BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN

University of Applied Sciences

Fachbereich VI - Informatik und Medien

Master-Arbeit

von

Benjamin Fichtner

zur Erlangung
des akademischen Grades
Master of Engineering

im Studiengang
Technische Informatik - Embedded Systems

Untersuchung von Techniken zur persönlichen E-Mail-Postfachverschlüsselung

Erstprüfer:	Prof. Dr. rer. nat. Thomas Scheffler
Gutachter:	Prof. Dr. rer. nat. Rüdiger Weis
Eingereicht am:	04.03.2019

Kurzfassung

E-Mails enthalten häufig sensible und schützenswerte Inhalte, die nicht für Dritte bestimmt sind. Obwohl die E-Mail ein weit verbreiteter Kommunikationsstandard ist, erfüllt sie viele aktuelle Sicherheitsanforderungen nicht [Foster et al., 2015]. Eines der Probleme ist, dass E-Mail-Inhalte im Klartext auf den Servern des E-Mail-Providers gespeichert sind. Dort haben berechtigte und unberechtigte Dritte eine dauerhafte Möglichkeit, auf diese zuzugreifen. Um das zu verhindern, wurde die sogenannte persönliche E-Mail-Postfachverschlüsselung entwickelt. Diese neuartige, serverseitige Schutzmaßnahme speichert eingehende E-Mails verschlüsselt ab. Sie sorgt dafür, dass die verschlüsselten E-Mails ausschließlich durch den Nutzer bzw. dessen Passwort entschlüsselt werden können.

In dieser Masterarbeit soll untersucht werden, ob der Einsatz von Postfachverschlüsselungstechniken Auswirkungen auf den E-Mail-Server-Betrieb hat und wie sich diese darstellen. Hierbei werden die vier Postfachverschlüsselungstechniken GPG-Sieve-Filter, MailCrypt, Scrambler und TREES miteinander verglichen. Dazu werden die verschiedenen Techniken analysiert, eine Testumgebung entwickelt, sowie eine Messreihe konzipiert und durchgeführt.

Abstract

In digital communication, e-mail is an indispensable means of communication. They often contain sensitive and protective content that is not intended for third parties. Although e-mail is such an important and widespread communication standard, it it does not meet modern security requirements [Foster et al., 2015]. One problem is that the e-mail content is stored in plain text on the e-mail provider's servers. There, authorized and unauthorized third parties have a permanent possibility to access them. To prevent this, the so-called personal mail box encryption was developed. This new, server-side protective measure stores incoming e-mails in encrypted form. It ensures that the encrypted e-mails can only be decrypted by the user or their password. This means that the e-mail server can only access the plaintext content as long as the user is logged in.

The aim of this master thesis is to find out whether the use of mailbox encryption techniques has an impact on e-mail server operation and how this is presented. The four mailbox encryption techniques GPG-Sieve-Filter, MailCrypt, Scrambler and TREES will be compared. It is investigated how existing effects depend on the mailbox encryption technique used. The techniques will be analyzed, a reproducible test environment developed, and a series of measurements designed and performed.

Danksagung

Ganz besonders möchte ich mich bei allen Elternteilen bedanken. Nur durch die finanzielle, moralische und inhaltliche Unterstützung von Detlef Fichtner, Catrin Scheller und Jürgen Scheller war es mir möglich, mein Studium in dieser Form zu vollenden.

Außerdem möchte ich mich ganz herzlich bei Jonas Meurer und Tim Dittler bedanken. Ihre Kritik und die daraus resultierenden Diskussionen haben diese Arbeit wesentlich bereichert.

Ein besonderer Dank gilt den Entwicklern der Programme GnuPG, Scrambler, TREES und Dovecot. Ohne eine Veröffentlichung der Quelltexte wäre diese Arbeit nie zustande gekommen.

Zusätzlich möchte ich mich Prof. Dr. rer. nat. Thomas Scheffler für den Vertrauensvorschuss und seine zahlreichen Hilfestellungen und konstruktiven Nachfragen bedanken.

Weiterer Dank gebührt allen Freunden und Freundinnen, die mich in den Jahren des Studiums begleitet und motiviert haben.

Vielen Dank!

Erklärung

Hiermit erkläre ich an Eides Statt, dass ich die vorliegende Arbeit selbst angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher keiner Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Berlin, den 04.03.2019	
Ort, Datum	Unterschrift

Inhaltsverzeichnis

1. Einleitung

1.1. Motivation

In der digitalen Kommunikation ist die E-Mail ein unverzichtbares Kommunikationsmittel. E-Mails enthalten häufig sensible und schützenswerte Inhalte, die nicht für Dritte bestimmt sind. Obwohl die E-Mail ein weit verbreiteter Kommunikationsstandard ist, erfüllt sie viele aktuelle Sicherheitsanforderungen nicht [Foster et al., 2015]. Gängige Sicherheitsfeatures, die einen umfassenden Schutz der Privatsphäre ermöglichen könnten, stehen in der Regel nur durch komplexe Zusatzsoftware zur Verfügung. Die meisten E-Mail-Anwender sind nicht versiert genug, diese Techniken zu verwenden [Ruoti et al., 2019] [mozilla.org, 2019]. Weitverbreitete serverseitige Schutzmaßnahmen zielen vorrangig auf die Absicherung der Transportwege und auf den Schutz der Nutzer-Zugangsdaten ab.

Dies hat zur Folge, dass E-Mails im Klartext auf den Servern des E-Mail-Providers gespeichert sind. Dort haben berechtigte und unberechtigte Dritte (Systemadministratoren, staatliche Behörden, Hacker) eine dauerhafte Möglichkeit, auf sensible Kommunikationsinhalte zuzugreifen. Dies geschieht meist ohne Wissen und Zustimmung des betroffenen Nutzers. Dass, E-Mail-Provider von staatlichen Behörden zur Herausgabe von E-Mail-Inhalten gezwungen werden, ist gängige Praxis [mailbox.org, 2019] [posteo.de, 2018] [Levison, 2014].

Aus Nutzerperspektive wäre es wünschenswert, wenn lediglich die beteiligten Kommunikationspartner Zugriff auf die Kommunikationsinhalte hätten. Hierzu müsste ein serverseitiger Zugriff auf die gespeicherten E-Mail-Inhalte unterbunden werden. Auch für einen datenschutzorientierten E-Mail-Provider wäre es vorteilhaft, wenn die Kommunikationsinhalte nicht im Klartext gespeichert wären. Im Falle eines erfolgreichen Server-Einbruchs würde dies für einen zusätzlichen Schutz der E-Mail-Inhalte sorgen.

Ein neues Verfahren, dass diesen Zustand ermöglicht, ist die sogenannte persönliche E-Mail-Postfachverschlüsselung.

1.2. Persönliche E-Mail-Postfachverschlüsselung

Das Verfahren der persönlichen E-Mail-Postfachverschlüsselung (in Folge kurz: Postfachverschlüsselung) sorgt für eine serverseitige Verschlüsselung eingehender E-Mails. Die Verschlüsselung erfolgt hierbei pro Nutzer. Zur Entschlüsselung ist die Eingabe des

jeweiligen Nutzer-Passworts notwendig. Die Techniken sind so konzipiert, dass der E-Mail-Server keinen dauerhaften Zugang auf die Klartextinhalte der abgespeicherten E-Mails erhalten kann. Der Zugriff auf die Inhalte ist nur für den Zeitraum möglich, in dem der Nutzer eingeloggt ist.

Das Verfahren der persönlichen Postfachverschlüsselung hat das Potential, auch bei einer größeren Anzahl von Nutzern Anwendung zu finden, da er ohne merkliche Änderungen auf Anwenderseite auskommt. Im Gegensatz zur Ende-zu-Ende-Verschlüsselung müssen hierbei jedoch Defizite beim Schutz der Kommunikationsinhalte auf dem Transportweg in Kauf genommen werden. Beim Einsatz der Postfachverschlüsselungstechniken werden die E-Mails auf dem Transportweg zum Beispiel nicht durch einen Schlüssel gesichert, der nur dem Anwender bekannt ist.

In den letzten fünf Jahren wurde die Postfachverschlüsselung von den E-Mail-Providern posteo.de[1,2], riseup.net[3,4], mailbox.org[5] und den Software-Entwicklern von Dovecot[6] vorangetrieben, implementiert und für deren Nutzer verfügbar gemacht.

Im Rahmen dieser Arbeit sollen die vier derzeit existenten Postfachverschlüsselungsverfahren näher untersucht werden.

1.3. Ziel der Untersuchung und Vorgehensweise

Ziel der Untersuchung ist es, herauszufinden, ob der Einsatz von Postfachverschlüsselungstechniken Auswirkungen auf den E-Mail-Server-Betrieb hat und wie sich diese darstellen. Weiterhin soll untersucht werden, ob eventuelle Auswirkungen abhängig von der eingesetzten Postfachverschlüsselungstechnik sind.

Dabei war es von besonderem Interesse, die Auswirkungen der Postfachverschlüsselung auf den E-Mail-Empfang und -Abruf zu untersuchen. Deshalb sollten unter anderem unterschiedliche Postfachformate, die Empfangs- und Abrufdauer, auch unter Berücksichtigung der E-Mail-Größe, die Speicherplatzbelegung und die Anzahl der Nutzer-Logins, hinsichtlich ihrer Beeinflussung durch den Einsatz dieser Technik untersucht werden.

Hierzu wird eine reproduzierbare Testumgebung entwickelt, sowie eine Messreihe konzipiert und durchgeführt.

[1] `https://posteo.de/site/verschluesselung#kryptomailspeicher`
[2] `https://github.com/posteo/scrambler-plugin`
[3] `https://riseup.net/en/about-us/press/canary-statement`
[4] `https://0xacab.org/riseuplabs/trees`
[5] `https://mailbox.org/im-stiftfilm-erklaert-das-vollstaendig-verschluesselte-postfach/`
[6] `https://wiki2.dovecot.org/Plugins/MailCrypt`

Diese hat folgende Versuche und Untersuchungen zum Gegenstand:

- Kompatibilität mit verschiedenen Postfachformaten beim E-Mail-Empfang
- Kompatibilität mit verschiedenen Postfachformaten beim E-Mail-Abruf
- Einfluss auf die E-Mail-Empfangsdauer
- Einfluss auf die E-Mail-Empfangsdauer in Abhängigkeit der E-Mail-Größe
- Einfluss auf den benötigten Speicherplatz
- Einfluss auf die Anzahl der IMAP-Logins
- Einfluss auf die E-Mail-Abrufdauer
- Einfluss auf den Nutzerkontenerstellungsprozess

Zur besseren Übersichtlichkeit werden die Ergebnisse der jeweiligen Teilversuche und deren Diskussion zusammengefasst beschrieben.

2. Theoretische Grundlagen

Dieses Kapitel behandelt die theoretischen Grundlagen, die zum besseren Verständnis dieser Arbeit hilfreich sind. Hierzu werden die Grundlagen und notwendigen Protokolle der E-Mail-Kommunikation, sowie die einzelnen Komponenten einer E-Mail-Architektur beschrieben. Danach erfolgt eine Einführung in die kryptographischen Grundlagen. Hierbei wird näher auf Verschlüsselungsarten und das Hashen von Passwörtern eingegangen. Zum Abschluss des Kapitels wird die eingesetzte Software vorgestellt.

2.1. Begriffsklärung

In diesem Unterkapitel werden verwendete Begriffe vorgestellt und deren Bedeutung erklärt.

- **E-Mail-Client:** Als E-Mail-Client wird ein Computerprogramm bezeichnet, welches E-Mails versenden, abrufen oder verwalten kann.
- **E-Mail-Server:** Als E-Mail-Server wird ein Computersystem bezeichnet, der E-Mails annehmen, versenden, weiterleiten und abspeichern kann.
- **E-Mail-Provider:** Als E-Mail-Provider wird ein Dienstleister bezeichnet, der Nutzern kostenlose oder kostenpflichtige E-Mail-Postfächer zur Verfügung stellt. Dieser betreibt in der Regel eine Vielzahl von E-Mail-Servern. Die Begriffe E-Mail-Server-Betreiber, E-Mail-Service-Anbieter und E-Mail-Service-Provider werden in dieser Arbeit als Synonyme verwendet.
- **Lokales Postfach:** Als lokales Postfach wird ein E-Mail-Postfach bezeichnet, welches sich auf dem Endgerät des Nutzers befindet.
- **Entferntes Postfach:** Als entferntes Postfach wird ein E-Mail-Postfach bezeichnet, das sich auf dem Server des E-Mail-Providers befindet.

2.2. E-Mail

Ein „Mail Object“, umgangssprachlich auch E-Mail genannt, besteht nach Simple Mail Transfer Protocol (SMTP)-Standard, aus zwei Teilen: dem Envelope und dem Content [Klensin, 2008] (siehe Abbildung 1).

Der Envelope enthält alle nötigen Informationen, damit eine E-Mail vom Sender zum Empfänger übertragen werden kann (siehe Listing 1).

Der Content unterteilt sich in zwei Abschnitte, Header und Body (siehe Listing 4). Im Header-Abschnitt befinden sich Informationen über die E-Mail (siehe Listing 2).

Dort können beispielsweise Informationen über den Weg der Nachricht, ihre Empfänger und eine Anti-Spam-Prüfung enthalten sein. Diese Informationen können unter anderem durch die Kommunikationsteilnehmer hinzugefügt werden. Hierzu zählen alle Server auf dem Transportweg einer E-Mail.

Im Body-Abschnitt befindet sich der eigentliche Inhalt der Nachricht (siehe Listing 3).

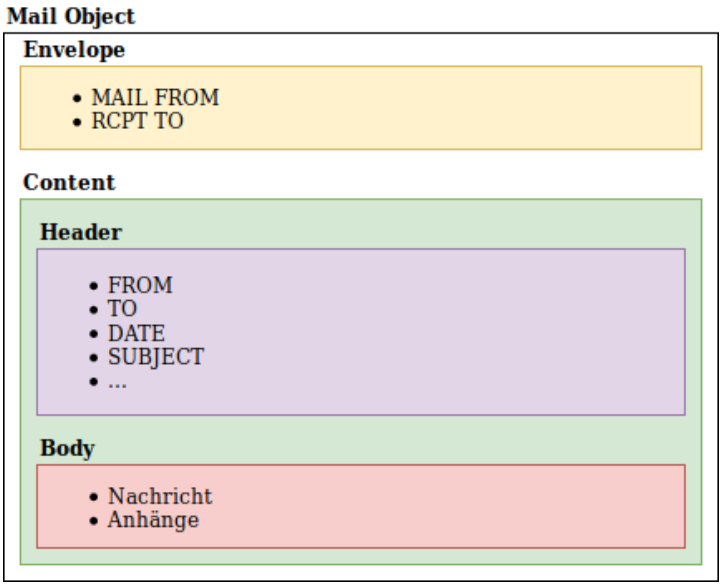

Abbildung 1: Bestandteile eines Mail Objects

Empfängt ein E-Mail-Server eine E-Mail, untersucht er den Envelope der E-Mail. Mit den entnommenen Informationen entscheidet er, wie er weiter mit ihr umgeht. Dies hat den Vorteil, dass der SMTP-Dienst auf dem Server lediglich einen kleinen, nach Vorgaben strukturierten, Teil betrachten muss, um zu entscheiden, ob er die E-Mail weiterleitet, annimmt oder ablehnt.

Listing 1: Mail Object Envelope

```
MAIL FROM: mailcrypt@mailcrypt.test
RCPT TO: mailcrypt@mailcrypt.test
```

Listing 2: Mail Object Content: Header-Abschnitt

```
Return-Path: <mailcrypt@mailcrypt.test>
Delivered-To: mailcrypt@mailcrypt.test
Received: from [192.168.66.1] (unknown [192.168.66.1])
    (Authenticated sender: mailcrypt@mailcrypt.test)
    by mailcrypt.test (Postfix) with ESMTPSA id 1F9D0210C6
    for <mailcrypt@mailcrypt.test>; Sat, 22 Nov 2018 08:54:10 +0000 (GMT)
To: mailcrypt@mailcrypt.test
From: paul <mailcrypt@mailcrypt.test>
Subject: Das ist der Betreff
Message-ID: <c89d2d13-bfc2-4dc9-b736-4293c7a637b5@mailcrypt.test>
Date: Sat, 22 Nov 2018 10:54:10 +0200
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
 Thunderbird/52.9.1
MIME-Version: 1.0
Content-Type: text/plain; charset=utf-8; format=flowed
Content-Transfer-Encoding: 8bit
Content-Language: en-US
```

Listing 3: Mail Object Content: Body-Abschnitt

```
Das ist der Body einer E-Mail.
```

Listing 4: Mail Object Content

```
Return-Path: <mailcrypt@mailcrypt.test>
Delivered-To: mailcrypt@mailcrypt.test
Received: from [192.168.66.1] (unknown [192.168.66.1])
    (Authenticated sender: mailcrypt@mailcrypt.test)
    by mailcrypt.test (Postfix) with ESMTPSA id 1F9D0210C6
    for <mailcrypt@mailcrypt.test>; Sat, 22 Nov 2018 08:54:10 +0000 (GMT)
To: mailcrypt@mailcrypt.test
From: paul <mailcrypt@mailcrypt.test>
Subject: Das ist der Betreff
Message-ID: <c89d2d13-bfc2-4dc9-b736-4293c7a637b5@mailcrypt.test>
Date: Sat, 22 Nov 2018 10:54:10 +0200
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101
 Thunderbird/52.9.1
MIME-Version: 1.0
Content-Type: text/plain; charset=utf-8; format=flowed
Content-Transfer-Encoding: 8bit
Content-Language: en-US

Das ist der Body einer E-Mail.
```

2.3. Protokolle

Um eine sichere Kommunikation zwischen E-Mail-Servern (Server-to-Server (S2S)) und E-Mail-Clients (Client-to-Server (C2S)) zu gewährleisten, ist der Einsatz von diversen Protokollen notwendig. Diese Protokolle sollen im folgenden Abschnitt kurz vorgestellt werden.

2.3.1. Simple Mail Transfer Protocol

Grundlage der E-Mail-Kommunikation im Internet bildet das Simple Mail Transfer Protocol (SMTP). Das Protokoll gehört zur Gruppe der Netzwerkprotokolle und ist in RFC 5321 definiert [Klensin, 2008].

In diesem Protokoll wird definiert, wie der Austausch von Mail Objects zwischen zwei Computern durchzuführen ist. SMTP wird für die Übermittlung und Weiterleitung von E-Mails im Internet, sowie zur Einspeisung von E-Mails ins Internet, eingesetzt.

Das Abrufen von E-Mails wird durch andere Protokolle geregelt. Deren bekannteste Vertreter POP3 und IMAP werden im Anschluss vorgestellt.

2.3.2. Post Office Protocol - Version 3

Das Post Office Protocol - Version 3 (Post Office Protocol - Version 3 (POP3)) standardisiert eine Möglichkeit, wie E-Mails von einem E-Mail-Server abgerufen werden können. Es gehört zur Gruppe der Netzwerkprotokolle und ist in RFC 1939 definiert [Myers and Rose, 1996].

Beim Abruf von E-Mails über POP3, werden diese in der Regel vollständig vom E-Mail-Server heruntergeladen und anschließend auf diesem gelöscht. Daraufhin sind sie nur noch auf dem E-Mail-Client verfügbar, der die E-Mails abgerufen hat.

Bei der Verwendung von POP3 findet die Synchronisation der E-Mails nur in eine Richtung, vom E-Mail-Server zum E-Mail-Client, statt.

2.3.3. Internet Message Access Protocol

Im Gegensatz zu POP3 synchronisiert das Internet Message Access Protocol ein Postfach auf dem E-Mail-Server mit dem Postfach auf dem E-Mail-Client. Im Protokoll ist

hierbei festgelegt, wie ein Nutzer auf E-Mails in seinem Postfach zugreifen und diese verändern kann. Das Protokoll gehört zur Gruppe der Netzwerkprotokolle und ist in RFC 3501 definiert [Crispin, 2003].

Bei der Verwendung von Internet Message Access Protocol (IMAP) wird eine lokale Kopie des serverseitigen E-Mail-Postfachs auf den E-Mail-Client(s) des Nutzers angelegt. Die lokale Kopie und das entfernte E-Mail-Postfach werden dauerhaft miteinander synchronisiert. Im Gegensatz zu POP3 verbleiben die E-Mails hierbei auch auf dem Server des E-Mail-Providers. Deswegen ist eine einfache Verwaltung eines E-Mail-Postfachs, auf mehreren Endgeräten gleichzeitig, möglich.

2.3.4. Transport Layer Security

Transport Layer Security (TLS) ist ein hybrides Verschlüsselungsprotokoll, welches unter anderem zur Absicherung von Datenübertragungen im Internet eingesetzt wird. Es ist eine Weiterentwicklung des Secure Sockets Layer (SSL)-Protokolls [Rescorla, 2018].

Das TLS-Protokoll garantiert zum einen eine verschlüsselte Datenübertragung zwischen zwei Kommunikationspartnern, und sorgt zum anderen für eine gegenseitige Authentifizierung der beteiligten Kommunikationspartner. Zum Verbindungsaufbau, zur Authentifizierung der beteiligten Kommunikationspartner und zur Übertragung eines symmetrischen Sitzungsschlüssels, kommen hierbei asymmetrische Verschlüsselungsverfahren zum Einsatz. Die zu übertragenden Daten werden mit Hilfe des symmetrischen Sitzungsschlüssels senderseitig verschlüsselt und empfängerseitig entschlüsselt.

Die zuvor vorgestellten Protokolle zur E-Mail-Übertragung werden in der Regel innerhalb einer TLS-abgesicherten Verbindung eingesetzt. Dies wird auch von der Internet Engineering Task Force (IETF) empfohlen [Moore and Newman, 2018].

Bekannte Implementierungen des TLS-Protokolls sind OpenSSL[7] und LibreSSL[8].

2.4. E-Mail-Architektur

Bei genauerer Betrachtung lässt sich feststellen, dass eine grobe Unterteilung der E-Mail-Kommunikation in die Akteure E-Mail-Server und E-Mail-Client unzureichend ist. Die beiden Akteure setzen sich aus einer Vielzahl von Komponenten, den sogenannten

[7] `https://www.openssl.org/`
[8] `https://www.libressl.org/`

„Message Agents“, zusammen. Diese sollen im Folgenden beschrieben werden. Die Gesamtheit aller Komponenten wird als „E-Mail-Architektur“ bezeichnet und ist in RFC 5598 definiert [Crocker, 2009] .

Im Folgenden soll ein Überblick geboten werden, welche Komponenten zum Transport einer E-Mail vom Sender zum Empfänger notwendig sind.

Zusätzlich zu den in RFC 5598 definierten Komponenten, wird in diverser Literatur eine weitere Komponente, der sogenannte „Message-“ oder „Mail Retrieval Agent“ eingeführt [McCarthy, 2018] [BSI, 2009]. Da ein Message Retrieval Agent eine zusätzliche Funktionalität abbildet, erscheint diese Ergänzung sinnvoll.

Daraus ergibt sich abschließend die folgende Struktur einer E-Mail-Architektur:

- E-Mail-Client:
  - Message User Agent
  - Message Retrieval Agent
- E-Mail-Server:
  - Message Submission Agent
  - Message Transfer Agent
  - Message Delivery Agent
  - Message Store

Die klare Unterscheidung der Komponenten gemäß ihrer spezifischen Aufgabe ist für eine bessere Übersicht notwendig. In der Realität müssen die Komponenten nicht so klar voneinander getrennt sein. Mehrere Komponenten können zum Beispiel innerhalb eines einzigen Software-Produkts realisiert werden.

Im Folgenden wird die Funktion aller aufgezählten Bestandteile und deren Zusammenspiel erklärt.

2.4.1. Message User Agent

Als Message User Agent (MUA) bezeichnet man die Software, mit der der Anwender E-Mails empfangen, betrachten, organisieren und versenden kann. Umgangssprachlich wird ein MUA auch als „E-Mail-Client“ oder „Mail-Client“ bezeichnet.

Ein Message User Agent vereint in den meisten Fällen die Fähigkeiten zum Überstellen von E-Mails an einen Message Submission Agent (MSA) und die Fähigkeiten zum Abrufen von E-Mails via Message Retrieval Agent (MRA) in einem Programm.

Zu den MUAs zählen zum Beispiel Computer-Programme (z.B. Mozilla's Thunderbird[9]), Smartphone-Apps (z.B. K9-Mail[10]) oder Webmail-Anwendungen auf den Servern eines E-Mail-Service-Providers (z.B. Roundcube[11]).

2.4.2. Message Submission Agent

Ein Message Submission Agent (MSA) hat die Aufgabe, E-Mails von einem Message User Agent entgegenzunehmen und diese an einen Message Transfer Agent (MTA) zu übergeben. Zur Übergabe wird das Message Submission Protokoll eingesetzt [Gellens and Klensin, 2011].

Vor der Annahme einer E-Mail überprüft der MSA, ob der Nutzer überhaupt authentifiziert und autorisiert ist, eine E-Mail über den jeweiligen Dienst zu versenden [Siemborski and Melnikov, 2007].

Weiterhin überprüft der MSA, ob die übergebene E-Mail den RFC-Vorgaben entspricht [Resnick, 2008]. Bei Abweichungen ist er befugt, fehlerhafte Angaben zu korrigieren, zusätzlich benötigte Informationen hinzufügen, oder die Annahme der E-Mail zu verweigern [Gellens and Klensin, 2011].

Zusätzlich sorgt der MSA dafür, dass serverseitige Richtlinien eingehalten werden. Eine Richtlinie könnte zum Beispiel sein, dass ein E-Mail-Server das Senden von E-Mails nur an bestimmte Empfänger erlaubt. Verstößt die übergebene E-Mail gegen eine dieser Richtlinien, lehnt der MSA die Annahme ab.

Da ein Message Submission Agent historisch gesehen eine Erweiterung von Message Transfer Agents ist [Gellens and Klensin, 2011], werden beide häufig in einem Programm realisiert und sind nicht explizit voneinander getrennt. Dies ist zum Beispiel bei der Software Postfix[12] der Fall.

2.4.3. Message Transfer Agent

Der Message Transfer Agent (MTA) ist das Herzstück der E-Mail-Kommunikation. Seine zentrale Aufgabe ist es, eine E-Mail zu ihrem Ziel zu transportieren. Ein MTA wird auch als „Mail Transfer Agent“ oder „Mail Relay“ bezeichnet.

[9] `https://www.thunderbird.net`
[10] `https://k9mail.github.io/`
[11] `https://roundcube.net/`
[12] `http://www.postfix.org/`

MTAs leiten eine E-Mail solange an andere MTAs weiter, bis diese den finalen MTA erreicht hat, der für die Empfänger-E-Mail-Adresse verantwortlich ist. Während dieses SMTP-Zustellvorgangs betrachtet jeder beteiligte MTA den Envelope der E-Mail (siehe 2.2), um festzustellen, ob diese in seine Zuständigkeit fällt. Ist dies nicht der Fall, überstellt er die E-Mail an den nächst zuständigen MTA. Der letzte MTA in dieser Kette stellt die E-Mail zu, indem er sie an den Message Delivery Agent (MDA) übergibt.

Im Gegensatz zum MSA führt ein MTA keine substantiellen Änderungen an der E-Mail durch. Lediglich Änderungen der Codierung, als auch das Hinzufügen von Sendungsverfolgungsinformationen sind zulässlig [Crocker, 2009].

Bekannte Software-Vertreter von MTAs sind zum Beispiel Exim[13] und Postfix[14].

2.4.4. Message Delivery Agent

Ein Message Delivery Agent (MDA) nimmt die E-Mails vom MTA entgegen und sortiert diese in das Postfach des entsprechenden Nutzers ein. Dies geschieht auf den Servern des E-Mail-Providers.

Beim Einsortieren der E-Mails berücksichtigt der MDA verschiedene Einsortierungsvorschriften, um die angenommenen E-Mails im richtigen Postfach abzulegen [Heinlein, 2014, S. 528]. Hierzu zählen zum Beispiel hinterlegte E-Mail-Aliase oder anzuwendende Sieve-Filter [Guenther and Showalter, 2008].

Ein bekannter Software-Vertreter von eigenständigen MDAs ist zum Beispiel procmail[15].

2.4.5. Message Store

Im Message Store (MS) werden E-Mails dauerhaft gespeichert. Dieser ist gleichbedeutend mit dem Postfach des Nutzers. Der MS befindet sich in der Regel auf dem Server des E-Mail-Providers. Der MS kann zusätzlich auch auf weiteren E-Mail-Clients abgespeichert werden. Umgangssprachlich wird ein MS, auch als „E-Mail-Postfach“ oder „Mailbox“ bezeichnet.

Die Synchronisierung zwischen den Message Stores auf dem E-Mail-Server und dem E-Mail-Client erfolgt in der Regel mit Hilfe eines POP3-/IMAP-Servers (siehe 2.4.6).

[13] `https://www.exim.org/`
[14] `http://www.postfix.org/`
[15] `https://linux.die.net/man/1/procmail`

2.4.6. POP3- und IMAP-Server

Um Nutzern Zugriff auf ein entferntes E-Mail-Postfach (Message Store) zu ermöglichen, stellen die meisten E-Mail-Provider einen POP3-/IMAP-Server bereit. Darüber lassen sich die Inhalte des Server-Postfachs abrufen bzw. synchronisieren.

Ein bekannter Software-Vertreter von POP3-/IMAP-Servern ist zum Beispiel Dovecot[16].

2.4.7. Message Retrieval Agent

Ein Message Retrieval Agent (MRA) ist Teil des E-Mail-Clients. Er hat die Aufgabe, E-Mails von einem E-Mail-Server abzuholen und diese in einem lokalen Postfach zu speichern. Dabei arbeitet der MRA in der Regel mit einem POP3-/IMAP-Server zusammen.

Häufig unterstützt ein Message Retrieval Agent diverse Protokolle zum Abholen von E-Mails. Dabei sind IMAP [Crispin, 2003] und POP3 [Myers and Rose, 1996] die geläufigsten.

Häufig sind die Funktionen eines MRAs in den Message User Agent integriert. Ein zusätzliches Programm zum Abholen von E-Mails ist in der Regel nicht notwendig.

Nichtsdestotrotz kann ein MRA auch als eigenständiges Programm betrieben werden. Ein bekannter Vertreter von MRAs ist das Programm getmail[17].

2.4.8. Zusammenspiel der Komponenten

Abschließend soll das Zusammenspiel der zuvor beschriebenen Akteure und deren Komponenten visualisiert werden. In Abbildung 2 ist der Weg einer E-Mail vom Sender bis zum Empfänger dargestellt. Hierbei werden die beteiligten Komponenten und die genutzten Protokolle visualisiert.

Es handelt sich hierbei um eine stark vereinfachte Darstellung. Zum einen können Zwischenstationen, komplexere E-Mail-Server-Lösungen, sowie mögliche Spezialfälle nicht abgebildet werden, zum anderen entspricht die dargestellte scharfe Trennung der einzelnen Message Agents nicht immer, der in Software abgebildeten, Realität. Zum Beispiel müssen MDA und POP3-/IMAP-Server keine unterschiedlichen Programme sein, und nicht immer enthält ein MUA einen expliziten MRA, oder einen lokalen MS.

[16] `https://www.dovecot.org/`
[17] `http://pyropus.ca/software/getmail/`

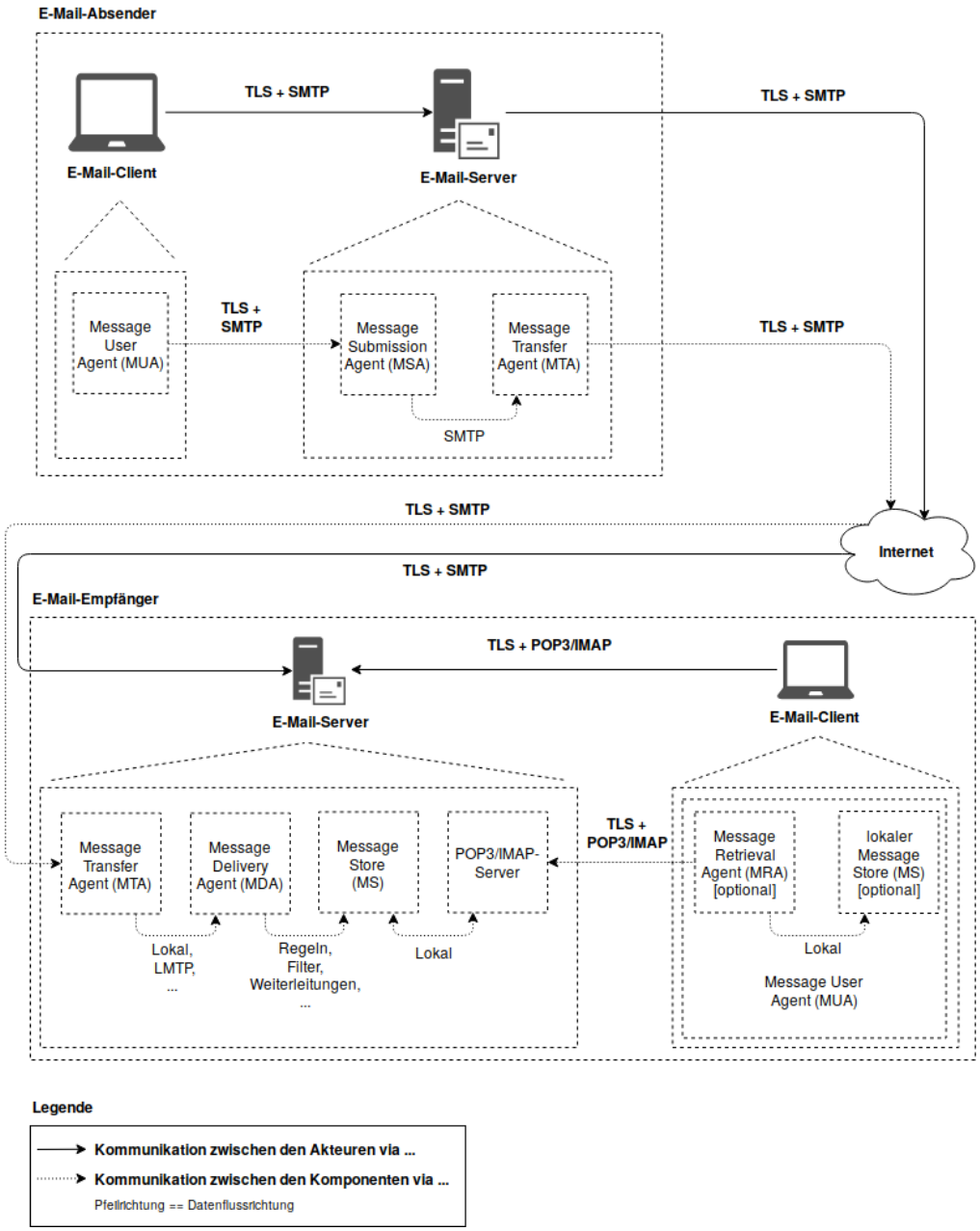

Abbildung 2: Komponenten der E-Mail-Kommunikation

2.5. Kryptographische Grundlagen

Zum besseren Verständnis der zu untersuchenden Postfachverschlüsselungsplugins sind, neben den Grundlagen der E-Mail-Kommunikation, zusätzliche Kenntnisse über kryptographische Konzepte und Techniken hilfreich. Diese sollen im Laufe dieses Unterkapitels eingeführt werden. Hierbei wird näher auf die verschiedenen Verschlüsselungsarten, Data-at-Rest-Verschlüsselung, sowie das Hashen von Passwörtern eingegangen.

2.5.1. Symmetrische Verschlüsselung

Zu den symmetrischen Verschlüsselungsverfahren zählen alle Verfahren, bei denen zur Ver- und Entschlüsselung der Daten der gleiche geheime Schlüssel verwendet wird. Dieser muss sowohl dem Sender, als auch dem Empfänger der verschlüsselten Nachricht, bekannt sein. Als Synonyme sind auch die Bezeichnungen Secret-Key-Kryptografie und Secret-Key-Verschlüsselung üblich.

Die kryptographischen Funktion von symmetrische Verschlüsselungsverfahren lassen sich in der Regel sehr gut in Hardware abbilden. Dadurch können sehr hohe Geschwindigkeiten bei der Ver- und Entschlüsselung erreicht werden [Gueron, 2012].

Bei dieser Art von Verfahren ist der Austausch des geheimen Schlüssels eine der größten Herausforderungen. Dieser muss über einen sicheren Kanal zwischen Sender und Empfänger ausgetauscht werden. Dies wird auch als das sogenannte „Schlüsselverteilungsproblem“ bezeichnet.

Ein bekannter Vertreter der symmetrischen Verschlüsselungsalgorithmen ist der Advanced Encryption Standard (AES)[18].

2.5.2. Asymmetrische Verschlüsselung

Bei asymmetrischen Verschlüsselungsverfahren wird zur Ver- und Entschlüsselung der Daten ein Schlüsselpaar verwendet. Das Schlüsselpaar besteht aus einem öffentlichen und einem geheimen Schlüssel. Beide hängen über einen mathematischen Algorithmus miteinander zusammen. Als Synonyme für diese Art von Verschlüsselung sind auch die Bezeichnungen Public-Key-Kryptografie und Public-Key-Verschlüsselung üblich.

In der Regel besitzen Sender und Empfänger ein eigenes Schlüsselpaar. Der Empfänger der zu verschlüsselnden Nachricht hat den öffentlichen Anteil seines Schlüsselpaares dem Sender mitgeteilt. Der Sender verschlüsselt eine Nachricht mit Hilfe des öffentlichen Schlüssels des Empfängers. Nach Erhalt der verschlüsselten Nachricht kann der Empfänger die Nachricht mit Hilfe seines geheimen Schlüssels entschlüsseln.

Bei dieser Art von Verschlüsselungen wird das Schlüsselverteilungsproblem umgangen, da beide Parteien den öffentlichen Schlüssel ungehindert verbreiten können. Der geheime Teil des Schlüsselpaares darf hingegen nur dessen Besitzer bekannt sein.

[18] `https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf`

Im Gegensatz zur symmetrischen Verschlüsselung, ist die asymmetrische Verschlüsselung häufig ressourcen-intensiver und deutlich langsamer. Sie eignet sich deswegen nicht zur Verschlüsselung von großen Datenmengen.

Ein bekannter Vertreter der asymmetrischen Verschlüsselungsalgorithmen ist Rivest-Shamir-Adleman (RSA)[19]. Er findet unter anderem Anwendung in der Software Secure Shell (SSH)[20].

Um die Vorteile von symmetrischer und asymmetrischer Verschlüsselung zu erhalten, können beide Verfahren miteinander kombiniert werden. Die Kombination wird als „hybride Verschlüsselung“ bezeichnet.

2.5.3. Hybride Verschlüsselung

Bei dieser Art von Verfahren erzeugt der Sender einen symmetrischen Schlüssel. Mit diesem wird eine Nachricht verschlüsselt. Der symmetrische Schlüssel wird wiederum, mit dem öffentlichen Schlüssel des Empfängers der Nachricht, verschlüsselt. Daraufhin wird sowohl der verschlüsselte Schlüssel, als auch die verschlüsselte Nachricht an den Empfänger übermittelt.

Der Empfänger entschlüsselt, mit Hilfe seines geheimen Schlüssels, den zur Verschlüsselung eingesetzten symmetrischen Schlüssel. Mit diesem ist es ihm möglich, die verschlüsselte Nachricht zu entschlüsseln.

Bekannte Vertreter, bei denen hybride Verschlüsselung eingesetzt wird, sind zum Beispiel die Software GnuPG[21] oder das TLS-Protokoll[22].

[19] `https://patents.google.com/patent/US4405829`
[20] `https://www.openssh.com/`
[21] `https://gnupg.org/`
[22] `https://www.openssl.org/`

2.5.4. Data-at-rest-Verschlüsselung

Zur Zustandsbeschreibung von digitalen Daten verwendet man in der Informationstechnologie häufig drei Bezeichnungen:

- Data-in-use
  - Hierzu gehören alle Daten, die sich in Verwendung befinden. Diese Daten werden auch als „aktive Daten“ bezeichnet und befinden sich in der Regel in einem kontinuierlichen Zustand der Veränderung. Zu dieser Gruppe können zum Beispiel die Daten in einer Datenbank zählen. In der Regel liegen diese Daten in den flüchtigen Speichern eines Computers, wie zum Beispiel RAM, CPU-Cache oder den CPU-Registern.
- Data-in-transit
  - Unter dieser Bezeichnungen werden die Daten zusammengefasst, welche sich durch Netzwerke bewegen.
- Data-at-rest
  - Unter dieser Bezeichnung werden alle Daten zusammengefasst, die aktuell nicht in Verwendung und zusätzlich auf einem nicht-flüchtigen, physikalischen Medium gespeichert sind. Diese werden auch als „inaktive Daten“ bezeichnet.

Dem folgend bezeichnet der Begriff „Data-at-rest-Verschlüsselung“ alle „inaktiven“ Daten, die verschlüsselt sind und auf einem nicht-flüchtigen, physikalischen Medium gespeichert sind. Ein bekannter Anwendungsfall von „Data-at-rest-Verschlüsselung“ ist zum Beispiel die Festplattenvollverschlüsselung eines Computers.

2.5.5. Hashes und Passwort-Hashing

Um Zugriff auf entfernte Netzwerkdienste zu erhalten, verlangt ein Server-Betreiber häufig eine Kombination aus Nutzernamen und Passwort. Damit der Server-Betreiber den Nutzer authentifizieren kann, muss auch er im Besitz des Nutzer-Passworts sein. Zum Schutz des Nutzer-Passworts vor unberechtigten Dritten sollte das Passwort auf Seiten des Server-Betreibers nicht im Klartext gespeichert werden. Der Dienst-Anbieter sollte lediglich den „Hash“ des Passworts speichern.

2.5.5.1. Hash

Hash-Funktionen bilden eine beliebig große Eingabemenge auf eine Zielmenge definierter Größe ab. Die Zielmenge wird als sogenannter Hashwert (kurz: Hash) bezeichnet. Hash-Funktionen erhalten als Eingabemenge zum Beispiel ein Klartext-Passwort und berechnen den zugehörigen Passwort-Hash.

Zur Berechnung eines Hashes werden mathematische Einwegfunktionen verwendet. Einwegfunktionen zeichnen sich durch eine besondere Eigenschaft aus. Sie sind leicht zu berechnen, aber schwer umkehrbar. Das bedeutet, es ist mit wenigen Ressourcen möglich, den Hash eines Klartext-Passworts zu berechnen, aber es ist aufwendig, oder nicht möglich, aus einem gehashten Passwort, dessen Klartext-Pendant zu berechnen.

Zur Berechnung von Passwort-Hashes werden kryptographische Hash-Funktionen verwendet. Diese haben die zusätzliche Eigenschaft, dass sie kollisionsresistent sind. Kollisionsresistenz bedeutet, dass es sehr schwer möglich ist, aus zwei unterschiedlichen Eingabewerten, einen identischen Hash-Wert zu berechnen.

2.5.5.2. Salt

Ein Salt ist eine zufällig generierte Zeichenkette, mit dem das Passwort kombiniert wird, bevor es an eine Hash-Funktion übergeben wird. Im Gegensatz zum Passwort muss das Salt nicht geheimgehalten werden. Es wird häufig mit dem generierten Passwort-Hash kombiniert und danach abgespeichert (siehe Listing 5).

Ein Salt wird als indirekter Schutz für das Klartext-Passwort verwendet. Es erschwert Wörterbuch-Angriffe bzw. deren gehashtes Äquivalent, den Rainbow-Table-Angriff. Ein Angreifer müsste für jedes Salt ein individuelles Wörterbuch erstellen, anstatt ein einziges Wörterbuch zum Nachschlagen der gespeicherten Passwörter, bzw. Passwort-Hashes verwenden zu können.

Listing 5: Beispielhafte Erstellung eines Passwort-Hashes

```
vagrant@mailcrypt.test:~ $ TMP_PWD=Password; SALT=RandomSalt; echo -ne \nPass: $TMP_PWD\
    nSalt: $SALT\nHash: ; mkpasswd --method=sha-256 --salt=$SALT $TMP_PWD;

Pass: Password
Salt: RandomSalt
Hash: $5$RandomSalt$QQwbA3HvZ88wZaaYwzHXZNr5dUHCLen/.6dOxPga2iC
```

2.6. Eingesetzte Software

Zum Abschluss des Grundlagen-Kapitels soll in diesem Unterkapitel die Software vorgestellt werden, die im Rahmen dieser Arbeit eingesetzt wird.

2.6.1. Postfix

Postfix[23] ist ein bekannter Vertreter von Message Transmission Agents (MTA) und Message Submission Agents (MSA). Nach Exim[24] ist es die am weitverbreitetste MTA/MSA-Software [securityspace.com, 2019].

Postfix bietet einen großen Funktionsumfang und ist modular aufgebaut [Heinlein, 2008, S. 91]. Die Software läuft in einer abgeschlossenen „chroot"-Umgebung, welche potentielle Angriffe erschwert und es dadurch für Angreifer uninteressant macht. [Heinlein, 2008, S.92].

Postfix wird in dieser Arbeit als leicht zu konfigurierende MTA-MSA-Kombination betrieben. Es dient jedoch lediglich als Hilfsmittel, um den MDA Dovecot (siehe 2.6.2) anzusprechen. Dazu kommuniziert Postfix mit dem, in Dovecot integrierten, „Local Delivery Agent (LDA)".

2.6.2. Dovecot

Die Open-Source-Software Dovecot[25] gehört zur Gruppe der Message Delivery Agents (MDA) und stellt gleichzeitig einen POP3/IMAP-Server zur Verfügung. Im Gegensatz zu ähnlichen Programmen, ist Dovecot's Konfiguration flexibler und dessen Verarbeitungsgeschwindigkeit, bei größerem Funktionsumfang, höher [Heinlein, 2014]. Dovecot kann als der Standard-MDA angesehen werden, da er mit Abstand der am weitesten verbreitete IMAP/POP3-Server ist [openemailsurvey.org, 2018].

Die Entwicklung von Dovecot erfolgt nach eigener Aussage mit einem besonderen Sicherheitsfokus. Die schlägt sich auch in der Selbstbezeichnung „Secure IMAP server" nieder [dovecot.org, 2018]. Im Rahmen eines Sicherheits-Audits durch Cure53[26] wurden Teile der Code-Basis auf Sicherheitslücken untersucht. Hierbei wurden nur wenige, nicht-kritische Sicherheitslücken gefunden und Dovecot eine sichere Code-Basis bescheinigt [Heiderich et al., 2016].

[23] `http://www.postfix.org/`
[24] `https://www.exim.org/`
[25] `https://www.dovecot.org/`
[26] `https://cure53.de/`

Ein funktionierender Dovecot-Server ist die Grundvoraussetzung für die Durchführung dieser Arbeit. Alle untersuchten Postfachverschlüsselungstechniken sind Dovecot -Plugins. Die Begriffe Postfachverschlüsselungstechnik und Postfachverschlüsselungsplugin werden synonym verwendet.

2.6.3. Ansible

Ansible[27] ist ein Open-Source-Konfigurationsmanagementsystem, mit dessen Hilfe eine einfache Orchestrierung und Konfiguration von vielen entfernten Computern bzw. Servern möglich ist. Neben einer zentralen Administration, ist es mit Ansible möglich, viele Formen von komplexen IT-Abläufen automatisiert abzubilden. Hierzu zählen zum Beispiel die Provisionierung (automatische Einrichtung) eines Servers.

Im Gegensatz zu anderen Konfigurationsmanagement-Systemen (z.B. Puppet[28] oder Chef[29]), benötigt Ansible lediglich eine SSH-Verbindung, sowie eine minimale Python-Umgebung[30], um entfernte Systeme zu verwalten.

Grundbaustein von Ansible sind sogenannte „Playbooks“. Diese sind eine Zusammenstellung von auszuführenden Aufgaben, sogenannte „Tasks“. Playbooks werden in der Markup-Sprache YAML[31] verfasst und sind dadurch einfach lesbar. Das von Ansible verwendete Template Engine Jinja2[32] ermöglicht die geschickte Ersetzung von Variablen in Dateien. Dadurch ist es unter anderem möglich, ein Konfigurationstemplate für verschiedenen Server zu verwenden.

Im Rahmen dieser Arbeit wurden Ansible-Playbooks entwickelt, welche die Erstellung und Konfiguration der eingesetzten E-Mail-Server automatisieren. Hierbei kommen je nach Technik unterschiedliche Playbooks zum Einsatz. Durch den Einsatz von Ansible ist die Konfiguration der eingesetzten E-Mail-Server nachvollziehbar und reproduzierbar.

2.6.4. VirtualBox

VirtualBox[33] ist eine Open-Source-Virtualisierungslösung, mit deren Hilfe man auf einem Computer (Host-System) virtuelle Maschinen (VM) betreiben kann. Dazu emuliert

[27] `https://www.ansible.com`
[28] `https://puppet.com/`
[29] `https://www.chef.io/chef/`
[30] `https://www.python.org/`
[31] `http://yaml.org/`
[32] `http://jinja.pocoo.org/`
[33] `https://www.virtualbox.org/`

die Software (der Hypervisor) auf dem Host-System die Hardware eines virtuellen Computers (Guest-System). In dieser kann der Anwender ein unterstütztes Betriebssystem seiner Wahl installieren.

VirtualBox setzt hierbei auf sogenannte „vollständige Virtualisierung“. Das bedeutet, dass der Hypervisor zahlreiche Hardware-Komponenten virtualisiert und das Guest-System sich als ein unabhängiges, abgeschlossenes System betrachtet. Dadurch können auf dem Guest-System auch Betriebssysteme installiert werden, die sich vom Betriebssystem des Host-Systems unterscheiden.

Im Rahmen dieser Arbeit fungiert VirtualBox als VM-Provider. Die Software wird von Vagrant angesprochen (siehe 2.6.5) und stellt die Grundlage zum Betrieb der eingesetzten Testumgebung dar.

2.6.5. Vagrant

Zur Erstellung der Testumgebung wird die Software Vagrant[34] eingesetzt. Sie ermöglicht die Erstellung, Verwaltung und Provisionierung von virtuellen Maschinen. Vagrant fungiert hierbei als Schnittstelle zwischen dem Konfigurationsmanagementsystem Ansible (2.6.3) und der Virtualisierungssoftware VirtualBox (2.6.4).

Die Konfiguration erfolgt über ein sogenanntes „Vagrantfile“. In diesem wird festgelegt welche virtuellen Maschinen erstellt werden und welcher Provisionierungsdienst verwendet wird. Weiterhin kann dort festgelegt werden, welche Ports zwischen dem Host-System und den virtuellen Maschinen weitergeleitet werden. Beim Aufruf von Vagrant werden die definierten virtuellen Maschinen erstellt, der Provisionierungsdienst aufgerufen, sowie die Portweiterleitungen zwischen Host- und Gast-Systemen automatisch eingerichtet.

Durch den Einsatz von Vagrant ist es möglich eine Testumgebung zu entwerfen, die sich schnell und einfach erstellen und wieder zerstören lässt. Zusätzlich ist sie über Betriebssystemgrenzen hinweg portabel.

2.6.6. Prometheus

Prometheus[35] ist eine von SoundCloud[36] entwickelte Monitoring-Lösung. Zur Datenhaltung setzt Prometheus auf eine „Time Series Database“. In ihr werden die Monitoring-Daten und der zugehörige Zeitstempel abgespeichert. Dies ermöglicht eine einfache zeit-

[34] `https://www.vagrantup.com/`
[35] `https://prometheus.io/`
[36] `https://soundcloud.com`

liche Auswertung der Daten, ohne auf Zusatzsoftware zurückgreifen zu müssen [Loschwitz, 2016]. Zusätzlich verfügt Prometheus über eine Programmierschnittstelle (API), mit deren Hilfe die Monitoring-Daten abgefragt werden können.

Im Rahmen dieser Arbeit wird Prometheus eingesetzt, um Performance-Daten von den eingesetzten virtuellen Maschinen aufzuzeichnen und zu analysieren.

2.6.7. Grafana

Zur Visualisierung der Monitoring-Daten wird die Software Grafana[37] eingesetzt. Für Grafana werden sogenannte „Dashboards“ entwickelt. Diese enthalten Graphen, die die von Prometheus aufgezeichneten Daten (siehe 2.6.6), darstellen. Mit Hilfe der Graphen wird der laufende Betrieb der virtuellen Maschinen überwacht.

2.7. Zusammenfassung

In diesem Kapitel wurden die Grundlagen eingeführt, die zum Verständnis dieser Arbeit hilfreich sind. Hierbei wurden die E-Mail, die E-Mail-Architektur, sowie die notwendigen Kommunikations- und Sicherheitsprotokolle erläutert. Danach wurden die kryptographischen Grundlagen beschrieben, bevor abschließend die eingesetzte Software vorgestellt wurde.

[37] `https://grafana.com/`

3. Stand der Technik

3.1. Einleitung

In diesem Kapitel soll die Technik der persönlichen E-Mail-Postfachverschlüsselung vorgestellt werden. Zur Motivation werden Angriffsszenarien auf E-Mail-Kommunikationsinhalte und Nutzerdaten skizziert. Weiterhin werden gängige Schutzmaßnahmen beschrieben, die ein E-Mail-Server-Betreiber treffen kann, um die Daten seiner Nutzer zu schützen. Es wird gezeigt, dass diese Maßnahmen nur unzureichend vor den beschriebenen Angriffsszenarien schützen. Schlussendlich wird die Technik der persönlichen E-Mail-Postfachverschlüsselung vorgestellt, die als zusätzliche Schutzmaßnahme die beschriebenen Angriffe verhindern, bzw. erschweren kann.

3.1.1. Aktive und passive Angreifer

Die Sicherheit der E-Mail-Kommunikationsinhalte und der Nutzer-Zugangsdaten kann von zwei Angreifertypen, einem aktiven und einem passiven Angreifer, bedroht werden.

Der aktive Angreifer verfügt über mindestens einen kurzfristigen oder dauerhaften Zugang zur E-Mail-Infrastruktur des E-Mail-Providers. Für die Dauer des Angriffs hat er die nötigen Zugriffsrechte, um auf das gesamte Dateisystem des E-Mail-Servers zuzugreifen und dieses zu kopieren. Weiterhin hat der Angreifer auch vollen Zugriff auf die Nutzerdatenbank des E-Mail-Servers und kann auch diese kopieren. Da sich der Angreifer nun im Besitz der Nutzerdatenbank befindet, kann er eine sogenannte „Offline-Attacke“ durchführen. Hierbei versucht er, ohne zeitliche Einschränkungen, die Klartext-Passwörter aus den kopierten Passwort-Hashes zu berechnen.

Bei dieser Art von Angreifer kann es sich zum Beispiel um einen unbefugten Hacker, oder einen befugten Systemadministrator des E-Mail-Providers handeln [Peterson, 2013] [Selyukh, 2013] [Cox, 2018].

Neben dem aktiven Angreifer gibt es einen passiven, häufig staatlich legitimierten Angreifer. Diese gesetzeskonforme Überwachung wird auch als „Lawful Interception“ bezeichnet. Bei dieser Art von Angriff kann eine berechtigte Entität den E-Mail-Provider zur Übergabe von bestimmten E-Mail-Postfächern zwingen. Dieser Angreifer hat hierbei keinen direkten Zugriff auf das Dateisystem des E-Mail-Servers, sondern kann den E-Mail-Provider indirekt als Mittelsmann einsetzen, um an die gewünschten Datensätze zu kommen. Dieser Vorgang wird in Deutschland als Postfachbeschlagnahmung bezeichnet. Weiterhin kann der passive Angreifer den E-Mail-Provider dazu bringen, in

den E-Mail-Verkehr von bestimmten Nutzern einzugreifen. Er kann den E-Mail-Provider für einen bestimmten Zeitraum verpflichten, ein- und ausgehende E-Mails zu kopieren und an einem Ablageort seiner Wahl zu deponieren. Diesen Vorgang bezeichnet man in Deutschland als Telekommunikationsüberwachung.

Passive Angriffe sind in vielen Staaten gesetzlich verankert und gehören zum Ermittlungsrepertoire von staatlichen Behörden. Die Grenzen für den Einsatz dieser Methoden variieren je nach Gesetzgebung.

3.1.2. Serverseitige Schutzmaßnahmen

Ein E-Mail-Server-Betreiber kann diverse Maßnahmen ergreifen, um die ihm anvertrauten Daten vor diesen Angreifern zu schützen. Hierzu zählen sowohl die E-Mail-Inhalte, als auch die Zugangsdaten der Nutzer. Diese Daten schützt ein E-Mail-Provider in der Regel, indem er die Übertragungswege absichert, das Passwort des Nutzers nur in Form eines Hashes abspeichert und den Zugang zu seinen Servern auf ein Minimum beschränkt.

Die Übertragungswege zwischen den beteiligten Kommunikationspartnern sind heutzutage in der Regel gut abgesichert [Google, 2018]. Dies geschieht durch standardisierte Protokolle und kann teilweise durch den Nutzer kontrolliert werden. Die gespeicherten E-Mails und Zugangsdaten unterliegen aber ausschließlich den Schutzmaßnahmen des jeweiligen E-Mail-Providers. Der Nutzer muss hierbei dem E-Mail-Provider vertrauen, dass dieser etwaige Schutzmaßnahmen korrekt implementiert, konfiguriert und regelmäßig evaluiert. Die Vergangenheit zeigt, dass dies nicht immer der Fall ist [Gosney, 2016] [Lord, 2016] [Lawler, 2017] [Hunt, 2019].

Die erwähnten Maßnahmen des Passwort-Hashings und der Transportverschlüsselung schützen zwar die Zugangsdaten sowie den Transportweg, an einer entscheidenden Stelle helfen sie jedoch nicht: Sobald eine E-Mail ihren Ziel-Server erreicht hat, wird sie dort im Klartext abgespeichert. Dort können aktive und passive Angreifer Zugriff auf die E-Mail-Inhalte erlangen.

Sowohl aus Nutzerperspektive, als auch aus Sicht eines datenschutz-orientierten E-Mail-Providers wäre es deswegen wünschenswert, eine zusätzliche Schutzmaßnahme zu implementieren, die die E-Mail-Inhalte auch auf dem E-Mail-Server schützt.

Eine vielversprechende Schutzmaßnahme gegen die beschriebenen Angreifer scheint die persönliche E-Mail-Postfachverschlüsselung zu sein.

3.2. Persönliche E-Mail-Postfachverschlüsselung

Die persönliche E-Mail-Postfachverschlüsselung ist eine neue, serverseitige Schutzmaßnahme und stellt eine Ergänzung zu den zuvor beschriebenen Maßnahmen dar. Mit ihr soll es möglich sein, die zuvor beschriebenen Angriffe zu verhindern bzw. zu erschweren. Im Folgenden werden vier Techniken zur persönlichen E-Mail- Postfachverschlüsselung detailliert vorgestellt.

Beim Einsatz dieser Technik werden eingehende E-Mails serverseitig verschlüsselt, bevor sie auf dem Server abgespeichert werden. Neben der Verschlüsselung sorgt die Postfachverschlüsselung zusätzlich dafür, dass der E-Mail-Server die verschlüsselten Nachrichten nicht eigenständig entschlüsseln kann. Um die abgespeicherten E-Mails zu entschlüsseln, ist nach der initialen Verschlüsselung immer eine Interaktion mit dem Nutzer, bzw. dessen Passwort notwendig.

Zur Umsetzung dieses Ansatzes setzen die Techniken zur persönlichen E-Mail-Postfachverschlüsselung auf eine Kombination aus symmetrischer und asymmetrischer Verschlüsselung. Im Rahmen dieser Arbeit sollen die folgenden Postfachverschlüsselungsplugins für Dovecot untersucht werden:

- GPG-Sieve-Filter
- MailCrypt-Plugin
- Scrambler-Plugin
- TREES-Plugin

Im Verlauf des Kapitels werden die zugrundeliegenden Funktionen und Konzepte der Plugins erläutert. Weiterhin werden die Ver- und Entschlüsselungsvorgänge, die beim Empfang und Abruf von E-Mails durchgeführt werden, beschrieben und dargestellt.

3.3. GPG-Sieve-Filter

Der GPG-Sieve-Filter setzt auf eine Kombination aus Sieve-Filtern und GnuPG[38] [Guenther and Showalter, 2008]. GnuPG ist die verbreitetste Implementation des OpenPGP-Standards und auch unter dem Namen „GPG“ bekannt. Als Sieve-Filter werden serverseitig ausgeführte Filterregeln bezeichnet, auf deren Basis empfangene E-Mails bearbeitet werden. Dazu gehören zum Beispiel automatische Abwesenheitsnachrichten oder das Einsortieren von E-Mails eines bestimmten Absenders in einen Unterordner des Postfachs. Die Filter sind in der Sprache Sieve verfasst und können in der Regel durch den Nutzer verwaltet werden [Guenther and Showalter, 2008].

[38] `https://www.gnupg.org/`

Die Verschlüsselung mit Hilfe eines GPG-Sieve-Filters wird unter anderem von den E-Mail-Anbietern Posteo [posteo.de, o.J.] und mailbox.org [mailbox.org, o.J.] angeboten.

3.3.1. Funktion

Der Nutzer kann diese Technik in der Regel selbstständig aktivieren und muss zusätzlich seinen öffentlichen GPG-Schlüssel auf den Server des E-Mail-Anbieters hochladen[39,40]. Da sie lediglich den Posteingang des Nutzers verschlüsselt, wird diese Technik auch als „Posteingangsverschlüsselung“ bezeichnet [posteo.de, o.J.].

3.3.2. Konzept

Der GPG-Sieve-Filter hat die Aufgabe, alle eingehenden E-Mails mit dem öffentlichen Schlüssel des Empfängers zu verschlüsseln.

3.3.3. E-Mail-Verschlüsselungsprozess beim E-Mail-Empfang

In Abbildung 3 ist der E-Mail-Empfang unter Verwendung des GPG-Sieve-Filter-Plugins dargestellt.

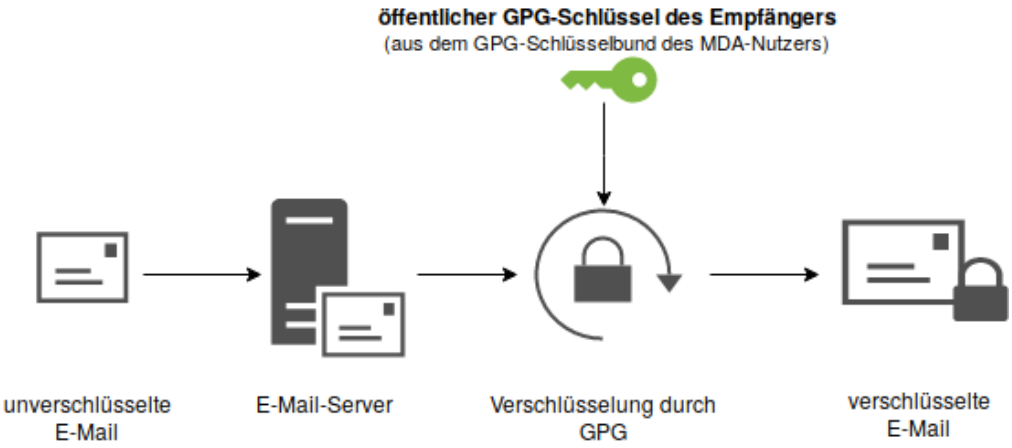

Abbildung 3: E-Mail-Verschlüsselungsprozess beim E-Mail-Empfang mit GPG-Sieve-Filter

[39] `https://systemausfall.org/wikis/howto/DovecotInboxVerschl%C3%BCsselung`
[40] `https://perot.me/encrypt-specific-incoming-emails-using-dovecot-and-sieve`

Beim Empfang einer E-Mail überprüft der Server, ob der öffentliche GPG-Schlüssel des Empfängers im GPG-Schlüsselbund des Message Delivery Agent (MDA)-Nutzers vorhanden ist. Ist dies der Fall, wird die eingehende E-Mail mit Hilfe von GPG verschlüsselt und im Postfach des Empfängers abgelegt.

Da ein Sieve-Filter nur auf E-Mails angewendet wird, die von anderen E-Mail-Servern eingeliefert werden (via SMTP), werden E-Mails, die vom Message User Agent des Nutzers kommen (via IMAP), unverschlüsselt auf dem Server abgespeichert.

Es ist wichtig zu erwähnen, dass nur der E-Mail-Body verschlüsselt wird. Die Header der E-Mail werden nicht verschlüsselt.

3.3.4. E-Mail-Entschlüsselungsprozess beim E-Mail-Abruf

Beim Abrufen der E-Mails wird durch das Plugin keine Arbeit verrichtet. Die verschlüsselten E-Mails können wie gewohnt per MUA abgerufen werden. Zur Entschlüsselung sind jedoch zusätzliche Programme nötig. Für möglichst hohen Komfort, können diese auch in den Message User Agent (MUA) integriert werden. Dies kann sowohl auf dem Computer des Nutzers[41], als auch auf den Servern des E-Mail-Anbieters[42,43] geschehen.

3.4. MailCrypt

Das Dovecot-Plugin „MailCrypt“[44] ist ein in C geschriebenes Programm, welches persönliche E-Mail-Postfachverschlüsselung in Dovecot integriert. Das Plugin ist seit 2016 Bestandteil des Dovecot-Kerns und wird von dessen Entwickler-Team aktiv betreut. Zur Verschlüsselung der eingehenden E-Mails setzt es auf die kryptographische Bilbliothek „OpenSSL“[45] und verwendet Schlüsselpaare auf Basis von elliptischen Kurven. Schlüssel, die auf dieser Art asymmetrischer Kryptographie beruhen, können deutlich kürzer sein, als Schlüssel die auf RSA beruhen und trotzdem das gleiche Maß an Sicherheit bieten [Biselli, 2018].

[41] https://www.enigmail.net/index.php/en/
[42] https://github.com/roundcube/roundcubemail/tree/master/plugins/enigma
[43] https://userforum.mailbox.org/knowledge-base/article/einfuehrung-in-mailbox-org-guard
[44] https://wiki2.dovecot.org/Plugins/MailCrypt
[45] https://www.openssl.org/

3.4.1. Funktion

Das Plugin unterstützt zwei Modi zur Verschlüsselung von E-Mails. Erstens einen Modus für sogenannte „Global Keys“, zweitens einen Modus für sogenannte „Folder Keys“ [Tuomi et al., 2016]. Im „Global Keys“-Modus gibt es ein globales Schlüsselpaar, mit welchem alle E-Mails aller Nutzer verschlüsselt werden. Dies ist etwa dann sinnvoll, wenn man E-Mails auf einen externen Speicher auslagern möchte und der Betreiber des Speichers keinen Zugriff auf die Nachrichten haben soll. Bei der Verwendung des „Folder Keys“-Modus werden für die Nutzer individuelle Schlüssel angelegt. Diese können entweder im Klartext oder verschlüsselt auf dem Server abgelegt werden.

Im Rahmen der durchgeführten Untersuchung ist lediglich der „Folder Key“-Modus, unter Einsatz von verschlüsselten Nutzer-Schlüsseln, interessant. Nur in diesem Modus sind die Voraussetzungen erfüllt, bei denen der Server-Betreiber keinen Zugriff auf das Schlüssel-Material des Nutzers und damit keinen Zugriff auf dessen E-Mails hat.

Ist das Plugin für einen bestehenden Nutzer aktiviert, werden alle zukünftig eingehenden E-Mails verschlüsselt. Die bereits existierenden E-Mails werden nicht nachträglich verschlüsselt. Sie werden ohne Plugin-Interaktion an den Nutzer durchgereicht.

3.4.2. Konzept

Das MailCrypt-Plugin verschachtelt symmetrische und asymmetrische Kryptographie zur Umsetzung von persönlicher E-Mail-Postfachverschlüsselung.

Beim ersten Nutzer-Login generiert das Plugin ein individuelles Schlüsselpaar. Dieses basiert auf elliptischen Kurven und wird im Folgenden als „Nutzer-Schlüsselpaar“ bezeichnet. Der private Anteil des „Nutzer-Schlüsselpaares“ wird verschlüsselt auf dem Server abgespeichert. Zur Verschlüsselung wird ein PBKDF2-Hash[46] verwendet. Dieser wird mit Hilfe des Nutzerpassworts gebildet. Das „Nutzer-Schlüsselpaar“ basiert in der Standardkonfiguration des Plugins auf der elliptischen Kurve „secp521r1“ [Tuomi et al., 2016].

Neben dem „Nutzer-Schlüsselpaar“ werden durch das Plugin Schlüsselpaare für jeden Postfach-Ordner (Inbox, Sent, Trash) sogenannte „Ordner-Schlüsselpaare“ generiert. Die privaten Anteile der „Ordner-Schlüsselpaare“, werden mit Hilfe des öffentlichen Schlüssel des „Nutzer-Schlüsselpaares“ verschlüsselt und auf dem Server abgespeichert.

Die Generierung aller notwendigen Schlüssel und Schlüsselpaare, sowie deren Verschlüsselung erfolgt automatisch durch das Plugin.

[46] `https://tools.ietf.org/html/rfc2898`

3.4.3. E-Mail-Verschlüsselungsprozess beim E-Mail-Empfang

In Abbildung 4 ist der E-Mail-Empfang unter Verwendung des MailCrypt-Plugins dargestellt.

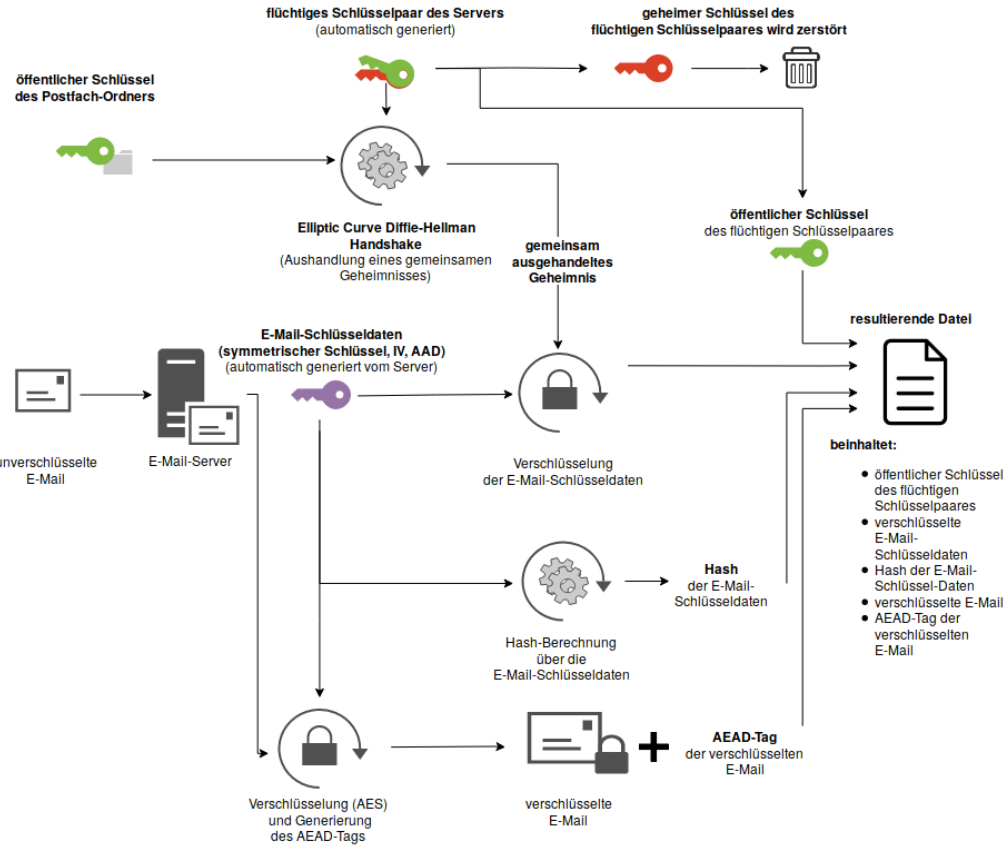

Abbildung 4: E-Mail-Verschlüsselungsprozess beim E-Mail-Empfang mit MailCrypt

Ist das Plugin für den Empfänger aktiviert, wird jede eingehende E-Mail verschlüsselt. Hierbei ist es unerheblich, ob die E-Mail über SMTP oder IMAP an den Server übermittelt wird. Die Verschlüsselung der E-Mail erfolgt mit Hilfe des Advanced Encryption Standard (AES). Um die Integrität der verschlüsselten Daten zu gewährleisten, wird AES im Galois/Counter-Mode (GCM) betrieben.

Das Plugin generiert zuerst zufällige Werte für den Verschlüsselungsprozess. Es erstellt einen Initialization Vector (IV), sowie Additional Authenticated Data (AAD) und einen symmetrischen Schlüssel. Sie werden im Folgenden nur noch als „E-Mail-Schlüssel-Daten“ bezeichnet. Mit diesen Werten wird die eingehende E-Mail verschlüsselt und der zugehörige Authenticated Encryption with Associated Data (AEAD)-Tag generiert [Tuomi et al., 2016].

Damit die E-Mail-Schlüsseldaten nur für den Nutzer zugänglich sind, müssen diese auch verschlüsselt abgespeichert werden. Dazu wird das Elliptic Curve Integrated Encryption Scheme (ECIES)-Verfahren angewendet. Hierbei wird zuerst ein flüchtiges Schlüsselpaar vom Server generiert. Mit dem privaten Schlüssel des flüchtigen Schlüsselpaares und dem vorliegenden öffentlichen Schlüssel des Postfach-Ordners wird ein Elliptic Curve Diffie-Hellman (ECDH)-Schlüsselaustausch durchgeführt. Hierbei wird zwischen dem privaten Schlüssel des flüchtigen Schlüsselpaares und dem öffentlichen Schlüssel des Postfach-Ordners ein gemeinsames Geheimnis ausgehandelt. Mit diesem Geheimnis werden die E-Mail-Schlüssel-Daten verschlüsselt. Damit der Server keinen Zugriff auf die verschlüsselten E-Mail-Schlüssel-Daten hat, wird der private Anteil des flüchtigen Schlüsselpaares zerstört. Zusätzlich wird ein Hash über die E-Mail-Schlüsseldaten gebildet, um später deren Integrität sicherstellen zu können.

Abschließend werden alle Daten in eine Datei geschrieben. An den Anfang der Datei werden der öffentliche Schlüssel des flüchtigen Schlüsselpaares, die verschlüsselten E-Mail-Schlüsseldaten und der Hash der E-Mail-Schlüsseldaten geschrieben. Danach folgt die verschlüsselte E-Mail und deren AEAD-Tag. [dovecot.org, 2017].

3.4.4. E-Mail-Entschlüsselungsprozess beim E-Mail-Abruf

Beim E-Mail-Abruf werden die gespeicherten E-Mails entschlüsselt. Der Ablauf des Entschlüsselungsprozess kann der Abbildung 5 entnommen werden.

Zum Abrufen der E-Mails loggt sich der Nutzer mit seinem Passwort auf dem E-Mail-Server ein. Das Passwort wird verwendet, um den PBKDF2-Hash zu bilden, mit dem der private Schlüssel des Nutzers verschlüsselt ist. Mit dem resultierenden Hash wird dieser entschlüsselt. Mit Hilfe des privaten Schlüssels kann der private Schlüssel des Postfach-Ordners (z.B. Inbox) entschlüsselt werden.

Nun wird die abgespeicherte Datei, die die verschlüsselte E-Mail und die zugehörigen Verschlüsselungsdaten enthält, abgerufen, und alle notwendigen Teile extrahiert. Hierzu gehören der öffentliche Schlüssel des zur Verschlüsselung eingesetzten flüchtigen Schlüsselpaares, die verschlüsselten E-Mail-Schlüssel-Daten, der Hash der E-Mail-Schlüsseldaten, die verschlüsselte E-Mail und der AEAD-Tag der verschlüsselten E-Mail.

Mit Hilfe des privaten Schlüssels des Postfach-Ordners und dem extrahierten öffentlichen Schlüssel wird per ECDH das Geheimnis berechnet, mit dem die „E-Mail-Schlüssel-Daten“ verschlüsselt wurden. Die E-Mail-Schlüssel-Daten werden entschlüsselt und mit Hilfe des extrahierten Hashes auf ihre Integrität überprüft. Mit den entschlüsselten E-Mail-Schlüssel-Daten wird die verschlüsselte E-Mail entschlüsselt und per AEAD-Tag

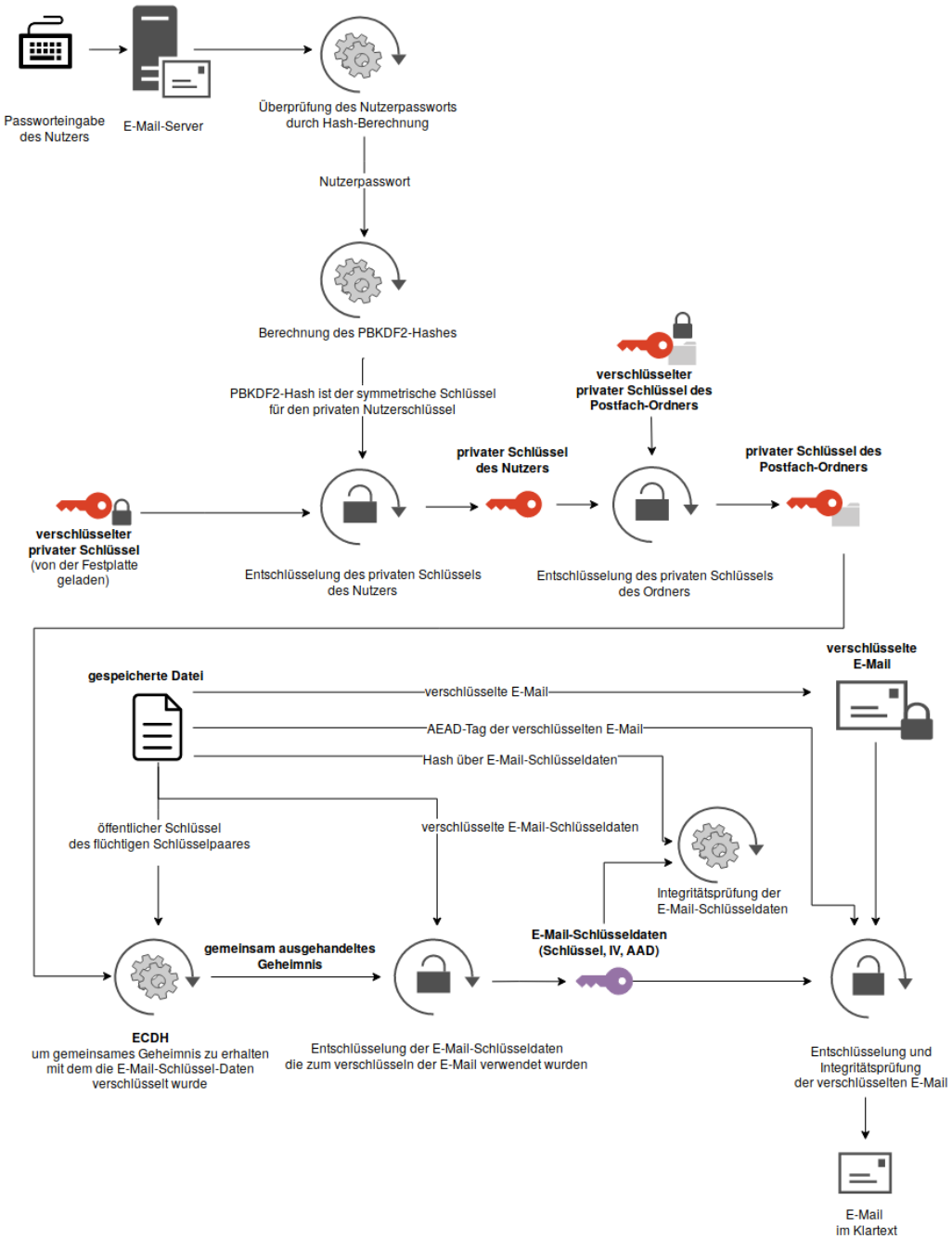

Abbildung 5: E-Mail-Entschlüsselungsprozess beim E-Mail-Abruf mit MailCrypt

auf ihre Integrität überprüft. Abschließend wird der Klartext der E-Mail dem Nutzer präsentiert.

3.5. Scrambler-Plugin

Das Dovecot-Plugin „Scrambler“[47] ist ein in C geschriebenes Programm, welches persönliche E-Mail-Postfachverschlüsselung für Dovecot nachrüstet. Es wurde im Jahr 2015 entwickelt und einem Sicherheits-Audit unterzogen [Brüll, 2015]. Seitdem ist es bei posteo.de im Produktivbetrieb [posteo.de, 2015]. Zur Verschlüsselung der eingehenden E-Mails setzt es auf die kryptographische Bilbliothek „OpenSSL“[48] und verwendet Schlüssel auf Basis von RSA.

3.5.1. Funktion

Ist das Plugin für einen bestehenden Nutzer aktiviert, werden alle zukünftig eingehenden E-Mails verschlüsselt. Hierbei ist unerheblich, ob die E-Mail über SMTP oder IMAP übermittelt wird. Bereits existierende E-Mails werden nicht automatisch verschlüsselt. Sie werden ohne Plugin-Interaktion an den Nutzer durch gereicht.

3.5.2. Konzept

Das Scrambler-Plugin setzt auf eine Kombination aus symmetrischer und asymmetrischer Kryptographie. Der Hash des Nutzer-Passworts wird als beiden Seiten bekanntes Geheimnis verwendet, um den privaten Schlüssel des Nutzers zu ver-/entschlüsseln.

Damit das Scrambler-Plugin für einen Nutzer verwendet werden kann, müssen die folgenden Schritte durchlaufen werden:

Zuerst wird ein bcrypt[49]-Hash des Nutzer-Passworts erstellt. Neben dem Passwort wird der Hash-Funktion ein zufällig generiertes Salt und die Anzahl der zu durchlaufenden Runden übergeben. Zusätzlich wird ein RSA-Schlüsselpaar mit einer Länge von 2048 Bit erstellt. Der geheime Schlüssel des Paares wird mit Hilfe des gehashten Passworts symmetrisch verschlüsselt. Hierfür wird der Algorithmus „aes-256-cbc“ eingesetzt.

[47] `https://github.com/posteo/scrambler-plugin`
[48] `https://www.openssl.org/`
[49] `https://www.usenix.org/legacy/events/usenix99/provos/provos.pdf`

Das zum Hashen verwendete Salt, die Anzahl der durchlaufenden Runden, der öffentliche Teil und der verschlüsselte geheime Teil des RSA-Schlüssels werden in einer zusätzlichen Datenbankstruktur abgelegt und dem jeweiligen Nutzer zugeordnet (siehe Tabelle 1).

Datenbankwert	**Beschreibung**
enabled	Legt fest, ob das Plugin für Nutzer aktiviert ist.
public_key	Öffentlicher Schlüssel des Nutzers.
private_key	Verschlüsselter, privater Schlüssel des Nutzers. Dieser wird mit dem gehashten Passwort des Nutzers verschlüsselt.
private_key_salt	Salt, welches beim Hashen des Passworts verwendet wird.
private_key_iterations	Anzahl der Runden, die bcrypt zum Hashen des Passworts durchläuft.

Tabelle 1: Notwendige Datenbankeinträge für das Scrambler-Plugin

Die Generierung der notwendigen Daten erfolgt nicht durch das Plugin. Hierzu müssen zusätzliche Skripte ausgeführt werden, die nicht Teil des Plugins sind.

3.5.3. E-Mail-Verschlüsselungsprozess beim E-Mail-Empfang

Der Verschlüsselungsprozess, der beim E-Mail-Empfang durchlaufen wird, ist in Abbildung 6 dargestellt.

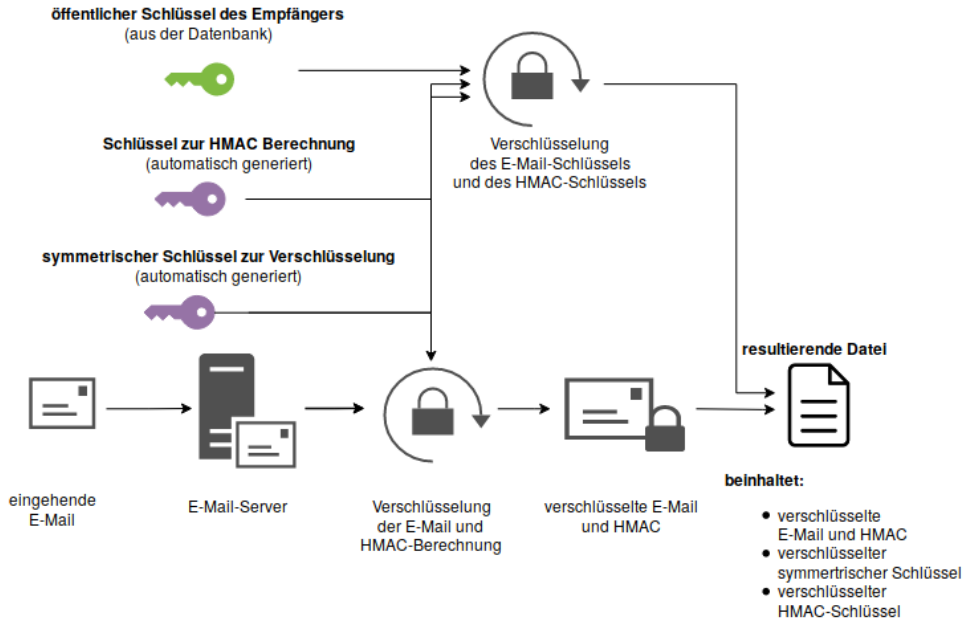

Abbildung 6: E-Mail-Verschlüsselungsprozess beim E-Mail-Empfang mit Scrambler

Beim Empfang einer E-Mail überprüft das Plugin zuerst, ob die Scrambler-Unterstützung für den Empfänger aktiviert ist. Ist dies der Fall, wird dessen öffentlicher Schlüssel aus der Datenbank abgerufen. Weiterhin generiert das Plugin zwei zufällige symmetrische Schlüssel. Mit einem dieser Schlüssel wird die eingehende E-Mail verschlüsselt. Der andere Schlüssel wird als Eingabe für eine HMAC-Funktion verwendet. Mit dem resultierenden Keyed-Hash Message Authentication Code (HMAC) ist es beim Abruf der E-Mail möglich, deren Integrität zu überprüfen.

Beide Schlüssel werden mit Hilfe des öffentlichen Nutzer-Schlüssels verschlüsselt. Nun wird eine Datei erstellt, in der die verschlüsselten Schlüssel und die verschlüsselte E-Mail abgespeichert werden. Dieses Verfahren wird nach der Aktivierung auf alle eingehenden E-Mails angewendet.

3.5.4. E-Mail-Entschlüsselungsprozess beim E-Mail-Abruf

Beim E-Mail-Abruf werden die gespeicherten E-Mails entschlüsselt. Der Entschlüsselungsprozess kann der Abbildung 7 entnommen werden.

Beim Abrufen einer E-Mail loggt sich der Nutzer wie gewohnt ein. Danach wird das Nutzer-Passwort an das Plugin weiter gereicht und mit Hilfe einer bcrypt-Hash-Funktion gehasht. Hierzu werden die Anzahl der zu durchlaufenden Runden, sowie das bei der

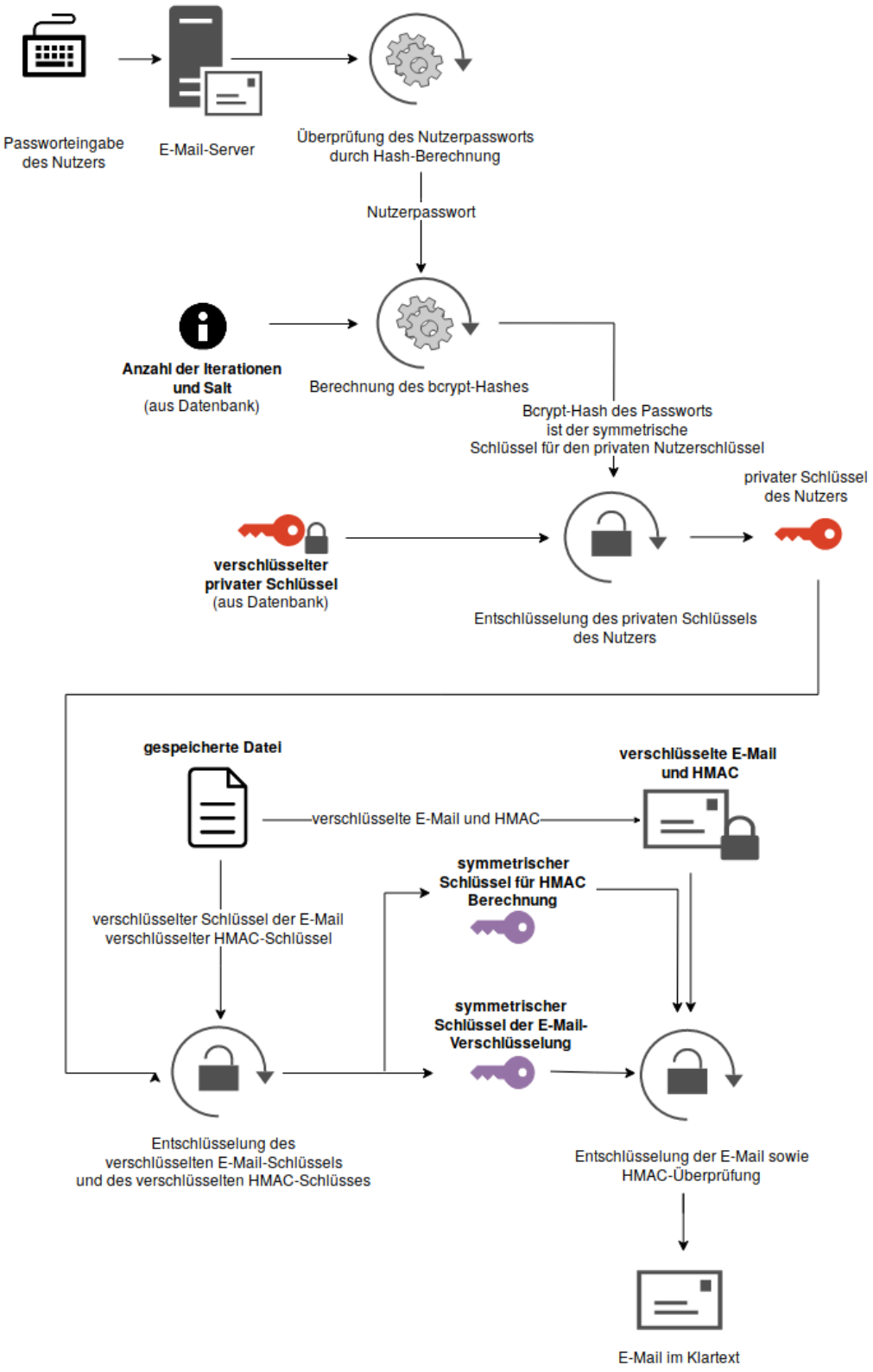

Abbildung 7: E-Mail-Entschlüsselungsprozess beim E-Mail-Abruf mit Scrambler

Generierung verwendete Salt aus der Datenbank abgerufen. Mit Hilfe des resultierenden bcrypt-Hashes wird der in der Datenbank liegende, verschlüsselte, private Teil des Nutzer-Schlüssels, entschlüsselt.

Der nun offengelegte geheime Schlüssel wird zur Entschlüsselung des symmetrischen Schlüssels verwendet, mit dem die E-Mail verschlüsselt wurde. Weiterhin wird der Schlüssel, der bei der HMAC-Tag-Berechnung verwendet wurde, entschlüsselt. Mit diesem Schlüssel ist es möglich, die HMAC-Tags der E-Mail zu berechnen und die Integrität der gespeicherten Daten zu verifizieren.

3.6. TREES

Das Dovecot-Plugin „TREES“[50] ist ein in C geschriebenes Programm, welches von Posteo's Scrambler Plugin inspiriert wurde (siehe 3.5). TREES ist hierbei das Akronym für „Technology for Resting Email Encrypted Storage“. Es wurde im Jahr 2016, als Reaktion auf einen FBI-Durchsuchungsbeschluss in Kombination mit einer Verschwiegenheitsanordnung, entwickelt. Auf Basis des Durchsuchungsbeschlusses wurde der Inhalt von zwei Postfächern beschlagnahmt. TREES ist seit Februar 2017 bei riseup.net im Produktivbetrieb [riseup.net, 2017]. Zur Verschlüsselung der eingehenden E-Mails setzt es auf die kryptographische Bibliothek „libsodium“[51] und verwendet Schlüssel auf Basis von elliptischen Kurven. Zusätzlich verwendet es zum Hashen von Passwörtern die Memory-hard Hash-Function (MHF) Argon2.

Im Folgenden wird die Funktionsweise des Plugins erläutert und näher auf die eingesetzte MHF eingegangen.

3.6.1. Funktion

Ist das Plugin für einen bestehenden Nutzer aktiviert, werden alle zukünftig eingehenden E-Mails verschlüsselt. Hierbei ist es unerheblich, ob die E-Mail über SMTP oder IMAP an den Server übermittelt wird. Bereits existierende E-Mails werden nicht automatisch verschlüsselt. Ein Zugriff auf bereits existierende E-Mails ist weiterhin möglich. Das Plugin überprüft beim Zugriff auf die abgespeicherten E-Mails, ob diese verschlüsselt sind. Dies geschieht anhand eines Headers, der bei der Verschlüsselung einer eingehenden E-Mail hinzugefügt wird. Ist die E-Mail unverschlüsselt, wird der Inhalt unangetastet an den Nutzer weiter gereicht.

[50] `https://0xacab.org/riseuplabs/trees`
[51] `https://libsodium.gitbook.io/doc/`

3.6.2. Konzept

Das TREES-Plugin setzt auch auf eine Kombination aus symmetrischer und asymmetrischer Verschlüsselung. Der Hash des Nutzer-Passworts wird als beiden Seiten bekanntes Geheimnis verwendet, um den privaten Schlüssel des Nutzers zu ver-/entschlüsseln.

Damit das TREES-Plugin für einen Nutzer verwendet werden kann, müssen die folgenden Schritte durchlaufen werden:

Zuerst wird ein Argon2-Hash[52] des Nutzer-Passworts erstellt. Der Hash-Funktion werden, neben dem Nutzer-Passwort, ein zufällig generiertes Salt, ein Wert zur Speicherbegrenzung und ein Wert zur Begrenzung der maximal durchzuführenden Berechnungen übergeben.

Zusätzlich wird ein Curve25519-Schlüsselpaar erstellt[53]. Der private Teil des Schlüsselpaares wird mit Hilfe des berechneten Passwort-Hashes verschlüsselt und als sogenannte „libsodium-SecretBox“ abgespeichert. Zusätzlich wird mit Hilfe einer Nonce ein AEAD-Tag generiert. Dieser Tag wird später verwendet, um die Integrität der verschlüsselten Daten zu verifizieren. Als Verschlüsselungsalgorithmus der SecretBox wird Salsa20[54] und als Authentifizierungsalgorithmus wird Poly1305[55] eingesetzt.

Die Generierung der notwendigen Nutzerdaten erfolgt nicht durch das Plugin. Diese Aufgabe wird an zusätzliche Skripte ausgelagert, die bei der Erstellung eines neuen Nutzers aufgerufen werden müssen. Sie sind im Repository des TREES-Plugins zu finden[56].

Der verwendete Hash-Algorithmus, die Konfigurationsparameter zur Hash-Berechnung, das eingesetzte Salt, der öffentliche Teil des Schlüsselpaares und die verschlüsselte SecretBox, werden in einer zusätzlichen Datenbankstruktur abgespeichert und dem jeweiligen Nutzer zugeordnet (siehe Tabelle 2).

[52] `https://eprint.iacr.org/2016/104.pdf`
[53] `https://cr.yp.to/ecdh.html`
[54] `https://cr.yp.to/snuffle/salsafamily-20071225.pdf`
[55] `https://cr.yp.to/mac.html`
[56] `https://0xacab.org/riseuplabs/trees/blob/master/bin/trees-create`

Datenbankwert	Beschreibung
enabled	Legt fest, ob das Plugin für den Nutzer aktiviert ist.
version	Legt fest welche Version des Plugins verwendet wird.
public_key	Öffentlicher Schlüssel des Nutzers.
pwhash_algo	Gibt an, welcher Hash-Algorithmus verwendet werden soll. Abhängig von libsodium-Version.
pwhash_opslimit	Legt die maximale Anzahl an durchgeführten Berechnungen der Argon2-Hashfunktion fest.
pwhash_memlimit	Legt den maximalen Speicherverbrauch der Argon2-Hashfunktion fest.
pwhash_salt	Gibt an, welches Salt beim Hashen des Nutzer-Passworts eingesetzt wurde.
sk_nonce	Gibt an, welche Nonce zur Verschlüsselung der libsodium-Secret Box verwendet wurde.
locked_secretbox	Verschlüsselte libsodium-SecretBox, die den geheimen Schlüssel des Nutzer-Schlüssels enthält.

Tabelle 2: Notwendige Datenbankeinträge für das TREES-Plugin

3.6.3. E-Mail-Verschlüsselungsprozess beim E-Mail-Empfang

In Abbildung 8 ist der E-Mail-Empfang unter Verwendung des TREES-Plugins visualisiert.

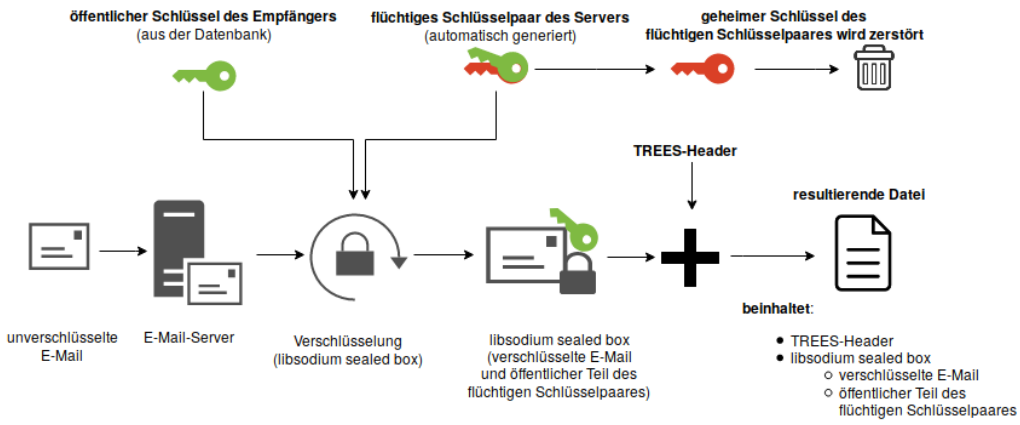

Abbildung 8: E-Mail-Verschlüsselungsprozess beim E-Mail-Empfang mit TREES

Beim Empfang einer E-Mail überprüft das Plugin zuerst, ob die TREES-Unterstützung für den Empfänger aktiviert ist. Ist dies der Fall, wird dessen öffentlicher Schlüssel aus der Datenbank abgerufen. Mit Hilfe von „libsodium SealedBoxes“ wird jede eingehende E-Mail verschlüsselt.

Die SealedBoxes sind hierbei von besonderer Bedeutung. Mit ihnen wird eine verschlüsselte Nachricht erzeugt, die der E-Mail-Server nicht mehr entschlüsseln kann [libsodium.org, o.J.]. Hierzu generiert die zugehörige Funktion ein flüchtiges Schlüsselpaar und verschlüsselt die eingehende E-Mail mit dem öffentlichen Schlüssel des Nutzers und dem privaten Schlüssel des flüchtigen Schlüsselpaares. Der private Schlüssel des flüchtigen Schlüsselpaares wird nach der Verschlüsselung zerstört. Der flüchtige, öffentliche Schlüssel wird an den Ciphertext der E-Mail angehängt und bildet mit ihr die „SealedBox“. Mit Hilfe des flüchtigen, öffentlichen Schlüssels kann der Empfänger später die Integrität der Nachricht überprüfen.

Abschließend wird eine Datei erstellt, an deren Anfang ein TREES-Header und danach die SealedBox geschrieben wird. Der TREES-Header gibt an, ob die Datei mit Hilfe von TREES verschlüsselt ist. Die Datei wird im Postfach des Nutzers gespeichert. Dieses Verfahren wird nach der Aktivierung auf alle eingehenden E-Mails angewendet.

3.6.4. E-Mail-Entschlüsselungsprozess beim E-Mail-Abruf

Beim E-Mail-Abruf werden die E-Mails des Nutzers entschlüsselt. Der Entschlüsselungsprozess ist in Abbildung 9 dargestellt.

Zum Abruf von E-Mails verbindet sich der Nutzer mit dem E-Mail-Server und übermittelt sein Passwort. Danach wird das Passwort an das TREES-Plugin weitergereicht. Auf dieses Passwort wendet das Plugin die MHF Argon2 an. Der resultierende Hash wird als symmetrischer Schlüssel verwendet, um die, in der Datenbank abgelegte, SecretBox[57] zu entschlüsseln. Vor der Entschlüsselung wird die Integrität der verschlüsselten Daten mit Hilfe des AEAD-Tags sichergestellt. Daraufhin wird die SecretBox entschlüsselt. In ihr befindet sich der private Schlüssel des Nutzers.

Nun wird die verschlüsselte Datei abgerufen und die SealedBox extrahiert. Die in ihr enthaltene E-Mail kann mit Hilfe des privaten Nutzer-Schlüssels, entschlüsselt werden. Weiterhin wird der öffentliche Schlüssel, des zur Verschlüsselung verwendeten flüchtigen Server-Schlüsselpaares, extrahiert. Mit diesem ist es möglich die Integrität der verschlüsselten Daten zu verifizieren. Abschließend wird die Klartext-E-Mail dem Nutzer präsentiert.

[57] `https://download.libsodium.org/doc/secret-key_cryptography/authenticated_encryption.html`

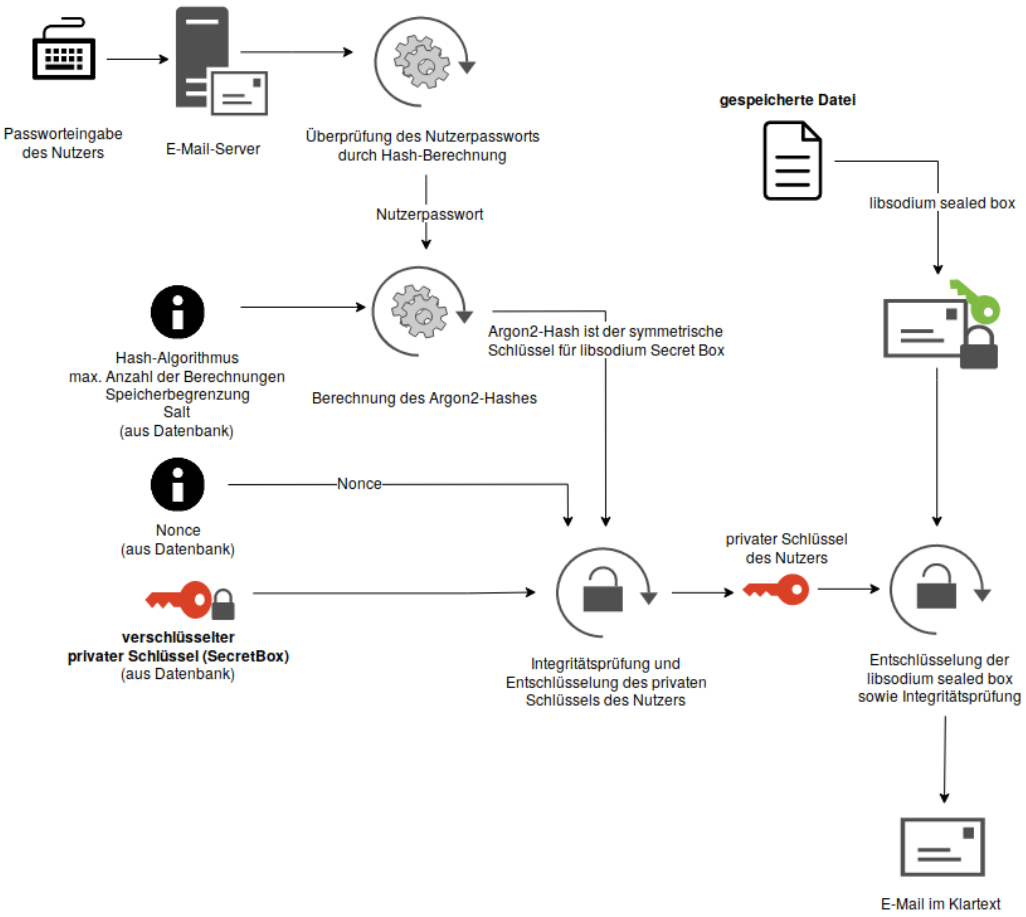

Abbildung 9: E-Mail-Entschlüsselungsprozess beim E-Mail-Abruf mit TREES

3.6.5. Einsatz einer Memory-Hard-Hash-Function (Argon2(i/d/id))

Eine Besonderheit des TREES-Plugins stellt der Einsatz einer MHF dar. Diese wird zum erneuten Hashen der Nutzerpasswörter verwendet. Die Idee von MHFs und der zugehörige Algrorithmus „scrypt“ wurden erstmalig im Jahr 2009 vorgestellt [Percival, 2009].

Memory-hard Hash-Function (MHF) steigern die Kosten von Offline-Attacken durch die Belegung von zusätzlichen Hardware-Ressourcen. Beim Einsatz von MHFs ist, neben der notwendigen Rechenleistung, immer eine bestimmte Menge an Arbeitsspeicher zur Durchführung der Berechnung notwendig. Der Hintergedanke ist, dass die Berechnungskosten der Passwörter von den Kosten des Arbeitsspeichers getrieben sind.

Während die Kosten für Rechenleistung in den letzten Jahren immer stärker gesunken sind und auch Spezialhardware zur Hash-Berechnung immer günstiger geworden ist [Blocki, 2017], sind die Kosten für Arbeitsspeicher jedoch relativ beständig [Lathan, 2018]. Dies trifft auch auf die preisliche Entwicklung über Plattformgrenzen hinweg

zu(PC, Server, ASICs). Ein Offline-Angreifer müsste dementsprechend über eine große Menge an Arbeitsspeicher verfügen, um effektive Offline-Attacken gegen derart gehashte Passwörter durchzuführen.

Das TREES-Plugin setzt auf eine solche MHF. Es verwendet Argon2 zum Hashen des Nutzer-Passworts. Argon2 ist der Gewinner der „Password Hashing Competition“[58]. Er existiert in mehreren Varianten. Je nach Konfiguration nutzt das Plugin den Algorithmus Argon2i oder Argon2id [password hashing.net, 2017]. Argon2i bietet die größte Sicherheit gegen Seitenkanalangriffe. Argon2d ist die Variante des Algrorithmus, die am widerstandsfähigsten gegen GPU-Cracking-Attacken ist [passwordhashing.net, 2015]. Argon2id stellt eine Kombination aus Argon2i und Argon2d dar.

3.7. Zusammenfassung

Zum Abschluss des Kapitels sollen die funktionalen Gemeinsamkeiten und Unterschiede der Techniken diskutiert werden. Hierbei werden vor allem die zentralen Merkmale diskutiert, da ein detaillierter Vergleich den Umfang der Arbeit überschreiten würde. Die Unterschiede im Implementierungsaufwand werden im weiteren Verlauf der Arbeit diskutiert (siehe 5.1).

Alle Techniken setzten auf eine Kombination aus symmetrischer und asymmetrischer Verschlüsselung. Bei den Plugins MailCrypt, Scrambler und TREES können die verschlüsselt abgespeicherten E-Mails durch die Eingabe des Nutzerpassworts entschlüsselt werden. Für den Nutzer entsteht hierbei kein zusätzlicher Aufwand. Beim Einsatz der GPG-Sieve-Filter-Technik benötigt der Nutzer zusätzliche Software. Weiterhin benötigt er ein GPG-Schlüsselpaar. Dieses ist häufig mit einem zusätzlichen Passwort gesichert. Der Einsatz dieser Technik bedeutet für den Nutzer einen größeren Aufwand, da er Kenntnisse über die GPG-Software benötigt.

Die vorgestellte GPG-Sieve-Filter-Technik unterscheidet sich von allen Plugins am Stärksten. Hierbei ist besonders hervorzuheben, dass diese Technik ausschließlich den Body und nicht den gesamten Content eines Mail Objects verschlüsselt. Weiterhin ist zu bemerken, dass die Technik lediglich die E-Mails verschlüsselt, die dem E-Mail-Server per SMTP übergeben werden. E-Mails, die über IMAP an den E-Mail-Server übergeben werden, werden unverschlüsselt abgespeichert. Das hat zur Folge, dass zum Beispiel der „Sent“-Ordner unverschlüsselt auf dem E-Mail-Server gespeichert ist. Positiv ist anzumerken, dass der Klartext der eingehenden E-Mail nach der initialen Verschlüsselung, nicht mehr durch den E-Mail-Server erreichbar ist. Auch nicht, wenn der Nutzer eingeloggt ist, da zur Entschlüsselung der private GPG-Schlüssel des Nutzers notwendig ist.

[58] `https://password-hashing.net/`

Dieser liegt in der Regel nicht auf dem E-Mail-Server und ist häufig durch ein zusätzliches Passwort geschützt, welches der E-Mail-Server nicht kennt.

Die Plugins Scrambler und TREES ähneln sich in ihrer Architektur am Stärksten. Dies ist nicht verwunderlich, da TREES von Scrambler inspiriert wurde. Beide Plugins brauchen zusätzliche Datenbankstrukturen und -werte, damit sie ordnungsgemäß funktionieren. Weiterhin verwenden beide Techniken eine zusätzliche Hash-Funktion, um aus dem Nutzerpasswort ein gemeinsames Geheimnis zu berechnen. Mit diesem wird der private Schlüssel des Nutzers verschlüsselt. Die Plugins unterscheiden sich in den verwendeten Verschlüsselungsbibliotheken, den eingesetzten Hash-Funktionen und den Verfahren, auf denen die Schlüsselpaare basieren. Scrambler verwendet als Verschlüsselungsbibliothek OpenSSL, setzt bcrypt zum Hashen ein und verwendet Schlüsselpaare auf Basis von RSA. TREES hingegen setzt auf die neuere Verschlüsselungsbibliothek libsodium, die MHF Argon2 und setzt auf Schlüsselpaare auf Basis von elliptischen Kurven. Weiterhin unterscheiden sich die beiden Plugins in der Umsetzung der E-Mail-Verschlüsselung. Scrambler generiert einen zufälligen symmetrischen Schlüssel zur E-Mail-Verschlüsselung. TREES hingegen erstellt ein flüchtiges Schlüsselpaar, um E-Mails zu verschlüsseln.

Das MailCrypt-Plugin unterscheidet sich deutlich von Scrambler und TREES. Es benötigt zum Beispiel keine zusätzlichen Datenbankstrukturen. Es legt alle notwendigen Daten als Dateien auf dem E-Mail-Server ab. Es verwendet die PBKDF2-Funktion, um aus dem Nutzerpasswort das Geheimnis zu berechnen, mit dem der private Anteil des Nutzerschlüsselpaares, verschlüsselt ist. Die Nutzerschlüssel basieren bei MailCrypt auf elliptischen Kurven. Auch bei der Erstellung des Geheimnis, mit dem die E-Mail schlussendlich verschlüsselt wird, unterscheidet sich MailCrypt von den anderen Plugins. Zu dessen Aushandlung setzt es auf das ECDH-Verfahren.

Aus dem Vergleich kann man entnehmen, dass sich die Plugins MailCrypt, Scrambler und TREES zum Teil in der Architektur, aber vor allem in der Wahl der Hash-Algorithmen, den eingesetzten Verschlüsselungsbibliotheken, den Verfahren auf denen die eingesetzten Schlüssel basieren und in den eingesetzten elliptischen Kurven, unterscheiden. Sowohl aus der Wahl der Hash-Algorithmen [Toponce, 2016], als auch aus der Schlüsselart [Sinha et al., 2013], resultieren diverse Performance-Unterschiede. Es wird vermutet, dass sich diese Unterschiede auf die Messergebnisse auswirken.

4. Untersuchung der Techniken

Dieses Kapitel beschäftigt sich mit der Untersuchung der verschiedenen Postfachverschlüsselungstechniken. Zuerst werden die Testumgebung und deren einzelne Bestandteile erläutert. Danach folgt die Beschreibung der eingesetzten Test-Software. Weiterhin werden die durchgeführten Versuche einzeln erläutert und ausgewertet.

4.1. Testumgebung

Die Testumgebung unterteilt sich oberflächlich in drei Komponentengruppen: E-Mail-Server, Test-Server und Monitoring-Server. Deren Funktionen und deren Mitglieder werden im Folgenden näher beschrieben.

- **E-Mail-Server:**
  Zu dieser Gruppe gehören die zu untersuchenden E-Mail-Server. Auf jedem E-Mail-Server ist eine Kombination aus Postfix, Dovecot und ein zu testendes Postfachverschlüsselungsplugin installiert. Zusätzlich werden auf ihnen Skripte und Software, zur Durchführung von lokalen Tests, installiert.
  - **gpgsieve.test:**
    Auf diesem Server ist der GPG-Sieve-Filter (siehe 3.3) installiert.
  - **mailcrypt.test:**
    Auf diesem Server ist das MailCrypt-Plugin (siehe 3.4) installiert.
  - **scrambler.test:**
    Auf diesem Server ist das Scrambler-Plugin (siehe 3.5) installiert.
  - **trees.test:**
    Auf diesem Server ist das TREES-Plugin (siehe 3.6) installiert.
  - **noencryption.test:**
    Auf diesem Server ist kein Postfachverschlüsselungsplugin installiert. In der Auswertung fungiert er als Referenzserver.
- **Test-Server:**
  Das Mitglied dieser Gruppe ist mit zusätzlicher Testsoftware ausgestattet.
  - **imaptester.test**
    Auf diesem Server ist die eingesetzte Testsoftware installiert (siehe 4.2). Von diesem Server werden entfernte Tests durchgeführt, bei denen Testanfragen an die E-Mail-Server gestellt werden.

- **Monitoring-Server:**
  Der Server in dieser Gruppe ist für das Abfragen und das Visualisieren der erhobenen Monitoring-Daten zuständig.

  - **monitor.test**
    Auf diesem Server ist eine Kombination aus Prometheus (siehe 2.6.6) und Grafana (siehe 2.6.7) installiert. Er ruft in regelmäßigen Abständen (1s) die Monitoring-Daten von den E-Mail-Servern ab.

Der beschriebene Versuchsaufbau ist in Abbildung 10 visualisiert:

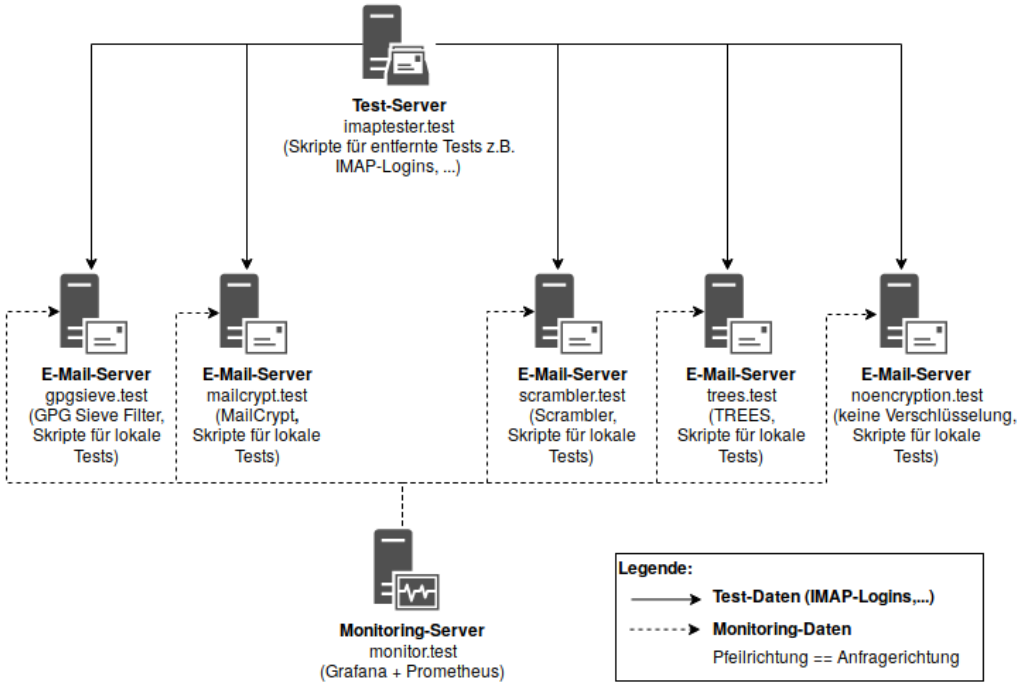

Abbildung 10: Versuchsaufbau

Die vorgestellte Testumgebung kann durch den Leser reproduziert werden. Dies ist durch den mitgelieferten Code und den Code im zugehörigen Repository[59] möglich. Die notwendigen Anforderungen und Installationsanweisungen befinden sich in der Datei *README.md*.

4.1.1. Virtualisierungshardware und -software

Die beschriebene Testumgebung wird mit Hilfe von Vagrant, VirtualBox und Ansible virtualisiert. Die Virtualisierung wird auf einem Laptop (Notebook Lenovo T470) mit

[59] `https://gitlab.com/bifi/mailencryption_testenvironment`

der folgenden Hardware-Spezifikation durchgeführt:

- CPU: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
- RAM: 16GB DDR4 @ 2133 MHz
- SSD: 500 GB Samsung SSD 970 EVO

Um die Testumgebung zu virtualisieren, wird der Laptop mit der folgenden Software betrieben:

- Betriebssystem: Ubuntu 18.04.1 LTS
- VirtualBox: Version 5.2.12r122591
- Vagrant: Version 2.0.2
- Ansible: Version 2.7.5

Mit Hilfe dieser Software werden virtuelle Maschinen (VMs) erstellt, die in ihrer Grundkonfiguration über eine CPU und 2048 MB RAM verfügen.

4.1.2. Basissoftware in den virtuellen Maschinen

Alle eingesetzten VMs werden mit dem Betriebssystem „Debian GNU/Linux 9.6 (stretch)“[60] betrieben. Auf den E-Mail-Servern liegt die Software in den folgenden Versionen vor:

- Postfix: 3.1.8
- Dovecot: 2.2.34 (874deae)
- MariaDB: 10.1.37

Dovecot wurde im Rahmen dieser Arbeit in Version 2.2.34 eingesetzt, da alle Postfachverschlüsselungsplugins mit dieser Dovecot-Version kompatibel sind. Mit den neueren Dovecot-Versionen des 2.3-Zweigs war dies, zumindest für das TREES-Plugin, nicht der Fall. Alle anderen Plugins wurden nicht auf eine Kompatibilität mit Dovecot 2.3.* untersucht.

4.1.3. Eingesetzte Postfachverschlüsselungsplugins

Neben der Basissoftware werden die jeweiligen Plugins auf den E-Mail-Servern installiert. Die Installation erfolgt in eindeutigen Versionen. Die Versionen sind durch den jeweiligen Git-Commit-Hashwert des Plugin-Repositories, oder durch die Versionsnummer des zugehörigen Debian-Pakets (Dovecot) festgelegt.

[60] `https://www.debian.org/`

- GPG-Sieve-Filter: e9432412f2eb6aca77fb4e7bb6fad41fcbfd8632
- Mailcrypt: Dovecot Version entsprechend 4.1.2
- Scrambler: cf23c9998fafb087937ddf1f9b711471fff1f8ef
- TREES: 14fed7d1ffa5d568224c1e4732fa563e83b22e3e

4.2. Eingesetzte Testsoftware

Zur Untersuchung der eingesetzten Postfachverschlüsselungstechniken werden diverse Bash[61]-Skripte entwickelt.

Diese setzen zum Versenden von E-Mails auf das Programm mail[62]. Zur Messung von Ausführungszeiten wird das Programm time[63] verwendet. Um den belegten Speicherplatz zu messen, wird das Programm du[64] eingesetzt. In einigen Versuchen wird die Prometheus-API des Monitoring-Servers angesprochen. Dies erfolgt mit Hilfe von cURL[65].

Alle Testskripte und zugehörige Testdateien sind im Repository der Testumgebung, unter den folgenden Pfaden, zu finden:

- ansible/roles/testscripts/files, templates
- ansible/roles/imaptester/templates

Neben den genannten Standard-Programmen wird zusätzliche Testsoftware verwendet. Diese soll im Folgenden kurz vorgestellt werden.

4.2.1. ImapTest

ImapTest ist ein Programm, mit dem man Belastbarkeitstest von IMAP-Servern durchführen kann. Im Rahmen dieser Arbeit wird es eingesetzt, um die Ausführung von IMAP-Kommandos zu messen.

ImapTest unterstützt von sich aus keine Tests über verschlüsselte Verbindungen. Da die eingesetzten E-Mail-Server lediglich verschlüsselte Verbindungen zulassen, muss diese über ein Zusatzprogramm aufgebaut werden. Hierfür wird stunnel[66] verwendet. stunnel baut eine verschlüsselte Verbindung zwischen dem zu testenden Server und dem Test-Server auf. Es legt diese Verbindung auf einen lokalen Port. ImapTest kann sich mit

[61] `https://www.gnu.org/software/bash/`
[62] `https://linux.die.net/man/1/mail`
[63] `https://linux.die.net/man/1/time`
[64] `https://linux.die.net/man/1/du`
[65] `https://curl.haxx.se/`
[66] `https://www.stunnel.org/`

dem lokalen Port verbinden und dadurch die Tests über eine verschlüsselte Verbindung durchführen.

4.3. Versuchsreihe

Im Rahmen dieser Arbeit wird untersucht, welche Auswirkungen der Einsatz von Postfachverschlüsselungstechniken auf den E-Mail-Server-Betrieb hat und wie sich diese darstellen. Hierzu werden diverse Versuche durchgeführt, welche in diesem Unterkapitel besprochen werden.

In dessen Verlauf werden die Versuche und deren Durchführung einzeln beschrieben und ausgewertet. Abschließend wird für jeden Versuch ein Fazit gezogen. Sollte ein Versuchsergebnis Auswirkungen auf einen nachfolgenden Test haben, wird dies im Fazit besprochen.

Die Postfachverschlüsselungstechniken werden den folgenden Versuchen unterzogen:

- Kompatibilität mit verschiedenen Postfachformaten beim E-Mail-Empfang
- Kompatibilität mit verschiedenen Postfachformaten beim E-Mail-Abruf
- Einfluss auf die E-Mail-Empfangsdauer
- Einfluss auf die E-Mail-Empfangsdauer in Abhängigkeit der E-Mail-Größe
- Einfluss auf den benötigten Speicherplatz
- Einfluss auf die Anzahl der IMAP-Logins
- Einfluss auf die E-Mail-Abrufdauer
- Einfluss auf den Nutzerkontenerstellungsprozess

4.3.1. Kompatibilität mit verschiedenen Postfachformaten beim E-Mail-Empfang

4.3.1.1. Versuchsbeschreibung

Dovecot unterstützt zum Abspeichern von E-Mails diverse Postfachformate [Sirainen and Andree, 2006]. In diesem Versuch soll untersucht werden, ob die eingesetzten Techniken mit den verschiedenen Postfachformaten kompatibel sind. Hierbei wird die Kompatibilität während des E-Mail-Empfangs betrachtet.

Im Rahmen des Tests werden nicht alle unterstützten Postfachformate untersucht. Auf die Untersuchung der Formate „Cyrus“, „imapc“ und „pop3c“ wird verzichtet. Dies liegt daran, dass „Cyrus“ nur zum Durchführen von Tests und Benchmarks empfohlen wird und „imapc“ und „pop3c“ einen weiteren Server als E-Mail-Speicher benötigen würden [Sirainen and Andree, 2006].

4.3.1.2. Versuchsdurchführung

Zu Beginn des Versuches werden alle E-Mail-Server zur Nutzung eines Postfachformats konfiguriert. Dazu wird in '`ansible/group_vars/all/main.yml`' die Variable für das Postfachformat '`dovecot_mailbox_type`' angepasst und die E-Mail-Server mit Hilfe von Ansible provisioniert.

Daraufhin wird auf dem E-Mail-Server ein Testskript aufgerufen, welches 1000 Test-E-Mails zustellt. Die Zustellung der E-Mails erfolgt lokal, um eventuelle Verfälschungen der Testergebnisse zu vermeiden (siehe Listing 6).

Listing 6: Exemplarischer Aufruf des Postfach-Kompatibilitätstest

```
user@hypervisor:07_Testumgebung $ ansible all −i ansible/hosts −−private−key=/home/user/Desktop/
    Masterarbeit/07_Testumgebung/keys/vagrant.id_ecdsa −u vagrant −l mailservers −m shell −a '/bin
    /bash −c ‘‘time ./testmail.sh‘‘'
```

Ist der Versand der E-Mails abgeschlossen, wird überprüft, ob die empfangenen E-Mails verschlüsselt abgespeichert wurden (siehe Listing 14 im Anhang). Ist dies der Fall, wird der Test als bestanden gewertet.

4.3.1.3. Versuchsauswertung

Technik / Postfach-format	GPG-Sieve-Filter	MailCrypt	Scrambler	TREES
mbox	✓	✗	✗	✗
Maildir	✓	✓	✓	✓
sdbox	✓	✓	✓	✓
mdbox	✓	✓	✓	✓

Tabelle 3: Kompatibilitätsübersicht zu unterschiedlichen Postfachformaten beim E-Mail-Empfang

Aus den Testergebnissen in Tabelle 3 ist ersichtlich, dass die Plugins MailCrypt, Scrambler und TREES nicht mit dem Postfachformat „mbox“ kompatibel sind.

4.3.1.4. Fazit

In diesem Test wurde untersucht, ob die Techniken während, des E-Mail-Empfangs, zu unterschiedlichen Postfachformaten kompatibel sind. Es wurde festgestellt, dass die

Plugins MailCrypt, Scrambler und TREES nicht mit dem Postfachformat „mbox“ kompatibel sind. Für alle weiteren Tests ist dieses Postfachformat deswegen ungeeignet und wird nicht weiter berücksichtigt.

Im nächsten Test wird untersucht, mit welchen der verbliebenen Postfachformate die Techniken kompatibel sind, wenn E-Mails abgerufen werden.

4.3.2. Kompatibilität mit verschiedenen Postfachformaten beim E-Mail-Abruf

4.3.2.1. Versuchsbeschreibung

In diesem Test soll untersucht werden, ob unter Einsatz der Techniken ein Abruf der verschlüsselt abgespeicherten E-Mails für die verschiedenen Postfachformate möglich ist.

Dieser Test wird für die Postfachformate „Maildir“, „sdbox“ und „mdbox“ durchgeführt. Das Abrufen der E-Mails erfolgt sowohl über POP3, als auch über IMAP.

4.3.2.2. Versuchsdurchführung

Zuerst werden alle E-Mail-Server für die Nutzung des jeweiligen Postfachformats konfiguriert. Daraufhin wird auf dem jeweiligen E-Mail-Server ein Testskript aufgerufen, welches 1000 Test-E-Mails zustellt (siehe Listing 6). Während des Versandvorgangs wird das Postfach mit Hilfe eines E-Mail-Clients (Thunderbird) über POP3 abgerufen. Der E-Mail-Client wird hierbei auf dem Hypervisor ausgeführt.

Abschließend wird überprüft, ob das Abrufen der E-Mails fehlerfrei funktioniert hat. Zusätzlich wird der Status des Dovecot-Daemons überprüft (siehe Listing 7). Wenn alle E-Mails erfolgreich abgerufen wurden und der Dovecot-Daemon ansprechbar ist, wird der Test als bestanden gewertet.

Listing 7: Kommando zur Überprüfung des Dovecot Status

```
user@hypervisor:07_Testumgebung $ ansible all -i ansible/hosts --private-key=/home/user/Desktop/
    Masterarbeit/07_Testumgebung/keys/vagrant.id_ecdsa -u vagrant -l mailservers -m shell -a '/bin
    /bash -c ''sudo systemctl status dovecot''
```

Nun wird das E-Mail-Postfach auf dem E-Mail-Server geleert und der Test erneut durchgeführt. Diesmal werden die E-Mails über IMAP abgerufen.

Abschließend wird die E-Mail-Server-VM zerstört, neu erstellt und mit dem nächsten Postfachformat provisioniert. Danach wird der Versuch erneut durchgeführt.

4.3.2.3. Versuchsauswertung

Technik / Postfach-format	GPG-Sieve-Filter		MailCrypt		Scrambler		TREES	
Zugriff via	POP3	IMAP	POP3	IMAP	POP3	IMAP	POP3	IMAP
Maildir	✓	✓	✓	✓	✗	✓	✗	✓
sdbox	✓	✓	✓	✓	✓	✓	✓	✓
mdbox	✓	✓	✓	✓	✓	✓	✓	✓

Tabelle 4: Kompatibilitätsübersicht zu unterschiedlichen Postfachformaten beim E-Mail-Abruf

Die Versuchsergebnisse in Tabelle 4 zeigen, dass die Techniken Scrambler und TREES nicht mit dem Postfachformats „Maildir“ kompatibel sind. Das Abrufen von E-Mails über POP3 führte, bei gleichzeitigem E-Mail-Empfang, zu einem reproduzierbaren Absturz des POP3-Dienstes des E-Mail-Servers. Dadurch ist es nicht mehr möglich weitere E-Mails für diesen Nutzer per POP3 abzurufen. Die resultierenden Fehlermeldungen für Scrambler und TREES, können dem Anhang entnommen werden (siehe Listing 15, 16). Ein Löschen der betroffenen E-Mail oder des kaputten Dovecot-Caches lösten das Problem nicht. Der E-Mail-Server ist für diesen Nutzer erst wieder per POP3 ansprechbar, wenn sein Postfach neu erstellt oder ein alter Stand des Postfachs wiederhergestellt werden würden. Die Löschung und Wiederherstellung des Postfachs würden ein manuelles Eingreifen durch einen Admin erfordern und für den Nutzer einen Datenverlust hervorrufen.

Erfolgte der Abruf der E-Mails über POP3 nicht während des E-Mail-Empfangs, trat dieser Fehler nicht auf.

4.3.2.4. Fazit

In diesem Versuch konnte gezeigt werden, dass die Techniken Scrambler und TREES nicht mit dem Postfachformat „Maildir“ kompatibel sind. Es konnte ein reproduzierbarer Fehler festgestellt werden, falls während des E-Mail-Empfangs E-Mails über POP3 abgerufen werden. Einmal ausgelöst verhinderte dieser Fehler, das weitere E-Mails des Nutzers abgerufen werden können. Dementsprechend kann das Postfachformat „Maildir“ nicht für die weiteren Tests verwendet werden. Lediglich „sdbox“ und „mdbox“ verbleiben als Kandidaten für die weiteren Versuche.

Für alle weiteren Versuche wird das Postfachformat „mdbox“ eingesetzt, da es vor allem in größeren E-Mail-Server-Installationen diverse Performance-Vorteile bietet [Heinlein, 2012].

4.3.3. Einfluss auf die E-Mail-Empfangsdauer

4.3.3.1. Versuchsbeschreibung

In diesem Versuch soll untersucht werden, welche Auswirkung der Einsatz von Postfachverschlüsselungstechniken auf die E-Mail-Empfangsdauer hat. Dazu wird gemessen, wie viel Zeit für die lokale Zustellung von 1000 E-Mails benötigt wird.

Die versendeten E-Mails haben hierbei einen identischen Inhalt. Dessen Größe beträgt 50 Kilobyte (KB). Sofern nicht anders angegeben, gilt dies auch für alle weiteren Tests. Diese Größe wird festgelegt, da die Literatur-Angaben für die durchschnittliche E-Mail-Größe sehr unterschiedlich sind und zwischen 25 KB und 75 KB variieren [esmtp.email, 2017] [Lyman et al., 2003] [Gray, 2018].

4.3.3.2. Versuchsdurchführung

Zur Durchführung des Versuchs wird ein Skript entwickelt, welches 1000 E-Mails à 50 KB lokal zustellt. Mit Hilfe des Programms „time“ wird die Ausführungszeit des Skripts gemessen (siehe Listing 8). Die Ausführungszeit entspricht der Dauer, die der E-Mail-Server zum Empfangen aller E-Mails benötigt. Der Test wird für jeden E-Mail-Server zehn mal wiederholt. Danach wird der nächste E-Mail-Server untersucht.

Um eine Beeinflussung der Versuchsergebnisse zu verhindern, werden zu Versuchsbeginn alle nicht benötigten VMs ausgeschaltet und unnötige Prozesse auf dem Hypervisor beendet.

Listing 8: Exemplarischer Aufruf des E-Mail-Empfangsdauertests

```
user@hypervisor:07_Testumgebung $ ansible all -i ansible/hosts --private-key=/home/user/Desktop/
    Masterarbeit/07_Testumgebung/keys/vagrant.id_ecdsa -u vagrant -l noencryption.test -m shell -a
    '/bin/bash -c ``time ./timingtest.sh``'
```

4.3.3.3. Versuchsauswertung

Zur Auswertung wird der Mittelwert und die Standardabweichung aus den gemessenen Ausführungszeiten gebildet. Danach werden die Ergebnisse visualisiert (siehe Abbildung 11).

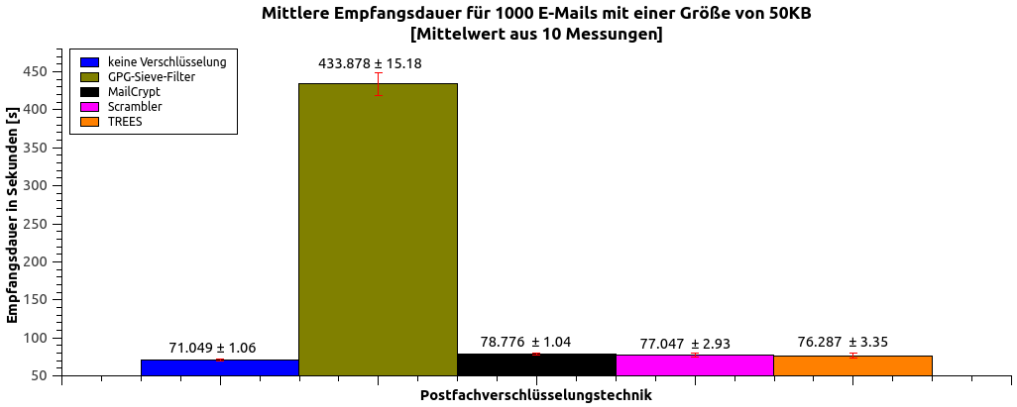

Abbildung 11: Mittlere Empfangsdauer für 1000 E-Mails mit einer Größe von 50 KB [Mittelwert aus 10 Messungen]

Den Versuchsergebnissen in Abbildung 11 kann man entnehmen, dass sich die durchschnittliche Empfangsdauer für den Empfang von 1000 E-Mails durch den Einsatz von Verschlüsselungstechniken verlängert. Weiterhin zeigen die Messwerte (siehe Abbildung 11), dass die Messergebnisse gut reproduzierbar sind.

Technik / Abweichung	absolute Abweichung zur Dauer des unverschlüsselten E-Mail-Empfangs [s]	relative Abweichung zur Dauer des unverschlüsselten E-Mail-Empfangs [%]
GPG-Sieve-Filter	362.83	510.67
MailCrypt	7.73	10.88
Scrambler	6.00	8.44
TREES	5.24	7.37

Tabelle 5: Absolute und relative Abweichungen der Mittelwerte der Empfangsdauer für 1000 E-Mails in Bezug zum unverschlüsselten E-Mail-Empfang

Den Werten aus Tabelle 5 kann man entnehmen, dass die Empfangsdauer, je nach eingesetzter Verschlüsselungstechnik, unterschiedlich stark ausfällt. Beim Einsatz der Plugins MailCrypt, Scrambler und TREES wird die Empfangsdauer um ca. 7%-11% verlängert. Da die E-Mail in der Regel nicht als Echtzeitkommunikationsmedium verwendet wird, dürfte diese Verzögerung in der Praxis vernachlässigbar sein.

Beim verschlüsselten Abspeichern via GPG-Sieve-Filter tritt hingegen eine gravierende Verzögerung auf.

Da der Empfang von E-Mails per GPG-Sieve-Filter eine derart gravierende Verzögerung aufweist, ist das Plugin für die nachfolgenden Tests ungeeignet. Der Einsatz des Plugins für viele Nutzer einer E-Mail-Installation erscheint nicht praktikabel. Aufgrund des-

sen werden die folgenden Tests lediglich mit den Verschlüsselungsplugins MailCrypt, Scrambler und TREES durchgeführt.

4.3.3.4. Fazit

In diesem Versuch konnte festgestellt werden, dass der Einsatz der Verschlüsselungstechniken den E-Mail-Empfang verzögert. Diese Verzögerung fällt für die Technik des GPG-Sieve-Filters so stark aus, dass diese in den folgenden Versuchen nicht weiter berücksichtigt wird. Für alle anderen Techniken beträgt die Verzögerung ca. 10%.

Im nächsten Versuch soll untersucht werden, ob die Verzögerung beim E-Mail-Empfang von der E-Mail-Größe abhängig ist.

4.3.4. Einfluss auf die E-Mail-Empfangsdauer in Abhängigkeit der E-Mail-Größe

4.3.4.1. Versuchsbeschreibung

Aus Kapitel „Einfluss auf die E-Mail-Empfangsdauer“ ist ersichtlich, dass der Einsatz von Verschlüsselungstechniken den E-Mail-Empfang verzögert. Weiterhin zeigte der Test, dass, je nach eingesetzter Technik, die Empfangsdauer unterschiedlich stark verzögert wird. Die vorherige Untersuchung beschränkte sich auf E-Mails mit einer Größe von 50 KB.

In diesem Test soll untersucht werden, ob die beobachtete Verzögerung von der E-Mail-Größe abhängig ist.

Da die Messwerte für die Empfangsdauer gut reproduzierbar sind (siehe Kapitel 4.3.3), wird die Anzahl der zu versendenden E-Mails auf 100 reduziert und der Versuch lediglich fünfmal wiederholt (siehe Listing 9).

4.3.4.2. Versuchsdurchführung

Zu Beginn des Versuchs werden alle nicht benötigten VMs ausgeschaltet. Auf dem verbliebenen E-Mail-Server wird ein Test-Skript aufgerufen, welches 100 E-Mails mit einer definierten Größe lokal zustellt. Hierbei wird die Ausführungszeit des Skripts gemessen, welche der Empfangsdauer der gesendeten E-Mails entspricht.

Die versendeten E-Mails haben hierbei einen identischen Inhalt. Die Größe des E-Mail-Inhalts beträgt je nach Testschritt 10 KB, 50 KB, 100 KB, 1 MB, 5 MB und 10 MB.

Listing 9: Exemplarischer Aufruf des E-Mail-Empfangsdauertests (in Abhängigkeit der E-Mail-Größe)

```
user@hypervisor:07_Testumgebung $ ./run_test_x_times.sh 5 ''time ./encryption_by_mailsize.sh''
    noencryption.test
```

4.3.4.3. Versuchsauswertung

Zur Auswertungen werden die Mittelwerte und Standardabweichungen für die erhobenen Messwerte gebildet und in Abbildung 12 visualisiert.

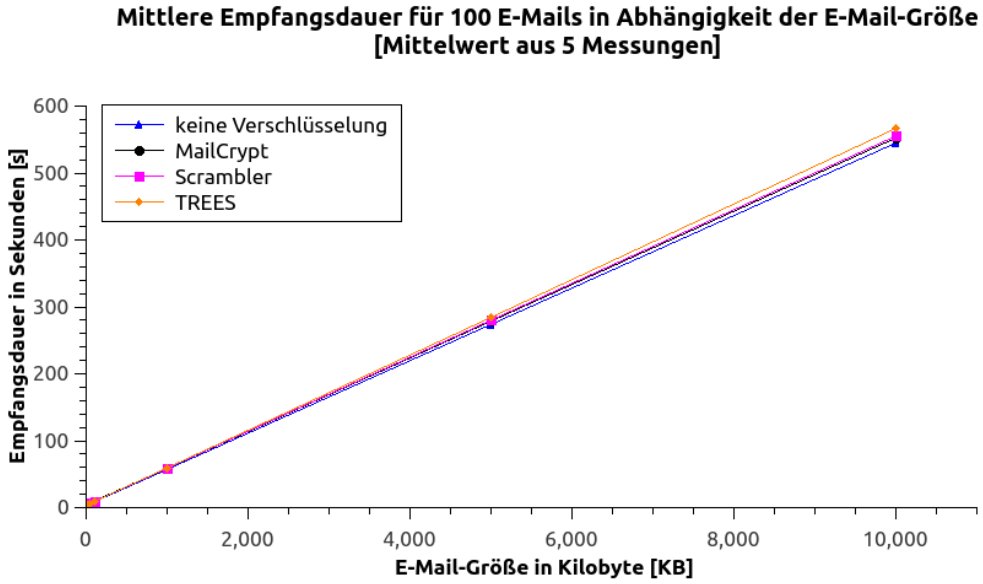

Abbildung 12: Mittlere Empfangsdauer für 100 E-Mails in Abhängigkeit der E-Mail-Größe [Mittelwert aus 10 Messungen]

Den Messergebnissen kann man entnehmen, dass es einen linearen Zusammenhang zwischen der Größe des E-Mail-Inhalts und der gemessenen Empfangsdauer gibt (siehe Abbildung 12). Zusätzlich kann man erkennen, dass der Einsatz einer Postfachverschlüsselungstechnik, den E-Mail-Empfang verzögert. Dies entspricht den Beobachtungen aus dem Versuch „Einfluss auf die E-Mail-Empfangsdauer“.

Weiterhin ist ersichtlich, dass die Verzögerung, je nach eingesetzter Technik, unterschiedlich stark ausfällt. Aus den Messwerten in Tabelle 5 ist zusätzlich ersichtlich, dass die Messung gut reproduzierbar ist.

Technik \ Empfangsdauer nach E-Mail-Größe	10 KB [s]	50 KB [s]	100 KB [s]	1 MB [s]	5 MB [s]	10 MB [s]
keine Verschlüsselung	4.02 ± 0.09	6.09 ± 0.08	8.70 ± 0.03	57.07 ± 0.37	273.80 ± 1.40	544.92 ± 2.09
MailCrypt	5.17 ± 0.15	7.32 ± 0.14	9.86 ± 0.06	58.81 ± 0.25	278.61 ± 0.41	552.44 ± 2.09
Scrambler	4.43 ± 0.16	6.54 ± 0.08	9.23 ± 0.15	58.95 ± 0.27	280.15 ± 1.84	555.28 ± 0.95
TREES	4.17 ± 0.07	6.36 ± 0.04	9.08 ± 0.10	59.33 ± 0.27	284.24 ± 0.60	566.80 ± 1.73

Tabelle 6: Mittelwert und Standardabweichung für die Empfangsdauer von 100 E-Mails in Abhängigkeit der E-Mail-Größe [Mittelwert aus 5 Messungen]

Im nächsten Schritt der Auswertung werden die absoluten und relativen Abweichungen zwischen den Tests mit aktivierter Verschlüsselungstechnik und ihrem unverschlüsselten Pendant gebildet. Die Ergebnisse werden in Abbildung 13 und Abbildung 14 visualisiert.

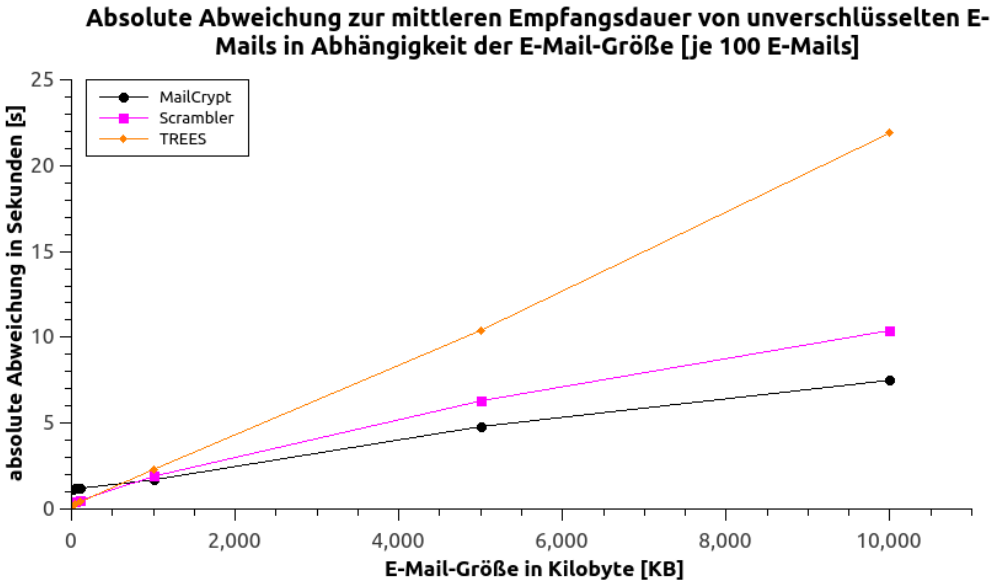

Abbildung 13: Absolute Abweichung zur mittleren Empfangsdauer von unverschlüsselten E-Mails in Abhängigkeit der E-Mail-Größe [je 100 E-Mails]

Technik \ abs. Abweichung nach E-Mail-Größe	10KB [s]	50 KB [s]	100 KB [s]	1MB [s]	5 MB [s]	10 MB [s]
MailCrypt	1.15	1.23	1.16	1.74	4.81	7.52
Scrambler	0.41	0.45	0.53	1.88	6.34	10.35
TREES	0.15	0.27	0.37	2.26	10.44	21.88

Tabelle 7: Absolute Abweichung zur mittleren Empfangsdauer von unverschlüsselten E-Mails in Abhängigkeit der E-Mail-Größe [je 100 E-Mails]

Aus der Graphik 13 und Tabelle 7 ist erkennbar, dass mit ansteigender Größe des E-Mail-Inhalts, auch die absolute Abweichung der Mittelwerte gegenüber der Dauer beim

unverschlüsselten E-Mail-Empfang, linear ansteigt. Das bedeutet, je größer die empfangende E-Mail ist, umso länger wird ihr Empfang verzögert. Dies ist ein guter Hinweis auf einen stattfindenden Verschlüsselungsprozess. Dieser dauert mit ansteigender E-Mail-Größe länger, da mehr Daten verschlüsselt werden müssen. Deswegen müssen mehr Operationen durchgeführt werden, was in einer längeren Empfangsdauer resultiert.

Zusätzlich kann man erkennen, dass das Verzögerungsverhalten von der eingesetzten Technik und der E-Mail-Größe abhängig ist. Im unteren Messbereich zwischen 10 KB und 100 KB verzögert die MailCrypt-Technik den Empfang von 100 E-Mails am stärksten. Diese wird gefolgt von der Scrambler-Technik. Das TREES-Plugin verzögert in diesem Bereich am geringsten. Dieses Verhalten dreht sich im Bereich zwischen 1 MB und 10 MB um. Hier verzögert das TREES-Plugin am stärksten, gefolgt vom Scrambler- und MailCrypt-Plugin.

Das Verhalten des MailCrypt-Plugins lässt sich damit erklären, dass MailCrypt immer eine gewisse Zeit benötigt, um das flüchtige Schlüsselpaar für den ECDH-Schlüsselaustausch zu generieren. Weiterhin muss das Plugin zufällige Daten für das E-Mail-Verschlüsselungspasswort, den Initialization Vector und für die Additional Authenticated Data abrufen. Mit wachsender E-Mail-Größe fällt die Dauer dieser Vorgänge immer weniger ins Gewicht. Hier dominiert die Zeit, die zum Verschlüsseln und Abspeichern der E-Mail benötigt wird.

Es wird vermutet, dass es beim Scrambler-Plugin ähnlich ist. Auch hier werden zufällig generierte Daten zum Verschlüsseln der E-Mails verwendet. Beim TREES-Plugin scheint sich der Verschlüsselungsprozess immer gleich stark auf die Empfangsdauer auszuwirken.

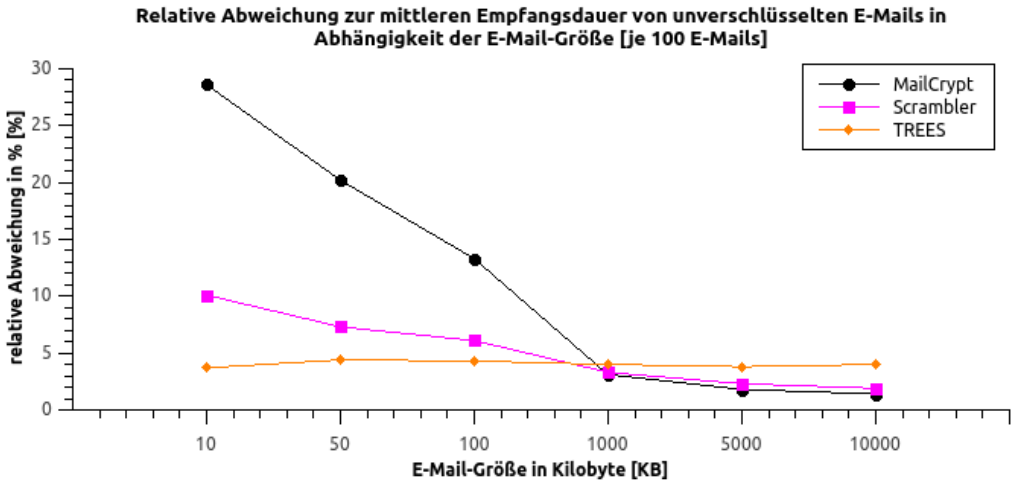

Abbildung 14: Relative Abweichung zur mittleren Empfangsdauer von unverschlüsselten E-Mails in Abhängigkeit der E-Mail-Größe [je 100 E-Mails]

Technik \ rel. Abweichung nach E-Mail-Größe	10 KB [%]	50 KB [%]	100 KB [%]	1 MB [%]	5 MB [%]	10 MB [%]
MailCrypt	28.58	20.20	13.31	3.05	1.76	1.38
Scrambler	10.13	7.32	6.07	3.29	2.32	1.90
TREES	3.67	4.40	4.30	3.96	3.81	4.01

Tabelle 8: Relative Abweichung zur mittleren Empfangsdauer von unverschlüsselten E-Mails in Abhängigkeit der E-Mail-Größe [je 100 E-Mails]

Bei der Betrachtung der relativen Abweichung (siehe Abbildung 8) fällt auf, dass diese je nach Technik und E-Mail-Größe unterschiedlich stark ausfällt.

Während die relative Abweichung der Empfangsdauer mit ansteigender Größe des E-Mail-Inhalts für die Plugins MailCrypt und Scrambler sinkt, bleibt sie für das TREES-Plugin nahezu konstant. Im Bereich von 10 KB - 100 KB verursacht der Einsatz des MailCrypt Plugins die größte Abweichung. Im Bereich zwischen 1 MB - 10 MB verursacht es hingegen die kleinste Abweichung.

Beim Empfang einer durchschnittlichen E-Mail, mit einem Inhalt von 50 KB dauert der Empfang zwischen ca. 4% (TREES) und 20% (MailCrypt) länger, als der Empfang einer E-Mail, die nicht serverseitig verschlüsselt wird.

Rechnet man den relativen Wert der Empfangsdauer in den absoluten Wert pro E-Mail um, beträgt diese für eine 50 KB E-Mail 0,01s (MailCrypt). Auch hier gilt, da E-Mail-Kommunikation in der Regel nicht als Echtzeitkommunikationsmedium eingesetzt wird, dürfte diese Verzögerung vermutlich irrelevant für den Nutzer sein.

4.3.4.4. Fazit

In diesem Versuch konnte gezeigt werden, dass der Einsatz der Verschlüsselungstechniken den Empfang von E-Mails verzögert. Hierbei wurde festgestellt, dass mit ansteigender E-Mail-Größe auch die hervorgerufene Verzögerung ansteigt. Weiterhin konnte gezeigt werden, dass die Verzögerung von der eingesetzten Verschlüsselungstechnik und der E-Mail-Größe abhängig ist. Beim Empfang einer 50 KB großen E-Mail tritt hierbei eine Verzögerung von bis zu 20% auf (MailCrypt). Betrachtet man die absoluten Zahlen, fällt die Verzögerung pro E-Mail jedoch so gering aus, dass sie für den Nutzer irrelevant sein sollte.

4.3.5. Einfluss auf den benötigten Speicherplatz

4.3.5.1. Versuchsbeschreibung

Im Kapitel „Stand der Technik“ wurde dargestellt, dass die Postfachverschlüsselungstechniken zusätzliche Daten bei der E-Mail-Verschlüsselung abspeichern.

Im Rahmen dieses Versuchs soll untersucht werden, ob und wie viel zusätzlicher Speicherplatz dadurch belegt wird.

4.3.5.2. Versuchsdurchführung

Auf dem jeweiligen E-Mail-Server wird ein Skript aufgerufen, welches 100 E-Mails mit verschiedenen, definierten Größen, lokal zustellt. Vor dem Versenden und nach der Zustellung wird der Speicherplatz ermittelt, den das Postfach belegt. Aus der Differenz der beiden Werte wird der benötige Speicherplatz berechnet.

Die versendeten E-Mails haben hierbei einen identischen Inhalt. Die Größe der E-Mail beträgt je nach Testschritt 10 KB, 50 KB, 100 KB, 1 MB. Der Versuch wird jeweils zehnmal wiederholt. Der Aufruf des Tests ist in Listing 10 zu finden.

Listing 10: Exemplarischer Aufruf des Versuchs zur Ermittlung der zusätzlichen Speicherplatzbelegung nach der Verschlüsselung

```
user@hypervisor:07_Testumgebung $ for host in {noencryption.test,mailcrypt.test,scrambler.test,trees.test
    }; do echo ‘‘$host‘‘ && vagrant ssh $host −− −t ’./diskspace_usage_after_encryption.sh’ && echo
    ‘‘‘‘; done
```

4.3.5.3. Versuchsauswertung

In Abbildung 15 sind die Mittelwerte der Messergebnisse visualisiert. Hier kann man die Speicherplatzbelegung pro 100 E-Mails in Abhängigkeit der E-Mail-Größe betrachten.

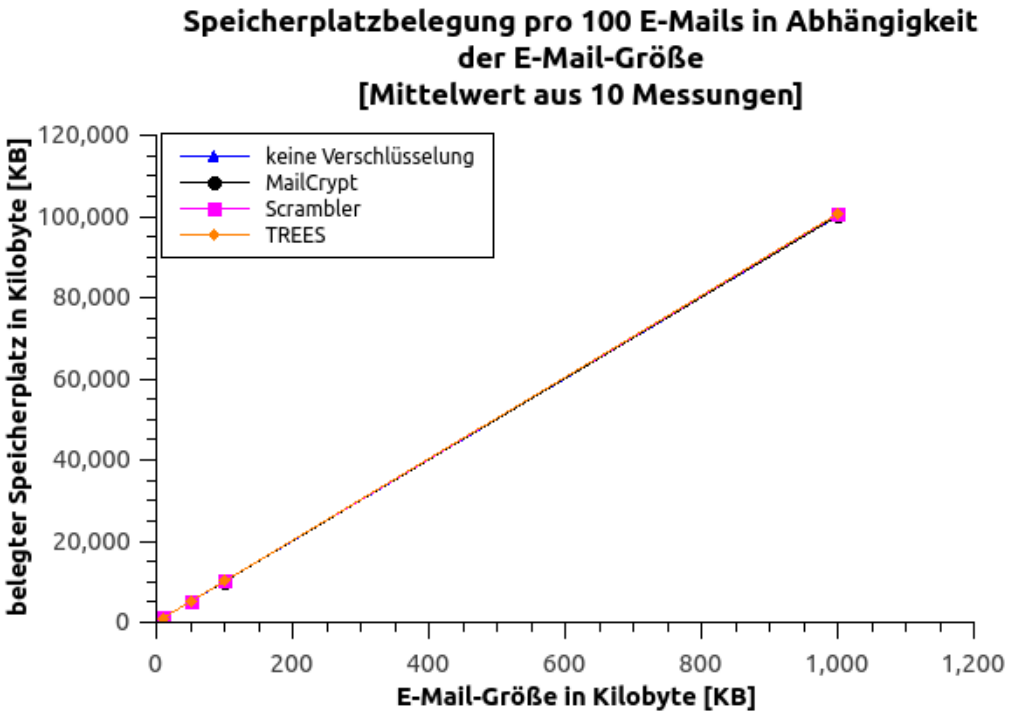

Abbildung 15: Mittlere Speicherplatzbelegung pro 100 E-Mails in Abhängigkeit der E-Mail-Größe [Mittelwert aus 10 Messungen]

Speicherplatzbelegung nach E-Mail-Größe / Technik	10 KB [KB]	50 KB [KB]	100 KB [KB]	1 MB [KB]
keine Verschlüsselung	1056.7 ± 0	5056.7 ± 0	10056.8 ± 0	100056.8 ± 0
MailCrypt	1082.2 ± 0	5082.2 ± 0	10082.4 ± 0	100082.4 ± 0
Scrambler	1092.5 ± 0	5109.5 ± 0	10130.1 ± 0	100504.1 ± 0
TREES	1062.0 ± 0	5086.0 ± 0	10115.0 ± 0	100643.0 ± 0

Tabelle 9: Mittlere Speicherplatzbelegung pro 100 E-Mails in Abhängigkeit der E-Mail-Größe [Mittelwert aus 10 Messungen]

Aus den Messergebnissen kann man schlussfolgern, dass mit ansteigender E-Mail-Größe auch der Speicherplatzbedarf ansteigt (siehe Abbildung 15). Weiterhin kann man erkennen, dass durch den Einsatz der Verschlüsselungstechniken zusätzlicher Speicherplatz belegt wird. Die Größe des zusätzlich benötigten Speicherplatzes ist hierbei abhängig von der eingesetzten Technik und der jeweiligen E-Mail-Größe. Zusätzlich ist ersichtlich, dass die Messwerte exakt reproduzierbar sind (siehe Tabelle 9).

Nun wird aus den Messwerten die zusätzliche Speicherplatzbelegung pro E-Mail berechnet.

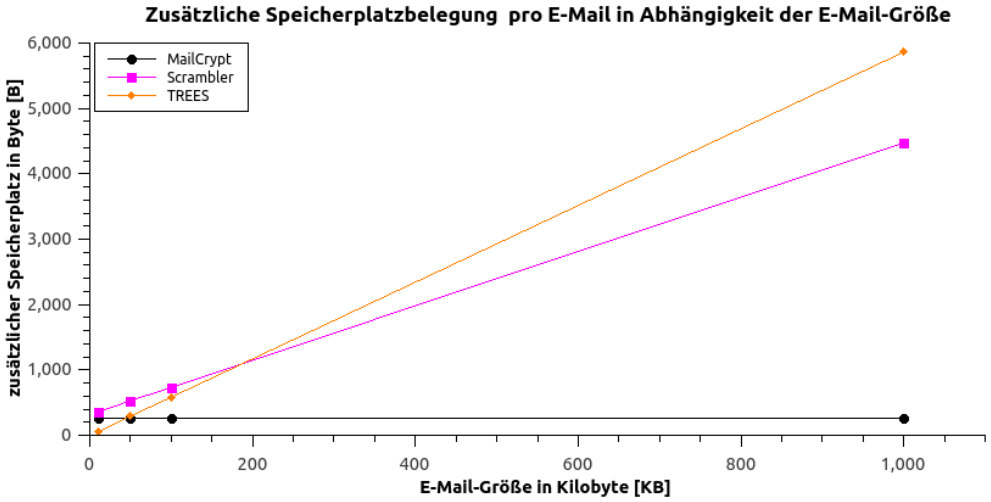

Abbildung 16: Zusätzliche Speicherplatzbelegung pro E-Mail in Abhängigkeit der E-Mail-Größe

Technik \ zusätzliche Speicherplatzbelegung nach E-Mail-Größe	10 KB [B]	50 KB [B]	100 KB [B]	1 MB [B]
MailCrypt	255	255	256	256
Scrambler	358	528	733	4473
TREES	53	293	582	5862

Tabelle 10: Zusätzliche Speicherplatzbelegung pro E-Mail in Abhängigkeit der E-Mail-Größe

In Abbildung 16 und Tabelle 10 ist der zusätzliche Speicherplatz aufgetragen, der pro E-Mail benötigt wird. Man kann erkennen, dass je nach eingesetzter Technik unterschiedlich viel zusätzlicher Speicherplatz belegt wird.

MailCrypt benötigt unabhängig von der E-Mail-Größe 255 Byte bzw. 256 Byte zusätzlichen Speicherplatz. Dort werden die Informationen abgespeichert, die zur Entschlüsselung notwendig sind [dovecot.org, 2017, File Format]. Diese sind relativ statisch und belegen deswegen immer den gleichen Speicherplatz.

Beim Einsatz von Scrambler und TREES hingegen steigt der zusätzliche Speicherplatzbedarf mit steigender E-Mail-Größe an. Scrambler benötigt 16 Byte zusätzlichen Speicherplatz pro Kilobyte E-Mail-Größe. TREES benötigt 5.87 Byte zusätzlichen Speicherplatz pro Kilobyte E-Mail-Größe. Es ist zu vermuten, dass in den zusätzlichen Bytes die jeweiligen Header und Informationen zur Integritätsüberprüfung enthalten sind. Weiterhin ist davon auszugehen, dass die Informationen zur Integritätsprüfung mit steigender E-Mail-Größe anwachsen. Eine eingehende E-Mail wird in sogenannte „Chunks“ zerlegt, welche einzeln verschlüsselt und mit einer Prüfsumme versehen werden. Je größer die eingehende E-Mail ist, umso mehr Prüfsummen müssen erstellt und abgespeichert

werden. Dies würde den linearen Zusammenhang zwischen dem zusätzlich benötigten Speicherplatz und der E-Mail-Größe erklären.

Allgemein fällt der zusätzlich benötigte Speicherplatz so gering aus, dass er im produktiven Einsatz nicht weiter ins Gewicht fallen sollte.

4.3.5.4. Fazit

In diesem Test konnte gezeigt werden, dass durch den Einsatz von Verschlüsselungstechniken zusätzlicher Speicherplatz belegt wird. Dieser fällt abhängig von der eingesetzten Technik und der E-Mail-Größe unterschiedlich groß aus.

Die Größe des zusätzlich belegten Speicherplatzes ist aber so gering, dass er im produktiven Einsatz nicht weiter ins Gewicht fallen sollte.

4.3.6. Einfluss auf die Anzahl der IMAP-Logins

4.3.6.1. Versuchsbeschreibung

Im Kapitel „Stand der Technik“ wurde gezeigt, dass zum Entschlüsseln der gespeicherten E-Mails der entschlüsselte, private Schlüssel des Nutzers vorliegen muss. Dieser muss im Rahmen des Login-Vorgangs entschlüsselt werden.

In diesem Versuch soll untersucht werden, ob der Einsatz von Postfachverschlüsselungsplugins einen Einfluss auf die Anzahl der durchführbaren IMAP-Logins hat. Weiterhin soll untersucht werden, ob die unterschiedlichen Ansätze zur Entschlüsselung des privaten Schlüssels, einen Einfluss auf die Anzahl der durchführbaren IMAP-Logins haben.

In diesem Versuch wird ausschließlich der Login-Vorgang via IMAP untersucht. Es wird davon ausgegangen, dass potentielle Einflüsse unabhängig vom eingesetzten Protokoll sind. Auf eine Untersuchung unter Einsatz von POP3 wird deswegen verzichtet.

4.3.6.2. Versuchsdurchführung

Zur Durchführung des Versuchs wird ein Skript entwickelt. Es testet, wie viele Logins pro Minute auf einem E-Mail-Server durchführbar sind. Dies geschieht mit einer variablen Anzahl von IMAP-Clients. Mit Übergabeparametern kann unter anderem bestimmt werden, wie viele gleichzeitige Clients simuliert werden sollen und wie oft der Test wiederholt werden soll. Das Skript setzt zur Login-Durchführung auf das Programm ImapTest. Es wird auf dem Server „imaptester.test“ aufgerufen und führt die Logins auf dem zu testenden System durch.

Abschließend fragt das Skript die Prometheus-API des Monitoring-Servers an. Es erhält daraufhin die Monitoring-Daten des zu testenden Servers für die Dauer des Tests. Zu den Monitoring-Daten gehören die durchschnittlichen Werte für RAM-Verbrauch, CPU-Auslastung, I/O-Wait und die Load. Der I/O-Wait-Wert gibt hierbei die Zeitdauer an, die eine CPU auf den Abschluss von I/O-Operationen warten muss. Der Load-Wert gibt hierbei einen Durchschnittswert an, wie viele Prozesse momentan durch die CPU verarbeitet werden oder auf die Verarbeitung warten.

Nach Abschluss des Tests gibt das Skript die Anzahl der erfolgreich durchgeführten Logins und die Monitoring-Daten aus.

Dieser Versuch wird für 1, 10 und 100 IMAP-Clients durchgeführt und jeweils 30 Mal wiederholt (siehe Listing 11). Um eine mögliche Beeinflussung der Versuchsergebnisse zu verhindern, werden während des Tests alle nicht benötigten VMs ausgeschaltet.

Listing 11: Exemplarischer Testaufruf des IMAP-Login-Tests für den Host mailcrypt.test mit einem IMAP-Client und 30 Wiederholungen

```
user@hypervisor:07_Testumgebung $ ansible all -i ansible/hosts --private-key=/home/paul/Desktop/
    Masterarbeit/07_Testumgebung/keys/vagrant.id_ecdsa -u vagrant -l imaptester.test -m shell -a '/
    bin/bash -c ``./imaplogins.sh mailcrypt.test mailcrypt 1 30``'
```

4.3.6.3. Versuchsauswertung

Aus den aufgenommenen Messwerten wird der Mittelwert und die zugehörige Standardabweichung gebildet. Die resultierenden Werte sind in Abbildung 17 visualisiert.

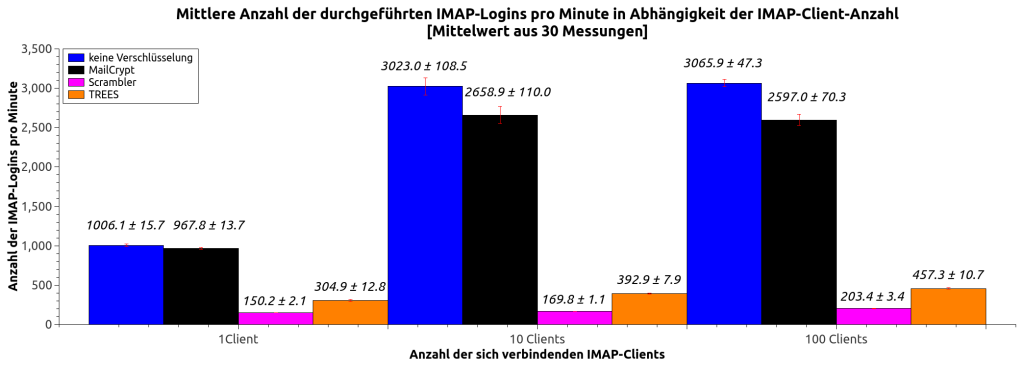

Abbildung 17: Mittlere Anzahl der durchgeführten IMAP-Logins pro Minute in Abhängigkeit der IMAP-Client-Anzahl [Mittelwert aus 30 Messungen]

Technik \ Anzahl der IMAP-Clients	1 Client		10 Clients		100 Clients	
statistische Kenngrößen	Mittelwert	Standardabweichung	Mittelwert	Standardabweichung	Mittelwert	Standardabweichung
keine Verschlüsselung	1006.1	15.7	3023.0	108.5	3065.9	47.3
MailCrypt	967.8	13.7	2658.9	110.0	2597.0	70.3
Scrambler	150.2	2.1	169.8	1.1	203.4	3.4
TREES	304.9	12.8	392.9	7.9	457.3	10.7

Tabelle 11: Mittlere Anzahl der durchgeführten IMAP-Logins pro Minute in Abhängigkeit der IMAP-Client-Anzahl [Mittelwert aus 30 Messungen]

Aus den Messwerten (siehe Abbildung 17, Tabelle 11) kann man erkennen, dass der Einsatz der Verschlüsselungstechniken die Anzahl der durchführbaren IMAP-Logins pro Minute verringert. Dies entspricht den Erwartungen, da mit dem Einsatz der Verschlüsselungstechniken zusätzliche Operationen ausgeführt werden müssen, um den privaten Schlüssel des Nutzers zu entschlüsseln.

Man kann erkennen, dass die Verringerung abhängig vom eingesetztem Plugin und Anzahl der gleichzeitigen IMAP-Clients ist (siehe Tabelle 12). Besonders bei den Plugins Scrambler und TREES ist eine erhebliche Reduzierung der erfolgreichen IMAP-Logins zu beobachten. Beim Einsatz des Scrambler Plugins sind je nach Client-Anzahl ca. 85%-94% weniger Logins möglich, als beim Einsatz keiner Verschlüsselungstechnik. Auch beim Einsatz des TREES Plugins werden die durchführbaren Logins um 70%-85% reduziert.

Technik \ Anzahl der IMAP-Clients	1 Client		10 Clients		100 Clients	
Abweichung	Absolut	Relativ [%]	Absolut	Relativ [%]	Absolut	Relativ [%]
MailCrypt	38.3	3.8	364.1	12.0	468.9	15.3
Scrambler	855.9	85.1	2853.2	94.4	2862.5	93.4
TREES	701.3	69.7	2630.1	87.0	2608.6	85.1

Tabelle 12: Mittlere absolute und relative Abweichung der durchgeführten IMAP-Logins pro Minute zum E-Mail-Server ohne Postfachverschlüsselung in Abhängigkeit der IMAP-Client-Anzahl

Technik \ Anzahl der IMAP-Clients	1 Client				10 Clients				100 Clients			
Monitoring-Daten für	RAM-Verbrauch [MB]	CPU-Auslastung [%]	I/O-Wait	Load	RAM-Verbrauch [MB]	CPU-Auslastung [%]	I/O-Wait	Load	RAM-Verbrauch [MB]	CPU-Auslastung [%]	I/O-Wait	Load
keine Verschlüsselung	14.24	34.64	0.72	0.6	13.03	98.37	0.75	4.93	112.83	98.39	0.39	5.27
MailCrypt	19.43	38.03	0.88	0.57	17.02	98.36	0.93	6.12	109.24	98.42	0.72	7.62
Scrambler	12.73	88.75	1.01	1.04	26.24	98.36	1.079	10.14	105.45	98.66	1.12	86.64
TREES	34.96	78.88	1.02	0.99	255.08	98.39	1.06	10.09	1425.11	98.61	5.69	54.16

Tabelle 13: Monitoring-Daten in Abhängigkeit der IMAP-Client-Anzahl

In Tabelle 13 sind die Monitoring-Daten des Tests dargestellt. Dort kann man erkennen, dass, bereits bei Verbindungen mit nur einem IMAP-Client, die Ressourcen-Belastung unterschiedlich stark ausfällt.

Weiterhin zeigen die Monitoring-Daten, dass der E-Mail-Server ohne Postfachverschlüsselung nicht vollständig ausgelastet ist. Gleiches gilt für den Server mit aktivem MailCrypt Plugin. Beim Einsatz von Scrambler und TREES hingegen, ist die CPU-Auslastung deutlich höher. Dies schlägt sich zum Beispiel in einer höheren Load nieder.

Beim gleichzeitigen Login von zehn IMAP-Clients arbeiten alle Systeme an ihrer Belastungsgrenze. Dies kann man daran erkennen, dass die Load deutlich über 1 liegt. Auch hier sind die E-Mail-Server mit Scrambler und TREES deutlich stärker belastet, als die anderen beiden Systeme. Die Load beim Einsatz des Scrambler- und TREES-Plugins ist zum Beispiel ungefähr doppelt so hoch, wie bei den anderen beiden Techniken.

Beim gleichzeitigen Login von 100 IMAP-Clients sind alle Systeme noch stärker belastet. Das kann man zum Beispiel am gestiegenen RAM-Verbrauch und der erhöhten Load erkennen. Betrachtet man die Load der verschiedenen Systeme, ist ersichtlich, dass es einen gravierenden Unterschied zwischen den Plugins gibt, wenn die Client-Anzahl von 10 Clients auf 100 Clients erhöht wird. Alle Systeme verzeichnen einen Anstieg der Load. Beim Einsatz von Scrambler und TREES ist dieser besonders hoch.

Weiterhin steigt beim TREES Plugin auch der I/O-Wait-Wert zwischen den Tests mit 10 und 100 Clients an. Das bedeutet, dass die CPU in diesem Fall deutlich länger auf den Abschluss von I/O-Operationen warten muss, als bei Tests mit weniger Clients. Vermutlich liegt es daran, dass das TREES-Plugin eine Memory-hard Hash-Function (MHF) verwendet und bei 100 Clients immer wieder auf den Abschluss der vorhergehenden Hash-Berechnungen gewartet werden muss, bevor neue Berechnungen im RAM stattfinden können. Weiterhin kann man vermuten, dass diese Wartezeit auch die Ursache für eine geringere Load, im Vergleich zum Scrambler Plugin, ist. Zusätzlich kann man erkennen, dass das TREES-Plugin bei allen drei Tests deutlich mehr RAM als die anderen Techniken verbraucht. Dies liegt ebenfalls an der Verwendung der MHF und entspricht den Erwartungen.

Aufgrund der dargestellten Beobachtungen wird vermutet, dass der deutlich höhere Ressourcenverbrauch, der beim Einsatz des Scrambler-und TREES-Plugins beobachtet werden konnte, auf deren Software-Design zurückzuführen ist. Beide Techniken verwenden ressourcen-intensive Hash-Funktionen, um nach dem Login einen Hash aus dem Nutzer-Passwort zu bilden (siehe Abbildung 3.5.4 und Abbildung 3.6.4). Scrambler setzt hierbei auf die Funktion bcrypt und TREES auf die Funktion Argon2.

4.3.6.4. Untersuchung mit erhöhten bcrypt- und Argon2-Parametern

Um diese Vermutung zu überprüfen, wird ein weiterer Tests durchgeführt, bei dem die Parameter der jeweiligen Hash-Funktionen (Scrambler: bcrypt, TREES: Argon2) erhöht

werden. Das hat zur Folge, dass die jeweilige Hash-Berechnung länger dauert und zusätzliche System-Ressourcen zur Berechnung notwendig sind.

Der bcrypt-Hash des Nutzer-Passworts wird für diesen Test mit 16 anstatt 12 Iterationen berechnet. Der Argon2-Hash des Nutzer-Passworts wird für diesen Test vom „interactive“-Modus auf den „moderate“-Modus umgestellt. Das hat zur Folge, dass die notwendigen Rechenoperationen von 4 auf 6 Operationen ansteigen und pro Hash 128 MB anstatt 32 MB RAM belegt werden.

Daraufhin werden die Hash-Werte berechnet und in der Datenbank abgelegt. Sie können im Branch „harder_argon_harder_bcrypt_values“ des Git-Repositories gefunden werden. Bei jedem zukünftigen Login müssen nun die neuen Hash-Werte berechnet werden.

Nach der Umstellung wird der Versuch erneut durchgeführt und 15 mal wiederholt. Daraufhin werden der Mittelwert und die Standardabweichung gebildet und die Werte visualisiert (siehe 18).

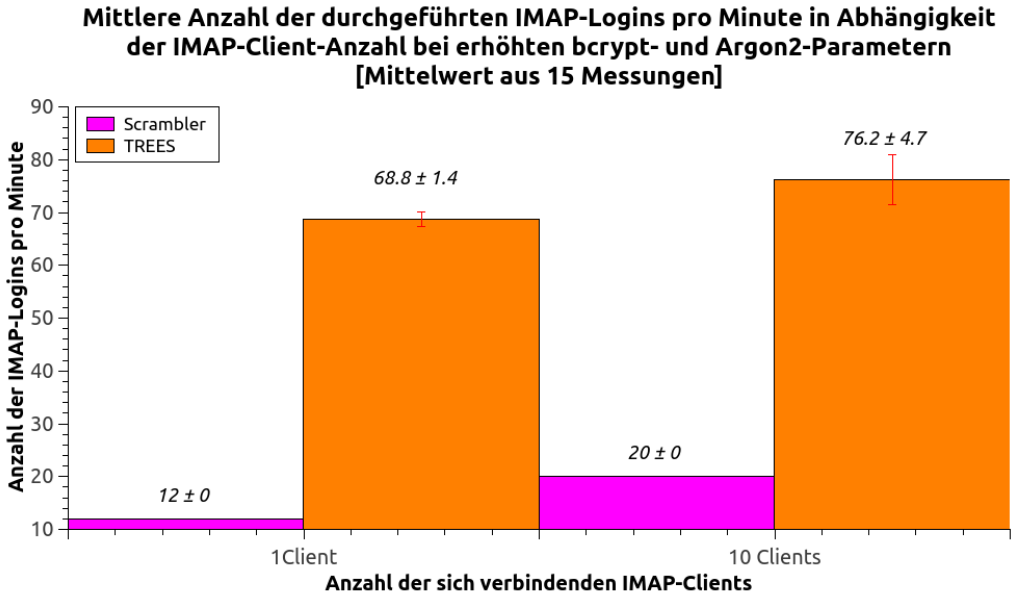

Abbildung 18: Mittlere Anzahl der durchgeführten IMAP-Logins pro Minute in Abhängigkeit der IMAP-Client-Anzahl bei erhöhten bcrypt- und Argon2-Parametern [Mittelwert aus 15 Messungen]

Anzahl der IMAP-Clients / Technik	1 Client		10 Clients	
statistische Kenngrößen	Mittelwert	Standardabweichung	Mittelwert	Standardabweichung
Scrambler	12.0	0	20.0	0
TREES	68.8	1.4	76.2	4.7

Tabelle 14: Mittlere Anzahl der durchgeführten IMAP-Logins pro Minute in Abhängigkeit der IMAP-Client-Anzahl bei erhöhten bcrypt- und Argon2-Parametern [Mittelwert aus 15 Messungen]

Die Messergebnisse zeigen, dass eine Erhöhung der Hash-Berechnungsparameter zu einer gravierenden Reduzierung der IMAP-Logins bei gleicher Client-Anzahl führt (siehe Abbildung 18, Tabelle 14). Dies wird besonders ersichtlich, wenn man die absoluten und relativen Abweichungen zur vorherigen Messung (Tabelle15) betrachtet.

Anzahl der IMAP-Clients / Technik	1 Client		10 Clients	
Abweichung	Absolut	Relativ [%]	Absolut	Relativ [%]
Scrambler	138.2	92.01	149.8	88.22
TREES	236.1	77.44	316.7	80.61

Tabelle 15: Mittlere absolute und relative Abweichung der durchgeführten IMAP-Logins pro Mintute im Vergleich zur Messung mit den voreingestellten Hash-Berechnungsparametern in Abhängigkeit der IMAP-Client-Anzahl

Diese Ergebnisse untermauern die Vermutung aus dem vorherigen Testschritt. Sie zeigen, dass die geringere Anzahl der IMAP-Logins beim Einsatz von Scrambler und TREES durch die zusätzlichen Hash-Berechnungen verursacht werden.

Die Monitoring-Daten (Tabelle 16) zeigen, dass sich mit ansteigender Client-Anzahl auch die Load auf den Systemen erhöht. Weiterhin wächst mit ansteigender Client-Anzahl auch der RAM-Verbrauch und die CPU-Auslastung. Im Vergleich zum vorherigen Test kann man erkennen, dass bereits bei der Verbindung eines IMAP-Client die CPU ausgelastet ist. Die Ergebnisse entsprechen den Erwartungen, da beide Hash-Algorithmen angewiesen sind, mehr Ressourcen zu belegen.

Anzahl der IMAP-Clients / Technik	1 Client				10 CLients			
Monitoring-Daten für	RAM-Verbrauch [MB]	CPU-Auslastung [%]	I/O-Wait	Load	RAM-Verbrauch [MB]	CPU-Auslastung [%]	I/O-Wait	Load
Scrambler	2.81	97.76	1.27	1.06	20.89	99.1	1.32	10.25
TREES	107.82	94.04	1.22	1.12	1123.4	98.52	1.25	10.12

Tabelle 16: Monitoring-Daten in Abhängigkeit der sich gleichzeitig verbindenden IMAP-Clients

Eine Durchführung des Tests mit 100 IMAP-Clients war sowohl mit dem Scrambler-Plugin als auch mit dem TREES-Plugin nicht möglich. Dies führte zu einer reproduzierbaren Dienstverweigerung des jeweiligen E-Mail-Servers.

Vom Scrambler-Server konnten keine E-Mails mehr versendet oder abgerufen werden. Im Prozessmanager htop[67] konnte zudem eine deutliche Belastung des E-Mail-Servers festgestellt werden. Bei Verbindung mit dem TREES-Server meldete das eingesetzte Test-Skript Fehler. Weiterhin war kein E-Mail-Versand oder -Empfang mehr möglich und auch hier zeigte die htop-Ansicht des Servers eine deutliche Belastung von CPU und RAM. Zusätzlich konnte man sehen, dass der E-Mail-Server swappt. Das bedeutet, dass Daten aus dem RAM auf die Festplatte ausgelagert werden. Sollen die ausgelagerten Daten wiederverwendet werden, müssen sie von der Festplatte wieder in den Arbeitsspeicher geladen werden. Dies ist deutlich langsamer, als wenn sich die Daten im RAM befinden würden. Dadurch wird das System spürbar verlangsamt.

Aus diesen Beobachtung geht hervor, dass erhöhte bcrypt- und Argon2-Parameter die Möglichkeit einer Denial-of-Service (DOS)-Attacke eröffnen. Diese ist bereits mit der Kenntnis, einer einzigen gültigen Zugangsdaten-Kombination und einer gewissen Anzahl von IMAP-Clients möglich. Es wird vermutet, dass eine DOS-Attacke auch bei der Verwendung der Standard-Parameter der Hash-Algorithmen möglich ist.

4.3.6.5. Fazit

Aus diesem Versuch geht hervor, dass der Einsatz von Postfachverschlüsselungstechniken die Anzahl der erfolgreichen IMAP-Logins pro Minute verringert. Diese Reduzierung fällt je nach eingesetzter Technik unterschiedlich stark aus.

Beim Einsatz von Scrambler und TREES werden die möglichen Logins besonders stark reduziert. Die Ursache dafür liegt vermutlich an den zusätzlichen Hash-Berechnungen, die diese Plugins durchführen, um den privaten Schlüssel des Nutzers zu entschlüsseln.

Weiterhin wurde in diesem Test gezeigt, dass der Einsatz von Scrambler und TREES einem Angreifer die Möglichkeit einer DOS-Attacke eröffnet. Diese ist mit relativ einfachen Mitteln durchführbar.

[67] `http://hisham.hm/htop/`

4.3.7. Einfluss auf die E-Mail-Abrufdauer

4.3.7.1. Versuchsbeschreibung

In diesem Versuch soll untersucht werden, ob und welchen Einfluss der Einsatz von Postfachverschlüsselungstechniken auf die Dauer des E-Mail-Abrufs hat. Zusätzlich soll untersucht werden, ob mögliche Einflüsse abhängig vom eingesetzten Abruf-Protokoll sind.

4.3.7.2. Versuchsdurchführung

Für diesen Versuch wird ein Skript entwickelt, welches auf dem Hypervisor ausgeführt wird und mehrere Aufgaben durchführt. Zuerst ruft es auf dem zu testenden E-Mail-Server ein Skript auf, das 100 E-Mails mit einer Größe von 50 KB dem lokalen Postfach zustellt. Danach lädt es die E-Mails in ein Postfach auf den Hypervisor herunter. Hierzu wird das Programm getmail[68] verwendet. Es verwendet die im Repository hinterlegten getmail-Konfigurationsdateien. Mit Hilfe des Programms time wird die Zeit gemessen, die zum Abruf der E-Mails benötigt wird. Danach werden die heruntergeladenen E-Mails auf dem Hypervisor gelöscht und der Test erneut durchgeführt. Der Abrufvorgang wird 100 Mal wiederholt.

Dem Skript müssen als Parameter der zu testende Server und das zu verwendende Protokoll übergeben werden (siehe Listing 12).

Listing 12: Exemplarischer Testaufruf des E-Mail-Abrufdauertests für den Host noencryption.test unter Verwendung von POP3

```
user@hypervisor:07_Testumgebung $ ./recieve_mails.sh noencryption.test pop3
```

Um mögliche Einflüsse auf die Testergebnisse zu vermeiden, sind während des Tests alle anderen VMs ausgeschaltet.

4.3.7.3. Versuchsauswertung

Aus den aufgenommenen Messwerten wird der Mittelwert und die zugehörige Standardabweichung gebildet. Die resultierenden Werte sind in den Abbildung 19 und 20 visualisiert.

[68] `http://pyropus.ca/software/getmail/`

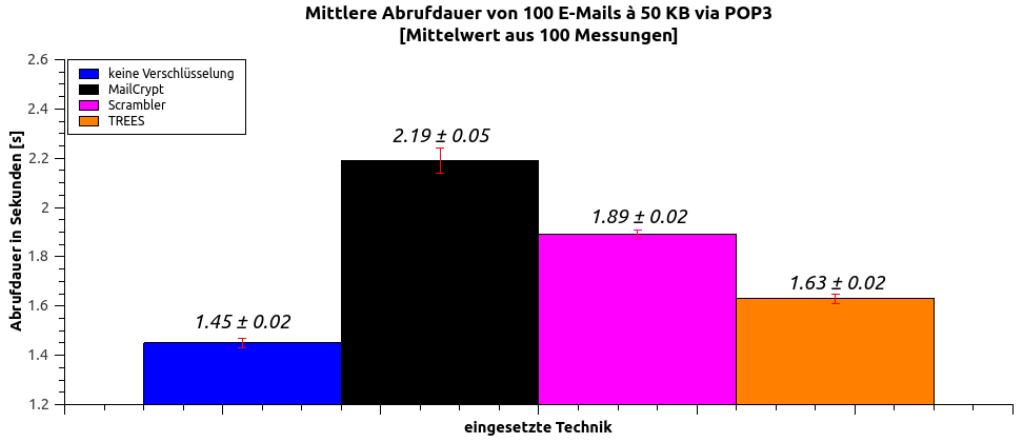

Abbildung 19: Mittlere Abrufdauer von 100 E-Mails à 50 KB via POP3 [Mittelwert aus 100 Messungen]

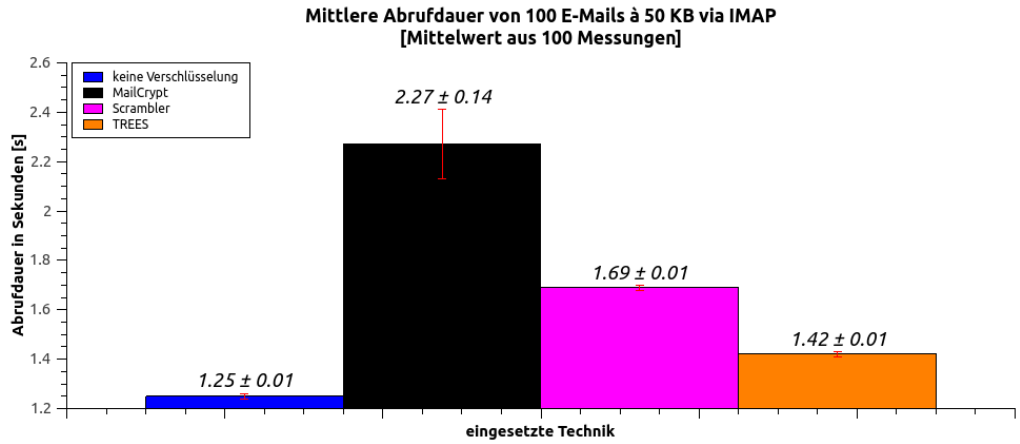

Abbildung 20: Mittlere Abrufdauer von 100 E-Mails à 50 KB via IMAP [Mittelwert aus 100 Messungen]

Aus den Abbildungen 19 und 20 kann man erkennen, dass die Abrufdauer von 100 E-Mails à 50 KB durch den Einsatz von Postfachverschlüsselungstechniken verlängert wird. Weiterhin ist ersichtlich, dass die Stärke der Verzögerung von der eingesetzten Technik abhängig ist. Das MailCrypt-Plugin verzögert den E-Mail-Abruf am stärksten, gefolgt vom Scrambler- und TREES-Plugin. Hierbei ist es unerheblich, ob die E-Mails über POP3 oder IMAP abgerufen werden.

Zusätzlich kann man aus diesen Werten eine Besonderheit im Verhalten des MailCrypt-Plugins erkennen. Beim Wechsel des Abrufprotokolls von POP3 auf IMAP steigen die Werte für die Abrufdauer an, während sie für alle anderen Techniken absinken.

eingesetztes Protokoll / Technik	POP3		IMAP	
Abweichung	Absolut [s]	Relativ [%]	Absolut [s]	Relativ [%]
MailCrypt	0.74	51.11	1.03	82.64
Scrambler	0.44	30.55	0.44	35.52
TREES	0.18	12.54	0.18	14.37

Tabelle 17: Mittlere absolute und relative Abweichung der E-Mail-Abrufdauer von 100 E-Mails à 50 KB in Abhängigkeit des eingesetzten Protokolls

Aus den relativen und absoluten Abweichungen in Tabelle 17 kann man die Stärke der Verzögerung ablesen. Die relative Verzögerung scheint bei einem Abruf über POP3 geringer auszufallen, als beim E-Mail-Abruf über IMAP.

4.3.7.4. Fazit

Im Rahmen dieses Versuchs konnte festgestellt werden, dass der Einsatz von Postfachverschlüsselungstechniken einen verzögernden Einfluss auf die E-Mail-Abrufdauer hat. Dieser fällt in Abhängigkeit der eingesetzten Plugins unterschiedlich stark aus.

Die hervorgerufene Verzögerung fällt in absoluten Zeiten jedoch so gering aus, dass sie vom Nutzer vermutlich unbemerkt bleibt.

4.3.8. Einfluss auf den Nutzerkontenerstellungsprozess

4.3.8.1. Versuchsbeschreibung

Damit die Postfachverschlüsselungstechniken eingehende E-Mails ver- und entschlüsseln können, muss für jeden Nutzer das notwendige Schlüsselmaterial vorliegen. Dieses muss im Rahmen des Nutzerkontoerstellungsprozesses erzeugt und verschlüsselt abgespeichert werden.

In diesem Versuch soll untersucht werden, ob und welchen Einfluss der Einsatz der Techniken auf die Erstellung von Nutzerkonten hat. Hierbei wird untersucht, wie viele Nutzerkonten pro Minute erstellt werden können.

4.3.8.2. Versuchsdurchführung

Zur Durchführung des Versuchs wird ein Skript entwickelt, welches neue Nutzer inklusive dem zugehörigen Schlüsselmaterial erstellt.

Beim Einsatz von MailCrypt wird das zugehörige doveadm-Kommando[69] verwendet. Für die Plugins Scrambler und TREES werden externe Skripte aufgerufen.

Dieser Test wird für die Dauer von einer Minute durchgeführt und wird 30 Mal wiederholt (siehe Listing 13).

Listing 13: Exemplarischer Testaufruf des Nutzerkontenerstellungstests für mailcrypt.test

```
user@hypervisor:Testumgebung $ ./run_test_x_times.sh 1 ‘‘time ./user_creation.sh‘‘ mailcrypt.test
```

4.3.8.3. Versuchsauswertung

Aus den aufgenommenen Messwerten wird der Mittelwert und die Standardabweichung gebildet. Die resultierenden Werte sind in Abbildung 21 visualisiert.

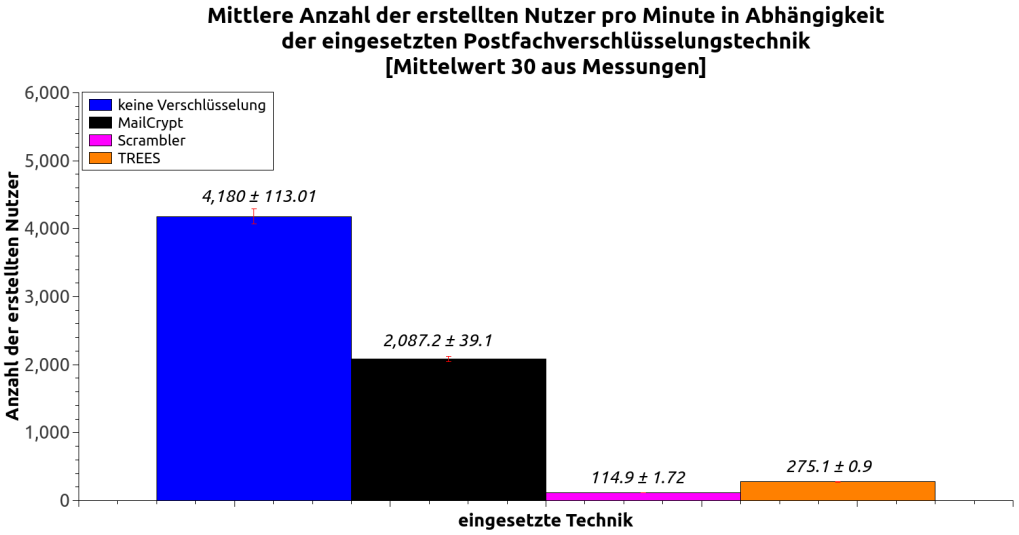

Abbildung 21: Mittlere Anzahl der erstellten Nutzer pro Minute in Abhängigkeit der eingesetzten Technik [Mittelwert aus 30 Messungen]

Aus Abbildung 21 ist ersichtlich, dass der Einsatz von Postfachverschlüsselungstechniken erhebliche Auswirkungen auf die Anzahl der erstellbaren Nutzer pro Minute hat. Durch den Einsatz der Techniken wird die Anzahl der erstellten Nutzer pro Minute deutlich reduziert. Es ist davon auszugehen, dass die Anzahl der erstellbaren Nutzerkonten

[69] `https://wiki.dovecot.org/Plugins/MailCrypt`

pro Minute durch die Erzeugung und Verschlüsselung des notwendigen Schlüsselmaterials begrenzt wird.

Aus Tabelle 18 kann man die absoluten und relativen Abweichungen der Messwerte zu einem E-Mail-Server ohne Postfachverschlüsselungstechnik entnehmen. Aus den Daten ist das Ausmaß der Reduzierung ersichtlich. Die Anzahl der erstellten Nutzerkonten wird bei der Verwendung von MailCrypt um ca. 50%, beim Einsatz von TREES um ca. 93% und bei der Verwendung von Scrambler um ca. 97% und reduziert.

Technik	**Abweichung zur Nutzererstellung ohne Verschlüsselungstechnik**	
Abweichung	Absolut	Relativ [%]
MailCrypt	2088.37	50.01
Scrambler	4060.63	97.25
TREES	3900.43	93.41

Tabelle 18: Mittlere absolute und relative Abweichung der erstellten Nutzer pro Minute zum E-Mail-Server ohne Postfachverschlüsselungstechnik

Man kann erkennen, dass es einen deutlichen Unterschied zwischen der Verwendung von MailCrypt und den Plugins Scrambler und TREES gibt. Es wird vermutet, dass die Ursache in deren Software-Design liegt.

Die Messergebnisse lassen vermuten, dass das MailCrypt-Plugin das notwendige Schlüsselmaterial schneller erstellt, als Scrambler und TREES. Im Gegensatz zu MailCrypt Scrambler und TREES ressourcen-intensivere Hash-Funktionen zur Berechnung des Hashes, mit dem das Schlüsselmaterial verschlüsselt wird. Diese Berechnungen dauern entsprechend lange und verringern somit die Anzahl der erstellbaren Nutzerkonten pro Minute.

4.3.8.4. Fazit

Aus dem Versuch geht hervor, dass die Anzahl der erstellbaren Nutzer pro Minute durch den Einsatz von Postfachverschlüsselungstechniken reduziert wird. Je nach eingesetzter Technik fällt die Reduzierung unterschiedlich stark aus. Sie beträgt zwischen 50% (MailCrypt) und 97% (Scrambler).

4.4. Fehlerbetrachtung

Alle beschriebenen Versuche wurden in virtuellen Maschinen durchgeführt. Jede VM greift hierbei auf die begrenzten Ressourcen des Hypervisors zurück. Sollte der Hypervisor im Rahmen eines Versuches unter unerwarteter Last stehen oder an der Grenze seiner Ressourcen arbeiten, kann dies die Messergebnisse beeinflussen.

Durch die Wahl dieser Methode könnten die Messergebnisse mit Fehlern behaftet sein. Da externe Einflüsse nicht vollständig ausgeschlossen werden konnten, wurden Maßnahmen ergriffen, um mögliche Fehler weitestgehend zu minimieren.

Hierzu zählen die Minimierung der Last auf dem Hypervisor, durch den ausschließlichen Betrieb der im Test benötigten VMs, die Überwachung des Hypervisors, die Wahl eines großen Messbereichs, die vielfache Wiederholung der Versuchsreihen, sowie die statistische Auswertung der Messergebnisse.

4.5. Zusammenfassung

Im Rahmen der Untersuchung konnte gezeigt werden, dass der **Einsatz** von **Postfachverschlüsselungstechniken** zu **Performance-Einbußen** des E-Mail-Servers führt. In Abhängigkeit der eingesetzten Technik fallen die Einbußen unterschiedlich stark aus.

Zuerst stellt sich dar, dass nicht alle Techniken mit den getesteten Dovecot-Postfachformaten kompatibel sind. Das trifft für MailCrypt, Scrambler und TREES beim Einsatz des „mbox“-Formats zu.

Weiterhin konnte festgestellt werden, dass die Plugins **Scrambler** und **TREES nicht vollständig mit** dem **Postfachformat „maildir“ kompatibel** sind. Beim gleichzeitigen Empfang und Abruf von E-Mails über POP3, trat ein reproduzierbarer Softwarefehler auf. Daraufhin wurde sich für die Verwendung des Postfachformats „mdbox“ entschieden.

Aus den Ergebnissen des Versuchs „Einfluss auf die E-Mail-Empfangsdauer“ ist erkennbar, dass der **Einsatz** der **Postfachverschlüsselungstechniken** den **E-Mail-Empfang verzögert**. Dabei verlangsamt MailCrypt, Scrambler und TREES den E-Mail-Empfang nur geringfügig (ca. 10%). Dagegen ist die Verzögerung für die GPG-SieveFilter-Technik so gravierend (ca. 510%), dass deren Einsatz für eine große Nutzer-Anzahl als ungeeignet erscheint. Außerdem setzt die Technik beim Nutzer Kenntnisse über die Verwendung von GPG voraus. Aufgrund dieser Umstände wurde die GPG-Sieve-Filter-Technik für alle weiteren Versuche als ungeeignet befunden.

Die Beobachtung des verzögerten E-Mail-Empfangs konnte auch im Versuch „Einfluss auf die E-Mail-Empfangsdauer in Abhängigkeit der E-Mail-Größe“ bestätigt werden: mit ansteigender E-Mail-Größe wird die hervorgerufene Verzögerung immer geringer. Der Empfang einer Standard-E-Mail (50 KB) dauert ca. 4% (TREES), ca. 7%(Scrambler), ca. 20% (MailCrypt) länger, als der Empfang ihres unverschlüsselten Pendants. Da E-Mail in der Regel nicht als Echtzeitkommunikationsmedium eingesetzt wird, sollte diese Verzögerung für den Nutzer unbemerkt bleiben.

Im Versuch „Einfluss auf den benötigten Speicherplatz“ wurde beobachtet, dass durch den Einsatz der Techniken **zusätzlicher Speicherplatz belegt** wird. Dieser fällt je nach Plugin unterschiedlich groß aus. Beim Empfang einer Standard-E-Mail (50 KB) werden zusätzlich 255 Byte (MailCrypt), 293 Byte (TREES) und 528 Byte (Scrambler) benötigt. Beim Einsatz von MailCrypt ist der zusätzliche Speicherplatzbedarf konstant und unabhängig von E-Mail-Größe. Dagegen wächst der zusätzlich benötigte Speicherplatz bei der Verwendung von Scrambler und TREES linear. Das Wachstum beträgt beim Einsatz von Scrambler 4.16 Byte / Kilobyte E-Mail-Größe und beim Einsatz von TREES 5.87 Byte / Kilobyte E-Mail-Größe. Damit ist der zusätzlich benötigte Speicherplatz so gering, dass er im produktiven Betrieb vernachlässigbar sein sollte.

Im folgenden Versuch wurde der „Einfluss auf die Anzahl der IMAP-Logins“ untersucht. In diesem Test konnte gezeigt werden, dass der Einsatz von Verschlüsselungstechniken die **Anzahl** der durchführbaren **IMAP-Logins reduziert**. Die Stärke der Reduzierung ist hierbei abhängig vom eingesetzten Plugin. Bei der Verbindung mit einem Client wird die Login-Anzahl für MailCrypt um ca. 4%, für TREES um ca. 70% und für Scrambler um ca. 85% reduziert. Mit steigender Client-Anzahl vergrößert sich die Reduzierung der Login-Anzahl. Dieser Test zeigte außerdem, dass sich durch den Einsatz von **Scrambler** und **TREES** mit **relativ einfachen Mitteln**, die Möglichkeit für eine **DOS-Attacke** bietet. Diese kann im schlimmsten Fall zum Absturz des E-Mail-Servers führen.

Weiterhin konnte gezeigt werden, dass sich die **E-Mail-Abrufdauer** durch den Einsatz von Postfachverschlüsselungstechniken **verlängert**. Die Stärke der Verzögerung ist hierbei abhängig vom Plugin und dem eingesetzten Abrufprotokoll. Hierbei ist besonders auffällig, dass beim Einsatz von MailCrypt der Abruf der E-Mails über IMAP länger dauert, als der Abruf über POP3. Ein umgekehrtes Verhalten konnte beim unverschlüsselten E-Mail-Empfang und dem Einsatz von Scrambler oder TREES beobachtet werden. Ein Abruf über IMAP ist dort schneller, als ein Abruf über POP3.

Abschließend wurde im Versuch „Einfluss auf den Nutzerkontenerstellungsprozess“ gezeigt, dass die Anzahl der **erstellbaren Nutzer pro Minute** durch den Einsatz der Postfachverschlüsselungstechniken **reduziert** wird. Dies fiel in Abhängigkeit der eingesetzten Technik unterschiedlich stark aus. Die Reduzierung betrug für MailCrypt ca. 50%, für TREES ca. 93% und für Scrambler ca. 97%.

5. Praktische Erfahrungen

In diesem Kapitel sollen praktische Erfahrungen diskutiert werden, die sich aus der Implementierung und der Verwendung der verschiedenen Techniken ergeben. Hierbei wird näher auf Implementierungsaufwände und Schwierigkeiten in der Schlüsselverwaltung eingegangen. Zum Abschluss werden die zusätzlichen Schritte skizziert, die bei Passwortänderungs- und Passwortwiederherstellungprozessen notwendig sind.

5.1. Implementierungsaufwand

Die vorgestellten Postfachverschlüsselungstechniken unterscheiden sich deutlich in ihrem Implementierungsaufwand.

Die Implementierung von MailCrypt gestaltet sich am einfachsten, da es bereits in Dovecot integriert ist. Hierfür muss lediglich eine Konfigurationsdatei hinzugefügt werden und Dovecot zur Verwendung des Plugins angewiesen werden. Alles Weitere erfolgt automatisch.

Zur Verwendung des GPG-Sieve-Filters müssen zusätzliche Programme installiert werden. Weiterhin muss Dovecot für den Einsatz von Sieve-Filtern konfiguriert werden. Die Verwendung von Sieve-Filtern gehört zum Quasi-Standard beim E-Mail-Server-Betrieb und ist gut dokumentiert[70]. Danach muss die GPG-Sieve-Filter-Software heruntergeladen werden und Dovecot zu deren Verwendung angewiesen werden. Hierbei müssen diverse Konfigurationsdateien angepasst werden und der öffentliche GPG-Schlüssel des Nutzers im Schlüsselbund des MDA-Nutzers hinterlegt werden. Der Schlüsselerstellungsprozess wird hierbei an den Nutzer ausgelagert und setzt Kenntnisse der GPG-Software voraus. Der Schlüsselerstellungsprozess kann auch an den E-Mail-Server delegiert werden. Hierbei ist jedoch eine zusätzliche Implementierung auf Seiten des Server-Betreibers erforderlich.

Um die Plugins Scrambler und TREES einsetzen zu können, muss ein deutlich höherer Aufwand betrieben werden. Hierbei müssen für jedes Plugin diverse Pakete installiert werden, bevor die Software heruntergeladen und erfolgreich kompiliert werden kann. Einige Pakete müssen hierbei in definierten Versionen installiert werden. Die resultierenden Dovecot-Module müssen im entsprechenden Verzeichnis abgelegt werden. Erst danach kann Dovecot zu deren Verwendung konfiguriert werden. Zum Betrieb der Plugins sind zusätzliche Daten notwendig, welche durch externe Skripte generiert werden müssen.

[70] `https://wiki.dovecot.org/Pigeonhole/Sieve`

Die Entwickler des TREES-Plugins liefern zu ihrer Software ein Ruby-Skript[71] mit. Um dieses verwenden zu können, müssen zusätzliche Debian- und Ruby-Pakete installiert werden. Die Ruby-Pakete müssen in einer bestimmten Version installiert werden, damit das Skript funktioniert.

Die Entwickler des Scrambler-Plugins verzichten auf die Bereitstellung eines vergleichbaren Skripts. Die Funktionstests im Repository liefern jedoch genügend Hinweise für den Nachbau eines Skripts, mit dem die notwendigen Daten generiert werden können. Zu dessen Verwendung sind ebenfalls zusätzliche Debian-Pakete notwendig.

Nach der initialen Implementierung und Plugin-Aktivierung sollte der E-Mail-Server-Betreiber sicherstellen, dass auch der Inhalt von existierenden Postfächern verschlüsselt wird. Die aktivierten Techniken sorgen lediglich dafür, dass zukünftig eingehende E-Mails verschlüsselt werden. Bereits abgespeicherte E-Mails werden nicht nachträglich verschlüsselt. Eine Möglichkeit wäre es einen bestehenden E-Mail-Account mit sich selbst zu synchronisieren. Dafür kann das Kommando *doveadm sync*[72] verwendet werden.

5.2. Schlüsselverwaltung

Beim Einsatz der Postfachverschlüsselungstechniken ist besondere Vorsicht bei der Aufbewahrung und Wartung der verschlüsselten Nutzerschlüssel geboten. Beim Verlust des Passworts oder des privaten Schlüssels wäre das Entschlüsseln der E-Mails nicht mehr möglich und würde zu dauerhaftem Datenverlust führen. Zur Vermeidung dieses Zustands sollte ein E-Mail-Server-Betreiber besondere Vorsichtsmaßnahmen treffen. Denkbar wäre zum Beispiel ein Passwortwiederherstellungsprozess mit Hilfe von Wiederherstellungscodes, mit dessen Hilfe der private Schlüssel wiederhergestellt werden kann.

Weiterhin muss ein Betreiber Prozesse implementieren, die dem Nutzer die Änderung seines Passworts erlauben, ohne hierbei den Zugriff auf den privaten Schlüssel zu verlieren. Ausschließlich beim MailCrypt-Plugin werden Routinen mitgeliefert, die relativ simpel in einen Passwortänderungsprozess integriert werden können. Diese sind in der Plugin-Dokumentation[73] beschrieben. Bei allen anderen Techniken muss der E-Mail-Server-Betreiber eine eigene Implementierung entwickeln.

Abschließend wird skizziert, wie ein Passwortänderungs- und ein Passwortwiederherstellungsprozess aussehen könnte. Für beide Prozesse ist es notwendig, dass der E-Mail-

[71] `https://www.ruby-lang.org/en/`
[72] `https://wiki.dovecot.org/Tools/Doveadm/Sync`
[73] `https://wiki.dovecot.org/Plugins/MailCrypt`

Server-Betreiber über ein Nutzerportal verfügt, in welchem der Nutzer kontenspezifische Einstellungen vornehmen kann.

5.2.1. Passwortänderung

Bei einer Passwortänderung muss der private Schlüssel des Nutzers mit Hilfe eines neuen Passworts neu verschlüsselt werden. Der Nutzer muss sich hierzu in das Nutzerportal einloggen und den Passwortänderungsprozess auslösen. Diesem übergibt er sein bisheriges sowie sein neues Passwort. Der ausgelöste Prozess entschlüsselt nun den privaten Schlüssel des Nutzers mit dem bisherigen Passwort und verschlüsselt ihn anschließend wieder mit dem neuen Passwort. Mit der Speicherung des Passworts und der Speicherung des verschlüsselten Schlüssels ist der Prozess abgeschlossen.

5.2.2. Passwortwiederherstellung

Damit ein Nutzer Zugriff auf seine Daten wiedererlangen kann, wenn er nicht mehr im Besitz seines Passworts ist, ist ein Passwortwiederherstellungsprozess notwendig. Dieser könnte wie folgt aussehen.

Bei der Registrierung über das Nutzerportal wird dem Nutzer ein Wiederherstellungsschlüssel generiert und einmalig angezeigt. Als Wiederherstellungsschlüssel wäre eine zufällige, durch den Server generierte, Zeichenkette denkbar. Diese wird lediglich dem Nutzer angezeigt, aber nicht auf dem Server abgespeichert. Der Nutzer wird im Anschluss aufgefordert, den Wiederherstellungscode an einem „sicheren“ Ort aufzubewahren. Mit dem Wiederherstellungscode wird eine Kopie des privaten Nutzerschlüssels verschlüsselt und in der Datenbank abgelegt.

Hat der Nutzer sein Passwort vergessen, kann er den Wiederherstellungsprozess über das Nutzerportal auslösen und den Wiederherstellungscode eingeben. Mit dem Wiederherstellungscode wird die Kopie seines privaten Schlüssels entschlüsselt. Daraufhin muss der Nutzer ein neues Passwort eingeben. Mit dem neuen Passwort wird der private Schlüssel neu verschlüsselt. Der nun neu verschlüsselte Schlüssel ersetzt den verschlüsselten Schlüssel, zu dem der Nutzer das Passwort vergessen hat.

Abschließend wird dem Nutzer ein neuer Wiederherstellungscode angezeigt, mit dem wiederum eine verschlüsselte Kopie seines privaten Schlüssels auf dem Server abgespeichert wird.

Ein Verlust des Passworts und des Wiederherstellungscode hat einen vollständigen Datenverlust zur Folge.

6. Fazit und Ausblick

6.1. Fazit

Ziel dieser Masterarbeit war es, herauszufinden, ob der Einsatz von Postfachverschlüsselungstechniken Auswirkungen auf den E-Mail-Server-Betrieb hat. Weiterhin sollte untersucht werden, ob eventuelle Auswirkungen abhängig von der gewählten Postfachverschlüsselungstechnik sind.

Im Ergebnis stellt sich dar, dass der Einsatz von Postfachverschlüsselungstechniken zu Performance-Einbußen auf dem E-Mail-Server führt. Diese fallen in Abhängigkeit der eingesetzten Technik unterschiedlich stark aus. Eine detaillierte Auswertung ist im Kapitel „Untersuchung der Postfachverschlüsselungstechniken“ in Abschnitt 4.5 zu finden. Der Einsatz der persönlichen E-Mail-Postfachverschlüsselung benötigt bei gleichem E-Mail-Durchsatz und Nutzeraufkommen zusätzliche Hardware-Ressourcen. Im Verhältnis zur zusätzlich gewonnenen Sicherheit bewegen sich die Zusatzaufwände aber in einem vertretbaren Rahmen.

Die größte Hürde für den breiten Einsatz der beschriebenen Techniken stellt der Implementierungsaufwand für den E-Mail-Server-Betreiber dar. Da nicht alle untersuchten Techniken, Skripte zur Nutzererstellung, Passwortänderung oder Passwortwiederherstellung mitliefern, ist er gezwungen individuelle Lösungen für diese Prozesse zu entwickeln, bevor er die Techniken produktiv einsetzen kann. Von den Entwicklern mitgelieferte Skripte könnten diesen Aufwand minimieren und damit die Akzeptanz der Techniken womöglich unterstützen.

Weiterhin konnte gezeigt werden, dass durch den Einsatz der Techniken der Zugang für berechtigte und unberechtigte Dritte erschwert wird. Dass das Klartext-Passwort des Nutzers zur Entschlüsselung seines privaten Schlüssels verwendet wird, mit dem schlussendlich seine E-Mails entschlüsselt werden, stellt sich als größte Stärke und gleichzeitig als größte Schwäche der Verfahren heraus.

Einerseits wird so verhindert dass der Nutzer zusätzliche Software installieren muss, um diese Technik zu verwenden. Andererseits kann das Klartext-Passwort auch weiterhin Ziel eines aktiven Angriffs werden - es kann beim Nutzer-Login abfangen, oder aus einer entwendeten Nutzerdatenbank berechnet werden. Deswegen sollte beim Einsatz der Techniken eine anspruchsvolle Hash-Funktion zur Erzeugung der Passwort-Hashes auf Seiten des Servers eingesetzt werden. Da die Postfachverschlüsselung letztendlich serverseitig erfolgt, ist ein vollkommener Schutz vor aktiven Angriffen hiermit nicht möglich.

Auch passive Angriffe können durch den Einsatz dieser Technik nicht vollständig verhindert werden. Ein passiver Angreifer kann den E-Mail-Server-Betreiber weiterhin zu einer Postfachüberwachung verpflichten. Hierbei werden die E-Mails vor der Verschlüsselung ausgeleitet. Dementsprechend bietet der Einsatz der Technik keinen Schutz vor dieser Art Angriff. Bei einer Postfachbeschlagnahmung hingegen, kann der passive Angreifer lediglich den verschlüsselten Inhalt des Postfachs erhalten.

Im Ergebnis dieser Arbeit ist festzustellen, dass durch den Einsatz der persönlichen E-Mail-Postfachverschlüsselung der Aufwand für beide Angreifertypen deutlich erhöht wird.

Nimmt ein E-Mail-Server-Betreiber die notwendigen Aufwände zum Einsatz der Technik in Kauf, ist diese eine überzeugende Schutzmaßnahme für die E-Mail-Inhalte seiner Nutzer. Obwohl die Technik gegenüber Ende-zu-Ende-Verschlüsselung Defizite aufweist, erscheint ihr Einsatz als sinnvoll, solange Ende-zu-Ende-Verschlüsselung nicht flächendeckend eingesetzt wird. Bis dahin sollte der Einsatz der persönlichen E-Mail-Postfachverschlüsselung als zusätzliche Schutzmaßnahme in Betracht gezogen werden.

6.2. Ausblick

Im Rahmen der Untersuchung ist aufgefallen, dass Dovecot einige Daten, wie z.B. Header und Subject der E-Mail, in eigenen Index-Dateien vorhält[74]. Diese sind hierbei unverschlüsselt auf dem E-Mail-Server abgespeichert. Weiterhin konnte beobachtet werden, dass auch die Ordner-Namen des E-Mail-Postfachs unverschlüsselt abgespeichert sind. In zukünftigen Untersuchungen sollte herausgefunden werden, welche Daten und Meta-Daten unverschlüsselt gespeichert bzw. kurzfristig unverschlüsselt vorgehalten werden. Optimal wäre es, wenn keine dieser Daten gespeichert werden würden. Deshalb sollten diese nach Möglichkeit auf das Minimum begrenzt werden.

Im Rahmen des Tests „Einfluss auf die Anzahl der IMAP-Logins“ konnte gezeigt werden, dass der Einsatz des Scrambler- und TREES-Plugins eine einfach Möglichkeit für eine DOS-Attacke bietet. In zukünftigen Weiterentwicklungen dieser Technologien sollte dieser Angriffsvektor minimiert werden. Denkbar wäre ein Programm, das ausschließlich für die Entschlüsselung des privaten Nutzerschlüssels zuständig ist. Dieses könnte von Dovecot angesprochen werden, das Nutzerpasswort erhalten und den entschlüsselten Nutzerschlüssel zurückliefern. Das Programm sollte nicht direkt in Dovecot integriert sein, um den vollständigen Absturz des E-Mail-Servers im Falle einer Überlastung zu verhindern.

[74] `https://wiki.Dovecot.org/IndexFiles`

Weiterhin ist zu bemerken, dass in neueren Dovecot-Versionen (>=2.3.0) die Hash-Algorithmen Argon2i und Argon2i/d integriert wurden [Tuomi and Sirainen, 2018]. Damit ist eine Verwendung einer MHF in Kombination mit MailCrypt relativ einfach möglich. Dieser Umstand könnte auch für die weitere Entwicklung von Scrambler und TREES interessant sein. Da beide Plugins bereits eine ressourcen-intensive Hash-Funktion einsetzen, ist eine Kombination mit einer weiteren, aufwendigen Hash-Funktion nicht ungefährlich und würde zu einer deutlich höheren Server-Belastung führen. In diesem Falle sollten E-Mail-Server-Betreiber zusätzliche Maßnahmen ergreifen, um sich vor DOS-Attacken zu schützen.

Schlussendlich bleibt abzuwarten, ob Techniken dieser Art in Zukunft weiter bestand haben. Denn in der Vergangenheit ist es immer wieder vorgekommen, dass Unternehmen oder Nutzer zur Herausgabe von Passwörtern gezwungen wurden [Hern, 2017] [Gierow, 2016]. Damit ist die Schutzfunktion der untersuchten Postfachverschlüsselungstechniken vollständig ausgehebelt.

Abkürzungsverzeichnis

AAD Additional Authenticated Data
AEAD Authenticated Encryption with Associated Data
AES Advanced Encryption Standard
C2S Client-to-Server
CPU Central Processing Unit
DOS Denial-of-Service
ECDH Elliptic Curve Diffie-Hellman
ECIES Elliptic Curve Integrated Encryption Scheme
GCM Galois/Counter-Mode
HMAC Keyed-Hash Message Authentication Code
IMAP Internet Message Access Protocol
IV Initialization Vector
MDA Message Delivery Agent
MHF Memory-hard Hash-Function
MRA Message Retrieval Agent
MSA Message Submission Agent
MS Message Store
MTA Message Transfer Agent
MUA Message User Agent
POP3 Post Office Protocol - Version 3
RAM Random-Access Memory
RSA Rivest-Shamir-Adleman
S2S Server-to-Server
SMTP Simple Mail Transfer Protocol
SSH Secure Shell
SSL Secure Sockets Layer
TLS Transport Layer Security
VM virtuelle Maschine
IETF Internet Engineering Task Force

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[Biselli, 2018] Biselli, A. (2018). Alice und Bob legen sich in die Kurve. https://www.golem.de/news/elliptische-kurven-alice-und-bob-legen-sich-in-die-kurve-1808-135105-4.html. Online; Zugriff am: 04.02.2019.

[Blocki, 2017] Blocki, J. (2017). Memory Hard Functions and Password Hashing. https://www.youtube.com/watch?v=8QxFsWszbyI. Online; Zugriff am: 26.12.2018.

[Brüll, 2015] Brüll, P. (2015). Dovecot encryption plugin. https://github.com/posteo/scrambler-plugin/blob/master/README.md. Online; Zugriff am: 23.10.2018.

[BSI, 2009] BSI, B. f. S. i. d. I. (2009). Prometheus für das Cloud- und Enterprise-Monitoring. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Internetsicherheit/isi_mail_server_studie_pdf.pdf;jsessionid=4D2F376DA1CD4D002C6A1EA69C4D6C26.2_cid341?__blob=publicationFile&v=1. Online; Zugriff am: 01.11.2018.

[Cox, 2018] Cox, J. (2018). Facebook Fires Employee Who Allegedly Used Data Access to Stalk Women. https://motherboard.vice.com/en_us/article/bjpqw4/facebook-fires-employee-stalk-women-online. Online; Zugriff am: 03.11.2018.

[Crispin, 2003] Crispin, M. (2003). Internet message access protocol - version 4rev1. RFC 3501, IETF. https://tools.ietf.org/pdf/rfc3501.pdf.

[Crocker, 2009] Crocker, D. (2009). Internet Mail Architecture. RFC 5598, IETF.

[dovecot.org, 2017] dovecot.org (2017). Design/Dcrypt. https://wiki.dovecot.org/Design/Dcrypt. Online; Zugriff am: 11.10.2018.

[dovecot.org, 2018] dovecot.org (2018). Secure IMAP server. https://www.dovecot.org/. Online; Zugriff am: 01.11.2018.

[esmtp.email, 2017] esmtp.email (2017). Benchmarking smtp servers with postal. https://esmtp.email/blog/2017/11/04/postal-benchmark/. Online; Zugriff am: 14.12.2018.

[Foster et al., 2015] Foster, I., Larson, J., Masich, M., Snoeren, A. C., Savage, S., and Levchenko, K. (2015). *Security by Any Other Name:On the Effectiveness of Provider Based Email Security*. CCS '15 Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. https://cseweb.ucsd.edu/~klevchen/flmssl-ccs15.pdf; Online; Zugriff am: 22.01.2019.

[Gellens and Klensin, 2011] Gellens, R. and Klensin, J. (2011). Message submission for mail. STD 72, IETF. https://tools.ietf.org/pdf/rfc6409.pdf.

[Gierow, 2016] Gierow, H. (2016). Apple will in seinem code nicht lügen. https://www.golem.de/news/streit-mit-fbi-apple-will-in-seinem-code-nicht-luegen-1602-119406.html. Online; Zugriff am: 17.01.2019.

[Google, 2018] Google (2018). Email encryption in transit. https://transparencyreport.google.com/safer-email/overview. Online; Zugriff am: 16.12.2018.

[Gosney, 2016] Gosney, J. M. (2016). How LinkedIn's password sloppiness hurts us all. https://arstechnica.com/information-technology/2016/06/how-linkedins-password-sloppiness-hurts-us-all/. Online; Zugriff am: 03.12.2018.

[Gray, 2018] Gray, J. (2018). Learn the average size of an email message. https://www.lifewire.com/what-is-the-average-size-of-an-email-message-1171208. Online; Zugriff am: 14.12.2018.

[Guenther and Showalter, 2008] Guenther, P. and Showalter, T. (2008). Sieve: An email filtering language. RFC 5228. https://tools.ietf.org/pdf/rfc5228.pdf.

[Gueron, 2012] Gueron, S. (2012). Intel® Advanced Encryption Standard (AES) New Instructions Set. https://software.intel.com/sites/default/files/article/165683/aes-wp-2012-09-22-v01.pdf. Online; Zugriff am: 23.11.2018.

[Heiderich et al., 2016] Heiderich, M., Wege, M., Weißer, D., Magazinius, J., and Krein, N. (2016). Pentest-Report Dovecot 11.2016. https://cure53.de/pentest-report_dovecot.pdf/. Online; Zugriff am: 01.11.2018.

[Heinlein, 2008] Heinlein, P. (2008). *Das Postfix-Buch*. Open Source Press.

[Heinlein, 2012] Heinlein, P. (2012). Mails im mdbox-Format - Mehr Speed mit Dovecot. https://www.heinlein-support.de/sites/default/files/Dovecot-mdbox-Mailperformance_0.pdf. Online; Zugriff am: 14.12.2018.

[Heinlein, 2014] Heinlein, P. (2014). *Dovecot: POP3/IMAP-Server für Unternehmen und ISPs*. Open Source Press.

[Hern, 2017] Hern, A. (2017). Uk tourists to us may get asked to hand in passwords or be denied entry. https://www.theguardian.com/us-news/2017/apr/09/uk-tourists-to-us-may-get-asked-to-hand-in-passwords-or-be-denied-entry. Online; Zugriff am: 17.01.2019.

[Hunt, 2019] Hunt, T. (2019). The 773 Million Record "Collection 1"Data Breach. https://www.troyhunt.com/the-773-million-record-collection-1-data-reach. Online; Zugriff am: 12.02.2019.

[Klensin, 2008] Klensin, J. (2008). Simple mail transfer protocol. RFC 5321, IETF. https://tools.ietf.org/pdf/rfc5321.pdf.

[Lathan, 2018] Lathan, P. (2018). RAM Price Report: DDR4 Same Price as Initial Launch. https://www.gamersnexus.net/industry/3212-ram-price-investigation-ddr4-same-price-as-initial-launch. Online; Zugriff am: 25.01.2019.

[Lawler, 2017] Lawler, R. (2017). Yahoo's 2013 hack impacted all 3 billion accounts. https://www.engadget.com/2017/10/03/yahoo-2013-hack-three-billion/?guccounter=2. Online; Zugriff am: 03.12.2018.

[Levison, 2014] Levison, L. (2014). Secrets, lies and Snowden's email: why I was forced to shut down Lavabit. https://www.theguardian.com/commentisfree/2014/may/20/why-did-lavabit-shut-down-snowden-email. Online; Zugriff am: 27.12.2018.

[libsodium.org, o.J.] libsodium.org (o.J.). Sealed boxes. https://libsodium.gitbook.io/doc/public-key_cryptography/sealed_boxes. Online; Zugriff am: 23.10.2018.

[Lord, 2016] Lord, B. (2016). Important Security Information for Yahoo Users. https://yahoo.tumblr.com/post/154479236569/important-security-information-for-yahoo-users. Online; Zugriff am: 03.12.2018.

[Loschwitz, 2016] Loschwitz, M. (2016). Prometheus für das Cloud- und Enterprise-Monitoring. http://www.linux-magazin.de/ausgaben/2016/03/prometheus/. Online; Zugriff am: 01.11.2018.

[Lyman et al., 2003] Lyman, P., Varian, H. R., Charles, P., Good, N., Jordan, L. L., and Pal, J. (2003). How much information? 2003. http://groups.ischool.berkeley.edu/archive/how-much-info-2003/printable_execsum.pdf. Online; Zugriff am: 14.12.2018.

[mailbox.org, 2019] mailbox.org (2019). mailbox.org Transparenzbericht 2018: Mehr Anfragen, über 60rechtswidrig. https://mailbox.org/de/pressemitteilung-transparenzbericht-2018. Online; Zugriff am: 24.01.2019.

[mailbox.org, o.J.] mailbox.org (o.J.). Das verschlüsselte E-Mail-Postfach. https://kb.mailbox.org/display/BMBOKB/Das+verschluesselte+Postfach. Online; Zugriff am: 23.10.2018.

[McCarthy, 2018] McCarthy, K. (2018). Mailconcept. https://gitlab.com/muttmua/mutt/wikis/MailConcept. Online; Zugriff am: 23.11.2018.

[Moore and Newman, 2018] Moore, K. and Newman, C. (2018). Cleartext considered obsolete: Use of transport layer security (tls) for email submission and access. RFC 8314, IETF.

[mozilla.org, 2019] mozilla.org (2019). Statistics for Enigmail. https://addons.thunderbird.net/en-US/thunderbird/addon/enigmail/statistics/?last=90. Online; Zugriff am: 21.01.2019.

[Myers and Rose, 1996] Myers, J. G. and Rose, M. T. (1996). Post office protocol - version 3. STD 53, IETF. https://tools.ietf.org/pdf/rfc1939.pdf.

[openemailsurvey.org, 2018] openemailsurvey.org (2018). Open email survey (March 2018). http://www.openemailsurvey.org/. Online; Zugriff am: 01.11.2018.

[password hashing.net, 2017] password hashing.net (2017). Argon2: the memory-hard function for password hashing and other applications. https://github.com/P-H-C/phc-winner-argon2/blob/master/argon2-specs.pdf. Online; Zugriff am: 26.12.2018.

[passwordhashing.net, 2015] passwordhashing.net (2015). Password Hashing Competition and our recommendation for hashing passwords: Argon2. https://password-hashing.net/. Online; Zugriff am: 26.12.2018.

[Percival, 2009] Percival, C. (2009). Stronger key derivation via sequential memory-hard functions. https://www.daemonology.net/papers/scrypt.pdf. Online; Zugriff am: 26.12.2018.

[Peterson, 2013] Peterson, A. (2013). LOVEINT: When NSA officers use their spying power on love interests. https://www.washingtonpost.com/news/the-switch/wp/2013/08/24/loveint-when-nsa-officers-use-their-spying-power-on-love-interests/??noredirect=on. Online; Zugriff am: 03.11.2018.

[posteo.de, 2015] posteo.de (2015). Neu: Posteo führt Krypto-Mailspeicher ein. https://posteo.de/blog/neu-posteo-f%C3%BChrt-krypto-mailspeicher-ein. Online; Zugriff am: 23.10.2018.

[posteo.de, 2018] posteo.de (2018). Posteo Transparenzbericht. https://posteo.de/site/transparenzbericht. Online; Zugriff am: 24.01.2019.

[posteo.de, o.J.] posteo.de (o.J.). Posteo-Eingangsverschlüsselung. https://posteo.de/site/verschluesselung#zugriff. Online; Zugriff am: 23.10.2018.

[Rescorla, 2018] Rescorla, E. (2018). The transport layer security (tls) protocol version 1.3. RFC 8446, IETF. https://tools.ietf.org/pdf/rfc8446.pdf.

[Resnick, 2008] Resnick, P. W. (2008). Internet message format. RFC 5322, IETF. https://tools.ietf.org/pdf/rfc5322.pdf.

[riseup.net, 2017] riseup.net (2017). Riseup moves to encrypted email in response to legal requests. https://riseup.net/en/about-us/press/canary-statement. Online; Zugriff am: 23.10.2018.

[Ruoti et al., 2019] Ruoti, S., Andersen, J., Zappala, D., and Seamons, K. (2019). Why Johnny Still, Still Can't Encrypt: Evaluating the Usability of a Modern PGP Client. https://arxiv.org/abs/1510.08555. Online; Zugriff am: 21.01.2019.

[securityspace.com, 2019] securityspace.com (2019). Mail (MX) Server Survey. http://www.securityspace.com/s_survey/data/man.201812/mxsurvey.html. Online; Zugriff am: 19.01.2019.

[Selyukh, 2013] Selyukh, A. (2013). NSA staff used spy tools on spouses, ex-lovers: watchdog. https://www.reuters.com/article/us-usa-surveillance-watchdog/nsa-staff-used-spy-tools-on-spouses-ex-lovers-watchdog-idUSBRE98Q14G20130927. Online; Zugriff am: 03.11.2018.

[Siemborski and Melnikov, 2007] Siemborski, R. and Melnikov, A. (2007). Smtp service extension for authentication. RFC 4954, IETF. https://tools.ietf.org/pdf/rfc4954.pdf.

[Sinha et al., 2013] Sinha, R., Srivastava, H. K., and Gupta, S. (2013). Performance Based Comparison Study of RSA and Elliptic Curve Cryptography. https://pdfs.semanticscholar.org/47c5/7bf7b1fce8600bf74bfb6825b2f707b8953e.pdf. Online; Zugriff am: 11.02.2019.

[Sirainen and Andree, 2006] Sirainen, T. and Andree, M. (2006). Mailbox Formats. https://wiki2.dovecot.org/MailboxFormat. Online; Zugriff am: 27.12.2018.

[Toponce, 2016] Toponce, A. (2016). Let's Talk Password Hashing. https://pthree.org/2016/06/28/lets-talk-password-hashing. Online; Zugriff am: 11.02.2019.

[Tuomi and Sirainen, 2018] Tuomi, A. and Sirainen, T. (2018). Password Schemes. https://wiki.dovecot.org/Authentication/PasswordSchemes. Online; Zugriff am: 17.01.2019.

[Tuomi et al., 2016] Tuomi, A., Sirainen, T., and MarttiRannanjarvi (2016). Dovecot encryption plugin. https://wiki2.dovecot.org/Plugins/MailCrypt. Online; Zugriff am: 23.10.2018.

A. Anhang

Listing 14: Ciphertext einer verschlüsselt abgespeicherten E-Mail

```
[09:41:12] root@trees:~ $ less /var/vmail/trees.test/trees/Maildir/mailboxes/INBOX/dbox-Mails/u.1
2 M1e C5be4045f
^A^BN 00000000000001EA
<EE><FF><CC>^@^@^@^A<88><88><D2><C3>c<C4>^X^G<A2><B7><C9><E7><DD>'<D5>$<
    F3>m<8B>N^C,<A0> <F9><94>e<BD><CB>!<E3>=<D4><FC>d<E9>D<A3> <B1>h8y<B2><
    AC>:a<FE><83><DF>a<A1>
d<86>^H<90><A7><D0>^Tqs<A3>m.<9A><AD><F3><91>Q<83>I<BB><84><8D><EC><88><DA
    >{<C7>2=^?<D9>o<9A>^L<D6>^N<B7>^Hv<90>5~<85><E8><EB>^_<E4><D5>(<BE><C1
    ><86>#<E5><AA><E3>E(f^D<9A><C1>L<E4>%<E7>^D<AB><F3><87>^C<E3><8D>^Z<8C
    ><FD>66r<AE><A4><B6><C0>^D<CA>^W<EE>"&<8A>U<E6>^S<B5><8C>I
<8A>T:<CD>^NMw^M^D[<AD>_<C8>g<AF><FF>D<E3>) <93>uDv^Ze6<92>'>D<B7><FE><E4>
    M<92><C6><FA>sUh\<96><DB>%P*<E7><F8><AC><D5>^Vs}<C3>B^M^G
<F7><D0><EC><A9><F1>a^Y<A4><A6>^T<E2>e<FF> ^A^C
Z1b3
R5be4045f
V1bf
G434bdd0e5f04e45b5
```

Listing 15: Dovecot Fehlermeldung beim Absturz des POP3-Dienstes beim gleichzeitigen Empfangen und Abruf von E-Mails via POP3 (Scrambler)

```
Dec 13 18:58:10 scrambler dovecot: lda(scrambler@scrambler.test): scrambler plugin initialized
Dec 13 18:58:10 scrambler dovecot: pop3(scrambler@scrambler.test): Error: Maildir filename has wrong
    S value, renamed the file from /var/vmail/scrambler.test/scrambler/Maildir/cur/1544727316.
    M615652P22865.scrambler,S=817:2, to /var/vmail/scrambler.test/scrambler/Maildir/cur
    /1544727316.M615652P22865.scrambler,S=817:2,
Dec 13 18:58:10 scrambler dovecot: pop3(scrambler@scrambler.test): Error: Corrupted record in index
    cache file /var/vmail/scrambler.test/scrambler/Maildir/dovecot.index.cache: UID 497: Broken
    physical size in mailbox INBOX: read(/var/vmail/scrambler.test/scrambler/Maildir/cur/1544727316.
    M615652P22865.scrambler,S=817:2,) failed: Cached message size larger than expected (817 > 475,
    box=INBOX, UID=497)
Dec 13 18:58:10 scrambler dovecot: pop3(scrambler@scrambler.test): Panic: file istream.c: line 175 (
    i_stream_read): assertion failed: (old_size <= _stream->pos - _stream->skip)
Dec 13 18:58:10 scrambler dovecot: pop3(scrambler@scrambler.test): Error: Raw backtrace: /usr/lib/
    dovecot/libdovecot.so.0(+0x9e412) [0x7fa05d39c412] -> /usr/lib/dovecot/libdovecot.so.0(+0
    x9e50d) [0x7fa05d39c50d] -> /usr/lib/dovecot/libdovecot.so.0(i_fatal+0) [0x7fa05d32bc51] -> /
    usr/lib/dovecot/libdovecot.so.0(i_stream_read+0x290) [0x7fa05d3a8050] -> /usr/lib/dovecot/
    libdovecot.so.0(i_stream_read_data+0x3d) [0x7fa05d3a885d] -> /usr/lib/dovecot/libdovecot.so.0(
    message_get_body_size+0xfd) [0x7fa05d3891dd] -> /usr/lib/dovecot/libdovecot-storage.so.0(
    index_mail_init_stream+0x248) [0x7fa05d6e90a8] -> /usr/lib/dovecot/libdovecot-storage.so.0(+0
    x6e0f0) [0x7fa05d69f0f0] -> /usr/lib/dovecot/libdovecot-storage.so.0(mail_get_stream_because+0
    x5d) [0x7fa05d66ae6d] -> /usr/lib/dovecot/libdovecot-storage.so.0(+0x6deb7) [0x7fa05d69eeb7]
    -> /usr/lib/dovecot/libdovecot-storage.so.0(mail_get_virtual_size+0x33) [0x7fa05d66aad3] ->
    dovecot-scrambler.test/pop3(client_init_mailbox+0x3d3) [0x55f8d9a91753] -> dovecot-
    scrambler.test/pop3(+0x4d85) [0x55f8d9a8fd85] -> dovecot-scrambler.test/pop3(+0x5038) [0
    x55f8d9a90038] -> /usr/lib/dovecot/libdovecot.so.0(+0x35d2e) [0x7fa05d333d2e] -> /usr/lib/
    dovecot/libdovecot.so.0(+0x36013) [0x7fa05d334013] -> /usr/lib/dovecot/libdovecot.so.0(+0
    x369cf) [0x7fa05d3349cf] -> /usr/lib/dovecot/libdovecot.so.0(io_loop_call_io+0x52) [0
    x7fa05d3b2812] -> /usr/lib/dovecot/libdovecot.so.0(io_loop_handler_run_internal+0x109) [0
    x7fa05d3b3eb9] -> /usr/lib/dovecot/libdovecot.so.0(io_loop_handler_run+0x3c) [0x7fa05d3b28ac]
    -> /usr/lib/dovecot/libdovecot.so.0(io_loop_run+0x38) [0x7fa05d3b2a58] -> /usr/lib/dovecot/
    libdovecot.so.0(master_service_run+0x13) [0x7fa05d336593] -> dovecot-scrambler.test/pop3(
```

```
    main+0x2aa) [0x55f8d9a8f7fa] −> /lib/x86_64−linux−gnu/libc.so.6(__libc_start_main+0xf1) [0
    x7fa05cf7f2e1] −> dovecot−scrambler.test/pop3(_start+0x2a) [0x55f8d9a8f97a]
Dec 13 18:58:10 scrambler dovecot: pop3(scrambler@scrambler.test): Fatal: master: service(pop3): child
    28465 killed with signal 6 (core dumps disabled)
```

Listing 16: Dovecot Fehlermeldung beim Absturz des POP3-Dienstes beim gleichzeitigen Empfangen und Abruf von E-Mails via POP3 (TREES)

```
Dec 13 19:01:07 trees dovecot: pop3(trees@trees.test): Debug: trees plugin initialized
Dec 13 19:01:08 trees dovecot: pop3(trees@trees.test): Error: Maildir filename has wrong S value,
    renamed the file from /var/vmail/trees.test/trees/Maildir/cur/1544727610.M760781P7535.trees,S
    =490:2, to /var/vmail/trees.test/trees/Maildir/cur/1544727610.M760781P7535.trees,S=490:2,
Dec 13 19:01:08 trees dovecot: pop3(trees@trees.test): Error: Corrupted record in index cache file /var/
    vmail/trees.test/trees/Maildir/dovecot.index.cache: UID 340: Broken physical size in mailbox
    INBOX: read(/var/vmail/trees.test/trees/Maildir/cur/1544727610.M760781P7535.trees,S=490:2,)
    failed: Cached message size larger than expected (490 > 435, box=INBOX, UID=340)
Dec 13 19:01:08 trees dovecot: pop3(trees@trees.test): Panic: file istream.c: line 175 (i_stream_read):
    assertion failed: (old_size <= _stream−>pos − _stream−>skip)
Dec 13 19:01:08 trees dovecot: pop3(trees@trees.test): Error: Raw backtrace: /usr/lib/dovecot/libdovecot.
    so.0(+0x9e412) [0x7fcf3a23d412] −> /usr/lib/dovecot/libdovecot.so.0(+0x9e50d) [0x7fcf3a23d50d]
    −> /usr/lib/dovecot/libdovecot.so.0(i_fatal+0) [0x7fcf3a1ccc51] −> /usr/lib/dovecot/libdovecot.so
    .0(i_stream_read+0x290) [0x7fcf3a249050] −> /usr/lib/dovecot/libdovecot.so.0(i_stream_read_data
    +0x3d) [0x7fcf3a24985d] −> /usr/lib/dovecot/libdovecot.so.0(message_get_body_size+0xfd) [0
    x7fcf3a22a1dd] −> /usr/lib/dovecot/libdovecot−storage.so.0(index_mail_init_stream+0x248) [0
    x7fcf3a58a0a8] −> /usr/lib/dovecot/libdovecot−storage.so.0(+0x6e0f0) [0x7fcf3a5400f0] −> /usr/
    lib/dovecot/libdovecot−storage.so.0(mail_get_stream_because+0x5d) [0x7fcf3a50be6d] −> /usr/lib/
    dovecot/libdovecot−storage.so.0(+0x6deb7) [0x7fcf3a53feb7] −> /usr/lib/dovecot/libdovecot−
    storage.so.0(mail_get_virtual_size+0x33) [0x7fcf3a50bad3] −> dovecot−trees.test/pop3(
    client_init_mailbox+0x3d3) [0x55ff96b45753] −> dovecot−trees.test/pop3(+0x4d85) [0
    x55ff96b43d85] −> dovecot−trees.test/pop3(+0x5038) [0x55ff96b44038] −> /usr/lib/dovecot/
    libdovecot.so.0(+0x35d2e) [0x7fcf3a1d4d2e] −> /usr/lib/dovecot/libdovecot.so.0(+0x36013) [0
    x7fcf3a1d5013] −> /usr/lib/dovecot/libdovecot.so.0(+0x369cf) [0x7fcf3a1d59cf] −> /usr/lib/
    dovecot/libdovecot.so.0(io_loop_call_io+0x52) [0x7fcf3a253812] −> /usr/lib/dovecot/libdovecot.so
    .0(io_loop_handler_run_internal+0x109) [0x7fcf3a254eb9] −> /usr/lib/dovecot/libdovecot.so.0(
    io_loop_handler_run+0x3c) [0x7fcf3a2538ac] −> /usr/lib/dovecot/libdovecot.so.0(io_loop_run+0
    x38) [0x7fcf3a253a58] −> /usr/lib/dovecot/libdovecot.so.0(master_service_run+0x13) [0
    x7fcf3a1d7593] −> dovecot−trees.test/pop3(main+0x2aa) [0x55ff96b437fa] −> /lib/x86_64−linux
    −gnu/libc.so.6(__libc_start_main+0xf1) [0x7fcf39e202e1] −> dovecot−trees.test/pop3(_start+0x2a
    ) [0x55ff96b4397a]
Dec 13 19:01:08 trees dovecot: pop3(trees@trees.test): Fatal: master: service(pop3): child 9766 killed
    with signal 6 (core dumps disabled)
```