-oriented architecture (SOA) has become an important concept when talking about service composition today. It is defined as a form of technology architecture that adheres to the principles of service-orientation [56]. It defines a set of guiding principles to enable the development and usage of applications that are built by combining services. Those services should be autonomous, interoperable, discoverable, potentially reusable, and vendor divers. The interactions between these services are formally defined through contracts that are independent of the underlying platform and from the programming language used. SOA establishes the potential to support and promote these principles throughout the business process and automation domains of an enterprise.

In PrimeLife, we define "*cross-domain service composition*" as any combination of existing services and resources hosted in different security domains. This composition can be an application (e.g. Java, C#), a workflow (e.g. Microsoft Work Flow [86], Business Process Execution Language (BPEL) [71]), or a mash-up (e.g. openkapow [73], Microsoft Popfly [76]). Exemplary use cases for cross-domain service composition could, for example, be: Very formal composition (e.g. virtual organizations, hospital workflow), data mash-up created with a visual editor, or more ad-hoc collaborations between multiple parties.

When services are composed, constraints have to be fulfilled. For instance, the type of data provided by a service has to be compliant with the type of data consumed by another depending service. Performing dynamic service compositions based on security and privacy constraints is only partially covered in state of the art. Only, rudimentary identity management and access control solutions are provided today, when heterogeneous, ad-hoc composed services require authentication. The same applies on privacy constraints. This chapter summarizes the research work planned in WP 6.3, which deals with security and privacy aspects of service composition.

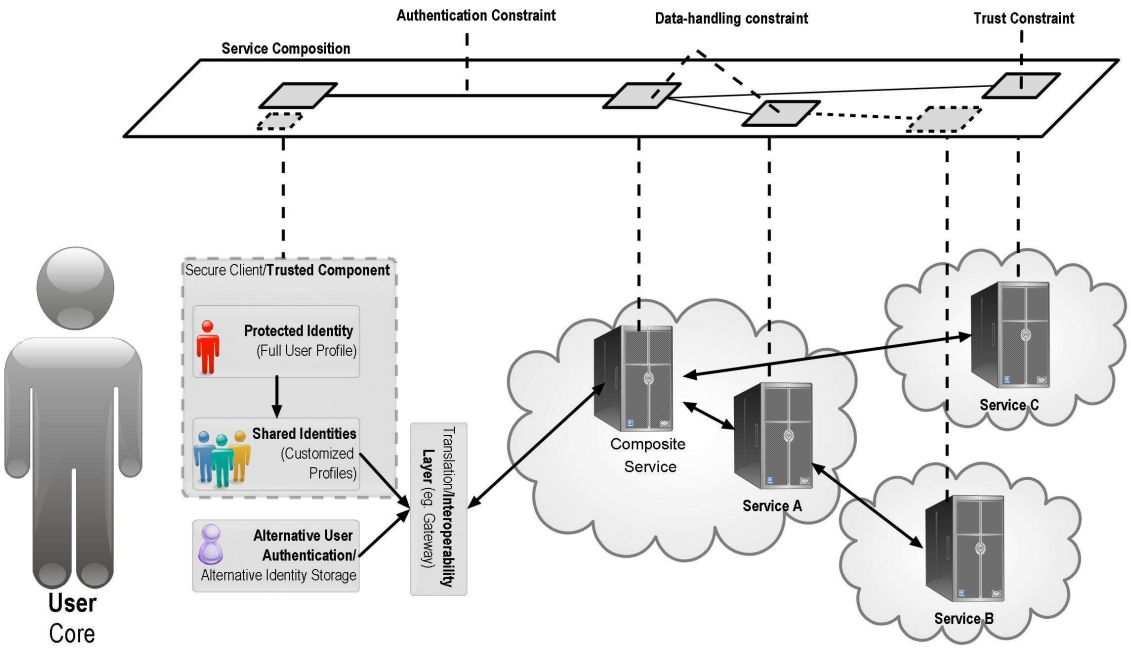


Figure 7: Overall vision of service composition with constraints.

Figure 7 gives a high-level vision of service composition: On top of the different services, the model shows the data flow between different services. They are represented as black boxes. Constraints such as data handling, trust or authentication needs are associated with the links between the services.[18]

3.2 Different Types of Service Composition

Cross-domain service composition follows different concepts and can be enabled by different technologies. These are briefly classified and explained below.

3.2.1 Mash-up

A mash-up is a web application that combines data from more than one source into a single integrated application. Mash-ups make use of APIs[19] published by service providers such as Yahoo!, Google, Amazon, eBay, Microsoft, and others. Those APIs are based on web standards and allow making use of the functionality of the services. The actual invocation of the APIs is quite different; it ranges from programming approaches where programmers consciously take advantage from all details of a given API to more "graphical" invocations which allow service composition without being an expert in programming. For instance, often mash-ups are created using a graphic designer where services can be dragged and dropped as e.g. boxes and connected to each other. Further user input and data can be provided to the mash-up and be consumed / distributed further by the mash-up. Hence, mash-ups can also use other mash-ups as data source. Depending on the technology, mash-ups can be executed either in the browser, on the client side, or on the server side.
Three types of mash-ups can be identified:

- **Presentation / Consumer mash-ups** bring information from more than one source into a common UI[20], such as web portals (e.g. Live.com [64], iGoogle [61], My Yahoo! [69]) displaying the

[18] Note: This Figure does not define the architecture of service composition with constraints but only a high-level representation of the problem space and envisioned approach.
[19] Note: API = Application Programming Interface. An API is a set of declarations of the functions (or procedures) that an operating system, library or service provides to support requests made by other computer programs.
[20] Note: UI = User Interface. The UI (or Human Computer Interface) is the aggregate of means by which people—the users—interact with the system—a particular machine, device, computer program or other complex tools.

information side by side. Little real integration is involved. Mechanisms such as drag and drop of pre-built widgets or RSS feeds are used.

- **Data mash-ups** extract data from multiple sources and combine it. They enable cross-referencing and comparison of data, e.g. mix of geographical data with Wi-Fi hotspot locations, house prices, or crime statistics. Extraction of these data might be hard and require programming. Visual development environments exist to build data mash-ups (Yahoo! Pipes [87], Popfly [76], Dapper [54], openkapow [73], Serena Software [80], etc.).

- **Logic/business mash-ups** combine data, people, and processes. They connect two or more applications and automate certain tasks. They always involve programming. Within the enterprise, they overlap with traditional workflow applications and with composite applications, but they should enable rapid customization and adoption (Serena Software). Mash-ups like these show more overlap with SOA than the other two categories where client-side and server-side mash-ups compete with server-side orchestration technologies such as BPEL.

The concept of mash-ups is advancing quickly. So are the ways to use and mix different types of mash-ups. Therefore, the description above gives just an indication of what is being done at the moment. However, further combinations and variations among them should clearly be expected.

3.2.2 Orchestration

Orchestration allows creating service compositions by connecting different processes through common workflow logics. An orchestration expresses business process logics that are typically owned and controlled by a single organization, even if that logic involves interaction with external business partners. An orchestration establishes a business process protocol that formally defines a business process structure. The internal workflow logic is broken down into a series of basic and structured activities that can be organized into sequences and flows. It is important to notice that the workflow acts as a meta-structure which organizes ("orchestrates") the communication between different service nodes. Orchestration is an important part of SOA because it provides the means of expressing business process logics in a standardized and service-oriented way. WS-BPEL [71], the Web Service Business Process Execution Language is an industry specification that standardizes orchestration.

3.2.3 Choreography

Choreography aims at organizing multiple applications within an organization and even information exchange between multiple organizations. The goal is to establish a kind of organized collaboration between services representing different entities (organizations). The participants in a choreography act in different roles and have different relationships. Typically, there is no single owner of the collaboration logic. Once the collaboration protocol with the message exchanges has been defined, the choreography is self-organizing. WS-CDL [51], the Web Service Choreography Description Language, is one of several specifications that deal with choreography.
The big difference between orchestration and choreography is the way the control of data flow is done. While orchestrations are organized by a central logic that controls how the data flows, the control within choreography is more decentralized and is governed by message exchange patterns. There are certainly overlaps between orchestration and choreography where choreography could be used to connect multiple orchestrations etc.

3.3 Requirements for Advanced Service Composition

Composing cross-domain services can be seen as connecting the black boxes presented in Figure 8. Those boxes can represent concrete services, abstract services, activities, or users and can be hosted in different trust domains. State of the art technologies (i.e. mash-ups, orchestration and choreography) only take functional constraints, such as the type of data exchanged between different blocks, into account. The

vision of Work Package 6.3 is to also consider non-functional constraints, such as security and privacy while composing services.
The connections between two blocks can be simple (e.g. direct link between data source to a data consumer[21] in a presentation mash-up) or complex (e.g. take a decision based on former results, combine multiple data, and hand the result over to another party).

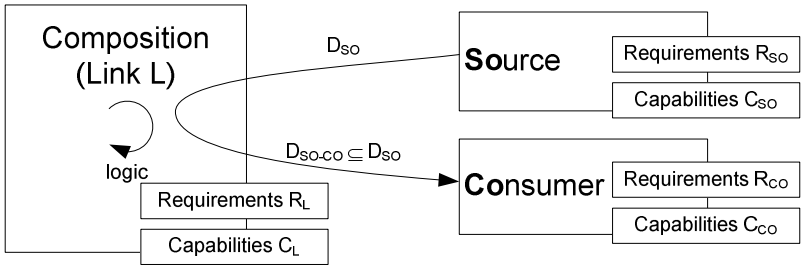


Figure 8: Atomic service composition.

Figure 8 explains the potential interactions:

The variable D represents the data. Any CO in the indexes means data consumer, SO refers to the source of the data, L the composition that links both entities together. Requirements are of the service are noted as R whereas its capabilities are C.

1) Source SO provides data D_{SO} to the composition L.
2) The composition L is the link between source SO and consumer CO. L uses D_{SO} to take a local decision.
3) L provides to the consumer CO a subset of D_{SO}: $D_{SO\text{-}CO}$

Variations:

- Multiple sources SO_n where $n \in \{0..N\}$, $N > 0$
- Multiple consumer CO_m where $m \in \{0..M\}$, $M > 0$
- Multiple parties (SO, CO, and/or L) implemented as one entity or hosted in same trust domain.

We do not pretend that this oversimplified picture is sufficient to deal with all aspects of service composition, but it makes clear that data handling policies are a key issue.
We also make a difference between static cases, where security and privacy constraints are solved at design time, and dynamic cases (also see section 1.3 on static and dynamic environments of services and security).
In static cases, a modelling tool to compose services will combine services potentially hosted in different trust domains in order to define the business logic. On the one hand, security aspects must be part of the composition, since the composite service must be authorized to access the services because privacy policies of each service must be reflected to the users of the resulting composite service. On the other hand, this must be done in an abstract way to avoid polluting the composition with non-relevant details which will complicate future updates of the composition.
In more dynamic cases, parts of those issues have to be solved at runtime, e.g. when the services that are part of the composition depend on the identity of the user, contextual information, or discovery mechanisms. For instance, the URL of a patient's online medical record service used by a hospital workflow depends on the patient and cannot be known at design time. Moreover, privacy constraints (e.g. data handling, access history, etc.) may, for example, vary depending on the medical record service.

Figure 7 has indicated that cross-domain services can face numerous constraints. In general, the following ten types of constraints can be identified:

[21] In the scope of privacy policies the data consumer matches the term 'Data Receipient' from the Data Protection Directive 95/46/EC. We will continue to use the term data consumer.

- **Interface:** When composing services, it is necessary to ensure that the interfaces match, i.e. type of data consumed should be compatible with the data provided. This set of problems is related to service discovery and somehow out of the scope of this work. Therefore, basic mechanisms will be used.
- **Channel**: Services must be able to communicate. This problem is somehow out of the scope of this work, e.g. SOAP versus REST.[22]
- **Channel Security**: The designer of the composite application should be able to define basic security goals for data passed between the services in the composition. He should be able to impose constraints such as, for example, that integrity of data passed to and from a service within the composition has to be protected through a signature typically generated by the respective sending service. Similarly, confidentiality of data could be enforced by encrypting the data and authenticating the consumer. This should be done either globally at the composition level or for single services used within the composition.
- **Trust:** This constraint defines restrictions on the origin of services that are used for composition. If specified, the composite service can only use services that fulfil the trust requirements. The constraint can have different level of trust, from absolute/verifiable trust required to partial/percentile requirements depending on the nature of the application to be designed. Trust can be verified through the use of a PKI with X.509 certificates [99, 100], through belonging to corporate domains, through a "white list" of services, or through ad-hoc mechanisms such as recommendation (web of trust [101], PGP [101], etc.) or reputation systems. This constraint can be applied at design time as well as at runtime depending on the purpose of abstract services. For example, when designing a travel advisor composite service for employees by using an abstract service to get flight information, the company policy might state that only services provided within the company may be used for composition.
- **Data Handling**: Data handling directly depends on the privacy policy language that will be defined in PrimeLife's Activity 5. Section 3.4 more precisely describes how service composition (WP6.3) and privacy policies (A5) are related.
- **Authentication and Identity Management**: Authentication is only slightly covered in state of the art service composition. For instance, Microsoft Popfly (see Section A.1.3) offers a basic mechanism to authenticate to Flickr [58]. Additional mechanisms will be studied in WP 6.1 on identity management and WP 6.2 on trusted infrastructure.
- **Authorization and Delegation of Access Rights**: Whoever wants to access such a service has to be authorized to do so. Given today's variety of access control mechanisms, every service in a service composition may use (and most likely will use) its own mechanism. Hence, if Alice wants to grant access rights for a service composition to Bob (act of delegation), she needs to express this new access control policy in different ways depending on the access control mechanism of each respective service. Hence, authentication as well as access control have to be taken into account during the modelling phase. The model has, however, to abstract the concrete details that are not always available at modelling time.
- **Audit / Storage / History / Non-Repudiation**: This is a set of constraints that deals with the purpose for and the way in which data is accessed, processed and stored within a composite service and the services it is composed of. A designer can have strong requirements regarding the validity and reliability of information within a composite service and define a set of rules to which all services have to comply in order to be auditable. Data storage constraints can be defined, e.g. for how long data is stored, how well protected is it from exposure, etc. Other constraints can require that all services keep a history of transactions and operations or acknowledge the fact of having executed an operation or received some data in a way that cannot be repudiated.
- **Reliability**: Another set of constraints deals with the reliability of services that are used for composition. For certain applications, the availability of services and information is critical and data loss or service outage could do a lot of damage. For this kind of applications, reliability constraints could be defined that would specify that the composite service only wants to use services that are duplicated, that are protected against DoS attacks, that backup their data, etc.

[22] REST stands for "Representational state transfer" and denotes a set of architecture principle for distributed systems. SOAP is seen as a complementing pattern that takes advantages of various web service standards.

- **Miscellaneous**: Finally, here is a non-exhaustive list of additional constraints:
    - Separation of duty, e.g. activity A cannot be executed by the same person/service as activity B.
    - QoS, e.g. only accept services with a given response time.
    - Price of the service usage, i.e. compare the cost of different services (network access, send message, etc.)

3.4 Requirements for Privacy Policy Composition

The objective of this section is to provide an updated list of technical requirements to PrimeLife's Activity 5. Therefore, this section lists some requirements that emerged from the discussions on security and privacy aspects of service composition. The requirements that are defined in "PrimeLife Privacy Policy Language" (i.e. Activity 5) need to be kept in mind to ensure a smooth integration of this language into the Work Package 6.3 (i.e. Service Composition).

3.4.1 Assumed Properties

The requirements described below will be refined, once a draft of the "PrimeLife Privacy Policy Language" will be available. In the meantime, however, we need to assume that this policy language will support the following key data handling features:

- A way to classify data types (e.g. personally identifiable information (PII[23]), E-mail, medical data). This certainly requires some structure or hierarchy of PII. It is still unclear whether the set of data types can be predefined or whether it should be extendable (inheritance, ontology, …).
- A way to describe the purpose for which data is collected (e.g. contact requestor, statistics, etc.). Similar to PII, some structure or hierarchy of purposes is necessary. It is still unclear whether the list of purposes can be predefined or whether it should be extendable (inheritance, ontology, …).
- A way to associate conditions and obligations to data handling (e.g. service x must delete E-mail address after 30 days). We did not yet find a well defined technical set of obligations required for the authorization and/or the data handling. For instance, the languages XACML and EPAL support obligations but do not specify them.

3.4.2 Composability

When composing services (i.e. creating the logic and dataflow that links services) it is necessary to detect incompatibilities between policies and to ensure that all policies and preferences that can be enforced locally will indeed be enforced at runtime.
In the "travel booking" scenario (see Section 4.4.1), for example, the privacy policy of the travel booking service could be the concatenation of the policies of all underlying services. In most cases, however, this is not appropriate because not all underlying services will be required during the interaction. For instance, a bike booking service could be used by the travel booking service. However, since only one percent of the travellers book a bike, it may be wiser to postpone decisions regarding the bike booking data handling policy.
In another example, a hospital workflow may need to access very specific medical data (e.g. organ donation consent) for a very small number of patients. As it is impossible to ask a patient during surgery, it is mandatory to get such a patient approval during registration. This decision is clearly application-specific and additional information (e.g. the annotation of the business process) is necessary to compose policies and define what is optional in these.

[23] Note: PII (Personally Identifiable Information) is defined as any piece of information which can potentially be used to uniquely identify, contact, or locate a person or to reduce a person's level of anonymity.

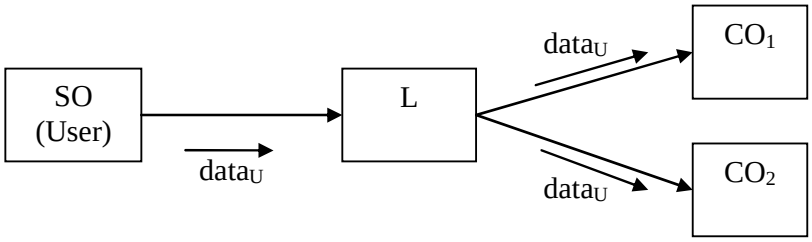


Figure 9: Basic service composition.

Part of policy composition is application-specific. The policy language expressing requirements and capabilities as well as service composition tools must enable the developer of composite services to make the right choices. We see different types of choices:

- *Optional Policy*: The developer of the composite service should decide whether some parts of a policy are optional (e.g. bike booking) or mandatory (e.g. organ donation). It is also necessary to have a way to describe why a part of the policy is optional. Possible reasons could be related to postponing approval (in the few cases where user approval is required, the user will be asked later) or to choices (discount or better QoS because user accepts to share his age for statistics).
- *More Generic Service Description*: The developer of a composite service should decide how the underlying services are described. Concatenating all policies would result in a statement that may be too precise, e.g. specifying the identity of underlying services, and thus would disable replacement of services. It is necessary to let the developer choose how services are described. Similarly, the purpose of data handling may be made more generic.

The following basic composition can be used to illustrate the above presented procedure (see Figure 9). The syntax used below is only defined to illustrate the example. Assuming that data consumer CO_1 uses "Email" for the purpose of "Confirmation" and that a Trusted Third Party ("TTP") certifies that CO_1 is a "BookingService". Let's assume that another data consumer CO_2 uses "Email" for purposes "Confirmation" and "Statistics" and that TTP certifies that CO_2 is a "BookingService". The default concatenated policy of the service composition L could therefore be:

- L uses "Email" for purpose "Confirmation"
- L sends "Email" to CO_1
- CO_1 uses "Email" for purpose "Confirmation"
- TTP certifies that CO_1 is a "BookingService"
- L sends "Email" to CO_2
- CO_2 uses "Email" for purpose "Confirmation" and "Statistics"
- TTP certifies that CO_2 is a "BookingService"

Some steps of this process could be replaced. In this case, the policy of L could turn into:

- L uses "Email" for purpose "Confirmation"
- L sends "Email" to any service certified as "BookingService" by TTP for purpose "Confirmation"
- <OPTIONAL> L sends "Email" to any service certified as "BookingService" by TTP for purpose "Statistics"

This could be seen as a "template". When the data source (i.e. the user) provides data, he must approve confirmation and can choose whether he approves statistics (optional). If CO_2 is called without pre-approval, interaction with the user will require getting approval.

3.4.3 Trust and Delegation of Trust Evaluation

In service composition scenarios where composite services are hosted in different trust domains, trust is a key issue since enforcement of the data handling will be distributed. Trust decisions can be based on the identity of the party (e.g. PKI-based), on certification (e.g. through a Trusted Third Party), or on reputation mechanisms. However, it is not always possible to ask the data source (e.g. the user) to decide whether he

trusts a data consumer behind the composition. In fact, the composite service may be in a better position to decide whether the data consumer is trustworthy. In such cases, the policy language should support a "delegation" of trust evaluations in a specific context.
In the future, trust may also be based on trusted hardware (e.g. certified TPMs). Such trusted hardware could prove that it fulfils the necessary conditions and obligations by applying the following mechanism: The trusted hardware runs a trusted Operating System which then controls a trusted service. Hence, the policy language of the service should be able to be hooked to the trusted hardware as the "root of trust". If the trusted hardware could unmistakeably be linked to an individual (e.g. through biometric applications) then, the identity chain could span from the individual to the respective service provider.

3.4.4 Disclosing Policy versus Disclosing Preferences

At the present research status, the negotiation of data handling[24] has not yet been considered because in most scenarios, the service decides the policy and may offer various choices. In general, however, and even in service-driven approaches, the matching of user preferences and service policies can be done in the following ways:

- The policy is displayed by the service (e.g. P3P[25]) and the user verifies that this policy matches his preferences. The user decides to abort the transaction if there is no match.
- The preferences of the user (e.g. sticky policy) are attached to the data and sent to the service that enforces them. The service decides to abort the transaction if there is no match.

The first approach is difficult to achieve in complex service compositions and the second approach requires the user to reveal all his preferences. Therefore, a trade-off is certainly necessary. Mechanisms based on "templates" seem appropriate (e.g. partially filled data handling statements). In case of optional policies, it may be necessary to have more than one round of interactive exchanges with the user to fill different parts of a same template.
In both ways, enhancing privacy might have inconvenient effects on the user experience. Ideally, interactions with the user should be minimized while privacy is still ensured. Further, the user should get meaningful questions and should on one hand be able to "generalize" his preferences to reduce the number of subsequent questions and on the other hand be able to divert from a generalized decision. The user should also be able to delegate the administration of his preferences to another party.

3.4.5 Same Language for Preferences and Policy

If possible, the same language should be used to express preferences (requirements) and policies (capabilities). This allows a matching within one language domain, without using a policy matcher that understands two or more languages.

3.4.6 Local Enforceability

Any party should be able to verify that it's applied mechanisms indeed regard and enforce all data handling requirements in place (i.e. policies, preferences, etc.). Any party should also be able to prove that it knows the policy which is to be applied to a specific data set.
The process of deciding whether an action fulfils all policies and preferences should be computable in an efficient way.

3.4.7 Separation from Access Control

[24] Mechanisms for privacy negotiation: instead of aborting the transaction in case of a mismatch between a privacy policy and user preferences, a negotiation could be engaged in order to find a compromise and access to a subset of the private data.
[25] Note: P3P stands for the "Platform for Privacy Preferences Project". This project enables websites to express their privacy practices in a standard format that can be retrieved automatically and interpreted easily by user agents. P3P user agents will allow users to be informed of site practices (in both machine- and human-readable formats) and to automate decision-making based on these practices when appropriate. Thus users need not read the privacy policies at every site they visit. [96]

Although data handling (DH) and access control (AC) are obviously related, it may be advisable to separate them. This is based upon the fact that DH and AC are not necessarily enforced at the same place. Indeed, there can be two places of enforcement when a client sends a request to a service. The term PEP refers to "Policy Enforcement Point".
In the first case (see left side of Figure 10), the client writes data to the service. The service enforces access control policies. The client provides data handling requirements with the data and those requirements (e.g. erase data after 30 days) are enforced by the service.
In the second case (see right side of Figure 10), the client reads data from the service. The service enforces access control policies. The service provides data handling requirements with the data and those requirements (e.g. erase data after 30 days) are enforced by the client. Even if a party acts as client and as service, access control and data handling related to a specific request are not always enforced at the same place.

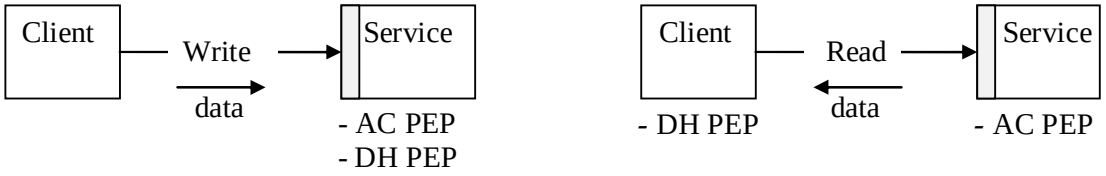


Figure 10: Enforcement of Data Handling and Access Control. [26]

Further, notice should be taken that both types of policies (i.e. AC and DH) may be mixed, if an existing authorization language (e.g. XACML [68] or SecPAL [49]) would be extended to deal with the data handling. Therefore, mechanisms to extract data handling requirements from the overall policy would be required.

3.4.8 **Audit of Data Handling**

The following mechanisms may apply to audit the data handling and obligations:

- Generating guarantees and proofs during the obligation enforcement: Declarative obligations are actually supported by traditional privacy dedicated languages but there are no existing enforcement mechanisms supporting the application and the correct enforcement of such obligations.
- Feedback and reputation mechanisms to evaluate the behaviour of users during their last interactions with protected private data: Did they respect obligations related to private data? Did they really delete Email after 30 days?
- Provision of logging and auditing mechanisms for securely logging and auditing policy compliance and creating privacy feedbacks based on the auditing results.

3.4.9 **Summary and Open Issues on Privacy Policy Composition**

The following issues remain open and call for further discussion in the future:

- Sticky Policies: It is unclear whether policies and preferences (filled template) have to travel and be stored with data. However, each party needs to know what has been promised in order to enforce data handling. For instance, in the travel booking scenario, the user's e-mail may be stored by TravelBooking and sent to HotelBooking later. TravelBooking must know that this e-mail can only be used for contact by third parties which fulfil specific requirements.
- DataHandling Update: Do we want to offer a way to change the data handling requirements associated with specific data? Is this related to a specific type of obligation: provide an endpoint to modify data handling requirements?

[26] PDP = Policy Decision Point. It analyzes policy information, takes the decision and relays it to the PEP.
PEP = Policy Enforcement Point, enforces the policy decisions taken by PDP.

These questions do have an considerable impact on the security architecture. PrimeLife has to answer the questions raised above before the first architecture deliverable of this work-package is executed.

Chapter 4

Potential Authentication Scenarios and Emerging Use Cases

4.1 Description Scheme for Authentication Scenarios

To gain a better understanding of the necessary infrastructure for secure authentication with the help of trusted components, it is useful to have a look at existing authentication scenarios and how Smart Cards or mobile trusted devices in general, as discussed in Chapter 2, could be an improvement.
In order to be able to later compare different use cases, the description of scenarios needs to be based on common structural elements. Therefore, we define the following four important characteristics for authentication scenarios. They shall be applied in future scenario and use case elaborations:

- **Roles**: A common view on the concept of roles is that people take different roles in different situations, depending on which aspect of their life they want to represent. An individual may even apply different roles in a single scenario. For example, roles can be the representation of different partial user identities within one scenario. From a technical point of view, roles are a set of access rights, which have been granted to the individual in a specific service environment [90]. More generally, roles also express tasks or functions which persons perform in a service environment. In the scenarios, we examine which roles are actually implemented and how the users interact with each other based on their roles.
- **Authentication mechanism**: Authentication against a service can be carried out in a variety of different ways. The basic procedure for most of the currently available web services or networked applications is a login sequence wherein a service requests the provision of a unique username combined with a personal password. However, a user normally creates a huge amount of disparate accounts during his web activities. Therefore, the mere combination of username and password can be expected to lose its status as ideal option. A need for different authentication mechanisms arises for convenience and security reasons. Furthermore, establishing interoperability (see section 2.5) between different authentication scenarios or mechanisms seems promising.
- **Supporting technology**: Authentication can be supported through various technologies, mostly depending on the required security level. A Smart Card, for example, is a trusted component which enables secure storage of data, serves as a token or assists in encrypting the data that is to be transferred. In the different scenario descriptions, we therefore investigate which technology is already being used and how the introduction of specific trusted components could improve the use case of the authentication process.
- **Value proposition**: Services largely depend on the users' acceptance/adoption of the applied technology. If, for instance, privacy or security requirements aren't met sufficiently, the service offering will be likely to fail in the market place due to a lack of trust in its adequate provision.

Further, the applied technology has to meet convenience requirements in order to embed the SoS concept into a use case, which is easy to use from the users' perspective. Since the here presented scenarios have already been implemented successfully, we focus on the value proposition that results from the introduction of a new authentication technology into these existing environments.

Based on the above specified description scheme, the following paragraphs outline two exemplary authentication scenarios to introduce the reader to real life examples: *Entropia Universe* serves as an interesting leisure-time-community scenario with involvement of real money transactions and therefore increased security requirements.
XING shows exemplary characteristics of business-oriented social networks with which Work Package 1.2 of PrimeLife also deals/will deal.
The initial purpose of these scenarios within the here presented deliverable is to provide colourful examples of concrete use cases. The cases will be investigated further in the future. Especially in PrimeLife's deliverable D6.1.1, they will, among others, deliver requirements for a privacy-enabled infrastructure for online communities.
Section 4.4. then presents further areas in which the above described technologies could be applied. To provide first technical insights, the two presented examples (Booking and Hospital scenarios) do not follow the above introduced description scheme, but rather dedicate attention to the technical perspectives of execution.

4.2 Managing Identities in Online Gaming Communities: Entropia Universe

Entropia Universe is a persistent online world in which people can create a virtual representation of themselves (so-called "avatars") to meet other users in a fictional 3D-environment (see Figure 11).
Operator is the Swedish company MindArk that focuses on the online entertainment market [91]. Since its start at the beginning of 2003, *Entropia Universe* has acquired approximately 730,000 users. Similar to the well-known life simulation *Second Life* [92], *Entropia Universe* contains a complex economic system. Players within the online world can perform specific tasks (e.g. crafting special items for other users or carrying out particular missions) to earn virtual currency ("Project Entropia Dollars") which they can then exchange for real money (US dollars) at a rate of 10:1. This link to the real world distinguishes *Entropia Universe* from gaming-only environments where users can't actually gain real-world profits.

Figure 11: Screenshot of Entropia Universe.

- **Roles**: In the *Entropia Universe*, the access to certain virtual features and sets of functions (roles) does not depend on the users' life in the real world. Here, the users' roles are implemented in an entirely virtual sense. Apart from the different professions the users might practice in the online world, they build their personal social network within the community. As they progress and get more and more involved into the virtual environment, they may earn the role of a mentor which enables them to "graduate" new players and to receive specific rewards. However, this does not involve an extension of access rights.

- **Authentication mechanism**: Basic authentication for the *Entropia Universe* is assured by the typical combination of username and password to log on to the service. To further protect the user's account, one-time passwords may be used as a complement to the static ones. Their main goal is an enhancement of the security level of the user accounts since these accounts may be associated with a considerable amount of real money due to the economic link.

- **Supporting technology**: For the creation of these one-time passwords, users may request a Smart Card (the so-called "Gold Card", see Figure 12 from the virtual world operator. It is provided free of charge as soon as the real monetary value stored in a user's account exceeds 500 US$ (otherwise it costs 20 US$). Together with the shipment of the Smart Card itself, the users receive a portable stand-alone reader whose sole function is to display the passwords generated by the Smart Card. Once a user has confirmed the use of the Smart Card in conjunction with his account, he will not be able to gain access to the virtual environment without it anymore, since there will always be a prompt for a valid one-time password during the login.

Figure 12: Entropia Universe "Gold Card" and the associated Smart Card reader.

However, introducing the "Gold Card" does not change the general mechanism: Still, typing user name and (one-time) passwords is the only form of authentication. Consequently, there may even be a reduction in user convenience if an additional password has to be entered in order to achieve a higher security level. Using a SIM-like Smart Card in conjunction with a personal mobile device (e.g. a Smartphone), as proposed in Chapter 2, could avoid this reduction in convenience as an additional card and reading device would not be needed. Strong authentication, however, could still be provided. In any case, further maximizing user convenience must always be kept in mind while designing overarching new authentication infrastructures.

- **Value proposition**: As mentioned earlier, the user adoption of a service directly correlates with the proposed security and adequate protection of the users' privacy. At the moment, *Entropia Universe*'s Smart Card implementation is mainly geared towards a few users that are deeply involved with the virtual environment and have therefore acquired great monetary value within their accounts. Employing the Smart Card assists in protecting their valuable achievements by impeding fraudulent use of accounts. However, starting with the present technical deployment, one can easily think of more sophisticated possibilities. For example, the Smart Card could store the account balance in order to prevent financial losses due to a server crash. With regard to the maximization of an authentication mechanism's ease of use, the Smart Card could furthermore be used as a token to grant access to the virtual world without having to enter the account information manually and therefore increase convenience.

A hindering aspect of the current implementation is the need for a separate Smart Card reader. If the users where able to insert the Smart Card into the PC (e.g. via a USB token or Internet Smart Card) or into a trusted device they already own and control (e.g. a Smartphone or PDA), it is to be expected that their willingness to adopt the technology increases. A substantial discussion on user acceptance of information technology in general can be found in [24] and [25].

4.3 Managing Identity in Professional Online Networks: XING.com

4.3.1 Present Status: Virtual Identities for Professional Purposes

With over 5.7 million registered users and translations of the services into 16 different languages, *XING* is one of the largest global business-oriented social networks on the Internet [93]. A popular definition of social networks in general is provided by Boyd & Ellison [94]: "[…] web-based services that allow individuals to:

- Construct a public or semi-public profile within a bounded system,
- Articulate a list of other users with whom they share a connection, and
- View and traverse their list of connections and those made by others within the system."

XING in fact conforms to this definition, since it serves as a platform for managing contacts, establishing business relationships and visualizing existing interpersonal networks. In addition, members can join a variety of different interest groups, organize networking events and use the built-in marketplace to find or offer jobs, services, or real estate. The promised career-building potential is often cited as the main reason for new users to join web-based business communities [95].

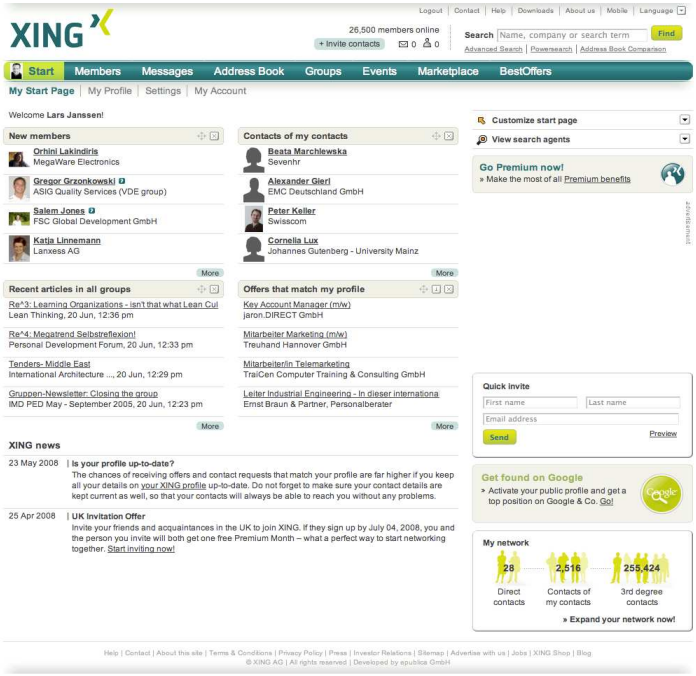


Figure 13: Personalized startpage of the XING network.

- **Roles**: Different roles based on the position a member holds, e.g. in a company, are currently not implemented in the *XING* network. However, the possibilities the network offers in this context are clearly visible. In a simple framework, users might for example take the role of an applicant for work, a researcher in search for thematic companions, or a recruiter for a company that is looking for employees. Regardless of the actual role, they all have one thing in common: the need for privacy regarding some aspects of their identity. Looking at the relationship between a candidate for a job and a human resources director of a company for example, both of them are likely to be interested in revealing as little information as possible and still being able to get in contact with each other. In a lot of cases, people are looking for a job while being employed at their current company. Hence, they tend to keep their present employer unmentioned. Recruiters, on the other hand, don't want to disclose the identity of the organization they are working for in the first step. The ePortfolio scenario in the following section 4.3.2 further describes such a situation.

- **Authentication mechanism**: *XING* deploys a commonly used login procedure with a username and password to authenticate users against the service. Communication between the server and the user's device is persistently encrypted using a 128-bit SSL connection.

- **Supporting technology**: Due to the lack of a strong authentication mechanism in the current version of *XING*, there are no technical requirements except for an up-to-date web browser. But, in order to support the earlier considerations regarding the introduction of different user roles into the community, a more complex, yet secure way of authentication would be necessary. Community members need an opportunity to define profiles/partial identities for different purposes they pursue within the business network. At this point, a secure device with a trusted component (e.g. a trusted mobile device in conjunction with a Smart Card, see Chapter 2) appears a feasible solution. Partial identities could be derived from a full profile defined within the trusted component, depending on which information a specific role demands.

- **Value proposition**: Operators of social networks are permanently looking for new possibilities to generate revenues by utilizing the user base of the respective community. In the last years, some community operators achieved that by simply selling all the data they acquired to third parties without consent of the respective users. Since this course of action is highly questionable, especially from a legal point of view, the focus is shifting towards more privacy-friendly business

models. Companies operating e.g. a social network site typically strive for a unique selling proposition that enables them to "stand out from the crowd". Enhanced privacy protection and secure exchange of personal information between different roles is a promising opportunity. However, it is unlikely to lead to economic success just by itself. Nevertheless, a secure identity management environment can act as an enabler for services otherwise not possible due to the lack of user acceptance. Studies on the various aspects of information privacy, especially the users' valuation thereof, can be found in [99], [100], [101], and [102]. Whereas users would normally use diverse portals for all aspects of their life, the support of various context-specific roles within a large and well-known community, in combination with convenient and secure authentication, seems attractive. *XING* already offers a premium membership for 5.95 € per month with 420,000 subscribers currently active. Among other things the premium membership includes the possibility to:

- Send personal and secure messages to other users,
- See which members have recently viewed the ownowner's profile,
- Perform advanced searches based on location, company or university,
- And receive discounts at *XING BestOffers*.

One could expect the number of subscribers to further increase with the implementation of identity management features and higher privacy and security standards as described above. Regardless of this opportunity, it should be noted that PrimeLife's approach is not the extension of a specific social network with security or privacy features, but rather to maintain an overall infrastructural perspective that can prove useful in different scenarios.

4.3.2 The Emerging Use Case of Employability Data Management

The above presented concept of professional online networks can be related to numerous technological solutions.[27] Generally, the need to join up sources of data, including legacy data, from Management Information Systems (MIS), ePortfolios and other sources, grows as the reliance on electronic records increases. This is not only to evidence and support the progression of the individual but also to provide services which are of value to institutions. Rather than accumulating and aggregating entire sets of information, individuals increasingly need to be able to reference and aggregate verified data from a variety of sources for specific purposes in a dynamic manner. Present and future employees need to be empowered to match authenticated distributed evidence to required skill sets and then share it as appropriate.

A scenario related to employability can be expected to be a suitable research demonstrator, because:

- Service composition is required and part of the composition is done at run-time.
- Data handling is required to deal with PII from different sources and potentially send them to other trust domains.
- There are clear privacy issues and fewer regulations than in e-health scenarios (see below).
- The scenario can be related to the blog story from Work Package 1.1 (i.e. prove that the applicant was active in a technical blog).

Here, we propose two situations/use cases within the area of employability. They could take place within a professional online network such as XING or could be used outside such environments:

In both cases, the user is an employee of a global organization, works in the Netherlands and has created an electronic portfolio (ePortfolio) that includes personal details such as name, address, age, sex, race, experience, qualifications, internal and external training, and references. External parties, who provide evidence of certification, are confirmed as trusted parties.

[27] Note: This scenario will need to be described even further in upcoming deliverables and prototypes in order to develop a full use case of the actual activities.

Case A: The employee decides to send an unsolicited job application to two sites, one in France and the other in the United Kingdom. The ePortfolio is submitted to the different sites and local Human Resource departments apply the relevant policies (i.e. a PAP-server side application of policies). Local policies are applied in order to expose only certain elements of the portfolio depending on local rules. For instance, "race" is a required element of a curriculum vitae in the U.K., whereas in France this is not the case. Local Human Resource departments would use a PDP and a PEP to respect the local policies. [28]

Case B: The employee decides to apply for two different jobs in two sites, the UK and France. One of the jobs is in management and the other in development. Therefore, two profiles from their ePortfolio are created and tailored to meet the needs of both offering types. An obligation (using EPAL[29] [67]) is imposed by the potential employee (sticky policy) to delete the application after 30 days and forbid forwarding details of the ePortfolio to the other sites in order to hide his global job intentions and keep the two applications separated.

4.4 Further Application Areas and Technology Details

The following two scenarios from the health care domain make the requirements for privacy policy supporting service composition more explicit:[30] First, a classical data handling scenario is sketched. Second, we describe more complex scenario. In addition to the above presented more compley ideas and concepts, focus is here laid on the actual process of exchanging single and rather simple data bundles to give the reader additional practical insight.

4.4.1 Data Flow in Travel Booking

This use-case describes a common data handling issue.
A user provides PII (e.g. E-mail address) to a Travel Booking composite service. This composite service may use the PII and hand it to other parties (Hotel Booking and Car Booking). Involved parties may get the PII for different purposes. The trust relationship between the user and the services may also differ.
The key points to address are:

- Composition of the policies (i.e. combining policies of Hotel Booking and Car Booking).
- Matching the policies and the user's preferences.

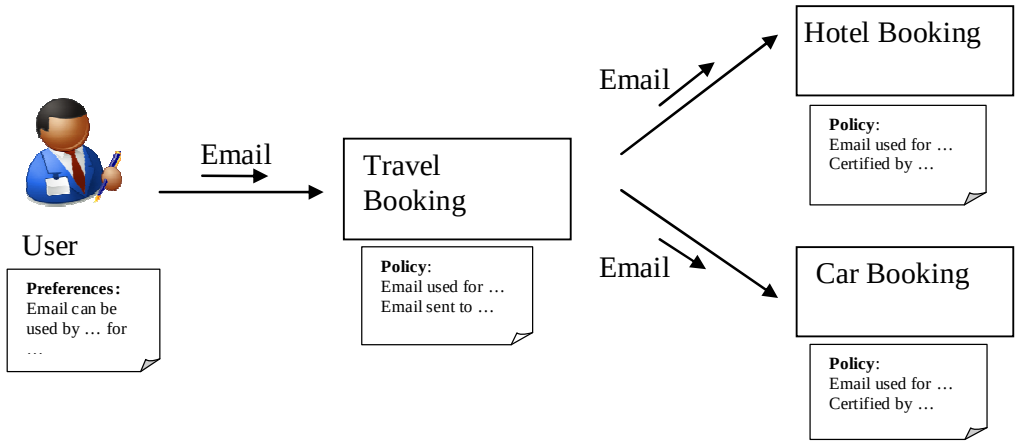


Figure 14: Data Flow in Booking Scenario.

[28] PAP = Policy Administration Point. This system entity creates a policy or policy set, i.e. a repository for policies. PAP writes policies and policy sets and makes them available to the PDP. These policies or policy sets represent the complete policy for a specified target.
PDP = Policy Decision Point. It analyzes policy information, takes the decision and relays it to the PEP.
PEP = Policy Enforcement Point, enforces the policy decisions taken by PDP.
[29] ELAP stands for Enterprise Privacy Authorization Language [67].
[30] The actual scenario(s) that will be addressed in WP6.3 and implemented as research demonstrator still have to be chosen and may not be related to the scenarios below. However, they will certainly be related to ePortfolio. As focus is laid on potential technological applications, a full description of the scenarios along the Authentication Scenario Description Scheme (see section 4.1) has not been included here.

4.4.2 Data Flow in Health Services

This second use-case describes a more complex data handling issue.

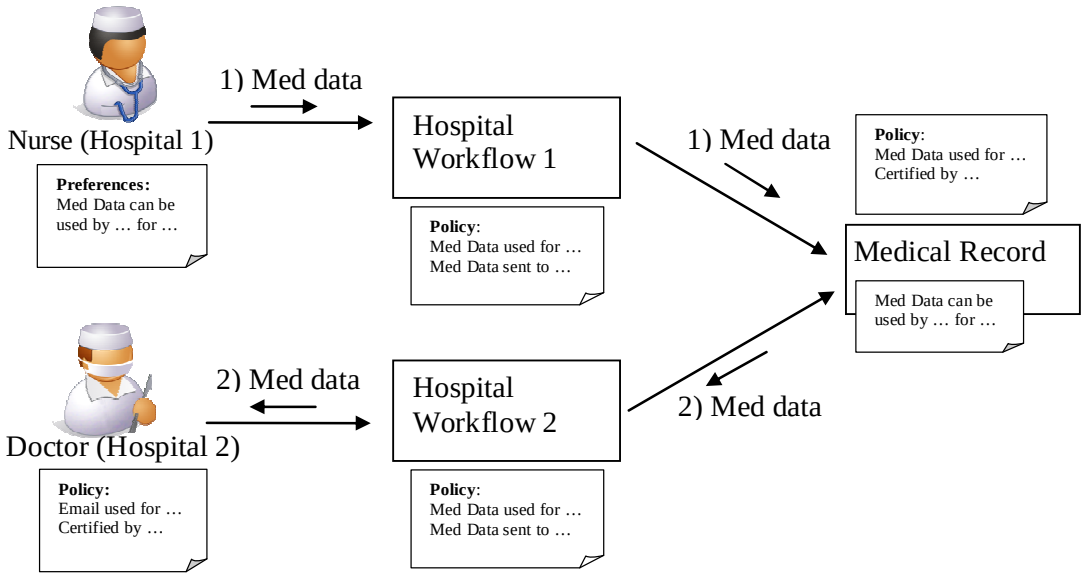


Figure 15: Data Flow in Medical Services.

A data source (e.g. nurse) provides PII to a composite service. This composite service provides those data to a data consumer (e.g. medical record), which stores them. Subsequently, another data consumer (e.g. doctor) accesses the medical data through a composite service. In this second phase, the medical record is a data source. The policy of the medical record has to be accepted by the nurse that provides data and the doctor has to fulfil this policy. This sample makes clear that:

- A party can act as data source and data consumer.
- Using the same language for capabilities (i.e. policies) and requirements (i.e. preferences) would simplify the matching between doctor's policy and medical record's data handling requirements. In other words, the border line between requirements (i.e. preferences) and capabilities (i.e. privacy policies) is not always clear.

4.4.3 Further Technology Applications

Some technologies (e.g. workflow, client side mash-up, etc.) are more appropriate than others to tackle specific scenarios (see Appendix 6 for more details on the state of the art concerning service composition). The following application examples could be implemented during the future course of this research when specific scenarios and use cases are elaborated in more market- and technology-related detail

- Client-side mash-up (e.g. presentation, data or logic/business mash-up): Has a direct impact on the trust model since the composition is run within the trust domain of the user.
- Service-side mash-up/workflows: More complex treatment of data with associated constraints can be envisioned.
- Composition of artefacts: Service composition can easily be extended to pervasive computing scenarios. More and more embedded systems (ranging from appliances such as video projectors to hidden sensors) will be visible as services. Modelling such composition may require specific types of constraints.
- Virtual Organization: Is directly related to cross-organisation and federated scenarios. Authentication and authorization are key issues.

- Online communities or virtual worlds: Could be bridged leading to interesting composition.
- Choreography: May be an interesting option to associate two different technologies in a common demonstrator, e.g. Windows WF Foundation and SAP NetWeaver.
- Service Composition with strong authentication (e.g. Smart Card): Composition of a service requiring strong authentication (e.g. medical record requiring electronic Identity Card). Taking into account non-functional services in the composition would be interesting as well.
- Non-functional service composition where security services are explicitly part of the composition has not been addressed in this document. This could lead to security- or privacy-oriented service composition.
- Data aggregations mash-up: Some composite service aim at aggregating and correlating data from different sources. Such compositions are directly threatening privacy and should be made compliant with data handling policies.

-

Chapter 5

Outlook and Conclusions

The findings presented here have outlined the available technologies for Secure Environments, Service Composition and Authentication Infrastructures. The means for enhancing trust in digitally processed and transferred content and for enhancing user privacy and overall security have been explained. Some of these goals can be achieved with given state of the art technologies. However, the current state of technology still lacks certain elements to enable the envisioned use cases. Therefore, the technology- as well as the market-side (i.e. the use cases) need further detailed research. Based on such further research, the next step in the direction of protecting personal or otherwise sensitive data can be envisioned and then taken.

Currently, a high level of security of environment can especially be achieved through mobile devices such as Smart Cards. Other approaches can be based on virtualization [39], secure bootstrapping [38], ARM Trust Zone [40], or a combination thereof. The isolation of the secure environment (or the Smart Card as secure Component in a wider service context) is a key advantage. Further, the often small size of the Smart Card-related systems can reduce the complexity of possible attacks. However, this small size may also result in limited computing power and memory capacity. Still though, Smart Cards can serve as an excellent starting point for all applications in which a higher level of security is necessary and the potential limitations of capacity do not hinder the implementation. This can be particularly promising if Smart Cards are combined with other secure environments in a mobile device [39 & 40].

Service Oriented Architectures can also assist in determining fine-grained elements of security-related services. Because SOAs define clear interfaces and their requirements are usually based on contractual agreements, they can be excellent complements to small-sized and mobile systems.

However, a challenge remains for future research: The orchestration or choreography of these services, especially in the combination of mobile devices and the internet. Here, the current technology status still calls for finding the right mix of languages to describe policies for the processing of specific data and linking this systemic approach to any legal requirements.

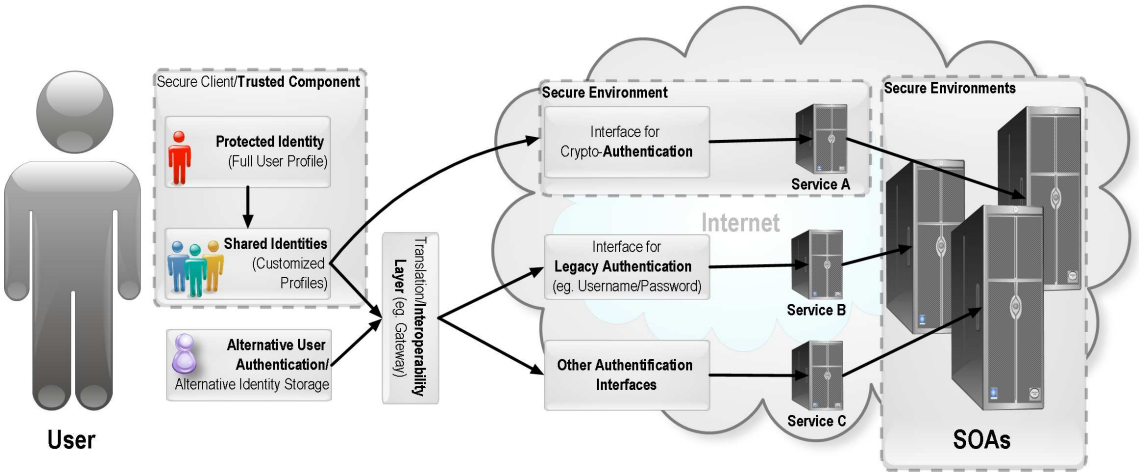


Figure 16: The full 'Security of Service' concept.

As a practical vision and as a blend of the technologies presented in this Work Package, a service provider could in the future be empowered to use the various credentials stored on a customer's mobile device. Thus, the service provider could trust the user's service requests as genuine and execute them efficiently. The user, in turn, could have control over his / her identity related data that is stored in the secure environment / secure component of the mobile device. The user could dynamically control the appearance of this data and thus tailor his / her identity to the different services.[31]
To turn this vision into reality, the future research activities will need to go into further detail regarding the actual execution of security and privacy in service composition.

[31] For a high level of privacy protection, it may be in the interest of the user to enforce the rules he has agreed upon for handling his data, also when he has released this data. For such cases, the services he uses also need to be secured. However, the development and detailed description of such systems is beyond the scope of this report and needs to be addressed in future research.

References

[1] Abadi, M., Bharat, K. and Marais, J. (1997) System and method for generating unique passwords. United States Patent 6141760.

[2] Adams, A., Sasse, M. A. and Lunt, P. (1997) Making Passwords Secure and Usable. In: Proceedings of HCI on People and Computers XII, Bristol, UK, Springer, 1-19.

[3] Ball, P. (2001) Hacktivism and Human Rights: Using Technology to Raise the Bar. In: Panel Discussion, DEF CON 9, Las Vegas, USA.

[4] Brown, B. J. and Callis, K. (2004) Computer Password Choice and Personality Traits Among College Students, Southeast Missouri State University, Cape Girardeau, Missouri, USA.

[5] DeCock, D., Wouters, K. and Preneel, B. (2004) Introduction to the Belgian EID Card. In: S. K. Katsikas, S. Gritzalis und J. Lopez (Eds.), Public Key Infrastructures. Berlin, Springer, 1-13.

[6] Economides, N. (1996) The Economics of networks. In: International Journal of Industrial Organization, 14, 673-699.

[7] Fraunhofer SIT (2006) Der PasswordSitter, White Paper, http://www.passwordsitter.de/fileadmin/downloads/pws-whitepaper-may-2006-de.pdf, 28.07.2008.

[8] Gabber, E., Gibbons, P., Matias, Y. and Mayer, A. (1997) How to Make Personalized Web Browsing Simple, Secure and Anonymous. In: Proceedings of the First International Conference on Financial Cryptography, Anguilla, British West Indies, Springer, 17-32.

[9] Halderman, J. A., Waters, B. and Felten, E. W. (2005) A convenient method for securely managing passwords. In: WWW '05: Proceedings of the 14th international conference on World Wide Web, Chiba, Japan, ACM Press, 471-479.

[10] Hvarre, J. (2004) Electronic signatures in Denmark: free for all citizens. In: e-Signature Law Journal, 1, 1, 12-17.

[11] Ives, B., Walsh, K. and Schneider, H. (2004) The Domino Effect of Password Reuse. In: Communications of the ACM, 4, 47, 75-78.

[12] Karp, A.H. (2003) Site-Specific Passwords. In: Technical Report, HP Laboratories, Palo Alto.

[13] Lopez, J., Opplinger, R. and Pernul, G. (2005) Why Have Public Key Infrastructures Failed So Far? In: Internet Research, 15, 5, 544 - 556.

[14] RSA Security (2005) RSA Security Survey Reveals Multiple Passwords Creating Security Risks and End User Frustration. In: Press Release, http://www.rsasecurity.com/press_release.asp?doc_id=6095. 28.07.2008.

[15] Rivest, R. L., Shamir, A. and Adleman, L. (1978) A Method for Obtaining Digital Signatures and Public Key Cryptosystems. In: Communications of the ACM, 21, 2, 120-126.

[16] Ross, B., Jackson, C., Miyake, N., Boneh, D. and Mitchell, J. C. (2005) Stronger Password Authentication Using Browser Extensions. In: Proceedings of the 14th Usenix Security Symposium, Baltimore, Maryland.

[17] Roßnagel, H., Zibuschka, J. (2007) Integrating Qualified Electronic Signatures with Password Legacy Systems. In: Digital Evidence Journal, 4, 1, 1-10.

[18] Roßnagel, H. (2007) Mobile Qualifizierte Elektronische Signaturen: Analyse der Hemmnisfaktoren und Gestaltungsvorschläge zur Einführung. Unpublished doctoral dissertation, Department of Business Administration and Economics, Johann Wolfgang Goethe University, Frankfurt am Main.

[19] Savard, J. (1999) Keystream Base Conversion and Random Bit Unbiasing, A Cryptographic Compendium.

[20] Secure Information Technology Center – Austria (2006) The Austrian Citizen Card, http://alt.buergerkarte.at/index_en.html, 28.02.2008.

[21] Weitzel, T. (2003) A Network ROI. In: Proceedings of the MISQ Academic Workshop on ICT standardization, ICIS 2003, Seattle WA, USA.

[22] Witty, R. J., Allan, A., Enck, J., Hirst, C., Runyon, B., Wagner, R., Perkins, E. L., Pescatore, J. and Wheatman, V. (2005) Hype Cycle for Identity and Access Management Technologies 2005, Gartner.

[23] Kosta E., Zibuschka J., Scherner T. & Dumortier J. Privacy issues in location based services - Legal considerations on privacy-enhancing Location Based Services using PRIME technology. In: Computer Law & Security Report (2008), 24, 2, 139-146.

[24] Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. In: MIS Quarterly, 13, 3, 319-340.

[25] Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User Acceptance of Information Technology: Toward a Unified View. In: MIS Quarterly, 27, 3, 425-478.

[26] Hinz, W., Spitz, S. (2006) Zur Sicherheit von neuen Chipkarten-betriebssysteme. In: Horster, P. (2006, Ed.) D-A-CH Konferenzband.

[27] Hansmann, U., Nicklous, M., Schäck, T., Seliger, F. (1999) Smartcard Application Development Using Java. Berlin, Springer.

[28] InspireD (2005) D6 Communication Architecture Definition (Draft), forthcoming on http://www.inspiredproject.com, 25.08.2008.

[29] InspireD: D7 System Architecture Definition (Draft), forthcoming on http://www.inspiredproject.com, 25.08.2008.

[30] ISO/IEC JTC1 (2003) ISO/IEC 7816: Part 1-9. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_tc_browse.htm?commid=45020, 28.07.2008.

[31] Rankl, W.; Effing, W. (2002) Handbuch der Chipkarten, 4th edition, Munich, Carl Hanser Verlag.

[32] Spitz, S. (2002) Integration von Chipkarten in elektronische Geschäftsprozesse. In: Horster, P. (2002) Konferenzband Elektronische Geschäftsprozesse.

[33] Swoboda, J., Spitz, S., Pramateftakis, M. (2008) Kryptographie und IT Sicherheit, Wiesbaden, Vieweg-Teubner, ISBN 978-3-8348-0248-4.

[34] OASIS Web Services Security (2004) SOAP Message Security 1.0. http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf, 28.07.2008.

[35] W3C. (2003) XML Encryption Working Group. http://www.w3.org/Encryption/, 28.07.2008.

[36] W3C. (2003) XML Signature Working Group. http://www.w3.org/Signature/, 28.07.2008.

[37] W3C. (1999) WAP XML Binary Content Format, http://www.w3.org/1999/06/NOTE-wbxml-19990624/, 28.07.2008.

[38] TCG (2008) TPM Spezifikationen der Trusted Computing Group [Specifications of the Trusted Computing Group], https://www.trustedcomputinggroup.org/specs/TPM, 28.07.2008.

[39] European Multilaterally Secure Computing Base (EMSCB) (2008) Turaja Technology, http://www.emscb.de/content/pages/About-Turaya-de.htm, 28.07.2008.

[40] ARM (2008) ARM processor architecture, security extension ARM Trust Zone technology, http://www.arm.com/products/esd/trustzone_home.html, 28.07.2008.

[41] Clair, L. S., Johansen, L., Enck, W., Pirretti, M., Traynor, P., McDaniel, P. and Jaeger, T. (2006) Password Exhaustion: Predicting the End of Password Usefulness. In: Proceedings of 2nd International Conference on Information Systems Security (ICISS), 37-55.

[42] Neumann, P. G. (1994) Risks of passwords. In: Communications of the ACM, 37, 4, 126.

[43] Rakesh A.; Jerry K.; Ramakrishnan S.; Yirong X.. (2003) An XPathbased Preference Language for P3P. IBM, http://portal.acm.org/citation.cfm?id=775152.775241&coll=ACM&dl=ACM&type=series&idx=SERIES968&part=series&WantType=Proceedings&title=WWW&CFID=6868422&CFTOKEN=36504156, 25.08.2008.

[44] Orchestr8 (2008) AlchemyPoint. http://www.orch8.net/ap/index.html, 28.07.2008.

[45] Anonymous (2008) Understanding Windows Live delegated authentication. White paper, Microsoft Corporation, http://msdn2.microsoft.com/en-us/library/cc287613.aspx, 28.07.2008.

[46] Liberty Alliance (2003) Liberty architecture framework for supporting privacy preference expression languages (ppels). In: Technical report, Liberty Alliance Project, Nov. 2003.

[47] W3C (2002). A P3P Preference Exchange Language 1.0 (APPEL1.0). W3C Working Draft, April 2002.

[48] Microsoft BizTalk Server. http://www.microsoft.com/biztalk/en/us/default.aspx, 28.07.2008.

[49] Becker, M. Y., Gordon, A.D. and Fournet, C. (2006) Secpal: Design and semantics of a decentralized authorization language. In: Technical Report MSR-TR-2006-120, Microsoft Research, Sept. 2006.

[50] Adobe Systems Inc., BEA Systems Inc., International Business Machines Corporation, Oracle Inc., Active Endpoints Inc. and SAP AG (2007). WS-BPEL Extension for People (BPEL4People). 1.0 edition, June 2007.
[51] W3C (2005) Web Services Choreography Description Language - W3C Candidate Recommendation. In: Web Services Choreography Working Group, 1.0 edition, Nov. 2005. http://www.w3.org/TR/ws-cdl-10/, 28.07.2008.
[52] Cantor, S., Kemp, J., Philpott, R. and Eve Maler (2005) Assertions and protocols for the OASIS security assertion markup language (SAML). In: OASIS Standard Document identifier saml-core-2.0-os, v2.0., OASIS, Mar. 2005.
[53] Canovas, O., Lopez, G. and Gomez-Skarmeta, A.F. (2004) A credential conversion service for saml-based scenarios. In: Lecture Notes in Computer Science (LNCS) (2004) Public Key Infrastructure, 3093, Berlin, Springer, 297 – 305.
[54] Dapper (2008) http://www.dapper.net/, 28.07.2008.
[55] Denodo Technology SL (2008) http://www.denodo.com/english/index.html, 28.07.2008.
[56] Erl, T. (2005) Service-Oriented Architecture, Concepts, Technology and Design. In: Prentice Hall Professional Technical Reference, July 2005. ISBN 0-13-185858-0.
[57] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A. and Stewart, L. (1999) HTTP Authentication: Basic and Digest Access Authentication. In: RFC, 2617, June 1999.
[58] Flickr (2008) http://www.flickr.com/, 28.07.2008.
[59] Freudenthal, E., Pesin, T., Port, L., Keenan, E. and Karamcheti, V. (2002) dRBAC: Distributed role-based access control for dynamic coalition environments. In: Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS '02), Washington, DC, USA, 411 - 420.
[60] Google Mashup Editor. http://code.google.com/gme/, 28.07.2008
[61] iGoogle. http://www.google.com/ig, 28.07.2008.
[62] Lopez, G., Canovas, O., Gomez-Skarmeta, A.F., Otenko, S. and Chadwick, D.W. (2005) Public Key Infrastructure. In: Lecture Notes in Computer Science (LNCS), Vol. 3545, Berlin, Springer, 55 – 72, ISBN 978-3-540-28062-0.
[63] Lang, B., Foster, I., Siebenlist, F., Ananthakrishnan, R. and Freeman, T. (2006) A multipolicy authorization framework for grid security. In: Fifth IEEE International Symposium on Network Computing and Applications, IEEE Press, 269 – 272.
[64] Microsoft Windows Live Services. http://my.live.com/, 28.07.2008
[65] Lotus mashup (2008),
http://www-306.ibm.com/software/lotus/products/mashups/, 28.07.2008
[66] Mukkamala, R., Atluri, V., Warner, J. and Abbadasari, R. (2006) A distributed coalition service registry for ad-hoc dynamic coalitions: A service-oriented approach. In: Damiani, E. and Liu, P. (2006) [98], 209-223.
[67] Powers, C., Schunter, M. (2003) The Enterprise Privacy Authorization Language (EPAL 1.1). IBM, December 2003.
[68] Moses, T. (2005) OASIS eXtensible Access Control Markup Language (XACML) Version 2.0. OASIS Standard oasis-access control-xacml-2.0-core-spec-os, OASIS, February 2005.
[69] My Yahoo!. http://in.my.yahoo.com/, 28.07.2008
[70] Microsoft (2008) NET 3.5 Framework. http://www.microsoft.com/downloads/details.aspx?FamilyID=333325fd-ae52-4e35-b531-508d977d32a6&DisplayLang=en, 28.07.2008.
[71] OASIS (2007) Web Services Business Process Execution Language. OASIS Web Services Business Process Execution Language (WSBPEL) TC, 2.0 edition, April 2007.
[72] OAuth Core Workgroup (2007) OAuth Core 1.0. Technical report, 2007.
[73] Openkapow (2008) http://openkapow.com/, 28.07.2008.
[74] W3C. Pling - w3c policy languages interest group.
[75] JackBe Corp. (2008) Presto enterprise mash-up. http://www.jackbe.com/, 28.07.2008.
[76] Microsoft (2008) Popfly. http://www.popfly.com/, 28.07.2008.
[77] Pli08 Working Group (2006) The Platform for Privacy Preferences 1.1 (P3P1.1) Specification, Pli08 Working Group Note, 13.11.2006.

[78] Robinson, P., Kerschbaum, F. and Schaad, A. (2006) From business process choreography to authorization policies. In: Damiani, E. and Liu, P. [98], 297 - 309.
[79] SAP (2008) Business ByDesign, http://www.sap.com/solutions/sme/businessbydesign/overview/index.epx, 28.07.2008.
[80] Serena Software (2008) http://www.serena.com/mashups/index.html , 28.07.2008.
[81] She, W., Thuraisingham, B. and Yen I.-L. (2007) Delegation-based security model for web services. In: Proceedings of 10th IEEE High Assurance Systems Engineering Symposium (HASE '07), Nov. 2007, 82 – 91, ISBN: 978-0-7695-3043-7.
[82] IBM (2008) WebSphere http://www-306.ibm.com/software/webservers/smash/, 28.07.2008
[83] Wimmer, M., Kemper, A., Rits, M. and Lotz, V. (2006). Consolidating the access control of composite applications and workflows. In: Damiani, E. and Liu, P. [21], 44 - 59.
[84] Active Endpoints Inc.; Adobe Systems Inc.; BEA Systems Inc.; International Business Machines Corporation; Oracle Inc. and SAP AG (2007) Web Services Human Task (WS-HumanTask), 1.0 edition, June 2007.
[85] WSO2 Mash-up Server (2008) http://wso2.com/products/mashup/, 28.07.2008
[86] Windows Workflow Foundation (2008) http://netfx3.com/content/WFHome.aspx, 28.07.2008.
[87] Yahoo Pipes (2008) http://pipes.yahoo.com/pipes/, 28.07.2008.
[88] Yu, W.D. (2006) An intelligent access control for web services based on service oriented architecture platform. In: Proceedings of the The Fourth IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems, and the Second International Workshop on Collaborative Computing, Integration, and Assurance (SEUS-WCCIA'06), 190 – 198, ISBN:0-7695-2560-1.
[89] Intel Mash Maker (2008) http://mashmaker.intel.com/web/index.php, 28.07.2008.
[90] Ferraiolo, D. and Kuhn, R. (1992) Role-Based Access Control. In: Proceedings of the 15th National Computer Security Conference, Baltimore, MD, 554-563.
[91] MindArk (2008) http://www.mindark.se/, 28.07.2008.
[92] Linden Research (2008) Second Life - Economic Statistics. http://secondlife.com/whatis/economy_stats.php, 25.02.2008.
[93] XING. (2008) http://www.xing.com/, 18.06.2008.
[94] Boyd, D. and Ellison, N. (2007) Social Network Sites: Definition, History, and Scholarship. In: Journal of Computer-Mediated Communication , 13, 1, 210-230.
[95] Andrews, D. C. (2002). Audience-Specific Online Community Design. In: Communications of the ACM , 45, 4, 64-68.
[96] See http://www.w3.org/P3P/
[97] Foster, I., Kesselman, C. and Tuecke, S. (2001) The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications, 15, 3, 200 - 222.
[98] Damiani, E. and Liu, P. (2006) Data and Applications Security XX. In: Proceedings of the 20th Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Sophia Antipolis, France, ISBN 978-3-540-36796-3.
[99] Olson, J. S., Grudin, J. and Horvitz, E. (2005) A Study of Preferences for Sharing and Privacy. In: CHI ’05 Extended Abstracts on Human Factors in Computing Systems, Portland, OR, USA, 1985-1988.
[100] Gross, R., Acquisti, A. and Heinz III., H. J. (2005) Information Revelation and Privacy in Online Social Networks. In: Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, Alexandria, VA, USA, 71-80.
[101] Earp, J. B., Poindexter, J. C. and Baumer, D. L. (2004) Modeling Privacy Values with Experimental Economics. In: Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society, Washington D.C., USA, 25-25.
[102] Acquisti, A. (2004) Privacy in Electronic Commerce and the Economics of Immediate Gratification. In: Proceedings of the 5th ACM Conference on Electronic Commerce, New York, NY, USA, 21-29.
[103] ITU-T Recommendation X.509 (2005) Information Technology - Open Systems Interconnection - The Directory: Authentication Framework, 08/05.

[104] Adams, C. and Farrell, S. (1999) "Internet X.509 Public Key Infrastructure: Certificate Management Protocols", RFC 2510, March 1999.
[105] Schneier, B. (1996) Applied Cryptography, 2nd edition, John Wiley & Sons
[106] International Organization for Standardization (1989) ISO 7498/2: Security Architecture, http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=14256, 25.08.2008.
[107] http://www.sc17.com/index.cfm?pageTitle=Structure&DepartmentID=6&HAT=78301601141219764913870-04350013-493032&ts=04350037-998319, 24.08.2008.
[108] https://www.trustedcomputinggroup.org/specs/mobilephone/CTIA_final_MPWG_FAQ_sept_8_2006.pdf, 24.08.2008.
[109] http://msdn.microsoft.com/en-us/library/ms733083.aspx , 25.08.2008.
[110] http://www.zurich.ibm.com/security/idemix/, 25.08.2008.
[111] http://openid.net/what/, 25.08.2008.
[112] http://msdn.microsoft.com/en-us/library/cc287610.aspx , 25.08.2008.

Appendix A

State of the Art on Security in Service Composition

A.1 Service Composition

A.1.1 Orchestration

- **WS-BPEL** [71] breaks down workflow logic into a series of predefined primitive activities. Basic activities (receive, invoke, reply, throw, wait) represent fundamental workflow actions which can be assembled using the logic supplied in structured activities (sequence (one activity after the other), switch, while, flow (activities in parallel), pick).

- The **WS-HumanTask** [84] specification introduces the definition of human tasks and notifications, including their properties, behaviour, and a set of operations used to manipulate human tasks. A coordination protocol is introduced in order to control autonomy and life cycle of service-enabled human tasks in an interoperable manner.

- The **BPEL4People** [50] specification introduces a WS-BPEL extension to address human interactions in WS-BPEL as a first-class citizen. It defines a new type of basic activity which uses human tasks as an implementation and allows specifying tasks local to a process or use tasks defined outside of the process definition. This extension is based on the WS-HumanTask specification.

- Developers often create composite applications directly in code (Java, C#, etc.). In this common approach, they code the entire orchestration into a set of classes. Frameworks such as the **.NET Framework 3.5** [70] provide developers with support for creating such applications. **Windows Workflow Foundation** [86] is a part of the .NET Framework 3.5 that enables developers to create workflow enabled applications. Another tool is **Microsoft BizTalk Server** [48] that is used to connect systems both inside and across organizations. This includes exchanging data and orchestrating business processes which require multiple systems. BizTalk Server is a solution for SOA and Business Process Management.

A.1.2 Choreography

- **WS-CDL** [51], the Web Service Choreography Description Language, is a specification that attempts to organize information exchange between multiple organizations (public collaboration). Web services assume a number of predefined roles (e.g. bound to WSDL definitions) and are grouped as participants. Every action in choreography corresponds to series of message exchanges between two services within a relationship (2 roles). Channels define the characteristics of the message exchange between two specific roles (channel information can be passed around in messages, and this enables dynamic discovery). Interactions encapsulate the actual logic behind a message exchange.

A.1.3 Mash-up

We distinguish client-side mash-ups that are generally executed in a web browser and service-side mash-ups that are executed on a server.
Client-side mash-ups can be customizable portals – e.g. Windows Live Personalized Experience [64], iGoogle [61], or My Yahoo! [69] – or rich internet applications that allow end users to build and share mash-ups, gadgets, and Web pages – e.g. Microsoft Popfly [76], Google Mash-up Editor [60], or ByDesign [79].
Service-side mash-ups allow users to combine feeds, web services, and/or even web pages to create data mash-ups. Examples are: Yahoo Pipes [87], openkapow [73], Dapper [54], JackBe Corp., Presto enterprise mash-up [75], enterprise mash-up software from Denodo Technologies SL [55], Serena Software [80], Mash Maker (Intel) [89], Lotus Mash-up [65], WebSphere sMash (IBM) [82], AlchemyPoint (Orchestr8) [44], WSO2 Mash-up Server [85], etc.

A.1.4 Taxonomy of Technologies for Service Composition

Technologies that support creation of composite services differ in the execution environment. Some of them support execution on the client side, i.e. in the web browser, while others execute the composite service on the server side. We can also imagine a hybrid form of execution where parts of the composition are run on the server side and others on the client side. At the same time, access to the composite service can be private (i.e. only the owner of the composite service can execute it), protected (i.e. behind access control), or public (i.e. publicly available to all end users).
Alongside these efforts, we also want to look at how the protection of the code behind the composite service is done. The code can be private, where only the creator of the composite service can look at it or reuse it. Further, the code can be protected or publicly available for reuse. It is interesting to see how different technologies treat the relation between code and access to the composite service.
Technologies for service composition may support authentication to services. The services can be hosted in other trust domains than the composition itself. If there is no such support, the compositions can only be executed within a single trust domain. For multiple trust domains, the support for authentication can be either static, e.g. preconfigured or based on previously established secrets, or dynamic, in the form of SSO (single sign on).

A.2 Security in Service Composition

Today's work on access control for web-services can be distinguished in four groups. The first group focuses on access control for individual web services and strives to come up with policy languages, improvements in policy decisions, or applies existing access control models to service-oriented architectures. The second group concentrates on access control for composite services in intra-organization applications. Hence, these publications explicitly or implicitly deal with scenarios where all services are inside a single trust domain. The third group of publications considers composed services distributed over multiple trust domains, but makes the implicit assumption or explicit proposal of a common authorization mechanism to circumvent the interoperability issue. The last group summarizes prior art that considers abstract management of access rights.

A.2.1 Authorization Policy Languages

- XACML [68] stands for eXtensible Access Control Markup Language. It is a declarative access control policy language implemented in XML and a processing model, describing how to interpret the policies. According to the charter of the OASIS technical committee responsible for XACML it allows “the use of arbitrary attributes in policies, role-based access control, security labels, time/date-based policies, indexable policies, 'deny' policies, and dynamic policies". XACML has been widely adopted both by industry and academics. Many publications [81, 62, 88, 63], which we will discuss below, use XACML as basis of their work, build extensions on top of it, or provide interoperability.
- SecPAL [49] is a declarative security policy language developed to meet the access control requirements of large-scale Grid Computing Environments. It is a declarative, logic-based, language that builds on a large body of work showing the value of such languages for flexibly expressing security policies. It was designed to be comprehensive and provides a uniform mechanism for expressing trust relationships, authorization policies, delegation policies, identity and attribute assertions, capability assertions, revocations, and audit requirements. This provides tangible benefits by making the system understandable and analyzable. It also improves security assurance by avoiding, or at least severely curtailing, the need for semantic translation and reconciliation between disparate security technologies.
- The Security Assertion Markup Language (SAML) [52] is an XML standard for exchanging authentication and authorization information between security domains. The standard is specified by OASIS' Security Services Technical Committee, latest official version is 2.0 which was adopted in March 2005. It defines SAML assertions and specifies a protocol for exchanging these assertions between an identity provider (producer of assertions) and a service (consumer of assertions). The service provider can then make security decisions for access control based on these assertions. Both parties have to trust the identity provider that is issuing the assertions, so SAML is useful in intra-organizational applications. Typically, SAML is used to realize single sign-on in web browser applications, and is well adopted by many products in this domain.

A.2.2 Access Control for Service Compositions within a Single Trust Domain

- Robinson et al. [78] present an approach to “automatically derive the minimal authorizations required for collaboration, as well as enable and disable the authorizations in a just-in-time manner that matches the control flow described in the choreography". They use it for “runtime management of authorization policies" of ad-hoc combined web-services.
- The approach presented by Mukkamala et al. [66] supports real ad-hoc collaboration between services. The “dynamic coalition-based access control (DCBAC) model facilitates the formation of dynamic coalitions through the use of a registry service". A central registry, which is also responsible for authorization decisions, is used as trusted third party, which makes the solution more suitable for single domain service compositions. A service registering itself at the registry may upload its individual service policy, e.g. expressed with WS-Policy.
- Wimmer et al. [83] also support our reasoning, that “when integrating autonomous sub-activities into workflows, security dependencies must be considered" and illustrate this with an e-health scenario. They propose a formal model that allows the computation of a summarized policy from a set of individual policies. It computes the least required privileges for the execution of the composite application.

A.2.3 Access Control for Service Compositions Across Different Trust Domains

- The position of She et al. [81] was already cited in Sect. A.2.1. They recognize that access control for service composition across multiple trust domains is not solved yet and come to the conclusion that delegation of access rights will become an important mechanism for service composition. Their approach is a “delegation-based security model [which] provides the framework and defines the key processes to secure the web services through issuing, sending, confirming, negotiating, and revoking the delegation tokens." This in fact means that all web services participating in this composition have to agree on a single token-based authorization mechanism, in She's case it is an extension of XACML policies. We think that this is not practicable and that a delegation solution has to integrate a large diversity of authorization mechanisms.
- An interesting case study how to integrate web services which are protected by XACML, SAML, and PERMIS that span across multiple trust domains is presented by Lopez et al. [62]. They base their integration on a previously published Credential Conversion System [53]. The approach makes use of role-based access control, which in turn “leads to the definition of loosely coupled multi-domain environments, where a predefined set of role mappings to mediate inter-domain accesses is defined." In other words, the solution requires a certain level of administration which makes it not suitable for fully ad-hoc mash-ups, which we are aiming at. However, Lopez et al. follow the same rationale we give in this deliverable: “standards for authorization systems are less widely adopted and accepted, and tend to work only within homogeneous systems".
- Freudenthal et al. [59] propose a model for distributed role-based access control (dRBAC) in “coalition environments". The model features a third-party delegation operation and allows a speaks-for metaphor. The credentials are stored in distributed credential stores at each participating party. These stores are called wallets and they contain “a collection of delegations". Freudenthal et al. propose a distributed network of wallets where each entity supports three basic operations. “An issuer of new delegations can post these delegations in a wallet so they can be located and used by others. [. . .] A trust-sensitive system resource can query a wallet for proofs authorizing whether a requested access is permitted. [. . .] Once a proof is returned, wallets provide functionality to allow continuous monitoring of its validity over the lifetime of the interaction." The whole approach is similar to the SecPAL policy language [49], which addresses the same question by means of a formal language whereas dRBAC utilizes an infrastructure.
- OAuth [72] is an open source protocol that allows users to delegate access to their resource to a delegatee. It explicitly covers the situation that the consumer is an application as well. The protocol keeps the delegator's identity and credentials confidential, which in fact allows anonymous delegation. OAuth is based on plain HTTP and extends HTTP Authentication [57]. The specification was published in December 2007 by the OAuth Working Group which is not yet part of a formal standardization body. The approach addresses the same problem that we do, but OAuth is again a proprietary mechanism.

A.2.4 Abstract Management of Access Rights

- Yu [88] proposes a “reusable access control layer, which is separate from the web services themselves. All web service requests pass through the authorization layer. An authorization decision is made and passed onto the web service." The goal of this paper is more to have a central policy decision point which additionally features an inference engine for policy decision. Although the authors touch extensibility for other access control mechanisms, the nature of a central decision point prevents this solution from being used in ad hoc service compositions.
- Lang et al. [63] present an approach for “multipolicy authorization" that allows to use XACML and SAML in Globus Toolkit release 4, a middleware for computing Grids [97]. They abstracted a policy decision point (PDP) with an object-oriented interface, which supports an abstract operation canAccess(). “Each specific policy is a subclass of the PDP abstraction, which implements the common interface inherited from PDP with its own policy and evaluation mechanism.".
- Windows Live ID Delegated Authentication [45] is “a way to permit access to personal information, but with more precise control over access and usage permissions than the [. . .] generally bad practice of handing over your account credentials to another Web site." Every service that wants to take advantage of this delegation technology registers one or more delegation

offers at a central registry. They describe the access that a delegatee can request at the service. The delegatee then asks the global delegation service for a consent token for an offer. If the central delegation service does not know a consent decision for this tuple of delegatee and offer yet, it requests a consent decision from the delegator. The decision is cached at the central delegation service and the delegatee gets back a consent token which includes a delegation token. Afterwards the delegatee can act on the delegator's behalf by presenting the delegation token to this specific service and get access. This approach fully supports our point of combining services from multiple trust domains. An interesting characteristic is to split the delegation in two phases, the consent phase and the delegation phase, which allows for the delegator only to be online for consenting. However, compared with the solution presented in Chapter 3 of this paper, this approach has two drawbacks. First, it requires a central instance which could potentially be a problem in terms of performance and denial of service. Second, the delegator and the delegatee are solely identified by their Windows Live ID accounts which implicitly introduces Windows Live ID as a trusted third party and as the sole authentication systems.

A.3 Privacy in Service Composition

State of the art regarding privacy in service composition will be done in Activity 5 and is thus not part of this deliverable. This upcoming work will focus on shortcomings of existing privacy policy and preference languages regarding service composition. Privacy Policy and Preference Languages (P3P [77], APPEL [47], XPref [43], PLING [74], EPAL [67], Liberty Alliance’s PPELs [46], PRIME Privacy Policy, ...), Authorization Languages that could be extended for privacy (e.g. XACML, SecPAL), and other related effort such as WS-Federations will be studied.