Practical Security in E-Mail Applications

Franziskus Kiefer[1], Alexander Wiesmaier[2], Christian Fritz[1]

[1] Technische Universität Darmstadt
Hochschulstraße 10, 64283 Darmstadt, Germany
kiefer@cdc.informatik.tu-darmstadt.de
c_fritz@rbg.informatik.tu-darmstadt.de
[2] AGT Group (R&D) GmbH
Hilpertstraße 20a, 64295 Darmstadt, Germany
awiesmaier@agtinternational.com

***Abstract*—This paper deals with practicability issues of encrypted e-mails. A quick survey on the status quo indicates that popular e-mail clients lack substantial practicability qualities, for example searching in encrypted e-mails. Other approaches such as De-Mail provide solutions, but offer transport encryption only. We present and discuss a number of improvements to the practicability of e-mail encryption. These enable efficient searching in encrypted e-mails as well as subject encryption and the use of cryptographic functions in calendar applications. We present a prototype, called CryptoBird, providing a proof of concept for the proposed core features.**

***Keywords*—Calendar, CryptoBird, E-Mail, Encryption, PGP, Practicability, Searching, S/MIME**

I. Introduction

Only few people use encryption to protect their e-mails [11]. While in private communication this might be acceptable, this is a serious issue in case of governmental or business usage as plain e-mails are inherently insecure [21].

Nowadays, all major e-mail clients (and servers) support encrypted e-mail transport using SSL. Once configured correctly, this is transparent to the users and protects their e-mails on their way through the Internet. It provides a reasonable level of privacy protection and is recommendable for day to day private e-mail communication. But in other scenarios such as handling confidential business e-mails or sharing devices with others, transport security is not sufficient, as the e-mails are stored in plaintext.

Possible solutions for this problem are using hard disk encryption or applying e-mail encryption, which both suffer from low popularity [11], [19]. In the work at hand we focus on problems related to e-mail encryption.

Reasons for the unpopularity of e-mail encryption might be unawareness and usability or comfort issues. A lot of usability research in terms of cognitive walkthroughs and user studies has been performed on e-mail encryption [29], [23], particularly on the software Pretty Good Privacy (PGP). We consider the following practicability issues with encrypted e-mails as most serious:[1]

1) Using asymmetric cryptography to encrypt e-mails requires some kind of public key infrastructure (PKI). While the use of X.509 [6] and the Secure/Multipurpose Internet Mail Extensions (S/MIME) [27] is suitable for companies, universities or other organizations, the centralized PKI approach is not practicable for private use. PGP [34] and its infrastructure offer good possibilities, but its correct application is too complicated for the regular user [33].
2) Even if the PKI problems are solved, the handling of encrypted e-mail correspondence comes with some drawbacks. One of the most serious issues is losing the ability to conduct full text searches on e-mail folders when using end to end encryption (which is desired for security reasons). More disturbances appear when using the contents of encrypted e-mails outside the e-mail application, e.g. in calendar applications, which are usually not capable of decrypting the e-mail content on the fly. This leads to either unreadable (encrypted) or insecure (decrypted) calendar entries.
3) Even though the e-mail body is encrypted, the e-mail header is usually not. This leads to a couple of security risks, of which the most serious one probably is that the subject of encrypted e-mails is not encrypted. Even worse is the fact that the user might not be aware of this.

A. Contribution and Delimitation

The work at hand focuses on practicability and security aspects of encrypted e-mail communication and how handling encrypted e-mails differs from handling plaintext e-mails. We also show how these differences can be minimized while still complying to existing standards, resulting in a substantially better aligned user experience.

This work is not a usability study on (plaintext or encrypted) e-mail communication. Long-term issues of public key cryptography are also out of the scope of this work.

[1] To see other reasons why users decide whether to use encrypted e-mails or not we refer to [15].

B. Outline

First, we define practicability criteria and use them for a survey on common e-mail clients (Sec. II). Then, we propose and discuss solutions which are in line with existing standard mechanisms and protocols to solve the identified problems (Sec. III). After that, we discuss related work and compare it to our solution (Sec. IV). We demonstrate the feasibility of our solution presenting a prototypical implementation (Sec. V). Finally, we give a conclusion and discussion on remaining open problems (Sec. VI).

II. Practicability of Secure E-Mails

In this section, we define and discuss criteria for the practicability of secure e-mail clients. With these criteria in mind we analyze existing e-mail clients and identify their shortcomings before proposing our solution to these problems in the subsequent section.

A. Practicability Criteria

Our main approach for optimizing secure e-mail handling is to provide equal treatment of encrypted and plaintext e-mail communication. We consider the smooth integration of security features with the existing user interfaces and work flows as one of the main concerns for practicability. The user should not have significant additional expenditure when using encrypted e-mails and user interaction should take place in a way the user is familiar with from dealing with plaintext e-mails. This is expressed by the following practicability criteria:

Automatic encryption and decryption (automation): To achieve a seamless integration into the e-mail client, the user must be able to write an e-mail in the usual way and toggle encryption so that the e-mail is *sent* and *stored* encrypted. Encrypted text should be automatically decrypted for viewing without requiring user interaction.

Efficient key management (key management): The user should not be obliged to perform demanding key management tasks. However, in PGP the user has to decide whether a key is trustworthy or not. But there are some actions that can be done automatically, for example selecting the correct key for encrypting or decrypting an e-mail (this overlaps with the automation criterion). The client might also search for a key on a (selectable) key-server or offer to import keys received via e-mail into the keyring. Due to the central approach of S/MIME, its certificate and key handling is easier. Keys usually can be downloaded from defined servers (which are assumed to be trustworthy) or come along with S/MIME signed e-mails from where they can be imported automatically into the client's key management. There is also the topic of management of expired keys and certificates, which is especially important in case of long-term archiving of encrypted e-mails. Although long-term storage is an important issue, it is not the most pressing one and is not considered in the work at hand.

Feature Range (feature range): The features offered for encrypted e-mails should not differ from the ones for plaintext e-mails (for example the possibility to search for keywords). In addition to the e-mail functionality, there are several integrated functionalities, such as converting e-mails into calendar entries that work fine with plaintext e-mail content. They should be usable with encrypted content, too.

Seamless integration (integration): Security related functionality should be provided via the existing interfaces, such as tool bars, context menus or filters. If this is not possible, the interfaces should be extended in the style and spirit of existing features and user interfaces. Another important topic is that the security functionality behaves as expected (which is currently not always the case, cf. Sec. III).

Support of common standards (standard support): There are different standards which have to be supported. Regarding e-mail security, these standards are S/MIME [27], PGP/MIME [5] and PGP/INLINE [5] supporting e-mail encryption and signing.

B. Client Analysis

With these criteria in mind we evaluate common e-mail clients regarding their practicability when using encryption features. Our distinction between "encrypted content" and "cleartext" refers to e-mails as well as calendar entries. Table I summarizes the survey. The table is to be read in the following way: 'yes' means that the feature is included in the e-mail client. '+' implies that the functionality in question can be added by installing add-ons or extra programs. 'no' stands for missing functionality. If a feature is only available for a certain standard, this is indicated by the name of the supporting standard, e.g. 'S/MIME'. The detailed results are described in the following paragraphs.

Client	Auto-mation	Key Mgnt.	Feat. Range	Inte-gration	Std. Sup.
Apple Mail	yes/+	yes	no	no	yes
Evolution	yes	+	no	yes	yes
Gmail	+S/MIME	+	no	no	+
MS Outlook	no	+	no	S/MIME	+
Thunderbird	+	+	no	yes	+

TABLE I
FEATURE COMPARISON OF E-MAIL CLIENTS

Apple Mail [1] (aka Mail.app) is the standard e-mail client for Mac OS X (and iOS devices). It supports S/MIME encryption and signing. With the help of the GPGMail [18] plug-in, it is possible to perform PGP encryption and signing (using the desktop application). E-mails are decrypted when the decryption button is pressed. GPGMail can find keys of recipients automatically in the keyring and is even able to download missing keys from a given server. It is not possible to search in encrypted e-mails. All in all only the criteria *key management* and *standard support* are sufficiently fulfilled.

Evolution [25] is an open-source e-mail client for Linux and is shipped with GNOME. It is able to encrypt, decrypt, sign and verify PGP- and S/MIME e-mails automatically. It uses a keyring provided by programs such as the free PGP implementation GnuPG. There is no possibility to directly add keys received by e-mail. Searching for text in encrypted e-mails is not supported. Evolution complies with the criteria

automation, *standard support* and *integration*. The *key management* criterion is only partially fulfilled.

Gmail [17] is a very popular example for web-mail interfaces. Additional software like GnuPG enables PGP usage with Gmail, but requires additional manual actions to be performed by the user. Recurity[28] recently published a first prototype[2] of a PGP plugin for the Chrome[16] browser and Gmail service. Another add-on that works with various web browsers is Penango [26]. It allows to encrypt and sign e-mails with Gmail using the S/MIME standard. The Penango Firefox add-on takes advantage of the build-in certificate management of Firefox. To our knowledge it is not possible to search in encrypted e-mails. None of the practicability criteria defined above is adequately fulfilled.

Microsoft Outlook [24] can handle S/MIME-encrypted e-mails but is not able to process PGP e-mails on its own. There are some approaches to add PGP compatibility, for example a project called gpg4win [20], which is a collection of different tools including a PGP plug-in for Microsoft Office 2003/2007. Currently, the plug-in does not work with Microsoft Outlook 2010.

The commercial software PGP Desktop [30] is a stand-alone solution that handles PGP-encrypted texts. It can be embedded (as an Add-In) into Outlook 2010 and other e-mail clients such as Thunderbird. The decryption requires some manual action: the encrypted e-mail has to be moved to the PGP Viewer window where it gets decrypted. The decrypted e-mail can be copied to the Outlook 2010 inbox - so the e-mails can be searched afterwards, but they are not stored encrypted any longer which might be unwanted due to confidentiality risks. In summary none of the practicability criteria is met completely.

Mozilla Thunderbird is a multi-platform e-mail client which can be easily extended by add-ons. It is able to handle S/MIME encrypted e-mails. By using the add-on Enigmail [31] and GnuPG [13] PGP-encrypted e-mails are supported and processed automatically. Enigmail assists the user by offering an automatic import of keys sent by e-mail and also picks the correct keys for the recipients from the keyring. Thunderbird is lacking a possibility to search in encrypted e-mails. As mentioned above, PGP Desktop can be used with Thunderbird, but regarding our criteria it is not as powerful as the Enigmail add-on. In summary, Thunderbird and its add-ons support *automation*, *key management*, *integration* and *standard support*. The criterion *feature range* is only partly fulfilled.

C. Practicability Conclusion

None of the introduced e-mail clients meets all our practicability criteria. In particular, no e-mail client is compliant with the *feature range* criterion, not even with third party plug-ins. For most clients encryption features are only available after installing extra software or add-ons. The e-mail clients that provide calendar functionality are not prepared to handle encrypted calendar content.

[2]http://gpg4browsers.recurity.com

III. Enhancing the Practicability of Secure E-Mails

In the following we consider an e-mail client with calendar functionality that handles S/MIME encrypted e-mails as well as PGP/INLINE and PGP/MIME encrypted e-mails.

We have seen that currently no e-mail client exists which provides equal treatment of encrypted and plaintext e-mails, in particular meeting all our criteria. In this section we propose and discuss concrete improvements to overcome the shortcomings discussed in Section II. As all of the investigated clients miss this quality, we focus on the equal treatment of plaintext and encrypted e-mails. Furthermore, we deal with some additional improvements that enhance the security of encrypted e-mails. We consider the ability to search in e-mails as one of the most critical shortcomings when it comes to the handling of encrypted e-mails. As shown in [32], searching in e-mails is one the most efficient ways to organize them.

In the following we investigate possibilities to reach a seamless integration of encrypted e-mails into existing clients. Additionally, we provide some details on our prototypical implementation of the proposed improvements. We implemented an open source prototype as Thunderbird add-on called CryptoBird as proof of concept[3] (cf. Section V).

In addition to the previously defined criteria we consider the following:

In order to provide reasonable efficiency when working with big amounts of data, some kind of indexing or caching might be necessary. In this case, it is important to work with an encrypted index or cache to avoid compromising the confidentiality. An encrypted index or cache might require an additional password, which should be integrated into the password manager of the e-mail client.

Current implementations of e-mail encryption show a strange and risky behavior: the encryption applies to the body only and does not include the header, especially not the subject line. To meet the user's expectation when encrypting a message, some kind of header encryption has to be applied to encrypted e-mails. This way the user is not lulled into a false feeling of security when encrypting e-mails.

A. Searching in Encrypted Content

As mentioned before, we consider conducting full text searches in the bodies of encrypted e-mails as one of the most pressing matters. This pays attention to the paradigm shift from ordering to searching big amounts of data, especially when dealing with e-mails [32]. While this is possible in some e-mail clients for the currently displayed e-mail, it is currently impossible to conduct reasonable full text searches over multiple encrypted e-mails in folders or whole mailboxes.

A main problem which has to be solved here is the efficiency. A search in encrypted e-mails would take a long time if each e-mail has to be decrypted on the fly while searching. This lack of efficiency has mainly two reasons. First, the straight forward full text search in stored e-mail bodies is slow

[3]The prototype can be found at https://code.google.com/p/cryptobird/.

anyway. Second, the cryptographic operations need additional time.

We solve this in CryptoBird by indexing all decrypted e-mails and storing them in a secure database. This database offers encrypted data storage while preserving efficient querying features. This procedure can also be applied for other sensitive content like calendar entries. Section V gives more technical details on this topic.

This solution especially supports the *integration* and *feature range* criteria while preserving efficiency.

B. Calendar Integration

Another issue is the conversion of encrypted e-mails into calendar entries (events or tasks) using the integrated calendar of the e-mail client. During the conversion a new calendar entry is created from an e-mail and inserted into a selectable calendar (local or remote). The new calendar entry is prefilled with the body of the e-mail and is displayed for editing. The e-mail's subject is used as title of the calendar entry. Furthermore, the conversion between tasks and events as well as all other calendar functions should work whether the content is encrypted or not.

It should be possible (and the default behavior) to keep the content encrypted for confidentiality reasons (especially when using third party online calendars). Then, the plaintext content should be displayed in cleartext to the user and all invitees in an automated manner. This can be achieved with the same techniques as with e-mails sent to multiple recipients, i.e. encrypting the session key with the public key of each participant. It should also be possible to easily generate a calendar entry containing the plaintext body of an encrypted e-mail. This is useful when the user wants to generate a public calendar entry or not all invitees are able to view encrypted calendar entries. Then, the user should be able to edit the calendar entry before saving it to have the possibility to delete confidential information. The other way round — converting a calendar entry into an e-mail — also has to be supported. In this case the same applies analogously. It should also be possible to search in encrypted calendar entries (cf. previous subsection).

C. Encryption Toggling

Another useful feature is toggling the encryption status of existing e-mails or calendar entries. This means (bulk) encryption or decryption of stored e-mails or calendar entries. This should be possible for individual or multiple selected items or for entire e-mail folders or calendars. We discuss the usefulness of both directions:

Encrypting plaintext e-mails / calendar entries: It is obvious that encrypting cleartext that has been submitted via an open network or has been stored in an untrusted environment does not annihilate the disclosures it may already have suffered. The point in encrypting such data is to decrease future risk of disclosure. An everyday example where this is useful is loosing the device the e-mail client is stored on, for example a laptop or a memory stick. By encrypting all e-mails (although received in cleartext) a possible thief or finder of the device is prevented from reading the e-mails. This could also be achieved by encrypting the entire device, but a recent study [19] shows that only very few users actually use hard disk encryption. An interesting feature in this scenario is the automatic encryption of all stored e-mails or calendar entries by the e-mail client. This would for example protect all server side data of Web based e-mail clients in case the user's account is hacked.

Sometimes cleartext e-mails or calendar entries never leave the trusted environment. For example, by sending intra-organization e-mails from computers within an organization The same applies for calendar entries created and stored in local calendars. When leaving the trusted environment, e.g. when the organization is about to outsource its e-mail and calendar services into the Cloud, it seems appropriate to encrypt these e-mails and calendar entries. This is also applicable to private users.

As we see, encrypting plaintext e-mails or calendar entries has useful applications and should be supported by the e-mail clients.

Decrypting ciphertext e-mails / calendar entries: At first glance, the permanent decryption of ciphertext seems unreasonable, in particular if the information is still to be protected. An applicable example is a local storage which is considered secure (for example if hard disk encryption is applied). By decrypting e-mails / calendar entries all cleartext based features of the e-mail client or additional software (such as spam or malware scanners) can be used, while the transport is protected by encryption[4]. If this is to be considered secure, it has to be made sure that the e-mail server / calendar server and the connection to the server is also secure, as local decryption might lead to decryption on the server (e.g. when using IMAP folders). In this scenario, automatic decryption of all stored e-mails by the e-mail client might be applied. It is important that the e-mail is per default encrypted again when it is forwarded or the user replies to it.

Another scenario where permanently decrypted e-mails or calendar entries might be appropriate is when a user leaves the company and has to deliver the information to his successor or superior. He can pass the information on without revealing his private key. A similar situation occurs when the user is forced by court order to reveal his e-mail communications or business appointments.

As we see, permanently decrypting ciphertext e-mails or calendar entries has reasonable scenarios. A further (practicability related) scenario is restoring accidentally encrypted plaintext e-mails.

D. Header Encryption

One major drawback that might lead to serious information disclosure is the following characteristic of e-mail encryption standards: Even if the body is encrypted, all header information including the subject is sent in plaintext. If a user

[4]This is exactly what happens in the upcoming De-Mail solution (cf. Section IV).

does not consider this, he might unwillingly disclose sensitive information. We discuss two different mechanisms to realize header encryption.

Enveloping: [27, 3.1] proposes a way to secure headers, especially the subject, of e-mails. The proposal is from the S/MIME standard but is also applicable to PGP/INLINE and PGP/MIME encrypted e-mails. In order to protect the header fields, the sending client wraps a full MIME message in a message/rfc822 [7] wrapper in order to apply security services to the header fields. This is possible since the entire e-mail (header and body) consists of printable characters and thus, can be handled as a message body itself. As discussed in [22], this approach comes with some drawbacks like duplicated header fields. According to our knowledge, no commonly used e-mail client implements this feature.

Field Encoding: The second possibility to secure header fields is to encrypt them separately one by one. The subject header field, for example, can be encrypted using standard S/MIME or PGP mechanisms. As both produce Base64 encoded ciphertexts, these can be stored in the header field instead of the original content. Before sending an e-mail the user can decide to encrypt for example the subject. The decisions may also be linked, so that the subject is encrypted as soon as the e-mail is encrypted, using the same encryption mechanism. The encryption and decryption has to be performed transparently. Obviously, the encrypted subject has to be handled the same way as encrypted bodies (cf. Subsection III-A). Due to the lack of standardization both sender and receiver have to make sure to be interoperable. Otherwise, the receivers e-mail client may display the Base64 encoded ciphertext as subject.

E. Future of Encrypted E-Mails

Contemporary e-mail clients support encryption algorithms that meet the security demands of the near future. But new algorithms – e.g. post-quantum algorithms – are developed. For long-term security it should be possible to react to the progress in cryptographic research by including new algorithms in a flexible way. We created a proof of concept called ThunderCrypt [2] to provide the possibility to use cutting edge cryptographic algorithms in e-mail communication, but we do not cover this topic in the work at hand.

IV. Related Work and Comparison

To our knowledge there is no previous work that deals with equal treatment of encrypted and plaintext e-mail communication, but there are some other approaches that offer improved handling of encrypted e-mails. We analyze the approaches which try to solve some of the identified problems. While our approach enhances the practicability and security and complies with the given standards and protocols as far as possible, other approaches deviate from existing standards.

A. Client Site Approaches

One possibility to overcome the described shortcomings of deployed encryption mechanisms is to use enhanced cryptographic schemes. We present two client site approaches that cover only certain aspects of our main approach.

Opportunistic Encryption: A very convenient way of avoiding practicability problems in e-mail encryption is to use so-called Streams with opportunistic encryption as proposed in [14]. Using opportunistic encryption, the e-mail client tries to find a key matching the receivers e-mail address. If the e-mail client is not able to find any matching key it falls back to the unencrypted mode. This is completely transparent for the user. The proposed Streams additionally create a sanitized e-mail header and encapsulate the original one in the e-mail to protect the subject.

From the practicability point of view, opportunistic encryption is very good in the sense that the encryption process does not interfere with the functionality. As it spares the user to interact with the encryption at all, the user does not need any knowledge of the underlying cryptography which is good for practicability. But regarding the security, opportunistic encryption is disastrous. The user does not have control over the actual security measures applied to his message. Furthermore, even if transferred encrypted, e-mails are stored in plaintext on the receiver's device, so that they are locally vulnerable. Thus, opportunistic encryption meets most of our criteria but lacks the standard support and also has some privacy issues.

Encryption with Keyword Search: Another proposal that covers only one aspect of our approach is the possibility to search for predefined keywords (tags) in encrypted e-mails [4], [3]. The focus is to hide all information from a third party (for example the e-mail server), which is nonetheless able to fulfill requests based on the tags. The approach does not allow to perform full text searches and is not standardized.

B. Server Site Approaches

Besides the possibilities to use enhanced cryptographic mechanisms in e-mail clients (on user site), there are also techniques where the security is achieved by server side measures (on provider site). We present De-Mail [10] as an example. It is an e-mail project promoted by the German federal government to make e-mail transfer secure and legally binding. Assuming that Alice wants to send an e-mail to Bob using De-Mail, she logs into her De-Mail account to send the e-mail without performing any additional steps. The connection between Alice and her e-mail provider is secured by standard mechanisms (e.g. mutual authenticated SSL/TLS [8]). The e-mail provider encrypts the e-mail and sends it (using a secure channel) to Bob's e-mail provider. Bob's e-mail provider decrypts the message and checks its integrity before storing it in Bob's inbox. When Bob now checks his e-mails (using a secure connection), he sees Alice's e-mail and can be sure that Alice wrote it and nobody else has been able to read or modify its content (if he trusts the De-Mail provider).

A major concern regarding such techniques is the privacy of the data, since the involved e-mail providers are able to read all messages in plaintext. Thus, the user has to put unconditional trust into all involved De-Mail providers. This

trust is to be established by a certification process supervised by the German Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik, BSI) [12], [9]. Furthermore, only the transport is secured. The e-mail is not secured locally and on the servers of the provider. Registration is possible since July 2010, the service is available for corporate customers since march 2012 and some providers state to start the service for private customers in 2012. Apart from a small pilot scheme in 2010, there are no practical experiences or studies on the ease of use yet.

C. Related Work & Conclusion

There is no implementation of opportunistic encryption available for current mainstream e-mail clients. Even if it was available, both sender and receiver had to use the same plugin as opportunistic encryption is not standardized. The same applies for encryption with keyword search.

Despite the fact that De-Mail service providers most likely will charge the users for sending e-mails, it is not possible to send or receive a De-Mail with a regular e-mail account since the e-mail provider is not able to process the De-Mail.

Therefore, none of these approaches are satisfying. They do not solve the practicability problems depicted in Section II or miss to use standardized protocols.

V. CryptoBird

As a proof of concept we implemented features described in Section III as a Thunderbird add-on called CryptoBird. In the following we describe the core features and how they implement the mechanisms described in Section III to make a step towards equal treatment of encrypted and plaintext e-mails. Thunderbird with the Enigmail and Lightning add-ons offers the basic functionality to handle plaintext and encrypted e-mails as well as calendar entries, which we extended to fulfill the following criteria:

A. Searching in encrypted e-mails

One of CryptoBird's core features is to allow the user to conduct full text searches in the bodies of encrypted e-mails over entire folders or mail boxes as introduced in Section III-A. Thereby, it supports all common standard mechanisms, namely S/MIME, PGP/INLINE and PGP/MIME.

To realize the search in encrypted e-mails the secure database software Derby[5] is used. The software allows password based encrypted storage of all database contents while still providing the common database functionalities, especially efficient data querying. Upon arrival of a new encrypted email, it is indexed and stored in the database. CryptoBird traces e-mails throughout their life cycle. Moving e-mails from one folder to another one or moving entire folders, cause the database to be updated so that the search results remain valid. Deleting an e-mail will also delete the according database entry. With CryptoBird, not only plaintext e-mails but also encrypted e-mails are considered when searching for a keyword using Thunderbird's standard search feature. Therefore not only the internal Thunderbird database is queried but also our password protected database.

[5]http://db.apache.org/derby/

B. Encrypted Calendar Entries

Using the e-mail client as personal information manager (PIM), a calendar feature is integrated in the e-mail client (in the case of Thunderbird Lightning undertakes this task). Converting an e-mail into an event or task works fine with plaintext e-mails. But encrypted e-mails are only decrypted on the fly to display. Converting an encrpyted e-mail to an event or task, Lightning inserts the ciphertext as description in the case of PGP/INLINE encrypted e-mails. S/MIME and PGP/MIME encrypted e-mails lead to empty events and tasks. CryptoBird offers the possibility to convert at least PGP/INLINE encrypted e-mails to calendar events (cf. Subsection III-B) for now. The event or task can be stored in plaintext or encrypted. Encrypted events and tasks lead to the same problem as with encrypted e-mails. To overcome this, CryptoBird decrypts events and tasks on the fly, similar to the decryption of e-mails, and stores them in the database for searching.

C. Header Encryption

CryptoBird does not only enhance the feature range for encrypted e-mails, but also brings additional privacy features. To solve the privacy problems of plaintext e-mail subjects in encrypted e-mails, CryptoBird allows to encrypt the subject field using PGP as explained in Subsection III-D using the encoding technique. To ensure equal treatment, the decrypted subjects are also stored in the encrypted database to consider them in the keyword search.

D. CryptoBird Conclusion

Having the criteria from Section II in mind, CryptoBird proofs the possibility of fulfilling all five criteria. As CryptoBird complies to the established standards, the regular e-mail encryption features stay interoperable with standard e-mail clients without CryptoBird. However, due to the lack of standardization in the case of header encryption and encrypted calendar entries, all participating parties have to install the CryptoBird extension to read encrypted subjects and calendar entries.

VI. Conclusion and Open Issues

We saw that the practicability of security in e-mail applications is an important yet not sufficiently addressed issue. The different mainstream e-mail clients provide different degrees of practicability in this respect, but none of them provides a solution comparable to the comfort of using plaintext e-mails. We discussed the four in our eyes most pressing practicability issues namely searching in encrypted content, calendar integration, toggling the encryption, and header encryption. We showed how it is possible to solve them while sticking to the most common standards, thereby guaranteeing interoperability. As a proof of concept we presented CryptoBird, a prototypical Thunderbird add-on implementation.

An open issue, regarding the subject and calendar encryption, is the lack of standardization. An adaption of the according RFCs may be reasonable here. Further projects should increase the practicability of CryptoBird by implementing the remaining not yet implemented features described in Section III. An indexed database may also be desirable for efficiency reasons. An integration of the PGP related features of CryptoBird into Enigmail seems be reasonable. It is also thinkable to integrate CryptoBird's S/MIME enhancements into Enigmail. In order to evaluate the usability enhancements CryptoBird provides, conducting a user study would be the next step.

REFERENCES

[1] Apple Inc. Apple mail. http://www.apple.com/macosx/whats-new/mail.html, 2012. Accessed: 18/02/2012.

[2] S. Arzt. Design and implementation of a cryptographic plugin for e-mail clients, Nov 2009. Whitepaper.

[3] J. Baek, R. Safiavi-Naini, and W. Susilo. Public Key Encryption with keyword Search Revisited. In *International conference on Computational Science and Its Applications*, pages 1249–1259. Springer, 2008.

[4] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public Key Encryption with keyword Search. In *Advances in Cryptology Eurocrypt*, volume 3027, pages 506–522. Springer, 2004.

[5] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. OpenPGP Message Format. RFC 4880 (Proposed Standard), November 2007. Updated by RFC 5581.

[6] M. Cooper, Y. Dzambasow, P. Hesse, S. Joseph, and R. Nicholas. Internet X.509 Public Key Infrastructure: Certification Path Building. RFC 4158 (Informational), Sept. 2005.

[7] D. Crocker. Standard for the format of ARPA Internet Text Messages. RFC 822 (Standard), Aug. 1982. Obsoleted by RFC 2822, updated by RFCs 1123, 2156, 1327, 1138, 1148.

[8] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), Aug. 2008. Updated by RFCs 5746, 5878, 6176.

[9] Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik). Akkreditierung von de-mail-diensteanbietern, 2011. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Egovernment/De_Mail/De-Mail-Akkreditierung-Prozessuebersicht.pdf?__blob=publicationFile.

[10] Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik). BSI - Technische Richtlinie DE-Mail. Technical Directive (BSI-TR-01201), Version 1.00, 2011. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/De_Mail/TR_De_Mail_pdf.pdf?__blob=publicationFile.

[11] Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik). De-mail - eine infrastruktur für sichere kommunikation. https://www.bsi.bund.de/DE/Themen/EGovernment/DeMail/DeMail_node.html, 2012. Accessed: 18/02/2012.

[12] F. O. for Information Security (Bundesamt für Sicherheit in der Informationstechnik). Verfahrensbeschreibung zur akkreditierung von de-mail-diensteanbietern, 2011. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Egovernment/De_Mail/Verfahrensbeschreibung-zur-Akkreditierung-De-Mail-Dienstanbieter.pdf?__blob=publicationFile.

[13] Free Software Foundation Inc. The GNU Privacy Guard. http://www.gnupg.org/, 2012. Accessed: 25/02/2012.

[14] S. Garfinkel. Enabling email confidentiality through the use of opportunistic encryption. In *Proceedings of the 2003 annual national conference on Digital government research*, pages 1–4. Digital Government Society of North America, 2003.

[15] S. Gaw, E. W. Felten, and F.-K. Patricia. Secrecy, flagging, and paranoia: adoption criteria in encrypted email. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 591–600. ACM, 2006.

[16] Google. Chrome. http://www.google.com/chrome, 2012. Accessed: 18/02/2012.

[17] Google Inc. Gmail. http://gmail.google.com/mail/, 2012. Accessed: 14/02/2012.

[18] GPGMail. Gpgmail. http://www.gpgtools.org/gpgmail/index.html, 2012. Accessed: 14/02/2012.

[19] O. Inc. ISecurity Industry Market Share Analysis, Sept. 2011.

[20] Intevation GmbH. Gpg4win - a secure solution... http://www.gpg4win.org/index.html, 2012. Accessed: 11/02/2012.

[21] J. Klensin. Simple Mail Transfer Protocol. RFC 5321 (Draft Standard), Oct. 2008.

[22] L. Liao and J. Schwenk. Header protection for s/mime draft-liao-smimeheaderprotect-05. http://tools.ietf.org/html/draft-liao-smimeheaderprotect-05, 2009. Accessed: 18/02/2012.

[23] D. G. T. Markotten. *Benutzbare Sicherheit in informationstechnischen Systemen*, chapter 4. Rhombos Verlag, Berlin, 2004.

[24] Microsoft Corporation. Mircosoft outlook 2010. http://office.microsoft.com/outlook/, 2012. Accessed: 18/02/2012.

[25] Novell Inc. Evolution. http://projects.gnome.org/evolution/, 2012. Accessed: 18/02/2012.

[26] Penango Inc. Penango. http://www.penango.com/index.html, 2012. Accessed: 18/02/2012.

[27] B. Ramsdell and S. Turner. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. RFC 5751 (Proposed Standard), Jan. 2010.

[28] recurity.com. Recurity. http://www.recurity.com/, 2012. Accessed: 18/02/2012.

[29] S. Sheng, L. Broderick, C. A. Koranda, and J. J. Hyland. Why Johnny Still Can't Encrypt: Evaluating the Usability of Email Encryption Software. In *Symposium On Usable Privacy and Security*, 2006.

[30] Symantec Corporation. Pgp desktop email - email encryption software for desktop and laptop computers. http://www.symantec.com/business/desktop-email, 2012. Accessed: 02/02/2012.

[31] The Enigmail Project & mozdev.org. Enigmail - openpgp email security for mozilla applications type. http://enigmail.mozdev.org, 2012. Accessed: 26/01/2012.

[32] S. Whittaker, T. Matthews, J. Cerruti, H. Badenes, and J. Tang. Am I wasting my time organizing email?: a study of email refinding. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 3449–3458. ACM, 2011.

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

[34] Zimmermann, Philip R. *The official PGP user's guide*. MIT Press, Cambridge, MA, USA, 1995.