HUMBOLDT-UNIVERSITÄT ZU BERLIN
MATHEMATISCH-NATURWISSENSCHAFTLICHE FAKULTÄT
INSTITUT FÜR INFORMATIK

Diplomarbeit

zur Erlangung des akademischen Grades
Diplominformatiker (Dipl.-Inf.)

Reduktion von Metadaten in PGP-verschlüsselten E-Mails

eingereicht von: Torsten Dänicke

Gutachter: Prof. Jens-Peter Redlich
Prof. Ernst-Günter Giessman

eingereicht am: …….. verteidigt am: ……..

Inhalt

Verzeichnis der Abbildungen

Verzeichnis des Anhangs

1 Einleitung

„The debate isn't security versus privacy. It's liberty versus control.“
- Bruce Schneier in [1]

1.1 Motivation

"Daten, die anderen Daten übergeordnet sind" werden laut Fremdwörterbuch [2] Metadaten genannt. Ab dem Sommer 2013 mehrten sich Berichte über ein besonderes Interesse von Geheimdiensten an Daten. Edward Snowdens Veröffentlichungen zeigten, dass mit großem Aufwand per Telekommunikation übertragene Inhalte und deren Metadaten erfasst und gespeichert werden. Aus Metadaten lassen sich Beziehungen und Kontakte zwischen Personen und Institutionen ableiten. Daraus erstellte soziale Graphen veranschaulichen Hierarchien und Gruppenzugehörigkeiten. Die Nutzung von Metadaten hat Vorzüge. Forschung in der Biologie ohne Taxonomie ist schwer vorstellbar. Ohne Metadaten wären Bibliotheken sinnlos. Andererseits ist es eine Bedrohung. General Michael Hayden sagte während einer außenpolitischen Fachtagung [3] 2014 „*We kill people based on metadata*“.

Auf Seite 22 in [4] schreibt Biermann, dass von Geheimdiensten abgegriffene E-Mails in unbekannter Menge und für unbekannte Zeit gefiltert und gespeichert werden, sind die E-Mails verschlüsselt, so werden sie „*wahrscheinlich für ewig*“ gespeichert.

Kurz: Die Interessen verschiedener Gruppen an Metadaten sind mannigfaltig. Metadaten sind nützlich, werden erstellt und fallen an, sie werden gesammelt, gebraucht und missbraucht. Dass sie einen Teil unserer privaten Lebensführung offenbaren ist unvermeidlich.

Der Schutz der Privatsphäre fällt unter den Datenschutz. Rechtliche Aspekte werden hier nicht im Einzelnen diskutiert.

Ein Bußgeld, 2013 vom Bayerischen Landesamt für Datenschutzaufsicht gegen eine Mitarbeiterin eines Unternehmens verhängt, zeigt jedoch an, dass eine E-Mail an mehrere Empfänger die Privatsphäre der Empfänger verletzen kann, jedoch nicht erlaubt ist, wenn die Empfängeradressen in das To- oder Cc-Feld eines Mailprogramms eingetragen werden. Alle, welche die versendete Nachricht bekommen, erfahren die Adressen der jeweils anderen Empfänger. E-Mailadressenharvester, angesiedelt im Übertragungsweg der E-Mail oder als Trojaner auf einem Computer des Endanwenders, können die gesammelten Adressen für Spamaktionen verwenden oder soziale Graphen erstellen. Auch Strafverfolgungsbehörden, die gezielt eine Person überwachen, könnten anhand der Adressen in einer beschlagnahmten oder abgefangenen E-Mail weitere Überwachungsmaßnahmen für andere Personen anordnen. Ein Absender kann jedoch Zurückhaltung in der Preisgabe von einem persönlichem Datum wie die E-Mailadresse zeigen, indem die E-Mailadressen in das Bcc-Feld des Mailprogrammes eingetragen werden. Diese darin enthaltenen Adressen werden nicht bei den Empfängern angezeigt. Die Anzahl der Empfänger wird bei Nutzung von Bcc ebenfalls nicht preisgegeben. Mit der Verwendung von Adressen in Bcc-Feldern des Mailclients sind diese E-Mailadressen in Rundmails vor eventuellem Missbrauch geschützt.

Anders ist es jedoch bei einer verschlüsselten Nachricht an mehrere Empfänger mit E-Mailadressen im Bcc-Feld. Ein Empfänger der Nachricht erfährt dann:

(a) die Anzahl der weiteren Empfänger der Nachricht und

(b) die jeweilige Key-ID der Public-Keys der anderen Empfänger.

Nun ist PGP so konzipiert, dass jedes Schlüsselpaar mit mindestens einer E-Mailadresse verknüpft ist. Verschiedene öffentliche Schlüsselserver im Internet können durch Angabe einer Key-ID eine dazugehörige E-Mailadresse liefern. Eine Preisgabe der E-Mailadressen lässt sich zwar vermeiden, indem Enigmail vor dem Versenden der Nachricht angewiesen wird den jeweiligen öffentlichen Schlüssel zu „verstecken“. Die Anzahl der mitgesendeten Schlüsselpakete und damit die Anzahl der Empfänger bleibt jedoch bekannt. Darüber hinaus warnt Enigmail in diesem Fall die Nutzer, dass bestimmte Implementationen von PGP die Nachricht nicht mehr öffnen könnten, sollten ″versteckte″ E-Mailadressen vorhanden sein.

Ein Dilemma: Eine OpenPGP-Implementation sichert, wie im Kapitel 2.1. in [5] angegeben, die Vertraulichkeit der Nachricht durch Verschlüsselung. Wird jedoch die Nachricht an Bcc-Empfänger verschickt, tauchen die Key-IDs respektive Bcc-Adressen in der OpenPGP-Nachricht wieder auf. Mit ein wenig Geschick und ohne Kenntnis von Passphrasen oder privaten Schlüsseln können diese E-Mailadressen von Strafverfolgungsbeamten aus einer konfiszierten Nachricht ermittelt werden. Mailserverbetreiber, die mit Hilfe sozialer Graphen ihre Werbestrategien optimieren wollen, könnten diese Daten ebenfalls wertvoll finden.

1.2 Ziel der Arbeit

Ziel dieser Arbeit ist zu klären, welche Daten ein Mailclient, im besonderen Fall einer an verschiedene Empfänger versendeten PGP-verschlüsselten E-Mail überträgt. Für den Fall der Existenz mehrerer Empfänger einer verschlüsselten Nachricht, wobei mindestens einer die E-Mail als Blind Carbon Copy erhalten soll, soll ermittelt werden, ob eine Möglichkeit besteht, die Nachricht, ohne sie mehrmals dem SMTP-Server zu schicken, derart gestaltet werden kann, dass die Empfänger keine Kenntnis darüber erlangen, dass und ob die Nachricht an einen oder mehreren Bcc-Empfängern übermittelt wurde.

Diese Arbeit wird zeigen, dass es eine Möglichkeit gibt den Versand PGP-verschlüsselter Nachrichten in E-Mails mit Bcc-Empfängern so zu gestalten, dass keine Rückschlüsse auf die Anzahl der Bcc-Empfänger und deren E-Mailadressen möglich sind und zweitens wird ein Konzept skizziert, wie ein Mailserver mittels Filtersoftware Metadaten einer PGP-verschlüsselten E-Mail eliminieren kann. Diese Arbeit versucht weder kryptographische Algorithmen zu schwächen, zu brechen oder eventuell vorhandene Schwächen auszunutzen.

1.3 Verwandte Arbeiten oder Konzepte

Projekte, die E-Mailempfängerinformationen trotz oder gerade bei Verschlüsselung verschleiern können, sind hier in nicht erschöpfender Menge benannt.

Schleuder[1]

Schleuder ist eine GPG-fähige Mailinglistensoftware mit Remailingfunktion, veröffentlicht unter der GNU GPL Version 2.

Die Listenteilnehmer können miteinander verschlüsselt unter Pseudonymen kommunizieren und auch an Nicht-Listenmitglieder über die Liste E-Mails verschicken.

Auf der Webseite [6] wird das Konzept folgendermaßen zusammengefasst:

> *„Each list has its own keypair. Schleuder decrypts every incoming email and verifies its signature with the keys from the list's keyring. Then Schleuder loops over the list of subscribers, creates for each a stripped down copy of the message, encrypts it with the subscriber's key and signs it with its own key, and sends it out.“*

Übersetzung: „Jede Liste hat ihr eigenes Schlüsselpaar. Schleuder entschlüsselt jede eingehende E-Mail und überprüft ihre Signatur anhand der Schlüssel des Listenschlüsselrings. Dann geht Schleuder die Liste der Abonnenten durch, kreiert für jeden eine enthüllte Kopie der Nachricht, verschlüsselt sie mit dem Abonnentenschlüssel und signiert sie mit ihrem eigenen Schlüssel, und schickt sie raus.“

Schleuder ist ein Man in the Middle mit Zugriff auf die Klartexte der Nachrichten. Die Empfänger können nur über eine korrekte Signatur des Absenders feststellen, ob die Nachricht authentisch ist und nicht verändert wurde. Auch hier ist es für einen Abonnenten nicht möglich festzustellen, wie viele Empfänger existieren und wer die Nachricht bekommen hat. Das Ergebnis jedoch deckt sich mit den Zielen der in dieser Arbeit vorgeschlagenen Lösung.

CipherMail Email encryption gateway[2] und **Z1 Secure Mail Gateway**[3]

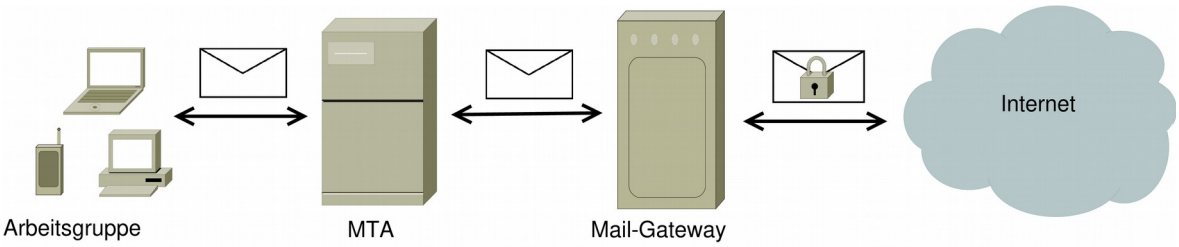

Abbildung 1: Konzept Mailgateway (Quelle modifiziert nach [7])

Beide Lösungen sind zentral gemanagte E-Mailserver als Gateway bzw. SMTP-Proxy konzipiert (siehe Abbildung 1). Ausgehende Nachrichten werden verschlüsselt und an die für die Empfänger zuständigen MTAs weitergeleitet. Da das Mail-Gateway Zugriff auf die noch unverschlüsselte Nachricht hat, könnten E-Mails mit mehreren Empfängern einzeln vom Gateway versendet werden. Ein Empfänger hätte weder Kenntnis von der Anzahl der Empfänger der Nachricht noch von deren E-Mailadressen. Eingehende verschlüsselte E-

[1] https://schleuder2.nadir.org/ (Stand 19.04.2018)

[2] https://www.ciphermail.com/ (Stand 19.04.2018)

[3] https://www.zertificon.com/loesungen/email-verschluesselung-gateway (Stand 19.04.2018)

Mails werden vom Gateway entgegengenommen, entschlüsselt und an den internen MTA weitergeleitet, der die Postfächer der Arbeitsgruppenmitglieder hostet. Nachteil dieser Lösungen ist wie bei Schleuder ein Man in the Middle, mit Zugriff auf den Klartext der Nachrichten.

GoodCrypto[4]

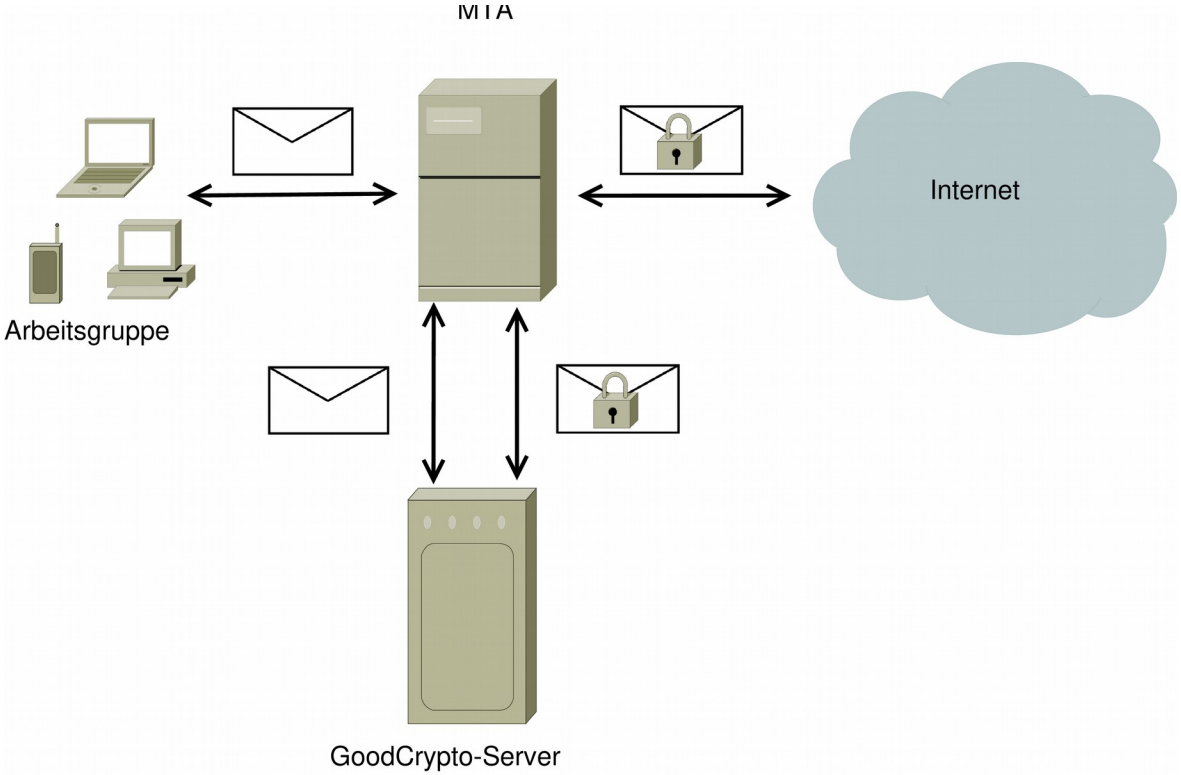

Abbildung 2: Design GoodCrypto (Quelle: modifiziert nach [8])

GoodCryptos Design ist ein Mailfilter. Mailclients der Arbeitsgruppe versenden ihre Mails unverschlüsselt an den MTA, der sie zum GoodCrypto-Server weiterleitet, wo die Nachrichten verschlüsselt werden. Danach schickt der GoodCrypto-Server die Nachrichten zurück zum MTA, der sie letztendlich ausliefert. Aus dem Internet eingehende verschlüsselte Nachrichten werden an den GoodCrypto-Server weitergereicht, der sie entschlüsselt und beim MTA abliefert. Von dort aus können sich die Arbeitsgruppenmitglieder ihre E-Mail abholen.

Das Schlüsselmanagement ist zentral auf dem GoodCrypto-Server ausgelagert. Die Webseite des Projektes wirbt damit, dass alle privaten Schlüssel auf dem firmeneigenen GoodCrypto-Server gehalten werden. Eine Verschlüsselung ist nur möglich, wenn zwei Firmen GoodCrypto einsetzen oder eine Person ohne GoodCrypto-Umgebung den eigenen öffentlichen Schlüssel auf dem GoodCrypto-Server hinterlegt hat. Theoretisch ist es dem GoodCrypto-Server möglich, die Nachrichten so aufzubereiten, dass der oder die Empfänger der verschlüsselten Nachrichten wie gewünscht keine Kenntnis von weiteren Empfän-

[4] https://goodcrypto-private-server.sourceforge.io/ (Stand 19.04.2018)

gern erlangen. Auch hier ist nachteilig, dass ein GoodCrypto-Server Klartextnachrichten verarbeitet.

1.4 Aufbau der Arbeit

Um die Struktur einer PGP-verschlüsselten E-Mail zu veranschaulichen, beschäftigt sich Kapitel 2 mit den Grundlagen und elementaren Begriffen, wie sie die in den von der Internet Society, speziell vom RFC-Editor, veröffentlichten Standards beschreibenden Request for Comments definiert sind.

Kapitel 3 beinhaltet die Beschreibung der praktischen Tätigkeiten der Arbeit auf der Suche nach den tatsächlichen Daten, die beim Austausch von Mails übertragen werden und den Versuch Metadaten zu identifizieren und zu extrahieren. Dazu werden Testmails generiert und deren Versand wird an der Netzwerkschnittstelle des Rechners protokolliert. Anschließend werden die protokollierten Daten mit den gespeicherten Daten verglichen und analysiert.

Mailfilterfunktionsweisen und Vorschläge zum Design eines Mailfilters, der Metadaten aus PGP-verschlüsselten E-Mails extrahiert finden sich im Kapitel 4.

Die Schlussbetrachtungen des Kapitels 5 beenden diese Arbeit mit einer Zusammenfassung der Ergebnisse und benennen Fragen, die diese Arbeit nicht oder nicht hinreichend genau beantworten konnte. Eine Auseinandersetzung mit diesen offenen Fragen ist für einen am Schutz der Privatsphären interessierten Mailfilterentwickler aus meiner Sicht notwendig, um das hier vorgestellte Design eines metadatenreduzierenden Mailfilters erfolgreich zu implementieren.

1.4.1 Konventionen

Die hier verwendete Abkürzung PGP steht für „Pretty Good Privacy“. Ein 1991 von Phil Zimmerman entwickeltes Programm zum Signieren, der Ver- und Entschlüsselung von Daten. Derzeit ist PGP in privater Hand. Die Open-Source-Implementation GNU Privacy Guard (GPG) ist eine freie Alternative zu PGP und folgt dem Standard von OpenPGP. GPG und PGP werden hier synonym betrachtet.

Jegliche in dieser Arbeit erwähnten Markennamen unterliegen uneingeschränkt dem jeweils gültigen Marken- und Besitzrecht des betreffenden Eigentümers. Das Weglassen einer ausdrücklichen Kennzeichnung des Markennamens soll nicht den Eindruck erwecken, der Markenname sei nicht durch Dritte geschützt.

Um das Lesen dieser Arbeit zu erleichtern, habe ich typografische Konventionen zu einer Schriftart gewählt:

- `Monospace`
  Monospace Schrift wird im Fließtext und Anhang verwendet für:
  - Beispiele von Mailquellen und OpenPGP-Zertifikaten,
  - einzelnen Header Fields, deren Optionen und Parameter,

- Verzeichnis- und Dateinamen und
- Kommandos und deren Optionen.

- **`Monospace fett`**
  Kommandos und Optionen in der Kommandozeile sind in fett gedruckter Monospace Schrift gehalten.

1.4.2 CD-Beilage

Auf der beigelegten CD befinden sich:

- Die aufgezeichneten und gespeicherten Datenübertragungsprotokolle im Verzeichnis: `/Wiresharkmitschnitte`. Die Dateinamen beschreiben die jeweilige Kategorie.
- Mailquelltexte im eml-Format unterhalb des Ordners: `/Mailquellen`. Zur Veranschaulichung der Unterschiede der Mailquellen von Sender und Empfänger sind sie nach den Personennamen sortiert. Der Ordner `Bob.sanitize` enthält Kopien der Dateien aus dem Ordner `Bob` mit nummerierten Dateinamen. Die Nummerierung der Dateien im Unterverzeichnis `Bob.sanitize` ist identisch mit der im Kapitel 3.2 verwendeten Nummerierung aus den Tabellen. Beispielhaft sind einige Dateien als binäre OpenPGP-Nachrichten gespeichert. Die im Abschnitt 3.2.4 erzeugten Dateien sind dort ebenfalls abgelegt.
- Das als Attachment verwendete `/Mailquellen/dummy-anhang.odt`.
- Die Diplomarbeit in elektronischer Form: `/Diplomarbeit.pdf`.

2 Grundlagen und Begriffe

Bevor mit Hilfe des Internets eine E-Mail von einer Person A geschrieben und gesendet und von einer Person B empfangen und gelesen werden kann, müssen die verwendeten E-Mail-, Verschlüsselungsprogramme und die an der Übertragung beteiligten Mailserver gewisse Standards der Protokolle und Prozesse einhalten. Diese Standarddefinitionen werden in den Request for Comments (RFC) festgehalten. Die RFCs werden von einigen Gremien der ISOC begutachtet und bewertet. RFCs mit einem bestimmten Reifegrad werden dann von den Gremien in einen bestimmten Status gehoben. RFC 2026 beschreibt den Standardisierungsprozess und die einzelnen Level, die ein Request for Comments erreichen kann. Erst die RFCs mit einem *"Standards Track Maturity Level"* aus Kapitel 4.1.in [9] sind die Basis für Softwareentwickler geeignete Programme zu schreiben.

2.1 Anatomie einer E-Mail

Dieses Kapitel beschreibt die Protokolle und Normen, die erforderlich sind, damit eine E-Mail mittels Telekommunikation übertragen und vom Empfänger korrekt dargestellt werden kann. Gewöhnlich wird die am Computer verfasste Nachricht vom Mailclient des Nutzers zu einem Mailserver geschickt. Mailclient und Mailserver kommunizieren untereinander mit Hilfe des Simple Mail Transfer Protokolls (SMTP).

Das Simple Mail Transfer Protokoll transportiert in der Terminologie der RFC 5321 ein Mailobjekt. Ein Mailobjekt umfasst den Envelope und den Content. Im Envelope werden die Steuerungsdaten für den Mailserver übertragen, um den Content zu transportieren und zuzustellen.

Vergleicht man einen konventionellen Brief mit einer E-Mail, so besteht er aus einem Umschlag (dem Envelope), einem Briefkopf (den Header Fields) und dem eigentlichen Brieftext (dem Body).

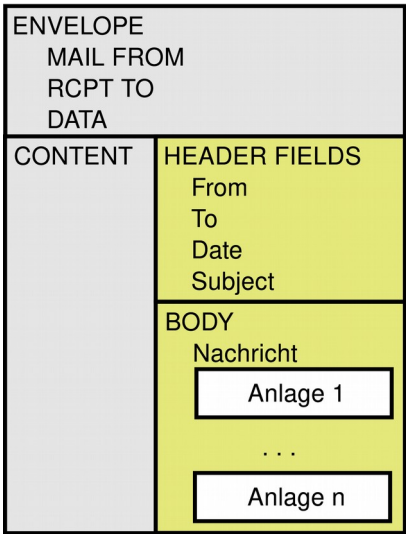

Abbildung 3: Schema einer E-Mail (Quelle: modifiziert nach [10])

Auf dem Umschlag sind Empfänger und Absender vermerkt. Um jedoch andere z.B. länderspezifische Zeichen oder Binärdaten als Anlagen übertragen zu können sind die Multipurpose Internet Mail Extensions (MIME) notwendig. Es gibt sowohl MIME Dokumente für Erweiterungen im Body als auch jene, die spezielle Formate in den Header Fields definieren.

Aus dem Beispiel der Abbildung 3 wird später ersichtlich, dass die Nachricht, bestehend aus den Header Fields und dem Body, im SMTP-Dialog eingebettet ist.

2.1.1 SMTP Envelope

Eine Folge von SMTP Protokollfeldern bilden den SMTP Envelope. Drei wichtige Felder überträgt ein Mailclient beim Versenden einer Nachricht immer:

- `MAIL FROM:`
- `RCPT TO:`
- `DATA`

Startet der Client eine Verbindung zum Server, der mit einer Eröffnungsnachricht antwortet, ist die SMTP Sitzung eröffnet. Gewöhnlich sendet der Client zu Beginn der Transaktion das `EHLO` Kommando. Einerseits um sich vorzustellen und andererseits signalisiert der Client seine Fähigkeit Serviceerweiterungen verarbeiten zu können, die ebenfalls mit diesem Kommando vom Server abfragt werden. Dem `MAIL FROM:` folgt die E-Mailadresse des Absenders, dem `RCPT TO:` die Adresse des Empfängers. Pro Empfänger wird ein `RCPT TO:` generiert und übertragen [11; Kapitel 3.3.]. Jedes Feld wird dabei vom MTA mit einer Meldung quittiert. Sendet der Mailclient den Befehl `DATA` quittiert der MTA dieses und wartet auf die Zustellung der zu übertragenden Nachricht. Der Mailclient sendet den Content. Beendet wird die Übertragung der Nachricht mit einem einzelnen Punkt ″.″ in einer neuen Zeile. Der MTA quittiert, ob die Nachricht angenommen wurde. Falls ja, verabschiedet sich der Client mit einem QUIT. Nach Auslieferung der E-Mail in das Postfach des Nutzers oder der erfolgreichen Weiterleitung zu einem anderen MTA werden die Envelopedaten gelöscht. Die verwendeten Zeichen der Envelope-Kommandos und deren Antworten sind 7-bit ASCII kodiert. Sollte der Transportdienst einen 8-bit Übertragungskanal anbieten, so werden die 7-bit Zeichen in 8-bit Oktette mit jeweils 0 als most significant bit umgewandelt und übertragen [11; Kapitel 2.4.].

2.1.2 SMTP Content

Der Content wird während der SMTP DATA Protokolleinheit gesendet und besteht laut der Vorgabe von [11; Kapitel 2.3.1.] wiederum aus zwei Teilen:

- einer Header Section und
- dem Body.

Der derzeitige Standard des Internet Message Formats ist in der RFC 5322 von 2008 festgelegt. Ist der Body strukturiert, erfüllt er die Standards der RFC 2045 und einer Reihe begleitender Vorschriften wie z.B. der RFC 1652 (8BITMIME) und der RFC 2047 (Nicht-

ASCII Text im Header). Der Content ist textueller Natur und verwendet Zeichen des US-ASCII Codes.

Ein Update der Nomenklatur findet sich in der RFC 5322 bezüglich des Headers. Während in „alten“ RFCs nur vom Header gesprochen wird, so sind diese jetzt präzisiert als Header Field für den Inhalt einer Kopfzeile und die Header Section für die gesamte Sammlung der einzelnen Kopfzeilen.

Der SMTP Content umfasst das was der Empfänger bekommt: Header Section und Body. Diese beiden Teile werden vom Mailclient dem Empfänger präsentiert.

```
C: $ nc wp147.webpack.hosteurope.de 587
S: 220 wp147.webpack.hosteurope.de ESMTP Host Europe Mail Service Wed, 03 Jan 2018 11:52:49
+0100
C: EHLO
S: 250-wp147.webpack.hosteurope.de Hello ip5f5bd703.dynamic.kabel-deutschland.de
[95.91.215.3]
S: 250-SIZE 36700160
S: 250-8BITMIME
S: 250-PIPELINING
S: 250-AUTH PLAIN LOGIN
S: 250-STARTTLS
S: 250 HELP
C: AUTH PLAIN [Username+Passwort Base64-kodiert]
S: 235 Authentication succeeded
C: MAIL FROM:alice@daenicke.org
S: 250 OK
C: RCPT TO:bob@daenicke.org
S: 250 Accepted
C: DATA
S: 354 Enter message, ending with „.“ on a line by itself
C: From: Alice <alice@daenicke.org>
C: To: Bob <bob@daenicke.org>
C: Subject: Testbetreff
C:
C: Hallo Bob!
C: .
S: 250 OK id=1doqbC-0007H1-L9
C: QUIT
```

Beispiel SMTP-Dialog

In diesem Beispiel wurde das Terminalprogramm `nc`, Abkürzung für netcat, zum Öffnen einer TCP-Verbindung auf den alternativen SMTP-Port 587 des Mailservers verwendet. Die Zeichenfolgen „C:“ (Client) und „S:“ (Server) sind nachträglich zur Verdeutlichung des Dialogs eingefügt worden. Anstelle des SMTP-Kommandos `HELO` wurde `EHLO` (Extended `HELO`) als freundliche Begrüßungsformel benutzt, um spezielle ESMTP-Serverfunktionen abzufragen. Bei einem `HELO` darf der Mailserver nicht mit der Liste seiner Erweiterung antworten [11; Kapitel 3.2.]. Dem Kommando `AUTH PLAIN` folgt ein String einer Ba-

se64-Kodierung, der aus Benutzernamen und Passwort als Authentifizierung des Absenders Alice erstellt wird. Die Authentifizierung soll den Missbrauch des Mailservers für Spam verhindern bzw. erschweren.

Alice schickt drei Zeilen der ausgefüllten Header Fields `From:`, `To:` und `Subject:` und eine Leerzeile. Daran anschließend folgt der Body der Mail, ein kurzer Satz. Da die Nachricht hier endet, muss Alice einen einzelnen Punkt in der nächsten Zeile des Dialogs senden und der Mailserver akzeptiert die Mail. Mit einem `QUIT` beendet Alice die Mailübertragung. Der Standard der RFC 5321 fordert nach jeder Zeile ein <CR><LF>.

2.1.2.1 Internet Message Format

Die aktuelle Version der syntaktischen Beschreibung der zwischen Anwendern ausgetauschten Textnachrichten ist die RFC 5322 aus dem Jahr 2008, die eine Revision der RFC 2822 von 2001 ist, die wiederum die RFC 822 aus dem Jahr 1982 ablöste.

Die Header Section und der Body bilden die grundlegende Struktur einer Nachricht, die während der SMTP-Content-Phase übertragen wird.

2.1.2.1.1 Header Section

Da während des SMTP-Dialogs, genauer im SMTP-Envelope, alle Informationen für die Zustellung einer Nachricht übertragen werden, enthalten die Header Fields bzw. die Header Section keine für die Zustellung nötigen Daten. Lediglich ein Header-From und ein Header-Date sind erforderlich [12; Kapitel 3.6.]. Dieser Standard legt ebenso fest, dass eine Zeile nicht länger als 998 Zeichen sein darf, nicht länger als 78 Zeichen sein sollte und mit einem <CR><LF> von den anderen Zeilen getrennt ist [12; Kapitel 2.1.1.].

<CR><LF>, kurz CRLF, sind zwei Zeichen des US-ASCII-Standards für Carriage Return und Line Feed. Hexadezimal: 0D0A.

Die für diese Arbeit wichtigen Header Fields sind die folgenden:

- `From:` enthält die Mailadresse des Nachrichtenverfassers,
- `To:` enthält die Mailadresse des primären Empfängers,
- `Cc:` enthält Mailadressen weiterer Empfänger,
- `Bcc:` enthält die E-Mailadressen anderer Empfänger, die geheim bleiben sollen. Sie werden gewöhnlich vom MTA beim Verarbeiten der E-Mail aus der Header Section entfernt, sofern sie vom Mailclient mitgesendet werden.

Es ist erlaubt in den Header Fields `To:`, `Cc:` und `Bcc:` mehrere Mailadressen einzutragen. Diese werden durch Kommata getrennt. Nicht erlaubt hingegen sind mehrere `To:`-, `Cc:`- und `Bcc:`-Header Fields. Sie dürfen nur jeweils einmal im Header erscheinen. Hervorzuheben ist, dass für jeden Empfänger im SMTP-Envelope ein `RCPT TO:` generiert wird, unabhängig davon in welchem Header Field ein Empfänger eingetragen ist.

2.1.2.1.2 Body

Der Body ist durch eine einfache Leerzeile von der Header Section getrennt. Als Element des SMTP-Contents enthält der Body nur Zeilen von Zeichen in 7-bit ASCII-Kodierung. Hier steht im einfachsten Fall der eigentliche Text der Mail. Sollen Zeichen außerhalb des 7-bit ASCII-Standards übertragen werden, so wird der Body gemäß der MIME Spezifikationen kodiert und strukturiert.

2.1.2.2 MIME

Die Wurzeln des Konzepts der E-Mail reichen bis in die 60er Jahre des 20. Jahrhunderts. Rasant fand die E-Mail zwischenzeitlich weltweite Verbreitung. Es trat nicht nur das Problem mit länderspezifischen Zeichen auf. Umlaute und auch Nachrichten mit Bildern, Programmen oder anderen Binärdaten sollten übertragen werden können. Das spezifizierte Datenformat der RFC 5322 (IMF) und RFC 5321 (SMTP) erlaubt nur 7-bit ASCII und maximal 998 Zeichen in den Zeilen.

Abhilfe schaffen die Multipurpose Internet Mail Extensions (MIME). Sie beschreiben die Mechanismen zum Transport von Bildern, Audio oder anderen strukturierten Daten durch E-Mail. Sie erweitern entweder die in der RFC 5322 definierte Syntax oder strukturieren die Nachricht so, dass sie syntaktisch konform ist. Mittels MIME gelingt es den Body zu strukturieren. Ein Sender übermittelt dem Empfänger Informationen zum Typ der gesendeten Nachricht unter Verwendung von einem Content-Type-Feld und gibt für die Übertragung eine Zeichencodierung im Content-Transfer-Encoding-Feld an. Eine Serie von RFCs mit Bezug zu MIME redefinieren das Nachrichtenformat [13; S.1] neu, um:

1. *„textuelle Bodies in anderen Zeichenkodierungen als US-ASCII,*
2. *eine erweiterte Menge von unterschiedlichen Formaten für nicht-textuelle Bodies (Binärdaten),*
3. *Multi-Part Bodies, und*
4. *textuelle Header Informationen in anderen Zeichensätzen als US-ASCII zuzulassen.“*

Der MIME Part One (RFC 2045) führt zusätzliche Felder in der Header Section der Mail ein. Diese sind hier aufgezählt:

1. `MIME-Version:` deklariert eine Nachricht als MIME-konform, um zwischen Nachrichten unterscheiden zu können, die eben solche sind oder von älterer nichtkonformer Software generiert wurden. Derzeitiger Wert der MIME-Version ist 1.0.
2. `Content-Type:` Der Content Type signalisiert dem Mailclient einen Mechanismus auszuwählen, dem Nutzer die Daten zu präsentieren bzw. in angemessener Form zu behandeln [13; S.10].
3. `Content-Transfer-Encoding:` definiert, ob die Daten des MIME-Bodys transformiert wurden und welchem Bereich das Ergebnis einzuordnen ist.
4. Zwei weitere Header Fields: `Content-ID` und `Content-Description`.

Die RFC 2046, als zweiter Teil der MIME-Spezifikationen, definiert eine kleine Menge der Medientypen und deren Subtypen des Content-Type-Feldes. Diese Typen klassifizieren und strukturieren die Daten der Nachricht. Der Body der MIME-strukturierten Nachricht kann selbst wieder eine MIME-kodierte Nachricht sein. Die Gesamtstruktur einer MIME-Nachricht ist rekursiv [13; Kapitel 2.6.].

2.1.2.2.1 Content Type Header Field

Hier wird der Medientyp und der Subtyp der Daten im Body der MIME-Entität spezifiziert. Subtypen können durch Parameter modifiziert werden [13; Kapitel 5.]. Exemplarisch aufgeführte Medientypen und deren Subtypen im Content-Type-Feld sind:

- **Application Medientypen**
  Die Autoren von [14] bezeichnen im Kapitel 4.5. den „application“ Medientypen als Kategorie für Daten, die nicht in andere Kategorien einzuordnen sind und von bestimmten Programmen verarbeitet werden sollen.
  - `application/postscript`: Dieser Medientyp ist für Programme vorgesehen, die PostScript interpretieren können.
  - `application/octet-stream`: Zeigt an, dass der folgende Body willkürliche Binärdaten enthält. Es ist ein generischer Subtyp und die Daten sind nicht näher spezifiziert. Implementationen sollten anbieten diese Daten in eine Datei zu speichern.

- **Audio Medientypen**
  Der Medientyp „audio“ zeigt an, dass der Body der MIME-Nachricht Audiodaten enthält [14; Kapitel 4.3.].
  - `audio/basic`: Ein-Kanal-Audio, 8-bit ISDN µ-law, 8000Hz Samplerate.
  - `audio/x-mpeg`: Steht für *.mp2-Dateien.

- **Image Medientypen**
  Der Medientyp „image“ wird für Bilder im Body der MIME-Nachricht verwendet. Der Subtyp bezeichnet ein bestimmtes Bildformat. Unbekannte ″image″-Subtypen sollten der Spezifikation [14; Kapitel 4.2.] folgend die Daten als ″application/octet-stream″ behandeln.
  - `image/jpeg`: Bilder des JPEG-Formats mit JFIF-Kodierung.
  - `image/png`: Bilder im PNG-Format.

- **Message Medientypen**
  Gekapselte Nachrichten werden hiermit markiert. Der Subtyp „partial“ wird zur fragmentierten Übertragung einer gemäß RFC 822 formatierten Nachricht verwendet, welche zu groß für den Transport ist. Der Subtyp „rfc822“ enthält selbst eine

nach RFC 822 konforme Nachricht. Dateien mit der Endung .eml sind von diesem Medientyp.
  - `message/partial`
  - `message/rfc822`

- **Multipart Medientypen**
  Werden ein oder mehrere verschiedene Datensätze in einem MIME-Body kombiniert, muss ein ″multipart″ Medientyp im MIME-Header verwendet werden. Der Body enthält demnach ein oder mehrere Body-Parts, mit jeweils einer vorangestellten Trennlinie (boundary delimiter line) und der letzte Body-Part eine abschließende Trennlinie. Die Boundary, als Pflichtparameter für diesen Medientyp, besteht aus maximal 70 alphanumerischen Zeichen. Allen Trennlinien werden im MIME-Body zwei Bindestriche vorangestellt. Die letzte Trennlinie endet auch mit zwei nachgestellten Bindestrichen als Hinweis auf das Ende des gesamten Bodys.
  - `multipart/alternative`
  - `multipart/encrypted`
  - `multipart/signed`

- **Text Medientypen**
  Repräsentieren Daten mit Text und sind theoretisch menschenlesbar.
  - `text/html`
  - `text/plain`:
    Einfachster und wichtigster Subtype von ″text″. Der Defaultmedientyp in parametrisierter Form lautet `text/plain; charset=us-ascii`.

- **Video Medientypen**
  In diesen Medientypen sind Daten mit Videos verschiedener Formate enthalten.
  - `video/mpeg`
  - `video/quicktime`

2.1.2.2.2 Content Transfer Encoding Header Field

Da die RFC 821 und deren Nachfolger E-Mailnachrichten auf 7-bit ASCII-Zeichen mit Zeilen nicht länger als 1000 Zeichen beschränkt, können einige Daten der im vorigen Kapitel vorgestellten Medientypen nicht in ihrer natürlichen Form (8-bit Zeichen, binär) übertragen werden [13; Kapitel 6.]. Entweder müssen die Daten in 7-bit ASCII transformiert werden oder es wird angezeigt, entsprechende Fähigkeit des Mailservers vorausgesetzt, wie die Daten zu interpretieren sind.

Das Content Transfer Encoding Header Field kann 6 mögliche Werte beinhalten:
  - 7bit
  - 8bit
  - quoted-printable

- base64
- binary
- x-token

`7bit`

Der Inhalt des Medientyps besteht nur aus 7-bit ASCII-Text mit einer Zeilenbegrenzung von 1000 Zeichen. Das MSB von jedem Byte ist 0. Eine gesonderte Behandlung der Daten ist nicht nötig.

`8bit`

Der Inhalt des Medientyps besteht aus 8-bit Zeichen mit Zeilenbegrenzung von 1000 Zeichen pro Zeile. Das MSB eines übertragenen Bytes kann 1 sein. Eine Übertragung ist aber nur möglich, wenn der Mailserver ″8BITMIME″ unterstützt. 8BITMIME ist eine Diensterweiterung für ESMTP-Mailserver, die in der RFC 6152 definiert ist.

`Quoted Printable`

Ein Kodierungsverfahren, bei dem Zeichen außerhalb des dezimalen Bereiches von US-ASCII 9, 33 bis 60 und 62 bis 126 gesondert kodiert werden. Die Regeln finden sich im Kapitel 6.7. in [13]. So lassen sich nicht 7-bit Inhalte mittels SMTP übertragen. Zeilen sind nach der Kodierung nicht länger als 76 Zeichen, Zeilenumbrüche des Inhalts werden ebenfalls gesondert kodiert und nicht-7-bit ASCII-Zeichen werden durch zwei 7-bit Zeichen, denen ein Gleichheitszeichen vorangestellt ist, dargestellt.

Beispiel: Inhalt vom Subject Header Field ″Grüße aus La Coruña!″ wird folgendermaßen quoted printable kodiert:

```
Subject: =?utf8?Q?Gr=C3=BC=C3=9Fe=20aus=20La=20Coru=C3=B1a!?=
```

`Binary`

Der Inhalt besteht aus einer unbeschränkten Folge von Oktetten. Die Nachricht hat binären Inhalt mit nicht limitierter Zeilenlänge. Die RFC 3030 definiert die Diensterweiterung der ESMTP-Server [15]. Analog zu 8BITMIME wird bereits im SMTP-Envelope mittels neu definiertem Schlüsselwort BINARYMIME die Fähigkeit abgefragt.

`Base64`

In diesem Kodierungsverfahren werden 24 Bit (drei Bytes) des Eingabestroms in jeweils vier 6-bit Blöcke geteilt. Die 6-bit Blöcke werden dann als ASCII-Zeichen des Base64-Alphabets, einer speziellen Umsetzungstabelle, interpretiert. Siehe [13; Kapitel 6.8.] und [16; Kapitel 4.]. Die Zeilenlänge ist auf 76 Zeichen begrenzt.

Zusammenfassend schließen die Autoren der RFC 2045, dass sich willkürliche Daten durch Nutzung von `MIME-Version`, `Content-Type` und `Content-Transfer-Encoding` standardisiert in eine Mailnachricht einbinden lassen [13; Kapitel 10.]. Dadurch werden Restriktionen der RFC 821, RFC 822 und deren Nachfolger nicht verletzt.

2.1.2.2.3 Content-ID Header Field

Mailclients können dieses Feld benutzen. So kann ein MIME-Body auf einen anderen verweisen [13; Kapitel 7.].

2.1.2.2.4 Content-Description Header Field

Beschreibende Informationen in Form von Text können mit dem MIME-Body verknüpft werden [13; Kapitel 8.].

2.2 Anatomie einer OpenPGP-Message (RFC 4880)

Dieser Abschnitt der Arbeit beginnt mit der Syntax einer OpenPGP-Nachricht anhand derer der strukturelle Aufbau einer solchen Nachricht verdeutlicht wird. Anschließend wird erläutert wie die einzelnen Datensätze strukturiert sind. Der letzte Teil dieses Abschnitts widmet sich der ASCII-Hülle, einer, neben der binären, alternativen Darstellungsform, die Schäden durch Zeichensatzübersetzungen oder Datenumwandlung vorbeugt. Somit lassen sich OpenPGP-Nachrichten durch E-Mail sicher übertragen.

2.2.1 OpenPGP-Syntax

Das Kapitel 11.3. der RFC 4880 definiert die Syntax einer OpenPGP-Message. Die Regeln sind aus [5] zitiert. Ein Komma stellt eine sequentielle Verkettung dar, ein vertikaler Strich trennt Alternativen:

„OpenPGP Message :- Encrypted Message | Signed Message |
Compressed Message | Literal Message.

Compressed Message :- Compressed Data Packet.

Literal Message :- Literal Data Packet.

ESK :- Public-Key Encrypted Session Key Packet |
Symmetric-Key Encrypted Session Key Packet.

ESK Sequence :- ESK | ESK Sequence, ESK.

Encrypted Data :- Symmetrically Encrypted Data Packet |
Symmetrically Encrypted Integrity Protected Data Packet

Encrypted Message :- Encrypted Data | ESK Sequence, Encrypted Data.

One-Pass Signed Message :- One-Pass Signature Packet,
OpenPGP Message, Corresponding Signature Packet.

Signed Message :- Signature Packet, OpenPGP Message |
One-Pass Signed Message.“

Anmerkung: ESK ist die Abkürzung für Encrypted Session Key. Damit kann sowohl ein Public-Key-Encrypted-Session-Key-Paket als auch ein Symmetric-Key-Encrypted-Session-Key-Paket gemeint sein.

Verfasst beispielsweise Alice eine verschlüsselte Nachricht an Bob und Eve, so wird die Nachricht gemäß Syntax folgendermaßen strukturiert sein:

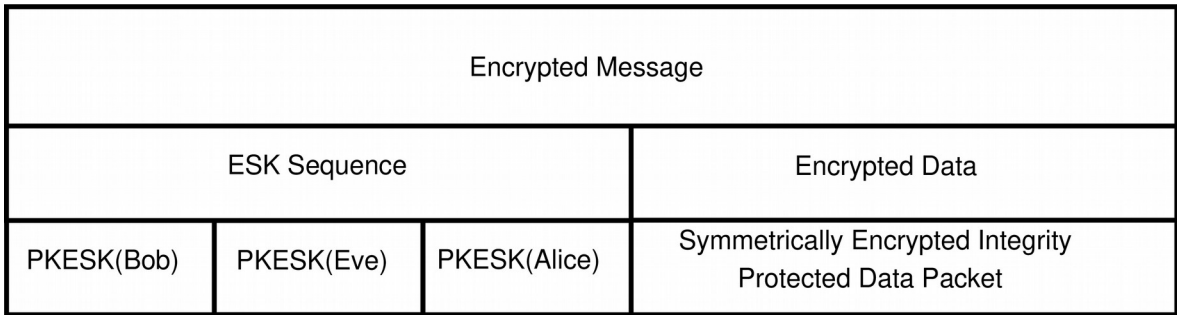

Abbildung 4: Schema einer OpenPGP-Message

2.2.2 Paketzusammensetzung

Die Autoren der RFC 4880 beschreiben 2007 eine OpenPGP-Nachricht als eine Anzahl von Datensätzen. Diese Datensätze werden Pakete genannt. Ein einzelnes Paket ist ein mit einem Tag (engl. für Markierung, Etikett) versehener Datenblock. Ein Tag bestimmt die Bedeutung des darauf folgenden Paket-Bodys.

Jedes Paket besteht aus einem Paket-Header und dem sich anschließenden Paket-Body:

Abbildung 5: Schema OpenPGP-Paket

2.2.2.1 Paket-Header

Der Paket-Header ist von variabler Länge. Im ersten Oktett ist der Paket-Tag kodiert, das Oktett definiert das Header Format und bezeichnet den Inhalt des Paket-Bodys, der Rest des Headers bestimmt die Länge des Pakets:

Abbildung 6: Schema Paket-Header

Mit dem Erscheinen der RFC 2440 von 1998, die 2007 von der RFC 4880 abgelöst wurde, führten die Autoren ein neues Format für die Länge des Pakets ein. Die Spezifikation unterscheidet seitdem (1998) zwischen Paketlängen im alten Format und denen im neuen Format.

2.2.2.1.1 Tag

Der Header startet mit dem Paket-Tag, welches den Paket-Typen definiert. Diese Tags werden in [5; Kapitel 4.3.] vorgestellt:

„0 -- Reserved - a packet tag MUST NOT have this value
1 -- Public-Key Encrypted Session Key Packet
2 -- Signature Packet
3 -- Symmetric-Key Encrypted Session Key Packet
4 -- One-Pass Signature Packet
5 -- Secret-Key Packet
6 -- Public-Key Packet
7 -- Secret-Subkey Packet
8 -- Compressed Data Packet
9 -- Symmetrically Encrypted Data Packet
10 -- Marker Packet
11 -- Literal Data Packet
12 -- Trust Packet
13 -- User ID Packet
14 -- Public-Subkey Packet
17 -- User Attribute Packet
18 -- Sym. Encrypted and Integrity Protected Data Packet
19 -- Modification Detection Code Packet
60 to 63 -- Private or Experimental Values“

Am Beispiel eines Public-Key-Encrypted-Session-Key-Pakets wird hier ein einleitendes Tag des Headers illustriert.

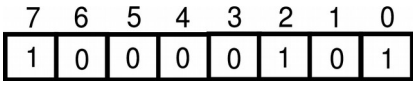

Abbildung 7: Ein Paket-Tag (Quelle: modifiziert nach [5; Kapitel 4.2.])

Im Paket-Tag befindet sich das most significant bit links, das Bit 7. Die Definition aus der RFC 4880 Kapitel 4.2. schreibt vor, dass:

- Bit 7 immer 1 ist,
- Bit 6, wenn es gesetzt ist, das neue Paketformat bestimmt. Aus Kompatibilitätsgründen mit PGP 2.6.x, welches das alte Paketformat benutzt, wird hier zwischen neuen und alten Formaten gewählt.

Unterschiede der Formate werden in den Kapiteln 4.2.1. und 4.2.2. der RFC 4880 erläutert.

Alte Paketformate enthalten:

- Bit 5-2: ein Paket-Tag,
- Bit 1-0: den Längen-Typ.

Neue Paketformate enthalten:

- Bit 5-0: Paket-Tag.

Das Beispiel aus Abbildung 7: 10000101_B = 0x85

Bit	Wert	Interpretation
7	1	Muss per Definition gesetzt sein.
6	0	Altes Paketformat wird benutzt.
5-2	0001	Public-Key-Encrypted-Session-Key-Paket.
1-0	01	Das Paket hat eine Länge von zwei Oktetten, der Header ist insgesamt drei Oktetts lang. [5; Kapitel 4.2.1.]

Bit 1 und Bit 0 haben zusammen einen Wert von 1. Die nächsten zwei Oktette des Headers werden als Länge des Bodys interpretiert. Nach der Längenangabe folgt der Body.

Schlussfolgerung: PKESK-Pakete im alten Format können folgende hexadezimalen Werte im Paket-Tag annehmen: 0x84, 0x85, 0x86, 0x87.

2.2.2.2 Paket-Body eines PKESK

Der Body des Public-Key-Encrypted-Session-Key-Pakets (PKESK) wird in [5; Kapitel 5.1.] definiert und besteht aus:

- *„einer Ein-Oktett Versionsnummer des Paket-Typs, derzeit ist es der Wert 3,*
- *einer Acht-Oktett Nummer, der die Key-ID des Public-Keys enthält, mit dem der Session Key verschlüsselt wurde. Sollte der Session Key mit einem Subkey verschlüsselt worden sein, so wird dessen Key-ID benutzt,*
- *das neunte Oktett ist der Parameter für den benutzten Public-Key Algorithmus,*
- *der Rest der Oktette ist der verschlüsselte Session Key.“*

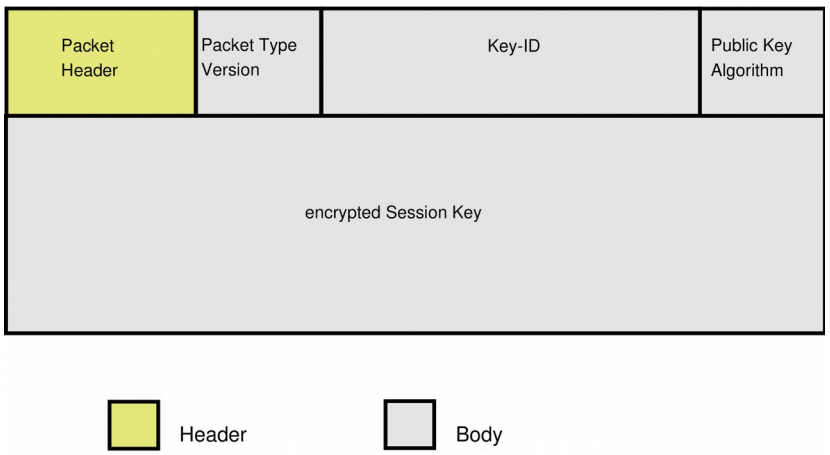

Abbildung 8: Schema Public-Key-Encrypted-Session-Key-Paket

Der Inhalt des verschlüsselten Session Keys füllt den Rest des Paketes aus und hängt vom verwendeten Public-Key Algorithmus (RSA oder Elgamal) ab.

Offenbar enthält das Public-Key-Encrypted-Session-Key-Paket Metainformationen; unter anderem die Key-ID, die ihrerseits an eine Identität in Form einer E-Mailadresse geknüpft ist.

Zur Verdeutlichung hier ein weiteres Beispiel: Der Anfang eines Public-Key-Encrypted Session-Key-Paketes

85 01 0c 03 13 51 16 13 14 14 01 d5 01...

Header			**Body**									
Tag	*Länge*		*Vers.*	*Key-ID*								*Alg.*
85	01	0c	03	13	51	16	13	14	14	01	D5	01

Hinweis: Die Definition des Wortes Key im Schlüsselmanagment ist sehr flexibel. Es kann einen bestimmten Public-Key oder Secret-Key meinen oder auf ein spezielles Schlüsselpaar verweisen. Ebenso ist es möglich damit ein OpenPGP-Zertifikat zu bezeichnen. Das OpenPGP-Zertifikat ist eine Zusammenstellung von Informationen, die mit einem Schlüssel oder einer Menge von Schlüsseln verknüpft sind. Eine detailliertere Beschreibung findet sich in „Anatomy of a GPG Key“ [17]. Wichtig zu erwähnen, dass ein solches Zertifikat mindestens aus einem Primary Key besteht, dass das Zertifikat mehrere Subkeys und User-IDs haben kann und jeder Subkey seine eigene Key-ID hat. Beispiele sind im Anhang beigefügt.

Fingerabdrücke nach der OpenPGP-Spezifikationen Version 4 sind 160 Bit lange SHA-1 Hashwerte über das Schlüsselmaterial und einigen anderen zugehörigen Daten. Die letzten 64 Bit des Fingerabdrucks bilden die (lange) Key-ID. Ein Fingerabdruck identifiziert den Schlüssel. OpenPGP verwendet die Key-ID um auf Schlüssel zu verweisen.

2.2.3 Weitere spezielle Paket-Typen

Ein Public-Key-Encrypted-Session-Key-Paket (Tag 1) enthält den verschlüsselten Session Key, mit dem die Daten verschlüsselt wurden. Die verschlüsselte Nachricht folgt dem oder der Sequenz von Public-Key-Encrypted-Session-Key-Paketen in einem Symmetrically-Encrypted-Data-Paket (Tag 9).

Der Inhalt des Paketes besteht gewöhnlicherweise aus einem Compressed-Data-Paket (Tag 8) [5; Kapitel 5.6.]. Dieses wiederum enthält ein Literal-Data-Paket (Tag 11). Da ohne den zum öffentlichen Schlüssel des PKESK-Paketes zugehörigen geheimen Schlüssel ein Zugriff auf die verschlüsselte Nachricht nicht möglich ist, soll es nur bei der Erwähnung dieser Pakettypen bleiben.

Dennoch sollen zwei weitere Pakete genannt werden. Das Symmetrically-Encrypted-Integrity-Protected-Data-Paket (Tag 18), was einem PKESK-Paket oder einer Sequenz davon folgen kann, ist eine Variante des Symmetrically-Encrypted-Data-Paket (Tag 9). In Kombination mit dem Modification-Detection-Code-Paket (Tag 19) lassen sich so Modifikationen an den verschlüsselten Daten feststellen. Dieses MDC-Paket wird ausschließlich im Symmetrically-Encrypted-Integrity-Protected-Data-Paket benutzt [5; Kapitel 5.14.]. In anderen Paket-Typen ist eine Nutzung des MDC-Paketes nicht vorgesehen.

2.2.4 ASCII-Hülle

Nachdem eine OpenPGP-Implementation eine verschlüsselte Nachricht aus ihren einzelnen Paketen zusammengesetzt hat, folgt ein weiterer Schritt, der hier skizziert wird.

Um Kapitel 2.3 vorweg zugreifen wird die OpenPGP-Nachricht für den Transport mittels E-Mail mit einer ASCII-Hülle gesichert, da Binärdaten in der E-Mail wegen der Spezifikationen der RFC 5322 und RFC 5321 nicht erlaubt sind. Die ASCII-Hülle entsteht durch Anwendung der RADIX-64-Kodierung auf die Binärdaten der OpenPGP-Nachricht. Die RADIX-64-Kodierung ist bis auf die angehängte Prüfsumme identisch mit der Base64-Kodierung aus Kapitel 2.1.2.2.2. Die Prüfsumme ist ein 24-bit Cyclic Redundancy Check (CRC), der mittels RADIX-64-Kodierung in 4 Zeichen konvertiert wird und dem Base64-Block mit einem vorangestellten Gleichheitszeichen „=“ angehängt wird. Die Check-summe wird über die binäre OpenPGP-Nachricht gebildet.

In [5; Kapitel 6.2.] wird der Aufbau einer ASCII-Armored OpenPGP-Nachricht definiert und besteht aus:

- *„An Armor Header Line, appropriate for the type of data*
- *Armor Headers*
- *A blank (zero-length, or containing only whitespace) line*
- *The ASCII-Armored data*
- *An Armor Checksum*
- *The Armor Tail, which depends on the Armor Header Line“*

Der in der Armor Header Line gewählte Text ist abhängig von den Daten, die in der ASCII-Hülle kodiert sind und wie sie kodiert wurden. Die Header Line kann folgende Strings beinhalten:

1. *„BEGIN PGP MESSAGE*
   Used for signed, encrypted, or compressed files.

2. *BEGIN PGP PUBLIC KEY BLOCK*
   Used for armoring public keys.

3. *BEGIN PGP PRIVATE KEY BLOCK*
   Used for armoring private keys.

4. *BEGIN PGP MESSAGE, PART X/Y*
   Used for multi-part messages, where the armor is split amongst Y parts, and this is the Xth part out of Y.

5. *BEGIN PGP MESSAGE, PART X*
   Used for multi-part messages, where this is the Xth part of an unspecified number of parts. Requires the MESSAGE-ID Armor Header to be used.

6. *BEGIN PGP SIGNATURE*
   Used for detached signatures, OpenPGP/MIME signatures, and cleartext signatures. Note that PGP 2.x uses BEGIN PGP MESSAGE for detached signatures.“

Beispiel einer ASCII-Armored PGP-Nachricht:

```
-----BEGIN PGP MESSAGE-----

hQEMAxNRFhMUFAHVAQgApSwAtLIDtpjQplCeJ2qYBlbpPIwCZn9Zud0R+k7G2fPt
0fNjzf0nS+t3xibbhFe3uOoo9F6NRCzFIZBZdPSAtmekWVqPlBjVCzk68czf0wDR
g/3dVTSsRUwAjhG8ImqwB61sZea0hvz+7Qw6qFxrReB3/pzycMGEx/T4i2GnTOP/
[...]
cH1EPag9efsX19YoVIpLxaJjFwzOGllMKVICh/T1iSypUyZaCsR5G2rrXwIt7bh1
n40Vov9PmROxCbFLmfmNLs/Q8EAtMcuvxQtMTz2zS3Uoa1m/x7aWQ+225XL7+aXU
GbbmGHf6dwsksyhPLP0=
=5jiU
-----END PGP MESSAGE-----
```

Offensichtlich sind in diesem Beispiel keine Daten zum Armor Header erstellt worden.

Wie später im praktischen Teil ersichtlich, wird auf OpenPGP-Nachrichten mit dem String „BEGIN PGP MESSAGE“ in der Armor Header Line eingegangen. Konflikte mit Implementierungen von PGP 2.x könnten erwartet werden, da sie Signaturen ebenfalls mit diesem String in der Header Line betiteln [5; Kapitel 6.2.].

2.3 MIME Security with OpenPGP (RFC 3156)

Die vorhergehenden Kapitel 2.1 und 2.2 erläutern den strukturellen Aufbau einer E-Mail einerseits und den einer OpenPGP-Nachricht andererseits mit Hinweis auf ihr zugrundeliegender Standards der IETF.

Die 1995 in der RFC 1847 mit dem Titel ″Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted″ eingeführten Medientypen `multipart/signed` und `multipart/encrypted` erweitern die grundlegenden MIME-Spezifikationen, um bestimmte Sicherheitsprotokolle zu unterstützen. Beide Medientypen unterteilen den MIME-Body in zwei Teile.

- `Multipart/signed` bietet Unterstützung für Authentizität und Integrität mittels digitaler Signaturen. Die Kontrollinformationen werden im zweiten Teil des Bodys übertragen. Die erforderlichen Parameter sind:
  - `boundary`,
  - `protocol` und
  - `micalg`.

- `Multipart/encrypted` unterstützt Vertraulichkeit durch Verschlüsselung. Kontrollinformationen werden hier im ersten Teil des Bodys transportiert. Die notwendigen Parameter sind:
  - `boundary` und
  - `protocol`.

Die RFC 2015 ″MIME Security with Pretty Good Privacy (PGP)″ integrierte 1996 diese beiden Medientypen, um PGP-verschlüsselte Nachrichten per E-Mail versenden zu können. Schwächen der RFC 2015 führten in der RFC-Historie zu einer Überarbeitung des Standards, der die Bedürfnisse des vorhandenen OpenPGP-Standards berücksichtigt. So löste 2001 die RFC 3156 die RFC 2015 ab.

OpenPGP-Implementationen können durch verschlüsseln, signieren oder dem Extrahieren von Public-Key-Daten zwei Varianten von Output erzeugen. Entweder ist der Output 8-bit binär oder er ist mit einer ASCII-Hülle versehen.

Um mit der Vorgabe der RFC 5322 (Internet Message Format), die bestimmt, dass übertragene Nachrichten nur aus Zeichen des 7-bit ASCII-Zeichensatzes bestehen dürfen[5], konform zu gehen, legt [18; Kapitel 2.] fest:

"The ASCII armor output is the REQUIRED method for data transfer."

Dazu werden in der RFC drei Medientypen definiert:

- `application/pgp-encrypted`,
- `application/pgp-signed` und

[5] Die RFC 2045, RFC 2046, RFC 2047, RFC 2049, RFC 4288 und RFC 4289 erweitern die RFC 5322 in dem sie auch Werte außerhalb des ASCII-Zeichensatzes erlauben.

- `application/pgp-keys`.

Fokus dieser Arbeit wird `application/pgp-encrypted` sein, da Key-IDs der Empfänger und des Absenders einer PGP-verschlüsselten E-Mail in diesen übertragen werden.

Kapitel 4. der RFC 3156 legt hier fest, dass der `multipart/encrypted` MIME-Body aus zwei Body-Teilen bestehen muss. Der erste Teil enthält den Content Type „`application/pgp-encrypted`“ mit den Kontrollinformationen. Eine standardkonforme Nachricht muss demnach ein Versionsfeld enthalten, der Wert: „Version: 1“.

Der zweite MIME-Body beinhaltet die verschlüsselten Daten, die mit dem Content Type „`application/octet-stream`“ beschriftet sind. Im Beispiel fällt ein weiteres MIME-Header Field auf: „`Content-Disposition`“. 1997 wurde es in [19] definiert um Präsentationsinformationen zu übertragen. Es ist optional und kann in jeder MIME-Entität verwendet werden. Fehlt es, kann der Mailclient eine Präsentation wählen, die er für angemessen hält. Zwei Typen legt die Spezifikation fest:

- `Inline`
- `Attachment`

`Inline`

Wird verwendet, wenn die Nachricht automatisch bei Erscheinen der Nachricht angezeigt werden soll. In der Reihenfolge wie die multipart-Nachricht semantisch strukturiert ist.

`Attachment`

Dieser Typ indiziert, dass der Body-Teil nicht automatisch angezeigt wird. In multipart-Body-Teilen ist für eine Anzeige zwingend eine Aktion des Anwenders erforderlich [19; Kapitel 2.9.]. Dieser Teil wird gewöhnlich als Anhang, separiert vom Hauptteil der Nachricht, präsentiert.

Zu den Typen wurden Parameter vorgestellt. Der Filename-Parameter ist ein Vorschlag des Senders der Nachricht, wenn der Mailclient des Empfängers die Nachricht extrahiert und speichert.

```
To: Bob@daenicke.org
From: Alice <alice@daenicke.org>
Subject: A2BBlindE
Date: Mon, 5 Mar 2018 11:32:16 +0100
MIME-Version: 1.0
Content-Type: multipart/encrypted;
 protocol="application/pgp-encrypted";
 boundary="7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J"

This is an OpenPGP/MIME encrypted message (RFC 4880 and 3156)
--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J
Content-Type: application/pgp-encrypted
Content-Description: PGP/MIME version identification

Version: 1

--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J
Content-Type: application/octet-stream; name="encrypted.asc"
Content-Description: OpenPGP encrypted message
Content-Disposition: inline; filename="encrypted.asc"

-----BEGIN PGP MESSAGE-----

hQEMAxNRFhMUFAHVAQgApSwAtLIDtpjQplCeJ2qYBlbpPIwCZn9Zud0R+k7G2fPt
0fNjzf0nS+t3xibbhFe3uOoo9F6NRCzFIZBZdPSAtmekWVqPlBjVCzk68czf0wDR
g/3dVTSsRUwAjhG8ImqwB61sZea0hvz+7Qw6qFxrReB3/pzycMGEx/T4i2GnTOP/
[...]
cH1EPag9efsX19YoVIpLxaJjFwzOGllMKVICh/T1iSypUyZaCsR5G2rrXwIt7bh1
n40Vov9PmROxCbFLmfmNLs/Q8EAtMcuvxQtMTz2zS3Uoa1m/x7aWQ+225XL7+aXU
GbbmGHf6dwsksyhPLP0=
=5jiU
-----END PGP MESSAGE-----

--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J--
```

Text 1: Beispielquellcode einer PGP-verschlüsselten E-Mail

3 Testszenario

„Data Is a Toxic Asset, So Why Not Throw It Out?“
- Bruce Schneier in [20]

Dieses Kapitel beschreibt die Tätigkeiten, die zur Überprüfung, der im 2. Kapitel erläuterten Standards [11], [12], [13], [14], [5], [18] durchgeführt wurden und wird die Frage beantworten, ob eine Veränderung einer gemäß RFC 3156 strukturierten Mail von Mozilla Thunderbird mit Enigmail-Addon erkannt wird.

Der Ablauf ist ansatzweise wie folgt umrissen:

- diverse E-Mails erstellen
- Datenverkehr beim Versenden der E-Mails mitschneiden und speichern
- Quellcode der Mails speichern
- Quellcodes mit Mitschnitten der Übertragung vergleichen
- OpenPGP-Nachricht aus einer Quelle extrahieren
- deren ASCII-Hülle entfernen
- mit einem Binäreditor unnötiges ESK-Paket[6] ausschneiden
- neue binäre OpenPGP-Nachricht speichern
- diese Nachricht RADIX-64 kodieren, ASCII-Hülle erzeugen
- dieses Päckchen anstelle der alten OpenPGP-Nachricht in die E-Mail einfügen
- E-Mail überprüfen

3.1 Vorbereitungen

Folgende Vorbereitungen zur Analyse des Mailverkehrs beim Versenden von E-Mails zwischen einem Mailclient und einem Mailserver wurden durchgeführt.

Mailserverseitig:

Einrichtung von drei echten Mailadressen alice@daenicke.org, bob@daenicke.org und eve@daenicke.org. Jede Adresse hat ihre eigene Identität bzw. Nutzernamen und Passwort, es sind keine Aliasadressen.

Clientseitig:

Im ersten Anlauf wurde ein Windows-Laptop benutzt. Da es Schwierigkeiten mit dem Grafiktreiber gab wurde das Testszenario auf Arch Linux umgestellt. Erneute Vorbereitungen beinhaltete die Installation von Arch Linux auf diesem Laptop mit Netzwerkschnittstelle. Es wurde überprüft und sichergestellt, dass folgende Programme und deren Abhängigkeiten installiert sind:

- Mozilla Thunderbird Version 52.2.1 als E-Mailclient.

[6] Die zur Verschlüsselung benutzten öffentlichen Schlüssel sollten nicht von Enigmail „versteckt“ worden sein. Enigmail ersetzt in diesem Fall die acht Oktette lange Key-ID durch Nullen.

- Enigmail initial zum Test in der Version 1.9.9 als Addon für Mozilla Thunderbird. Nach einem automatischen Update des Plugins als Version 2.0.7[7]. Es ist eine Benutzeroberfläche für GnuPG und integriert die OpenPGP-Verschlüsselung und Authentifizierung in Mozilla Thunderbird und dessen Derivat Seamonkey.
- `gpg` GnuPG Version 2.2.5 als freies Kryptographiesystem, mit der libgcrypt 1.8.2.
- Wireshark Version 2.2.7 ist ein Netzwerk Protokoll Analyseprogramm.
- Dhex Version 0.68 ist ein Kommandozeilenprogramm zum Vergleich von mehreren Binärdateien.
- Ghex Version 3.18.3 ist ein grafischer Hexeditor.
- Mousepad Version 0.4.0 ist ein Texteditor für die Xfce-Arbeitsumgebung.
- `mcedit` GNU Midnight Commander Version 4.8.19 und nano Version 2.8.5 sind zwei Kommandozeilentexteditoren.
- `pgpdump` Version 0.33 ist ein Paketvisualisierer für OpenPGP-Pakete (RFC 4880) und PGP-Pakete Version 2 (RFC 1991).

Weiterhin wurden die drei zuvor auf dem Mailserver eingerichteten E-Mailadressen als E-Mail-Konten im Mailclient Mozilla Thunderbird mit Enigmail-Addon erstellt. Für jede E-Mailadresse wurde ein Schlüsselpaar erzeugt. Die Key-IDs sind in der Tabelle zu finden.

Name	**Key-ID**
Alice	EE 27 18 AE 9D 41 79 41
Bob	13 51 16 13 14 14 01 D5
Eve	80 3A 04 C1 44 4E EB 6B

3.2 Testmails

Verschiedene Testmails wurden mit Hilfe des Mailclients Mozilla Thunderbird generiert. Alice fungierte immer als Absender einer Mail, Bob wurde immer im Header-To als Hauptempfänger eingetragen, Eve wurde alternierend im Header-Cc und Header-Bcc gesetzt.

Begonnen wurde mit diversen unverschlüsselten E-Mails ohne Anhang und unverschlüsselte Mails mit einer Dummydatei als Anhang. Vor dem Versenden der Nachrichten wurde eine Capture-Sitzung mit dem Netzwerkprotokollanalyseprogramm Wireshark zum Belauschen des entstehenden Netzwerkverkehrs der aktiven Netzwerkschnittstelle. Auf eine Transportverschlüsselung der SMTP-Verbindung wurde verzichtet, um die übertragenen Daten im Klartext zu erhalten.

Schema unverschlüsselt:

Nummer	**Eve**	**Anhang**	**Subject**
1	Cc	Nein	A2BCcE
2	Bcc	Nein	A2BBccE

[7] Ein Downgrade auf die vorherige Version 1.99 erschien unnötig. Am OpenPGP-Standard hatte sich zwischenzeitlich nichts geändert.

Nummer	Eve	Anhang	Subject
3	Cc	Ja	A2BCcE+Att
4	Bcc	Ja	A2BBccE+Att

Diesem Muster folgend in verschlüsselter Form für alle Empfänger sowohl ohne als auch mit einer Dummydatei als Anhang. Das Versenden der E-Mails wurde ebenfalls mit Wireshark mitgeschnitten.

Um später die Frage beantworten zu können, ob eine Veränderung der OpenPGP-Nachricht auch bei verschlüsselten und signierten Nachrichten erkannt wird, wurde bei den E-Mails № 9 bis № 12 Enigmail angewiesen die Nachricht zu verschlüsseln und zu signieren.

Schema verschlüsselt:

Nummer	Eve	Anhang	Signatur	Subject
5	Cc	Nein	Nein	A2BCcE+enc
6	Bcc	Nein	Nein	A2BBccE+enc
7	Cc	Ja	Nein	A2BCcE+Att+enc
8	Bcc	Ja	Nein	A2BBccE+Att+enc
9	Cc	Nein	Ja	A2BCcE+enc+sig
10	Bcc	Nein	Ja	A2BBccE+enc+sig
11	Cc	Ja	Ja	A2BCcE+Att+enc+sig
12	Bcc	Ja	Ja	A2BBccE+Att+enc+sig

In der folgenden Tabelle sind jene E-Mails aufgeführt, bei denen Enigmail vorschlug die Key-ID vom Bcc-Empfänger zu verstecken.

Schema Key-ID vom Bcc versteckt:

Nummer	Anhang	Signatur	Subject
13	Nein	Nein	A2BBccE+enc+hiddenKey
14	Ja	Nein	A2BBccE+Att+enc+hiddenKey
15	Nein	Ja	A2BBccE+enc+sig+hiddenKey
16	Ja	Ja	A2BBccE+Att+enc+sig+hiddenKey

3.2.1 Transportmitschnitt vs. Mailclientquelle

Folgendes Beispiel ist ein Auszug der Quelle aus Alices Ordner gesendeter Nachrichten nachdem eine E-Mail mit Anhang an Bob (Header Field `To:`) und Eve (Header Field `Bcc:`) verschickt wurde. Enigmail sollte diese Nachricht verschlüsseln und signieren. Diese Quelle wird jetzt mit dem Wiresharkmitschnitt auf dem beigelegten Datenträger (`Wiresharkmitschnitte/Versand-verschlüsselt+signiert.pcapng`) verglichen.

```
X-Mozilla-Status: 0001
X-Mozilla-Status2: 00800000
X-Mozilla-Keys:
BCC: eve@daenicke.org
From: Alice <alice@daenicke.org>
Subject: A2BBccE+Att+enc+sig
To: Bob@daenicke.org
Openpgp: preference=signencrypt
Autocrypt: addr=alice@daenicke.org; prefer-encrypt=mutual; keydata=
 xsFNBFmkHyUBEADSX4p4kMW2GXenk5uI5+hHRzkz1jklgf/T+f+o95JJLN+eKxgbI/CHKMuX
 2zP/tqUSljp51KNf2OYWTrRVKKKNq36i0SKMlokOiJO6VldUe2TkwdmkhotHdnptRpnYuDB7
 [...]
 y0ekIgK2JF4yYwd/9T9EtYpv8HAjSWpfDC3NvhsM9Tog6eBmZGg+5ccfTxGcEpAQOwG+eper
 po8mnfl7dpoP4P+cpVLkv9s4X55Vhiu5T52bTllEnU4V+FXm1+Xt1USJx0Y=
Message-ID: <90f92f16-7400-a6ad-fe82-f7de232542dc@daenicke.org>
Date: Mon, 18 Jun 2018 17:12:22 +0200
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
 Thunderbird/52.6.0
MIME-Version: 1.0
Content-Type: multipart/encrypted;
 protocol="application/pgp-encrypted";
 boundary="G2M9Ou4D61WpKhMzBnXbCFj9kmTf0nJDi"

This is an OpenPGP/MIME encrypted message (RFC 4880 and 3156)
--G2M9Ou4D61WpKhMzBnXbCFj9kmTf0nJDi
Content-Type: application/pgp-encrypted
Content-Description: PGP/MIME version identification

Version: 1

--G2M9Ou4D61WpKhMzBnXbCFj9kmTf0nJDi
Content-Type: application/octet-stream; name="encrypted.asc"
Content-Description: OpenPGP encrypted message
Content-Disposition: inline; filename="encrypted.asc"

-----BEGIN PGP MESSAGE-----

hQEMAxNRFhMUFAHVAQf/bsu4bGB9t8dlbR7jLmMvCDzEyUCskx01omkTqL0l6gPv
Z7Hj+UXicRbVFH4zrzKHiOx0QBEZIW5KvUpfjInwBlFCIbwxt4IWt8RL46MRPhCg
[...]
jpz45UuHMTgeSILWB+4eZx/Nz2SB64Ws9AUbfRgL+3LIk8FsSz5MknhmY28lxOZG
KAferPpskRl1+tvhVulEqE2os6AcGx0+
=ASlg
-----END PGP MESSAGE-----

--G2M9Ou4D61WpKhMzBnXbCFj9kmTf0nJDi--
```

Text 2: E-Mailquellcodeauszug von Alices gespeicherter Mail

Die SMTP-Content Phase, eingeleitet durch das SMTP-Komando `DATA`, beginnt im Wiresharkmitschnitt an Paketnummer 163 und endet mit dem MTA-Response im Paket 184.

Es ist festzustellen, dass die ersten vier Header Fields der Quelle

```
X-Mozilla-Status:
X-Mozilla-Status2:
X-Mozilla-Keys:
BCC:
```

nicht übertragen wurden. Header Fields beginnend mit dem String „X-“ werden in der RFC 822, Kapitel 4.7.5 „user-defined-field“ definiert. Sie werden seit dem Erscheinen der RFC 2822 als veraltet angesehen. Thunderbird nutzt diese Header Fields zur internen Mailverwaltung und kann so bestimmen, ob eine Nachricht gelesen, beantwortet oder weiterge-

leitet wurde. Das Header Field `Bcc:` wurde in Übereinstimmung mit dem im Kapitel 3.6.3. der RFC 5322 „Internet Message Format“ festgelegten Vorgehensweise nicht mitgesendet.

Die übertragene Nachricht beginnt mit dem Header Field `From:` und endet mit dem <CR><LF> nach dem letzten „-“ der Boundary der MIME-kodierten Nachricht.

3.2.2 Analyse der Mailquelle

In diesem Abschnitt werden die einzelnen Header Fields überprüft, die MIME-Struktur erläutert, sowie die OpenPGP-Nachricht seziert.

3.2.2.1 Herkunft der Header Fields

Folgende Tabelle gibt einen Überblick der übertragenen Header Fields und deren Herkunft aus den zugrundeliegenden Standards.

Header Field	**Definiert in**	**Beschreibung**
`From:`	[12; Kapitel 3.6.2.]	Originator Field, Pflichtfeld, indiziert den Autor der Nachricht
`Subject:`	[12; Kapitel 3.6.5.]	Informational Field, optional[8]
`To:`	[12; Kapitel 3.6.3.]	Destination Address Field, spezifiziert den Empfänger der Nachricht
`Openpgp:`	Internet Draft [21; Kapitel 3.3.]	Von Enigmail 2.x generiertes Header Field. Der „preference“-Parameter signalisiert dem Empfänger das bevorzugte Schutzlevel für Nachrichten an den Sender.
`Autocrypt:`	[22; Kapitel 3.1.]	In diesem Header Field wird das öffentliche Schlüsselmaterial des Absenders hinterlegt.
`Message-ID:`	[12; Kapitel 3.6.4.]	Identification Field, enthält einmaligen Identifikator der E-Mail
`Date:`	[12; Kapitel 3.6.1.]	Origination Date Field, Pflichtfeld, enthält Datum und Uhrzeit zum Zeitpunkt der Fertigstellung der E-Mail. Kann vom tatsächlichen Zeitpunkt des Versendens abweichen.
`User-Agent:`	[12; Kapitel 3.6.8.] [23; Kapitel 5.5.3.]	Optional Field, nach RFC 5322 nicht eindeutig definiert. In der HTTP-Spezifikation der RFC 7231 wird es als Request Header Field bezeichnet und stellt zusätzliche Informationen zum verwendeten Mailclient bereit.
`MIME-Version:`	[13; Kapitel 4.]	MIME-Version Header Field
`Content-Type:`	[13; Kapitel 5.]	Content-Type Header Field

[8] Kann nach Definition mit dem String „Re: “ beginnen, wenn eine Nachricht beantwortet wird. Soll aber in mehreren Instanzen nur einmal verwendet werden, um unerwünschte Konsequenzen zu vermeiden.

Das Header Field `Content-Type:` mit den Parametern `protocol` und `boundary` leitet an dieser Stelle die erste Instanz der MIME-strukturierten Nachricht ein.

3.2.2.2 MIME-Korpus

Der verwendete Medientyp `multipart/encrypted` aus der RFC 1847 weist grundsätzlich auf einen zweiteiligen MIME-Body hin, im ersten Teil stehen die Kontrollinformationen und im zweiten Teil die verschlüsselten Daten als Nutzdaten, getrennt von der Boundary. Der zwingend vorhandene Protokollparameter von `multipart/encrypted` enthält den Medientyp und dessen Subtyp aus dem Content-Type-Feldes des folgenden ersten Body-Teils. Nach dem Boundary-Parameter endet die Header Section der E-Mail gemäß der RFC 5322, durch eine Leerzeile beziehungsweise ein <CR><LF> angezeigt. Betrachtet man die E-Mail nach den Regeln der RFC 5322 „Internet Message Format“ erstreckt sich die MIME-Nachricht über den Header und dem Body der E-Mail.

Diese beiden Teile wurden vom Thunderbird-Enigmail-Duo konform zu den Spezifikationen der RFC 4880 und RFC 3156 zusammengestellt. Nicht PGP-fähige Mailclients würden zwei Anhänge anzeigen.
Zur Veranschaulichung hier der Korpus der PGP/MIME-strukturierten Nachricht.

```
└┬multipart/encrypted
 ├──application/pgp-encrypted
 └──application/octet-stream
```

Der Korpus einer in den offenen Fragen benannten PGP/Inline-strukturierten Nachricht kann folgendermaßen verdeutlicht werden:

```
└──text/plain
```

3.2.2.2.1 Kontrollinformationen in application/pgp-encrypted

Wie in Kapitel 2.3 erläutert, ist der erste Body-Teil einer `multipart/encrypted` MIME-Entität mit Kontrollinformationen gefüllt. Im `Content-Type:`-Feld beinhaltet den erforderlichen Medientyp und dessen Subtyp, wie in der Spezifikation vorgeschrieben. Für verschlüsselte Nachrichten wird `application/pgp-encrypted` gefordert. Das darauf folgende `Content-Description:` als optionales MIME-Header-Feld beschreibt den Inhalt. Das Versions-Feld mit dem Wert „Version: 1“ ist erforderlich und standardkonform vorhanden.

3.2.2.2.2 Nutzdaten in application/octet-stream

Der zweite Body-Teil beginnt mit dem erforderlichen `Content-Type:`-Feld mit der Angabe des Medien- und Subtyps `application/octet-stream`. Als nächstes MIME-Header-Feld folgt die Beschreibung des Body in `Content-Description:`.

Das `Content-Disposition:`-Feld weist mit dem Wert `inline` den Mailclient an, die Nachricht sofort anzuzeigen. Ist eine PGP-fähige Erweiterung im Mailclient vorhanden, muss diese die folgende PGP-Nachricht entschlüsseln. Der Parameter `filename` enthält denselben Wert wie im zwei Zeilen zuvor erschienenen optionalen Vorschlag des Parameters `name` vom `Content-Type:`-Feld.

Es folgt eine ASCII-Armored OpenPGP-Nachricht wie in Kapitel 2.2.4 beschrieben.

3.2.2.3 Vorläufiges Fazit

Bis hierher gibt es erwartungsgemäß keine Hinweise auf die Möglichkeit eine Key-ID zu entfernen, die offenbart für welche Empfänger die Nachricht verschlüsselt wurde.

3.2.3 Analyse der OpenPGP-Nachricht

Der praktische Teil dieser Arbeit fährt mit dem „Dearmoren der OpenPGP-Nachricht“ fort. GnuPGPs Kommandozeilenprogramm `gpg` bietet mit der Option `--dearmor` die, nach Aussage der `man`-Page, „nicht sehr nützliche“ Funktion ASCII-Armored Daten auszupacken beziehungsweise im Binärformat anzuzeigen oder zu speichern an. OpenPGP-Nachrichten im Binärformat lassen sich hingegen mit dem Gegenstück `--enarmor` mit einer ASCII-Hülle versehen.

3.2.3.1 Veranschaulichung der Paketstruktur

Die einzelnen Pakete einer OpenPGP-Nachricht aus Kapitel 2.2 ließen sich mit einem Hexeditor wie Ghex nur mühsam identifizieren. Verwendet man hingegen einen Hexeditor, der Binärdateien vergleichen kann, und benutzt zwei fast identische Nachrichten wie № 11 und № 12 fallen die Gemeinsamkeiten sofort auf.

Auf diese Art mehrere Samples der OpenPGP-Nachrichten aus den übertragenen E-Mails miteinander verglichen, fällt auf, dass verschlüsselte und gleichzeitig signierte Nachrichten strukturell mit den verschlüsselten aber nicht signierten Nachrichten identisch sind.

Für das Programm `pgpdump` müssen die Nachrichten nicht vorher von der ASCII-Hülle befreit werden um die Struktur der Nachricht zu visualisieren.

Die Syntax einer OpenPGP-Nachricht aus Kapitel 2.2.1 folgt diesem Schema:

OpenPGP Message :- Encrypted Message | *Signed Message* |
Compressed Message | *Literal Message.*

Eine verschlüsselte E-Mail enthält die *Encrypted Message*. Für diese gilt folgende syntaktische Regel:

Encrypted Message :- Encrypted Data | ESK Sequence, Encrypted Data.

Für *Encrypted Data* gilt:

Encrypted Data :- Symmetrically Encrypted Data Packet |
Symmetrically Encrypted Integrity Protected Data Packet.

Für *ESK Sequence und ESK* sind diese Regeln maßgeblich:

ESK Sequence :- ESK | ESK Sequence, ESK.
ESK :- Public-Key Encrypted Session Key Packet |
Symmetric-Key Encrypted Session Key Packet.

Durch Substitution der Syntaxelemente erhält man für eine diesem Szenario entsprechende PGP-verschlüsselte OpenPGP-Nachricht folgende Komposition:

OpenPGP Message :- Public-Key Encrypted Session Key Packet,
Public-Key Encrypted Session Key Packet,
Public-Key Encrypted Session Key Packet,
Symmetrically Encrypted Integrity Protected Data Packet.

Basis dieser Regel ist die Annahme, dass der verschlüsselnde Mailclient eine Nachricht auch für den Absender verschlüsselt und die Nachricht zwei Empfänger hat.

Die Encrypted-Session-Key-Pakete sind in diesem Fall Public-Key-Encrypted-Session-Key-Pakete. Ein PKESK enthält nicht nur den mit öffentlichem Schlüssel verschlüsselten Sitzungsschlüssel sondern auch die dazugehörige Key-ID. Die Key-ID ist mit einer E-Mailadresse verknüpft. Die OpenPGP-Zertifikate der öffentlichen Schlüssel sind auf Schlüsselservern im Internet hinterlegbar. Die Schlüsselserver sollen den Schlüsselaustausch unter den Anwendern erleichtern.

Schreibt Alice eine PGP-verschlüsselte E-Mail an mehrere Empfänger, so wird die Nachricht nur einmal mit dem Sitzungsschlüssel verschlüsselt und für jeden einzelnen Empfänger wird der Sitzungsschlüssel mit seinem dazugehörigen öffentlichen Schlüssel verschlüsselt und in ein Paket verpackt. Alle Pakete werden nun aneinandergereiht, das Resultat mit der ASCII-Hülle versehen und konform zu den hier erläuterten Standards zu einer E-Mail gestaltet, die der Mailclient unabhängig von der Anzahl der Empfänger genau einmal auf Reise schickt. Alle Empfänger bekommen dieselbe E-Mail[9]. Jeder Empfänger erlangt Kenntnis über die Anzahl der weiteren Empfänger und, sofern Enigmail nicht angewiesen wurde die Key-IDs zu verstecken, via Schlüsselserverabfrage auch die E-Mailadressen der anderen Empfänger.

Nicht nur das könnte beunruhigen. Auch wissbegierige Mail-Server-Betreiber, die am E-Mailtransfer beteiligt sind, könnte man unterstellen mit einem `gpg --dearmor` umgehen zu können. Den erlangten Wissensstand könnten sie für maßgeschneiderte Werbung, soziale Graphen oder weiteres nutzen.

[9] Die Empfänger bekommen denselben Body der E-Mail. Die Header Fields unterscheiden sich je nach Empfänger und beteiligten MTA.

Nachdem die binäre OpenPGP-Nachricht in der Praxis strukturell „entzaubert“ war, stellte sich die Frage, ob ein Sicherungsmechanismus existiert, der Veränderungen an der ESK-Sequenz erkennt. Zur Klärung dieser Frage bedarf es einen Hexeditor. Die PKESK-Pakete der OpenPGP-Nachricht sind identifiziert, die Länge der Pakete steht in deren Metadaten.

3.2.4 Extraktion eines Public-Key-Encrypted-Session-Key-Paketes

Das Vorgehen im Einzelnen:

1. Bob speichert eine E-Mail als eml-Datei in `/Mailquellen/Bob.sanitize/6.eml`
2. GnuPG entfernt die ASCII-Hülle: `$` **`gpg -dearmor 6.eml`**
3. die Binärnachricht liegt als `6.eml.gpg` vor
4. `6.eml.gpg` in einem Hexeditor (Ghex) laden
5. komplettes PKESK-Paket mit Eves Key-ID identifizieren und ausschneiden
6. die binäre, nun neue, OpenPGP-Nachricht speichern als `6.eml.ohne-eve.gpg`
7. neue OpenPGP-Nachricht armoren:
   `$` **`gpg --enarmor 6.eml.ohne-eve.gpg`**
8. `gpg` speichert Resultat als `6.eml.ohne-eve.gpg.asc`
9. `6.eml.ohne-eve.gpg.asc` in Texteditor laden
10. *ASCII-Armored data* und *Armor Checksum* (Siehe Kapitel 2.2.4) ausschneiden oder kopieren (Strg-C)
11. `6.eml` in einen Texteditor laden
12. die *ASCII-Armored data* und *Armor Checksum* markieren und gegen die Daten aus der Zwischenablage austauschen (Strg-V)
13. diese neue Datei als `6.neueASCII-Armor.eml` speichern
14. `6.neueASCII-Armor.eml` mit Thunderbird/Enigmail öffnen

Sollte ein Mechanismus existieren, der die Säuberung dieser OpenPGP-Nachricht erkennt, wäre an dieser Stelle die Möglichkeit gegeben, den Anwender des Mailclients über eine Integritätsverletzung der vollständigen OpenPGP-Nachricht zu informieren. Eine zugesicherte Integrität erstreckt sich lediglich über das Encrypted-Data-Paket.

Damit ist der praktische Teil dieser Arbeit am Ende.

3.3 Ergebnis

Gezeigt wurde, dass eine Säuberung einer ESK-Sequenz einer PGP-verschlüsselten E-Mail möglich ist. Es ist sogar für einen unbeteiligten Dritten ohne Kenntnis eines Passwortes für ein OpenPGP-Zertifikat realisierbar.

Weder Thunderbird, Enigmail noch GnuPG informieren über die getätigte Bereinigung. Offensichtlich ist kein Mechanismus vorhanden, der Veränderungen an der ESK-Sequenz erschwert, verhindert oder erkennt.

Da eine „händische“ Extraktion eines PKESK-Paketes nach derzeitigem Standard der RFC 4880 wie gezeigt möglich ist, sollte eine automatische Behandlung der diesem Szenario entsprechenden Kommunikation realisierbar sein.

Das führt zu einem Blick Richtung Mailfilter, deren Funktionsweisen im nächsten Kapitel vorgestellt werden.

4 Modellierung eines Mailfilters

4.1 Ziel eines Mailfilters

Ein Mailserver hat die zentralen Aufgaben Mails entgegenzunehmen, sie weiterzuleiten oder zuzustellen. Ausschlaggebend dafür sind die Protokolleinheiten des SMTP-Envelopes. Der SMTP-Content, bestehend aus den Header Fields und dem Body, ist für den Transport der Mail nicht maßgeblich.

Für die Anwender ist es hingegen komfortabel, wenn Drittprogramme E-Mails nach unerwünschten Inhalten durchforsten, markieren oder gleich beim Empfang zurückweisen.

Mailfilter erfüllen ihre Aufgaben sowohl auf Seiten des MTA als auch im MUA. Im MUA können sie beispielsweise E-Mails nach bestimmten Kriterien in einen dafür vorgesehenen Ordner verschieben und somit das Postfach sortieren. Die wichtigsten Ziele der Mailfilter sind die eingelieferten Mails und deren Anhänge auf Schadcode zu überprüfen oder Spam in den Mails zu erkennen und als solche zu markieren, zu isolieren oder abzuweisen.

Dazu sind Änderungen an Header und Body der Mail möglich und notwendig. Ein modifiziertes Subject ist eine Änderung im Header der E-Mail, entfernte oder ersetzte Anhänge sind Änderungen im Body. Enthält der Body Anhänge, so muss bei Änderungen des Inhalts auch die MIME-Struktur überarbeitet werden.

4.2 Mailfiltervarianten

Anhand zweier MTAs werden hier grundsätzliche Verfahren zum Filtern von Mail vorgestellt. Moderne Mailserver bieten die Optionen, die Mails vor oder nach dem Queueing der Mails zu filtern, sie beherrschen das Mailfiltern ebenfalls in Echtzeit zum Zeitpunkt des SMTP-Dialogs. So können Spammails anhand bestimmter Kriterien sofort abgewiesen werden.

4.2.1 Sendmail Milter

Beim Milter, ein Kofferwort gebildet aus **M**ail und F**ilter,** handelt es sich sowohl um ein Protokoll als auch um die Routinen, die es Programmen eines Drittanbieters erlauben auf die Mailnachrichten zuzugreifen um Metainformationen und Inhalte zu filtern [24].

Ein Milter kann die SMTP-Envelope-Protokollelemente, Nachrichtenverbindungsinformationen, Nachrichtenheader und Nachrichtenbody verarbeiten.

Die Sendmail API ″Milter″ unterstützt das Open-Source-Paket von Sendmail seit der Version 8.12.0 [25; S.243]. In der README zu libmilter, enthalten in den Sendmailquellen, liest man, dass Sendmail seit der Version 8.13 standardmäßig den Support für libmilter einkompiliert hat. Die Sendmailalternative Postfix unterstüzt das Milter-Protokoll seit Version 2.3 [25; S.234].

Die Sendmail-library libmilter selbst ist in der Programmiersprache C geschrieben. Zum Parsen von E-Mailnachrichten gibt es jedoch leistungsfähigere Programmiersprachen.

Milter werden als separate Prozesse ausgeführt. Das wirkt sich vorteilhaft auf die Sicherheit und Performanz des Mailservers aus und eventuelle Störungen im Fehlerfall eines Milters beeinflussen nicht den MTA oder andere Filter [24].

Die Kommunikation zwischen dem Milter und dem MTA findet über TCP oder Unix-Domain-Sockets statt, das heißt, sie können auf einer separaten Maschine laufen oder auf derselben wie der MTA.

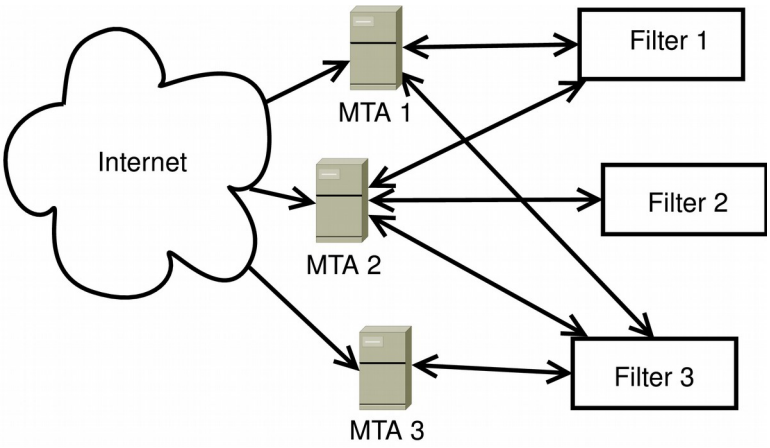

Abbildung 9: Mehrere MTAs kommunizieren mit mehreren Filtern (Quelle: modifiziert nach [24])

Diese Abbildung 9 verdeutlicht, dass mehrere MTAs gleichzeitig über lokale oder entfernte Verbindungen mit einen Filter kommunizieren können.

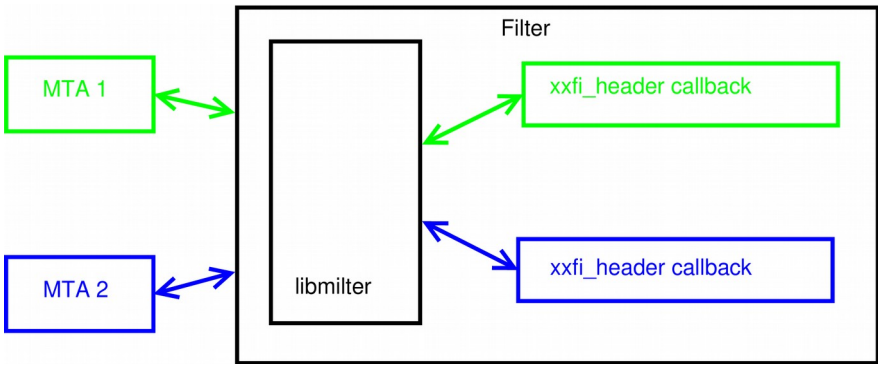

Abbildung 10: Ein Filter kommuniziert mit zwei MTAs (Quelle:modifiziert nach [24])

Obige Grafik aus [24] veranschaulicht, dass ein Filter über die Milterschnittstelle mehrere MTAs bedienen kann.

4.2.1.1 MIMEDefang

MIMEDefang[10] ist ein Open Source Framework zum Filtern von E-Mail. Es greift auf die Sendmail API Milter Schnittstelle (libmilter) zu. Das Framework selbst besteht haupt-

[10] https://mimedefang.org/

sächlich aus in Perl geschriebenen Routinen, in C geschriebener „Glue Code“ lässt die in Perl geschriebenen Mailfilter mit dem MTA kommunizieren. Bemerkenswert an dieser Stelle ist eine Funktionalität namens „Streaming“ [26; S.122] der MIMEDefang-Kollektion. Wird Streaming zu einem Zeitpunkt der Milterarbeit aktiviert, so beendet sich der Milter, da die E-Mail re-übertragen wird. Sie verlässt den Milter und es wird in diesem Fall eine erneute Übertragung pro `RCPT TO:`-Empfänger initiiert. Das könnte nützlich sein, wenn die Empfänger unterschiedliche, weil gesäuberte, E-Mails erhalten sollen. Durch die Re-Übertragung treten anschließend mehrere Kopien in das Sendmail-System wieder ein, die erneut vom Milter zur Säuberung der OpenPGP-Nachricht bearbeitet werden müssen.

4.2.2 Postfix

Im Gegensatz zum monolithisch designten Sendmail wurde Postfix von Wietse Zweitze Venema modular konzipiert. Die hier vorgestellten Postfix-Schemata der Abbildungen 11 und 12 veranschaulichen den Postfix-Eingang und den Postfix-Ausgang.

Die gelben Ellipsen stellen Programmmodule von Postfix dar, die gelben Rechtecke illustrieren die Postfix-Queues.

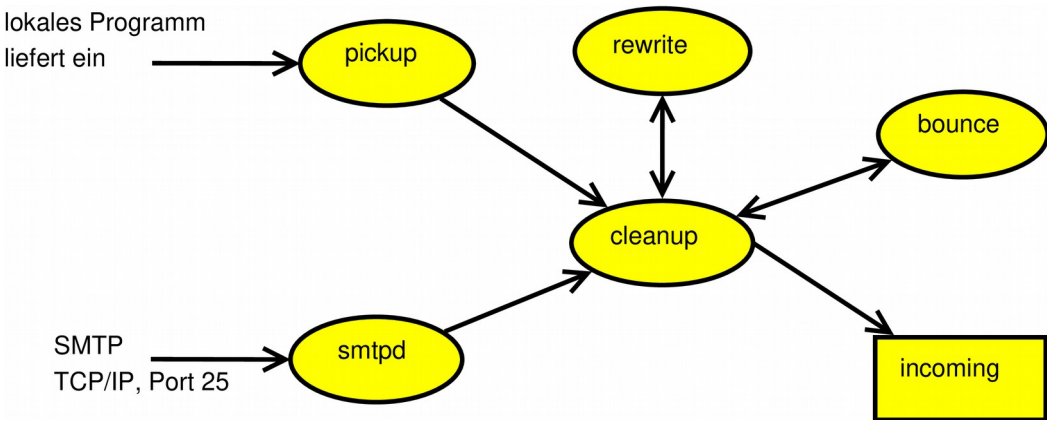

Abbildung 11: Schema des Postfix-Eingangs (Quelle: modifiziert nach [25; S.93])

Die Abbildung 11 zeigt den möglichen Verlauf einer E-Mail beim Empfang durch Postfix. Dabei kann die E-Mail über zwei Wege in Postfix eingeliefert werden.

Lokale Einlieferung:

Das Modul `pickup` holt sich die von lokalen Mailprogrammen über eine Queue mit Namen `maildrop` (hier nicht in der Abbildung) eingelieferte Mail und übergibt sie an `cleanup`.

Einlieferung über SMTP:

Das Modul `smtpd` überwacht den Port 25 und nimmt die Mails an, die von externen Mailclients per SMTP eingeliefert werden. Die zentrale Annahmestelle `cleanup` erhält vom Modul `smtpd` anschließend die eingelieferte E-Mail.

Postfix-Module und deren Funktionen

Folgende Tabelle stellt einen modifizierten Auszug aus [27; S.102] dar.

Modul	Funktion
`master`	Postfix Masterprogramm, koordiniert die anderen Module
`bounce`	unzustellbare Mails werden von diesem Modul behandelt
`cleanup`	Mailheader eingehender Mails werden überprüft und eventuell fehlende Felder werden hinzugefügt
`lmtp`	sendet E-Mails per LMTP
`local`	E-Mails werden in lokale Postfächer gespeichert
`pickup`	Die aus der Queue `maildrop` aufgenommenen Mails werden an `cleanup` weitergereicht
`pipe`	leitet E-Mails an externe Programme
`qmgr`	Verwalter der Warteschlange und des Mailversands
`resolve`	Zielserver und Zustellungsmethode für `RCPT TO:`-Empfänger wird ermittelt
`rewrite`	bereinigt die Empfängeradressen falls nötig, läuft auch an beim Mailaliasing
`smtp`	sendet E-Mails per SMTP
`smtpd`	E-Mail empfangener Dämon

Sind die eingegangenen Mails für die weitere Behandlung vorbereitet legt `cleanup` sie in der Queue `incoming` ab, d.h. sie werden in einem Verzeichnis im Postfix Dateisystem gespeichert.

Die Queue `incoming` ist Ausgangspunkt für die weitere Bearbeitung der Mails im Postfix-System.

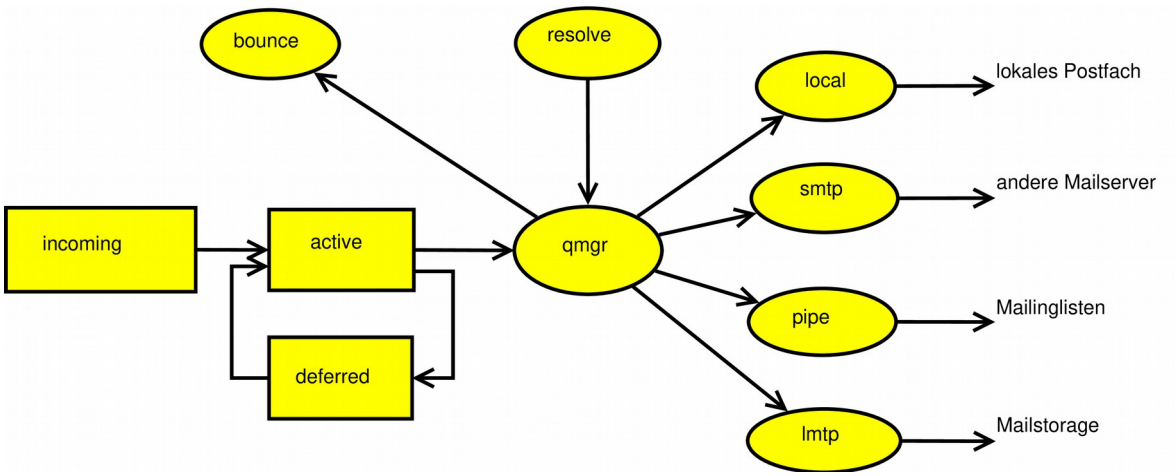

Abbildung 12: Schema des Postfix-Ausgangs (Quelle: modifiziert nach [25; S.97])

Das Modul `qmgr` verschiebt einzelne Mails aus der Queue `incoming` in die `active`-Queue. Sollte es bei der Auslieferung zu Verzögerungen kommen, so werden die Mails zeitweise in der `deferred`-Queue gespeichert. Mails, die nicht zugestellt werden können, werden an den Absender „gebounct“.

4.2.2.1 Postfix Filtermechanismen

Ein Postfix-Mailserver bietet mehrere Möglichkeiten zum Zugriff auf vom Filter zu verarbeitende Mails an:

- Interne Content-Filter
- Externe Content-Filter

4.2.2.1.1 Interne Content-Filter

Im Kapitel 11 aus [25] erläutern die Autoren wie die in Postfix eingebauten internen Filtermechanismen `*_checks` und `*_restrictions` funktionieren. Mittels `*_checks` werden die Daten, die in der SMTP-Content-Phase übertragen werden durch reguläre Ausdrücke überprüft. Die Restriktionen analysieren hingegen die Daten des übrigen SMTP-Dialogs. Die `*_checks` nehmen einfache Aufgaben wahr und können abhängig vom Resultat der Überprüfung die Nachrichten abweisen, umleiten, verwerfen oder an einen mächtigeren externen Filter schicken.

4.2.2.1.2 Externe Content-Filter

Kapitel 13 aus [25] beschreibt wie externe Content-Filter in Postfix funktionieren und zählt drei Mechanismen auf:

- `content_filter`
- `smtpd_proxy_filter`
- `smtpd_milters`

Der Weg der eingelieferten E-Mail in der `content_filter`-Variante ist in der Abbildung 13 illustriert.

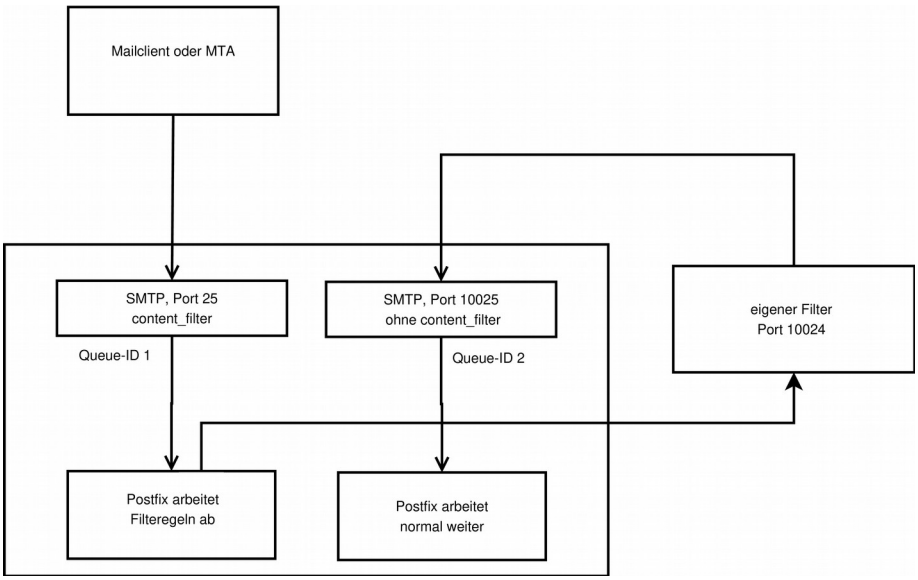

Abbildung 13: Schema eines Postfix content_filters (Quelle: modifiziert nach [27; S.480])

Die E-Mail wird von Postfix angenommen und in eine Queue gestellt, der Queuemanager entscheidet anhand der Transportregeln respektive Filterregeln über den weiteren Weg der E-Mail. Hier wird sie zum externen „eigenen Filter“ gesendet. Der Filter behandelt die E-Mail und sendet das Ergebnis zurück zu Postfix, der einen weiteren SMTP-Port zur Zyklenvermeidung ohne `content_filter` zur Verfügung stellen muss. Zur Absicherung gegen Spam sollte der Port 10025 mittels geeigneter Mechanismen nur vom Filter erreichbar sein.

In der `smtpd_proxy_filter`-Variante der Abbildung 14 bestimmen die Transportregeln, dass die E-Mail bei Einlieferung zum Port des eigenen Filters weitergeleitet wird.

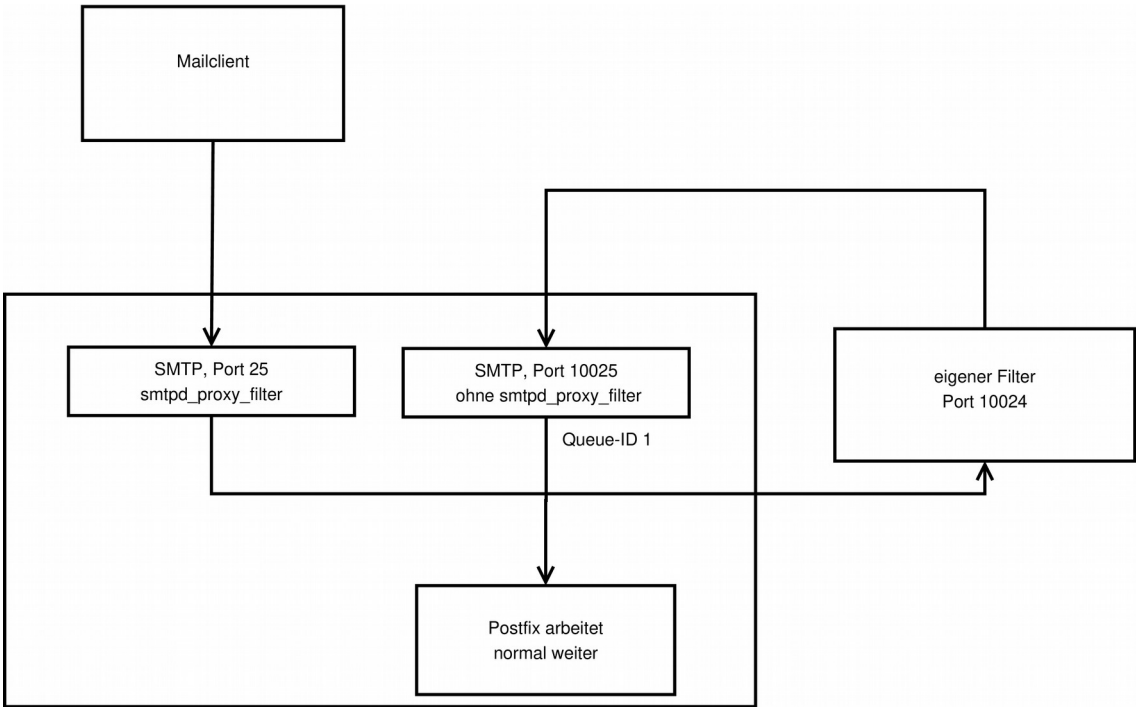

Abbildung 14: Schema eines Postfix smtpd_proxy_filters (Quelle: modifiziert nach [27; S.481])

Analog zu der `content_filter`-Variante aus Abbildung 13 liefert der eigene Filter aus Abbildung 14 das Ergebnis nach Fertigstellung zu einer Postfix-Instanz auf einem anderen Port aus. Die `smtpd_proxy_filter`-Variante behandelt die E-Mail bevor sie in die Queue eingestellt wird, während der `content_filter`-Mechanismus nach dem Queueing der Mail startet.

Um kompatibel mit bereits existierenden für Sendmail entwickelten Miltern zu sein unterstützt Postfix das Milterprotokoll aus Kapitel 4.2.1. Eine Milterreadme[11] unterscheidet zwischen SMTP-only Milter und non-SMTP Milter. Postfix präsentiert für beide Varianten Schnittstellen in Form von zwei Transportmechanismen:

- `smtpd_milters`
- `non_smtpd_milters`

[11] http://www.postfix.org/MILTER_README.html

Die Autoren geben in [25; S.244] an, dass SMTP-Only Milter auf E-Mails angewendet werden, die über den `smtpd`-Daemon Postfix betreten haben. Non-SMTP Milter behandeln E-Mails, die nicht über eine SMTP-Session Postfix betreten haben, etwa über das Postfix-`sendmail`-Kommando.

4.3 Funktionsweise eines neuen Filters

In diesem Abschnitt werden zwei Filtervarianten vorgestellt. Eine Variante säubert die vom Mailclient eingelieferten E-Mails in einer Sendmailumgebung und die andere für einen postfixbasierten Mailserver. Filter für einen MTA, der E-Mails von einem Mailrelay empfängt, werden hier nicht vorgestellt. Diese Filter sind entsprechend der Szenarien aus Abbildung 17 anzupassen.

4.3.1 SMTP-only Milter in einer Sendmailumgebung

Simpler Ablaufplan eines Sendmail SMTP-only Milter:

1. Die E-Mail trifft auf den Milter.
2. Sendet der Mailclient eine OpenPGP-Nachricht? Ein Parser sucht in der MIME-Entität wird nach den Strings `multipart/encrypted`, `application/pgp-encrypted`, `application/octet-stream` und der Armor Header Line mit dem String „-----BEGIN PGP MESSAGE-----“. Das sind die Identifikatoren einer PGP/MIME-Nachricht.
   Antwort Nein: Nachricht ist nicht verschlüsselt, weiter mit Schritt 11.
   Antwort Ja: Weiter mit Schritt 3.
3. Sind mehr `RCPT TO:`-Adressen generiert worden als in den Header Fields `To:` und `Cc:` Adressen enthalten sind? Das ist ein Hinweis auf Bcc-Empfänger, wenn der Mailclient die E-Mail beim MTA zustellt.
   Antwort Ja: Weiter zu Schritt 4.
   Antwort Nein: Weiter zu Schritt 6.
4. Speichern der Armor Checksum als Identifikator für Schritt 6.
5. MIMEDefangs Streaming einschalten. Für jede `RCPT TO:`-Adresse wird eine Kopie der E-Mail generiert. Die Filterfunktion endet hier. Nach Wiedereintritt mit Schritt 1 fortfahren.
6. Bei unbekanntem ASCII-Armor Checksum weiter mit 11.
7. Aus jeder einzelnen Mail wird die OpenPGP-Message extrahiert und dearmored. OpenPGP-Nachricht liegt jetzt im Binärformat vor.
8. Milterinstanzen, die E-Mails mit Adressen aus den Header Fields `To:` und `Cc:` bearbeiten, extrahieren aus der ESK-Sequenz die PKESK-Pakete der Bcc-Adressen und ersetzen die alte ESK-Sequenz durch die neue ESK-Sequenz.
   Milterinstanzen, die E-Mails der Bcc-Adressen bearbeiten, entfernen jene PKESK-Pakete der Bcc-Adressen aus der ESK-Folge, für die sie nicht arbeiten und lassen

die eigene in der ESK-Folge stehen und ersetzen die alte ESK-Folge durch die neue ESK-Folge.

9. Die neue OpenPGP-Nachricht wird RADIX-64 kodiert.
10. Die neue OpenPGP-Nachricht ersetzt die alte OpenPGP-Nachricht.
11. Die E-Mails werden vom Milter zum MTA zur weiteren Bearbeitung zurück geschickt.

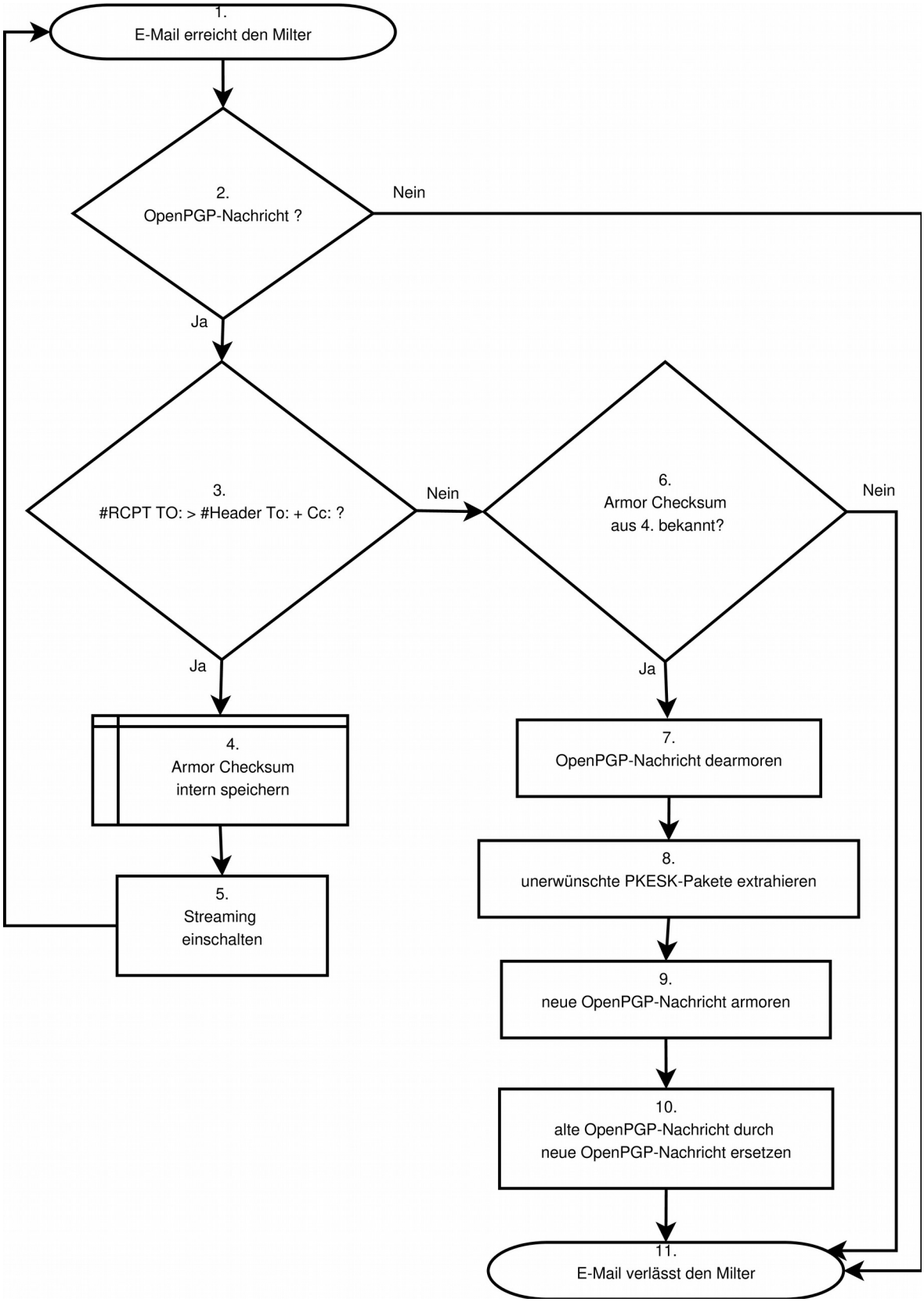

Abbildung 15: Simples Milter-Ablaufdiagramm

Zu Schritt 6 sei angemerkt, dass eine E-Mail entweder genauso viele `RCPT TO:`-Adressen hat wie in den Header Fields `To:` und `Cc:` enthalten sind, dann dürften keine Bcc-Adressen vorhanden sein und die E-Mail kann den Milter unbearbeitet verlassen, oder die E-Mail kam von einem Mailrelay. In der Abbildung 16 ist MTA 1 derjenige MTA, der die E-Mails vom Mailclienten direkt annimmt. Die Säuberung in einem anderen Szenario, wie derer vom MTA 2 oder MTA 3, muss anders erfolgen. Ein einfaches Zählen der jeweiligen Adressen in den Adressfeldern reicht in diesen beiden Szenarien nicht aus. Milter für MTA 2 und MTA 3 müssen die `RCPT TO:`-Adressen mit den tatsächlichen Adressen der Header Fields `To:` und `Cc:` vergleichen und dann bestimmen, ob eine Säuberung nötig ist.

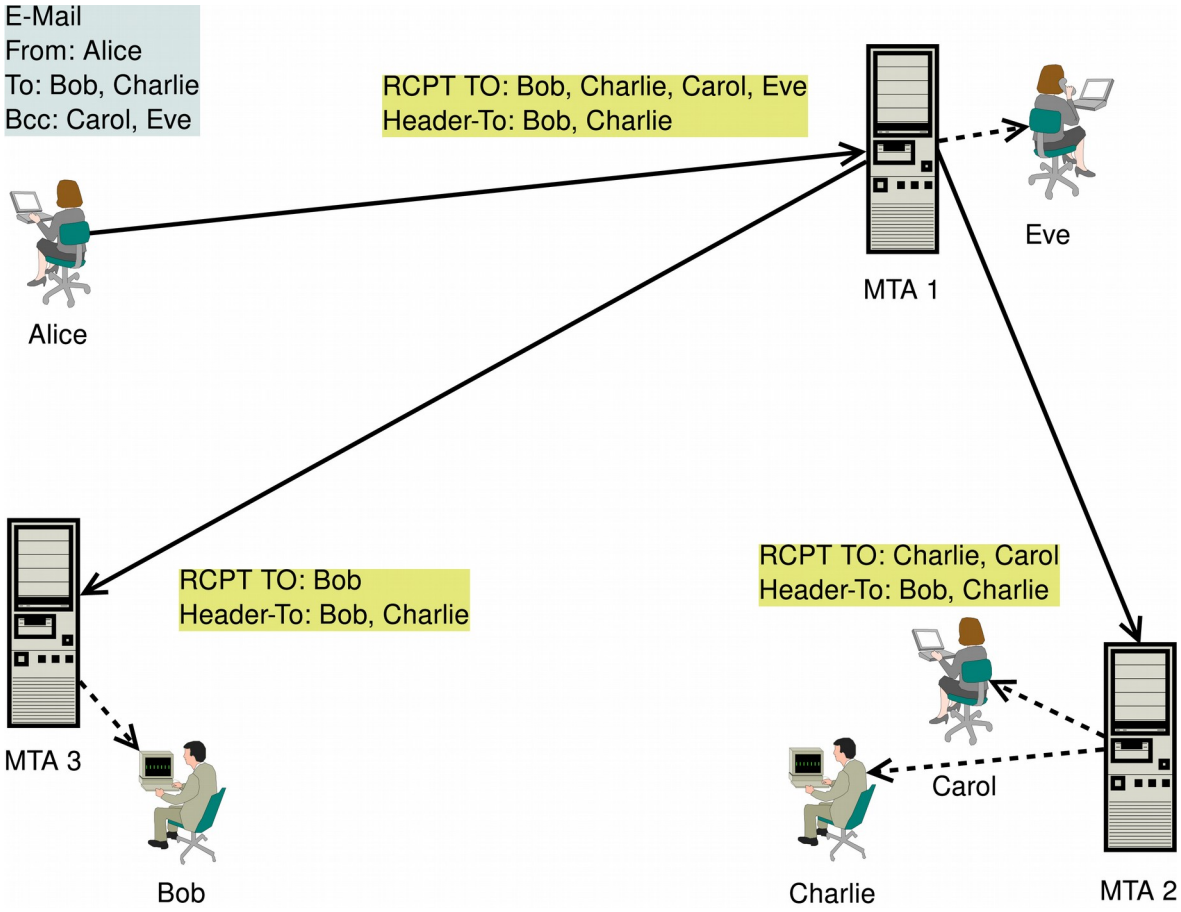

Abbildung 16: RCPT TO vs. Header-To: (Quelle: modifiziert nach [10])

Im Schritt 8 ist es notwendig, die PKESK-Pakete zu identifizieren. Ein PKESK-Paket beinhaltet die Key-ID eines Empfängers und nicht dessen E-Mailadresse. Ein möglicher Weg ist einen öffentlich verfügbaren Keyserver zu fragen. Dieses kann jedoch eine Firewall wirkungsvoll verhindern. Da eine Key-ID mit mehreren E-Mailadressen verknüpft werden kann, können unterschiedliche Keyserver verschiedene Angaben zur Key-ID machen und eventuell die nicht gültigen E-Mailadressen zurückgeben. Sinnvoll wäre in diesem Entwurf eine eigene lokale Datenbank mit den Key-IDs und deren zugehörigen E-Mailadressen zu betreiben und zu verwalten. Der Milter fragt dann diese Datenbank ab. Die libmilter über-

gibt die `RCPT TO:`-Adressen einer Milterinstanz. So „weiß“ ein Milter aus Schritt 8 für wen die E-Mail bestimmt ist und kann das unerwünschte PKESK-Paket extrahieren.

4.3.2 Ein Postfix content_filter

Da Postfix mehrere Möglichkeiten zur Filterung von E-Mails anbietet sei hier eine Variante eines `content_filter` aus Abbildung 13 vorgestellt. Der Queuemanager sendet entsprechend der Postfixfilterregeln die E-Mail an den Filter, der sie bearbeitet und nach Filterdurchlauf an einen freien SMTP-Port ohne `content_filter` zurück schickt, wo die E-Mail von Postfix normal weiterbearbeitet werden kann.

- In Schritt 2 wird wie beim SMTP-only Milter ermittelt, ob die E-Mail eine OpenPGP-Nachricht enthält. Ist diese eine verschlüsselte Nachricht, so durchläuft sie den Filter zur Säuberung der Metadaten weiter. Alle anderen Nachrichten, die unverschlüsselt, nur signiert oder Schlüsselmaterial enthalten verlassen den Filter wieder.
- Schritt 3 ist analog zum SMTP-only Milter. Sollte die Anzahl der `RCPT TO:`-Adressen größer sein als die Anzahl der E-Mail-Adressen aus den Header Fields, so ist anzunehmen, dass Bcc-Adressen vorhanden sind. Diese Nachrichten erfahren eine fortgesetzte Filterbehandlung. Andere Nachrichten können den Filter verlassen.
- Im Schritt 4 wird die OpenPGP-Nachricht von der ASCII-Hülle befreit.
- Im fünften Schritt wird zwischen „normalen“, in den Header Fields enthaltenen, E-Mail-Adressen und den Bcc-Adressen, die nur im `RCPT TO:`-Feld enthalten sind, unterschieden.
- Für Erstere wird im weiteren Verlauf des Filters (Schritte 6, 7 und 8) eine E-Mail generiert, aus der alle PKESK-Pakete mit den Key-IDs der Bcc-Adressen aus der OpenPGP-Nachricht entfernt werden. Diese neue E-Mail kann dann den Filter verlassen. Da sie erneut gequeuet wird, müssen für diese E-Mail die `RCPT TO:`-Informationen angepasst werden.
- Für die Nachrichten (ab Schritt 9), die die Bcc-Empfänger erreichen sollen, muss für jeden Bcc-Empfänger eine eigene Kopie der Ursprungsnachricht gesäubert werden. PKESK-Pakete mit den Key-IDs anderer Bcc-Empfänger werden hier entfernt. Auch hier muss bei Verlassen des Filters die Liste der `RCPT TO:`-Empfänger aktualisiert werden.

Haben alle gesäuberten E-Mails den Filter verlassen, kann die ursprüngliche E-Mail, die in Schritt 1 den Filter erreichte, verworfen werden.

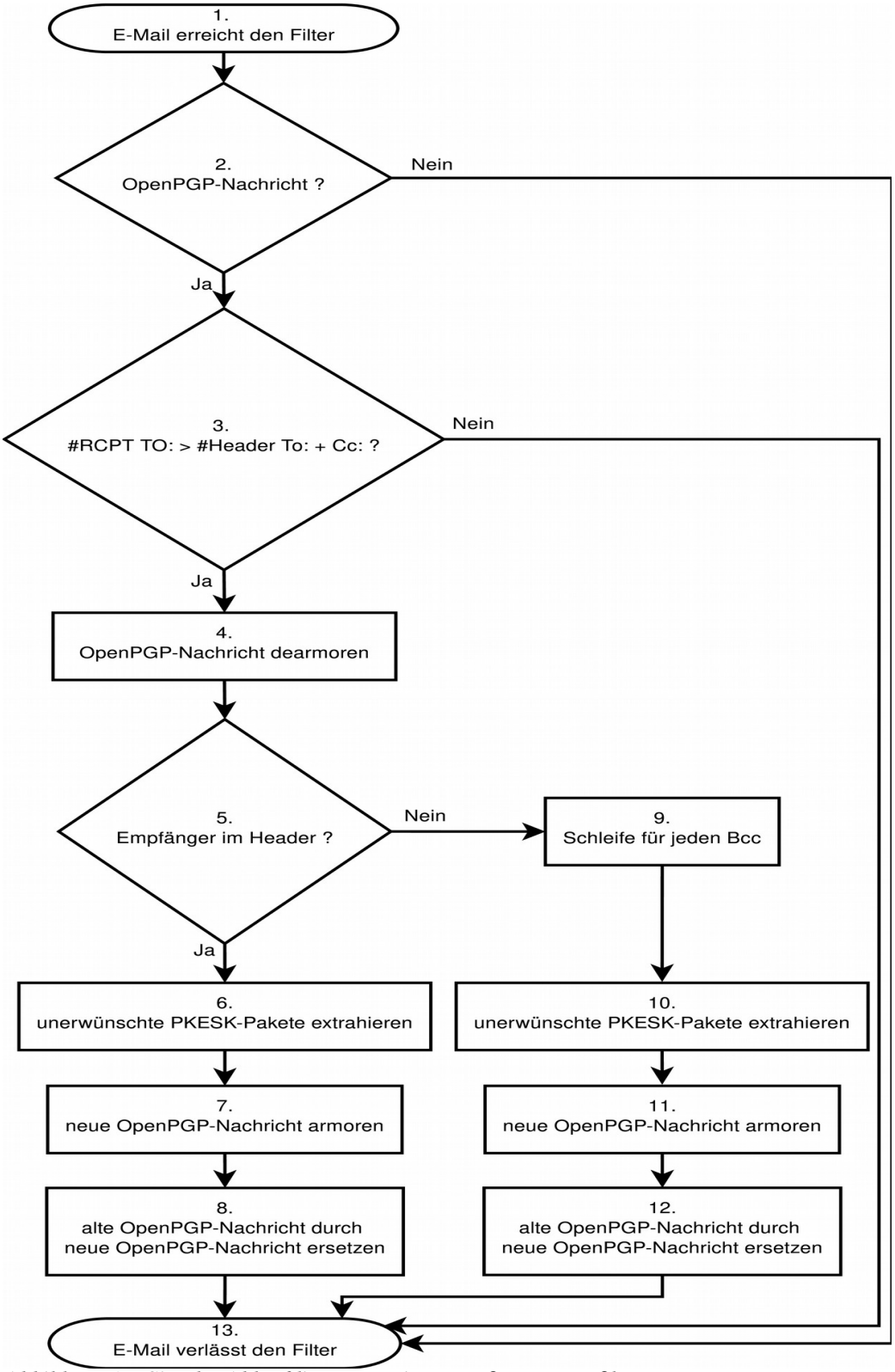

Abbildung 17: Simples Ablaufdiagramm eines Postfix content_filters

5 Schlussbetrachtung

5.1 Fazit

Die durch das Szenario festgelegte Fragestellung, ob Metadaten aus einer einmalig vom Mailclient an einen SMTP-Server übertragenen PGP-verschlüsselten E-Mail an mehrere Empfänger, von denen mindestens eine E-Mail-Adresse im Bcc-Feld des Mailclients gesetzt ist, entfernt werden können, wurde positiv beantwortet.

Anhand der vielfältigen der E-Mail-Kommunikation zugrunde liegender Standards und einem praktischen Versuch konnte der strukturelle Aufbau einer PGP-verschlüsselten E-Mail detailreich gezeigt werden.

Da sich das Internet und mit ihm die E-Mail seit seiner Entstehung stetig entwickelt, die Standards unermüdlich von den jeweiligen Autoren angepasst und diskutiert werden, ist es eine Herausforderung die an dem Szenario beteiligten Standarddefinitionen in Form von RFCs zu identifizieren und anzuwenden. Verblüffend ist die Fülle der einer einfachen PGP-verschlüsselten E-Mail begleitenden Definitionen.

Das Kapitel 2 beleuchtet die elementaren Begriffe der E-Mail-Kommunikation im Allgemeinen sowie die Struktur einer OpenPGP-Nachricht und deren Einbettung in eine E-Mail im Besonderen. Basierend auf diesem Wissen wäre bereits an dieser Stelle eine positive Antwort der einleitenden Fragestellung plausibel. Bereits in diesem Kapitel können die Metadaten tragenden Elemente identifiziert werden. Aus den Regelwerken der ISOC konnte auch kein Sicherungsmechanismus abgeleitet werden, der die Metadaten vor Extraktion schützt.

In den Tätigkeiten zum Kapitel 3 überraschte, dass die Untersuchungen der OpenPGP-Nachrichten mit vergleichendem Hexeditor Dhex oder `pgpdump` zügiger erleuchten als tagelange Rezeption der Standarddokumente. Mithilfe der motivierenden Erkenntnisse aus dem praktischen Teil im Kapitel 3 konnte ein Mechanismus erarbeitet werden, der aus PGP-verschlüsselten E-Mails die Metadaten entfernt. Eine Anwendung dieses Mechanismus lässt sich als Mailfilter realisieren, der in Kapitel 4 vorgeschlagen wurde.

Der hier vorgestellte Mechanismus ist in der Lage die Pakete mit den darin enthaltenen Key-IDs aus einer PGP-verschlüsselten Nachricht in einer E-Mail zu entfernen ohne die Verschlüsselung zu brechen. Insbesondere muss der Versand durch die Mailclientsoftware nicht in mehrere einzelne E-Mails beim Client aufgebrochen werden, es kann also Bandbreite und bei der Archivierung auch Speicherplatz gespart werden. Eine Anwendung dieses Verfahrens führt dazu, dass durch Verwendung von Bcc-Feldern im Mailclient versteckte Empfänger auch bei PGP-verschlüsselten E-Mails verborgen bleiben. Der Mechanismus ermöglicht somit eine PGP-verschlüsselte Kommunikation per E-Mail ohne Preisgabe von nicht benötigten Metadaten. Ein Empfänger hat in diesem Fall weder Kenntnis von der tatsächlichen Anzahl weiterer versteckter Empfänger noch von der Key-ID, mit der sich eine E-Mailadresse durch Schlüsselserverabfrage ermitteln ließe.

5.2 Offene Fragen

- Nicht beleuchtet wurde ein anderer weit verbreiteter Standard zum Verschlüsseln von E-Mails. Der Content-Type `application/pkcs7-mime`, wie er vom S/MIME Standard verwendet wird, beinhaltet ebenfalls verschlüsselte Daten.
- Ferner wurden nur beiläufig PGP/Inline-formatierte E-Mails im Kapitel 3.2.2.2 erwähnt. Es ist einleuchtend anzunehmen, dass sie ebenfalls von Metadaten gesäubert werden können. Ein konkreter Nachweis fehlt hier.
- Da der praktische Teil lediglich mit Mozilla Thunderbird und dem Addon Enigmail durchgeführt wurde, könnte die Nichtexistenz eines Hinweises auf Veränderung der ESK-Sequenz ein Bug der Programme sein. Zwar zeigen Tests nur die Anwesenheit von Fehlern, dennoch könnten Versuche mit anderen PGP-fähigen Mailclients die Interoperabilität bestätigen.
- Gibt es PGP-fähige Mailclients, die das neue Paketformat in den PKESK-Paketen verwenden?
- Offen ist auch das Problem mit der Verknüpfung einer Key-ID mit mehreren E-Mailadressen. Der hier vorgestellte Mechanismus kann nur mit einer eineindeutigen Zuordnung von Key-ID zu E-Mailadresse die gewünschten Resultate liefern.

Mögliche Gegenmaßnahmen zur Sabotage des Mailfilters:

- Seit Version 2 unterstützt Enigmail das Verschlüsseln des Subjects aus dem Header einer E-Mail. Dazu bedient es sich eines Verfahrens namens Memoryhole. Mittels Memoryhole ist es möglich beliebige Header Fields im Chiffrat zu verstecken, so auch die Header Fields `To:` und `Cc:`, was für den hier vorgestellten Mechanismus das Aus bedeutet.
- Eine in die OpenPGP-Nachricht platzierte ESK-Sequenz-Nummer könnte Basis für Hinweise auf Veränderung der ESK-Sequenz sein. Ebenso „Blockchaining“ der OpenPGP-Pakete.

Andere Veränderungen der ESK-Sequenz:

- Da der Mechanismus beweist, dass Veränderungen an der ESK-Sequenz unerkannt bleiben, stellt sich die eher paranoide Frage, wie eine mögliche Beigabe eines „Fake-PKESK-Paketes“ wirkt. Adresssammler werden mit falschen Key-IDs versorgt, daraus erstellte soziale Graphen haben neue Kanten. Ermittlungsbehörden und Geheimdienste repressiver Regimes könnten Verdächtigungen oder Indizien erzeugen wollen um Existenzen von Menschen zu ruinieren. Auf den ersten Blick sieht die OpenPGP-Nachricht in einem solchen Szenario RFC-konform aus. Mit dem falschen Session Key des Fake-PKESK-Paketes ist eine Entschlüsselung der Nachricht jedoch unmöglich.

5.3 Ausblick

In den letzten Jahren haben verschiedene Interessengruppen die Nützlichkeit vom Anwender in Computersystemen hinterlassener Daten und Metadaten erkannt und für ihre Zwecke aufbereitet.

War es zu Beginn der elektronischen Kommunikation mittels einer einfachen E-Mail die Empfängerinformationen ganz leicht durch Verwendung von Bcc-Feldern im Mailclient zu verbergen, so ist es im Falle einer PGP-verschlüsselten E-Mail an diverse Adressaten nur mit ein wenig, vom Anwender nicht beeinflussbaren, Aufwand möglich diese konkreten Datenspuren zu entfernen.

Die in dieser Arbeit behandelten Metadaten einer OpenPGP-Nachricht sind für die Entschlüsselung derselben essentiell. Es ist jedoch hinreichend pro Empfänger nur einen verschlüsselten Session Key in der Nachricht zu übertragen. Die mitübertragenen Key-IDs der weiteren Empfänger sind für den Inhalt der Nachricht unbedeutend und können von den Interessengruppen für ihre Bestreben gebraucht oder missbraucht werden. Die hier vorgestellten Standards sehen vor, eine PGP-verschlüsselte E-Mail an mehrere Empfänger im Bcc nur einmal zu übertragen und die Key-IDs aller Beteiligter mitzusenden.

Diese Datenspuren liegen wie die Brotkrümel aus Grimms Märchen „Hänsel und Gretel“ verteilt in den Postfächern der Empfänger.

Es ist für den Schutz der Privatsphäre von PGP-verschlüsselten E-Mails nutzenden Anwendern wünschenswert, wenn eine Reduktion der Metadaten stattfindet. Vorstellbar sind Mailfilter, die die Säuberung bewerkstelligen. Auch ein „Cleanup-PGP-Metadata-Modul“ im Postfix-System wäre denkbar.

Mailfilter dieser Art erweitern das Konzept vom Empfängerverstecken durch Bcc bis in die OpenPGP-Nachrichten hinein.

Da aber eine Key-ID mit multiplen E-Mailadressen verknüpft sein kann und gleichfalls nicht garantiert werden kann die jeweils nötige korrekte E-Mailadresse zu ermitteln, ist abzusehen, dass es bei Verwendung eines solchen Mailfilters das Verfahren versagen kann.

Anhang

Glossar

Body:
Körper der E-Mail oder MIME-kodierten Nachricht

Content:
Inhalt, im SMTP-Dialog die zu übertragenen Daten

Envelope:
Umschlag, im SMTP-Dialog die Steuerungsinformationen für den Mailserver

Header:
Nachrichtenkopf einer E-Mail oder einer MIME-kodierten Nachricht

Header Field:
Kopffeld, ein Header Field ist eine einzelne Kopfzeile des Headers

Header Section:
Kopfsektion, ist die Zusammenfassung der Header Fields, bei E-Mail der durch RFC präzisierte Ausdruck für den Header

Mail Transport Agent (MTA):
Ein Programm zum Verbreiten von E-Mail

Mail User Agent (MUA):
Ein Programm zum Verfassen, Senden, Empfangen und Darstellen einer E-Mail

Proxy:
[2] bezeichnet einen Proxyserver als *"Server, der als Vermittler und Filter zwischen dem Computer des Benutzers und anderen Servern dient"*.

Verzeichnis der Akronyme

A	
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
B	
Bcc	Blind Carbon Copy
C	
Cc	Carbon Copy
CRC	Cyclic Redundancy Check
E	
eml	Dateinamenserweiterung für E-Mail-Nachrichten
ESK	Encrypted Session Key
ESMTP	Extended Simple Mail Transfer Protocol
G	
GNU	GNU's Not Unix
GNU GPL	GNU General Public License
GPG	GNU Privacy Guard
H	
HTML	Hypertext Markup Language
I	
IETF	Internet Engineering Task Force
IMF	Internet Message Format
ISOC	Internet Society
L	
LMTP	Local Mail Transfer Protocol

M

MIME	Multipurpose Internet Mail Extensions
MSB	Most significant bit
MTA	Mail Transport Agent
MUA	Mail User Agent

P

PGP	Pretty Good Privacy
PKESK	Public Key Encrypted Session Key

R

RFC	Request for Comments

S

SKESK	Symmetric Key Encrypted Session Key
SMIME	Secure Multipurpose Internet Mail Extensions
SMTP	Simple Mail Transfer Protocol

OpenPGP-Zertifikate

```
$ gpg -a -export „Name“ | gpg -list-packets -verbose
```

```
# off=0 ctb=99 tag=6 hlen=3 plen=525
:public key packet:
        version 4, algo 1, created 1503928101, expires 0
        pkey[0]: [4096 bits]
        pkey[1]: 010001
        keyid: 63F16806B479DCA7
# off=528 ctb=b4 tag=13 hlen=2 plen=26
:user ID packet: "Alice <alice@daenicke.org>"
# off=556 ctb=89 tag=2 hlen=3 plen=596
:signature packet: algo 1, keyid 63F16806B479DCA7
        version 4, created 1503928101, md5len 0, sigclass 0x13
        digest algo 8, begin of digest 0a 14
        hashed subpkt 33 len 21 (issuer fpr v4 4A0F729F69DBA36205712DA763F16806B479DCA7)
        hashed subpkt 2 len 4 (sig created 2017-08-28)
        hashed subpkt 27 len 1 (key flags: 23)
        hashed subpkt 9 len 4 (key expires after 5y0d0h0m)
        hashed subpkt 11 len 4 (pref-sym-algos: 9 8 7 2)
        hashed subpkt 21 len 5 (pref-hash-algos: 8 9 10 11 2)
        hashed subpkt 22 len 3 (pref-zip-algos: 2 3 1)
        hashed subpkt 30 len 1 (features: 01)
        hashed subpkt 23 len 1 (keyserver preferences: 80)
        subpkt 16 len 8 (issuer key ID 63F16806B479DCA7)
        data: [4096 bits]
# off=1155 ctb=b9 tag=14 hlen=3 plen=525
:public sub key packet:
        version 4, algo 1, created 1503928101, expires 0
        pkey[0]: [4096 bits]
        pkey[1]: 010001
        keyid: EE2718AE9D417941
# off=1683 ctb=89 tag=2 hlen=3 plen=572
:signature packet: algo 1, keyid 63F16806B479DCA7
        version 4, created 1503928101, md5len 0, sigclass 0x18
        digest algo 8, begin of digest 08 2a
        hashed subpkt 33 len 21 (issuer fpr v4 4A0F729F69DBA36205712DA763F16806B479DCA7)
        hashed subpkt 2 len 4 (sig created 2017-08-28)
        hashed subpkt 27 len 1 (key flags: 0C)
        hashed subpkt 9 len 4 (key expires after 5y0d0h0m)
        subpkt 16 len 8 (issuer key ID 63F16806B479DCA7)
        data: [4096 bits]
```

Text 3: OpenPGP-Zertifikat Alice

```
# off=0 ctb=99 tag=6 hlen=3 plen=269
:public key packet:
        version 4, algo 1, created 1503928912, expires 0
        pkey[0]: [2048 bits]
        pkey[1]: 010001
        keyid: F2F7417CEAB97C37
# off=272 ctb=b4 tag=13 hlen=2 plen=22
:user ID packet: "Bob <bob@daenicke.org>"
# off=296 ctb=89 tag=2 hlen=3 plen=340
:signature packet: algo 1, keyid F2F7417CEAB97C37
        version 4, created 1503928912, md5len 0, sigclass 0x13
        digest algo 8, begin of digest fe 02
        hashed subpkt 33 len 21 (issuer fpr v4 4663F866C1984BE96A6FADECF2F7417CEAB97C37)
        hashed subpkt 2 len 4 (sig created 2017-08-28)
        hashed subpkt 27 len 1 (key flags: 23)
        hashed subpkt 9 len 4 (key expires after 5y0d0h0m)
        hashed subpkt 11 len 4 (pref-sym-algos: 9 8 7 2)
        hashed subpkt 21 len 5 (pref-hash-algos: 8 9 10 11 2)
        hashed subpkt 22 len 3 (pref-zip-algos: 2 3 1)
        hashed subpkt 30 len 1 (features: 01)
        hashed subpkt 23 len 1 (keyserver preferences: 80)
        subpkt 16 len 8 (issuer key ID F2F7417CEAB97C37)
        data: [2048 bits]
# off=639 ctb=b9 tag=14 hlen=3 plen=269
:public sub key packet:
        version 4, algo 1, created 1503928912, expires 0
        pkey[0]: [2048 bits]
        pkey[1]: 010001
        keyid: 13511613141401D5
# off=911 ctb=89 tag=2 hlen=3 plen=316
:signature packet: algo 1, keyid F2F7417CEAB97C37
        version 4, created 1503928912, md5len 0, sigclass 0x18
        digest algo 8, begin of digest e6 bb
        hashed subpkt 33 len 21 (issuer fpr v4 4663F866C1984BE96A6FADECF2F7417CEAB97C37)
        hashed subpkt 2 len 4 (sig created 2017-08-28)
        hashed subpkt 27 len 1 (key flags: 0C)
        hashed subpkt 9 len 4 (key expires after 5y0d0h0m)
        subpkt 16 len 8 (issuer key ID F2F7417CEAB97C37)
        data: [2048 bits]
```

Text 4: OpenPGP-Zertifikat Bob

```
# off=0 ctb=99 tag=6 hlen=3 plen=525
:public key packet:
        version 4, algo 1, created 1503930018, expires 0
        pkey[0]: [4096 bits]
        pkey[1]: 010001
        keyid: 75FA2FA373CAA6DE
# off=528 ctb=b4 tag=13 hlen=2 plen=22
:user ID packet: "Eve <eve@daenicke.org>"
# off=552 ctb=89 tag=2 hlen=3 plen=596
:signature packet: algo 1, keyid 75FA2FA373CAA6DE
        version 4, created 1503930018, md5len 0, sigclass 0x13
        digest algo 8, begin of digest 4d 1b
        hashed subpkt 33 len 21 (issuer fpr v4 9BA0236414F4ED1401510AEE75FA2FA373CAA6DE)
        hashed subpkt 2 len 4 (sig created 2017-08-28)
        hashed subpkt 27 len 1 (key flags: 23)
        hashed subpkt 9 len 4 (key expires after 5y0d0h0m)
        hashed subpkt 11 len 4 (pref-sym-algos: 9 8 7 2)
        hashed subpkt 21 len 5 (pref-hash-algos: 8 9 10 11 2)
        hashed subpkt 22 len 3 (pref-zip-algos: 2 3 1)
        hashed subpkt 30 len 1 (features: 01)
        hashed subpkt 23 len 1 (keyserver preferences: 80)
        subpkt 16 len 8 (issuer key ID 75FA2FA373CAA6DE)
        data: [4096 bits]
# off=1151 ctb=b9 tag=14 hlen=3 plen=525
:public sub key packet:
        version 4, algo 1, created 1503930018, expires 0
        pkey[0]: [4096 bits]
        pkey[1]: 010001
        keyid: 803A04C1444EEB6B
# off=1679 ctb=89 tag=2 hlen=3 plen=572
:signature packet: algo 1, keyid 75FA2FA373CAA6DE
        version 4, created 1503930018, md5len 0, sigclass 0x18
        digest algo 8, begin of digest 49 2e
        hashed subpkt 33 len 21 (issuer fpr v4 9BA0236414F4ED1401510AEE75FA2FA373CAA6DE)
        hashed subpkt 2 len 4 (sig created 2017-08-28)
        hashed subpkt 27 len 1 (key flags: 0C)
        hashed subpkt 9 len 4 (key expires after 5y0d0h0m)
        subpkt 16 len 8 (issuer key ID 75FA2FA373CAA6DE)
        data: [4096 bits]
```

Text 5: OpenPGP-Zertifikat Eve

Mailquellenbeispiele

```
X-Mozilla-Status: 0001
X-Mozilla-Status2: 00800000
X-Mozilla-Keys:
BCC: eve@daenicke.org
To: Bob@daenicke.org
From: Alice <alice@daenicke.org>
Subject: A2BBlindE
Message-ID: <97dbd667-1924-5174-099a-6fad6c28c2ac@daenicke.org>
Date: Mon, 5 Mar 2018 11:32:16 +0100
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
 Thunderbird/52.2.1
MIME-Version: 1.0
Content-Type: multipart/encrypted;
 protocol="application/pgp-encrypted";
 boundary="7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J"

This is an OpenPGP/MIME encrypted message (RFC 4880 and 3156)
--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J
Content-Type: application/pgp-encrypted
Content-Description: PGP/MIME version identification

Version: 1

--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J
Content-Type: application/octet-stream; name="encrypted.asc"
Content-Description: OpenPGP encrypted message
Content-Disposition: inline; filename="encrypted.asc"

-----BEGIN PGP MESSAGE-----

hQEMAxNRFhMUFAHVAQgApSwAtLIDtpjQplCeJ2qYBlbpPIwCZn9Zud0R+k7G2fPt
0fNjzf0nS+t3xibbhFe3uOoo9F6NRCzFIZBZdPSAtmekWVqPlBjVCzk68czf0wDR
[…]
cH1EPag9efsX19YoVIpLxaJjFwzOGllMKVICh/T1iSypUyZaCsR5G2rrXwIt7bh1
n40Vov9PmROxCbFLmfmNLs/Q8EAtMcuvxQtMTz2zS3Uoa1m/x7aWQ+225XL7+aXU
GbbmGHf6dwsksyhPLP0=
=5jiU
-----END PGP MESSAGE-----

--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J--
```

Text 6: Gespeicherte Nachrichtendatei von Alice

```
X-Account-Key: account3
X-UIDL: 0000002058f3989c
X-Mozilla-Status: 0001
X-Mozilla-Status2: 00000000
X-Mozilla-Keys:
Return-path: <alice@daenicke.org>
Delivery-date: Mon, 05 Mar 2018 11:32:18 +0100
Received: from mi029.mc1.hosteurope.de ([80.237.138.226])
        by wp147.webpack.hosteurope.de running ExIM with esmtps
(TLS1.2:ECDHE_RSA_AES_256_GCM_SHA384:256)
        id 1esnPm-0004Vs-3F; Mon, 05 Mar 2018 11:32:18 +0100
Received: from wp147.webpack.hosteurope.de ([80.237.132.154])
        by mx0.webpack.hosteurope.de (mi029.mc1.hosteurope.de) with esmtps (TLSv1:ECDHE-
RSA-AES128-SHA:128)
        id 1esnPl-000296-FD; Mon, 05 Mar 2018 11:32:17 +0100
Received: from [2a02:8109:98c0:58ec::2]; authenticated
        by wp147.webpack.hosteurope.de running ExIM with esmtpa
        id 1esnPl-0004Vb-6p; Mon, 05 Mar 2018 11:32:17 +0100
To: Bob@daenicke.org
From: Alice <alice@daenicke.org>
Subject: A2BBlindE
Message-ID: <97dbd667-1924-5174-099a-6fad6c28c2ac@daenicke.org>
Date: Mon, 5 Mar 2018 11:32:16 +0100
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
 Thunderbird/52.2.1
MIME-Version: 1.0
Content-Type: multipart/encrypted;
 protocol="application/pgp-encrypted";
 boundary="7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J"
X-bounce-key: webpack.hosteurope.de;alice@daenicke.org;1520245937;41eeb18b;
X-HE-SMSGID: 1esnPl-0004Vb-6p
X-HE-Virus-Scanned: Yes
X-HE-Spam-Level: -
X-HE-Spam-Score: -1.7
X-HE-Spam-Report: Content analysis details:   (-1.7 points)
  pts rule name              description
 ---- ---------------------- --------------------------------------------------
 -0.7 RCVD_IN_DNSWL_LOW      RBL: Sender listed at http://www.dnswl.org/, low
                             trust
                             [80.237.132.154 listed in list.dnswl.org]
 -1.0 ENCRYPTED_MESSAGE      Message is encrypted, not likely to be spam
  0.0 TVD_SPACE_RATIO        TVD_SPACE_RATIO
Envelope-to: bob@daenicke.org

This is an OpenPGP/MIME encrypted message (RFC 4880 and 3156)
--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J
Content-Type: application/pgp-encrypted
Content-Description: PGP/MIME version identification

Version: 1

--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J
Content-Type: application/octet-stream; name="encrypted.asc"
Content-Description: OpenPGP encrypted message
Content-Disposition: inline; filename="encrypted.asc"

-----BEGIN PGP MESSAGE-----

hQEMAxNRFhMUFAHVAQgApSwAtLIDtpjQplCeJ2qYBlbpPIwCZn9Zud0R+k7G2fPt
0fNjzf0nS+t3xibbhFe3uOoo9F6NRCzFIZBZdPSAtmekWVqPlBjVCzk68czf0wDR
[…]
cH1EPag9efsX19YoVIpLxaJjFwzOGllMKVICh/T1iSypUyZaCsR5G2rrXwIt7bh1
n40Vov9PmROxCbFLmfmNLs/Q8EAtMcuvxQtMTz2zS3Uoa1m/x7aWQ+225XL7+aXU
GbbmGHf6dwsksyhPLP0=
=5jiU
-----END PGP MESSAGE-----

--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J--
```

Text 7: Gespeicherte Nachrichtendatei von Bob

```
X-Account-Key: account4
X-UIDL: 0000001858f39903
X-Mozilla-Status: 0001
X-Mozilla-Status2: 00000000
X-Mozilla-Keys:
Return-path: <alice@daenicke.org>
Delivery-date: Mon, 05 Mar 2018 11:32:18 +0100
Received: from mi029.mc1.hosteurope.de ([80.237.138.226])
        by wp147.webpack.hosteurope.de running ExIM with esmtps
(TLS1.2:ECDHE_RSA_AES_256_GCM_SHA384:256)
        id 1esnPm-0004Vr-3B; Mon, 05 Mar 2018 11:32:18 +0100
Received: from wp147.webpack.hosteurope.de ([80.237.132.154])
        by mx0.webpack.hosteurope.de (mi029.mc1.hosteurope.de) with esmtps (TLSv1:ECDHE-
RSA-AES128-SHA:128)
        id 1esnPl-000296-FD; Mon, 05 Mar 2018 11:32:17 +0100
Received: from [2a02:8109:98c0:58ec::2]; authenticated
        by wp147.webpack.hosteurope.de running ExIM with esmtpa
        id 1esnPl-0004Vb-6p; Mon, 05 Mar 2018 11:32:17 +0100
To: Bob@daenicke.org
From: Alice <alice@daenicke.org>
Subject: A2BBlindE
Message-ID: <97dbd667-1924-5174-099a-6fad6c28c2ac@daenicke.org>
Date: Mon, 5 Mar 2018 11:32:16 +0100
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
 Thunderbird/52.2.1
MIME-Version: 1.0
Content-Type: multipart/encrypted;
 protocol="application/pgp-encrypted";
 boundary="7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J"
X-bounce-key: webpack.hosteurope.de;alice@daenicke.org;1520245937;41eeb18b;
X-HE-SMSGID: 1esnPl-0004Vb-6p
X-HE-Virus-Scanned: Yes
X-HE-Spam-Level: -
X-HE-Spam-Score: -1.7
X-HE-Spam-Report: Content analysis details:   (-1.7 points)
  pts rule name              description
 ---- ---------------------- --------------------------------------------------
 -0.7 RCVD_IN_DNSWL_LOW      RBL: Sender listed at http://www.dnswl.org/, low
                             trust
                             [80.237.132.154 listed in list.dnswl.org]
 -1.0 ENCRYPTED_MESSAGE      Message is encrypted, not likely to be spam
  0.0 TVD_SPACE_RATIO        TVD_SPACE_RATIO
Envelope-to: eve@daenicke.org

This is an OpenPGP/MIME encrypted message (RFC 4880 and 3156)
--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J
Content-Type: application/pgp-encrypted
Content-Description: PGP/MIME version identification

Version: 1

--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J
Content-Type: application/octet-stream; name="encrypted.asc"
Content-Description: OpenPGP encrypted message
Content-Disposition: inline; filename="encrypted.asc"

-----BEGIN PGP MESSAGE-----

hQEMAxNRFhMUFAHVAQgApSwAtLIDtpjQplCeJ2qYBlbpPIwCZn9Zud0R+k7G2fPt
0fNjzf0nS+t3xibbhFe3uOoo9F6NRCzFIZBZdPSAtmekWVqPlBjVCzk68czf0wDR
[…]
cH1EPag9efsX19YoVIpLxaJjFwzOGllMKVICh/T1iSypUyZaCsR5G2rrXwIt7bh1
n40Vov9PmROxCbFLmfmNLs/Q8EAtMcuvxQtMTz2zS3Uoa1m/x7aWQ+225XL7+aXU
GbbmGHf6dwsksyhPLP0=
=5jiU
-----END PGP MESSAGE-----

--7TJGTdhjFlSDTM2q0SBw5xSjDAIF8CM9J--
```

Text 8: Gespeicherte Nachrichtendatei von Eve

Quellenverzeichnis

[1] Schneier, B. (**2008**). *Security vs. Privacy*. Online: <https://www.schneier.com/blog/archives/2008/01/security_vs_pri.html>. (Abrufdatum: 27.04.2018).

[2] Dudenredaktion (Ed.), (**2015**). *Duden - Das Fremdwörterbuch*. Dudenverlag, Berlin.

[3] Hayden, M. (**2014**). *The Johns Hopkins Foreign Affairs Symposium Presents: The Price of Privacy: Re-Evaluating the NSA*. Online: <https://www.youtube.com/watch?v=kV2HDM86XgI#t=18m00s>. (Abrufdatum: 30.07.2018).

[4] Bierman, K. (**2013**). *Leben im Überwachungsstaat*. In: Markus Beckedahl, A. M. (Ed.), *Überwachtes Netz - Edward Snowden und der größte Überwachungsskandal der Geschichte*, Seiten 20-25. newthinking communications.

[5] Callas, J.; Donnerhacke, L.; Finney, H.; Shaw, D. and Thayer, R. (**2007**). *OpenPGP Message Format*. RFC 4880 (Proposed Standard). Online: <https://www.rfc-editor.org/rfc/rfc4880.txt>.

[6] Schleuderteam (**2012**). *Schleuder/ concept*. Online: <https://schleuder2.nadir.org/documentation/v2.2/concept.html>. (Abrufdatum: 19.04.2018).

[7] Zertificon Solutions GmbH (**2014**). *E-Mail-Verschlüsselung für Unternehmen – S/MIME Gateway und mehr DSGVO ready*. Online: <https://www.zertificon.com/loesungen/email-verschluesselung-gateway>. (Abrufdatum: 31.07.2018).

[8] GoodCrypto (**2017**). *Secure gateway for mail and browsing | GoodCrypto*. Online: <https://goodcrypto-private-server.sourceforge.io/>. (Abrufdatum: 19.04.2018).

[9] Bradner, S. (**1996**). *The Internet Standards Process -- Revision 3*. RFC 2026 (Best Current Practice). Online: <https://www.rfc-editor.org/rfc/rfc2026.txt>.

[10] Carius, F. (**kein Jahr**). *Microsoft Exchange FAQ*. Online: <https://www.msxfaq.de/internet/envelope.htm>. (Abrufdatum: 01.08.2018).

[11] Klensin, J. (**2008**). *Simple Mail Transfer Protocol*. RFC 5321 (Draft Standard). Online: <https://www.rfc-editor.org/rfc/rfc5321.txt>.

[12] Resnick (Ed.), P. (**2008**). *Internet Message Format*. RFC 5322 (Draft Standard). Online: <https://www.rfc-editor.org/rfc/rfc5322.txt>.

[13] Freed, N. and Borenstein, N. (**1996**). *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*. RFC 2045 (Draft Standard). Online: <https://www.rfc-editor.org/rfc/rfc2045.txt>.

[14] Freed, N. and Borenstein, N. (**1996**). *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*. RFC 2046 (Draft Standard). Online: <https://www.rfc-editor.org/rfc/rfc2046.txt>.

[15] Vaudreuil, G. (**2000**). *SMTP Service Extensions for Transmission of Large and Binary MIME Messages*. RFC 3030 (Proposed Standard). Online: <https://www.rfc-editor.org/rfc/rfc3030.txt>.

[16] Josefsson, S. (**2006**). *The Base16, Base32, and Base64 Data Encodings*. RFC 4648 (Proposed Standard). Online: <https://www.rfc-editor.org/rfc/rfc4648.txt>.

[17] Steele, D. (**2014**). *Anatomy of a GPG Key*. Online: <https://davesteele.github.io/gpg/2014/09/20/anatomy-of-a-gpg-key/>. (Abrufdatum: 02.08.2018).

[18] Elkins, M.; Torto, D. D.; Levien, R. and Roessler, T. (**2001**). *MIME Security with OpenPGP*. RFC 3156 (Proposed Standard). Online: <https://www.rfc-editor.org/rfc/rfc3156.txt>.

[19] Troost, R.; Dorner, S. and Moore (Ed.), K. (**1997**). *Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field*. RFC 2183 (Proposed Standard). Online: <https://www.rfc-editor.org/rfc/rfc2183.txt>.

[20] Schneier, B. (**2016**). *Data Is a Toxic Asset, So Why Not Throw It Out?*. Online: <https://www.schneier.com/essays/archives/2016/03/data_is_a_toxic_asse.html>. (Abrufdatum: 11.02.2018).

[21] Atom Smasher, S. J. (**2014**). *The "OpenPGP" mail and news header field*. Online: <https://tools.ietf.org/html/draft-josefsson-openpgp-mailnews-header-07>. (Abrufdatum: 25.06.2018).

[22] Team Autocrypt (**2018**). *Autocrypt Level 1 Specification*. Online: <https://autocrypt.org/autocrypt-spec-1.0.0.pdf>. (Abrufdatum: 26.06.2018).

[23] Fielding (Ed.), R. and Reschke (Ed.), J. (**2014**). *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*. RFC 7231 (Proposed Standard). Online: <https://www.rfc-editor.org/rfc/rfc7231.txt>.

[24] ElandSystems (**ohne Jahr**). *Filtering Mail with Sendmail/ Architecture*. Online: <http://www.elandsys.com/resources/sendmail/libmilter/>. (Abrufdatum: 10.06.2018).

[25] Hildebrandt, R. and Koetter, P., (**2008**). *Postfix - Einrichtung, Betrieb und Wartung*. dpunkt.verlag GmbH, Heidelberg.

[26] Skoll, D. F. (**2004**). *Combatting Spam Using Sendmail, MIMEDefang and Perl*. Online: <https://mimedefang.org/static/mimedefang-lisa04.pdf>. (Abrufdatum: 25.07.2018).

[27] Heinlein, P., (**2008**). *Das Postfix-Buch - Sichere Mailserver unter Linux*. Open Source Press, München.

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

Berlin, den

_____

Torsten Dänicke