Secure Hardware-Based Public Cloud Storage

Bernd Zwattendorfer[1], Bojan Suzic[2], Peter Teufl[2], Andreas Derler[3]

[1]E-Government Innovationszentrum (EGIZ)
bernd.zwattendorfer@egiz.gv.at

[2]A-SIT – Secure Information Technology Center – Austria
{bojan.suzic, peter.teufl}@a-sit.at

[3]Graz University of Technology
andreas.derler@student.tugraz.at

Abstract: The storage of data on remote systems such as the public cloud opens new challenges in the field of data protection and security of the stored files. One possible solution for meeting these challenges is the encryption of the data at the local device, e.g. desktop, tablet, or smartphone, prior to the data transfer to the remote cloud-based storage. However, this approach bears additional challenges itself, such as secure encryption key management or secure and effective sharing of data in user groups. Including an additional encryption layer and security checks may additionally affect the system's usability, as higher security requirements and a group sharing workflow increase general overhead through the complete organization of processes. To overcome such issues, we propose a solution which is based on highly secure and attack-resistant hardware-based encryption applied through the use of the Austrian citizen card public key infrastructure. As the citizen card infrastructure is already deployed and available to a wide population, the service overhead and additional requirements of our proposed solution are lower in comparison to other approaches, while at the same time synergistic and networking effects of the deployed infrastructure facilitate its usage and further potentials.

1 Introduction

The quantity of digital information increases steadily as businesses improve processing and managing of information by digitizing and structuring them. Additionally, the amount of data stored by private users is boosted by high quality multimedia files and constantly declining storage prices. The way of accessing data ought to be independent of location and device, especially with the rising popularity and broader usage of mobile devices such as smartphones and tablets. These factors contributed to increased demand for storage capabilities e.g. for archiving or backup purposes. From that point, many subjects identified public cloud storage services as adequate or optimal means to lower costs and increase service flexibility and potential by outsourcing data storage and

providing file synchronization across multiple clients. Popular examples of such public cloud storage services are e.g. DropBox[1] or Google Drive[2].

While insensitive information and data can simply be stored on such public cloud providers, security and confidentiality plays an inevitable role if sensitive data needs to be stored in the cloud. Most cloud providers cannot easily fulfill such requirements, as the providers usually are able to inspect the stored data. Even if the cloud provider encrypts the data and stores it in encrypted format, the provider is always in possession of the decryption key.

To still be able to store sensitive data securely and confidentially in the cloud, some cloud providers offer solutions where data is encrypted on client-side prior to its transfer to the cloud. We introduce such solutions briefly in Section 2. However, most of those solutions have the drawback that the encryption and decryption process relies on software-based keys, which are stored on the respective client device and under some conditions could be accessible by unauthorized parties. To bypass security issues raised with that approach, we propose a solution which uses a hardware-based key pair kept on a smart card to protect data stored in the cloud. Our solution therefore relies on the Austrian citizen card, which represents the official eID in Austria [HKR+08]. The usage of the Austrian citizen card has the advantage that it is based on a solid and independent Public-Key-Infrastructure (PKI). Hence, data can be practically encrypted for each Austrian citizen and securely stored and shared in the cloud. In this paper, we present the implementation of this approach and compare it with existing solutions.

2 Related Work

As importance of security and privacy concerning cloud storage services increased, several designs enhancing these properties have been proposed. In this section, we firstly introduce two different designs for cloud storage, namely “cloud storage services” and “encryption middleware”. Secondly, we describe related work in the area of encryption middleware designs, as our proposed solution also fits into this design approach. Finally, the related work will also serve as a basis for our evaluation in Section 5.

Cloud storage services usually consist of a user-friendly client application and server-side software to store data. Some of these services also provide a web interface and a service API. The aim of cloud storage services is to provide cost-effective solutions for users to store and backup their data remotely, which should be easily accessed by different clients. Variable amount of storage can be bought as packages, while limited space is available for free. All information is redundantly stored in different places in order to increase availability of files. The client application creates a specific folder in the user's home directory. File actions within this directory trigger automatically the syncing to the cloud storage. Furthermore, if available, files can be accessed and managed through a web interface. Typical features of cloud storage services are backup,

[1] https://www.dropbox.com
[2] https://drive.google.com

synchronization, and sharing. Typical implementations of cloud storage services are DropBox, Google Drive, Microsoft SkyDrive[3], Wuala[4], or SugarSync[5].

Encryption middleware describes an encryption interface between client and cloud storage provider, with the purpose to ensure security and confidentiality, independent of the cloud storage provider. As many users doubt the security features of cloud storage providers, encryption middleware tries to resolve this issue. It provides an additional security layer in the form of client-side file encryption, which is performed before files are uploaded to the cloud storage service. This process involves management of required encryption keys, which are required for the en/decryption process.

In the next sub-sections we briefly describe the encryption middleware implementations Boxcryptor[6], CloudFogger[7], and Viivo[8]. An evaluation of these solutions as well as of our proposed approach is given in Section 5.

2.1 Boxcryptor

Boxcryptor is available for multiple platforms, e.g. Windows, Mac OS X, iOS, and Android. Boxcryptor provides support for the cloud storage services Dropbox, SugarSync, Microsoft SkyDrive, and Google Drive. A basic version of BoxCryptor is offered for free. Additionally to this free version, Boxcryptor can be purchased in an unlimited version, which enables filename encryption. Storage is managed in volumes, where each volume is mapped to a specific cloud storage service. Copying files into a volume invokes encryption and the encrypted file is copied into a corresponding subfolder of the cloud storage service directory. For example, copying files into a volume mapped to DropBox will store the encrypted files into a Boxcryptor specific subfolder of the DropBox folder.

2.2 CloudFogger

CloudFogger is freely available for Windows, Mac OS X, Android, and iOS platforms. Supported cloud storage services are DropBox, SkyDrive and Google Drive. Users need to specify which cloud storage services they wish to protect, with the option of disabling protection for subfolders. Protected cloud storage service directories can be accessed and manipulated as usual. However, before uploading files to the cloud storage, CloudFogger encrypts each file and uploads the encrypted file instead.

2.3 Viivo

Viivo is a free product and available for iOS, Android, Mac OS X, and Windows platforms. As of April 2013, DropBox is the only supported cloud storage service. When copying files into the Viivo folder within the user's home directory, it causes the encrypted versions of those files to be stored into a specific subfolder of the DropBox

[3] https://www.sugarsync.com
[4] http://www.wuala.com
[5] https://www.sugarsync.com
[6] https://www.boxcryptor.com
[7] http://www.cloudfogger.com
[8] http://www.viivo.com

directory, which are subsequently uploaded to Dropbox servers. The opposite way around, encrypted files added to the DropBox subfolder are decrypted automatically and consequently stored in the Viivo home folder.

3 Citizen Card Encrypted (CCE)

The following two sub-sections explain the concept of the Austrian citizen card and the Citizen Card Encrypted (CCE) software, which takes use of the Austrian citizen card functionality for encrypting and decrypting data.

3.1 The Austrian Citizen Card Concept

The Austrian citizen card [HKR+08], the official eID in Austria, constitutes a core element within the Austrian e-Government concept. The main aim is to facilitate electronic communication processes between citizens and public authorities. Moreover, by the help of the Austrian citizen card such electronic communication processes can be accelerated and secured at the same time.

In general, the term "citizen card" is more seen as a concept rather than a card. The Austrian e-Government Act [EGovG], which defines the Austrian citizen card in legal terms, emphasizes especially its technology neutrality and its independence of technical components. Due to declared technology neutrality, different implementations are possible and do already exist for the citizen card. Currently, the most dominant citizen card implementation in Austria is a smart card. For instance, each Austrian citizen gets issued a health insurance card (*e-card*), which can easily be activated to use citizen card functionality. Nevertheless, another emerging citizen card technology is based on mobile phones. In this implementation, a server-side hardware security module stores the citizens' secret keys, which can be activated by the use of the citizen's mobile phone.

In general, the most important functionalities of the Austrian citizen card, as regulated in the Austrian e-Government Act, are (1) citizen identification and authentication, (2) generation of qualified electronic signatures and (3) data encryption and decryption.

By using the Austrian citizen card, citizens can be uniquely identified and securely authenticated at governmental or private sector online applications. Additionally, the Austrian citizen card contains a qualified signature certificate according to the EU Signature Directive [EP95]. Hence, electronic signatures created with an Austrian citizen card are legally equivalent to handwritten signatures. Besides this signature certificate, an additional key pair is stored on the card, which can be used for the secure encryption and decryption of data. Thereby, the public encryption keys of every Austrian citizen are available through a central LDAP directory. Hence, data can be encrypted for each Austrian citizen and stored confidentially. In the remainder of this paper, we focus on the encryption and decryption functionality of the Austrian citizen card only.

3.2 The CCE Software

The CCE (Citizen Card Encrypted Software) is a platform-independent and open source software developed by A-SIT (Secure Information Technology Center – Austria). The

software is available through the JoinUp platform, a service initiated and supported by European Commission[9]. CCE especially supports the public authorities demanding high data security and easy and flexible data management. Basically, CCE allows for the encryption and decryption of arbitrary data and the management of files or directories both for single and multiple users.

For file and directory encryption and decryption CCE relies on hardware-based keys, which are stored on the Austrian citizen card. However, also software-based keys can be used within CCE. Particularly the use of the Austrian citizen card enables a highly secure and confidential data exchange since the required keys are stored in hardware and thus cannot be read out by an application. CCE currently supports the smart card-based implementation of the Austrian citizen card only, as no encryption and decryption functionality is provided by the mobile phone signature at the moment. However, other smart card implementations can be easily integrated by implementing an application interface for a particular implementation.

CCE relies on the well-known and established S/MIME [RT04] standard as container format for storing data. S/MIME is also widely integrated in several e-mail clients for encrypting e-mails. In the following, we briefly explain main features of the CCE software.

- *Smart card as secure decryption unit*

  The CCE software supports the use of smart cards to decrypt the keys used in S/MIME containers. The process of decryption is directly carried out on the smart card, initiated by the user entering a personal PIN.

- *Support of group encryption*

  Files and directories can be encrypted for multiple users, which can be organised in a group-like hierarchy. The management of groups is handled manually by the users on their own. However, the support of multiple users also allows for the inclusion of appropriate backup keys.

- *Support of the Austrian PKI infrastructure*

  Asymmetric public key encryption facilitates encryption procedures of users and groups. The public keys of recipients are hence publicly available through the Austrian PKI infrastructure by querying the central LDAP directory. Nevertheless, CCE also enables the integration of arbitrary PKI infrastructures (e.g. from an enterprise context), which can be done by extending its open-source application interface to support the new infrastructure.

4 Architecture and Implementation

In this section we explain the architecture and implementation of our smart card-based approach for storing data securely and confidentially in the public cloud.

[9] http://joinup.ec.europa.eu/software/cce/description

4.1 Architecture

For our solution the CCE software has been extended in order to be able to store data also at public cloud providers and not only on the local storage. Citizens can thereby select between different cloud storage services where data should be stored. The current implementation supports the providers DropBox and Google Drive.

Fig. 1 illustrates our architecture for secure encryption and decryption of data by using the Austrian citizen card functionality and storing the encrypted data in the public cloud. In this architecture, in fact three different entities are involved: (1) the citizen who wants to store some file or directory securely in the public cloud, (2) the Austrian citizens the files or directory should be encrypted for and, (3) the public cloud provider where the encrypted files will be stored.

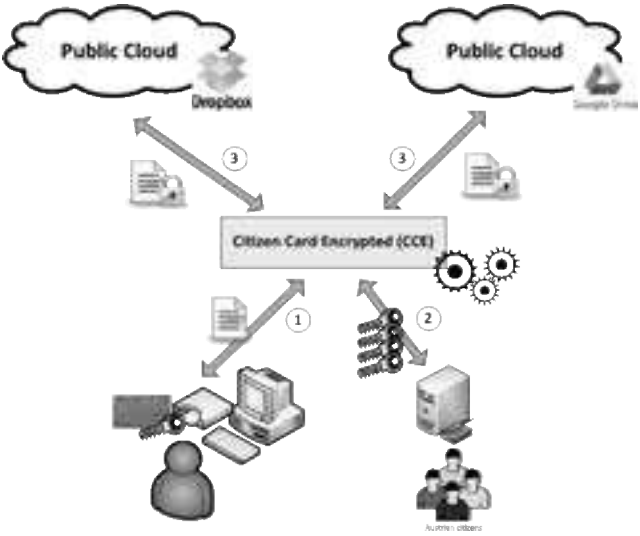

Figure 1: Architecture for securely storing data in the public cloud using the Austrian citizen card

Fig. 1 also illustrates the encryption process using CCE and subsequently the process of storing the encrypted data in the public cloud. In a first step (Step 1), the citizen selects the files and directories she wants to store securely and confidentially in the cloud. In the next step (Step 2), the citizen selects one or more other persons (Austrian citizens) the chosen files or directories should be encrypted for. If citizens' encryption certificates are not known by CCE yet, they can be queried from the central LDAP directory[10]. In this directory, all public certificates of every Austrian citizen registered in the system are stored. Before starting the encryption process, the validity of the encryption certificates of the selected persons is checked. Finally, in Step 3 the data are encrypted for the intended citizens and transferred to the selected public cloud provider. Authentication credentials for accessing the public cloud provider need to be provided during the

[10] The querying of the external LDAP service is not necessary if the users have exchanged the certificates, e.g. using email or by using organizational certificate store. It is also possible to include own LDAP server.

configuration and setup of CCE. During the data transfer, the credentials are retrieved from the CCE configuration and provided to the public cloud provider automatically.

The decryption process is similar to the encryption process; hence the decryption process will not be illustrated. In the decryption process, the encrypted data are downloaded from the public cloud into the local file system by the user. Afterwards, the data are decrypted by using CCE and invoking the citizen's citizen card. Now, the citizen is able to inspect the plain data.

4.2 Implementation

For supporting public cloud storage as an option, CCE had to be amended and extended accordingly. In particular, emphasis was put on flexible adding of additional public cloud providers besides DropBox and Google Drive. For adding an additional cloud provider, the server communication with the cloud provider and its configuration management needs to be implemented. Hence, the modular internal architecture of CCE allows for an easy implementation of new providers.

The creation of a new public cloud provider configuration requires a smart card because the smart card is linked to credential information necessary to access cloud provider services. The credential information for the cloud provider is thereby encrypted by the affiliated smart card, stored in the local file system, and assigned to the corresponding person. Hence, an automatic mapping between smart card and cloud provider authentication credentials is achieved. The advantage of this approach lies in the fact that cloud specific authentication data need to be entered once during configuration; it is then accessed automatically during each subsequent cloud data transfer.

In details, configuration of authentication credentials for cloud provider access is as follows. Authentication at the cloud provider is based on the authorization protocol OAuth[11] for both cloud providers DropBox and Google Drive. Required authentication tokens of OAuth are ascertained during the configuration of a new cloud provider in CCE. This requires the input of the authentication credentials from the user, which in turn adds CCE as trusted cloud application and gives CCE access to the user's cloud account. Subsequently, CCE receives an access token from the cloud provider for the secure access to the cloud storage. According to the OAuth protocol, this access token can be continuously used for cloud provider authentication, so that additional provision of user authentication credentials is not required anymore.

To store data confidentially, users are able to select their desired storage location. The default location is the local file system, whereas users are now able to also store encrypted data at different cloud providers, which are linked with their citizen card. During data upload, saved cloud provider credentials are decrypted by using the user's smart card and are used for cloud provider authentication.

Besides extending the pure CCE application, integration into the operating system's file system has been implemented too. In this case, users are able to copy files into a specific

[11] http://oauth.net

folder of the personal HOME directory and files are then automatically encrypted and transferred to the cloud. When moving files into this specific folder, the CCE wizard starts automatically. Recognition of moved or newly created files in this specific folder is implemented using WatchServices[12], which observes file system operations. Using the CCE wizard, not only files can be automatically encrypted but also desired recipients can be selected. For distribution of encrypted files the existing mechanisms of the respective cloud provider can be used.

5 Evaluation

In this section we evaluate encryption related features and functionalities of middleware implementations for cloud storage in terms of encryption and data sharing.

5.1 Boxcryptor

Boxcryptor **encrypts** files using the AES-256 encryption algorithm. The encryption scheme is volume specific, where all files inside one volume are encrypted with its particular key. This volume-specific key is generated randomly, encrypted with the master key derived from the user's password, and placed in the volume's root. Therefore, in this approach encryption keys are derived from the user's password, which may be leaked through phishing attacks, caught by Trojans, or accidentally published to vicious third parties. Another disadvantage of Boxcryptor's approach is the fact that filename encryption is performed only in the unlimited and retail version of the software. The standard and free version of the software does not obfuscate filenames, which poses additional security risk and information channels for attackers.

Sharing in Boxcryptor is possible only for entire volumes mapped to a specific provider. In order to gain volume access, it is required for the user to share the password, which is not considered as a highly secure practice.

5.2 CloudFogger

During a new account creation on the CloudFogger service, a user specific RSA key pair is generated locally on the user's device. The private key is then **encrypted** with a user provided password, using AES-256 and uploaded together with the public key to CloudFogger servers. In this approach, the encrypted private key information is always downloaded and decrypted with the user's password locally on the user's device, allowing access to protected files. This way, CloudFogger is never able to gain knowledge of private key or password information, making it possible for the user to consume the service on different devices. Each file is individually encrypted using AES-256 whereby AES-keys are encrypted with the user's public key and embedded in the file. Due to file encryption based on user passwords, phishing and Trojan attacks, as well as password leaking, are viable threats to the security of this approach.

As AES-keys are embedded directly in each of the encrypted files, they can easily be **shared** with other subjects. For such purpose, embedded AES-keys files are encrypted

[12] http://docs.oracle.com/javase/7/docs/api/java/nio/file/WatchService.html

with the public keys of invitees[13]. This allows the invitee to locally decrypt shared files with her private key. Sharing can be handled independent of the underlying cloud storage services. However, all participants are required to be registered to CloudFogger.

5.3 Viivo

Similarly as for CloudFogger, RSA key pairs of the users are created locally during the process of account registration. Both public and **encrypted** private keys are stored on Viivo servers. The encrypted private key is downloaded on the user's client device and decrypted by providing the corresponding password. Moreover, each file in the system is encrypted using the AES-256 encryption algorithm, whereby AES-keys are encrypted with the private key associated with the user. As the encryption approach of Viivo is basically similar to the one of CloudFogger, they both share similar disadvantages from the security perspective. Having the encryption keys derived from user passwords, attacks ranging from phishing and Trojan attacks to information leakage are possible for both of the approaches. As all the keys and files depend on one master user password, its leakage may render the whole service and system unusable.

The **sharing** of files with others is performed by inviting the respective user, which has to manually allow sharing of particular files. Creating a share invokes generation of new AES-keys for all files in the share. These keys are in then encrypted with the public key of every invitee. Then, the encrypted keys are sent by the inviting user to each invitee.

From the user's perspective, sharing of a file stored on DropBox is done in two activities. Firstly, the file has to be shared through the DropBox sharing mechanism. Secondly, the sharing of specific files has to be allowed by the invitee through the Viivo interface. In contrary, when access to shared files is revoked, the shared files are not re-encrypted. Instead, new keys are created. New keys ensure that newly created files are no longer accessible by the previously invited user.

5.4 CCE

CCE uses a slightly different approach for file **encryption** than other evaluated solutions. Instead to create RSA key pairs for new users each time they register, and store them on (potentially insecure) local storage prior to the encryption, CCE relies on the existing Austrian citizen card PKI infrastructure. This way, it uses independent, third-party smart card and secure hardware based encryption.

The containers in CCE, which can hold files and directories, are encrypted with AES symmetric keys. These keys are further encrypted using the public RSA key of the Austrian citizen card, taking the public RSA key of each user being allowed to access the container. The containers itself are stored in S/MIME format, which is compatible with a broad range of other applications, including popular e-mail clients. For the decryption of encrypted files, the Austrian citizen card in the form of smart cards is used. This presumes that encryption keys are encrypted with the user's public key and are decrypted in the smart card, using the securely stored private key.

[13]Invitees – persons having access rights on the file

The advantage of this approach is that the private keys are never loaded into the computer system, nor can they be directly accessed or read. Instead, they are contained in the smart card and operations involving them are executed on smart card hardware only when necessary conditions are met (e.g. PIN-based authentication). However, CCE is not limited on the use of Austrian citizen cards only. It can support other PKI infrastructures or smart card implementations, or can rely on software-based keys too.

The **sharing** of files in CCE is performed in two steps. First, the user selects intended recipients during the encryption process. CCE encrypts files for these users by encrypting and storing the symmetric key in the container for each particular user, using her public key. The public key of the user can be stored locally or retrieved from the public LDAP directory of the Austrian citizen card PKI. Furthermore, it is possible to encrypt files for not previously known or contacted users, where prior key exchange or establishing of contact is not necessary. In the second step, the user enables access to the underlying cloud storage for intended users and performs upload of the encrypted containers or synchronization with the local directory with the containers.

The credentials to access remote cloud services in CCE are stored in secure manner. They are encrypted and stored in a local XML file. In order to enable access to remote cloud services, the user has to insert her smart card used during credentials initialization. This approach prevents the leakage of the cloud credentials to unauthorized third parties.

5.5 Summary

In the previously presented evaluations and based on summarized comparison in Table 1, we demonstrate the advantage of our CCE-based solution. From the security perspective, our solution relies on hardware-based encryption, where the private key used for decryption never leaves the smart card. This case does not require the usage of a master password and consequent key derivation as it is the case for BoxCryptor, CloudFogger, and Viivo. The CCE approach is prone to phishing and keylogger attacks[14], which may render the complete system unusable in the case the master password is compromised. From the broader sharing and usability view, our solution's advantage lies in the fact that it relies on a third-party public PKI infrastructure.

The Austrian citizen card is available to all persons living in Austria as a part of several implementations, including bank cards or social health insurance cards, which are basically most widely deployed. As the software is open-source, the support for other PKI infrastructures can be easily implemented by extending the application interface.

Thus, the existing, already present and widely deployed infrastructure is used without incurring additional costs or overhead. That enables the exploration of networking effects, as there are already many users having their public certificate enabled and reachable through the public LDAP endpoint. Users can simply encrypt and exchange encrypted files between each other without the necessity to maintain prior contacts and engage in secure key exchange.

[14] If PIN-Pad smart card readers are used.

Feature \ Middleware	BoxCryptor	CloudFogger	Viivo	CCE
Encryption				
AES support	√	√	√	√
RSA support	-	√	√	√
Volume-based encryption	√	-	-	-
File-based encryption	-	√	√	√[15]
Container-based encryption	-	-	-	√
File names securely stored	-	√	√	√
Software keys supported	√	√	√	√
Hardware keys supported	-	-	-	√
User master password derivation based encryption	√	√	√	-
Encryption keys stored locally	-	-	-	√
Encryption keys stored remotely	√	√	√	-
Phishing attack prone	√	√	√	-
Keylogger attack prone	√	√	√	-
Sharing				
Prior key exchange necessary	√	√	√	-
Public LDAP Key discovery	-	-	-	√
Encryption for unknown users	-	-	-	√[16]
Volume based sharing	√	-	-	-
File/directory based sharing	-	√	√	√
Feature \ Middleware	**BoxCryptor**	**CloudFogger**	**Viivo**	**CCE**
Relies on third-party PKI	-	-	-	√[17]
Multiplatform support	√	√	√	√[18]

Table 1: Comparison of middleware encryption and sharing features

[15] Possible by encrypting one file per container
[16] Using public LDAP endpoint for searching/browsing recipients
[17] Using the Austrian citizen card public key infrastructure. Can be extended with a private PKI.
[18]The mobile versions of the software are not publicly available, however, they are currently in the development phase (iOS and Android platforms)

6 Conclusion and Further Work

The storage of data in the public cloud is becoming popular and a widely used scenario. As the cloud market is intensively growing and providing many new and innovative service and integration solutions, it can be expected that the necessity to store personal or business data in the public cloud will grow even further in nearly future. However, storing data in remote public cloud systems brings new challenges and security risks.

In our work, we focused on data confidentiality and general security aspects of cloud storage in the multi-group and multi-provider scenario. For such purpose we extended the file encryption tool CCE, which acts as an encryption middleware on the local user computer. This extension includes the support for data encryption and sharing via public cloud services. Furthermore, we analyzed publicly available middleware encryption solutions, compared their features, and provided an overview of the features of these tools. Based on this evaluation, we demonstrated that our solution provides significant advantages in terms of data security and resistance to several popular attack techniques. As the proposed solution is based on already deployed and widely used infrastructure, it requires minimal costs or overhead in order to be applied.

There are several directions which could be taken for further development of our work. Currently, there are two versions of the software for iOS and Android in development. We plan to integrate them with the desktop solution presented in this work. Another possible task for the future is the integration of the tool into the web browser, which may provide additional quality in user experience and broader platform support. In addition, we evaluate possibilities to provide cloud storage redundancy at the middleware level, meaning to store data distributed on different cloud services. Finally, we try to integrate our solution in the user's operating system, so that the data can be visible, accessed, and manipulated directly at the operating system level.

References

[ABC01] Abraham, N.; Bibel, U.; Corleone, P.: Formatting Contributions for LNI. In (Glück, H.I. Hrsg.): Proc. 7th Int. Conf. on Formatting of Workshop-Proceedings, New York 1999. Noah & Sons, San Francisco, 2001; S. 46-53.

[Ez99] Ezgarani, O.: The Magic Format – Your Way to Pretty Books, Noah & Sons, 2000.

[EGovG] Federal Act on Provisions Facilitating Electronic Communications with Public Bodies (The Austrian E-Government Act - E-GovG) StF: BGBl. I Nr. 10/2004

[EP95] Data Protection Directive 95/46/EG, EU Parlament, Official Gazette Nr. L 281 from 23/11/1995 P. 0031 – 0050

[HKR+08] A. Hollosi, G. Karlinger, T. Rössler, M. Centner: Die österreichische Bürgerkarte, Version 1.2, 2008, http://www.buergerkarte.at/konzept/securitylayer/spezifikation/aktuell/

[RT04] B. Ramsdell, S. Turner (2004): Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, 2004, http://www.ietf.org/rfc/rfc3851.txt