HOCHSCHULE
HAMM-LIPPSTADT

Bachelorarbeit

„Welche Maßnahmen sind dazu geeignet, um die Verbreitung des WKD-Standards zu fördern?“

im Studiengang
Computervisualistik und Design

vorgelegt von

Christoph Klassen
Matr.-Nr.: 2180089

am 15.02.2022
an der Hochschule Hamm-Lippstadt

Betreuer: Prof. Dr. Georg Birkenheuer

Abstact

This thesis deals with Web Key Directories (WKD), a procedure which makes it possible to retrieve public keys for the encryption of emails. The goal of WKD is to improve the user experience of obtaining these keys by improving this process. For many users encryption is a complicated process so it is welcomed to simplify it by using WKD. Accordingly, the goal of this work is to spread WKD.

To increase the spreading different methods, known from software engineering, will be used. One of these methods is to test chosen free software which is able to encrypt with OpenPGP keys to find out if it is compatible with the WKD-standard and how user-friendly the usage of WKD is. Another important method is the implementation for a product to make it compatible with the standard.

The results of the tests show that the usability of the usage of WKD is not satisfying overall and there is a huge potential for improving it. Often it requires much effort from users to prepare the software for WKD or it is complicated to apply WKD. For this reason criteria were created that should be employed to counteract these problems. The method containing the implementation reveals the importance of writing the WKD-standard and other standards in a precise way. Further, it is significant that persons who formulate a standard consider different kinds of products and different platforms, on which they are offered, while they are checking if the technical options allow the implementation of the standard.

On one side the revision of the WKD-standard is a possibility to reduce the effort of the standard's implementation. That way the probability of more implementations will be increased. On the other side the improvement of products can raise the usability of these products. Together all measurements have the potential to spread WKD by network effects. The reason is that WKD will be available for a greater amount of users so that the number of possible communication partners who can use encryption grows.

Zusammenfassung

Diese Arbeit beschäftigt sich mit Web Key Directories (WKD), einem Verfahren, das es ermöglicht, öffentliche Schlüssel für die Verschlüsselung von Mails zu beziehen. Das Ziel von WKD ist es, die User Experience beim Erlangen dieser Schlüssel zu verbessern, indem dieser Vorgang erleichtert wird. Da die Verschlüsselung für Benutzer:innen oft noch zu kompliziert ist, ist es zu begrüßen, einen Teil dieses Prozesses durch WKD zu vereinfachen. Aus diesem Grund hat sich diese Arbeit das Ziel gesetzt, einen Beitrag zur Verbreitung von WKD zu leisten.

Um die Verbreitung zu fördern, werden verschiedene Methoden des Software Engineerings verwendet. Eine dieser Methoden besteht darin, ausgewählte Freie Software, die bereits die Verschlüsselung mithilfe von OpenPGP-Schlüsseln unterstützt, zu testen, um festzustellen, ob die Anwendungen kompatibel zum WKD-Standard sind und wie benutzerfreundlich die Verwendung von WKD ist. Eine weitere wichtige Methode ist die Implementierung für ein Produkt, damit dieses kompatibel zum Standard wird.

Die Ergebnisse aus den Tests machen deutlich, dass die Usability bei der Benutzung von WKD insgesamt nicht zufriedenstellend ist und es noch ein großes Verbesserungspotential gibt. Oft müssen Anwender:innen einen großen Aufwand betreiben, bis sie WKD einsetzen können oder die Verwendung von WKD ist zu kompliziert. Infolgedessen beinhaltet diese Arbeit auch die Erstellung von Kriterien, die bei der Implementierung von WKD angewandt werden sollten, um diesen Problemen entgegenzuwirken. Durch die Implementierung entsteht die Erkenntnis, dass sowohl für den WKD-Standard als auch für andere Standards gilt, diese möglichst präzise zu formulieren. Es ist zudem wichtig, dass Personen bei der Entwicklung eines Standards verschiedene Arten von Anwendungen bzw. die unterschiedlichen Plattformen, auf denen die Software angeboten wird, berücksichtigen, während sie überprüfen, ob die technischen Möglichkeiten die Umsetzung des Standards erlauben.

Auf der einen Seite ist die Überarbeitung des WKD-Standards eine Möglichkeit, die Hürde bei der Implementierung von WKD zu verringern. Somit steigt die Wahrscheinlichkeit für dessen Implementierung. Auf der anderen Seite können Verbesserungen an den Produkten für eine bessere Usability bei der Nutzung von WKD sorgen. Zusammen haben diese Maßnahmen das Potential, WKD durch Netzwerkeffekte zu verbreiten, da die WKD-Funktion weiteren Benutzer:innen zur Verfügung steht und die Anzahl möglicher Teilnehmer:innen für eine verschlüsselte Kommunikation steigt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar

Advanced Method: Eine Methode im WKD-Standard, die verwendet wird, um eine URL zu erstellen und daraufhin einen öffentlichen Schlüssel zu beziehen. Sie kommt zum Einsatz, wenn die Subdomain *openpgpkey* eines WKD-Servers vorhanden ist [9].

Direct Method: Eine Methode im WKD-Standard, um eine URL zu erstellen und einen öffentlichen Schlüssel zu beziehen. Sie wird nur eingesetzt, wenn ein WKD-Server die Subdomain *openpgpkey* nicht anbietet [9].

Freie Software: Ein feststehender Begriff für Software, ...

> ... die von allen Personen verbessert werden darf.
>
> ... die von jeder Person für den eigenen Bedarf angepasst werden darf.
>
> ... die für jeden Zweck verwendet werden darf.
>
> ... die untersucht werden darf, um die Funktionsweise zu verstehen.
>
> ... deren Quellcode als Voraussetzung für die vorangegangenen Aspekte öffentlich ist. [26].

Owner Trust: Das Vertrauen einer Person in eine andere Person bzw. deren Schlüssel. Dieses wirkt sich auf die Berechnung der Validität eines Schlüssels aus. Die Stufen des Owner Trusts lauten: „Unknown“, „None“, „Marginal“ und „Full“ [10].

Schlüssel: In dieser Arbeit wird mit Schlüssel der öffentliche Schlüssel in der asymmetrischen Verschlüsselung gemeint. Wenn von einem privaten Schlüssel gesprochen wird, wird dies explizit erwähnt.

Validität: Gibt die Vertrauenswürdigkeit eines Schlüssels wieder und wird z.B. in GnuPG verwendet. Wie valide ein Schlüssel ist, hängt von seinem Owner Trust, seinen Signaturen sowie dem Owner Trust der Schlüssel, die ihn signiert haben [10].

Web Key Directory (WKD): WKD ist ein dezentrales Verfahren, bei dem eine URL aufgerufen wird, um öffentliche Schlüssel von einem Verzeichnis auf einem Server herunterzuladen [9].

WKD-Standard: Es existiert ein Entwurf für einen Standard von Werner Koch, dem Hauptentwickler von GnuPG, der das Verfahren WKD beschreibt. Zu Beginn der Arbeit liegt der zwölfte Entwurf vor [9].

1. Einleitung

Motivation

Damit zwei Seiten in einer Kommunikation vertrauliche Informationen austauschen können, sind meist verschlüsselte Kanäle notwendig. In der Politik können diese Informationen Entscheidungen zu Konflikten zwischen verschiedenen Ländern der Welt betreffen. Für ein Unternehmen ist es wichtig, Strategien oder Informationen über einen Prototypen geheim zu halten. Journalisten versuchen, ihre Quellen zu schützen, während sie Einblicke in heikles Wissen erhalten. Aber auch für andere Personen ist Verschlüsselung eine Option, denn sie kann zusätzlichen Schutz bieten, da Angreifer:innen versuchen könnten, sich Zugriff zu sensiblen Daten zu verschaffen. Für diese und weitere Situationen wurden verschiedene Verschlüsselungsverfahren erfunden und entwickelt. Dazu gehören sowohl symmetrische als auch asymmetrische Verfahren. Diese werden für synchrone und asynchrone Nachrichten verwendet. Zur letzteren Gruppe gehören Mails, auf die der Fokus dieser Arbeit liegt.

Pretty Good Privacy (PGP) ist ein Programm, welches die Verschlüsselung von Mails ermöglicht und 1991 von Philip Zimmermann implementiert wurde [1]. Seit 2010 war es im Besitz von Symantec und wurde von diesem Unternehmen als proprietäre Software angeboten [2]. Symantec wurde 2019 wiederum vom Unternehmen Broadcom aufgekauft [3]. Als Nachfolger von PGP kann GNU Privacy Guard (GnuPG) betrachtet werden, welches 1997 eingeführt wurde und als Freie Software bereitgestellt wird. GnuPG setzt den OpenPGP-Standard um [4]. Dieser basiert auf Teilen der Technologie, die in PGP zum Einsatz kam. Die aktuelle Version des OpenPGP-Standards wurde 2007 für die Öffentlichkeit zugänglich gemacht [5].

Für das Verschicken von verschlüsselten Mails mit dem asymmetrischen Verfahren ist es notwendig, dass die Person, die eine Mail schreibt, den öffentlichen Schlüssel der Person besitzt, die diese Mail erhalten soll. Eine Option ist, dass sich beide Kommunikationspartner:innen persönlich treffen und die Schlüssel über einen USB-Stick oder Ähnliches austauschen. Dieses Vorgehen ist jedoch weder komfortabel noch einfach praktizierbar, wenn sich beide Personen an weit auseinander liegenden Orten befinden. Abhilfe schafft dabei ein Verfahren, das Web Key Directories (WKD) genannt wird. Bei diesem Verfahren muss lediglich eine URL erstellt und aufgerufen werden. Daraufhin wird der passende öffentliche Schlüssel heruntergeladen und kann direkt verwendet werden. Es ist also ein geringer Aufwand nötig, um eine verschlüsselte Kommunikation zu beginnen.

Das Verschlüsseln von Mails findet in einer sehr kurzen Zeit statt und bietet dabei eindeutige Vorteile gegenüber unverschlüsselten Mails. Trotzdem ist die Verschlüsselung nicht weit verbreitet. Eine Studie [6] aus dem Jahr 2018, die im Auftrag von WEB.DE und GMX durchgeführt wurde, zeigt, dass nur

knapp 14 % der befragten Personen verschlüsselte Mails versendete, um ihre persönlichen Daten im Internet zu schützen (Abbildung 1). Bei der Frage, warum bisher keine Verschlüsselung verwendet wurde, gab fast die Hälfte der Personen (46,6 %) an, dass sie den Aufwand für die Verschlüsselung als zu hoch betrachteten. Fast genauso viele Personen (43,7 %) wussten überhaupt nicht, wie sie ihre Mails verschlüsseln könnten (Abbildung 2). Befragt wurden 1.008 deutsche Internetnutzer:innen ab einem Alter von 14 Jahren. Genauere Angaben zu den befragten Personen gehen nicht aus der Studie hervor. Es ist also nicht bekannt, wie technikaffin diese Personen waren oder, ob sie sich schon mit dem Thema Verschlüsselung auseinandergesetzt zu haben.

Die Umfrage zeigt jedoch, dass es wichtig ist, die Verschlüsselung von Mails zu vereinfachen. Das Ziel des Verfahrens WKD ist die Verbesserung der User Experience beim Beziehen von öffentlichen Schlüsseln [7]. Gleichzeitig trägt es dazu bei, dass die gesamte Verschlüsselung von Mails ein Stück einfacher wird. WKD bietet deutliche Vorteile, denn öffentliche Schlüssel können auf eine einfache Art und Weise geholt werden und bieten dabei eine grundlegende Vertrauenswürdigkeit. Deshalb beschäftigt sich diese Arbeit mit der Frage, welche Maßnahmen dazu geeignet sind, um den WKD-Standard zu verbreiten. Sie wird im Rahmen eines Praktikums beim Unternehmen Intevation verfasst. Dabei wird sie von einem der Geschäftsleiter dieses Unternehmens, Bernhard Reiter, betreut. Er hat bereits seit einigen Jahren bei der Entwicklung von GnuPG und Gpg4win mitgewirkt und zum WKD-Standard beigetragen.

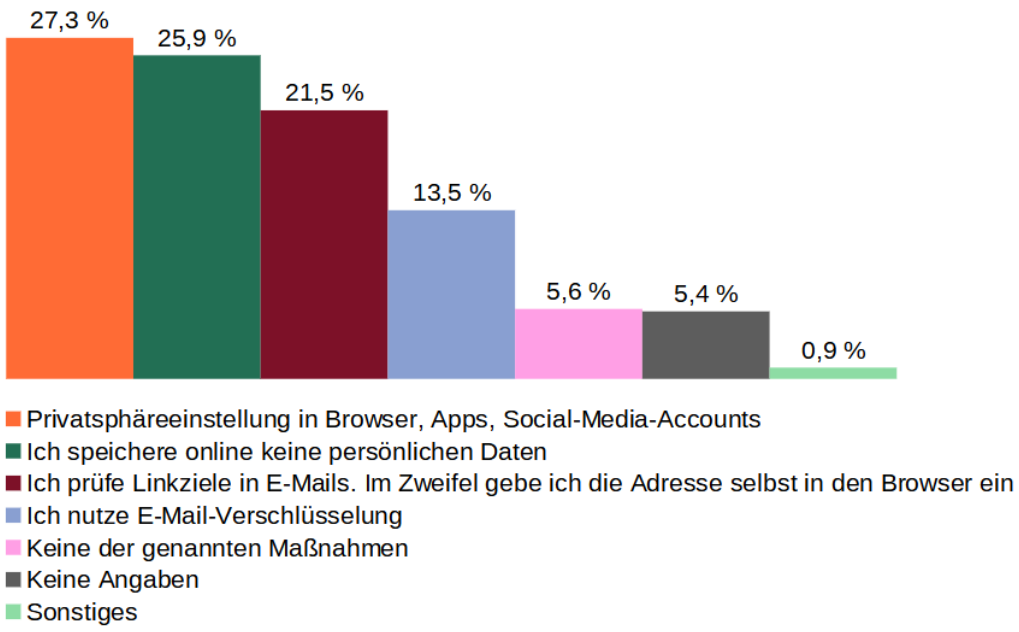


Abbildung 1: Maßnahmen zum Schutz der persönlichen Daten

Antworten auf die Frage „Was tun Sie, um Ihre persönlichen Daten im Internet zu schützen?“ (Daten aus [6])

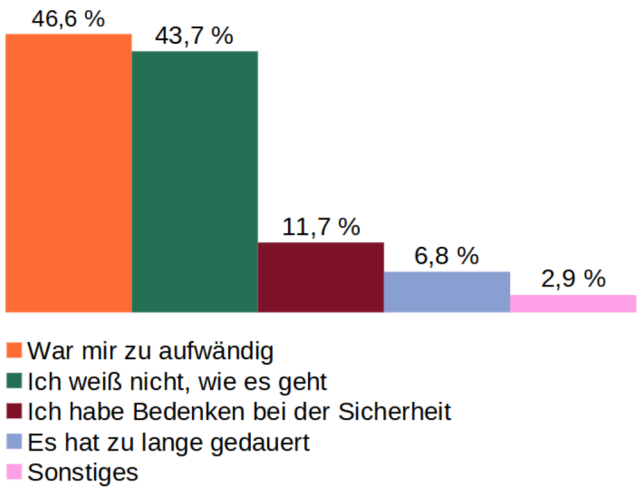


Abbildung 2: Gründe gegen Verschlüsselung

Antworten auf die Frage „Warum haben Sie sich bisher keine sichere Ende-zu-Ende Verschlüsselung installiert?“ (Daten aus [6])

Zielsetzung

Das übergeordnete Ziel, zu dessen Erreichung die vorliegende Arbeit beitragen soll, ist eine stärkere Verbreitung der Nutzung der Verschlüsselung mithilfe von OpenPGP-Schlüsseln. Dafür soll der WKD-Standard weiter verbreitet werden, da dieser einen Teil beim Prozess des Verschlüsselns vereinfacht. Einige Produkte enthalten die WKD-Funktion bereits. Leider wird durch das Testen verschiedener Produkte, das in einem der folgenden Kapitel erläutert wird, deutlich, dass viele Produkte nicht kompatibel zur aktuellen Version des WKD-Standards sind. Dazu kommt, dass die WKD-Funktion teilweise eine komplizierte Einrichtung benötigt, um verwendet werden zu können. Zuletzt wird sichtbar, dass die Funktion häufig nicht benutzerfreundlich verwendet werden kann.

Einerseits ist die Aufgabe dieser Arbeit, herauszufinden, welche Maßnahmen dafür sorgen, dass die Nutzung von WKD steigt. Andererseits soll aus den oben genannten Gründen ermittelt werden, wie eine bessere Grundlage für die Implementierung des WKD-Standards geschaffen werden kann. Diese soll ermöglichen, dass es für Entwickler:innen einfacher ist, WKD so zu implementieren, dass die Verwendung von WKD dabei eine gute Usability bietet. Damit soll am Ende auch erreicht werden, dass direkte Netzwerkeffekte eintreten. Wie in [8] erläutert wird, wird der Nutzen eines Netzwerks für eine einzelne Person desto größer, je mehr Personen zu diesem Netzwerk gehören. Wenn mehr Personen die WKD-Funktion verwenden, wird auch das Netzwerk, bestehend aus möglichen Kommunikationsteilnehmer:innen, die verschlüsselte Mails verschicken können, größer. Es wird erwartet, dass dadurch die Nutzung von WKD attraktiver wird und ein Kreislauf entsteht, in dem die Zahl der Nutzer:innen immer weiter steigt.

Aufbau der Arbeit

Das folgende Kapitel beschreibt zunächst den Hintergrund und die Funktionsweise von Web Key Directories. Außerdem umreißt es das schon länger bekannte Vertrauensmodell, welches als „Web of Trust“ bezeichnet wird sowie ein Modell, das im Rahmen der Entwicklung von WKD entstanden ist. Da die Usability in Produkten eine wichtige Rolle in dieser Arbeit spielt, bilden auch die von Jakob Nielsen entwickelten Heuristiken zur Usability einen Teil des zweiten Kapitels.

Im dritten Kapitel geht es darum, Anwendungsfälle zu entwickeln, die beschreiben, wie Benutzer:innen ohne großen Aufwand eine verschlüsselte Mail schreiben oder die Signatur einer Mail überprüfen. Auf der Basis dieser Anwendungsfälle ergeben sich Kriterien, die dazu dienen, WKD so in Produkten zu implementieren, dass dabei eine gute Usability erreicht wird. Da in dieser Arbeit auch verschiedene Produkte getestet werden, zeigen einige Abschnitte des dritten Kapitels, welche Produkte für die Tests ausgewählt werden und wie die Produkte getestet werden. Neben den Tests stellt das Kapitel weitere Maßnahmen vor, die ebenfalls im Rahmen dieser Arbeit durchgeführt werden, wie z.B. eine Implementierung für eines der ausgewählten Produkte.

Die Beobachtungen in den Tests füllen das vierte Kapitel, während sich die weiteren durchgeführten Maßnahmen das fünfte Kapitel teilen. Es folgt eine Diskussion im sechsten Kapitel, welche die Erkenntnisse der Arbeit zusammenfasst. Ein Ausblick mit möglichen weiteren Schritten, welche in weiteren Arbeiten in Angriff genommen werden können, schließt die Arbeit ab.

Hinweise zur Arbeit

In dieser Arbeit werden unter anderem Befehle und Kommandos für die Konsole mithilfe einer `weiteren Schriftart` abgegrenzt. Domains werden durch eine *kursive Schrift* hervorgehoben. Zudem werden die englischen Begriffe aus der Versionsverwaltung, wie z.B. Fork und Pull Request, verwendet, da sie unter Entwickler:innen geläufiger sind, als die deutschen Übersetzungen dieser Begriffe.

In dieser Arbeit taucht immer wieder der Begriff Schlüssel auf. Wenn nicht anders beschrieben, handelt es sich um einen öffentlichen Schlüssel. Um den Text nicht zu stark aufzublähen, wurde an vielen Stellen auf das Adjektiv verzichtet.

2. Forschungsstand und theoretische Grundlage

Web Key Directories

In diesem Abschnitt soll das Verfahren Web Key Directories (WKD) und dessen Funktionsweise vorgestellt werden. Wie das Verfahren server- und clientseitig implementiert werden soll, wird von Werner Koch, dem Hauptentwickler von GnuPG, in einem Entwurf für einen neuen Standard beschrieben. Zu Beginn dieser Arbeit ist der zwölfte der aktuelle Entwurf. Deshalb bezieht sich die Arbeit auf diesen. WKD bietet eine einfache Möglichkeit, an öffentliche Schlüssel zu kommen. Dabei wird eine URL zusammengestellt, die daraufhin aufgerufen wird. Schließlich kann der öffentliche Schlüssel der dazugehörigen Mail-Adresse heruntergeladen werden. Voraussetzung dafür ist, dass die Mail-Adresse bekannt ist und der Schlüssel auf einen WKD-Server hochgeladen wurde. Zur Erstellung der URL gibt es zwei Möglichkeiten, die als Advanced Method und Direct Method bezeichnet werden. In dieser Arbeit werden die Namen der beiden Methoden übernommen und nicht übersetzt. Bis zum sechsten Entwurf des Standards gab es nur die Direct Method. Im siebten Entwurf wurde die Advanced Method ergänzt. Der Grund für die Einführung der Advanced Method ist, dass sie besser geeignet ist, wenn mehrere Mail-Domains angeboten werden [9].

Im WKD-Verfahren wird eine URL erstellt, damit ein WKD-Server kontaktiert wird, auf dem sich der öffentliche Schlüssel einer Mail-Adresse befindet. Die URL muss dabei die Information enthalten, um welche Mail-Adresse es sich handelt, sodass der richtige Schlüssel vom Server heruntergeladen wird. Verzeichnisse eines Rechners haben jedoch nicht denselben möglichen Zeichenumfang, wie es beim lokalen Teil einer Mail-Adresse der Fall ist. Aus diesem Grund muss der lokale Teil umgewandelt werden, bevor er in die URL eingesetzt wird. Dabei werden zuerst alle ASCII-Zeichen in Kleinbuchstaben umgewandelt. Es folgen das Erstellen eines Hash-Werts mithilfe des SHA-1-Verfahren und eine Kodierung mit dem Verfahren Z-Base-32, bis zuletzt eine Zeichenkette aus 32 Bytes das Ergebnis ist. Im nächsten Schritt unterscheiden sich Direct Method und Advanced Method. Bei der Advanced Method ist die URL so aufgebaut:

```
https://openpgpkey.domain/.well-known/openpgpkey/domain/hu/
Hash-Wert?l=lokaler_Teil
```

Für die Mail-Adresse jdoe@wkdtest.ntvtn.de sieht die URL beispielsweise so folgendermaßen aus:

```
https://openpgpkey.wkdtest.ntvtn.de/.well-known/openpgpkey/
wkdtest.ntvtn.de/hu/4pkteh5be3b3shuzepabpupwbi95cirz?l=jdoe
```

Bei der Direct Method fehlt die Subdomain *openpgpkey* und die Domain kommt in der URL nur einmal am Anfang vor. Somit ist die URL bei der Direct Method wie folgt aufgebaut:

```
https://domain/.well-known/openpgpkey/hu/Hash-
Wert?l=lokaler_Teil
```

Im Beispiel mit der Adresse jdoe@wkdtest.ntvtn.de sieht die konkrete URL bei Anwendung der Direct Method dann so aus:

```
https://wkdtest.ntvtn.de/.well-known/openpgpkey/hu/
4pkteh5be3b3shuzepabpupwbi95cirz?l=jdoe
```

Die Abbildung 3 zeigt ein Diagramm, in dem die Funktionsweise von WKD in groben Schritten zusammengefasst wird, beginnend mit der Eingabe einer Mail-Adresse durch die Benutzer:innen.

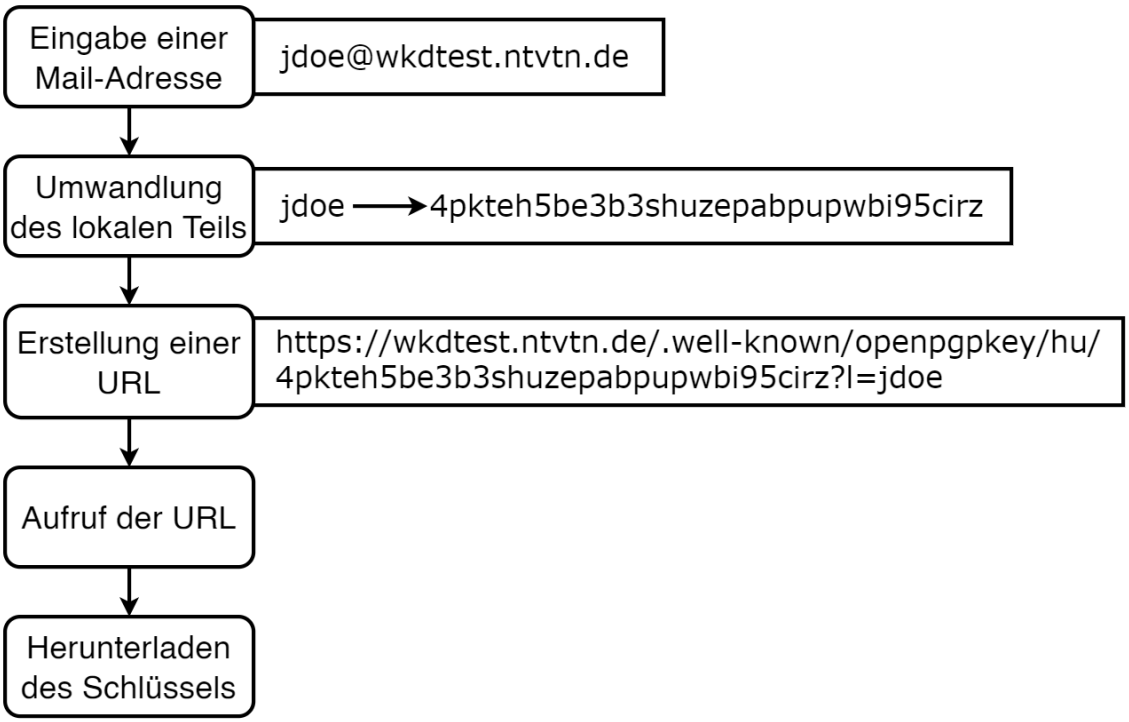


Abbildung 3: Funktionsweise von WKD

Das Diagramm stellt die Funktionsweise von WKD am Beispiel der Direct Method dar.

Im Entwurf wird vorgegeben, dass die Advanced Method verwendet werden soll, wenn die Subdomain *openpgpkey* existiert. Nur wenn dies nicht der Fall ist, kann die Direct Method verwendet werden. Wenn die Advanced Method beim Beziehen eines Schlüssels erfolglos ist, ist dies kein Grund für die Verwendung der Direct Method.

Was beim Betrachten der URL, die in der Direct bzw. Advanced Method erstellt wird, auffällt, ist, dass die Domain der Mail-Adresse auch diejenige ist, die am Ende aufgerufen wird. Das bedeutet, dass es notwendig ist, dass

die dazugehörigen Mail-Provider einen WKD-Server anbieten, der Schlüssel für diese Domain speichert. Sonst gibt es keine Möglichkeit, den Schlüssel für eine Mail-Adresse zu erlangen.

Da diese Schlüssel den Nutzer:innen gehören, die ihren Account bei den Mail-Provider besitzen, wissen letztere, dass ein Schlüssel zu einer bestimmten Person gehört. Außerdem müssen Unternehmen, die Mail-Accounts anbieten, Sicherheitsstandards erfüllen, während bei Privatpersonen, die einen Schlüssel-Server betreiben, keine Garantie für die Sicherheit gegeben ist. Aus diesen Gründen besitzen Schlüssel, die per WKD bezogen werden, zumindest eine grundlegende Vertrauenswürdigkeit.

Vertrauensmodell für öffentliche Schlüssel

Da öffentliche Schlüssel nicht immer aus erster Hand erhalten werden, ist es wichtig, überprüfen zu können, wie vertrauenswürdig ein Schlüssel ist. Das Thema Vertrauen spielt bei der Verschlüsselung also eine große Rolle. Aus diesem Grund wird von GnuPG ein Vertrauensmodell verwendet [10], welches sich aus zwei Teilen zusammensetzt: Zum Einen können User:innen festlegen, wie sehr sie den Besitzer:innen anderer Schlüssel vertrauen. Im Folgenden wird dafür der englische Begriff „Owner Trust“ verwendet. Beim Owner Trust gibt es verschiedene Abstufungen, mit denen das Vertrauen ausgedrückt werden kann. Diese haben Auswirkungen darauf, wie stark einem Schlüssel vertraut werden kann, der von diesen Besitzer:innen signiert wurde. Die Abstufungen lauten: „Unknown“, „None“, „Marginal“ und „Full“.

Der andere Teil des Modells besteht aus der Validität eines Schlüssels. Diese gibt an, wie sehr Anwender:innen einem Schlüssel vertrauen können. Hier gibt es die Möglichkeit, dass ein Schlüssel nicht, marginal oder vollständig valide ist. Angenommen, Jane hätte einen Schlüssel von Harry signiert. Damit sagt sie aus, dass Harry diesen Schlüssel wirklich besitzt und sie diesem Schlüssel vertraut. Das führt dazu, dass dieser Schlüssel vollständig valide ist. Dies ist gleichzeitig die höchste Stufe der Validität. Entscheidet sich Jane dazu, dem Schlüssel von Harry nicht zu signieren, aber Harry voll zu vertrauen (full) und Harry unterschreibt den Schlüssel von Kim, so ist der Schlüssel von Kim für Jane ebenfalls vollständig valide (Szenario 2 in Abbildung 4).

Andere Möglichkeiten für valide Schlüssel sind, dass ein Schlüssel von drei anderen Schlüsseln signiert wurde, denen eine Person marginal vertraut (Szenario 3 in Abbildung 4) oder, dass der Weg zu einem Schlüssel nur über signierte Schlüssel führt, dieser Schlüssel vom letzten Schlüssel signiert wurde und der Weg nicht länger als fünf Schlüssel beträgt (Szenario 1 in Abbildung 4). Zu beachten ist, dass die Werte, z.B. wie viele marginale Schlüssel einen anderen Schlüssel signieren müssen, damit dieser vollständig valide ist, von den Einstellungen in GnuPG abhängen.

Am Ende entsteht durch mehrere Signaturen von verschiedenen Personen ein Netzwerk, in dem nachvollzogen werden kann, wer wem vertraut und schlussendlich auch, welcher Schlüssel für eine beliebige Person wie vertrauenswürdig ist. Dieses Netzwerk wird auch als „Web of Trust“ bezeichnet.

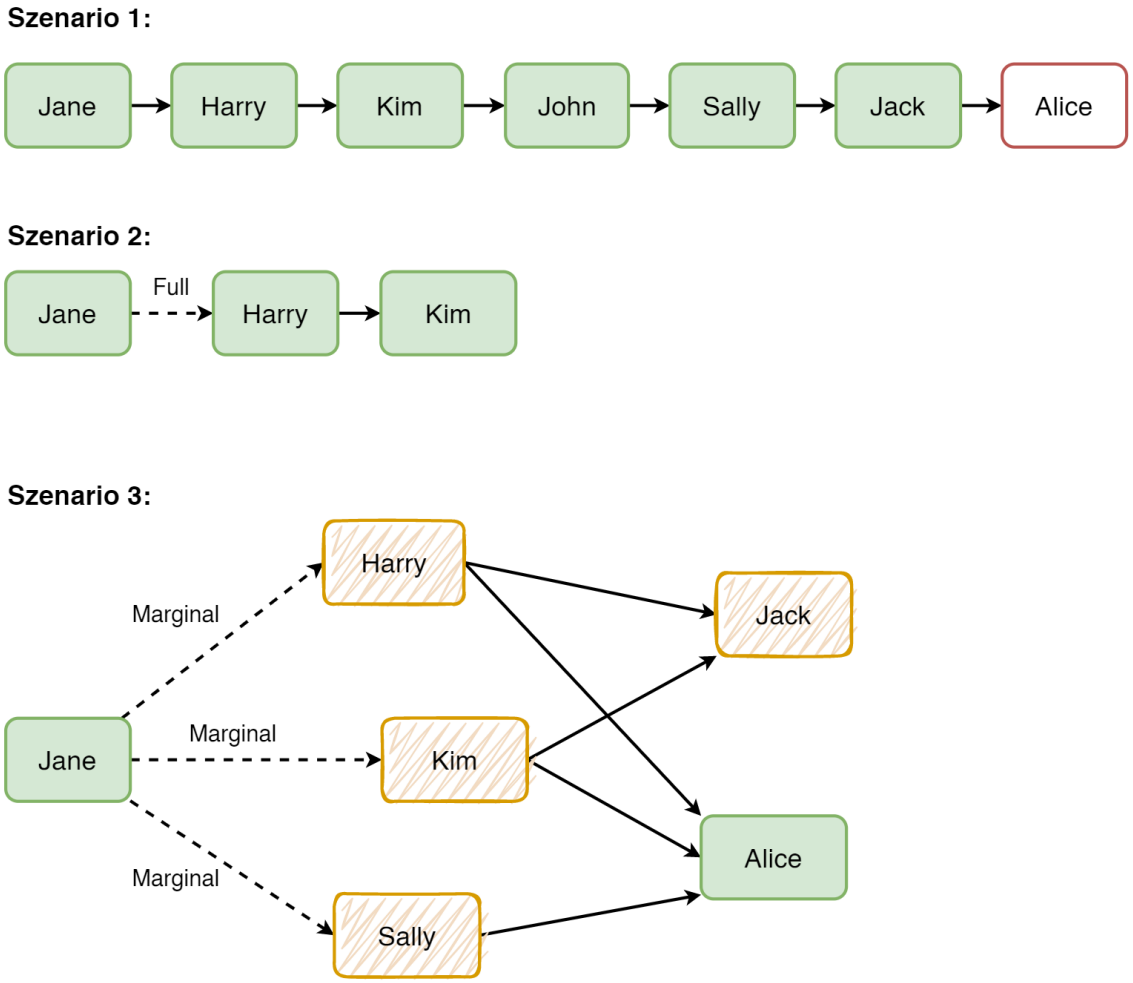


Abbildung 4: Beispiele für das Web of Trust

Es werden drei verschiedene Szenarien gezeigt. Dabei wird die Validität der Schlüssel aus Sicht der Person Jane dargestellt. Pfeile mit einer durchgehenden Linie bedeuten, dass der Schlüssel einer Person signiert wurde. Wenn die Linie eines Pfeils gestrichelt ist, wurde das Owner Trust einer Person verändert. Der neue Wert befindet sich dann am jeweiligen Pfeil. Der Kasten, der den Namen einer Person umgibt, zeigt die Validität eines Schlüssels an. Ein grün ausgefüllter Kasten steht für einen vollständig validen, ein Kasten mit rotem Rand für einen nicht-validen und ein Kasten, der eine gelbe Schraffur enthält für einen marginal validen Schlüssel.

Im Rahmen der Entwicklung des WKD-Standards wurde ein weiteres Vertrauensmodell kreiert [11]. Dieses verwendet Informationen aus dem Web of Trust und ergänzt diese mit weiteren Aspekten, um einschätzen zu können, wie vertrauenswürdig ein Schlüssel ist. So kommt hinzu, dass auch berücksichtigt wird, ob ein Schlüssel immer wieder über einen längeren Zeitraum verwendet wird. Über einen längeren Zeitraum einen Man-in-the-

Middle-Angriff auszuführen, ist kostspieliger und unwahrscheinlicher. Wenn der Schlüssel häufiger verwendet wird, ist die Wahrscheinlichkeit höher, dass ein Angriff auffällt.

Es spielt auch eine Rolle, aus welcher Quelle ein erhaltener Schlüssel stammt. Wie schon erklärt, sind Schlüssel vom WKD-Server eines Mail-Providers grundlegend vertrauenswürdig. Dieser Aspekt fließt im neuen Modell mit ein. Wenn ein Schlüssel per WKD bezogen wird, landet dieser automatisch in der dritten von fünf Stufen der Vertrauenswürdigkeit.

Usability Heuristiken

Es ist heutzutage nicht mehr nur wichtig, was im Hintergrund von Softwareprodukten passiert. Die grafische Oberfläche hat deutlich an Bedeutung zugenommen. Somit ist eine gute Usability ein wichtiger Bestandteil, wenn ein Produkt erfolgreich sein soll. Jakob Nielsen hat sich bereits 1993 mit dem Testen der Usability auseinandergesetzt und zehn Heuristiken aufgestellt, die dabei unterstützen, verbesserungswürdige Aspekte bei der Usability von Produkten zu finden [12]. Auf der Internetseite der Nielsen Norman Group hat Nielsen die Heuristiken in einer aktualisierten Version veröffentlicht [13].

Die erste Heuristik „Visibility of system status“ [14] bedeutet, dass Anwender:innen über den aktuellen Status der Anwendung informiert werden sollten. Dazu gehört z.B., dass der Fortschritt beim Laden einer Applikation angezeigt wird, aber auch, dass Feedback gegeben wird, wenn Benutzer:innen eine Aktion durchführen. Ein konkretes Beispiel ist das Eingabefeld für ein Passwort, bei dem angezeigt wird, ob noch bestimmte Zeichengruppen fehlen (Abbildung 5). Beim Einhalten dieser Heuristik haben Anwender:innen das positive Gefühl, alles unter Kontrolle zu haben und Ergebnisse von Aktionen vorhersagen zu können.

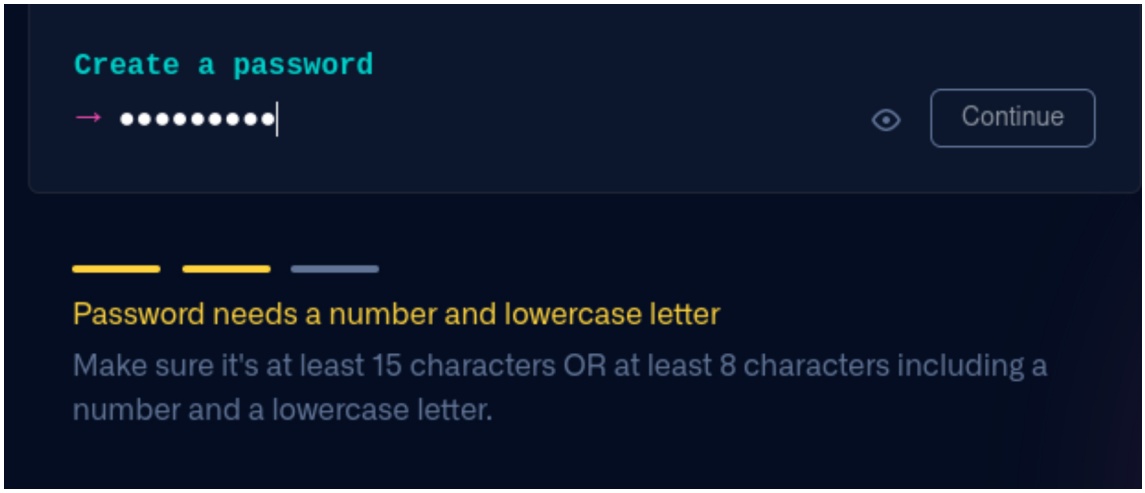


Abbildung 5: Beispiel für die Heuristik „Visibility of system status“

Bei der Registrierung auf der Seite github.com wird angezeigt, wie viele oder welche Zeichen noch fehlen.

Ergänzend zu dieser Heuristik wurde eine weitere mit der Bezeichnung „User control and freedom“ [15] aufgestellt. Die Bezeichnung verrät bereits, dass auch dadurch das Gefühl der Kontrolle verstärkt wird. Konkret geht es darum, dass Anwender:innen immer die Möglichkeit haben sollten, eine Aktion abzubrechen oder rückgängig zu machen. Abbildung 6 zeigt einen Screenshot aus der Anwendung LibreOffice Writer, in der Eingaben im Dokument rückgängig gemacht werden können.

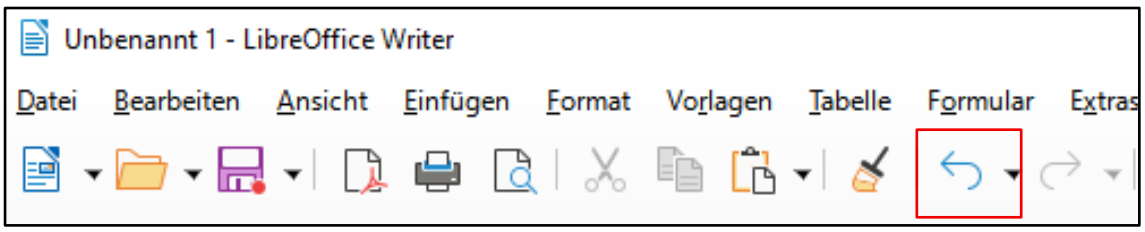


Abbildung 6: Beispiel für die Heuristik „User control and freedom“

Die Anwendung LibreOffice Writer ermöglicht es Nutzer:innen jederzeit, ihre Aktionen rückgängig zu machen.

Für Benutzer:innen ist es eine Unterstützung bei der Verwendung einer Software, wenn die Heuristik „Match between system and the real world“ [16] eingehalten wird. Die Heuristik bedeutet, dass die Begriffe in einer Anwendung verwendet werden, die auch die Benutzer:innen gebrauchen, sodass sie diese Begriffe sofort verstehen und nicht dazu gezwungen sind, sie nachzuschauen. Ein weiterer Bereich betrifft die Bedienelemente einer Anwendung. Auch hier können Zusammenhänge zu Gegenständen aus der Realität geschaffen werden, damit Benutzer:innen sofort verstehen, wie sie die Elemente verwenden müssen. Vor einigen Jahren wurden bspw. Buttons viel stärker so gestaltet, dass sie eine Illusion von Dreidimensionalität geschaffen haben. Auf diese Weise wurden sie wie Knöpfe wahrgenommen, wie sie aus dem Alltag bekannt waren. Benutzer:innen wussten deshalb, dass sie die Buttons durch einen Klick betätigen konnten.

Zur Erleichterung bei der Bedienung trägt zusätzlich die Heuristik „Consistency and standards“ bei [17]. Bei einer konsistenten Gestaltung sehen Elemente der grafischen Oberfläche immer gleich aus, unabhängig an welcher Stelle sie verwendet werden. Das führt dazu, dass Nutzer:innen nicht jedes Mal neu herausfinden müssen, wofür ein Element zuständig ist. Auch das Einhalten von allgemeinen Konventionen hilft bei der Wiedererkennung von Elementen. So werden üblicherweise sogenannte Radiobuttons verwendet, wenn nur eine Option und Checkboxen, wenn mehrere Optionen gleichzeitig ausgewählt werden können. Weichen Anwendungen von dieser Konvention ab, führt dies zu Irritationen. Konsistenz und Standards sind aber auch bei Sprache, Gesten etc. zu beachten. Ein Beispiel für diese Heuristik ist die Kombination der Tasten „Strg“ und „S“ auf der Tastatur zum Speichern einer Datei. Diese Kombination wird von vielen Anwendungen unterstützt und hilft den Benutzer:innen, effizient zu arbeiten, weil sie die Tasten der Kombination mehr oder weniger automatisch drücken, weil sie es gewohnt sind.

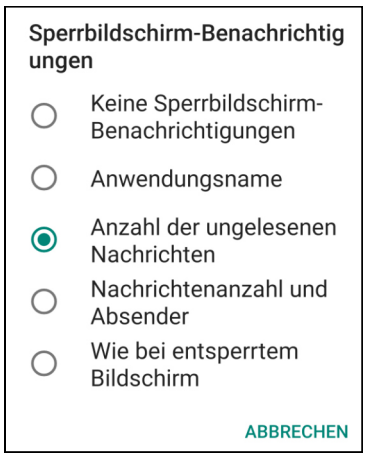


Abbildung 7: Beispiel für die Heuristik „Consistency and standards“

Werden Radiobuttons, wie hier in K9Mail, angezeigt, wissen Benutzer:innen, dass sie nur eine Option gleichzeitig auswählen können.

Da das Wiedererkennen leichter ist als das Erinnern, gibt es dazu eine eigene Heuristik „Recognition rather than recall“ [18]. Nicht nur Konsistenz ist dabei hilfreich, sondern auch, wenn Informationen im GUI angezeigt werden. Ein Beispiel für diese Heuristik ist der Verlauf, der im Browser gespeichert werden kann (Abbildung 8). Wenn Benutzer:innen sich nicht mehr an den Namen einer Seite, aber an wenige Details erinnern, kann es beim Wiederfinden einer Seite hilfreich sein, den Verlauf zu sehen. Durch das Betrachten des Seitentitels oder der URL einer Seite ist es wahrscheinlich, dass Benutzer:innen eine Seite wiedererkennen und wissen, dass sie die gesuchte Seite ist. Deshalb ist es für sie in verschiedenen Fällen sehr nützlich, solche Gedankenstützen zu besitzen.

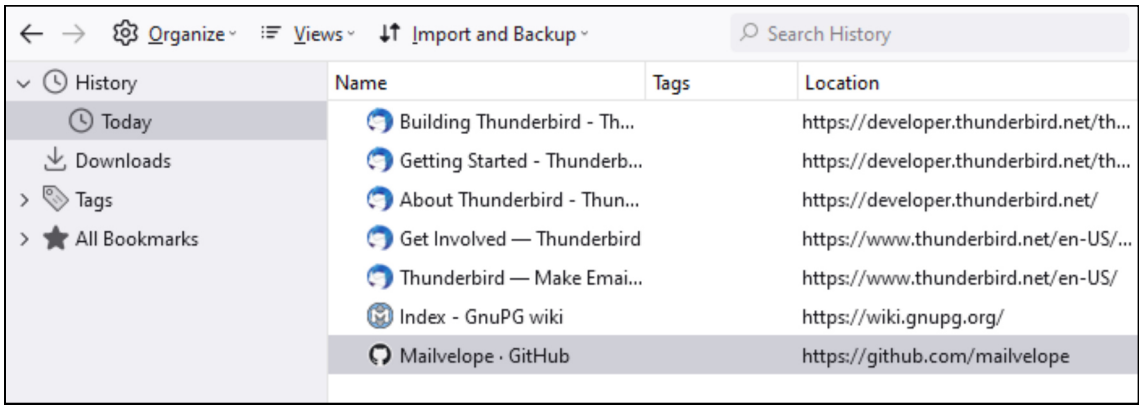


Abbildung 8: Beispiel für die Heuristik „Recognition rather than recall“

Wenn der Verlauf im Browser (hier: Firefox) gespeichert wird, fällt es den Nutzer:innen deutlich leichter, Seiten wieder zu finden, die sie schon mal besucht haben.

Eine weitere Heuristik – „Error prevention“ [19] – beschäftigt sich mit Fehlern bzw. wie Designer:innen und Entwickler:innen sie schon durch die Gestaltung einer Anwendung verhindern können. Weit verbreitet ist die Bestätigung beim Schließen eines Programms, wenn eine Datei verändert, aber noch nicht gespeichert wurde (Abbildung 9). Diese Bestätigung bewahrt Benutzer:innen davor, dass ihre Arbeit verloren geht. Aber auch, wenn Eingaben von Anwender:innen, z.B. eine IBAN, formatiert werden, damit sie besser gelesen werden können, kann diese Maßnahme dafür sorgen, dass den Benutzer:innen Fehler bei der Eingabe unterlaufen.

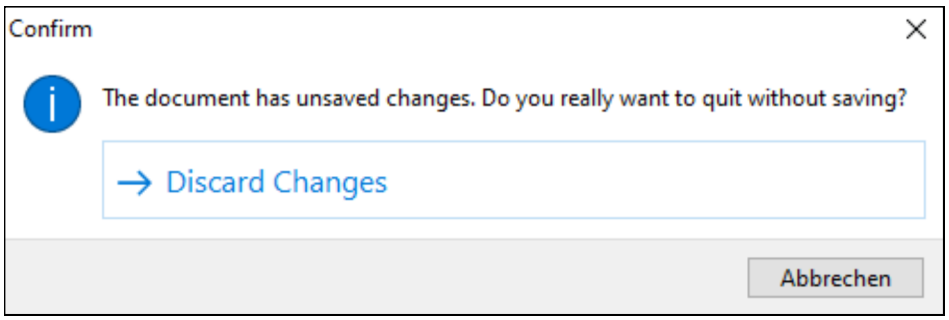


Abbildung 9: Beispiel für die Heuristik „Error prevention“

Wenn Benutzer:innen den Button zum Schließen der Anwendung draw.io betätigen, eine veränderte Datei aber noch nicht gespeichert wurde, erscheint eine Meldung, die sie erst bestätigen müssen, bevor das Programm endgültig geschlossen wird.

Wenn wider Erwarten Fehler auftreten, greift die Heuristik „Help users recognize, diagnose, and recover from errors“ ein [12, 13]. Diese besagt unter anderem, dass Fehlermeldungen in einfacher Sprache gehalten werden und nicht nur das Problem erwähnen, sondern auch Lösungen anbieten sollen. Wichtig ist außerdem, dass diese Meldungen auch als solche erkannt werden sollten, also dementsprechend gestaltet werden sollen. Klassisch sind dabei eine rote Farbe oder ein Symbol mit einem Ausrufezeichen. Affinity Publisher, ein Programm zum Erstellen von Layouts für Bücher, Broschüren etc., bietet eine Checkliste an, in der auch Fehler angezeigt werden. In einigen Fällen können die Fehler direkt mithilfe eines Buttons neben der Meldung behoben werden (Abbildung 10).

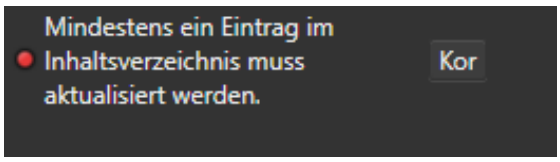


Abbildung 10: Beispiel für die Heuristik „Help users recognize, diagnose, and recover from errors“

Im Layout-Programm Affinity Publisher gibt es eine Checkliste, die anzeigt, wenn etwas nicht in Ordnung ist. Im abgebildeten Fall fehlt ein Eintrag im Inhaltsverzeichnis. Der Button auf der rechten Seite kann verwendet werden, um den Fehler direkt an dieser Stelle zu beseitigen. Es ist für Benutzer:innen nicht notwendig, erst zu der betroffenen Seite zu navigieren, um dort den Fehler zu beheben.

Bei umfangreichen Anwendungen kann es notwendig sein, dass eine Dokumentation erstellt wird, mit deren Hilfe Benutzer:innen herausfinden können, wie sie ihre Aufgaben erfolgreich erledigen können. Die Dokumentation sollte einfach durchsucht werden können und Anleitungen in dieser Dokumentation so geschrieben sein, dass sie leicht befolgt werden können. Damit sie für Benutzer:innen leichter zu lesen sind, sollten keine riesigen Textblöcke verwendet werden. Diese Aspekte sind in der Heuristik „Help and documentation“ zu finden [20].

Damit Anwender:innen schnell herausfinden können, welche Schritte sie als nächstes durchführen müssen, um ihre Ziele zu erreichen, sollte eine Applikation wenige oder gar keine unnötigen Elemente enthalten. Dies entspricht der Heuristik „Aesthetik and minimalist design“ [12, 13]. Je mehr Elemente gleichzeitig sichtbar sind, desto eher gehen diejenigen unter, die

wirklich zum Erfolg einer Aufgabe beitragen. Für das schnelle Erreichen eines Ziels sollte zudem die Heuristik „Flexibility and efficiency of use“ [21] berücksichtigt werden. Wesentlich geht es bei dieser Heuristik darum, dass sogenannte „Beschleuniger“ eingebaut werden sollen, die für Personen nicht störend sind, die die Anwendung noch nicht lange verwenden, aber für fortgeschrittene Personen hilfreich sind. Zu diesen Beschleunigern gehören z.B. Tastenkombinationen für bestimmte Aktionen.

3. Erläuterung der Methodik

Erläuterung zu den Tests

Die Maßnahmen, die bei verschiedenen Produkten angewendet werden, entstammen den Phasen des Software-Engineerings. Unter anderem finden in dieser Arbeit Tests verschiedener Produkte statt. Weitere Abschnitte dieser Arbeit erläutern, welche Produkte einem Test unterzogen werden und wie die Testumgebung aufgebaut wird. Der Zweck der Tests ist, herauszufinden, ob die Produkte kompatibel zur letzten Version des WKD-Standards sind. Das bedeutet, dass die Tests überprüfen sollen, ob Direct und Advanced Method verwendet werden und ob die Methoden auch nur dann ausgeführt werden, wenn die richtigen Bedingungen gegeben sind. Wenn also die Subdomain *openpgpkey* existiert, sollten Produkte nicht versuchen, mithilfe der Direct Method an öffentliche Schlüssel zu gelangen.

Neben der Überprüfung, ob die Spezifikation für WKD eingehalten wird, soll festgehalten werden, welche Schritte notwendig sind, um WKD in einem Produkt zu aktivieren, falls WKD nicht standardmäßig nach der Installation aktiviert ist. Auch die Verwendung der WKD-Funktion ist ein Teil der Betrachtung. In beiden Fällen wird getestet, wie benutzerfreundlich die Vorgänge sind.

Requirements Engineering: Anwendungsfälle

Das Requirements Engineering soll in dieser Arbeit dazu dienen, festzulegen, was notwendig ist, damit der WKD-Standard nicht beliebig implementiert wird, sondern dabei auch ein gewisses Level bei der Usability geboten wird. Die Erstellung von Anwendungsfällen mithilfe einer Schablone für Anwendungsfälle nach Cockburn [22] führt dazu, dass festgehalten werden kann, ob die getesteten Produkte es ermöglichen, dass Anwender:innen ihre Aufgaben erfüllen können oder nicht. Die angelegten Anwendungsfälle (siehe Anhang) beschreiben die zwei wichtigen Fälle, in denen das Beziehen von Schlüsseln per WKD notwendig sein kann. Dazu gehören das Schreiben einer verschlüsselten Mail und das Validieren der Signatur einer Mail. Je nach dem, wie die Erfahrungen und Bedürfnisse der Anwender:innen sind, kann es sein, dass Anwender:innen zusätzliche Informationen benötigen, z.B. ob ein Schlüssel vertrauenswürdig ist.

In die Anwendungsfälle fließt der Grundgedanke des Buchs „Don’t Make Me Think“ von Steve Krug [23] mit ein: Wenn eine gute Usability angeboten werden soll, ist es wichtig, dass ein Produkt den Anwender:innen einen großen Teil des Nachdenkens bei der Bedienung abnimmt, damit die Anwender:innen sich auf ihre Aufgabe konzentrieren können. Wenn Benutzer:innen viel über Bedienelemente nachdenken müssen, dauert es

lange, bis sie herausfinden, wie sie ihre Aufgabe vollenden können oder es führt dazu, dass sie Fehler machen. Am Ende kann eine umständliche Bedienung zur Folge haben, dass Anwender:innen sich auf die Suche nach einem anderen Produkt begeben. Die erarbeiteten Anwendungsfälle berücksichtigen aus diesem Grund, dass Anwender:innen nicht viel Aufwand haben sollten, um die jeweilige Aufgabe zu vollenden.

Im ersten Anwendungsfall (AF1, Anhang) möchte eine beliebige Person, hier als Person A bezeichnet, eine vertrauliche Nachricht versenden. Die Person A stammt aus keiner speziellen Gruppen, da die Verschlüsselung für so viele Menschen wie möglich zugänglich gemacht werden sollte. Da die Nachricht vertraulich ist, darf keine andere Person außer der Person B, an die diese Nachricht verfasst wird, diese Nachricht lesen können. Voraussetzung für diesen Anwendungsfall ist, dass die Person A, bereits einen Mail-Client installiert hat, der das Verschlüsseln nach OpenPGP ermöglicht und öffentliche Schlüssel automatisch per WKD bezieht. Person B hat bereits einen öffentlichen Schlüssel für ihre Mail-Adresse erstellt und diesen Schlüssel auf einen WKD-Server hochgeladen. Person A besitzt diesen Schlüssel jedoch noch nicht. Eine verschlüsselte Mail wird versandt, wenn der Anwendungsfall erfolgreich ist. Ein Misserfolg kann bedeuten, dass eine unverschlüsselte Mail oder gar keine Mail verschickt wird. Eine weitere Situation bei einem fehlgeschlagenen Anwendungsfall entsteht, wenn Person B die verschlüsselte Mail nicht entschlüsseln kann, weil z.B. ein falscher öffentlicher Schlüssel verwendet wird, um die Nachricht zu verschlüsseln.

Der Ablauf in diesem Anwendungsfall sieht folgendermaßen aus: Person A öffnet ihren Mail-Client, navigiert zum Bereich, in dem eine Mail geschrieben werden kann und gibt dort die Adresse von Person B, den Betreff und die Nachricht ein. Der letzte Schritt beinhaltet das Absenden der Mail. Da Personen bzw. Personengruppen existieren, die Nachrichten mithilfe unterschiedlicher Sicherheitsstufen klassifizieren oder ein stärkeres Sicherheitsbedürfnis haben, gibt es für den Anwendungsfall zwei Erweiterungen. Eine dieser Erweiterungen geht davon aus, dass Person A wissen möchte, ob der verwendete Schlüssel vertrauenswürdig ist oder nicht. In der zweiten Erweiterung möchte Person A prüfen, wie vertrauenswürdig der Schlüssel ist, das heißt, sie möchte verschiedene Stufen der Vertrauenswürdigkeit unterscheiden können.

Im zweiten Fall (AF2, Anhang) geht es darum, dass eine beliebige Person A herausfinden möchte, ob eine Nachricht wirklich von Person B stammt, deren Mail-Adresse zum Versenden der Nachricht verwendet wurde. Bei Nachrichten, deren Verlust oder Offenlegung das Potential zu massiven, negativen Auswirkungen haben, ist es mit großer Wahrscheinlichkeit wichtig, zu wissen, ob die Person, die hinter einer Mail-Adresse vermutet wird, tatsächlich diese Nachricht geschrieben hat. Ein Beispiel ist der Auftrag zur Ausführung einer finanziellen Transaktion einer großen Summe. Die Vorbedingungen für diesen Anwendungsfall entsprechen denen für den

zuerst genannten Fall. Der Erfolg hängt davon ab, ob die Signatur der erhaltenen Mail überprüft werden kann oder nicht.

Die durchzuführenden Schritte im zweiten Anwendungsfall werden ergänzend in Abbildung 11 dargestellt. Person A öffnet zuerst den Mail-Client und navigiert zu dem Bereich, in dem sich alle erhaltenen Mails befinden. Dort öffnet sie eine beliebige signierte Mail und startet durch eine Interaktion die Signaturüberprüfung. Wenn die Überprüfung erfolgreich ist, schaut sich Person A das Ergebnis an und interpretiert dieses. Möchte Person A außerdem noch erfahren, wie vertrauenswürdig der Schlüssel ist, der für die Überprüfung verwendet wurde, schaut sie sich an, wie stark die Vertrauenswürdigkeit ist.

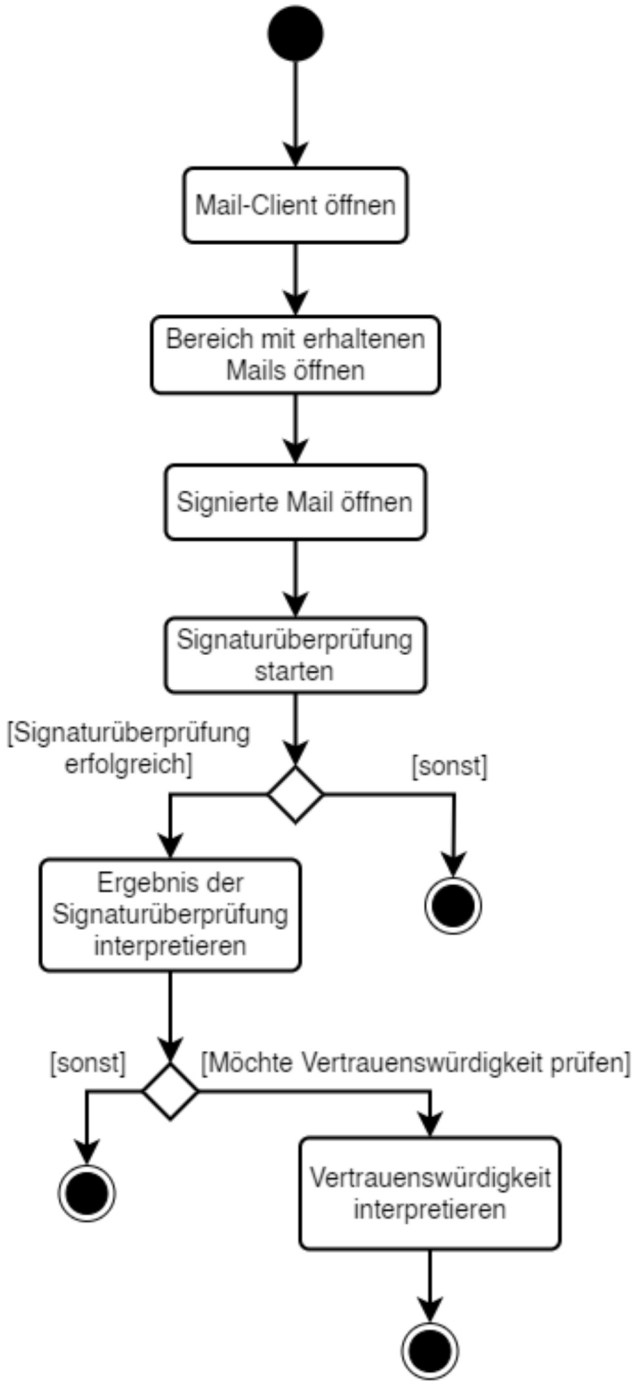


Abbildung 11: UML-Aktivitätsdiagramm zum zweiten Anwendungsfall

Dieser Fall beschreibt die Überprüfung der Signatur einer Mail

Kriterien für das Implementieren von WKD mit einer guten Usability

Nach dem Erstellen der Anwendungsfälle ist es nun möglich, Kriterien aufzustellen, die Produkte erfüllen sollen, um ein grundlegendes Maß an guter Usability zu erreichen. Die Kriterien werden dabei mit einem „K“ und einer Zahl nummeriert und farblich hervorgehoben. Im Kapitel, in dem die Testergebnisse vorgestellt werden, und in der Tabelle, die diese Ergebnisse zusammenfasst, werden diese Kürzel ebenfalls verwendet. Auf diese Weise können Leser:innen schneller zwischen der Erläuterung eines Kriteriums und den Stellen im Text, in denen ein Produkt anhand dieses Kriterium überprüft wird, hin- und herwechseln, um nachvollziehen zu können, wie die Ergebnisse in der zusammenfassenden Tabelle zustande kommen.

Zuallererst ist es wichtig, dass ein Produkt kompatibel zum letzten Entwurf des WKD-Standards ist (K1), denn sonst kann nicht garantiert werden, dass alle Schlüssel geholt werden können. Die WKD-Funktion sollte verwendet werden können, ohne dass Anwender:innen irgendwelche Vorbereitungen treffen müssen (K2). Sie sollten keine Einstellungen verändern oder andere Schritte ausführen müssen. Das Kriterium gilt auch als erfüllt, wenn sie ein Schlüsselpaar erstellen müssen, da ein Schlüsselpaar für eine verschlüsselte Kommunikation notwendig ist. Wenn sie aber z.B. noch einen Schlüssel oder eine Verschlüsselungsmethode auswählen müssen, erfüllt das jeweilige Produkt dieses Kriterium nicht mehr. Die Funktion zum Beziehen von Schlüsseln per WKD sollte direkt dort angeboten werden, wo eine Mail-Adresse beim Verfassen einer Mail eingegeben (K3) oder eine Signatur überprüft wird (K4). Das bedeutet, dass Benutzer:innen nicht zusätzliche Fenster oder Bereiche öffnen müssen, bevor sie die Funktion starten können. Noch besser ist es, wenn die WKD-Funktion nach dem Eingeben der Adresse automatisch ausgeführt wird (K5). Aus Gründen der Sicherheit wird dies nicht für die Signaturüberprüfung empfohlen. Das automatische Aufrufen der WKD-Funktion bei der Überprüfung könnte Angreifer:innen die Möglichkeit geben, zu erfahren, ob eine Mail-Adresse aktiv genutzt wird.

Zur Verbesserung der Usability ist es auch hilfreich, automatisch den Schlüssel für die Verschlüsselung auszuwählen, der die größte Vertrauenswürdigkeit besitzt, wenn sich mehrere Schlüssel für eine Mail-Adresse in einem Schlüsselbund befinden. Dadurch besteht keine Notwendigkeit, den Anwender:innen einen Dialog zur Auswahl zu zeigen, in dem sie die Entscheidung selbst treffen müssen. Gleichzeitig werden sie dabei unterstützt, sicher zu kommunizieren. Wenn also mehrere Schlüssel zur Verfügung stehen und nur einer davon per WKD bezogen wurde, während die anderen Schlüssel direkt importiert wurden und keine Informationen über die Vertrauenswürdigkeit anbieten, dann sollte der Schlüssel für die Verschlüsselung eingesetzt werden, der mithilfe der WKD-Funktion geholt wurde. Das Kriterium gilt sowohl für den Bereich zum Schreiben einer Mail (K6) als auch für den Bereich, in dem die Signaturüberprüfung stattfindet (K7).

Damit die Erweiterungen der Anwendungsfälle erfüllt werden können, soll die grafische Oberfläche anzeigen, ob ein Schlüssel keine Informationen über seine Vertrauenswürdigkeit besitzt oder ob dieser Schlüssel mehr oder weniger vertrauenswürdig ist. Dieser Aspekt gilt sowohl beim Eingeben einer Mail-Adresse beim Verfassen einer Mail (K8) als auch bei der Signaturüberprüfung (K9). „Mehr oder weniger vertrauenswürdig“ ist ein Schlüssel, wenn er mindestens eine grundlegende Vertrauenswürdigkeit besitzt. Im Web of Trust ist eine grundlegende Vertrauenswürdigkeit beispielsweise gegeben, wenn ein Schlüssel marginal valide ist. Die exakte Stufe, die ein Schlüssel bei diesem Kriterium erreichen muss, um für eine andere Anzeige in der GUI zu sorgen, ist bei diesen Kriterien irrelevant.

Das Kapitel zu den Vertrauensmodellen erläutert, dass es verschiedene Modelle gibt, mit deren Hilfe berechnet werden kann, wie vertrauenswürdig ein Schlüssel ist. Die hier vorgestellten Kriterien gelten jedoch für die Implementierung des WKD-Standards und es wird davon ausgegangen, dass die Schlüssel, die per WKD bezogen werden, grundlegend vertrauenswürdig sind. Aus diesem Grund verlangen die Kriterien, dass Produkte für solche Schlüssel visualisieren, dass diese das Mindestmaß an Vertrauenswürdigkeit erfüllen. Dies gilt für den Bereich zum Verfassen (K10) und für die Signaturüberprüfung (K11).

Um das nächste Kriterium zu erfüllen, sollen die Produkte mehrere Stufen der Vertrauenswürdigkeit darstellen. Bei der Implementierung dieser Anzeige soll darauf geachtet werden, dass sie dort platziert wird, wo die Anwender:innen ihre Aufgabe durchführen, also direkt in dem Bereich, in dem eine Mail verfasst wird (K12) oder die Signatur einer Mail überprüft wird (K13).

Weitere umzusetzende Maßnahmen

Die vorherigen Abschnitten dieses Kapitels enthalten mit dem Testen und dem Requirements Engineering bereits sehr wichtige Maßnahmen aus dem Software Engineering. Mögliche Ergebnisse beim Testen sind, dass Produkte den WKD-Standard entweder gar nicht oder nur unvollständig umsetzen. In diesem Fall gehört die Implementierung zu den Maßnahmen, die zusätzlich durchgeführt werden können. Da es aus zeitlichen Gründen nicht möglich ist, jedes Produkt mithilfe einer Implementierung zu verändern, werden stellenweise Verbesserungen für die Usability durch Vorschläge in Mailing-Listen und Foren angeregt.

Normalerweise gibt es beim Software Engineering zwei unterschiedliche Dokumentationen. Eine dient den Entwickler:innen, um einen Überblick über die Software und Beschreibungen zu Methoden etc. zu erhalten. In dieser Arbeit wird jedoch nur die zweite Dokumentation, die für Benutzer:innen gedacht ist, berücksichtigt. Für diese Gruppe von Personen entstehen Anleitungen im Rahmen der Arbeit, damit sie wissen, wie sie ein

Produkt auf den Einsatz der WKD-Funktion vorbereiten und wie sie diese Funktion verwenden können. Wie die Tests zeigen, ist die Usability nicht optimal und muss verbessert werden. Deshalb sind die Anleitungen eine Übergangslösung, bis die Verbesserungen stattfinden und erleichtern den Einstieg in die Verschlüsselung bzw. die Verwendung von WKD.

Aufbau der Testumgebung

Bevor verschiedene Produkte getestet werden können, ist eine geeignete Testumgebung notwendig, die dem aktuellen WKD-Standard entspricht. Durch die Umgebung wird deutlich, welche der Methoden von WKD eingesetzt werden – Advanced Method oder Direct Method. Das Unternehmen Intevation hat einen Server und einige Domains mit dazugehörigen Mail-Adressen bereitgestellt, sodass verschiedene Szenarien getestet werden können. Im Rahmen der Arbeit werden Schlüsselpaare für die Mail-Adressen generiert, die Verzeichnisse auf dem Server erstellt und die öffentlichen Schlüssel in diesen Verzeichnissen platziert. Tabelle 1 am Ende dieses Abschnitts listet die Mail-Adressen und nennt auch die Methode, mit der der dazugehörige Schlüssel bezogen werden soll.

Für die Mail-Adresse jdoe@wkdtest.ntvtn.de soll einmal das Vorgehen beim Erstellen des dazugehörigen Schlüssels erläutert werden. Zuerst erzeugt der Befehl **`gpg --generate-key`** ein neues Schlüsselpaar. Dabei wird als Name Jane Doe und als Mail-Adresse die eben genannte eingegeben. Der Sicherheitsforscher Mike Kuketz verrät, dass das Kommando **`gpg --with-wkd-hash --fingerprint`** mit der entsprechenden Mail-Adresse als Parameter den Hash-Wert für den lokalen Teil der Mail-Adresse erzeugt [24]. Der in Abbildung 12 markierte Text zeigt, wo der Hash-Wert abgelesen werden kann. Als Nächstes exportiert folgender Befehl den Schlüssel als Datei: **`gpg --no-armor --export jdoe@wkdtest.ntvtn.de > 4pkteh5be3b3shuzepabpupwbi95cirz`** (Zeilenumbruch im Hash-Wert aus Gründen der Formatierung eingefügt).

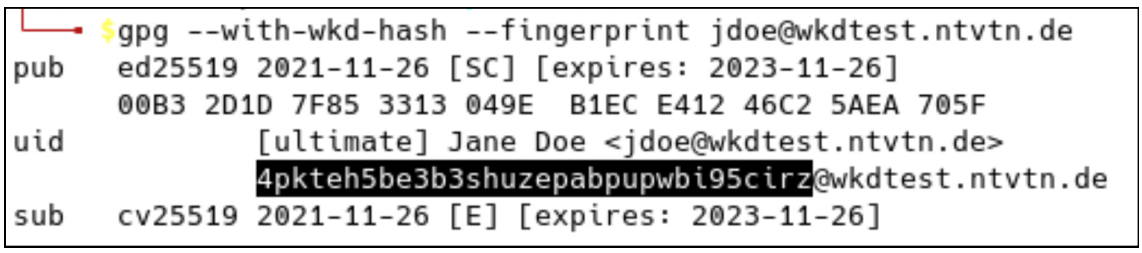


Abbildung 12: Herausfinden des Hash-Werts

Nachdem der Befehl gpg –-with-wkd-hash –-fingerprint ausgeführt wurde, wird eine Mail-Adresse angezeigt, deren lokaler Teil aus dem gesuchten Hash-Wert besteht.

Auf diese Weise werden die Schlüssel aller Test-Mail-Adressen erstellt. Danach erfolgt das Speichern auf dem WKD-Server. Die folgenden Absätze erläutern, welche Verzeichnisse angelegt werden und welche Domains auf diese Verzeichnisse verweisen. Tabelle 2 fasst zusammen, welche Verzeichnisse zu welcher Domain gehören und wie sie erreichbar sind. Die Domain *openpgpkey.wkdtest.ntvtn.de* zeigt auf das Verzeichnis /var/www/html/wkdtest-openpgpkey/. Aufgrund der Subdomain *openpgpkey* soll nur die Advanced Method verwendet werden, wenn Schlüssel bezogen werden. Da die Advanced Method zum Einsatz kommt, wird das Unterverzeichnis .well-known/openpgpkey/wkdtest.ntvtn.de/hu/ für die Schlüssel erstellt, die mit dieser Methode erreichbar sein sollen.

Die Überprüfung soll auch erkennen lassen, ob Produkte die Direct Method verwenden, wenn kein Schlüssel mit der Advanced Method geholt werden kann, obwohl dies wegen der Subdomain nicht geschehen darf. Dazu werden auch Schlüssel im Verzeichnis /var/www/html/wkdtest/.well-known/openpgpkey/hu/ hinterlegt, wobei die Domain *wkdtest.ntvtn.de* auf den Ordner /var/www/html/wkdtest/ zeigt.

Schlüssel mit der Domain *direct.wkdtest.ntvtn.de* sollen mit der Direct Method bezogen werden. Diese verweist auf das Verzeichnis /var/www/html/wkdtest-direct/. Daneben gibt es noch die Domain *direct2.wkdtest.ntvtn.de*, deren Inhalte sich im Verzeichnis /var/www/html/wkdtest-direct2/ befinden. Für diese Domain existieren außerdem ein Wildcard-Record **.direct2.wkdtest.ntvtn.de* und ein TXT-Record. Durch das Wildcard-Record würde das Aufrufen von *openpgpkey.direct2.wkdtest.ntvtn.de* zu keinem Fehler führen. Das TXT-Record soll jedoch dafür sorgen, dass diese Domain ignoriert und die Direct Method verwendet wird. Beide Domains enthalten jeweils das Unterverzeichnis /.well-known/openpgpkey/hu/. Die Domain, die mit „direct2“ beginnt, umfasst zusätzlich das Verzeichnis /.well-known/openpgpkey/direct2.wkdtest.ntvtn.de/hu/. Die Schlüssel, die dorthin verschoben werden, beweisen, ob fälschlicherweise die Advanced Method verwendet wird.

Aus den Domains der Test-Mail-Adressen ergibt sich die Platzierung der dazugehörigen Schlüssel. Die Adressen werden im Folgenden durch Zahlen ersetzt, die die Adressen durch die Nummerierung in Tabelle 1 erhalten. Die Schlüssel der Adressen 2 und 4 befinden sich im Verzeichnis /var/www/html/wkdtest-openpgpkey/.well-known/openpgpkey/wkdtest.ntvtn.de/hu/, sodass sie per Advanced Method erhalten werden können und die Schlüssel von 1 und 3 im Verzeichnis /var/www/html/wkdtest/.well-known/openpgpkey/hu/, sodass sie nur per Direct Method zu bekommen wären. Aufgrund der vorhandenen Subdomain *openpgpkey* dürfen sie jedoch nicht geholt werden.

Die Schlüssel der Mail-Adressen 5 bis einschließlich 8 halten sich in /var/www/html/wkdtest-direct/.well-known/openpgpkey/hu/ auf, sodass sie nur durch die Direct Method bezogen werden können. Die Schlüssel der Adressen 9 und 11 werden im Verzeichnis /var/www/html/wkdtest-direct2/

.well-known/openpgpkey/hu/ und die der Adressen 10 und 12 im Pfad /var/www/html/wkdtest-direct2/.well-known/openpgpkey/direct2.wkdtest.ntvtn.de/hu/ verstaut, sodass nur die Schlüssel der Adressen 9 und 11 zu holen sein sollten.

Die Vorbereitung für die Tests betrifft neben den Servern auch die zu testenden Produkte. Die Umgebung der Produkte Mailvelope und Claws Mail besteht aus einer virtuellen Maschine mit dem Betriebssystem Linux Mint 20.2. Der Test von KMail findet ebenfalls in einer virtuellen Maschine statt, jedoch mit dem Betriebssystem KDE neon, da dort darauf geachtet wird, dass die Software von KDE – und damit auch KMail – möglichst aktuell gehalten wird. Die Vorbereitung von Thunderbird besteht unter anderem aus dem Bauen des Produkts. Die Empfehlung für das Bauen von Thunderbird ist ein Computer mit mindestens 8 GB RAM [25]. Aus diesem Grund wird für den Test von Thunderbird auf eine virtuelle Maschine verzichtet und der Test läuft auf einem Computer mit dem Betriebssystem Parrot OS. Die Installation der Android-Apps folgt ebenfalls nicht in einer virtuellen Maschine, sondern auf einem Smartphone mit dem Betriebssystem /e/ OS, welches auf Android basiert. Die Produkte werden entweder installiert oder gebaut, um eine aktuelle Version zu erhalten und zu testen. Dabei ist die App F-Droid der Ursprung für die Android-Apps, während Mailvelope von der offiziellen Seite für Firefox-Erweiterungen bezogen wird.

Nr.	Mail-Adresse	Methode
1	jdoe@wkdtest.ntvtn.de	-
2	kim_luca.ross-mueller@wkdtest.ntvtn.de	Advanced
3	harry+sally@wkdtest.ntvtn.de	-
4	wkdtest@wkdtest.ntvtn.de	Advanced
5	jdoe@direct.wkdtest.ntvtn.de	Direct
6	kim_luca.ross-mueller@direct.wkdtest.ntvtn.de	Direct
7	harry+sally@direct.wkdtest.ntvtn.de	Direct
8	wkdtest@direct.wkdtest.ntvtn.de	Direct
9	jdoe@direct2.wkdtest.ntvtn.de	Direct
10	kim_luca.ross-mueller@direct2.wkdtest.ntvtn.de	-
11	harry+sally@direct2.wkdtest.ntvtn.de	Direct
12	wkdtest@direct2.wkdtest.ntvtn.de	-

Tabelle 1: Test-Mail-Adressen

Die Test-Mail-Adressen, deren Schlüssel auf einem WKD-Server gespeichert werden. Die Methode gibt an, welche Methode aus dem WKD-Standard verwendet werden muss, um den Schlüssel zu erlangen. Ein „-“ bedeutet, dass dieser Schlüssel nicht bezogen werden darf, wenn das getestete Produkt kompatibel zum aktuellen Entwurf des WKD-Standards ist.

Domain	Verzeichnis der Schlüssel	Methode	Kommentar
wkdtest.ntvtn.de	/var/www/html/wkdtest/.well-known/openpgpkey/hu/	Direct	Sollten nicht bezogen werden, da die Subdomain openpgpkey existiert.
openpgpkey.wkdtest.ntvtn.de	/var/www/html/wkdtest-openpgpkey/.well-known/openpgpkey/wkdtest.ntvtn.de/hu/	Advanced	
direct.wkdtest.ntvtn.de	/var/www/html/wkdtest-direct/.well-known/openpgpkey/hu/	Direct	
direct2.wkdtest.ntvtn.de	/var/www/html/wkdtest-direct2/.well-known/openpgpkey/hu/	Direct	
openpgpkey.direct2.wkdtest.ntvtn.de	/var/www/html/wkdtest-direct2/.well-known/openpgpkey/direct2.wkdtest.ntvtn.de/hu/	Advanced	Ein TXT-Record soll dafür sorgen, dass die Advanced Method nicht durchgeführt wird.

Tabelle 2: Verzeichnisse für die Schlüssel der Test-Mail-Adressen

Diese Tabelle zeigt, welche Domains aufgerufen werden müssen, um Schlüssel in bestimmten Verzeichnissen zu erreichen. Außerdem gibt sie an, welche Methode dafür ausgeführt werden muss.

Auswahl von Produkten

Da es unmöglich ist, Maßnahmen bei allen verfügbaren Produkten im zeitlichen Rahmen der Arbeit anzuwenden, grenzt eine Auswahl die Anzahl der Produkte ein, für die Maßnahmen ergriffen werden. Die ausgewählten Produkte sind ausschließlich Freie Software. Freie Software ist hierbei ein feststehender Begriff, der Software beschreibt, die bestimmte Bedingungen erfüllt [26]. Unter anderem muss der Quellcode der Software für jede Person einsehbar sein. Dazu kommt, dass es möglich ist, dass verschiedene Entwickler:innen zum Quellcode beitragen können, um die Software zu verbessern. Diese beiden Aspekte ermöglichen erst, dass überhaupt Maßnahmen an diesen Produkten ergriffen werden können und erhöhen die Wahrscheinlichkeit, dass Veränderungen auch in die Produkte integriert werden.

Ein weiterer Punkt, der beim Aussuchen der Produkte eine Rolle spielt, ist, dass diese Produkte die Verschlüsselung nach OpenPGP unterstützen. Logischerweise ist es nur in diesem Fall sinnvoll, den WKD-Standard zu nutzen, um an OpenPGP-Schlüssel zu gelangen. Ein Ausschlusskriterium für potentielle Produkte ist, wenn die Produkte nicht mehr aktiv entwickelt werden. Das bedeutet in dieser Arbeit, dass es in den vergangenen zwölf Monaten keine neuen Versionen gab.

Bevor die Produkte selektiert werden, bestimmt ein kurzes Verfahren, wie relevant sie sind. Dies zu bewerten ist schwierig, da es oft keine Zahlen dazu gibt, wie oft ein Produkt heruntergeladen wurde oder wie viele Anwender:innen ein Produkt verwenden. Deshalb entscheidet die Anzahl der Suchergebnisse in einer Suchmaschine die Relevanz, wenn nach dem jeweiligen Produkt gesucht wird. Konkret beinhaltet die Suche den Produktnamen und den Zusatz „ mail“ (mit Leerzeichen), z.B. „Alpine mail“. Die verwendete Suchmaschine ist dabei die Meta-Suchmaschine Metager (https://metager.org/) und die bei der Suche berücksichtigten Produkte stammen aus einer umfangreichen Auflistung auf der Seite Wikipedia.de [27]. Ein Nachteil dieser Methode ist allerdings, dass die Produktnamen mehrdeutig sein können. Ein Beispiel dafür ist der Mail-Client Alpine. Der Name Alpine ist ebenfalls Teil des Namens der Linux-Distribution Linux Alpine.

Die Tabelle 3 ist das Resultat dieses Verfahrens und hält die Anzahl der Suchergebnisse für verschiedene Produkte fest, die außerdem die vorhin genannten Voraussetzungen erfüllen. Dabei wird sichtbar, dass Claws Mail mit 8.640.000 Suchergebnissen die größte Relevanz besitzt. Aus diesem Grund wird diese Anwendung in die Auswahl aufgenommen.

Auch Android-Apps kommen für die Auswahl in Frage. Unter Android werden Applikationen ausgewählt, die die meisten Anwender:innen haben. In diesem Fall ist es möglich, diese Zahl im Play Store abzulesen. Bei den Mail-Clients, die OpenPGP unterstützen, sind die Apps mit den meisten Nutzer:innen FairEmail mit 19.523 und K9Mail mit 95.357 Nutzer:innen

(Stand: 14.12.2021). Außerdem wird unter den Erweiterungen für den Browser Firefox nach geeigneten Kandidaten gesucht. Dort hat Mailvelope 43.211 Anwender:innen (Stand: 14.12.2021). Sowohl bei den Android-Applikationen als auch bei den Browser-Erweiterungen sind keine weiteren Produkte zu finden, die eine ähnlich hohe Anzahl an Anwender:innen aufweisen können.

In einem Projekt, das in [7] beschrieben wird, wurde der WKD-Standard entwickelt und es war geplant, dass die aus dem Projekt hervorgehenden Verbesserungen in Thunderbird und KMail übernommen werden sollten. Somit ist es sinnvoll, diese beiden Produkte ebenfalls einem Test zu unterwerfen. Zusammengefasst bilden Claws Mail, FairEmail, K9Mail, KMail, Mailvelope und Thunderbird die Liste an ausgewählten Produkten für die Tests.

Produkt	**Suchergebnisse bei metager.org**
Balsa	48.100
KMail	123.000
Geary	171.000
Alpine	179.000
Evolution	196.000
Mutt	1.680.000
Thunderbird	4.170.000
Claws Mail	8.640.000

Tabelle 3: Produkte und Anzahl der Suchergebnisse
Datum der Suche: 31.12.2021

4. Testen der Produkte

Dieses Kapitel beschreibt die getesteten Produkte und inwiefern sie die Kriterien erfüllen, die im dritten Kapitel genannt wurden. Die Kürzel der Kriterien, wie z.B. K6, kommen hier zum Einsatz, damit Leser:innen schneller herausfinden können, warum Produkte ein bestimmtes Kriterium erfüllen oder nicht, wenn sie in die Liste der Ergebnisse am Ende dieses Kapitels schauen.

Mailvelope

Mailvelope ist eine Erweiterung für die Browser Firefox und Chrome. Sie ermöglicht die Verwaltung von Schlüsseln und das Verschlüsseln von Mails. Dabei können Benutzer:innen sie direkt in den Internetportalen von unterschiedlichen Unternehmen verwenden, z.B. Mailbox oder Posteo. Als die Erweiterung untersucht wird, wird sie in der Version 4.4.1 angeboten. Zu diesem Zeitpunkt ist die WKD-Funktion schon integriert, allerdings beweist ein Test, dass Mailvelope nicht zum aktuellen Entwurf für WKD kompatibel ist, da nur Schlüssel heruntergeladen werden, die per Direct Method erreichbar sind (K1). Nach einer frischen Installation ist bereits eingestellt, dass WKD-Server als Quelle für die Suche berücksichtigt werden (K2). Wie unten beschrieben wird, wird die WKD-Funktion jedoch nicht an allen Stellen verwendet, wo die Möglichkeit gegeben wäre.

Im Produkt gibt es mehrere Stellen, an denen das Holen von Schlüsseln per WKD durchgeführt werden kann und teilweise wird. Nach dem Öffnen der Erweiterung können Anwender:innen den Reiter “Encrypt” in der oberen Leiste anklicken. Dadurch öffnet sich ein Bereich, in dem eine Datei hochgeladen und eine Nachricht eingegeben werden kann, sodass diese verschlüsselt werden können (Abbildung 13). Über dem Eingabefeld für die Nachricht befindet sich ein weiteres Feld, in dem die Mail-Adressen der Empfänger:innen eingegeben werden können. Nach Eingabe einer Adresse sucht Mailvelope automatisch nach dem dazugehörigen Schlüssel. Hier werden auch WKDs in die Suche einbezogen. Ist das Beziehen des Schlüssels erfolgreich, erscheint die Adresse mit einem grünen Hintergrund und einem Icon, das ein Häkchen enthält. Bei einem Misserfolg ist der Hintergrund rot und es wird ein Ausrufezeichen anstelle des Häkchens angezeigt. Mailvelope bietet für vorhandene Schlüssel jedoch keine Informationen zu deren Vertrauenswürdigkeit.

Verschiedene Unternehmen unterstützen den Einsatz von Mailvelope, sodass in deren Internetportal Elemente von Mailvelope angezeigt werden. Abbildung 14 zeigt einen Button von Mailvelope im Interface von mail.de. Wenn Benutzer:innen auf diesen Button klicken, öffnet sich ein neues Fenster, in dem eine Nachricht eingegeben werden kann (Abbildung 15).

Genau wie im Bereich „Encrypt“ geben Anwender:innen auch hier die Adressen der Empfänger:innen der Nachricht ein. Nach der Betätigung des Buttons “Encrypt” in diesem Fenster, verschlüsselt Mailvelope die Nachricht und leitet sie an das eigentliche Fenster, in dem die Seite von mail.de geöffnet ist, weiter, sodass die Nachricht dort abgeschickt werden kann. Das Ergebnis ist in Abbildung 16 zu sehen. Beim Eingeben einer Mail-Adresse wird der dazugehörige Schlüssel automatisch auch in Web Key Directories gesucht (K3, K5). Nachdem die Schlüssel heruntergeladen wurden, sind jedoch keine Informationen zu deren Vertrauenswürdigkeit sichtbar (K8, K10, K12).

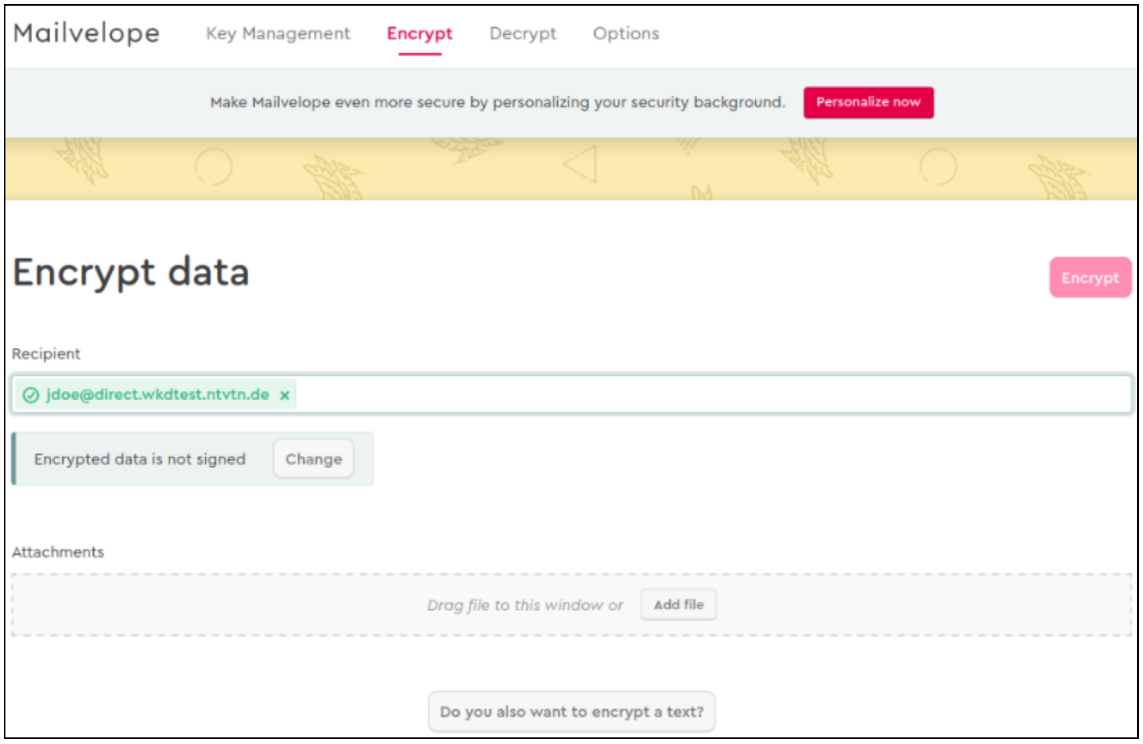


Abbildung 13: Mailvelope – Verschlüsseln von Dateien und/ oder Text

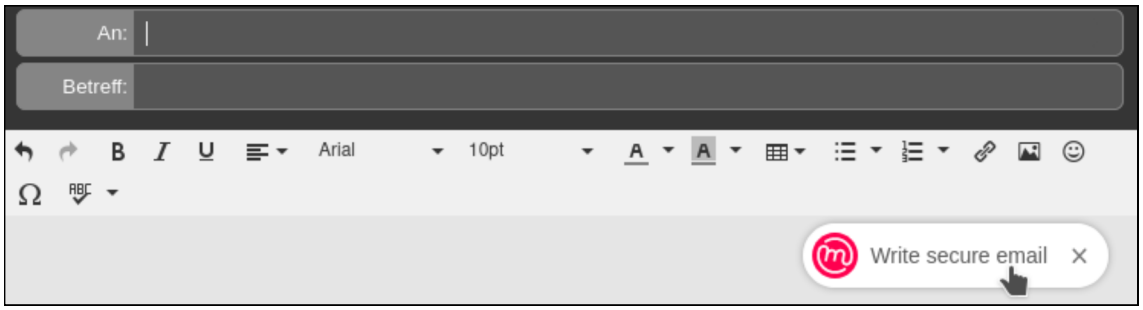


Abbildung 14: Mailvelope-Button im Interface von mail.de

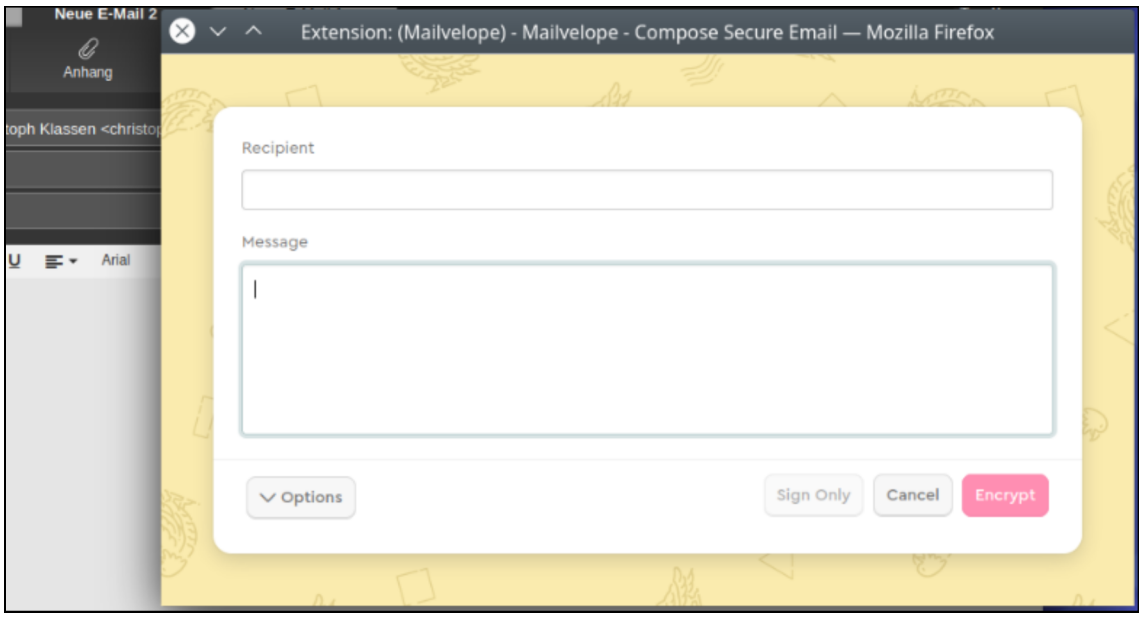


Abbildung 15: Mailvelope – Fenster zum Verfassen einer verschlüsselten Mail

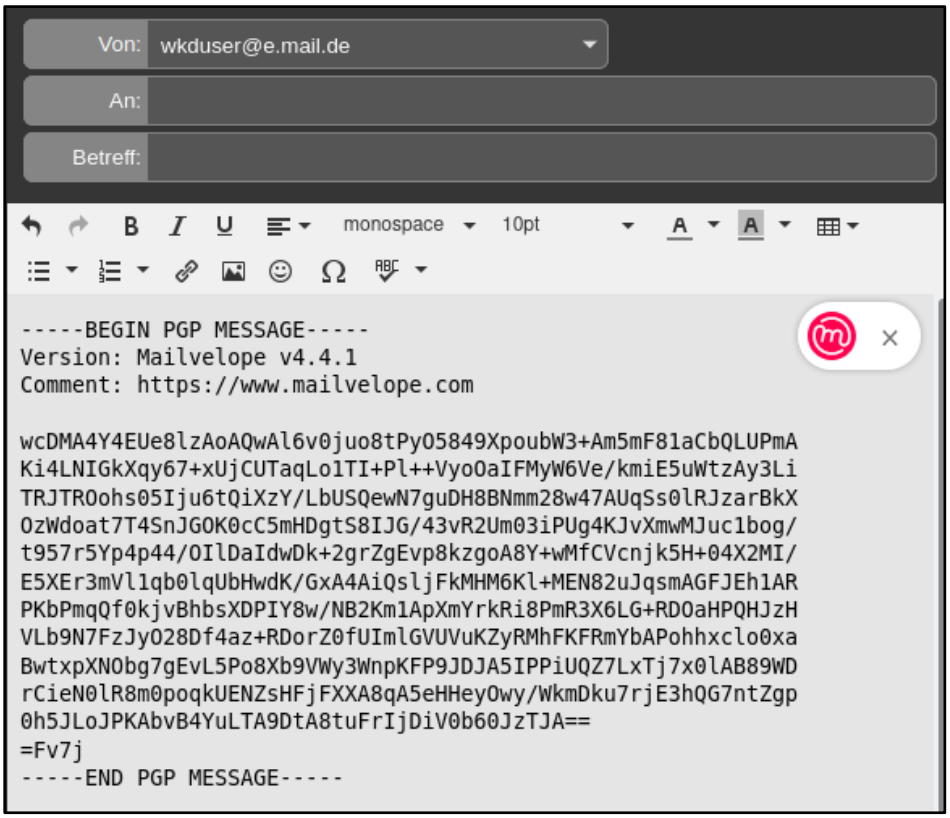


Abbildung 16: Mithilfe von Mailvelope verschlüsselte Nachricht in mail.de

Bei einem Klick auf “Key Management” in der Menüleiste (Abbildung 13 oben) kann der gleichnamige Bereich geöffnet werden. Dort gibt es einen Button entweder mit der Beschriftung “Import Key”, wenn sich noch kein einziger Schlüssel im Schlüsselbund der Anwendung befindet oder “Import”, wenn mindestens ein Schlüssel vorhanden ist. Wird dieser Button betätigt, öffnet sich ein Bereich mit der Überschrift “Add Keys” mit zwei Reitern. Wenn Benutzer:innen auf den Reiter “Import” klicken, öffnet sich ein Bereich, in

dem sie Schlüssel hochladen können, um sie dann mit der Browser-Erweiterung nutzen zu können. Entscheiden sich Nutzer:innen für den zweiten Reiter mit dem Namen “Search”, erreichen sie einen Bereich mit einer Suchfunktion (Abbildung 17). Diese dient dazu, Schlüssel aus externen Quellen zu beziehen. An dieser Stelle werden WKD-Server jedoch nicht miteinbezogen, wie ein kurzer Versuch zeigt.

Abbildung 17: Mailvelope – Suchfunktion für öffentliche Schlüssel

Die Seite mit häufig gestellten Fragen zu Mailvelope erklärt, dass die Erweiterung eine Signatur automatisch überprüft [28]. Um diese Aussage zu testen, werden Mails von verschiedenen Mail-Clients an eine Mail-Adresse eines Accounts bei mailbox.org geschickt. Vorher wurden diese Mails signiert. In diesem Account werden die signierten Mails geöffnet. Dort zeigt Mailvelope jedoch nicht an, dass es eine Signatur gibt (Abbildung 18). Dass eine Signatur verwendet wurde, kann jedoch bestätigt werden, weil sich diese im Anhang befindet. Da keine Signatur in der Anzeige von Mailvelope zu sehen ist, kann keine Aussage dazu gemacht werden, ob dabei WKD verwendet und ob Informationen zur Validität des Schlüssels gezeigt werden (K9, K11, K13). Gleichzeitig kann nicht überprüft werden, ob die WKD-Funktion bei der Signaturüberprüfung eingesetzt wird, um fehlende Schlüssel zu erhalten (K4) und, ob ein per WKD erhaltener Schlüssel gegenüber Schlüsseln bevorzugt werden würde, die keine Informationen zu ihrer Vertrauenswürdigkeit enthalten (K7).

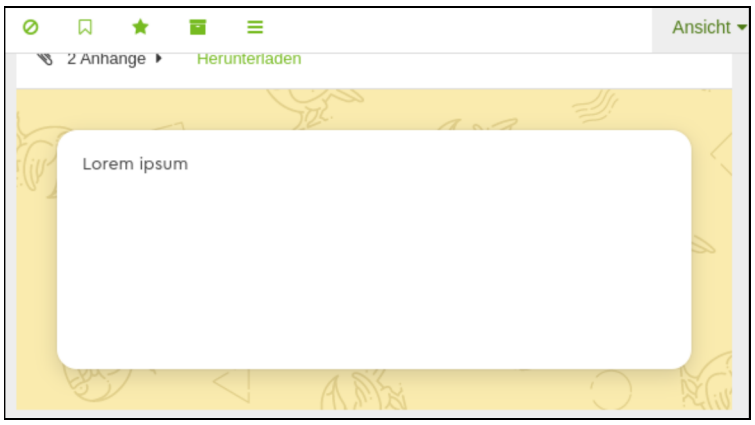


Abbildung 18: Mailvelope – Signatur wird nicht angezeigt

Obwohl die Mail signiert wurde, gibt es von Mailvelope keinen Hinweis darauf, dass es überhaupt eine Signatur gibt.

Zuletzt zeigt ein Experiment, ob Mailvelope automatisch den Schlüssel für die Verschlüsselung auswählt, der die größere Vertrauenswürdigkeit aufweist. Dazu wird zuerst ein Schlüssel per WKD bezogen, bevor zwei weitere Schlüssel direkt importiert werden. Dann wird eine Mail von einem Account bei mailbox.org an die Mail-Adresse, zu der die drei Schlüssel gehören, mithilfe von Mailvelope verschlüsselt. Nach der Eingabe der Adresse zeigt die Erweiterung keine Informationen über den verwendeten Schlüssel an. Deswegen wird mithilfe eines anderen Mail-Clients überprüft, welcher der drei Schlüssel für die Verschlüsselung verwendet wurde. Das Ergebnis ist, dass Mailvelope den zuletzt erstellten Schlüssel eingesetzt hat. Somit bevorzugt die Erweiterung einen Schlüssel, der per WKD erhalten wurde, nicht gegenüber anderen Schlüsseln, die direkt importiert wurden und keine weiteren Informationen über ihre Vertrauenswürdigkeit besitzen (K5). Ein Test, bei dem zuerst Schlüssel importiert und dann ein Schlüssel per WKD bezogen wird, ist nicht möglich, da Mailvelope keine neuen Schlüssel sucht, wenn schon ein Schlüssel für eine Mail-Adresse vorhanden ist.

Claws Mail

Die Entwickler:innen von Claws Mail behaupten, dass ihr Produkt das Abrufen per WKD-Funktion seit der Version 3.18.0 bzw. 4.0.0 unterstützt [29]. Zum Zeitpunkt der Analyse kann Claws Mail nur als Version 3.17.5 per Paketmanager des Betriebssystem Linux Mint installiert werden. Deshalb wird der Quellcode der Version 3.18.0 von der Webseite der Entwickler:innen heruntergeladen und das Produkt gebaut.

Damit Mails per OpenPGP verschlüsselt werden können, ist es zunächst notwendig, zwei Plug-ins in die Anwendung zu integrieren: „PGP/Core“ und „PGP/MIME“. Nach dem Herunterladen des Quellcodes sind diese und weitere Erweiterungen bereits als Dateien enthalten. Diese Dateien müssen Benutzer:innen jedoch manuell einmalig in der Anwendung auswählen, damit Claws Mail sie lädt und verwendet. Dafür öffnen sie in der Menüleiste das Menü „Configuration“ und wählen den Eintrag „Plugins…“. Daraufhin erscheint das Fenster, welches in Abbildung 19 dargestellt wird. Dort klicken sie auf den Button „Load…“. Ein weiteres Fenster mit einem Dateimanager öffnet sich, in dem die Anwender:innen die Dateien der genannten Plug-ins selektieren und bestätigen müssen.

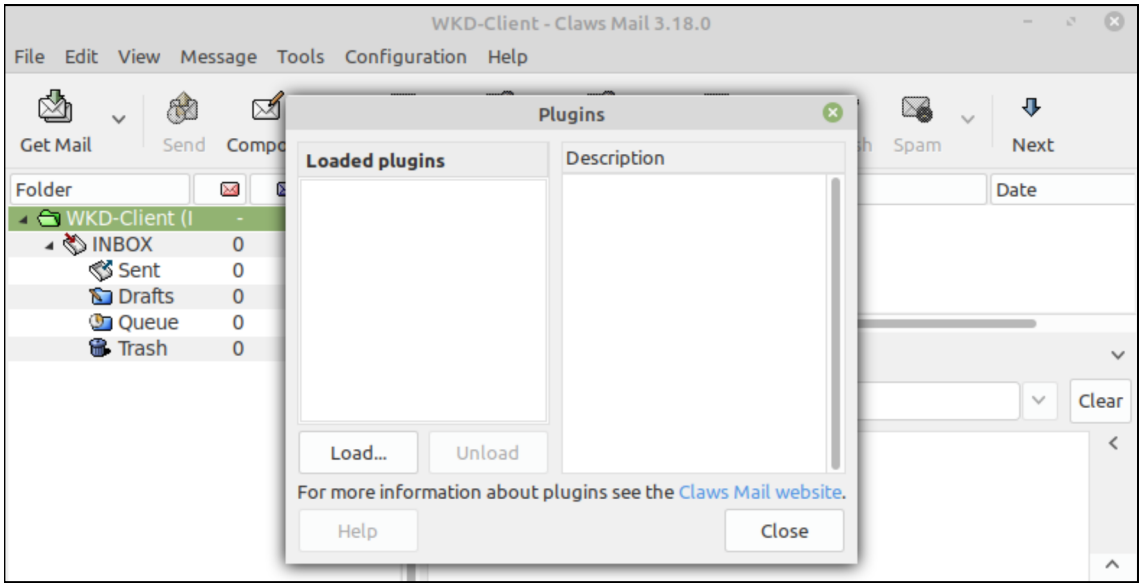


Abbildung 19: Claws Mail – Plug-ins

Nach dem Bauen der Anwendung werden standardmäßig keine Plug-ins geladen

Anschließend ist die Auswahl eines sogenannten “Privacy Systems” notwendig. Damit legen Benutzer:innen die Methode fest, mit der Mails verschlüsselt werden, z.B. PGP/MIME. Dafür gibt es zwei Wege. Über den ersten Weg stellen Anwender:innen das Privacy System permanent ein. Dieser Weg führt über das Menü “Configuration” in der oberen Leiste und den dort vorhandenen Eintrag “Edit accounts…”. Es öffnet sich ein Fenster, in dem der passende Account ausgewählt und der Button “Edit” angeklickt wird (Abbildung 20). Ein weiteres Fenster mit den Einstellungen für diesen Account öffnet sich (Abbildung 21). Hier befindet sich auf der linken Seite ein

Bereich mit verschiedenen Kategorien. Unter der Kategorie “Account” befindet sich der Punkt “Privacy”. Wählen Nutzer:innen diesen aus, erscheinen Einstellungen zu ebendiesem Bereich. Unter anderem können sie hier einstellen, welches Privacy System verwendet werden soll.

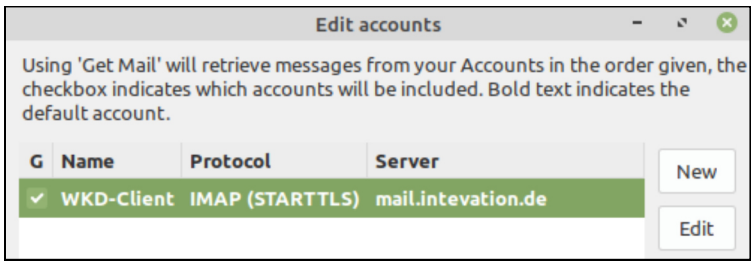


Abbildung 20: Claws Mail – Einstellungen für Accounts

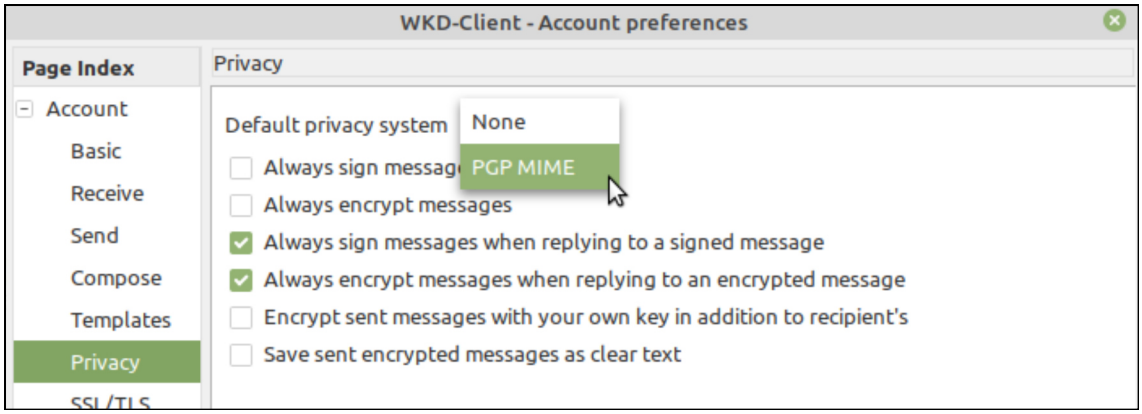


Abbildung 21: Claws Mail – Auswahl des Privacy Systems

Wenn das Privacy System in den Accounteinstellungen geändert wird, bleibt die Veränderung permanent erhalten.

Die zweite Möglichkeit befindet sich direkt im Fenster zum Verfassen einer E-Mail (Abbildung 22). In der oberen Leiste dieses Fensters befindet sich das Menü “Options” und dort der Eintrag “Privacy System”, wo das System ausgewählt werden kann. Stellen Benutzer:innen das Privacy System auf diesem Weg ein, ist die Veränderung temporär und gilt nur für die zu verfassende Mail. Nachdem die Auswahl des Systems getroffen wurde, kann ebenfalls im Menü “Options” die Option “Encrypt” aktiviert werden. Vorher haben Nutzer:innen keine Möglichkeit, eine verschlüsselte Mail zu versenden. Durch die vielen Schritte bei der Vorbereitung kann Claws Mail das Kriterium K2 nicht erfüllen.

In einer geöffneten Mail befindet sich auf der rechten Seite eine Leiste mit einigen Buttons (Abbildung 23). Einer davon ist mit einem Schlüssel als Icon versehen. Gibt es noch keinen Schlüssel für die Mail-Adresse der geöffneten Mail, taucht beim Anklicken dieses Buttons ein kleines Fenster auf, in dem zwei Möglichkeiten bestehen, den Schlüssel zu erhalten (Abbildung 24). Eine davon beinhaltet das Beziehen des Schlüssels per WKD. Das Beziehen von Schlüsseln von einem WKD-Server funktioniert in Claws Mail wie vom Standard vorgegeben (K1).

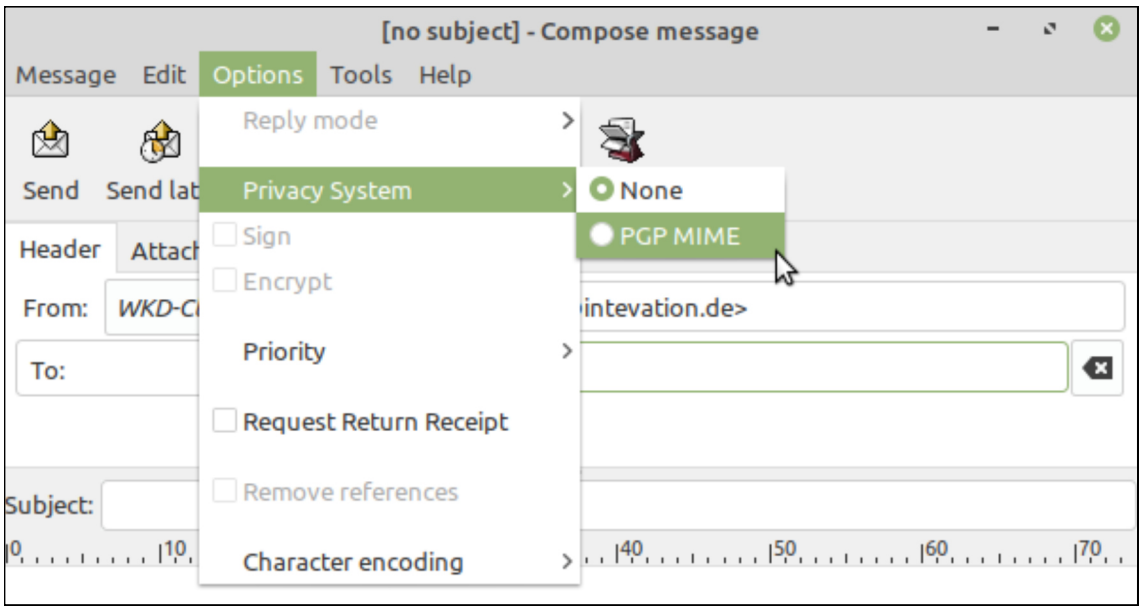


Abbildung 22: Claws Mail – Auswahl des Privacy Systems

Beim Verfassen einer Mail kann das Privacy System temporär für diese eine Mail angepasst werden.

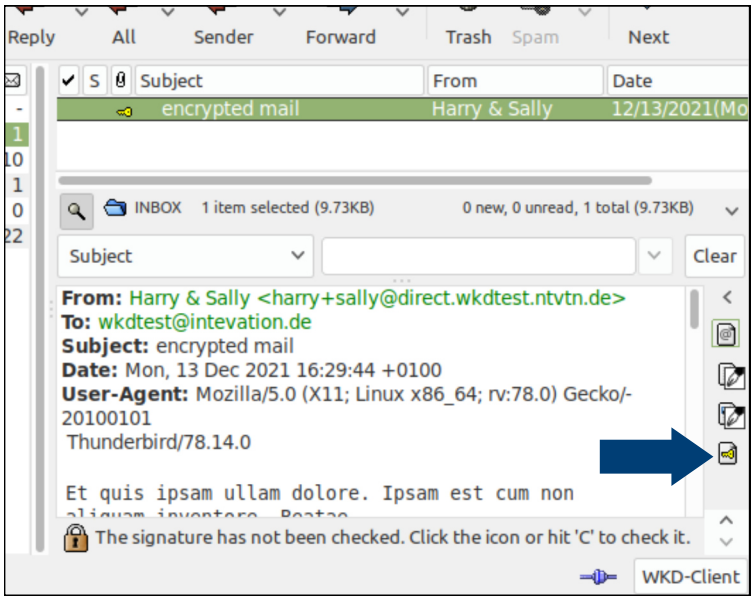


Abbildung 23: Claws Mail – Suchfunktion für öffentliche Schlüssel

Die Funktion zum Suchen eines Schlüssels ist erst bei genauerer Betrachtung zu sehen.

Abbildung 24: Claws Mail – Auswahl der Quelle für die Schlüsselsuche

Nachdem eine Mail geöffnet wird, zeigt Claws Mail am unteren Rand an, dass die Taste „C“ auf der Tastatur gedrückt werden muss, um die Signatur zu überprüfen (Abbildung 25). Folgen Nutzer:innen dieser Aufforderung, ändern sich daraufhin das Icon und der Text am unteren Rand. Der Text enthält am Ende den Hinweis auf die Validität des Schlüssels, der für die Signaturüberprüfung verwendet wird (Abbildung 26). Dabei werden genau die Stufen der Validität angezeigt, die von GnuPG verwendet werden (K9,

K13). Allerdings zeigt die Anwendung für Schlüssel, die mithilfe der WKD-Funktion bezogen wurden, nicht an, dass diese ein Mindestmaß an Vertrauenswürdigkeit besitzen (K11). Da sich der Button zum Beziehen eines Schlüssels im selben Fenster befindet, in dem Benutzer:innen auch die Signaturüberprüfung durchführen können, gilt das Kriterium K4 als erfüllt. Bei der Signaturüberprüfung wird die WKD-Funktion nicht automatisch eingesetzt. Wenn ein Schlüssel fehlt, weist der Text lediglich darauf hin, dass der Schlüssel nicht vorhanden ist (Abbildung 27).

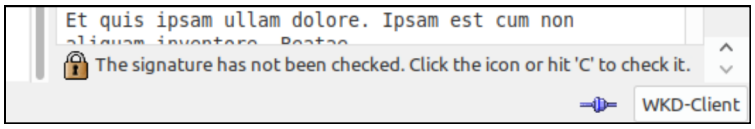


Abbildung 25: Claws Mail – Signaturüberprüfung

Abbildung 26: Claws Mail – Erfolgreiche Signaturüberprüfung

Ganz rechts im Hinweis zu der Signatur befindet sich die Validität des Schlüssels. Im dargestellten Fall ist der Schlüssel vollständig valide.

Abbildung 27: Claws Mail – Fehlgeschlagene Signaturüberprüfung

Während Benutzer:innen bei einer erhaltenen signierten Mail die WKD-Funktion aufrufen können, gibt es im Fenster zum Schreiben einer Mail (Abbildung 28) kein Bedienelement, das diese Funktion startet. Außerdem enthält die grafische Oberfläche keine Anzeige über vorhandene Schlüssel und damit auch keine Informationen über deren Vertrauenswürdigkeit (K8, K10, K12).

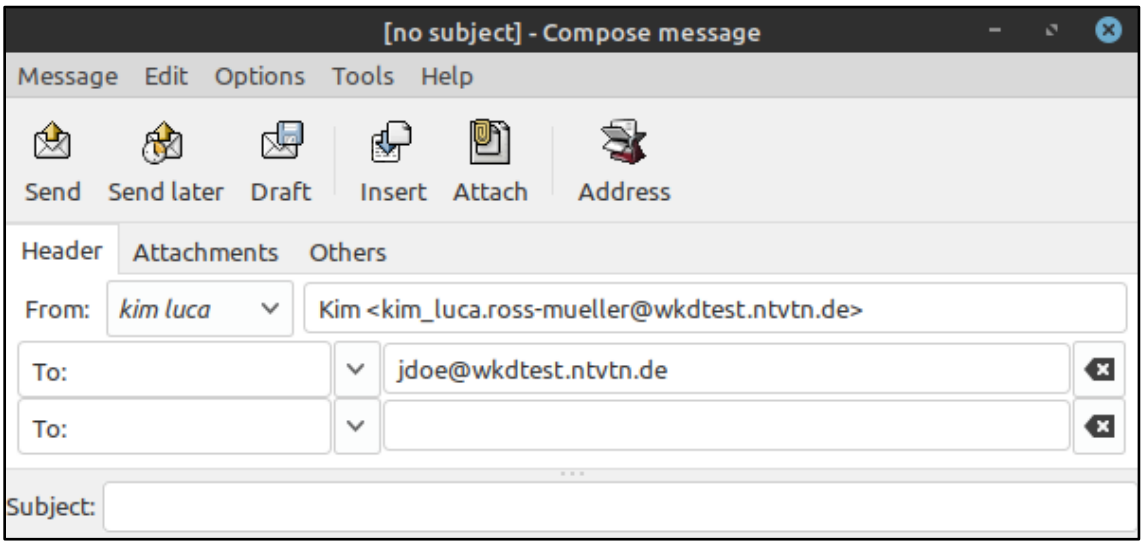


Abbildung 28: Claws Mail – Fenster zum Schreiben einer Mail

Sobald Anwender:innen versuchen, eine verschlüsselte Mail abzuschicken, gibt es drei Möglichkeiten. Die erste ist, dass das Absenden funktioniert. In diesem Fall schließt sich das Fenster, in dem die Mail geschrieben wurde. Die

zweite Möglichkeit trifft ein, wenn für eine Adresse kein Schlüssel vorhanden ist. Dabei erscheint ein Fenster, in dem Benutzer:innen gefragt werden, was sie als Nächstes tun möchten Dieses Fenster sieht ebenso aus wie in Abbildung 29, doch ist die Liste in der Mitte in diesem Fall leer. Die Nutzer:innen können sich dort dafür entscheiden, die Mail doch nicht zu verschlüsseln oder sie geben eine Schlüssel-ID eines Schlüssels einer anderen Mail-Adresse an, sodass die Mail für diese Adresse verschlüsselt wird. Die dritte Möglichkeit beinhaltet dasselbe Fenster. Diesmal gibt es jedoch mehrere vorhandene Schlüssel und Anwender:innen können in der Liste eine Auswahl treffen (Abbildung 29).

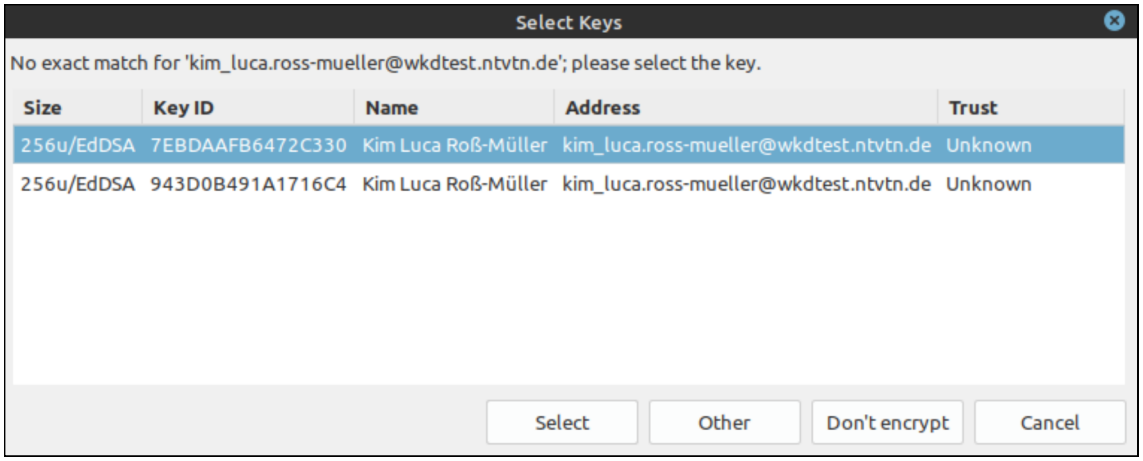


Abbildung 29: Claws Mail – Schlüsselauswahl

Beim Versuch, eine verschlüsselte Mail zu versenden, obwohl kein passender Schlüssel existiert, kann herausgefunden werden, dass die WKD-Funktion an dieser Stelle nicht verwendet wird (K3, K5). Außerdem zeigen die eben genannten möglichen Ereignisse beim Absenden einer Mail, dass Claws Mail keinen Schlüssel automatisch wählt. Schlüssel, die per WKD bezogen wurden, erhalten keinen Vorrang gegenüber direkt importierten Schlüsseln, die keine Informationen über ihre Vertrauenswürdigkeit anbieten (K6). Auch bei der Signaturüberprüfung spielt es keine Rolle, ob ein Schlüssel direkt importiert oder von einem WKD-Server erhalten wurde (K7).

Thunderbird

Ein weiteres Produkt, welches auf mehreren Plattformen verfügbar ist, ist Mozilla Thunderbird. Der Mail-Client wird für den Test nach der Anleitung des Teams hinter Thunderbird gebaut [30]. Der gebaute Client besitzt die Versionsnummer 97.0a1. Sobald Anwender:innen nach der Einrichtung des Accounts eine verschlüsselte Mail auswählen, erscheint bereits ein Meldung, in der erklärt wird, dass der private Schlüssel zur Entschlüsselung der Nachricht fehlt (Abbildung 30). Um diesen Umstand zu ändern, führt der Weg in die Einstellungen für den eigenen Account. Hier befindet sich auf der linken Seite des Fensters eine Liste mit verschiedenen Kategorien. Eine davon ist die Kategorie “End-To-End Encryption”. Wird diese geöffnet, fällt sofort ein Schlüssel-Icon auf (Abbildung 31). Daneben befindet sich ein Text, der besagt, dass es noch keinen persönlichen Schlüssel für die eigene Mail-Adresse gibt. Direkt rechts daneben erlaubt ein Button das Hinzufügen eines Schlüssels. Betätigen Nutzer:innen diesen, können sie im nächsten Schritt auswählen, ob sie einen neuen Schlüssel erstellen oder einen vorhandenen importieren möchten. Ist das Hinzufügen eines Schlüssels abgeschlossen, landen Anwender:innen wieder im Fenster mit dem Schlüssel-Icon. Darunter befindet sich der aktuell ausgewählte Schlüssel. Es ist notwendig, diesen anzuklicken, um ihn auszuwählen, damit die Verschlüsselung aktiviert werden kann (Abbildung 32). Dieser letzte Schritt disqualifiziert Thunderbird für das Kriterium K2, da dies ein Schritt ist, den die Anwendung automatisch vornehmen könnte.

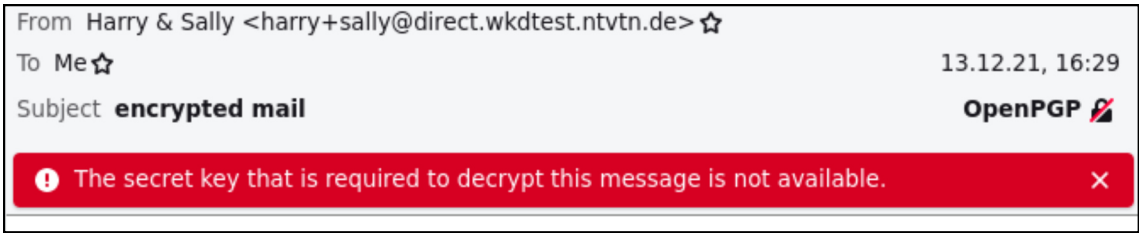


Abbildung 30: Thunderbird – Meldung bei fehlendem privaten Schlüssel

Abbildung 31: Thunderbird – Bereich „End-To-End Encyption“

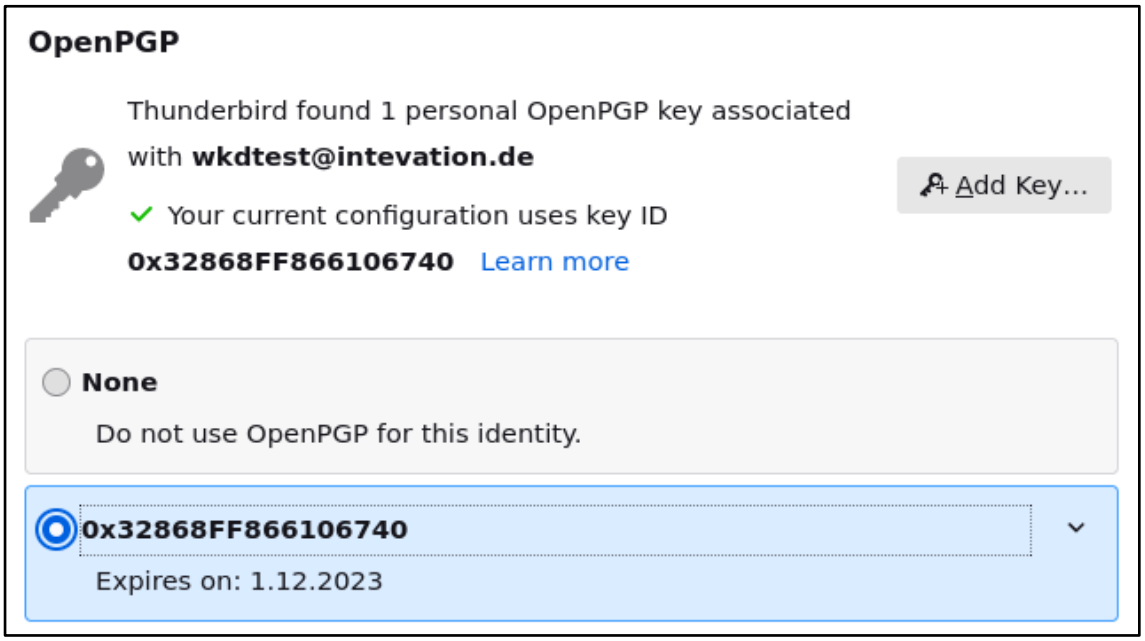


Abbildung 32: Thunderbird – Auswahl des persönlichen Schlüssel

Nach der Auswahl eines privaten Schlüssels als persönlichen Schlüssel ist es möglich, Mails zu verschlüsseln.

Ebenfalls im Bereich “End-To-End Encryption” ist der Button mit der Aufschrift “OpenPGP Key Manager” zu finden. Innerhalb des Fensters dieses Managers können Benutzer:innen über das Menü “Keyserver” den Eintrag “Discover Keys Online” finden (Abbildung 33). Bei einem Klick auf diesen Eintrag gelangen sie zu einem weiteren Fenster, in dem sie eine Mail-Adresse oder eine Schlüssel-ID eingegeben und mit einem Klick auf „OK“ die Suche nach öffentlichen Schlüsseln starten können. Der Text über dem Eingabefeld verrät, dass die Suche neben den Schlüsselservern auch WKD-Server einschließt (Abbildung 34).

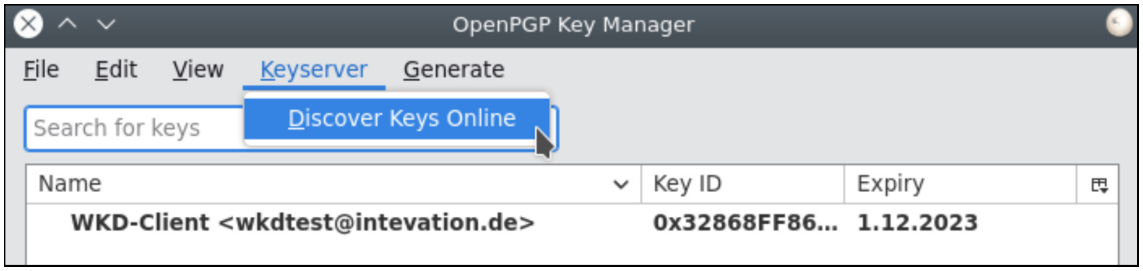


Abbildung 33: Thunderbird – OpenPGP Key Manager

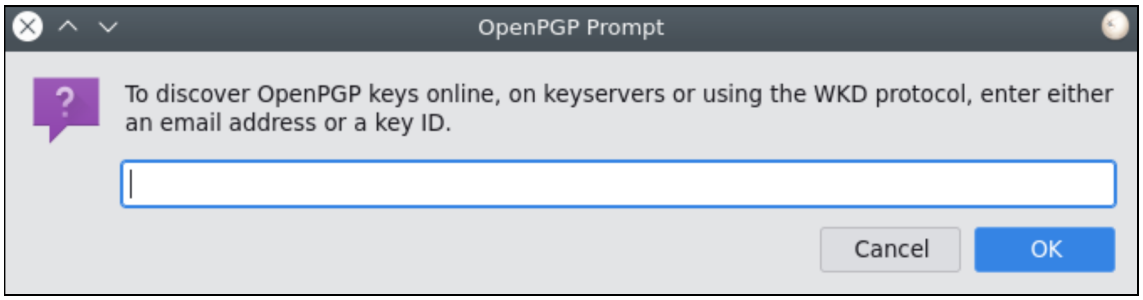


Abbildung 34: Thunderbird – Eingabefeld zum Suchen eines Schlüssels

Ein anderer Weg, um an die öffentlichen Schlüssel anderer Personen zu kommen, existiert im Fenster zum Schreiben einer Mail. In der zweiten Menüleiste ist dort ein Button mit der Aufschrift „Security“ zu sehen (Abbildung 35). Ein Klick darauf führt zu einem Fenster „OpenPGP Message Security“, in dem die vorher eingegebenen Mail-Adressen aufgelistet sind (Abbildung 36). Rechts neben den Adressen werden Anwender:innen darüber informiert, ob eine Verschlüsselung möglich ist oder der öffentliche Schlüssel fehlt. Im letzteren Fall können sie die betroffene Adresse und dann den Button direkt unter der Liste anklicken. Daraufhin macht Thunderbird ein weiteres Fenster sichtbar, in dem die Anwender:innen den Button „Discover new or updated key“ betätigen können, um die Suche nach dem ausgewählten Schlüssel zu starten (Abbildung 37).

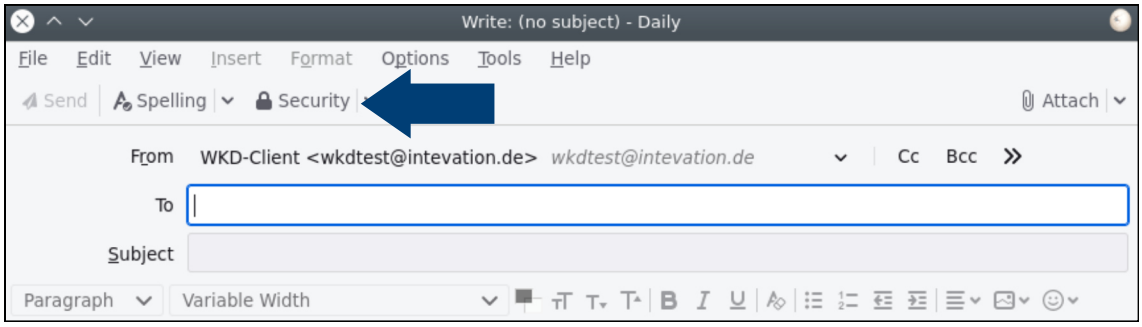


Abbildung 35: Thunderbird – „Security“-Button im Fenster zum Verfassen einer Mail

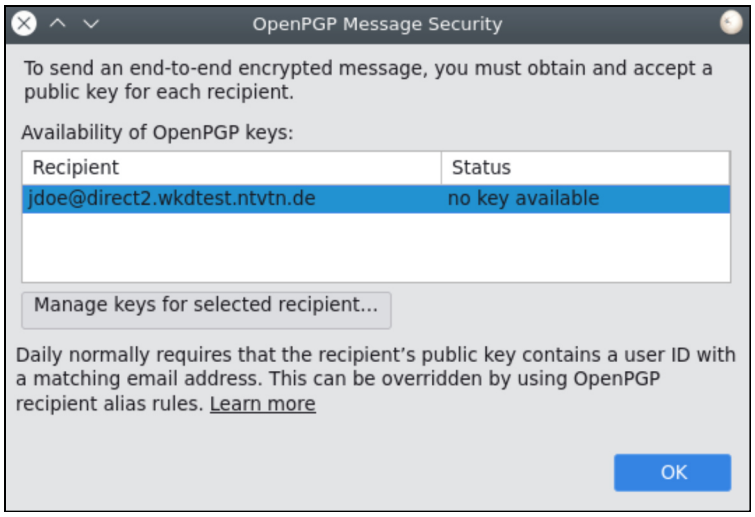


Abbildung 36: Thunderbird – Status von Schlüsseln

Im Status wird angegeben, ob ein Schlüssel für die angegebenen Mail-Adressen vorhanden ist.

Abbildung 37: Thunderbird – Starten der Schlüsselsuche

Nach einem Klick auf „Discover new or updated key“ wird nach einem öffentlichen Schlüssel gesucht.

Versuchen Personen, eine Mail zu verschicken, obwohl kein öffentlicher Schlüssel für die eingegebene Adresse vorhanden ist, erscheint eine Fehlermeldung, die auf dieses Problem hinweist. Nach dem Schließen dieser Meldung taucht wieder das Fenster „OpenPGP Message Security“ (Abbildung 36) auf, sodass auch dort die oben genannten Schritte durchgeführt werden können, um die Schlüssel zu erhalten. Somit wird die WKD-Funktion weder automatisch ausgeführt, noch befindet sich ein Bedienelement für diese Funktion direkt im Fenster zum Schreiben einer Mail (K3, K5). Nach einer erfolgreichen Suche nach den fehlenden Schlüsseln kann die Mail verschlüsselt und verschickt werden. Vor dem Abschicken gibt es direkt im Fenster zum Schreiben einer Mail keinen Hinweis darauf, ob Schlüssel vorhanden sind, geschweige denn wie vertrauenswürdig die Schlüssel sind (K8, K10, K12).

Beim Test mithilfe der Test-Mail-Adressen fällt auf, dass mehr Schlüssel heruntergeladen werden, als dies der Fall sein sollte. Die Schlüssel, die per WKD geholt werden können sollen, werden wie erwartet bezogen. Dass der Mail-Client die Schlüssel der Adressen 1 und 3 herunterlädt, macht jedoch deutlich, dass der WKD-Standard nicht eingehalten wird (K1).

Wenn eine erhaltene Mail geöffnet wird, befindet sich auf der rechten Seite ein Button mit der Aufschrift „OpenPGP“. Dieser enthält bereits Icons, die darauf hinweisen, ob eine Mail verschlüsselt oder signiert wurde. Bei einer Interaktion mit diesem Button öffnet sich ein Pop-up mit weiteren Informationen (Abbildung 38). Unter anderem steht dort eine Erläuterung zur Signatur. Über dieser Erläuterung gibt es einen Hinweis, wenn der Schlüssel fehlt, der zur Signaturüberprüfung notwendig ist. Der Button „Discover…“ auf der rechten Seite dient dazu, nach dem Schlüssel zu suchen. Ein Schlüssel, der sich auf einem WKD-Server befindet, wird dabei von Thunderbird allerdings nicht heruntergeladen. Somit kann ausgeschlossen werden, dass WKD an dieser Stelle verwendet wird (K4).

Abbildung 38: Thunderbird – Schlüsselsuche bei der Signatur

Die Signaturüberprüfung unter Thunderbird verrät nichts über die Validität eines Schlüssels, sondern über dessen Owner Trust. Dieses können Anwender:innen festlegen, wenn sie beim OpenPGP Key Manager einen Doppelklick auf einen Schlüssel ausführen. Es erscheint ein Fenster, in dem eine Auswahl für das Vertrauen getroffen werden kann (Abbildung 39). Dabei können Benutzer:innen entscheiden, dass sie dem Schlüssel gar nicht vertrauen, dass sie ihn akzeptieren, aber nicht überprüft haben, dass sie ihm vertrauen, nachdem sie den Fingerabdruck überprüft haben oder, dass sie diese Entscheidung noch nicht treffen möchten.

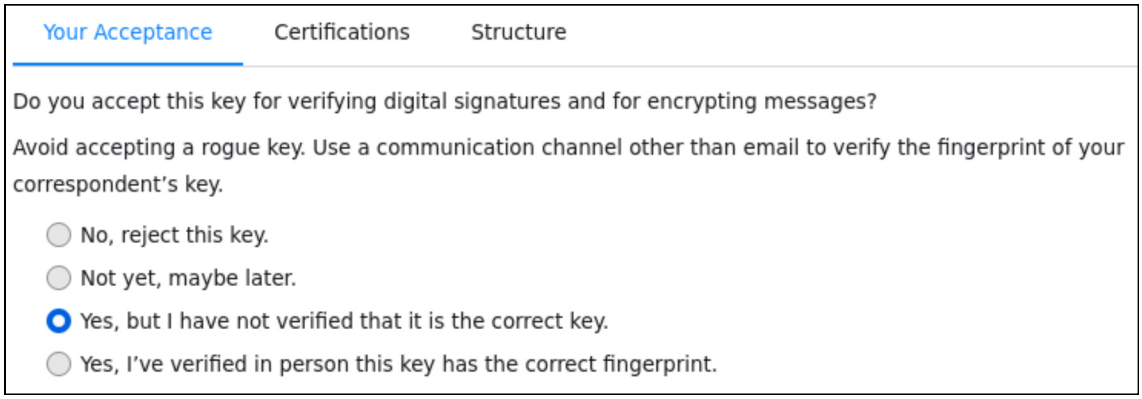


Abbildung 39: Thunderbird – Akzeptanz eines Schlüssels

Je nach Auswahl in diesem Dialog ändert sich das Symbol, welches der Mail-Client nach der Signaturüberprüfung anzeigt. Wenn Nutzer:innen keine Entscheidung treffen, bleibt das blaue Symbol sichtbar, welches auch dann angezeigt wird, wenn kein Schlüssel vorhanden ist (Abbildung 38, oben rechts). Bei einer Ablehnung wird ein rotes Symbol anzeigt. Ein gelbes Symbol tritt zutage, wenn der Schlüssel akzeptiert wurde, ohne eine Überprüfung des Fingerabdrucks durchzuführen. Wenn Nutzer:innen auch die Überprüfung durchgeführt haben, wird das Symbol durch ein grünes ausgetauscht (Abbildung 40). Somit zeigt Thunderbird verschiedene Stufen der Vertrauenswürdigkeit an (K9, K13), doch reicht es nicht aus, wenn ein Schlüssel mithilfe der WKD-Funktion bezogen wurde, damit dieser eine höhere Stufe erreicht, als andere Schlüssel ohne Informationen über ihre Vertrauenswürdigkeit (K11).

Abbildung 40: Thunderbird – Signatur mit einem akzeptierten Schlüssel

Das Symbol auf der rechten Seite beschreibt das Owner Trust des Schlüssels, der zum Signieren verwendet wurde. In diesem Fall wurde der Schlüssel zuvor in Thunderbird akzeptiert.

Weder bei der Signaturüberprüfung noch beim Versenden einer Mail findet eine Bevorzugung von per WKD bezogenen Schlüsseln statt (K6, K7). Wenn zwei Schlüssel ohne Informationen zur Vertrauenswürdigkeit direkt importiert werden und einer per WKD bezogen wird, wird der Schlüssel bei der Überprüfung herangezogen, der dieselbe Schlüssel-ID hat wie der Schlüssel, der die Mail signiert hat. Auch das Verhalten bei der Verschlüsselung wird

vom Test betrachtet. Dabei werden drei verschiedene Durchläufe durchgeführt. Ein Schlüssel wird per WKD bezogen, während zwei weitere direkt importiert werden. Nur die Reihenfolge, in der dies geschieht, variiert bei diesem Versuch. Tabelle 4 zeigt, welche Schlüssel Thunderbird daraufhin für die Verschlüsselung verwendet. Es wurde nicht jede Kombination ausprobiert. Dennoch lässt das Ergebnis die Vermutung zu, dass der Schlüssel verwendet wird, der zuerst zu Thunderbird hinzugefügt wurde.

Reihenfolge für das Hinzufügen von Schlüsseln	**Verwendeter Schlüssel**
WKD, Import 1, Import 2	WKD
Import 1, WKD, Import 2	Import 1
Import 2, Import 1, WKD	Import 2

Tabelle 4: Bevorzugung von Schlüsseln in Thunderbird

Der kurze Test legt nahe, dass die Schlüssel für die Verschlüsselung verwendet werden, die zuerst importiert wurden.

K9Mail

Für den Test des Clients K9Mail wird die Version 5.806 herangezogen. Voraussetzung für die Verschlüsselung in der Android-Applikation K9Mail ist die Anwendung OpenKeyChain. Der Test von OpenKeyChain zeigt, dass diese App nicht kompatibel zum aktuellen Entwurf des WKD-Standards ist. Somit gilt dieselbe Aussage auch für K9Mail (K1). Nach der Installation von OpenKeyChain müssen noch einige Handgriffe in den Einstellungen von K9Mail durchgeführt werden, damit die App auf die Schlüssel zugreifen kann (K2). In den geöffneten Einstellungen wählen Nutzer:innen den Account aus, für den die Verschlüsselung aktiviert werden soll. Im Fenster mit den accountspezifischen Einstellungen (Abbildung 41) heißt einer der Punkte “Ende-zu-Ende Verschlüsselung”. Wenn sie diesen öffnen, erscheint ein Fenster, in dem sie die Option “OpenPGP-Unterstützung aktivieren” einschalten können. Ein kleiner Text unter dieser Option weist nach der Aktivierung darauf hin, dass die Anwendung mit OpenKeyChain verbunden ist (Abbildung 42). Direkt die nächste Option mit der Beschreibung “Configure end-to-end key” und der Beschriftung “Kein Schlüssel ausgewählt” muss ausgewählt werden, um den eigenen Schlüssel festzulegen (Abbildung 43 unten). Danach ist die Anwendung bereit für die Verschlüsselung.

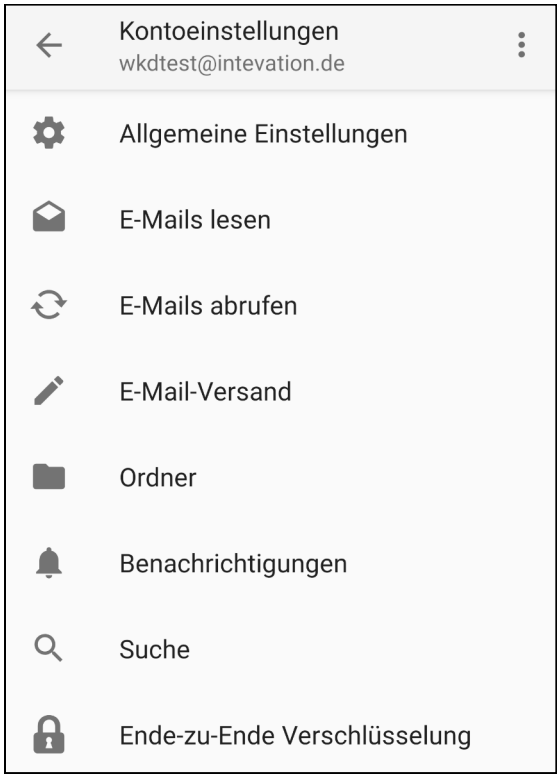


Abbildung 41: K9Mail – Kontoeinstellungen

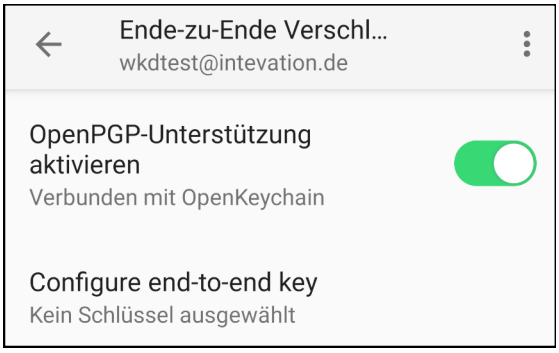


Abbildung 42: K9Mail – Verschlüsselung aktiviert

Wenn Nutzer:innen jetzt eine Mail verfassen und dabei eine Mail-Adresse eintippen, zeigt die Applikation automatisch bei den ersten eingegebenen Buchstaben Vorschläge für Adressen. Diese stammen aus dem Adressbuch des Smartphones, aber auch aus der Anwendung OpenKeyChain, wenn dort

bereits Schlüssel importiert wurden. Zudem lässt ein Schlüssel-Icon neben den Vorschlägen erkennen, für welche Mail-Adressen bereits ein Schlüssel vorhanden ist (Abbildung 43). Nachdem Anwender:innen die Mail-Adresse komplett eingegeben haben, zeigt ein schwarzes Dreieck im Feld der Adresse an, dass ein Schlüssel verfügbar ist. Außerdem signalisiert ein durchgestrichenes Schloss als Icon neben der eigenen Mail-Adresse, wenn für alle eingegebenen Adressen ein Schlüssel vorhanden ist (Abbildung 44). Durch das Antippen dieses Schlosses werden sowohl dieses als auch das schwarze Dreieck grün und die Verschlüsselung aktiviert (Abbildung 45). K9Mail nutzt dasselbe grüne Schloss sowohl, wenn ein Schlüssel direkt in OpenKeyChain importiert wurde, als auch dann, wenn der Schlüssel mithilfe der WKD-Funktion geholt wurde. Folglich bescheinigt die Anwendung den per WKD bezogenen Schlüsseln keine höhere Vertrauenswürdigkeit (K10). Wenn der Fingerabdruck des verwendeten Schlüssels in OpenKeyChain bestätigt wurde, verändert sich das Symbol, indem noch drei zusätzliche Punkte angezeigt werden (Abbildung 46). Durch dieses Verhalten erfüllt K9Mail das Kriterium K8, aber nicht K12, da keine weiteren Stufen entdeckt werden können.

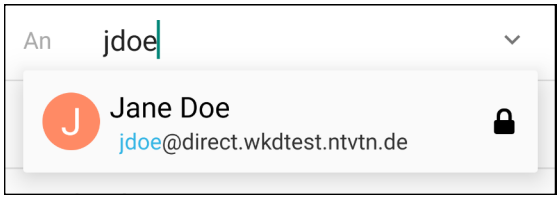


Abbildung 43: K9Mail – Anzeige vorhandener Schlüssel

Bei den Vorschlägen wird angezeigt, ob ein Schlüssel zu einer Adresse vorhanden ist.

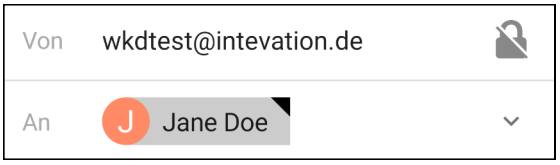


Abbildung 44: K9Mail – Graues Schloss

Ist die Möglichkeit zur Verschlüsselung gegeben, wird ein graues Schloss angezeigt.

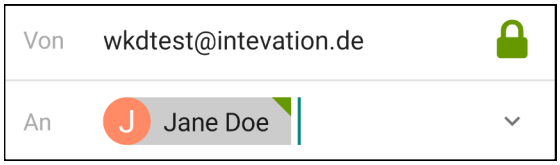


Abbildung 45: K9Mail – Grünes Schloss

Ein grünes Schloss zeigt an, dass die Mail verschlüsselt wird.

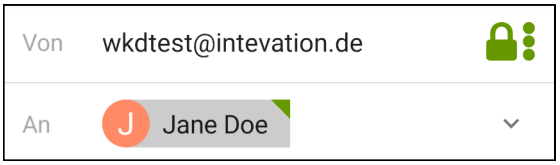


Abbildung 46: K9Mail – Grünes Schloss mit Punkten

Die drei Kreise bestätigen, dass der Fingerabdruck des Schlüssels in OpenKeyChain bestätigt wurde.

Geben Nutzer:innen ausschließlich Mail-Adressen ein, deren Schlüssel nicht vorhanden ist, wird kein Schloss angezeigt. Sie können aber oben rechts im Fenster einen Button betätigen, der ein Menü öffnet. Hier können sie dann versuchen, die Verschlüsselung trotzdem zu aktivieren. Daraufhin wird eine Meldung angezeigt, die erklärt, dass die Verschlüsselungsoption nur dann angezeigt wird, wenn sie von den Empfänger:innen unterstützt wird (Abbildung 47). Schließen Anwender:innen diese Meldung, erscheint ein rotes Schloss mit einem “X” an der Stelle, an der z.B. auch das graue Schloss

platziert war (Abbildung 48). Wird dieses angetippt, wird eine andere Meldung angezeigt, die darauf hinweist, dass eine Verschlüsselung nicht möglich ist (Abbildung 49). Als Ausweg wird angeboten, die Verschlüsselung zu deaktivieren. In K9Mail wird zwar angezeigt, ob ein Schlüssel für eine Mail-Adresse vorhanden ist, es wird jedoch nicht ermöglicht, einen Schlüssel aus der Anwendung heraus zu suchen (K3). Zudem bezieht die Anwendung Schlüssel nicht automatisch per WKD (K5). Dafür müssen Benutzer:innen die Applikation OpenKeyChain aufsuchen.

Abbildung 47: K9Mail – Meldung zur Verschlüsselung

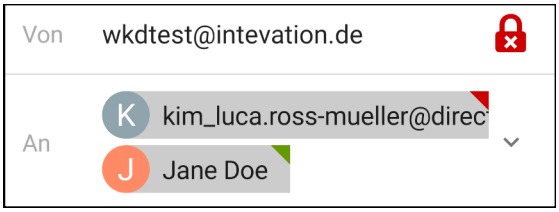


Abbildung 48: K9Mail – Rotes Schloss

Fehlt ein Schlüssel wird dies durch ein rotes Schloss verdeutlicht.

Abbildung 49: Keine Verschlüsselung möglich

Bei einer erhaltenen Mail, die nur signiert, aber nicht verschlüsselt wurde, erscheint neben dem Betreff ein blaues Symbol mit einem Häkchen, wenn der öffentliche Schlüssel, der für die Signatur verwendet wurde, in OpenKeyChain vorhanden ist (Abbildung 50). Wurde der Fingerabdruck in OpenKeyChain bestätigt, treten zusätzlich drei blaue Punkte rechts neben dem blauen Kreis auf (Abbildung 51, K9). Hat ein Schlüssel die Mail signiert, der nicht in OpenKeyChain verwaltet wird, erscheint ein orangefarbener Kreis mit einem Fragezeichen (Abbildung 52). Auch wenn ein Schlüssel für dieselbe Mail-Adresse vorhanden ist, dieser aber nicht dieselbe Schlüssel-ID aufweist, wird das orangefarbene Symbol dargestellt. Damit zeigt K9Mail, dass per WKD bezogene Schlüssel kein größeres Vertrauen bei der Signaturüberprüfung genießen (K11). Weitere Stufen außer den genannten werden nicht angezeigt (K13). In OpenKeyChain ist es nämlich nur möglich, einen Fingerabdruck zu bestätigen oder festzulegen, dass dieser nicht stimmt. Nutzer:innen haben keine Möglichkeit, das Vertrauen in einen Schlüssel in verschiedenen Stufen auszudrücken. Die drei Punkte, die in K9Mail verwendet werden, lassen vermuten, dass weitere Stufen, wie eine

marginale Validität, angezeigt werden können, doch da es keine Abstufungen für das Vertrauen in OpenKeyChain gibt, zeigt K9Mail diese Stufen nicht an. Sollte ein Schlüssel fehlen haben Anwender:innen keine Möglichkeit, eine Schlüsselsuche zu starten (K4).

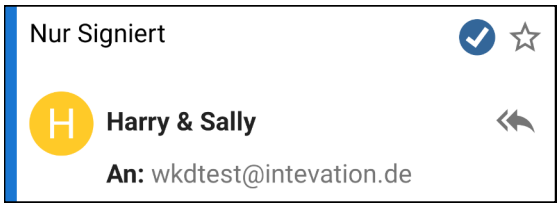


Abbildung 50: K9Mail – Signatur mit einem vorhandenen Schlüssel

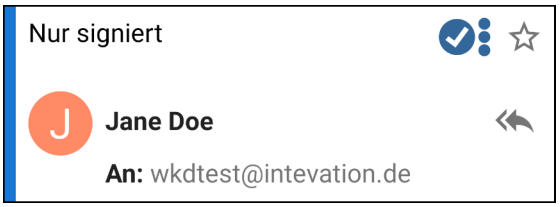


Abbildung 51: K9Mail – Signatur mit einem bestätigten Schlüssel

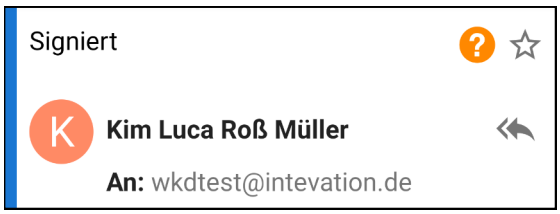


Abbildung 52: K9Mail – Signatur mit einem unbekannten Schlüssel

Ein weiteres zu testendes Kriterium ist, ob K9Mail einen Schlüssel zur Verschlüsselung einsetzt, der durch die WKD-Funktion bezogen wurde, wenn sonst nur Schlüssel vorhanden sind, die direkt importiert wurden, aber keine Informationen zur Vertrauenswürdigkeit besitzen. Hat OpenKeyChain zuerst einen Schlüssel per WKD geholt, bevor es zwei andere direkt importiert, wird bei der Verschlüsselung der Schlüssel verwendet, der per WKD heruntergeladen wurde. Um auszuschließen, dass die Reihenfolge eine Rolle spielt, in der die Schlüssel zu OpenKeyChain hinzugefügt wurden, wurde das Hinzufügen der Schlüssel auch in anderen Reihenfolgen durchgeführt und der Test erneut durchgeführt. Das Ergebnis bleibt bestehen: K9Mail entscheidet sich für den Schlüssel mit der höheren Vertrauenswürdigkeit (K6). Tabelle 5 stellt dar, in welcher Reihenfolge die Schlüssel bei diesem kurzen Experiment hinzugefügt wurden und welcher Schlüssel bei der Verschlüsselung verwendet wurde. Anders sieht es bei der Überprüfung der Signatur aus. Hier verwendet K9Mail den Schlüssel, der zum Schlüsselpaar gehört, dessen privater Schlüssel für die Signatur eingesetzt wurde (K7).

Reihenfolge für das Hinzufügen von Schlüsseln	Verwendeter Schlüssel
WKD, Import 1, Import 2	WKD
Import 1, WKD, Import 2	WKD
Import 2, Import 1, WKD	WKD

Tabelle 5: Bevorzugung von Schlüsseln in K9Mail

Unabhängig von der Reihenfolge beim Hinzufügen der Schlüssel entscheidet sich K9Mail dafür, den per WKD bezogenen Schlüssel bei der Verschlüsselung einzusetzen.

FairEmail

Der Client FairEmail wurde in der Version 1.1776 getestet. Für die Verschlüsselung mithilfe von OpenPGP-Schlüsseln ist die Installation der Anwendung OpenKeyChain eine Voraussetzung. Ohne OpenKeyChain können Benutzer:innen in FairEmail keine Schlüssel verwenden (K2). Wie für K9Mail gilt auch für FairEmail, dass die Anwendung nicht kompatibel zum aktuellen Entwurf des WKD-Standards ist, da bereits OpenKeyChain diese Kompatibilität nicht besitzt (K1). In den Einstellungen von FairEmail finden Nutzer:innen einen Reiter mit der Beschriftung “Verschlüsselung”. Der dazugehörige Bereich erlaubt ihnen, eine Anwendung auszuwählen, die zur Verschlüsselung verwendet werden soll (Abbildung 53). Wenn die Applikation OpenKeyChain installiert ist, ist diese bereits ausgewählt. Somit ist es nicht notwendig, dass Benutzer:innen Änderungen in den Einstellungen von FairEmail vornehmen müssen, um eine Mail verschlüsseln zu können.

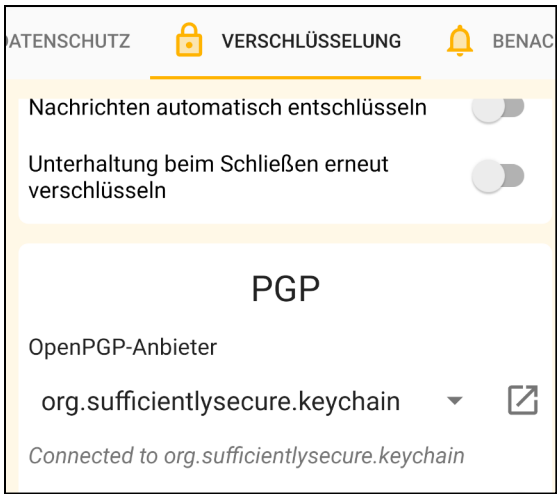


Abbildung 53: FairEmail – Verbunden mit OpenKeyChain
Die Anwendung ist ohne das Zutun von Benutzer:innen mit OpenKeyChain verbunden.

Beim Schreiben einer Mail haben Benutzer:innen die Möglichkeit, den Button mit dem Schloss-Icon zu betätigen, um die Verschlüsselung zu aktivieren (Abbildung 54). Daraufhin erhält das Symbol eine grüne Farbe und der Button zum Abschicken der Mail erhält dann die Beschriftung “Verschlüsseln”. Es ist ebenfalls möglich, direkt auf “Senden” zu tippen. In diesem Fall erscheint ein Dialog, in dem Nutzer:innen noch Einstellungen vornehmen können, bevor die Nachricht endgültig abgeschickt wird (Abbildung 55). Auf diese Weise können sie auch hier die Verschlüsselung aktivieren. Wenn Anwender:innen versuchen, die Mail zu verschicken, obwohl der Schlüssel für eine Mail-Adresse nicht vorhanden ist, taucht ein Fenster der Anwendung OpenKeyChain auf, das auf diesen Umstand hinweist (Abbildung 56). In diesem Fenster können sie öffentliche Schlüssel auswählen, die sie bereits mit der Applikation importiert haben. Nach einer Bestätigung durch das Antippen des Buttons “OKAY” werden diese Schlüssel zum Verschlüsseln dieser Mail verwendet. Zu beachten ist dabei, dass diese Schlüssel nicht zu der eingegebenen Mail-Adresse passen. Sonst würde

dieses Fenster nicht erscheinen. Falls Anwender:innen hier dennoch einen Schlüssel auswählen, erhalten Empfänger:innen, an deren Adresse die Mail geschickt wird, eine verschlüsselte Mail, die sie nicht entschlüsseln können.

Wenn die passenden Schlüssel vor dem Versuch, die Mail zu versenden, vorhanden sind, verschlüsselt FairEmail die Mail beim Absenden. Bevor dies geschieht, gibt es allerdings keinen Indikator, ob die Verschlüsselung erfolgreich sein wird. Zudem gibt es im Fenster zum Schreiben einer Mail keine Möglichkeit, Schlüssel zu beziehen noch werden Schlüssel automatisch geholt (K3, K5). Ob ein Schlüssel für eingegebene Mail-Adressen vorhanden ist, können Nutzer:innen im Fenster zum Schreiben einer Mail nicht sehen. Somit erhalten sie auch keine Auskunft darüber, wie vertrauenswürdig die verwendeten Schlüssel sind (K8, K10, K12).

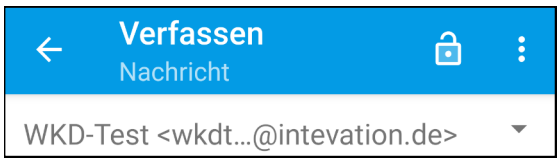


Abbildung 54: FairEmail – Schloss-Icon

Das Schloss-Icon dient zum Aktivieren der Verschlüsselung der aktuellen Mail.

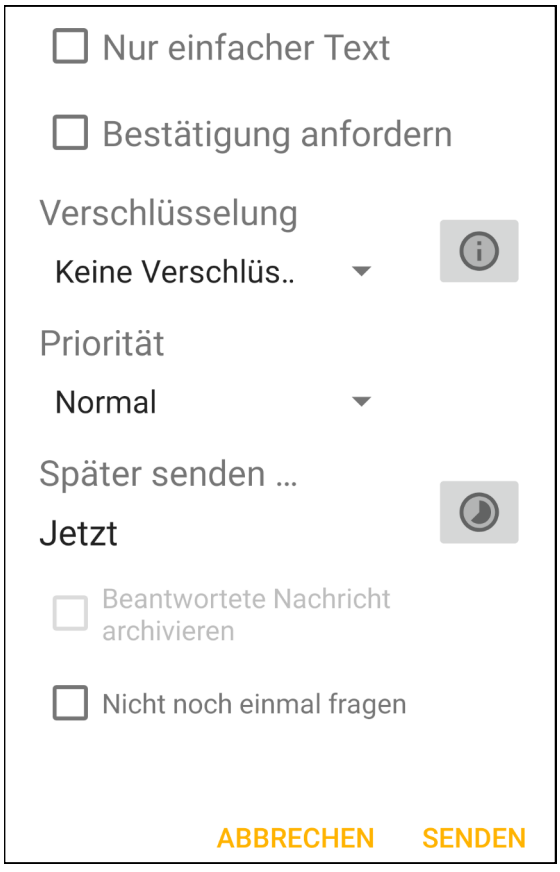


Abbildung 55: FairEmail – Einstellungen vor dem Senden

Vor dem endgültigen Absenden der Mail können noch Einstellungen vorgenommen werden – auch Einstellungen zur Verschlüsselung.

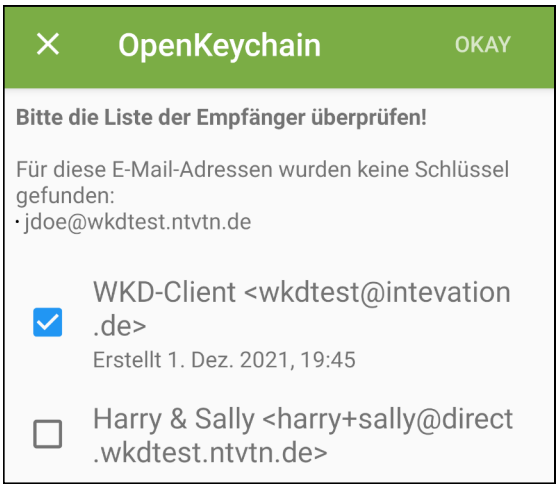


Abbildung 56: FairEmail – Fehlender Schlüssel

Fehlt der Schlüssel zu einer Adresse, wird beim Versuch, die Mail zu senden, dieses Fenster angezeigt.

Nachdem die Anwendung eine erhaltene Mail, die signiert und verschlüsselt ist, entschlüsselt hat, erscheint ein Button, den Anwender:innen antippen können, um die Signaturüberprüfung zu starten (siehe Abbildung 57). Außerdem erscheint schon beim Entschlüsseln die Meldung „Unsignierte Nachricht“, obwohl die Mail signiert ist. Diese Meldung ist ein Fehler der Anwendung, welcher an den Hauptentwickler von FairEmail weitergeleitet wurde, wie im Kapitel zu den umgesetzten Maßnahmen zu lesen ist. Nachdem Benutzer:innen den Button zur Überprüfung der Signatur betätigen, wird eine Fehlermeldung am unteren Rand der Anwendung sichtbar, wenn kein geeigneter öffentlicher Schlüssel zum Schlüsselbund von OpenKeyChain gehört (Abbildung 58). Dabei sucht FairEmail keine Schlüssel mithilfe des WKD-Verfahrens, um diesen Zustand zu ändern. Die grafische Oberfläche verfügt an dieser Stelle auch nicht über einen Button, den Benutzer:innen antippen können, um eine Schlüsselsuche zu starten (K4). Wenn der Schlüssel jedoch vorhanden ist und die Signatur gültig ist, gibt eine Meldung, wie sie in Abbildung 59 zu sehen ist, diese Information weiter. Diese Meldung verschwindet jedoch nach kurzer Zeit und verrät keine weiteren Informationen über den Schlüssel, mit dem die Mail signiert wurde. Auch die restliche Oberfläche gibt keinen Hinweis auf solche Informationen (K9, K11, K13).

Abbildung 57: FairEmail – Signatur überprüfen
Nach dem Entschlüsseln einer Nachricht ist ein Button zum Überprüfen der Signatur verfügbar.

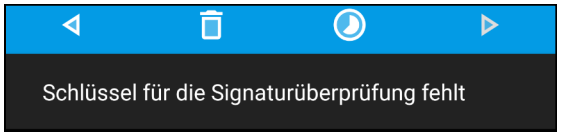


Abbildung 58: FairEmail – Fehlgeschlagene Signaturüberprüfung

Abbildung 59: FairEmail – Gültige Signatur

Aus diesem Grund kann auch nicht gesagt werden, welcher Schlüssel für die Überprüfung verwendet wurde und ob die Schlüssel, die eine größere Vertrauenswürdigkeit besitzen, von FairEmail bevorzugt werden (K7). Das Kriterium in Bezug auf das Versenden einer Mail kann hingegen getestet werden. Verwahrt OpenKeyChain mehrere Schlüssel, wobei ein Schlüssel per WKD bezogen wurde und die anderen, die keine Informationen zu ihrer Vertrauenswürdigkeit enthalten, direkt importiert wurden, zeigt der Mail-Client einen Dialog zur Auswahl eines Schlüssels an (Abbildung 60). Somit ist das Ergebnis, dass mithilfe von WKD bezogene Schlüssel durch den Client nicht bevorzugt und automatisch verwendet werden, wenn sonst nur Schlüssel ohne Informationen zur Vertrauenswürdigkeit vorhanden sind (K6).

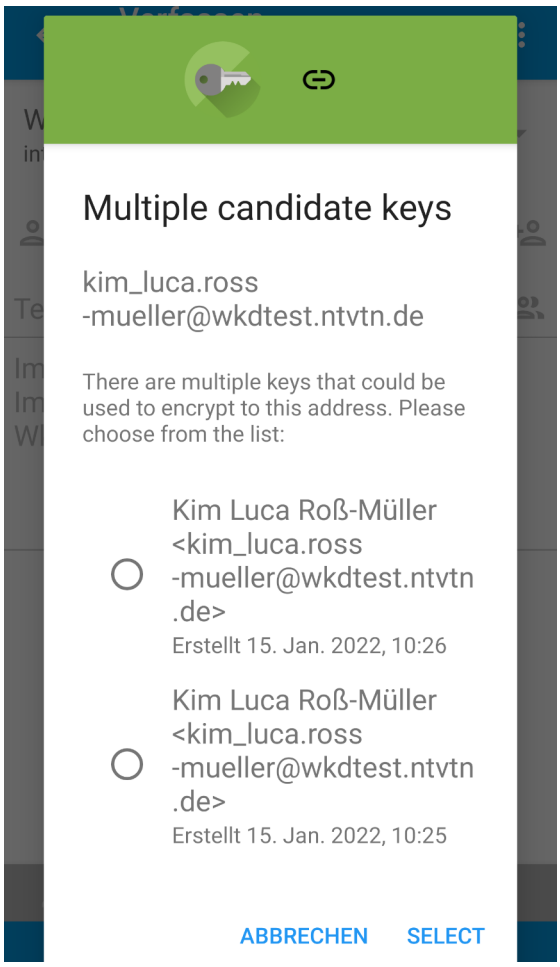


Abbildung 60: FairEmail – Schlüsselauswahl

OpenKeyChain

Die Anwendung OpenKeyChain ist kein Mail-Client, sondern dafür zuständig, private und öffentliche OpenPGP-Schlüssel zu verwalten. Andere Applikationen können auf den Schlüsselbund von OpenKeyChain zugreifen und die Schlüssel zur Verschlüsselung verwenden. Zur Verwaltung gehört neben dem Importieren auch das Suchen nach Schlüsseln in externen Quellen. Beim Testen wurde Version 5.7.5 der Anwendung untersucht. Damit die Suche auch das WKD-Verfahren einbezieht, muss dies in den Einstellungen festgelegt werden. Nach der Installation ist das standardmäßig der Fall, sodass Anwender:innen keine Änderungen vornehmen müssen.

In der Hauptansicht von OpenKeyChain befindet sich unten rechts ein Button mit einem “+”. Wenn Nutzer:innen diesen betätigen, erscheinen oberhalb dieses Buttons weitere Buttons, von denen einer zur Schlüsselsuche führt (Abbildung 61). In dem Fenster, das geöffnet wird, können Nutzer:innen eine Mail-Adresse eingeben und die Suche nach dem Schlüssel mit Enter starten. Ist die Suche erfolgreich, sehen sie ein Feld, in dem Details zu dem gefundenen Schlüssel aufgelistet werden (Abbildung 62). Durch das Tippen auf den Button „Import“ bestätigen sie, dass der Import durchgeführt werden soll und OpenKeyChain fügt den gefundenen Schlüssel zum Schlüsselbund hinzu.

Auch in OpenKeyChain wird versucht, die Schlüssel aller Test-Mail-Adressen zu erhalten. Die Anwendung verhält sich nicht so, wie es der WKD-Standard vorschreibt, weil die Direct Method verwendet wird, auch wenn die Subdomain *openpgpkey* vorhanden ist. Damit sind weder OpenKeyChain noch Anwendungen, die auf OpenKeyChain zugreifen, kompatibel zum aktuellen Entwurf des WKD-Standards.

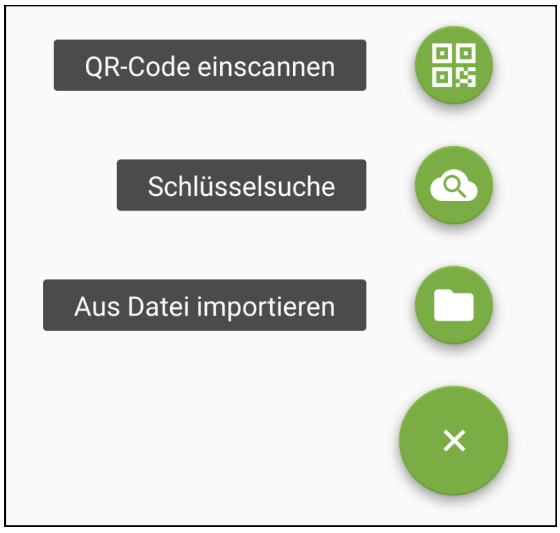


Abbildung 61: OpenKeyChain – Öffnen der Schlüsselsuche

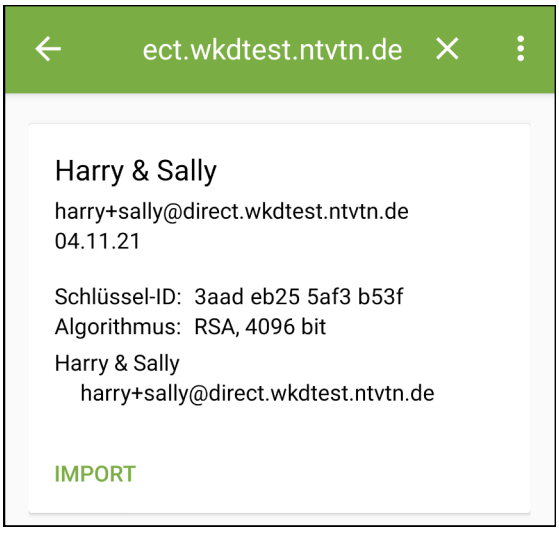


Abbildung 62: OpenKeyChain – Importieren eines Schlüssels

KMail

Die Software KMail von KDE ist für verschiedene GNU/Linux-Distributionen verfügbar. Der Test zieht die Version 5.18.3 heran. Ebenso wie in Mailvelope wird in KMail automatisch nach einem öffentlichen Schlüssel gesucht, wenn eine Mail-Adresse eingegeben wurde. Damit dies geschieht, ist es allerdings notwendig, dass Benutzer:innen die WKD-Funktion in den Einstellungen aktivieren (K2).

Dazu müssen sie zuerst in der Menüleiste das Menü „Settings“ öffnen und dann auf „Configure KMail…“ klicken (Abbildung 63), um die Einstellungen zu öffnen, in denen auch die Konten konfiguriert werden können (Abbildung 64). Dafür wählen sie das jeweilige Konto aus und klicken dann auf „Modify…“. Es öffnet sich ein Fenster, in dem sie zum Reiter „Cryptography“ wechseln. In diesem Bereich gibt es eine Option mit der Beschreibung „Automatically encrypt messages when possible“ (Abbildung 65). Zusammen mit dieser Option aktivieren die Nutzer:innen auch die WKD-Funktion.

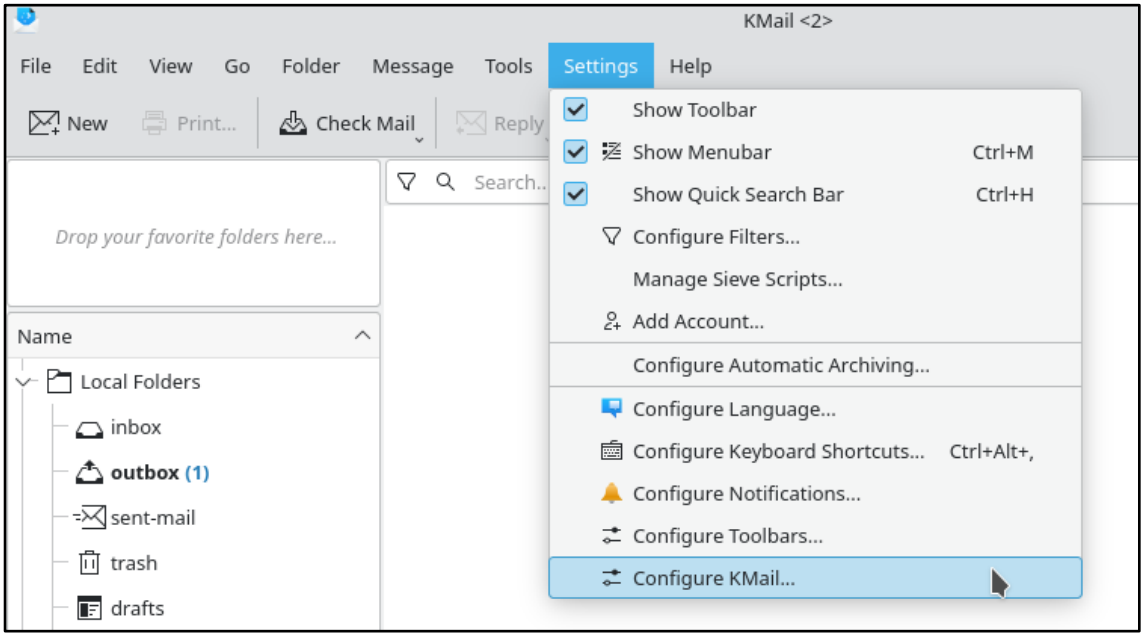


Abbildung 63: KMail – Accounts

Wenn Benutzer:innen danach eine Mail schreiben und eine Mail-Adresse eingegeben haben, dauert es eine kurze Zeit, bis links neben der Adresse ein Icon angezeigt wird, das verdeutlicht, dass ein Schlüssel gefunden wurde (Abbildung 66). Demzufolge führt KMail die WKD-Funktion hier ohne eine Interaktion von Anwender:innen aus (K3, K5). Das Aussehen des Icons kann variieren. Bei einem Schlüssel mit unbekannter Vertrauenswürdigkeit wird ein blaues Quadrat mit einem „i“ angezeigt. Dass keine Informationen zur Vertrauenswürdigkeit vorhanden sind, teilt jedoch erst ein Tooltipp mit, wenn der Mauszeiger über das Icon bewegt wird (Abbildung 67). Auch, wenn die Anwendung einen Schlüssel mithilfe der WKD-Funktion erhält, zeigt sie das Symbol mit dem „i“ an. Es findet also keine Unterscheidung zwischen

diesen beiden Arten von Schlüsseln bzw. deren Quellen statt (K10). Wenn Benutzer:innen mithilfe von GnuPG angeben, dass sie einem Schlüssel ultimativ vertrauen, verändert sich das Icon und enthält einen Stern (Abbildung 68). Somit werden nur zwei verschiedene Stufen des Vertrauens dargestellt (K8, K12).

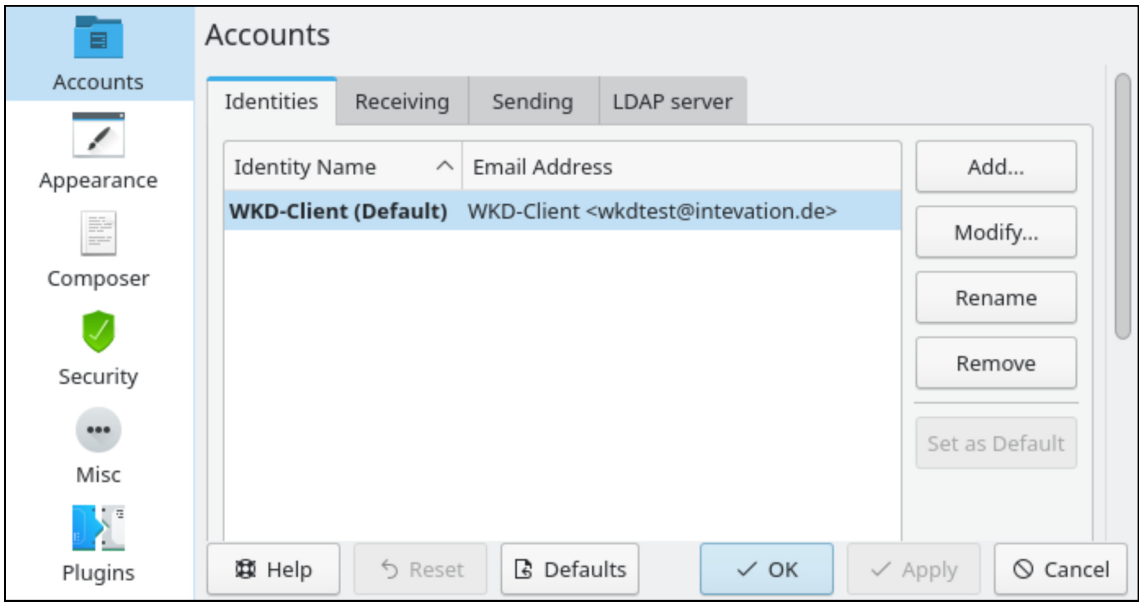


Abbildung 64: KMail – Accounts

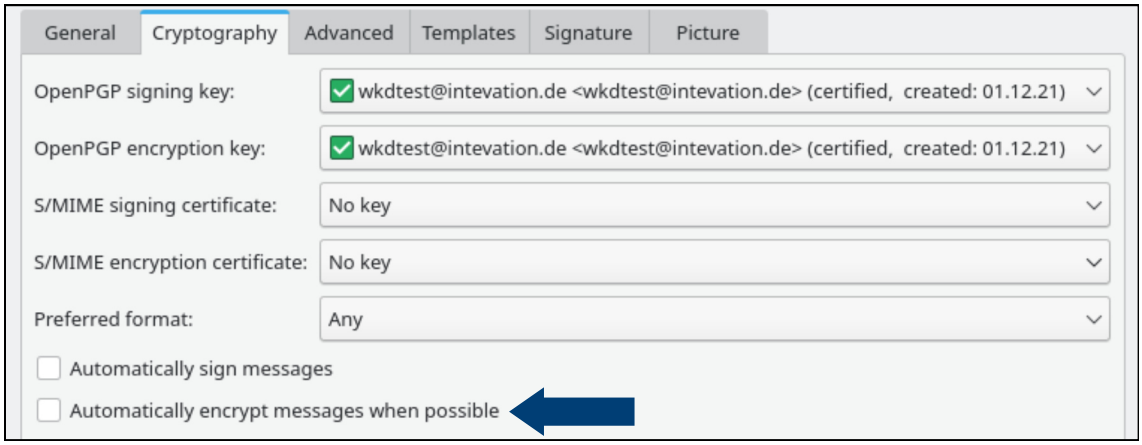


Abbildung 65: KMail – Einstellungen für die Verschlüsselung

Wenn die Option „Automatically encrypt messages when possible“ aktiviert ist, findet auch die Suche nach öffentlichen Schlüsseln automatisch statt.

Abbildung 66: KMail – Schlüssel mit unbekannter Vertrauenswürdigkeit

Nachdem ein Schlüssel für eine Adresse gefunden wurde, wird dies durch ein Icon links neben der Adresse veranschaulicht. In diesem Fall gibt es keine Informationen über die Vertrauenswürdigkeit.

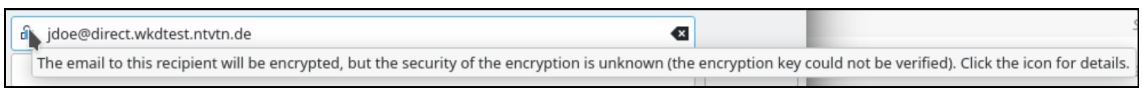


Abbildung 67: KMail – Tooltipp für den Schlüssel mit unbekannter Vertrauenswürdigkeit

Abbildung 68: KMail – Schlüssel mit ultimativer Vertrauenswürdigkeit

Wenn User:innen eine signierte Nachricht öffnen, überprüft KMail die Signatur automatisch. Wenn sich ein Schlüssel auf einem WKD-Server befindet und der Anwendung noch nicht zur Verfügung steht, wird dieser nicht automatisch geholt. Stattdessen bekommen Nutzer:innen einen Text wie in Abbildung 69 zu Gesicht. Sie haben in diesem Bereich auch keine Möglichkeit, nach einem Schlüssel zu suchen (K4). Ist jedoch ein Schlüssel vorhanden, mit dem die Signaturüberprüfung durchgeführt werden kann, verrät der Text nach der Prüfung, wie stark das Vertrauen in den verwendeten Schlüssel ist (Abbildung 70). Hier zeigt die Applikation ebenfalls nur zwei Stufen an: Entweder ist das Vertrauen unbekannt oder dem Schlüssel kann vollständig vertraut werden. Letzteres ist nur der Fall, wenn Anwender:innen in GnuPG angeben, dass sie dem Schlüssel ultimativ vertrauen. Marginales oder volles Vertrauen erscheinen in KMail als „unbekannt“ (K9, K13). Wenn ein Schlüssel von einem WKD-Server stammt, besagt der Text, dass es keine Informationen zur Vertrauenswürdigkeit gibt (Abbildung 71). Wenn ein Schlüssel direkt importiert wird, sieht der Text ebenfalls so aus (K11).

Abbildung 69: KMail – Kein Schlüssel für die Signaturüberprüfung vorhanden

Wenn kein Schlüssel für die Signaturüberprüfung vorhanden ist, wird die WKD-Funktion nicht verwendet.

Abbildung 70: KMail – Vertrauen bei der Signaturüberprüfung

In der Signatur wird eine Information zum Vertrauens des Schlüssels angezeigt.

Abbildung 71: KMail – Vertrauenswürdigkeit eines Schlüssels von einem WKD-Server

Der Schlüssel, der für die Überprüfung der Signatur ausgewählt wird, hängt nicht davon ab, ob die Software ihn durch die WKD-Funktion bezogen oder direkt importiert hat (K7). Eine mögliche Priorisierung wird auch beim Verfassen einer Mail getestet. Im Falle, dass Anwender:innen an dieser Stelle eine Mail-Adresse eingeben, für die mehrere Schlüssel vorhanden sind, zeigt KMail einen Dialog an, wie er in Abbildung 72 zu sehen ist. Dort können Benutzer:innen den Schlüssel auswählen, den sie für die Verschlüsselung verwenden möchten. Die Anwendung wählt den Schlüssel nicht selbstständig aus (K6).

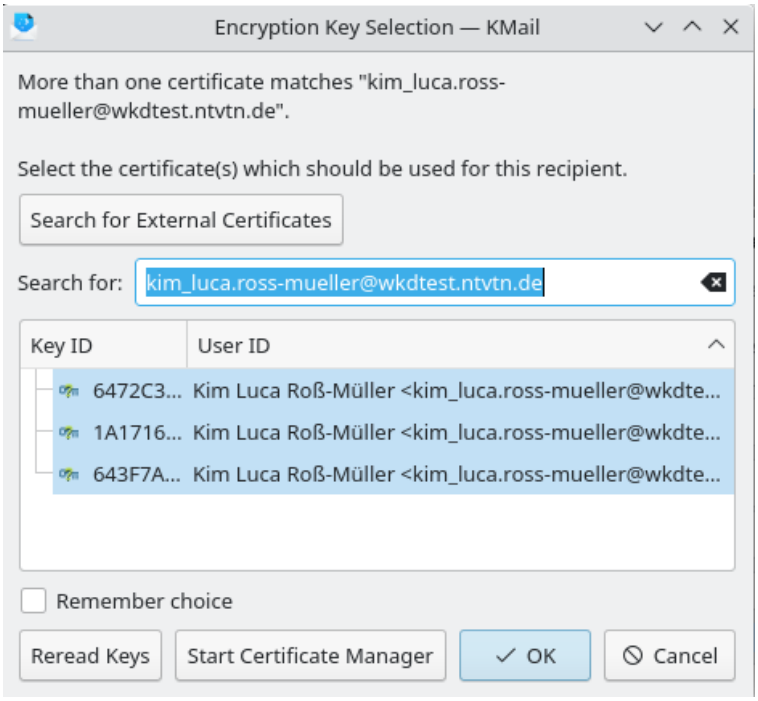


Abbildung 72: KMail – Schlüsselauswahl

Zum Schluss wird auch bei KMail überprüft, ob der WKD-Standard korrekt implementiert wurde. Nach dem Versuch, die Schlüssel der Test-Mail-Adressen zu beziehen, werden genau diejenigen erhalten, die bei korrekter Implementierung des aktuellen Entwurfs des WKD-Standards zu bekommen sein sollten. Dies bestätigt, dass KMail in dieser Hinsicht so funktioniert, wie der Standard es vorschreibt (K1).

mail.de

Während des Zeitraums der Arbeit ist beim Betrachten der Einstellungen für einen Account bei mail.de aufgefallen, dass das Unternehmen einen WKD-Server betreibt. Auch wenn das Angebot von mail.de nicht zur Auswahl der Produkte gehört, soll ein Test durchgeführt werden, um zu überprüfen, ob die Schlüssel per WKD geholt werden können, da der Test eines WKD-Servers mit einem geringeren Aufwand verbunden ist. Ein Test mit GnuPG 2.2.27 und dem Befehl **`gpg -vv –locate-keys –auto-key-locate clear,nodefault,wkd`** zusammen mit einer Mail-Adresse als Parameter, die zu einem Account bei mail.de gehört, zeigt, dass Benutzer:innen Schlüssel nicht mithilfe der Advanced Method von WKD erhalten können. Die Ursache für diese Tatsache ist, dass die Domain *openpgpkey.mail.de* ohne Fehler aufgerufen werden kann, der Aufruf dann jedoch auf die Domain **`mail.de`** weitergeleitet wird [31]. Im zwölften Entwurf für den Standard WKD steht jedoch: “Only if the required sub-domain does not exist, they SHOULD fall back to the direct method” [9]. Da die Subdomain *openpgpkey* von mail.de aufgerufen werden kann, versucht GnuPG nicht mehr, die Direct Method durchzuführen. Wenn die URL, die bei der Direct Method erstellt wird, direkt in den Browser eingegeben und dann auf-gerufen wird, kann der Key heruntergeladen werden, sodass er importiert und verwendet werden kann. Das lässt darauf schließen, dass der Schlüssel von mail.de korrekt gespeichert wurde.

Zusammenfassung der Tests

Die folgende Tabelle hält fest, ob die getesteten Produkte die Kriterien erfüllen, die das Kapitel 3 eingeführt hat. Wie bereits der Abschnitt zu Mailvelope erläutert, ist eine Signatur einer erhaltenen Mail auf mailbox.org im Test nicht sichtbar gewesen. In den häufig gestellten Fragen wird allerdings behauptet, dass die Signatur überprüft wird. Dazu kommt, dass die Verwendung von Mailvelope auf den Seiten verschiedener Mail-Provider möglich ist. Der Aufwand, Mailvelope auf verschiedenen Seiten zu testen, wird in dieser Arbeit nicht betrieben. Aus diesem Grund werden keine Ergebnisse für Mailvelope bei den Kriterien für die Signatur eingetragen.

Kriterium/Produkt	Claws Mail	FairEmail	K9Mail	KMail	Mailvelope	Thunderbird
K1: Kompatibel mit der aktuellen Version des WKD-Standards	✓	✗	✗	✓	✗	✗
K2: Ist ohne weitere Vorkehrungen direkt bereit für WKD	✗	✗	✗	✗	✓	✗
K3: WKD ist direkt dort verfügbar, wo eine Mail-Adresse eingegeben wird	✗	✗	✗	✓	✓	✗
K4: WKD ist direkt dort verfügbar, wo eine Signatur überprüft wird	✓	✗	✗	✗	?	✗
K5: WKD wird nach dem Eingeben einer Mail-Adresse **automatisch** verwendet	✗	✗	✗	✓	✓	✗
K6: Es wird automatisch der per WKD bezogene Schlüssel für die Verschlüsselung ausgewählt, wenn sonst nur Schlüssel ohne Informationen zur Vertrauenswürdigkeit vorhanden sind (beim **Verfassen**)	✗	✗	✓	✗	✗	✗
K7: Es wird automatisch der per WKD bezogene Schlüssel für die Verschlüsselung ausgewählt, wenn sonst nur Schlüssel ohne Informationen zur Vertrauenswürdigkeit vorhanden sind (bei der **Signaturüberprüfung**)	✗	✗	✗	✗	?	✗
K8: Unterscheidet bei der Anzeige zwischen Schlüssel ohne Informationen über die Vertrauenswürdigkeit und Schlüsseln, die mehr oder weniger Vertrauenswürdigkeit besitzen (beim **Verfassen**)	✗	✗	✓	✓	✗	✗
K9: Unterscheidet bei der Anzeige zwischen Schlüssel ohne Informationen über die Vertrauenswürdigkeit und Schlüsseln, die mehr oder weniger Vertrauenswürdigkeit besitzen (bei der **Signaturprüfung**)	✓	✗	✓	✓	?	✓
K10: Zeigt bei per WKD bezogenen Schlüsseln an, dass diese ein Mindestmaß an Vertrauenswürdigkeit besitzen (beim **Verfassen**)	✗	✗	✗	✗	✗	✗
K11: Zeigt bei per WKD bezogenen Schlüsseln an, dass diese ein Mindestmaß an Vertrauenswürdigkeit besitzen (bei der **Signaturüberprüfung**)	✗	✗	✗	✗	?	✗
K12: Zeigt verschiedene Stufen des Vertrauens an (beim **Verfassen**)	✗	✗	✗	✗	✗	✗
K13: Zeigt verschiedene Stufen des Vertrauens an (bei der **Signaturprüfung**)	✓	✗	✗	✗	?	✓

Tabelle 6: Erfüllung der Kriterien durch die getesteten Produkte

✓ = ja, ✗ = nein, ? = kein Ergebnis

5. Angewandte Maßnahmen

Anregungen zur Verbesserung der Usability von WKD in Produkten

Vorbereitung

Bevor Implementierungen vorgenommen oder Beiträge geleistet werden können, ist es notwendig, zu erfahren, an welchen Stellen Personen mit den Entwickler:innen eines Produkts interagieren können. Die Suche nach diesen Stellen macht deutlich, dass Entwickler:innen dafür verschiedene Kanäle verwenden, sodass kein einheitliches Vorgehen möglich ist. Bei Mailvelope findet die Kommunikation, wie bei vielen anderen Produkten, durch Fälle und Pull Requests statt, die auf der Plattform Github angelegt werden. Wollen Personen eigene Code-Beiträge leisten, müssen sie dafür neue Forks erstellen, deren Branches dann in das Hauptprojekt integriert werden. Auch für das Produkt K9Mail wird Github als zentrale Anlaufstelle für die Entwicklung verwendet.

Anders sieht es bei den Produkten Claws Mail und FairEmail aus. Für Claws Mail gibt es eine eigene Plattform, um Fälle zu erstellen (https://www.thewildbeast.co.uk/claws-mail/bugzilla/index.cgi). Diese Seite soll auch dafür verwendet werden, um Beiträge zum Quellcode einzureichen [32]. Daneben existieren aber auch noch Mailing-Listen (https://www.claws-mail.org/Mls.php), in die sich Personen eintragen können. In diesen Mails können sie Fragen stellen und Vorschläge für Veränderungen machen.

Der Hauptentwickler von FairEmail, Marcel Bokhorst, nutzt ein Forum (https://forum.xda-developers.com/t/app-5-0-fairemail-fully-featured-open-source-privacy-oriented-email-app.3824168/), um mit anderen Personen über den Mail-Client zu diskutieren und Vorschläge entgegen zu nehmen. Code-Beiträge können bei diesem Produkt in Form von Pull Requests bei Github eingebracht werden [33].

Claws Mail

Wie oben beschrieben ist das Verwenden der WKD-Funktion in Claws Mail nur an einer Stelle in der grafischen Oberfläche möglich, nämlich dort, wo eine erhaltene Mail geöffnet ist. Dadurch können Benutzer:innen jedoch nur dann einen öffentlichen Schlüssel per WKD beziehen, wenn sie bereits eine Mail der dazugehörigen Adresse erhalten haben. Hatten sie bisher noch keinen Kontakt zu der Person der Mail-Adresse, haben sie in der Anwendung keine Möglichkeit, an diesen Schlüssel zu gelangen.

Aus diesem Grund wird ein Vorschlag in einer Mailing-List von Claws Mail eingereicht, die WKD-Funktion im Fenster zum Verfassen einer Mail zu integrieren [34]. In diesem Fenster können Benutzer:innen, wie auch in

anderen Mail-Clients, im oberen Bereich die Mail-Adressen von Empfänger:innen eingeben. Bei einem Rechtsklick auf bereits eingegebene Adressen wird ein Kontextmenü sichtbar, welches einige Aktionen zulässt, z.B. das Hinzufügen dieser Adresse zu einem Adressbuch. Es wird vorgeschlagen, dort eine weitere Aktion hinzuzufügen, um einen Schlüssel für diese Adresse zu suchen. Im weiteren Verlauf der Kommunikation wird ergänzend erwähnt, dass die Anzeige dieser Aktion abhängig davon sein kann, ob ein Plug-in zur Verschlüsselung eingebunden wurde. Somit könnten nur Personen die Aktion sehen, für die diese Aktion relevant ist. Durch die Umsetzung dieses Vorschlags würde Claws Mail das Kriterium K3 erfüllen.

Um die Usability beim Einrichten der Verschlüsselung zu verbessern, wird zudem vorgeschlagen, dass Claws Mail das Privacy System automatisch auswählen soll, wenn vorher keines ausgewählt war und durch das Einbinden eines Plug-ins ein neues Privacy System zur Auswahl steht [35]. Ein Grund dafür ist, dass dies dazu führte, dass Benutzer:innen einen Schritt weniger bräuchten, um die Verschlüsselung zu aktivieren. Das Produkt wäre dann näher an der Erfüllung des Kriteriums K2. Der zweite Grund ist, dass vermutlich nicht für alle Anwender:innen verständlich ist, warum die Option “Encrypt” nicht auswählbar ist, wenn das passende Plug-in installiert wurde. Es mag auch sein, dass der Begriff Privacy System sich nicht so anhört, als hätte er direkt etwas mit der Option “Encrypt” zu tun. Bei einer automatischen Auswahl des Privacy Systems gäbe es diese Probleme nicht.

FairEmail

Da in FairEmail keine Informationen darüber angezeigt werden, ob Schlüssel zu eingegebenen Mail-Adressen vorhanden sind, wissen Anwender:innen vor dem Versuch, eine Mail zu verschicken nicht, ob eine Verschlüsselung erfolgreich durchgeführt werden kann. Deswegen wird vorgeschlagen, dass FairEmail im Fenster zum Schreiben einer Mail anzeigen soll, ob es einen Schlüssel für eine Adresse gibt oder nicht [36].

Zusätzlich wird empfohlen, einen Button in FairEmail zu integrieren, mit dessen Hilfe Schlüssel automatisch bezogen werden können. [37] Damit sparten sich Benutzer:innen das Navigieren zur App OpenKeyChain. Das wiederum bedeutete, dass sie einige Interaktionen weniger durchführen müssten und weniger Zeit damit verbrächten, an die Schlüssel zu gelangen. Zuletzt würde die Anwendung damit das Kriterium K3 erfüllen.

Das Kapitel, das die Testergebnisse wiedergibt, zeigt zudem, dass in FairEmail nach dem Entschlüsseln einer verschlüsselten signierten Mail eine Meldung auftaucht, dass die Mail nicht signiert wurde (Abbildung 73). Wenn Benutzer:innen dann auf den Button für die Signaturüberprüfung tippen, erscheint eine weitere Nachricht, die besagt, dass die Signatur gültig ist. Die erste Meldung ist somit ein Fehler in der Anwendung und wurde im Forum weitergegeben [38]. Kurz darauf hat der Hauptentwickler diesen Fehler in einer neuen Version von FairEmail behoben.

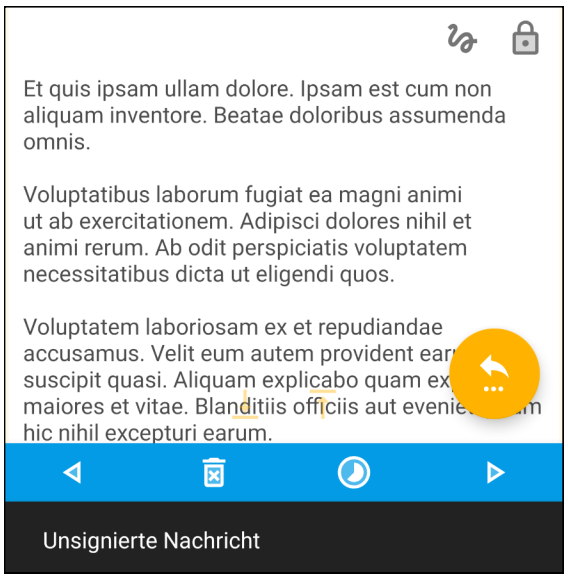


Abbildung 73: KMail – Schlüsselauswahl

mail.de

Um das Team von mail.de auf das Problem beim Beziehen von Schlüsseln per WKD hinzuweisen, wurde das Kontaktformular auf der Homepage von mail.de verwendet. In der folgenden Antwort des Entwicklers Michael Kliewe von mail.de gab dieser zu verstehen, dass der WKD-Standard in einer früheren Version implementiert wurde. Das Unternehmen verwendet Wildcard DNS, welches dafür sorgt, dass beim Aufruf der Subdomain *openpgpkey* kein Fehler auftritt. Für diesen Fall gibt der Standard vor, einen leeren TXT-Record für die Subdomain zu erstellen. So einen TXT-Record hat Micheal Kliewe deshalb angelegt, sodass die Schlüssel von Adressen des Providers mail.de nun auch mithilfe eines WKD-kompatiblen Clients be-zogen werden können.

Erstellung von Anleitungen

Das Ergebnis der Tests dieser Arbeit zeigt, dass Benutzer:innen einiger Produkte mehrere Schritte durchführen müssen, bevor sie WKD einsetzen können. Bei den meisten Produkten kommt dazu, dass es nicht offensichtlich ist, wie die WKD-Funktion verwendet werden kann. Deshalb gehört zu den umgesetzten Maßnahmen dieser Arbeit auch das Erstellen von Anleitungen, um WKD aktivieren und verwenden zu können. Die Schritte in den Anleitungen sind kurz und werden durch Screenshots ergänzt. Auf diesen zeigen Pfeile auf die Elemente in der grafischen Oberfläche, mit denen Benutzer:innen interagieren müssen, um den jeweiligen Schritt durchzuführen. Die Anleitungen wurden in der Wikipedia zu GnuPG veröffentlicht. Die Links zu den Anleitungen der jeweiligen Produkte sind in Tabelle 7 aufgelistet.

Für Claws Mail und Thunderbird ist es nötig, zu beschreiben, wie die Verschlüsselung allgemein aktiviert wird, da die WKD-Funktion erst danach

verfügbar ist. Die App OpenKeyChain ist zwar keines der ausgewählten Produkte, dafür aber eine Voraussetzung, damit die Verschlüsselung mithilfe von OpenPGP-Schlüsseln in FairEmail und K9Mail funktioniert. Deshalb wurde für diese Anwendung eine Instruktion geschrieben und verfügbar gemacht, die erläutert, wie WKD aktiviert werden kann. Zuletzt wird auch die Aktivierung von WKD in KMail dargestellt. Wie die WKD-Funktion praktisch verwendet werden kann, wird nur für die Produkte Claws Mail, OpenKeyChain und Thunderbird beschrieben, da KMail und Mailvelope die Funktion automatisch ausführen.

Produkt	**Link zu den Anleitungen**
Claws Mail	https://wiki.gnupg.org/EMailClients/ClawsMail
KMail	https://wiki.gnupg.org/EMailClients/KMail
OpenKeyChain	https://wiki.gnupg.org/OpenKeyChain
Thunderbird	https://wiki.gnupg.org/EMailClients/Thunderbird

Tabelle 7: Links zu den erstellten Anleitungen

Implementierung in Mailvelope

Da die Browser-Erweiterung Mailvelope eine ältere Version des WKD-Standards enthielt, fehlte die Advanced Method in diesem Produkt zu Beginn der Arbeit. Deswegen wurde im Rahmen der Arbeit die Advanced Method implementiert. Die Änderungen, die am Quellcode durchgeführt wurden, befinden sich im Anhang und können auch unter https://github.com/mailvelope/mailvelope/pull/782/files gefunden werden.

Neben der Implementierung der Advanced Method ist es notwendig, dass ein Produkt korrekt zwischen Direct und Advanced Method entscheidet. Abbildung 74 zeigt ein Aktivitätsdiagramm, welches die Entscheidungen visualisiert, wie sie bei der Implementierung getroffen werden. Zu Beginn prüft der Browser, ob die API mit dem Namen `dns` vorhanden ist. Dieser Fall trifft zum Zeitpunkt der Arbeit nur auf den Browser Firefox zu. Wahrscheinlich ist dies auch bei Browsern der Fall, die von Firefox abstammen. Unter Firefox wird daraufhin zur Auflösung des DNS-Namens – zusammengesetzt aus der Domain der Mail-Adresse und der Subdomain *openpgpkey* – die Methode `browser.dns.resolve()` verwendet. Damit kann der Browser eindeutig feststellen, ob ein DNS-Name aufgelöst werden kann oder nicht. Wenn der Name aufgelöst werden kann, darf der Browser nur die Advanced Method ausführen. Ansonsten setzt er die Direct Method ein.

Unter Chrome gibt es eine vergleichbare Methode nicht. Aus diesem Grund probiert Chrome direkt die Advanced Method mit der Methode `fetch()`. Ruft der Browser die URL der Advanced Method auf und lädt erfolgreich einen Schlüssel herunter, ist der Vorgang der WKD-Funktion zu Ende. Bei einem Fehler muss der Browser jedoch eine weitere Entscheidung treffen. Lieferte die Methode `fetch()` einen Status-Code, wie z.B. 404, bedeutet dies, dass die Domain erreichbar ist, da der Server eine Antwort zurückschickt. Damit ist klar, dass der Browser die Direct Method nicht mehr ausführen darf. Erst wenn ein TypeError geworfen wird, verwendet der Browser zusätzlich die Direct Method, um einen öffentlichen Schlüssel zu erhalten. Der Grund für diese Vorgehensweise ist, dass ein TypeError keine Informationen über die Ursache des Fehlers bietet. Deswegen kann der Browser nicht unterscheiden, ob der DNS-Name aufgelöst werden kann oder nicht.

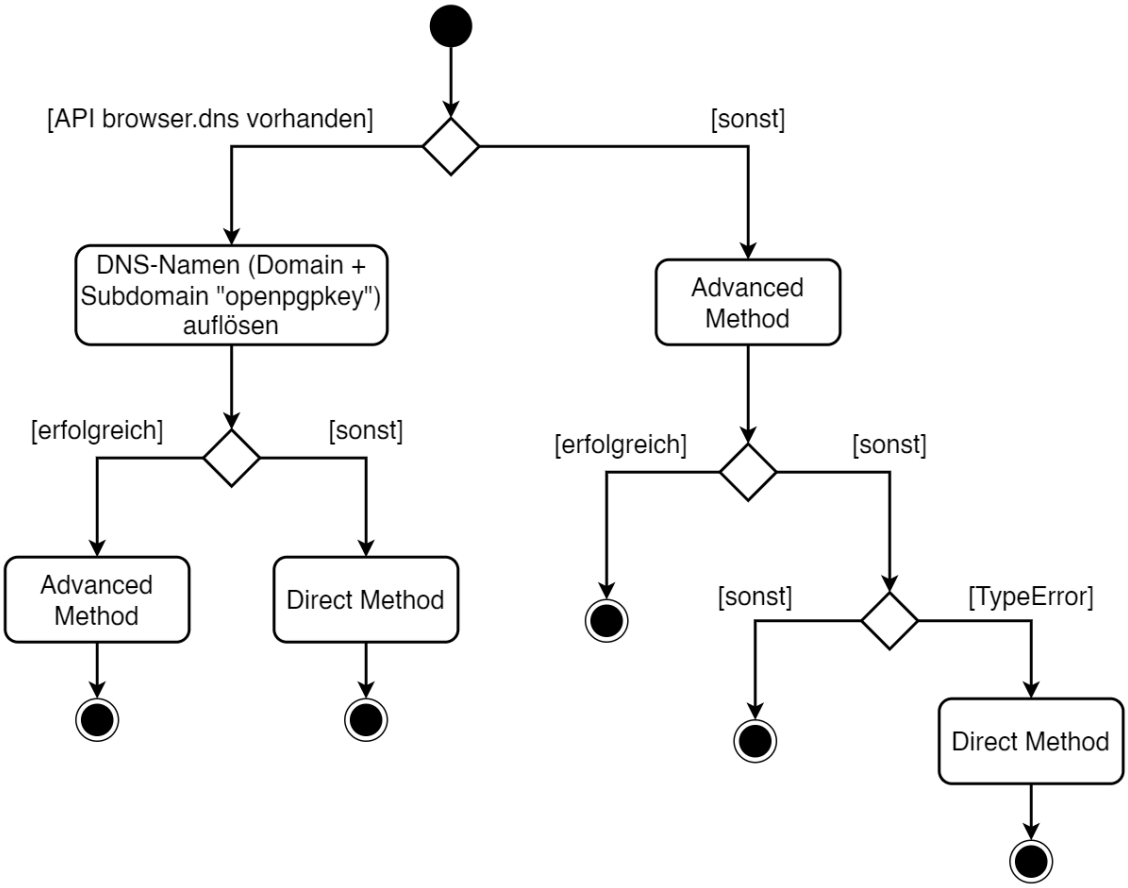


Abbildung 74: UML-Aktivitätsdiagramm für Mailvelope

Das Diagramm zeigt, welche Entscheidungen getroffen werden, um die passende Methode des WKD-Standards zu wählen.

6. Diskussion

Verfeinerung des WKD-Standards

In einem vorherigen Kapitel wird erklärt, dass und wie die WKD-Funktion in Mailvelope im Rahmen der Arbeit auf den Stand des aktuellen Entwurfs des WKD-Standards gebracht wurde. Bei der Umsetzung fällt auf, dass die Implementierung der Entscheidung, ob die Advanced Method ausgeführt werden soll, nicht so einfach ist. Im Standard ist folgender Abschnitt vorhanden:

> „There are two variants on how to form the request URI: The advanced and the direct method. Implementations MUST first try the advanced method. Only if the required sub-domain does not exist, they SHOULD fall back to the direct method.“ [9]

Berechtigterweise kann die Frage auftauchen, was es bedeutet, dass eine Subdomain nicht existiert. Bedeutet dies, dass es keinen DNS-Eintrag für die Subdomain gibt? Oder zählt der Fall dazu, in dem ein WKD-Server nicht erreichbar ist? In dieser Arbeit wurde bei der Implementierung für Mailvelope davon ausgegangen, dass versucht werden sollte, den DNS-Namen der Subdomain aufzulösen. Dies ist jedoch nicht bei allen Browsern möglich. Unter Firefox gibt es zum Auflösen eines DNS-Namen den Befehl `resolve()`, doch der Browser Chrome bietet eine solche Möglichkeit nicht. Bei der Implementierung wird die Domain in Chrome deshalb so getestet, dass direkt die Advanced Method ausgeführt wird. Dies geschieht, indem die URL mithilfe der Methode `fetch()` aufgerufen wird. Wird diese ausgeführt und das Ergebnis erhält einen Status-Code, wie z.B. 404, ist gewiss, dass es die Subdomain gibt, da der Server eine Antwort gibt. In dem Fall, dass ein `TypeError` auftaucht, ist jedoch nicht eindeutig, was die Ursache ist. Es kommt unter anderem in Frage, dass der Server in dem Moment nicht erreichbar ist, was aber unabhängig davon ist, ob die Subdomain existiert. Deshalb bleibt die Frage, was zu tun ist, wenn ein solcher Fall auftritt, in dem es nicht genügend Informationen gibt, um sagen zu können, ob die Subdomain existiert.

Im Standard sollte deswegen spezifiziert werden, was es bedeutet, wenn eine Subdomain nicht existiert. Hilfreich kann es auch sein, genau festzulegen, welche Schritte bei der Überprüfung durchgeführt werden sollen. Wird angegeben, dass der DNS-Name aufgelöst werden muss, um herauszufinden, ob die Subdomain existiert, ist eindeutig definiert, wie die Überprüfung stattfinden muss. Entwickler:innen müssen sich somit nicht mit dieser zusätzliche Frage beschäftigen, wenn sie die WKD-Funktion implementieren.

Zudem wird deutlich, dass nicht immer die Möglichkeit besteht, den DNS-Namen einer Domain zu überprüfen. Das liegt daran, dass die technischen

Möglichkeiten von der Plattform abhängen. Wenn zur Überprüfung der Subdomain also der DNS-Name aufgelöst werden soll, sollte ebenso erwähnt werden, was geschehen soll, wenn es diese Möglichkeit gar nicht gibt.

Wenn der WKD-Standard eindeutiger ist und erläutert, wie die Aussage überprüft werden kann, dass eine Subdomain existiert und auch Fälle behandelt werden, in denen eine Überprüfung technisch nicht möglich ist, wird die Implementierung des Standards insgesamt einfacher, da Entwickler:innen weniger Spielraum bei der Interpretation des Standards haben und sich weniger Gedanken um seine Umsetzung machen müssen. Dadurch sinkt der Aufwand bei der Entwicklung, während die Wahrscheinlichkeit steigt, dass der Standard häufiger umgesetzt und weiter verbreitet wird.

Benutzerfreundlichkeit von WKD in den Mail-Clients

Beim Testen der Produkte fällt auf, dass teilweise viele Schritte notwendig sind, um WKD einsetzen zu können. Für das Verwenden der WKD-Funktion in Claws Mail müssen zuerst die Plug-ins im Programm geladen und dann ein Privacy System ausgewählt werden. Dass ein Privacy System ausgewählt werden muss, kann ein zusätzliches Problem darstellen, da Einsteiger:innen dies wahrscheinlich nicht für selbstverständlich halten. Das Problem in KMail ist, dass die Option für die Aktivierung der WKD-Funktion keinen Rückschluss darauf zulässt, ob sie etwas mit WKD zu tun hat. In K9Mail ist es notwendig, dass Benutzer:innen den Client manuell mit OpenKeyChain verbinden müssen, obwohl FairEmail zeigt, dass dies auch automatisch geschehen kann. Viele Schritte zur Vorbereitung entsprechen jedoch nicht dem Ziel, den Anwender:innen möglichst viel Arbeit abzunehmen, damit sie sich auf ihre Aufgaben konzentrieren können.

Ein anderer Aspekt betrifft die Heuristiken „Visibility of system status“ und “Error prevention”. In den Produkten Claws Mail, FairEmail und Thunderbird wird nicht angezeigt, ob ein öffentlicher Schlüssel zu einer Mail-Adresse vorhanden ist. Damit ist der Status des Systems nicht sichtbar. Das führt dazu, dass Benutzer:innen nicht im Voraus wissen, ob die Mail erfolgreich verschlüsselt werden kann. Folglich müssen Anwender:innen zuerst versuchen, die Mail zu senden, um herauszufinden, ob die Verschlüsselung erfolgreich ist. Wenn sie nicht erfolgreich ist, bedeutet das aber, dass ein Fehler verursacht wurde. Dieses Verhalten entspricht allerdings nicht der Heuristik der Prävention von Fehlern entspricht. Hilfreich ist es, bereits vor dem Abschicken anzuzeigen, ob der Vorgang voraussichtlich erfolgreich sein wird. Noch hilfreicher ist es, anzuzeigen, ob einzelne öffentliche Schlüssel vorhanden sind, da Benutzer:innen direkt versuchen können, fehlende öffentliche Schlüssel zu bekommen. Wenn es nicht möglich ist, diese zu erhalten, kennen sie jedoch wenigstens sofort den Grund dafür,

dass eine Verschlüsselung nicht möglich ist. K9Mail, KMail und Mailvelope schneiden beim Darstellen des Systemstatus bei den ausgewählten Produkten besser ab, da sie anzeigen, ob öffentliche Schlüssel für die eingegebenen Adressen vorhanden sind. K9Mail zeigt zusätzlich an, ob die Verschlüsselung möglich ist oder nicht. Sicherlich ist nicht allen Benutzer:innen bewusst, dass ein öffentlicher Schlüssel für jede Mail-Adresse vorhanden sein muss, damit die Mail verschlüsselt versendet wird. Normalerweise verschicken Mail-Clients die Mails aber entweder an alle verschlüsselt oder gar nicht. Durch die zusätzliche Anzeige bei K9Mail ist es sehr wahrscheinlich, dass Anwender:innen sehr früh erkennen, wenn sie die Mail nicht verschlüsseln können.

Aufgrund der Tatsache, dass Mailvelope die WKD-Funktion anbietet, wird erwartet, dass diese Funktion überall dort verwendet wird, wo nach einem öffentlichen Schlüssel gesucht wird. Der Abschnitt, der die Ergebnisse des Tests von Mailvelope bespricht, erläutert auch, in welchen Teilen von Mailvelope nach öffentlichen Schlüsseln gesucht wird. Außerdem erwähnt er, dass an verschiedenen Stellen auch die WKD-Funktion verwendet wird. In einem Bereich, der ausschließlich dafür existiert, um öffentliche Schlüssel zu suchen, wird die Funktion jedoch nicht eingesetzt. Die Beschreibung in diesem Bereich behauptet zwar nicht das Gegenteil, doch kann nicht erwartet werden, dass User:innen diese lesen. Johnson schreibt dazu: „when people are navigating through software or Web sites, they don't scrutinize screens carefully and read every word. They scan quickly for relevant information“ [39]. Somit überlesen die Anwender:innen eventuell den Hinweis, dass öffentliche Schlüssel nur auf Schlüsselservern gesucht werden. Zudem wird WKD auch an anderen Stellen verwendet. Warum sollte hier eine Ausnahme gemacht werden? Dies entspricht nicht der konsistenten Gestaltung einer Anwendung, wie sie in der Heuristik “Consistency and standards” verlangt wird.

Konsistent ist Mailvelope nicht überall. Ein Vorbild ist es jedoch, wenn es um das Starten des Beziehens von öffentlichen Schlüsseln geht. Genau wie bei KMail werden die Schlüssel von der Browser-Erweiterung automatisch bezogen. Dadurch werden die Benutzer:innen sehr komfortabel mit neuen öffentlichen Schlüsseln ausgestattet. Anders sieht es z.B. bei Thunderbird aus, wo Benutzer:innen erst verschiedene Fenster öffnen müssen, bevor eine Schlüsselsuche stattfinden kann. Dies kann dazu führen, dass sie die Suchfunktion für Schlüssel gar nicht finden. Außerdem ist es ein hoher Aufwand zu diesem Fenster zu navigieren.

Beim Thema Usability bzw. User Experience geht es jedoch nicht nur um die Verwendung der Produkte. Auch die Installation einer Anwendung hat einen Einfluss auf die Erfahrungen der Benutzer:innen. Auch hier gibt es deutliche Unterschiede zwischen leicht installierbaren Produkten und solchen, die einen höheren Aufwand benötigen. Schon dieser Schritt kann zur der Entscheidung führen, ein Produkt nicht verwenden zu wollen. Eine beliebige Version kann bei allen ausgewählten Produkten auf eine einfache Weise

installiert werden. Der Weg führt bei Mailvelope über die offizielle Seite mit Erweiterungen für Firefox bzw. Chrome. FairEmail und K9Mail können Nutzer:innen über die Anwendungen Play Store oder F-Droid installieren. Über die Paketmanager verschiedener GNU/Linux-Distributionen gelangen sie an die Produkte Claws Mail, KMail und Thunderbird. Claws Mail und Thunderbird sind zudem auch für das Betriebssystem Windows verfügbar. In diesem Fall können Benutzer:innen die Homepage der Produkte besuchen, um die jeweilige Installationsdatei herunterzuladen. Wenn sie jedoch eine Version installieren möchten, die die WKD-Funktion anbietet, sieht es bei Claws Mail etwas anders aus. Linux Mint bietet zu Beginn der Arbeit die Version 3.17.5-2 im Paketmanager an. Diese Version unterstützt WKD jedoch nicht. Dadurch entsteht eine weitere Hürde für Benutzer:innen, die WKD verwenden möchten, da sie das Produkt bauen oder darauf warten müssen, dass eine aktuelle Version im Paketmanager veröffentlicht wird. Dass neue Versionen später im Paketmanager erscheinen, hat den Vorteil, dass diese Versionen bereits bewährt und intensiv genutzt wurden. Der Preis dafür ist, wie bereits geschildert, dass neue Funktionen, wie z.B. WKD, später zugänglich sind.

Zusammenfassend muss bei einem Blick auf die im Rahmen der Arbeit aufgestellten Kriterien der Schluss gezogen werden, dass noch viel Luft nach oben ist, bis Produkte den WKD-Standard so umsetzen, dass eine gute Usability geboten wird. In einigen Produkten ist es aufwändig, das Produkt auf den Einsatz der WKD-Funktion vorzubereiten. In vielen Produkten ist es aufwändig, die WKD-Funktion zu verwenden. Außerdem werden, falls überhaupt, nur wenige Informationen zur Vertrauenswürdigkeit der Schlüssel verraten.

Das Ziel bei der Entwicklung von WKD war, dass Benutzer:innen eine einfache Möglichkeit haben, an öffentliche Schlüssel zu kommen. Gleichzeitig bieten die Schlüssel, die sie durch dieses Verfahren erhalten, eine grundlegende Vertrauenswürdigkeit, da sich die Schlüssel auf den Servern der Mail-Provider befinden. Obwohl diese Schlüssel aufgrund ihrer Quelle vertrauenswürdiger sind, als Schlüssel, die von anderen Arten von Schlüsselservern heruntergeladen werden, nutzen die meisten getesteten Produkte dieses Wissen nicht, um den per WKD bezogenen Schlüsseln ein größeres Vertrauen zu bescheinigen und sie automatisch zu bevorzugen, wenn sonst keine besser geeigneten Schlüssel vorhanden sind.

Kompatibilität zum aktuellen Entwurf des WKD-Standards

Während der Arbeit wurden Aussagen über Produkte gefunden, in denen behauptet wird, dass diese Produkte den WKD-Standard unterstützen, z.B. in der Ankündigung der Version 3.18.0 bzw. 4.0.0 von Claws Mail [29] oder in einem Blogeintrag von Mailvelope [40]. Das Testen hat jedoch gezeigt, dass nicht immer beide Methoden des WKD-Standards zum Einsatz kommen. Während beide der eben genannten Produkte damit beworben werden, dass sie den WKD-Standard umsetzen, kann Mailvelope dieses Versprechen nicht erfüllen.

Aus diesem Grund sollte eine Grenze zwischen älteren Entwürfen des Standards, die nur die Direct Method enthalten, und neueren Entwürfen mit der Advanced Method gezogen werden. Diese bilden zwei verschiedene Generationen und sollten dementsprechend eigene Bezeichnungen erhalten, wie z.B. „WKD 1.0“ und „WKD 2.0“. Wenn Entwickler:innen von Produkten dann angeben, ob sie die ältere oder die neue Generation implementieren, hat dies zwei Vorteile: Zum Einen können Benutzer:innen direkt sehen, welche Generation von WKD ein Produkt beherrscht und damit auch, welche Methoden es kennt. Dann können sie Produkte bei ihrer Auswahl aussortieren, wenn diese ihre Anforderungen nicht erfüllen. Zum Anderen können Tests gezielter durchgeführt werden, da die Advanced Method in einigen Fällen gar nicht erst getestet werden müsste.

Zusätzlich haben die Tests herausgestellt, dass die meisten Produkte dem aktuellen Entwurf des WKD-Standards nicht entsprechen. Einerseits liegt dies daran, dass – wie eben beschrieben – nur eine der beiden Methoden des Standards eingesetzt wird. Andererseits gibt es auch Produkte, die die Direct Method auch dann ausführen, wenn die Subdomain *openpgpkey* existiert. Vermutlich führen diese Produkte zuerst die Advanced Method aus und lassen die Direct Method folgen, wenn die Advanced Method bekommt. Das mag auch einen positiven Effekt haben, da es passieren kann, dass Entwickler:innen die Schlüssel auf ihrem WKD-Server so ablegen, dass diese per Direct Method geholt werden können, aber gleichzeitig die Subdomain *openpgpkey* vorhanden ist, während ein TXT-Record fehlt. In diesem Fall würden Anwender:innen trotz dieses Fehlers an die Schlüssel gelangen. Allerdings ist es wahrscheinlich, dass diese Fälle eine Ausnahme sind und es kann angenommen werden, dass die meisten Betreiber:innen ihre Server so vorbereitet haben, wie es durch den WKD-Standard beschrieben wird. In diesem Fall würden Produkte, die nicht kompatibel zum Standard sind, bei jeder Schlüsselsuche eine doppelte Anfrage stattfinden lassen, wenn die Advanced Method nicht erfolgreich war. Dabei wäre die zweite Anfrage jedes Mal überflüssig. Überflüssige Anfragen können jedoch zum Sicherheitsrisiko werden, da jede Anfrage eine Information bietet, die eventuell ausgenutzt werden kann. Auch aus Gründen der Nachhaltigkeit sind überflüssige Anfragen zu vermeiden, da diese den Datenverkehr erhöhen.

Auswirkungen der verschiedenen Maßnahmen

Um die Wirkung der Maßnahmen dieser Arbeit präzise messen zu können, gäbe es zwei Möglichkeiten. Auf der einen Seite könnte die Protokollierung der Interaktion von Benutzer:innen mit den Produkten festhalten, wie oft die WKD-Funktion genutzt wird. Bei Freier Software ist ein solches Vorgehen allerdings unüblich. Welche Auswirkungen es hat, wenn Produkte dennoch das Verhalten von Anwender:innen aufzeichnen, zeigt der Fall der Software Audacity zum Bearbeiten von Audiodateien. Wie Stenner [41] berichtet, hat ein Unternehmen die Anwendung gekauft und die Datenschutzbestimmungen aktualisiert. Das ermöglichte es, dass die Anwendung Daten sammeln und an Dritte weiterleiten durfte. Im Artikel wird bereits angedeutet, dass Entwickler:innen aus Unzufriedenheit über diesen Umstand einen Fork der Software erstellen wollten. Mindestens ein Fork mit dem Namen Tenacity (https://tenacityaudio.org/) ist seitdem entstanden. Dieses Beispiel ist zwar überspitzt, da die Daten sogar an Personen außerhalb des Unternehmens hinter Audacity weitergegeben werden sollten. Dennoch zeigt es die Tendenz, dass die Gemeinschaft hinter Freier Software empfindlich reagiert, wenn Produkte ihre Daten sammeln.

Die zweite Möglichkeit zur Messung ist, die Aktivität auf den WKD-Servern zu beobachten. Wenn die angewandten Maßnahmen dazu führen, dass mehr Aufrufe durch die WKD-Funktion ausgeführt werden, können sie als wirksam betrachtet werden. Dabei ist es notwendig, dass diese WKD-Server bereits intensiv genutzt werden, damit eine Veränderung durch die Maßnahmen sichtbar wird. Deshalb hat Bernhard Reiter einen Aufruf in einer Mailing-Liste veröffentlicht, um Personen zu erreichen, die WKD-Server betreiben [42]. Außerdem hat er das Team von ProtonMail per Mail kontaktiert und um Unterstützung gebeten, da diese ebenfalls einen WKD-Server betreiben und viele Benutzer:innen haben dürften. Leider waren die Bemühungen fruchtlos, sodass diese Art der Messung nicht durchgeführt werden konnte.

Für einige Maßnahmen kann trotz dieser Schwierigkeiten zumindest die potentielle Wirkung beschrieben werden. Der Abschnitt über die Auswahl der Produkte hält fest, dass die Browser-Erweiterung Mailvelope mehr als 43.000 Benutzer:innen hat. Sobald die Implementierung, die im Rahmen der Arbeit fertiggestellt wurde, in das Produkt integriert wird, haben diese Benutzer:innen die Möglichkeit, Schlüssel zusätzlich mithilfe der Advanced Method zu beziehen. Damit steigt die Wahrscheinlichkeit, dass sie überhaupt einen Schlüssel bekommen und damit auch der Einsatz einer Verschlüsselung.

Der Test des WKD-Servers von mail.de mithilfe von GnuPG hat herausgefunden, dass der Server nicht kompatibel zum aktuellen Entwurf des WKD-Standards war. Daraufhin hat der Entwickler Michael Kliewe einen TXT-Record angelegt, sodass die Kompatibilität wieder gegeben war. In einer Mail in der persönlichen Kommunikation am 06.12.2021 gab er die Information weiter, dass es bei mail.de 165 Accounts gibt, die für 171 Mail-Adressen einen Schlüssel auf dem WKD-Server gespeichert haben. Durch

diese Maßnahme sind also 171 Schlüssel auch durch Produkte erreichbar, die kompatibel zur aktuellen Version des WKD-Standards sind. Außerdem erklärt der Entwickler, dass mail.de u.a. noch einen HKP-Server besitzt, auf dem Schlüssel von 598 Mail-Adressen hinterlegt sind. Dieser Server ist bereits seit 2016 in Betrieb, während der WKD-Server zwei bis drei Jahre später an den Start ging. Diese Information legt die Vermutung nahe, dass sich die Zahl der Schlüssel auf dem WKD-Server in wenigen Jahren auch noch verdoppeln oder sogar verdreifachen wird.

Erfahrungen mit der Community

Für einen Teil der Maßnahmen fand die Kommunikation mit den Gemeinschaften hinter den Produkten statt. Dabei reagierten die Personen in den Gemeinschaften auf Nachrichten meist zügig, während die Art der Reaktionen stärker variierte. Teilweise wurden Diskussionen um Vorschläge für die Produkte geführt, die im Sande verliefen und zu keinem Ergebnis führten. In einem anderen Fall wurden Vorschläge positiv aufgenommen und einige Änderungen innerhalb sehr kurzer Zeit durchgeführt.

Im Repository von Mailvelope gab es bereits zu Beginn der Arbeit einen Fall, der das Fehlen der Advanced Method betraf. Aufgrund dessen und nach einem kurzen Austausch mit dem Hauptentwickler war die Bahn frei für eine Implementierung, um die Methode im Rahmen der Arbeit konform zum WKD-Standard zu integrieren. Als die Implementierung fertiggestellt war, wurde sie als Pull Request eingereicht und nach einer Überprüfung durch den Hauptentwickler verbessert. Danach gab es nach über zwei Monaten keine Rückmeldung und der Code wurde noch nicht übernommen.

Die Erfahrungen zeigen, dass Personen nicht erwarten dürfen, dass sie innerhalb weniger Monate große Veränderungen an Freier Software durchgeführen können. An einigen Stellen ist es vorher notwendig, dass mehr Aufwand betrieben werden muss, damit die Community soweit ist, eine Änderung an „ihrem“ Produkt zu akzeptieren. Außerdem kann es viel Zeit in Anspruch nehmen, bis eine Änderung in das Produkt übernommen wird. Am Ende gibt es keine Garantie dafür, dass die Implementierung überhaupt oder rechtzeitig integriert wird, damit die Konsequenzen der Änderungen beobachtet werden können. Dies sollte für folgende Arbeiten berücksichtigt werden.

Auswahl und Verfahren beim Testen

Insgesamt beinhaltete das Testen acht verschiedene Produkte. Sechs davon wurden bereits vorher festgelegt, doch die Anwendung OpenKeyChain ist eine Voraussetzung für die Verschlüsselung in zwei anderen Produkten. Somit musste auch diese App überprüft werden. Eher zufällig kam die

Erkenntnis, dass mail.de auch einen WKD-Server anbietet. Da das Testen eines WKD-Servers mithilfe von GnuPG relativ einfach ist, wurde dies auch getan. Die Auswahl der verschiedenen Produkte bietet einen guten Überblick über den Fortschritt des WKD-Standards auf verschiedenen Plattformen.

Die Tests besaßen einen iterativen Charakter. Zuerst sollte nur herausgefunden werden, ob die Produkte kompatibel zum aktuellen Entwurf des WKD-Standards sind. Dabei wurde sichtbar, dass das Einrichten der WKD-Funktion und die letztendliche Nutzung teilweise einen größeren Aufwand erfordern, als wünschenswert wäre. Darum wurden zwei Anwendungsfälle und darauf aufbauend verschiedene Kriterien erstellt. Diese sollen dafür sorgen, dass Produkte eine bessere Usability bei der Verwendung von WKD bieten. Daraufhin wurden die Produkte erneut und intensiver getestet, die Kriterien weiter verfeinert und die Produkte wieder getestet. Abbildung 75 veranschaulicht diesen iterativen Prozess.

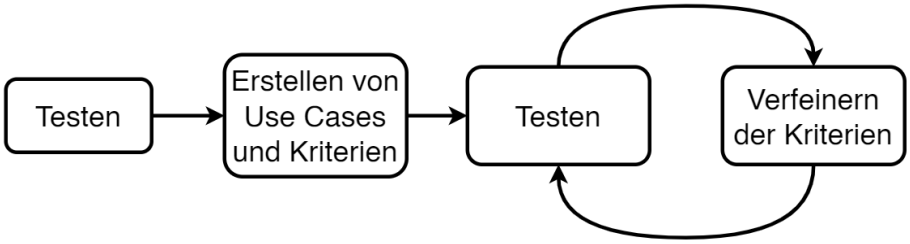


Abbildung 75: Ablauf beim Testen

Da die Auswahl verschiedene Plattformen abdeckt, war sie eine gute Entscheidung. Eine Erkenntnis aus dem Testen ist jedoch, dass es effizienter wäre, zuerst ein einzelnes Produkt zu testen, um den Stand des Produkts festzuhalten. Nachdem dann Kriterien aufgestellt und dieses Produkt weiter getestet wurde, können die Kriterien so verfeinert werden, dass sie präzise sind und die Anforderungen abdecken, die z.B. durch einen Anwendungsfall deutlich werden. Die Arbeit hat gezeigt, dass es notwendig ist, die Kriterien anzuwenden, um zu erkennen, wie ausgereift sie sind. Erst wenn die Kriterien zufriedenstellend sind, sollten weitere Produkte mit diesen Kriterien überprüft werden. Auf diese Weise ist es nicht notwendig, alle Produkte dem oben erläuterten iterativen Prozess zu unterwerfen.

Fazit

Die vorliegende Arbeit hat sich mit der Frage beschäftigt, wie der WKD-Standard weiter verbreitet werden kann. Die Tests haben Aufschluss darüber gegeben, wie der Fortschritt von WKD in verschiedenen Produkten ist. Das allein ist eine wichtige Erkenntnis, da jetzt an den passenden Stellen angesetzt werden kann, um den Standard weiter zu verbreiten. Außerdem ist dabei deutlich geworden, dass es wichtig ist, die Usability von WKD zu

verbessern, denn in allen getesteten Produkten gab es das Potential zur Verbesserung. Sobald dieses Potential ausgenutzt und damit die Usability gesteigert wird, steigt die Wahrscheinlichkeit, dass Benutzer:innen die WKD-Funktion einsetzen und auch anderen Personen empfehlen, das Produkt bzw. die Funktion zu verwenden. Somit wird auch die Zahl der Teilnehmer:innen bei der verschlüsselten Kommunikation größer. Letztendlich kommen direkte Netzwerkeffekte zur Geltung, die bereits in der Einleitung dieser Arbeit erläutert werden. Diese tragen ihren Teil zu der Verbreitung von WKD bei.

Eine weitere wichtige Erkenntnis ist, dass der Standard mehr Präzision benötigt und Fälle behandeln sollte, in denen fehlende technische Möglichkeiten verhindern, dass Entwickler:innen den Standard genauso umsetzen, wie die Intention hinter dem Standard ist. Dafür ist es bedeutend, dass die Verfasser:innen von Standards berücksichtigen, dass es verschiedene Plattformen gibt, auf denen Produkte angeboten werden. Neben dem Betriebssystem kann auch die Hardware eine entscheidende Rolle spielen. Zuletzt gibt es auch mehrere Arten von Produkten, z.B. installierbare Anwendungen und Erweiterungen. Diese Faktoren haben einen Einfluss darauf, welche technischen Möglichkeiten gegeben sind. Deswegen ist es empfehlenswert, Implementierungen durchzuführen, um den Einfluss dieser Faktoren überhaupt zu erfassen.

Wenn die eben genannten Aspekte beim Schreiben eines Standards berücksichtigt werden, treten weniger Fälle auf, in denen Unklarheiten bei der Umsetzung existieren. Folglich müssen sich die Entwickler:innen eines Produkts weniger Gedanken um die Anwendung des Standards machen. Somit besteht insgesamt weniger Aufwand bei der Implementierung. Dadurch sinkt die Hürde, diesen Standard in ein Produkt zu integrieren, was wiederum die Wahrscheinlichkeit erhöht, dass er in weiteren Produkten implementiert wird. Sobald die Anzahl der Produkte steigt, die diesen Standard anbieten, wächst außerdem der Druck auf andere Entwickler:innen, da er im Laufe der Zeit immer wichtiger wird und von Benutzer:innen als vorausgesetzt betrachtet wird. Genau dieses Ziel sollte mit dem WKD-Standard erreicht werden.

Solange der WKD-Standard bei der Entscheidung zwischen Direct Method und Advanced Method nicht präzise formuliert ist, können Entwickler:innen, die vor demselben Problem stehen und keine Möglichkeit zum Auflösen eines DNS-Namens haben, sich an der Implementierung, die im Rahmen der Arbeit für Mailvelope durchgeführt wurde, orientieren. Das fünfte Kapitel beschreibt die Implementierung und auch, wie die Implementierung vorgeht, wenn die Auflösung eines DNS-Namen nicht möglich ist.

Im Hinblick auf das Thema Standards ist während der Tests aber auch aufgefallen, dass ein Standard auch festlegen muss, was bei der Gestaltung der grafischen Oberfläche einer Anwendung, die Benutzer:innen zu Gesicht bekommen, beachtet werden soll. Wenn ein Standard keine Vorgaben dazu macht, kann nicht garantiert werden, dass Produkte eine gute User

Experience offerieren, auch wenn das die Intention des Standards ist. Die Tests dieser Arbeit beweisen, dass die Gestaltung der grafischen Oberfläche, die einen Standard betreffen, völlig unterschiedlich aussehen kann, obwohl derselbe Standard umgesetzt wurde. Dabei kann es zum Beispiel auch sein, dass einige Produkte mehr Informationen zum Systemstatus anzeigen als andere. Schon allein das verursacht Unterschiede bei der Usability. Wenn also Produkte, die einen Standard umsetzen, dabei auch ein bestimmtes Level bei der Usability erreichen sollen, ist es notwendig, dass auch für die grafische Oberfläche bestimmte Spezifikationen gemacht werden.

7. Ausblick

Verbesserung von Usability und User Experience

Das Kapitel 3 erläutert verschiedene Kriterien, deren Berücksichtigung zu einer Implementierung führt, die WKD auf eine benutzerfreundliche Art anbietet. Diese Kriterien dienen als Unterstützung von Entwickler:innen, damit diese ihre Produkte verbessern können. Außerdem sind sie eine Hilfe für diejenigen Entwickler:innen, die WKD zum ersten Mal in ihrem Produkt anbieten möchten. Als Nächstes müssen die Kriterien an verschiedene Entwickler:innen herangetragen werden, um mit ihnen darüber zu diskutieren, damit die Kriterien zum Einsatz kommen. Wenn Personen Gegenargumente nennen, müssen diese analysiert und gegebenenfalls Anpassungen an den Kriterien vorgenommen werden.

Die Kriterien sind jedoch nur eine Grundlage, um die Usability von WKD in Produkten zu verbessern. Um weitere Verbesserungsmöglichkeiten herauszufinden, sollten verschiedene Produkte, die die Kriterien erfüllen, in Usability-Tests untersucht werden. Es ist wichtig, zu erfahren, welche Schwierigkeiten Anwender:innen bei der Bedienung der Produkte haben. Zudem spielt es eine große Rolle, wie Benutzer:innen die Elemente interpretieren, die in der grafischen Oberfläche sichtbar sind, besonders diejenigen, die für die Anzeige von öffentlichen Schlüsseln und deren Vertrauenswürdigkeit verantwortlich sind. Auf diese Weise können weitere Aspekte entdeckt werden, die bei der Implementierung von WKD berücksichtigt werden sollten.

Diese Schritte führen näher zu dem Ziel von WKD, die User Experience bei der Verschlüsselung zu verbessern. WKD betrifft allerdings nur das Beziehen von öffentlichen Schlüsseln und dies ist nur ein Schritt bei der Verschlüsselung einer Mail. Somit gibt es weitere Aspekte der Verschlüsselung, deren Verbesserungspotential analysiert werden kann, z.B. das Erstellen eines Schlüsselpaars. Danach kann auch für diese Aspekte ein Standard oder Ähnliches entwickelt werden, damit Entwickler:innen sie benutzerfreundlich umsetzen können.

Ein Aspekt, an den viele Personen wahrscheinlich nicht denken, wenn sie von Usability sprechen, ist die Installation eines Produkts. Wie das Kapitel zur Diskussion erläutert, ist das Installieren der aktuellen Version von Claws Mail nicht bei jedem Betriebssystem ein einfacher Vorgang. Bei einigen Systemen wird im Paketmanager keine aktuelle Version der Anwendung angeboten. Das Bauen von Claws Mail kann durchschnittlichen Personen nicht zugemutet werden. Dies ist eindeutig keine Möglichkeit, das Produkt benutzerfreundlich anzubieten. Aus diesem Grund wird empfohlen, mit den verantwortlichen Personen hinter den im Paketmanager angebotenen Produkten zu diskutieren, um anzuregen, eine aktuelle Version zur Verfügung zu stellen. Ein anderer Weg ist das Anbieten eines anderen Formats.

Ein Format, das von unterschiedlichen GNU/Linux-Distributionen unterstützt wird, ist zum Beispiel AppImage (https://appimage.org). Ein weiterer Weg ist die Distribution über Plattformen wie Snap Store (https://snapcraft.io/snap-store) oder Flatpak (https://www.flatpak.org). Diese beiden Plattformen werden ebenfalls von unterschiedlichen Distributionen unterstützt. Auf der Homepage von Flatpak ist davon die Rede, dass sogar 33 verschiedene Distributionen unterstützt werden. Auf diese Weise sparen sich die Entwickler:innen die Arbeit, für jede Distribution eine eigene Version anzubieten. Gleichzeitig ist die Installation für Benutzer:innen relativ einfach.

Bisher sind in dieser Arbeit schon häufiger die Begriffe Usability und User Experience gefallen. Dabei geht User Experience über das hinaus, was die Usability beinhaltet. Wenn eine Anwendung eine gute User Experience bietet, heißt das, dass Benutzer:innen nicht nur effizient ihre Aufgaben bewältigen können und die Software für sie leicht zu bedienen ist, sondern, dass sie dabei Freude empfinden und die Anwendung gerne erneut verwenden. Ein Mittel zur Steigerung der User Experience ist die sogenannte „Gamification“. Dabei werden Spielmechaniken in Produkten verwendet, die ansonsten nichts mit Spielen zu tun haben. Damit wird die Freude bei der Bedienung verstärkt und die Motivation, ein Produkt einzusetzen, gesteigert [43]. Die Gamification kann dazu führen, dass die Wahrscheinlichkeit steigt, mit der die Produkte weiterempfohlen werden. Deshalb sollte untersucht werden, welche Spielmechaniken für Produkte geeignet sind, die eine Verschlüsselung ermöglichen und welche Auswirkungen sie haben.

Weitere mögliche Schritte

Diese Arbeit gibt einen Überblick über den Stand der Implementierung von WKD in einigen Produkten verschiedener Plattformen. Diese Auswahl an Produkten ist jedoch nur ein Bruchteil der Produkte, die die Verschlüsselung von Mails anbieten. Um Benutzer:innen einen vollständigeren Überblick über Produkte und deren Kompatibilität anbieten zu können, müssen Tests weiterer Produkte durchgeführt werden. Beispielhaft sei hier der Mail-Client Outlook erwähnt. Dieser hat mithilfe der Erweiterung GpgOL die Fähigkeit, Mails zu verschlüsseln und WKD zu verwenden. Die Erweiterung wird zusammen mit der Freien Software Gpg4win installiert. Je mehr Produkte getestet werden, desto mehr Produkte deckt eine Übersicht ab und desto besser können sich Benutzer:innen daran orientieren, wenn sie auf der Suche nach einem Produkt sind, bei dem sie WKD voraussetzen.

Für die beiden Apps FairEmail und K9Mail ist die parallele Installation von OpenKeyChain notwendig, damit die Verschlüsselung funktioniert. Diese Anwendung sollte untersucht werden, um festzustellen, ob sie überhaupt das Potential hat, die Kriterien aus Kapitel 3 zu erfüllen. Sollte dies nicht der

Fall sein, ist es wichtig, dass notwendige Funktionen entwickelt und ergänzt werden. Es könnte aber auch über die Entwicklung einer neuen Bibliothek nachgedacht werden, die direkt in die Produkte integriert werden kann. OpenKeyChain bietet zwar bereits viele Möglichkeiten an und verschiedene Apps können auf OpenKeyChain zugreifen. Der Nachteil dieser App ist jedoch, dass sie zusätzlich installiert werden muss. Auch wenn die App dadurch den Vorteil besitzt, dass Schlüssel zentral verwaltet werden, bedeutet ihre Installation zusätzlichen Aufwand. Außerdem ist die Wahrscheinlichkeit bei durchschnittlichen Benutzer:innen gering, dass es notwendig ist, dass mehrere Anwendungen auf die Schlüssel zugreifen müssen. Eine Bibliothek, die direkt mit den Produkten ausgeliefert wird, bedeutet weniger Aufwand, den Benutzer:innen betreiben müssen.

Anhang

Anwendungsfälle

Anwendungsfall AF1	**Verschicken einer verschlüsselten Mail**
Ziel	Eine vertrauliche Nachricht verschicken
Vorbedingungen	• Ein Mail-Client wurde installiert, der die Verschlüsselung nach OpenPGP ermöglicht • Die sendende Person besitzt den öffentlichen Schlüssel der erhaltenden Person nicht • Die erhaltende Person hat einen öffentlichen Schlüssel erstellt und ihn auf einen WKD-Server hochgeladen • Der öffentliche Schlüssel der erhaltenden Person muss automatisch bezogen werden
Nachbedingung Erfolg	Eine verschlüsselte Mail wurde verschickt.
Nachbedingung Fehlschlag	• Eine unverschlüsselte Mail wurde verschickt • Gar keine Mail wurde verschickt • Die erhaltende Person kann die verschlüsselte Mail nicht entschlüsseln
Akteure	Beliebige Person
Auslösendes Ereignis	Eine Person möchte einer anderen Person eine vertrauliche Nachricht zukommen lassen, die sonst keiner lesen können soll.
Beschreibung	1. Person öffnet den Mail-Client 2. Person öffnet den Bereich zum Verfassen einer Mail 3. Person gibt Mail-Adresse, Betreff und Nachricht ein 4. Person verschickt die Mail
Erweiterungen	• Person überprüft, ob der öffentliche Schlüssel vertrauenswürdig ist • Person überprüft, wie vertrauenswürdig der öffentliche Schlüssel ist

Anwendungsfall AF2	**Überprüfen der Signatur einer Mail**
Ziel	Überprüfen, ob eine Mail wirklich von der Person geschrieben wurde, deren Mail-Adresse zum Versenden genutzt wurde
Vorbedingungen	• Ein Mail-Client wurde installiert, der die Verschlüsselung nach OpenPGP ermöglicht • Die erhaltende Person besitzt den öffentlichen Schlüssel der sendenden Person noch nicht • Die Person, die die Mail verschickt hat, hat einen öffentlichen Schlüssel erstellt und ihn auf einem WKD-Server verfügbar gemacht • Der öffentliche Schlüssel wird automatisch bezogen, der zu der Mail-Adresse der erhaltenen Mail gehört.
Nachbedingung Erfolg	Die Signatur der erhaltenen Mail konnte überprüft werden.
Nachbedingung Fehlschlag	Die Signatur der erhaltenen Mail konnte nicht überprüft werden.
Akteure	Beliebige Person
Auslösendes Ereignis	Eine Person bekommt eine Mail mit wichtigen Informationen und möchte wissen, ob diese Mail wirklich von der Person geschrieben wurde, deren Mail-Adresse zum Versenden genutzt wurde.
Beschreibung	1. Person öffnet den Mail-Client. 2. Person wechselt in den Bereich mit den erhaltenen Mails. 3. Person öffnet eine signierte Mail. 4. Person startet eine Signaturüberprüfung. 5. Person interpretiert das Ergebnis der Signaturüberprüfung
Erweiterungen	• Person überprüft, ob der öffentliche Schlüssel vertrauenswürdig ist • Person überprüft, wie vertrauenswürdig der öffentliche Schlüssel ist

Änderungen am Quellcode von Mailvelope

Die Änderungen sind auch hier zu finden: https://github.com/mailvelope/mailvelope/pull/782/files

1 ■□□□□ src/firefox/manifest.json

```
@@ -33,6 +33,7 @@
  },
  "permissions": [
    "*://*/*",
+   "dns",
    "nativeMessaging",
    "storage",
    "tabs",
```

Änderungen an der Datei im Verzeichnis src/firefox/manifest.json des Github-Projekts Mailvelope

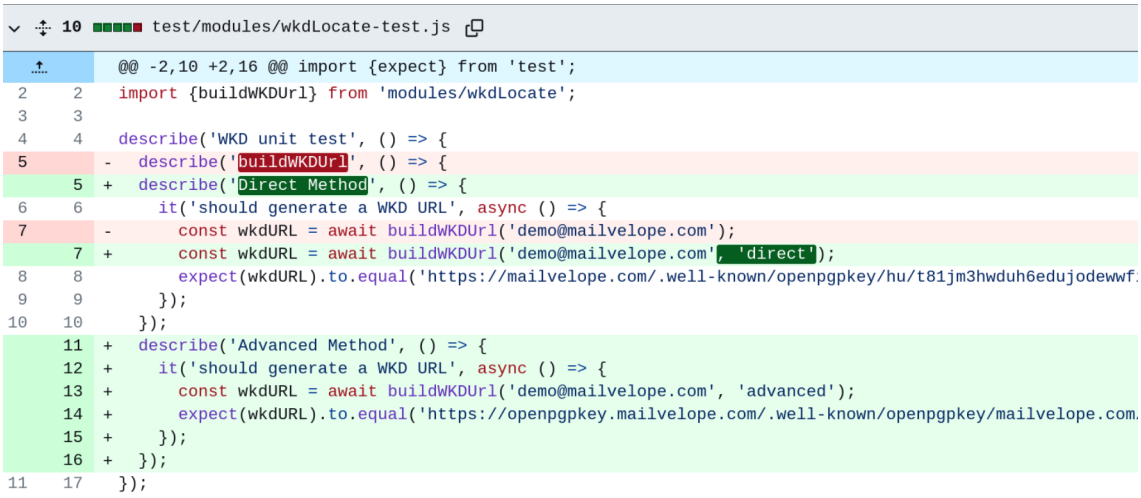
67 ■■■■□ src/modules/wkdLocate.js

```
@@ -12,6 +12,7 @@ import {str2ab} from '../lib/util';
import {prefs} from './prefs';
import {filterUserIdsByEmail} from './key';
import defaults from '../res/defaults.json';
+import browser from 'webextension-polyfill';

export const name = 'WKD';

@@ -61,24 +62,61 @@ export async function lookup(email) {
    return;
  }

- const url = await buildWKDUrl(email);
+ let data;

- // Impose a size limit and timeout similar to that of gnupg.
- const data = await timeout(TIMEOUT * 1000, window.fetch(url)).then(
-   res => sizeLimitResponse(res, SIZE_LIMIT * 1024));
+ /** For the WKD standard (draft version 13, https://datatracker.ietf.org/doc/draft-koch-openp
+  * case we should use the advanced method. The optimal way to do this check, is to try to
+  * resolve the DNS name. On the 21st of November 2021 only Firefox (since version 60) supporte
+  * (https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/API/dns/resolve).
+  * That's why we check first, if this method exists.
+  */
+ if (typeof browser.dns !== 'undefined') {
+   const host = `openpgpkey.${domain}`;
+   const canDNSNameBeResolved = await browser.dns.resolve(host).then(result => {
+     if (result.addresses.length > 0) {
+       return true;
+     }
+     return false;
+   }).catch(() => false);

+   if (canDNSNameBeResolved) {
+     data = await retrievePubKeyViaWKD(email, 'advanced');
+   } else {
+     data = await retrievePubKeyViaWKD(email, 'direct');
+   }
+ } else {
+   try {
+     data = await retrievePubKeyViaWKD(email, 'advanced');
+   } catch (error) {
```

Änderungen an der Datei im Verzeichnis src/modules/wkdLocate.js des Github-Projekts Mailvelope

```
     91  +       /** On Chrome we can't resolve the DNS name. But, if the server returns some status code
     92  +        * like 404, we know that the DNS name can be resolved. If we get a TypeError,
     93  +        * we don't know the exact reason, so we also try the direct method.
     94  +        */
     95  +       if (error.name.toString() === 'TypeError') {
     96  +         data = await retrievePubKeyViaWKD(email, 'direct');
     97  +       } else {
     98  +         return;
     99  +       }
    100  +     }
    101  +   }
    102  +   // If we got nothing the get error was already logged and
    103  +   // we do not need to throw another error. TH
 70 104     if (!data) {
 71     -     // If we got nothing the get error was already logged and
 72     -     // we do not need to throw another error. TH
 73 105       return;
 74 106     }
 75     -
 76 107     // Now we should have binary keys in the response.
 77 108     const armored = await parseKeysForEMail(data, email);
 78 109
 79 110     return {armored, date: new Date()};
 80 111   }
 81 112
    113  + async function retrievePubKeyViaWKD(email, methodOfWKD) {
    114  +   const url = await buildWKDUrl(email, methodOfWKD);
    115  +   // Impose a size limit and timeout similar to that of gnupg.
    116  +   return timeout(TIMEOUT * 1000, window.fetch(url)).then(
    117  +     res => sizeLimitResponse(res, SIZE_LIMIT * 1024));
    118  + }
    119  +
 82 120   /**
 83 121    * Check if a domain is blacklisted.
 84 122    *
        @@ -107,19 +145,26 @@ function isBlacklisted(domain) {
107 145    * under the terms of the GNU Lesser General Public License Version 3
108 146    *
109 147    * @param {String}   email  The canonicalized RFC822 addr spec.
    148  + * @param {String} methodOfWKD   Determines the WKD method, which has to be used. Possible parame
110 149    *
111     -  * @returns {String} The WKD URL according to draft-koch-openpgp-webkey-service-06.
    150  +  * @returns {String} The WKD URL according to draft-koch-openpgp-webkey-service-13 (https://datat
112 151    */
113     - export async function buildWKDUrl(email) {
    152  + export async function buildWKDUrl(email, methodOfWKD) {
114 153     const [, localPart, domain] = /(.*)@(.*)/.exec(email);
115 154     if (!localPart || !domain) {
116     -     throw new Error(`WKD: failed to parse: ${email}`);
    155  +     throw new Error(`WKD failed to parse: ${email}`);
117 156     }
118 157     const localPartBuffer = str2ab(localPart.toLowerCase());
119 158     const digest = await crypto.subtle.digest('SHA-1', localPartBuffer);
120 159     const localEncoded = openpgp.util.encodeZBase32(new Uint8Array(digest));
121 160     const localPartEncoded = encodeURIComponent(localPart);
122     -   return `https://${domain}/.well-known/openpgpkey/hu/${localEncoded}?l=${localPartEncoded}`;
    161  +   // Create URL with Advanced Method
    162  +   if (methodOfWKD === 'advanced') {
    163  +     return `https://openpgpkey.${domain}/.well-known/openpgpkey/${domain}/hu/${localEncoded}?l=${local
    164  +   // Create URL with Direct Method
    165  +   } else {
    166  +     return `https://${domain}/.well-known/openpgpkey/hu/${localEncoded}?l=${localPartEncoded}`;
    167  +   }
123 168   }
124 169
125 170   /** Convert a promise into a promise with a timeout.
```

Änderungen an der Datei im Verzeichnis src/modules/wkdLocate.js des Github-Projekts Mailvelope (Fortsetzung)



```
10 ■■■■■ test/modules/wkdLocate-test.js
@@ -2,10 +2,16 @@ import {expect} from 'test';
2  2  import {buildWKDUrl} from 'modules/wkdLocate';
3  3
4  4  describe('WKD unit test', () => {
5     -   describe('buildWKDUrl', () => {
      5  +   describe('Direct Method', () => {
6  6       it('should generate a WKD URL', async () => {
7     -       const wkdURL = await buildWKDUrl('demo@mailvelope.com');
      7  +       const wkdURL = await buildWKDUrl('demo@mailvelope.com', 'direct');
8  8         expect(wkdURL).to.equal('https://mailvelope.com/.well-known/openpgpkey/hu/t81jm3hwduh6edujodewwf
9  9       });
10 10    });
      11 +   describe('Advanced Method', () => {
      12 +     it('should generate a WKD URL', async () => {
      13 +       const wkdURL = await buildWKDUrl('demo@mailvelope.com', 'advanced');
      14 +       expect(wkdURL).to.equal('https://openpgpkey.mailvelope.com/.well-known/openpgpkey/mailvelope.com
      15 +     });
      16 +   });
11 17  });
```

Änderungen an der Datei im Verzeichnis test/modules/wkdLocate-test.js des Github-Projekts Mailvelope

Literaturverzeichnis

[1] P. Zimmermann. „Creator of PGP.“ philzimmermann.com. https://philzimmermann.com/EN/background/index.html (aufgerufen 12.02.2022).

[2] P. Zimmermann. „Where to Get PGP.“ philzimmermann.com. https://philzimmermann.com/EN/findpgp/ (aufgerufen 05.09.2021).

[3] D. Riley. „Symantec is now NortonLifeLock as Broadcom closes purchase of its enterprise business.“ SiliconANGLE. https://siliconangle.com. https://siliconangle.com/2019/11/05/symantec-now-nortonlifelock-broadcom-completes-acquisition-enterprise-business/ (aufgerufen 20.09.2021).

[4] The GnuPG Project. „The GNU Privacy Guard.“ gnupg.org. https://gnupg.org/ (aufgerufen 05.09.2021).

[5] J. Callas, l. Donnerhacke, H. Finney, D. Shaw und R. Thayer. „OpenPGP Message Format.“ IETF. https://datatracker.ietf.org/doc/html/rfc4880 (aufgerufen 07.02.2022).

[6] C. Friemel. „Interesse an verschlüsselten Mails steigt.“ GMX Newsroom. https://newsroom.gmx.net/2018/08/24/interesse-an-verschluesselten-mails-steigt/ (aufgerufen 06.09.2021).

[7] B. Reiter. „Contract ‘EasyGpg’.“ gnupg.org. https://wiki.gnupg.org/EasyGpg2016 (aufgerufen 31.12.2021).

[8] U. Heimeshoff. Internet und Wettbewerb: Ökonomische Grundlagen [Powerpoint-Präsentation]. Verfügbar: https://www.jura.hhu.de/fileadmin/redaktion/Fakultaeten/Juristische_Fakultaet/Kersting/Sonstiges/Vortrag-oekonomische-Grundlagen-Internet.pdf (aufgerufen 21.12.2021).

[9] W. Koch. „OpenPGP Web Key Directory.“ datatracker.ietf.org. https://datatracker.ietf.org/doc/draft-koch-openpgp-webkey-service/12/ (aufgerufen 11.09.2021).

[10] The GnuPG Project. „Validating other keys on your public keyring.“ gnupg.org. https://www.gnupg.org/gph/en/manual.html#AEN335 (aufgerufen 04.01.2022).

[11] A. Heinecke. „Automated Encryption.“ wiki.gnupg.org. https://wiki.gnupg.org/AutomatedEncryption (aufgerufen 16.01.2022).

[12] J. Nielsen, *Usability Engineering*, San Francisco, USA: Morgan Kaufmann, 1993.

[13] J. Nielsen. „10 Usability Heuristics for User Interface Design.“ Nielsen Norman Group. https://www.nngroup.com/articles/ten-usability-heuristics/ (aufgerufen 04.01.2022).

[14] A. Harley. „Visibility of System Status (Usability Heuristic #1).“ Nielsen Norman Group. https://www.nngroup.com/articles/visibility-system-status/ (aufgerufen 05.01.2022).

[15] M. Rosala. „User Control and Freedom (Usability Heuristic #3).“ Nielsen Norman Group. https://www.nngroup.com/articles/user-control-and-freedom/ (aufgerufen 05.01.2022).

[16] A. Kaley. „Match Between the System and the Real World: The 2nd Usability Heuristic Explained.“ Nielsen Norman Group. https://www.nngroup.com/articles/match-system-real-world/ (aufgerufen 05.01.2022).

[17] R. Krause. „Maintain Consistency and Adhere to Standards (Usability Heuristic #4).“ Nielsen Norman Group. https://www.nngroup.com/articles/consistency-and-standards/ (aufgerufen 05.01.2022).

[18] R. Budiu. „Memory Recognition and Recall in User Interfaces.“ Nielsen Norman Group. https://www.nngroup.com/articles/recognition-and-recall/ (aufgerufen 05.01.2022).

[19] P. Laubheimer. „Preventing User Errors: Avoiding Unconscious Slips.“ Nielsen Norman Group. https://www.nngroup.com/articles/slips (aufgerufen 05.01.2022).

[20] A. Joyce. „Help and Documentation: The 10th Usability Heuristic.“ Nielsen Norman Group. https://www.nngroup.com/articles/help-and-documentation/ (aufgerufen 05.01.2022).

[21] P. Laubheimer. „Flexibility and Efficiency of Use: The 7th Usability Heuristic Explained.“ Nielsen Norman Group. https://www.nngroup.com/articles/flexibility-efficiency-heuristic/ (aufgerufen 05.01.2022).

[22] H. Balzert. *Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering*, 3. Auflage. Heidelberg, Deutschland: Spektrum Akademischer Verlag, 2009.

[23] S. Krug. *Don’t make me think, Revisited: A Common Sense Approach to Web Usability*, San Francisco, USA: Pearson Education, 2014.

[24] M. Kuketz. „GnuPG: Web Key Directory (WKD) einrichten.“ Kuketz IT-Security Blog. https://www.kuketz-blog.de/gnupg-web-key-directory-wkd-einrichten/ (aufgerufen 16.01.2022).

[25] MZLA Technologies Corporation. „Building Thunderbird.“ Thunderbird.net. https://developer.thunderbird.net/thunderbird-development/building-thunderbird (aufgerufen 09.02.2022).

[26] B. Reiter. „Wandel der IT: Mehr als 20 Jahre Freie Software.“ intevation.de. https://intevation.de/~bernhard/publications/200408-hmd/200408-wandel_der_it_20j_fs.html (aufgerufen 16.01.2022).

[27] Wikipedia. „Comparison of Mail clients.“ Wikipedia. https://en.wikipedia.org/wiki/Comparison_of_E-Mail_clients (aufgerufen 30.10.2021).

[28] T. Oberndörfer. „Häufig gestellte Fragen.“ mailvelope.com. https://mailvelope.com/de/faq#proof_signing (aufgerufen 06.01.2022).

[29] Claws Mail. „Latest News.“ claws-mail.org. https://www.claws-mail.org/news.php, (aufgerufen 30.10.2021).

[30] MZLA Technologies Corporation. „Getting Started.“ Thunderbird.net. https://developer.thunderbird.net/thunderbird-development/getting-started (aufgerufen 09.12.2021).

[31] I. Klöcker. „Error when trying to locate key via WKD.“ gnupg.org. https://lists.gnupg.org/pipermail/gnupg-users/2021-October/065510.html (aufgerufen 30.10.2021).

[32] Claws Mail. „Developers.“ claws-mail.org. https://www.claws-mail.org/devel.php (aufgerufen 23.12.2021).

[33] Marcel Bokhorst. „Contributing.“ GitHub. https://github.com/M66B/FairEmail, (aufgerufen 23.12.2021).

[34] C. Klassen. „Use WKD through GPGME.“ claws-mail.org. https://lists.claws-mail.org/pipermail/users/2021-November/028871.html (aufgerufen 09.11.2021).

[35] C. Klassen. „Suggestion: Automatically activate privacy system.“ claws-mail.org. https://lists.claws-mail.org/pipermail/users/2021-November/028870.html (aufgerufen 09.11.2021).

[36] C. Klassen. „Vorschlag der Anzeige vorhandener Schlüssel in FairEmail.“ XDA Forums. https://forum.xda-developers.com/t/app-5-0-fairE-Mail-fully-featured-open-source-privacy-oriented-E-Mail-app.3824168/post-85941965 (aufgerufen 11.01.2022).

[37] C. Klassen. „Vorschlag des automatischen Beziehens von Schlüsseln in FairEmail.“ XDA Forums. https://forum.xda-developers.com/t/app-5-0-fairE-Mail-fully-featured-open-source-privacy-oriented-E-Mail-app.3824168/post-85945451 (aufgerufen 11.01.2022).

[38] C. Klassen. „Falsche Fehlermeldung.“ XDA Forums. https://forum.xda-developers.com/t/app-5-0-fairemail-fully-featured-open-source-privacy-oriented-email-app.3824168/post-86025199 (aufgerufen 03.02.2022).

[39] J. Johnson, *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines*, 2. Auflage. Waltham, USA: Elsevier Inc., 2014.

[40] T. Oberndörfer. „Mailvelope 3.0: Verschlüsselte Web-Formulare.“ mailvelope.com. https://mailvelope.com/de/blog/mailvelope-3.0 (aufgerufen 17.01.2022).

[41] P. Stenner. „Audio-Schnittprogramm will Daten sammeln und mit Regierungen teilen.“ netzpolitik.org. https://netzpolitik.org/2021/audacity-audio-schnittprogramm-will-daten-sammeln-und-mit-regierungen-teilen/ (aufgerufen 28.01.2022).

[42] B. Reiter. „WKD Research: Measuring use. An mailinglist maintainers that would help?.“ gnupg.org. https://lists.gnupg.org/pipermail/gnupg-users/2021-October/065489.html (aufgerufen 28.01.2022).

[43] A. Matallaoui, N. Hanner und R. Zarnekow, „Introduction to Gamification: Foundation and Underlying Theories,“ in *Gamification: Using Game Elements in Serious Contexts*, S. Steiglitz, C. Lattemann, S. Robra-Bissantz, R. Zarnekow und T. Brockmann, Cham, Schweiz: Springer International Publishing, 2017.

Eidesstattliche Versicherung

Name:	Klassen
Vorname:	Christoph
Matrikel-Nr.:	2180089
Studiengang:	Computervisualistik und Design

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig und nur unter Benutzung der angegebenen Literatur und Hilfsmittel angefertigt habe. Wörtlich übernommene Sätze und Satzteile sind als Zitate belegt, andere Anlehnungen hinsichtlich Aussage und Umfang unter Quellenangabe kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen und ist auch noch nicht veröffentlicht.

Auszug aus dem Strafgesetzbuch (StGB)
§156 StGB: Falsche Versicherung an Eides Statt
Wer von einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

Lippstadt, 15.02.2022 _____

Ort, Datum Unterschrift