UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

TRANSAÇÕES DE EMAIL SEGURAS COM DESCOBERTA AUTOMÁTICA DE CHAVES CRIPTOGRÁFICAS PÚBLICAS

MESTRADO EM SEGURANÇA INFORMÁTICA

Jorge Manuel Rema Príncipe

Dissertação orientada por:
Prof. Doutor Mário João Barata Calha
Eng. João Manuel Malcata Alves da AnubisNetworks

2016

Agradecimentos

Um agradecimento muito grande à minha família por me ter acompanhado, apoiado e de me ter conseguido fornecer as condições necessárias para a realização de todo este percurso académico, pois sem este apoio não teria sido possível.

Um agradecimento aos meus coordenadores, o Prof. Doutor Mário Calha e o Eng. João Malcata da AnubisNetworks, por me terem acompanhado e apoiado no desenvolvimento deste projeto. Um agradecimento ao José Ferreira e ao Emanuel Alves da AnubisNetworks pela disponibilidade e pelo apoio dado durante todo o desenvolvimento deste projeto. Tiveram um papel fundamental.

Não poderia deixar de agradecer a todos os colaboradores da AnubisNetworks pela forma como me receberam e apoiaram durante todo o estágio, e pela experiência de poder concretizar este projeto num ambiente empresarial.

Por fim, um agradecimento a todos os meus colegas que fizeram parte do meu percurso académico, que proporcionaram momentos muito marcantes e positivos durante todo este percurso.

Um muito obrigado a todos!

Resumo

Nesta era posterior às revelações de *Edward Snowden*, existe uma crescente preocupação com questões de segurança e privacidade não apenas em relação a criminosos bem como a sistemas de vigilância a nível governamental. A forma mais óbvia de contrariar este tipo de invasão passa pela utilização de sistemas criptográficos que sejam resistentes a ataques do tipo "*Man-in-the-Middle*" (MitM) evitando assim quer a visualização, quer a alteração dos conteúdos. No entanto este tipo de tecnologia tem tido grande dificuldade de adoção devido, essencialmente, à sua natureza que implica a troca e manutenção de certificados e chaves públicas.

Há mais de 30 anos que o protocolo SMTP é responsável pela comunicação entre servidores de email. Quando foi desenhado, não havia preocupação com questões de segurança, e como tal, mecanismos dessa natureza não foram incorporados no protocolo. Entretanto, foram criados mecanismos adicionais para colmatar essa necessidade (STARTTLS, DKIM), mas por serem extensões opcionais, não há garantia que estas sejam utilizadas em todas as transações de email. Portanto, estas extensões melhoraram a segurança na comunicação entre servidores, mas não necessariamente entre utilizadores. Para tal foram criados mecanismos ponto-a-ponto, como o PGP e S/MIME, que dão controlo ao utilizador das propriedades de segurança na comunicação, no entanto a sua utilização é complexa.

O objetivo deste trabalho, passa pela criação de uma solução que permita adicionar automatismo e transparência a protocolos seguros ponto-a-ponto retirando ao utilizador o ónus da gestão da manutenção de chaves e de processos criptográficos.

Palavras-chave: SMTP, Email, PGP, S/MIME, DNS

Abstract

In an era subsequent to the revelations of Edward Snowden, concern has been growing with security and privacy issues not only related with criminals, but also with monitoring systems from governments. The most obvious way to counter this type of invasion involves the use of cryptographic systems that are resistant to attacks such as "Man-in-the-Middle" (MitM) avoiding either viewing or changing the contents. However, this type of technology has not been widely adopted, essentially due to the need of exchanging and maintaining certificates and public keys.

For over 30 years, the SMTP protocol is responsible for communication between mail servers. When it was designed, there were no concerns about security, and as such, the protocol had no mechanisms to provide security. In the meantime, with the rise of security issues, additional mechanisms have been created to fill this need (STARTTLS, DKIM). But, because these mechanisms are optional extensions, there is no guarantee that they are used in all e-mail transactions. Therefore, these extensions have improved security in communication between servers, but not necessarily between users. For such, were created mechanisms point-to-point like PGP and S/MIME, giving to the user, control of security properties in communication, but their use is complicated.

The objective of this work is the creation of a solution to add automation and transparency to secure protocols point-to-point, pulling out to the user the burden of maintenance management of keys and cryptographic processes.

Keywords: SMTP, Email, PGP, S/MIME, DNS

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1 Introdução

1.1 Motivação

Numa era onde tem havido uma crescente preocupação com questões de segurança e privacidade não apenas em relação a criminosos como também a sistemas de vigilância a nível governamental, encontrar mecanismos que melhorem estas questões tanto nas comunicações eletrónicas em geral como na área do email, são desafios não só para a AnubisNetworks, como também para a comunidade científica.

O serviço de email permite que os utilizadores comuniquem entre si, sendo que essa comunicação pode incluir informação confidencial, como contas bancárias ou recuperação de passwords. Os utilizadores esperam que as mensagens transmitidas sejam privadas, e não possam ser alvo de falsificação ou roubo. O protocolo SMTP [7,8] é responsável por transmitir as mensagens de email entre servidores de email, normalmente definidos por *Mail Transfer Agent* (MTA).

Originalmente, o protocolo SMTP não continha qualquer tipo de segurança na sua comunicação. Posteriormente, extensões de segurança como STARTTLS [11,12], DKIM [26], SPF [27] e DMARC [28] foram criadas com o intuito de o enriquecer. Uma vez que estas extensões são opcionais, a comunicação pode ser feita sem qualquer proteção, sendo esta ausência de segurança do total desconhecimento do utilizador [13].

O *OpenPGP* [14] e o S/MIME [15] são protocolos que surgiram para permitir utilizadores finais utilizarem sistemas criptográficos, podendo assim adicionar confidencialidade, autenticidade, integridade e não-repudiação nas mensagens trocadas com outros utilizadores. Problemas na usabilidade destes sistemas [16,17,18], bem como a falta de capacidade para utilizarem estes sistemas leva a que a sua adoção não abranja grande parte dos utilizadores.

Propostas com foco em garantir a privacidade no armazenamento de mensagens de email e confidencialidade na comunicação entre utilizadores têm surgido como o *EmailCloak* [19], *P2P Mail Encryption* [20] e *Lightweight Encryption for Mail* [21], mas dada a sua complexidade, tanto nas infraestruturas, como nos processos, traduz-se na ausência de um mecanismo que possa ser adotado pela maioria dos utilizadores ou

fornecedores de serviço de email.

Este projeto propõe uma solução que permite automatizar e transparecer a utilização de protocolos criptográficos ponto-a-ponto, como o PGP [8] e S/MIME [9]. Dependendo do conhecimento em criptografia do utilizador, podem ser criados vários canais seguros com diferentes longitudes, e diferentes combinações de propriedades de segurança. A solução deve ser capaz de lidar com um canal *User-to-User*, criado por utilizadores.

1.2 Objetivo

O objetivo deste projeto passa pela simplificação e automação na criação de um canal seguro no serviço de email que garanta algumas propriedades de segurança, como confidencialidade, integridade, autenticidade e não-repudiação. O problema central para garantir estas propriedades num canal seguro consiste essencialmente na dificuldade na gestão de chaves e na falta de conhecimentos em criptografia por parte de utilizadores comuns, visto que os sistemas criptográficos já permitem garantir estas propriedades. Portanto, uma solução deva ser desenhada de modo a que todo o seu processo seja autónomo e transparente para o utilizador final.

O projeto consiste na construção de um filtro de email, colocado nos servidores de email que terá a função de cifrar/decifrar bem como assinar/validar assinaturas das mensagens de email em trânsito, descobrindo de forma automática as chaves privadas, chaves públicas ou certificados de utilizadores remetentes ou destinatários.

Para possibilitar automatismo, o DNS é utilizado para a publicação e descoberta de certificados de chave pública S/MIME e chaves públicas *OpenPGP*. Sendo o foco desta do projeto um ambiente empresarial, onde se assume que os pontos finais da comunicação segura é o MTA empresarial, as chaves privadas podem ser armazenadas no servidor de email responsável pela administração do seu domínio ou na posse dos utilizadores.

Para possibilitar a transparência nas operações, o canal seguro pode ser MTA para MTA (*MTA-to-MTA)* garantindo total transparência nas operações criptográficas e proteção das mensagens de email, MTA para utilizador (*MTA-to-User*) ou utilizador para MTA *(User-to-MTA),* garantindo a transparência nas operações apenas para um dos utilizadores envolvidos na comunicação, sendo que um dos utilizadores, tem de estar capacitado a manusear sistemas criptográficos.

Embora as operações criptográficas necessitem de processamento, e consequentemente o desempenho do serviço de email possa ser afetado por esse processamento adicional, o filtro de email não deve introduzir uma sobrecarga excessiva. Logo, em situações de carga típica, a sobrecarga introduzida deve permitir o normal funcionamento do serviço de email.

Este projeto, deve também ele, ser suportado por uma plataforma de gestão de chaves que permita a publicação de chaves púbicas e certificados de chave pública no DNS.

1.3 Integração do Projeto

Este projeto foi realizado em colaboração com a empresa AnubisNetworks. Fundada em 2006 por especialistas da indústria de telecomunicações, a AnubisNetworks é atualmente um dos líderes em *Threat Intelligence* na Europa e líder na área de segurança de email em Portugal. Alguns dos maiores operadores de telecomunicações mundiais tal como a Vodafone e a BT utilizam a tecnologia da AnubisNetworks.

A solução *Mail Protection Service* (MPS) oferecida pela AnubisNetworks faz parte do portfólio de Segurança de e-mail, que é composta por um conjunto de soluções customizadas para a gestão e filtragem de mensagens em ambiente de *multitenancy*[1], sendo a mais utilizada em Portugal.

Este trabalho insere-se no âmbito do MPS e pretende validar, através de um protótipo, uma solução que possibilite a criação de um canal seguro a nível aplicacional no serviço de email, que possa ser no futuro implementado no produto da AnubisNetworks.

1.4 Contribuições

Neste projeto, foi proposto um mecanismo para a criação de um canal seguro, que garanta a confidencialidade, autenticidade, integridade e não repúdio nas transações de email. Este mecanismo consiste em automatizar e tornar transparente os processos criptográficos de utilizadores finais através de um filtro de email, passando estas operações para os servidores de email.

[1] *Multitenancy* – É a referência ao modo de operação de software onde múltiplas instancias independentes de uma ou múltiplas aplicações operam num ambiente partilhado.

O filtro de email deve ser capaz de realizar uma descoberta das chaves criptográficas dos remetente e destinatários, usando para isso, a publicação de chaves públicas no DNS e o armazenamento local para chaves privadas.

O mecanismo foi desenhado com os seguintes objetivos:

- A aplicação de protocolos criptográficos ponto-a-ponto convencionalmente utilizados no serviço de email, como S/MIME e *OpenPGP*.
- Permitir vários modos de canal seguro (*MTA-to-MTA*, *MTA-to-User e User-to-MTA*), tendo em conta diferentes tipos de utilizadores.

Algumas propostas que permitem adicionar segurança no serviço de email foram analisadas, sendo que o projeto centrou-se na adaptação sistemas criptográficos já existentes para o serviço de email (S/MIME e *OpenPGP*), com foco na publicação de chaves públicas via DNS, e transparência e automatização das operações criptográficas.

No decurso do projeto, foi escrito e proposto, para o 8º Simpósio Nacional de Informática INFORUM, o seguinte artigo: J. Príncipe, J. Ferreira, M. Calha, J. Malcata e E. Alves "Transações de Email Seguras com Descoberta Automática de Chaves Criptográficas Públicas".

1.5 Planeamento

O desenvolvimento do projeto foi inicialmente dividido em tarefas, onde são descritas pelo seus respetivos *ID*s, representado no seguinte mapa da *Gantt* [Figura 1-1]:

ID	Início	Fim	Duração	Q3 15	Q4 15			Q1 16			Q2 16		
				Set	Out	Nov	Dez	Jan	Fev	Mar	Abr	Mai	Jun
1	15/09/15	30/10/15	46d										
2	02/11/15	20/11/15	19d										
3	23/11/15	31/12/15	39d										
4	04/01/16	01/03/16	58d										
5	02/03/16	08/04/16	37d										
6	11/04/16	13/05/16	29d										
7	25/04/16	15/06/16	52d										

Figura 1-1 – Mapa de *Gantt* representando o planeamento inicial do projeto.

1. Revisão de literatura (RFCs e artigos científicos): Levantamento de informação relacionada com PKI, SMTP e ESMTP e respetivos problemas e ataques para compreensão do problema. Estudo do S/MIME, PGP e análise de outras propostas que garantem um canal seguro no serviço de email, e respetiva análise.

2. Escrita do Relatório Preliminar: Documentação do estudo realizado no ponto **1** para o Relatório Preliminar, incluindo já alguns detalhes da solução.

3. Desenho da solução encontrada: Desenho de uma solução, definindo os casos de uso, requisitos funcionais e não funcionais, desenho da arquitetura e desenho do software.

4. Implementação de um protótipo: (*i*) Criação de ambiente para a implementação protótipo, que deve estar em concordância com o ambiente existente no produto da AnubisNetworks, e (*ii*) desenvolvimento do protótipo.

5. Validação do protótipo: (*i*) Testes funcionais e não funcionais, e (*ii*) validação dos requisitos identificados.

6. Análise de resultados: (*i*) Conclusões sobre a solução desenhada e o protótipo desenvolvido e (*ii*) validação de uma futura integração no produto na AnubisNetworks.

7. Escrita do Relatório Final: Documentação de todas as etapas realizadas.

Comparativamente com o planeamento final, inicialmente, e devido a algum desconhecimento técnico tanto na área do serviço de email como na forma como a solução seria desenvolvida, houve algumas alterações a nível da duração das tarefas.

Algumas das tarefas foram divididas, devido ao tempo despendido nas mesmas, nomeadamente a tarefa **4**, que se dividiu em 2 tarefas (tarefa **4** – Construção de um ambiente desenvolvimento e testes tarefa **5** – Implementação do protótipo no novo planeamento).

Contudo, o planeamento não sofreu alterações significativas, sendo que as datas definidas no planeamento inicial, foram compridas. O novo mapa de *Gantt*, representa o planeamento final [Figura 1-3], que se assemelha com o planeamento inicial, com a adição das novas tarefas acima identificadas.

As tarefas **1, 2 e 3** mantêm-se inalteradas.

4. Construção de um ambiente de desenvolvimento e testes: Criação de um ambiente onde seja possível implementar e testar um protótipo**.**

5. Implementação de um protótipo: Criação do protótipo sobre o ambiente criado.

6. Validação do protótipo: (*i*) Testes funcionais e não funcionais, e (*ii*) validação dos requisitos identificados.

7. Análise de resultados: (*i*) Conclusões sobre a solução desenhada e o protótipo desenvolvido e (*ii*) validação de uma futura integração no produto na Anubis-Networks.

8. Escrita do Relatório Final: Documentação de todas as etapas realizadas.

ID	Início	Fim	Duração	Q3 15	Q4 15			Q1 16			Q2 16		
				Set	Out	Nov	Dez	Jan	Fev	Mar	Abr	Mai	Jun
1	15/09/15	30/10/15	46d										
2	02/11/15	20/11/15	19d										
3	23/11/15	31/01/16	68d										
4	04/01/16	01/03/16	58d										
5	02/03/16	30/04/16	60d										
6	01/04/16	15/05/16	45d										
7	06/05/16	20/05/16	16d										
8	25/04/16	15/06/16	52d										

Figura 1-2 – Mapa de *Gantt* representando o planeamento final do projeto.

Capítulo 2 Conceitos e trabalho relacionado

2.1 Criptografia

Com o evoluir da tecnologia e dos seus meios de comunicação surgiu a necessidade de a transmissão de dados ser feita com confidencialidade, e nesse sentido surgiu a criptografia simétrica. A criptografia simétrica consiste numa única chave secreta, que é partilhada entre dois ou mais utilizadores, onde apenas quem possui essa chave está capacitado a cifrar e decifrar os dados, através do algoritmo de cifra, algoritmo de decifra, da chave secreta partilhada e do criptograma [Figura 2-1].

Figura 2-1 - Criptografia Simétrica.

Como a criptografia simétrica apenas garantia a confidencialidade dos dados transmitidos, surgiu a necessidade de se poder autenticar utilizadores. Para colmatar esta necessidade, *Diffie* e *Hellman*, nos finais dos anos 70, propuseram a criptografia de chave pública [1]. A sua proposta definia que qualquer utilizador de um sistema, podia enviar uma mensagem cifrada para outro utilizador, de tal modo, que apenas o utilizador a quem a mensagem se destinava, a podia decifrar. Após a descoberta, *Rivert*, *Shamir* e *Adelman* introduziram este conceito numa realização prática de um sistema de chave pública [2, 4].

A criptografia assimétrica baseia-se nos conceitos de chave pública e chave privada, onde a chave pública de um utilizador é publicamente conhecida, enquanto que a chave privada é apenas do conhecimento do próprio utilizador. Para garantir a confidencialidade [Figura 2-2a], os dados são cifrados com a chave pública do destinatário, sendo que só este os pode decifra-os utilizando a sua chave privada. Para garantir a autenticidade [Figura 2-2b], os dados são cifrados com a chave privada do utilizador que envia a mensagem, e o remetente, com a chave pública desse utilizador, decifra-os, podendo assim autentica-lo.

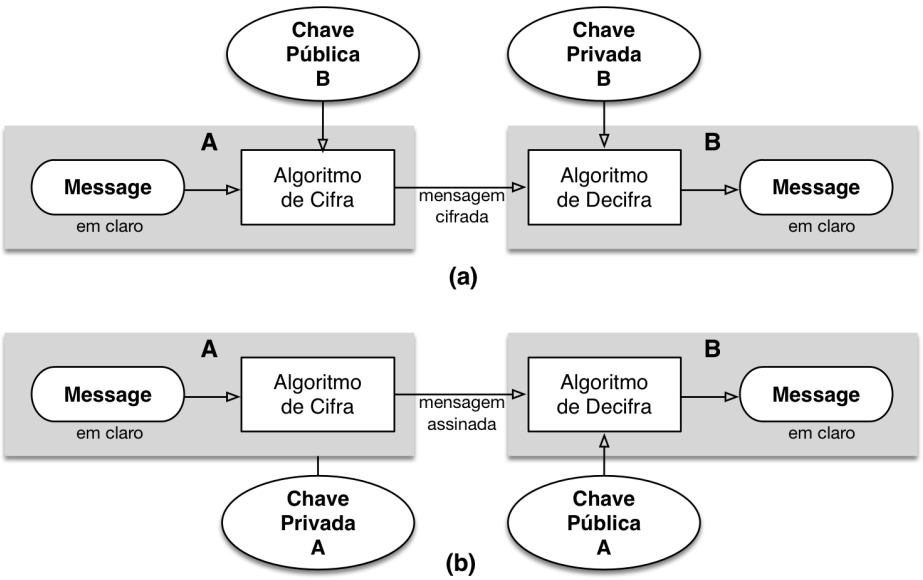

Figura 2-2 - Criptografia Assimétrica.
(a) Garantia de confidencialidade **(b);** Garantia de autenticidade

Como sugerido por *Diffie e Hellman* na sua proposta [1], as chaves públicas deveriam ser armazenadas num diretório público, onde utilizadores pudessem consultar e obter as chaves públicas de quem pretendem comunicar. Por forma a uma chave pública ser associada a um utilizador ou entidade, *Loren Kohnfelder* [3] propôs que essa associação poderia ser feita através de certificados, inseridos num sistema de distribuição de chaves, eficiente e escalável [3]. Mais tarde foi normalizado, sendo a normalização X.509 [4,5] a mais utilizada em certificados, contendo informação como, versão, número de serie, nome da entidade, chave pública, informação do algoritmo, período de validade, entre outros.

2.2 Serviço de Email

"O E-mail é regularmente utilizado (e inapropriadamente) para troca de dados sensíveis, confidenciais, ou comerciais, e muitos dos mais significantes ataques contra os protocolos de email alavancam a ausência geral de controlos de privacidade nesses mesmos protocolos, resultando em ataques aos dados do email." [6].

O Simple Mail Transfer Protocol (SMTP) [7,8] é um protocolo de internet utilizado para o envio e entrega de mensagens de email, entre servidores de email. É um dos mais antigos protocolos de comunicação e foi desenhado para entregar de forma confiável mensagens de email, operando sobre o protocolo de transporte TCP no porto 25.

O SMTP [7,8] é um protocolo orientado a texto que encaminha emails com base no conteúdo de vários campos, que no seu conjunto se denominam por cabeçalho. Os dados da mensagem de email (abaixo do cabeçalho) denomina-se de corpo da mensagem, separado por uma linha em branco do cabeçalho [Figura 2-3].

```
From: <user@smtpa.com>                                    1
To: <user@smtpc.com>
Subject: Book
Date: Mon, 17 Sept 2001 17:00:00 -700

Do you have the for the final manuscript?                 2
Your editor
```

Figura 2-3 - **Mensagem de email em texto simples.**
(1) Cabeçalho da mensagem; **(2)** corpo da mensagem

Sendo orientado a texto, possui algumas limitações como: (*i*) não permitir o envio de ficheiros executáveis, (*ii*) não transmitir mensagens com dados de texto que contivessem carateres de linguagem nacional, (*iii*) rejeitar mensagens de email acima de um determinado tamanho, entre outros. Para colmatar estas limitações surgiu o *Multipurpose Internet Mail Extensions* (MIME) [9]. O MIME especifica novos campos de cabeçalhos dando informação sobre o corpo da mensagem, formatos de conteúdo e respetiva codificações de transferência, bem como a versão do MIME utilizada. Sendo possível enviar múltiplos anexos de diferentes tipos (áudio, imagem, ficheiros e executáveis, entre outros). As mensagens de email que fazem utilização do MIME são designadas de mensagens com múltiplas partes [Figura 2-4].

A Figura 2-4 mostra uma mensagem de email com múltiplas partes, que contêm uma parte em texto simples, e uma parte com um anexo, nomeadamente um ficheiro de texto. Este tipo de mensagem contêm um campo no cabeçalho, *Content-Type* onde deve ser especificado que contem múltiplas partes e de que tipo são. Além desse conteúdo,

devem também conter um campo com a especificação de um delimitador (*Boundary=”*). Este delimitador é responsável por dividir o corpo da mensagem de email em várias partes, sendo que cada delimitador deve iniciar “--“, e o último terminar com “--“. Entre cada delimitador podem estar estão contidos campos de cabeçalho, que especificam o tipo (*Content-Type*), a disposição (*Content-Disposition*) e codificação de transferência (*Content-Transfer-Encoding*) de cada uma das partes.

```
From: <user@smtpa.com>
To: <user@smtpc.com>
MIME-Version: 1.0                                                    1
Content-Type: multipart/mixed; boundary="boundary"

--boundary                                                           2
Content-Type: text/plain

this is the body text

--boundary                                                           2
Content-Type: text/plain;
Content-Disposition: attachment; filename="test.txt"

this is the attachment text

--boundary--                                                         3
```

Figura 2-4 - **Mensagem de email com múltiplas partes.**
(1) Campos do cabeçalho da mensagem; **(2)** delimitador inicial e intermédio(s); **(3)** delimitador final.

O *Mail Transport Agent* (MTA), também conhecido como servidor de email, é responsável pela transferência das mensagens de email entre servidores, utilizando uma arquitetura de aplicação do cliente-servidor. Um MTA implementa uma parte cliente (envio de mensagens de email) e servidor (receção de mensagens de email). O processo de envio de uma mensagem de email ocorre da seguinte forma [Figura 2-5]:

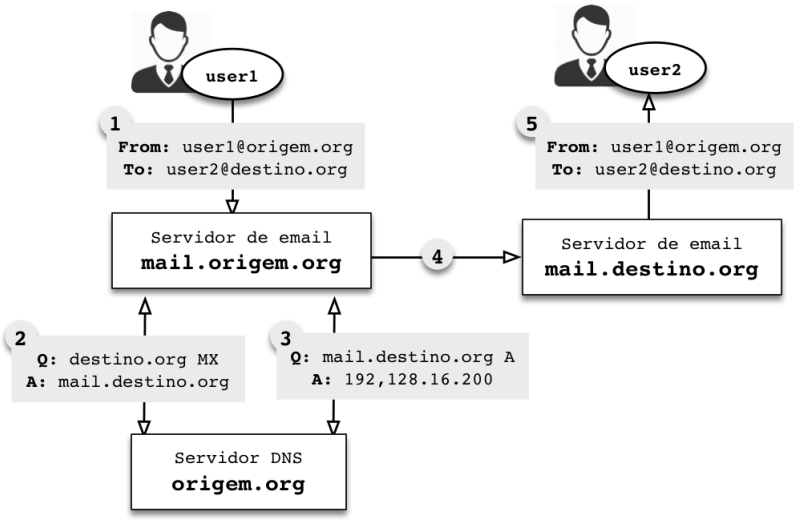

Figura 2-5 – Protocolo de SMTP.

1. O cliente `user1@origem.org` envia um email para o servidor SMTP local;
2. O servidor SMTP local faz um pedido DNS registo de recurso `MX` do domínio `destino.org`, que contem *hostname* do servidor SMTP de destino `mail.destino.org`;
3. O servidor SMTP local faz um segundo pedido DNS do endereço IP do servidor destinatário;
4. É estabelecida uma ligação ao servidor SMTP `mail.destino.org` e a mensagem reencaminhada;
5. Destinatário pode entregar mais tarde a mensagem ao `user2@destino.org` através do protocolo IMAP ou POP3.

O facto do SMTP [1,2] ter sido concebido no início dos anos 80, onde problemas relacionados com segurança não tinham a relevância dos dias hoje, levou a que a proteção da confidencialidade das mensagens de email em trânsito e da autenticidade na chegada ao destinatário não tenha sido considerada. Esta ausência de segurança pode ser explorada tanto por atacantes passivos, observando o tráfego na rede, como por atacantes ativos, alterando as mensagens de email.

Por forma a um MTA ter conhecimento das funcionalidades que o outro MTA suporta, surgiu o protocolo *Extended SMTP* (ESMTP) [10], que permite na comunicação entre MTAs, estes trocarem informação sobre as extensões que suportam, para posteriormente executá-las. Existem várias extensões como:

- *EHLO* – para inicialização uma sessão ESMTP;
- *SIZE* – para indicar o tamanho máximo de mensagens que o MTA aceita;
- *STARTTLS* [11] – para indicar que um MTA suporta TLS [12] e assim iniciar uma comunicação segura;
- *AUTH* – Para iniciar uma sessão ESMTP autenticada.

O ESMTP [10] funciona da seguinte forma [Figura 2-6]:

1. O cliente `user1@origem.org` envia um email para o servidor SMTP local;
2. O servidor SMTP local faz um pedido DNS registo de recurso `MX` do domínio `destino.org`, que contem *hostname* do servidor SMTP de destino `mail.destino.org`;
3. O servidor SMTP local faz um segundo pedido DNS do endereço IP do servidor destinatário;
4. É enviado um pedido ao servidor `mail.destino.org` para inicialização de uma sessão *ESMTP*, através da extensão *EHLO*;

5. O servidor `mail.destino.org` responde enviando a lista das extensões que suportada;
6. O servidor `mail.origem.org` estabelece uma ligação ao servidor SMTP `mail.destino.org` com e as extensões suportadas por ambos e a mensagem é reencaminhada;
7. Destinatário pode entregar mais tarde a mensagem ao `user2@destino.org` através do protocolo IMAP ou POP3.

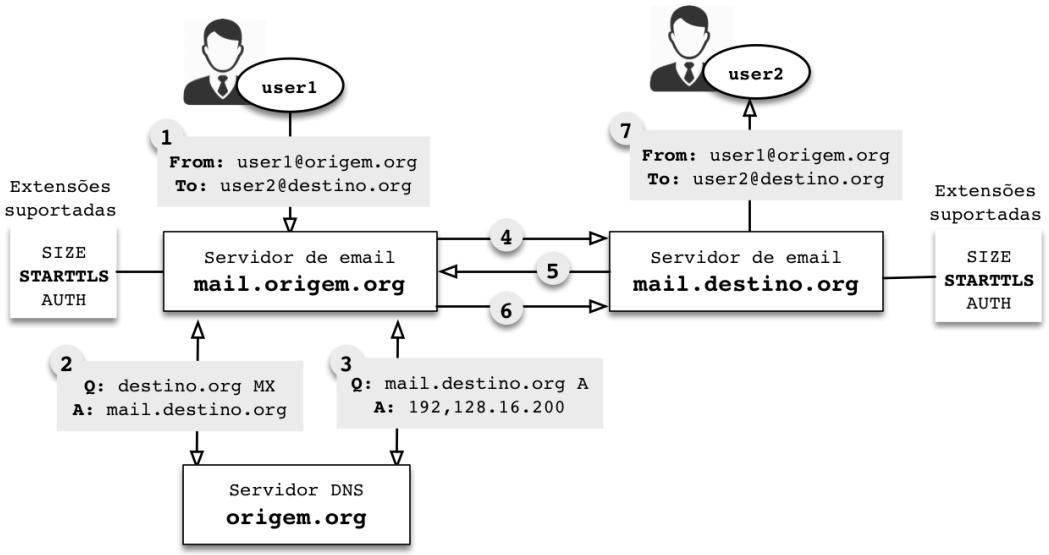

Figura 2-6 – Protocolo de ESMTP.

2.2.1 Proteção de mensagens em trânsito

Os servidores e clientes SMTP normalmente comunicam em claro sobre a internet. Em muitos casos a comunicação transita sobre encaminhadores que não são controlados ou da confiança de uma ou outra entidade. Tal que um encaminhador não confiável pode permitir uma terceira parte monitorizar ou alterar o conteúdo da comunicação entre servidor e cliente.

A extensão STARTTLS [11], permite o encapsulamento das mensagens de email sobre SMTP através de uma sessão TLS [12]. Uma sessão TLS é estabelecida, inicialmente através da troca de extensões suportadas pelos servidores SMTP envolvidos no envio e receção da mensagem de email. Se ambos suportarem a extensão STARTTLS é inicializado um TLS *handshake* padrão [12]. Assim, o conteúdo das mensagens de email é protegido no trânsito entre servidores.

A execução da extensão STARTTLS [11] é estabelecida de forma oportunista, ou seja, uma sessão TLS [12] só concretizada se ambos os servidores (remetente e destinatário) a suportarem, e houver o desejo de ser executada pelos mesmos. Isto leva à possibilidade de ocorrência de ataques de *Poisoning* e *Downgrade* no canal seguro [13].

2.2.2 Ataques no SMTP

Dado o oportunismo da execução das extensões do protocolo ESMTP, nomeadamente da extensão STARTTLS, ataques como *Poisoning* nos servidores de email, *Downgrade* no canal seguro, e consequente *Sniffing* da comunicação entre servidores de email [Figura 2-7].

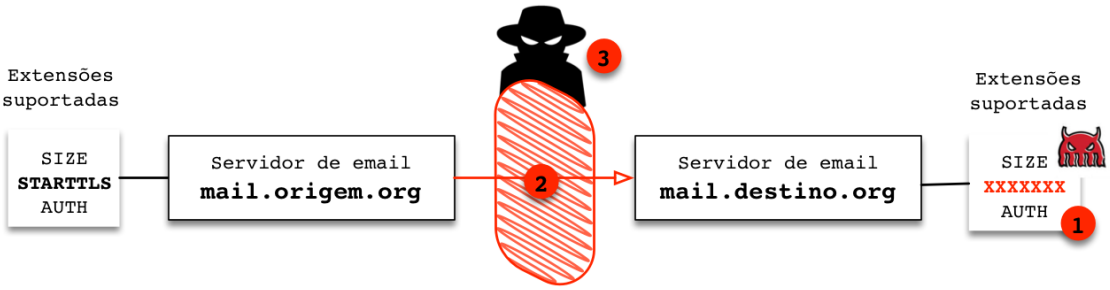

Figura 2-7 – Ataques no protocolo SMTP e ESMTP.
(1) *Poisoning*, **(2)** *Downgrade* e **(3)** *Sniffing*.

1. *Poisoning* – consiste na substituição da extensão STARTTLS, por outro conteúdo (XXXXXXXX).
2. *Downgrade* – por consequência de um ataque de *Poisoning,* resultando na não execução da extensão STARTTLS, a comunicação passa a ser feita em claro, ao invés de protegida por uma sessão TLS.
3. *Sniffing* – por consequência dos ataques de *Poisoning* e *Downgrade*, a comunicação, feita em claro, pode ser observada por uma terceira parte.

Estes três ataques levam a que a comunicação entre servidores de email possa ser alvo de ataques *Man-in-the-Middle* (MitM) [13].

2.2.3 Criptografia ponto-a-ponto

A necessidade de os utilizadores poderem eles próprios utilizarem sistemas criptográficos, criando entre eles um canal seguro sem a dependência de terceiros levou à criação de protocolos como o *Pretty Good Privacy* (PGP) [14] e o *Secure*/*Multipurpose Internet Mail Extensions* (S/MIME) [15].

O *OpenPGP* [14] é uma implementação do protocolo PGP que fornece serviços de segurança nas comunicações eletrónicas e armazenamento de dados, utilizando uma combinação de chaves públicas e criptografia simétrica. Estes serviços incluem confidencialidade, gestão de chaves, autenticação e assinaturas digitais. O *Secure/Multipurpose Internet Mail Extensions* (S/MIME) [15] fornece uma forma consistente para enviar e receber dados MIME seguros. Baseado na popular norma de internet MIME, o S/MIME fornece os seguintes serviços seguros criptográficos: autenticação, integridade e não repúdio da origem (assinaturas digitais), e confidencialidade dos dados (cifra).

Tanto o *OpenPGP* como o S/MIME criptograficamente garantem a possibilidade de se criar um canal seguro entre utilizadores finais, mas ambos apresentam falhas a nível de usabilidade [16,17,18]. Além destas falhas, a falta de automatismo e transparência na gestão das chaves perante os utilizadores também é um problema, e não existindo ainda uma solução normalizada que possa ser adotada pelas organizações.

O *OpenPGP* apresenta os seguintes problemas [16, 17]:

- *Obter e verificar as chaves públicas OpenPGP não é um processo integro*;
- *Os servidores de chaves não possuem métodos para validar as chaves públicas inseridas pelos utilizadores: limitam*-se a armazenar e publicar as chaves. Apenas o utilizador pode revogar as suas chaves públicas;
- A falta de um meio seguro para procurar uma chave pública impede a cifra de uma mensagem de email à sua saída.
- *Diferentes tipos de chaves (DDS/Diffie-Hellman e RSA):* o utilizador não possui conhecimentos criptográficos para poder distinguir os diferentes tipos de chaves.
- *Metáforas criptográficas:* algumas das metáforas criptográficas como a assinatura, não fazem referência ao uso da chave privada, não passando essa informação para o utilizador.
- *Demasiada informação visível:* o utilizador não tem consciência de ações involuntárias irreversíveis que este pode efetuar, e que levam ao comprometimento das suas chaves, como a perda acidental da chave privada.

Sendo que o S/MIME apresenta os seguintes problemas [18]:

- *Necessidade de Autoridades de Certificação (CA):* A constante necessidade de verificar a identidade de um certificado perante uma Autoridade de Certificação (CA);
- *Ausência de mecanismos para a validação da identidade de certificados auto assinados.*

Além do *OpenPGP* e S/MIME, outras propostas têm surgido, com foco em apresentar soluções que permitem adicionar criptografia ponto-a-ponto no serviço de email (*P2P Mail Encryption*, *Lightweight Encryption for Mail*), assim como privacidade no armazenamento (*EmailCloak*).

- **EmailCloak**

A proposta *EmailCloak* [19], foi desenvolvida com o propósito de garantir a privacidade no armazenamento de emails, ou seja, para quem não confia no armazenamento fornecido pelos serviços de email existentes (*Gmail, Hotmail*, entre outros). No entanto, esta proposta garante um canal seguro parcial.

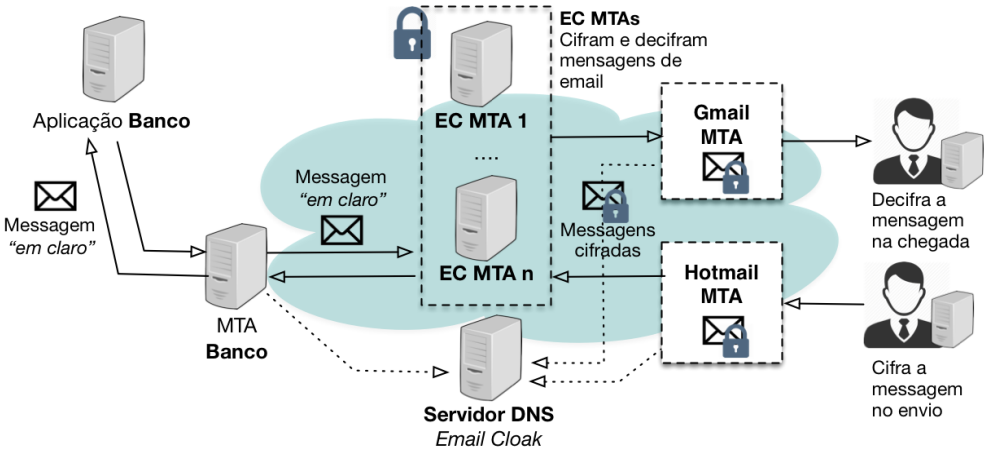

Figura 2-8 – *EmailCloak*: funcionamento geral.

Esta proposta consiste num *proxy* de vários *EmailCloak* MTAs (que podem ser distribuídos geograficamente) que intercetam emails que vão até utilizadores registados no serviço de *EmailCloak*. Estes *EmailCloak* MTAs cifram os emails que vêm para o utilizador com a sua chave pública. Quando o utilizador envia um email, este cifra o email, e quando chega aos *EmailCloak* MTAs, estes decifram o email, para este seguir para o destino [Figura 2-8].

Esta proposta apresenta, da mesma forma que os protocolos *OpenPGP* e S/MIME, alguns problemas a nível da gestão de chaves, pois as chaves públicas do utilizador têm de ser fornecidas pelos utilizadores ao serviço de EmailCloak (através de um registo via HTTPS).

- **P2P Mail Encryption**

A proposta *P2P Email Encryption* [20] propõe a implementação de um *Key Distribution Center (*KDC) nos serviços de email, ou seja, nos MTAs, que é responsável por gerar pares de chaves públicas e privadas, com base na identidade dos utilizadores, e partilhá-las com eles.

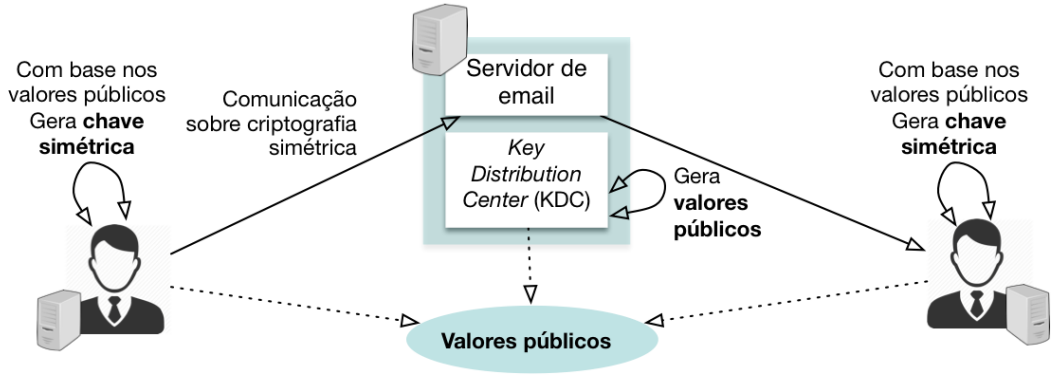

Figura 2-9 - *P2P Email Encryption*: funcionamento geral.

A geração das chaves é feita com base em alguns valores públicos (do conhecimento dos utilizadores e do KDC). Através destes valores públicos, e das chaves públicas e privadas, dois utilizadores conseguem gerar uma chave simétrica entre si, com base na identidade de cada um, e criar o canal seguro [Figura 2-9]. A chave é obtida através de emparelhamentos bilineares [20].

Devido ao processo de geração de chaves resultar na mesma chave simétrica para dois ou mais utilizadores, só é possível garantir confidencialidade num canal entre eles, deixando este canal carente das propriedades de autenticidade, integridade e não repudio.

- **Lightweight Encryption for Mail**

A proposta *Lightweight Encryption for Mail* [21] destina-se a garantir um canal seguro para utilizadores finais no serviço de email. Esta solução faz uso do DNS para publicação de *Master Public Keys* (`MPK`) associadas a domínios, e de um Servidor para

armazenamento de Certificados de utilizadores. Através da MPK, são geradas chaves para os utilizadores do seu domínio, que com as suas chaves públicas e privada (a pública, publicada no servidor de armazenamento de certificados), dois utilizadores conseguem alcançar uma mesma chave (simétrica), que é possível através de mapas Bilineares [21] [Figura 2-10].

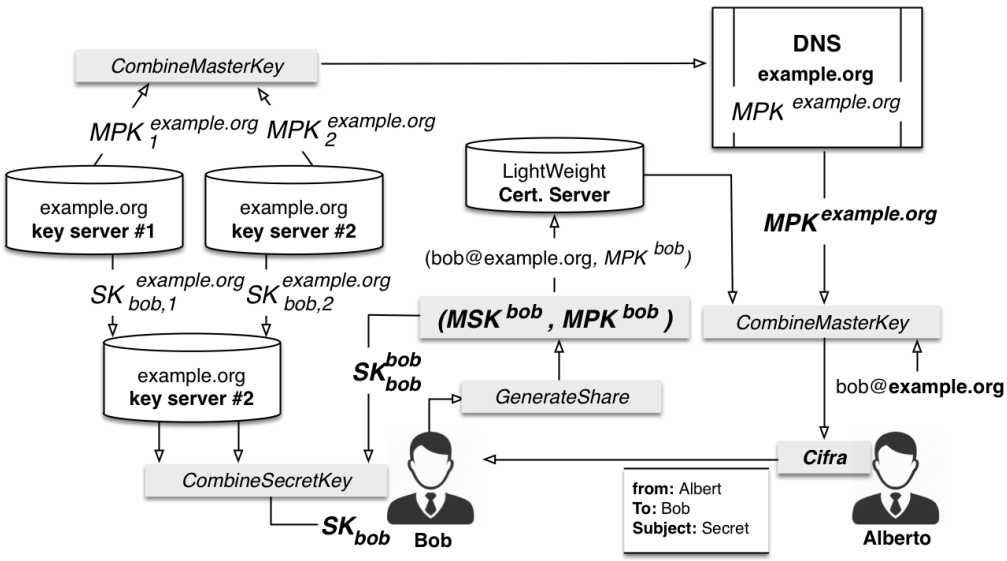

Figura 2-10 - *LightWeight Encryption for Mail*: funcionamento geral.

Para a publicação e gestão de chaves:

- Cada domínio (`foo.com`) com *n* servidores de chaves, cada um na posse de partilhas de uma chave pública, $MSK_1^{example.org}, ..., MSK_n^{example.org}$.
- As n partilhas são combinadas numa única chave pública $MPK^{example.org}$, e esta é armazenada no servidor de DNS;
- Cada domínio envia para o Bob uma partilha da sua chave secreta $SK_1^{foo.com}{}_{bob}, ..., SK_n^{example.org}{}_{bob}$ gerando uma $Sk_{bob}^{example.org}$.
- Para alcançar privacidade o Bob gera um par de chaves $(MSK^{bob}, MPK^{bob}) = SK^{bob}{}_{bob}$ e guarda a chave privada MSK^{bob};
- Bob publica `certificate = (`MPK^{bob}`, bob@example.org)` num Servidor de Armazenamento de certificados

Para enviar uma mensagem de email cifrada:

- Para o Alberto enviar uma mensagem cifrada para o Bob obtém a $MPK^{example.org}$, o certificado que contêm a MPK^{bob} e combina as chaves utilizando o `combineMasterKey(`$MPK^{example.org}$`, `MPK^{bob}`)`, cifrando a mensagem com a chave resultante;
- O Bob através das suas partilhas secretas: $SK_1^{foo.com}{}_{bob}, ..., SK_n^{foo.com}{}_{bob}$ gera uma $SK_{bob}^{foo.com}$, e com o par de chaves gerado por ele, $(MSK^{bob},$

MPK^{bob}) = SK^{bob}_{bob}, combina a chave secreta combineSecretKey($Sk_{bob}^{foo.com}$, SK^{bob}_{bob}).

- Neste momento, ambos têm a mesma chave e podem obter o canal seguro.

Esta proposta foi pensada para proteção contra possíveis ataques, como por exemplo, no caso de algum dos servidores de chaves ser comprometido. Caso ocorra, tanto a MPK como as chaves dos utilizadores nunca são comprometidas, pois são divididas pelos vários servidores de chaves e posteriormente combinadas para a realização de operações. No entanto traduzir esta proposta para uma implementação, leva a uma mudança grande nas infraestruturas existentes, com a adição de servidores de chaves contendo as chaves dos utilizadores, partilhadas entre servidores, servidor de certificados contendo certificados com a associação dos endereços de email dos utilizadores com a sua chave pública, e ainda publicações das chaves públicas dos domínios de email no DNS.

2.2.4 Conclusão

O *EmailCloak* [19] apresenta melhoria na gestão de chaves, pois os MTAs *proxies* tratam de cifrar os dados de forma automática e transparente, sem interações do utilizador, mas a exigência de uma troca de chaves entre o utilizador e uma interface, para os MTAs *proxies* cifrarem as mensagens, pode traduzir-se em problemas de usabilidade. Sendo uma boa proposta para alcançar a privacidade no armazenamento, garante o principal objetivo (canal seguro) de forma parcial (dados cifrados entre utilizador e MTAs *proxies*), não sendo por isso uma opção uma opção a considerar.

O *P2P Email Encryption* [20] não melhora o processo de gestão de chaves em aspetos como usabilidade, transparência e automatismo, pois é necessário a distribuição de valores pelos utilizadores, a publicação de valores públicos para serem geradas chaves simétricas (de chave secreta), e ainda, a necessidade de um *Key Distribution Center* (KDC) para a distribuição dos valores.

O *Lightweight Encryption for Email* [21] efetua a gestão de chaves à custa da utilização de servidores de chaves, e de uma infraestrutura para o armazenamento e consulta de certificados, além de novos processos e infraestruturas dificultarem a usabilidade to sistema, torna-se pouco automático e transparente devido à quantidade de processos envolvidos para concretizar um canal seguro.

Tanto a *Lightweight Encryption for Email* [21] como *P2P Email Encryption* [20] apresentam uma solução que permite obter um canal seguro, assim como novas formas para gestão e partilha de chaves, mas o facto de ainda não terem sido testadas e implementadas na indústria, bem como utilizarem métodos criptográficos inovadores para a

partilha e geração de chaves secretas, pode ser uma barreira para a sua adoção. Além destes pontos, o facto dos processos para concretizar o canal seguro, resultarem em chaves simétricas, apenas permite garantir confidencialidade, tornando estas propostas mais frágeis do ponto vista de segurança comparativamente às já utilizadas *OpenPGP* [14] e S/MIME [15], que além da confidencialidade, garantem a autenticidade, integridade e não repúdio.

É necessário com isto, encontrar mecanismos que permitam tornar a utilização do *OpenPGP* [14] e do S/MIME [15] perante o utilizador e o sistema, mais automática e transparente no que toca às operações criptográficas, e usável no que diz respeito ao processo da gestão de divulgação das chaves criptográficas públicas.

2.3 DNS

A Internet funciona devido à associação de endereços IP a todos os seus componentes (*endpoints)*. Um endereço de IP consiste num conjunto de 32 bits para IPv4 ou 128 bits para IPv6. Por exemplo, associando o endereço IP `216.58.208.3` ao nome de domínio `example.net`, que quando acedido, estabelece uma ligação a página *Web*. É muito mais fácil para os humanos fixar nomes de domínios do que endereços IP, e sendo que existem milhões de *endpoints*, cada um necessita de um endereço de IP, que por sua vez está associado a um nome de domínio. O protocolo de DNS resolve esta associação, sendo que traduzindo nomes em endereços IP, bem como o contrário.

O DNS [22, 23] utiliza uma estrutura hierárquica, onde o topo desta estrutura é designado de *root*. O segundo nível designa-se por *Top-Level domains* (TLDs), e os restantes abaixo, designados por *Second Level domains* (SLDs). Os TLDs podem ser separados em dois grupos: *Generic Top-Level domains* (gTLD) e *Country Code Top-Level Domains* (ccTLD), o primeiro para domínios com `.net`, `.com` e `.org`, o segundo para domínios como `.pt`, `.uk` e `.us`. No DNS, cada nome de domínio consiste em uma ou mais camadas, onde a mais à direita representa o maior domínio na sua estrutura hierárquica [Figura 2-11]. A representação do nome de domínio com o "." à direita designa-se *Full Qualified Domain Name* (FQDN).

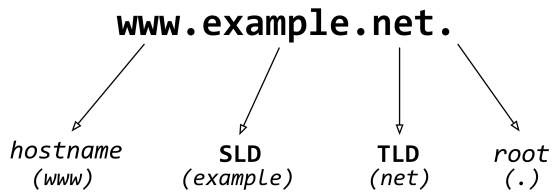

Figura 2-11 – *Full Qualified Domain Name* (FQDN).

Cada nó, no espaço de nomes de domínio é atribuído a uma autoridade, isto é, alguém que é responsável pela gestão desse nó. As autoridades mantêm um ou vários servidores de nomes, podendo delegar autoridade a outras partes (outros servidores de nomes de menor nível), sendo que devem estar capacitados a referir que outros servidores podem responder ao pedido. Os Servidores de Nomes Autoritários mantêm a informação das suas zonas em chamados de ficheiros de zona. Estes ficheiros de zona resultam de uma coleção de registos de recursos (RR). Cada registo tem um nome, um tipo (por exemplo, `A`, `NS`, `MX`, entre outros), um *time-to-live* (TTL), uma classe e tipo de dados específico. Essencialmente, os conjuntos de registos de recursos no DNS formam um enorme repositório de dados distribuído.

As comunicações DNS são tratadas numa única mensagem, constituída por várias secções:

- *question section,* que contém o nome de domínio pedido
- *answer section,* contém o registo (`A, NS, MX`), que responde ao pedido.
- *authority section,* contém um apontador para o servidor de nome autoritário que respondeu ao pedido
- *additional section,* contém registos (`RR`) como endereços IP de servidores de nomes com autoridade que respondem para o domínio.

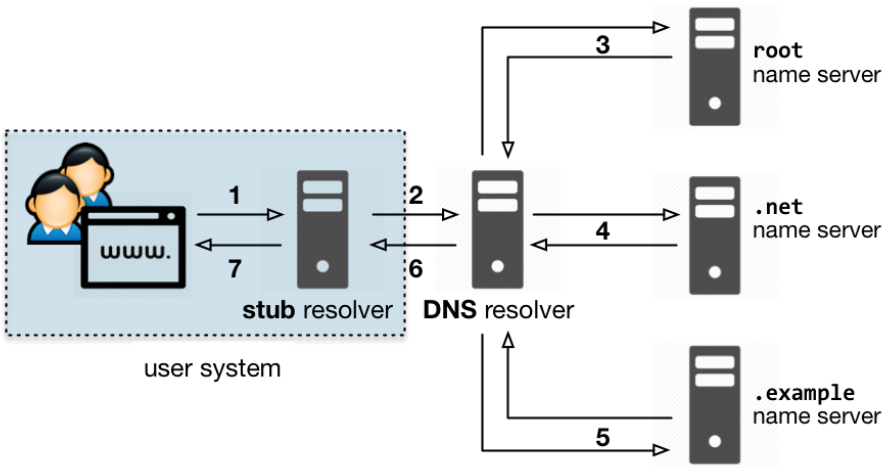

Figura 2-12 - Pedido de DNS: fluxo de trabalho.

O fluxo de trabalho de um pedido DNS no acesso a um domínio (por exemplo, `www.example.net`) via um *web browser* funciona na seguinte forma [Figura 2-12]: O *browser* invoca, localmente, o *stub resolver*, que é responsável por controlar todos os pedidos de DNS feitos pelo utilizador (1), que por sua vez envia um pedido a um DNS *resolver* recursivo (operado por um *Internet Service Provider*) (2). O DNS *resolver* en-

via um pedido para um servidor de nomes com autoridade para a zona de `root` (que possuem endereços IP padrão). O servidor contactado não tem autoridade para `www.example.net`, mas possui endereços IP de servidores de nomes com autoridade para o domínio `net`, respondendo ao DNS *resolver* com referencia a esses servidores (3). O DNS *resolver* envia um pedido para um servidor de nomes com autoridade para a zona de `net`. O servidor contactado não é autoritário para `www.example.net`, mas possui endereços IP de servidores de nomes com autoridade para o domínio `example.net`, respondendo ao DNS *resolver* com referencia a esses servidores (4). O DNS *resolver* faz um pedido a um dos servidores de nomes `example.net`, que ao ter autoridade para `www.example.net`, responde com o seu endereço IP respetivo, `216.58.208.3` (5). DNS *resolver* responde com esse mesmo IP ao *stub resolver*, que o entrega ao *web browser*.

2.3.1 DNSSEC

O DNSSEC (*Domain Mail System Security Extensions)* [24] surgiu para fornecer autenticidade e integridade dos dados do tradicional DNS, ao juntar a cada resposta de DNS uma sequência de assinaturas digitais.

O DNSSEC introduziu novos tipos de registos de recurso (RR) como:

- RRSIG – utilizado para armazenar a assinatura digital de um conjunto de RRs.
- DNSKEY – contem a chave pública de um par de chaves criptográficas (pública e privada). No DNSSEC, são utilizados dois tipos de chaves *Zone Signing Keys* (ZSK) e a *Key Signing Key* (KSK). A ZSK assina todos os conjuntos de RRs na zona. A KSK assina ambas as chaves, para serem autenticadas cada uma das chaves.
- DS – utilizado para verificar a autenticidade do registo KSK DNSKEY para uma determinada zona.
- NSEC(3) – contem todos os tipos de registos de um dado nome de domínio numa zona e o nome para o próximo registo. Assim, cada nome de domínio tem o seu próprio NSEC. Para prevenir que um atacante descubra todos os nomes de domínio de uma zona, o registo NSEC3 contem uma síntese do nome de domínio ao contrário do NSEC que o contêm em claro.

Quando um *resolver* validador recebe uma resposta DNSSEC com um conjunto de RRs e uma assinatura, ele pode solicitar a chave pública ao servidor de nomes autoritário e verificar se a assinatura corresponde ao conjunto de RRs solicitado. Para proporcionar autenticação completa, o *resolver* validador também deve verificar a autenticidade da chave pública de ZSK, verificando a assinatura da ZSK com chave púbica do par KSK. Por sua vez, a autenticidade do KSK também deve ser verificada e, logo, uma síntese da KSK é colocada como um registo DS na zona acima. O registo DS por sua vez, é assinado por um outro conjunto de chaves e esta cadeia de confiança pode ser verificada e validada até à zona da raiz (`root`). [Figura 2-13].

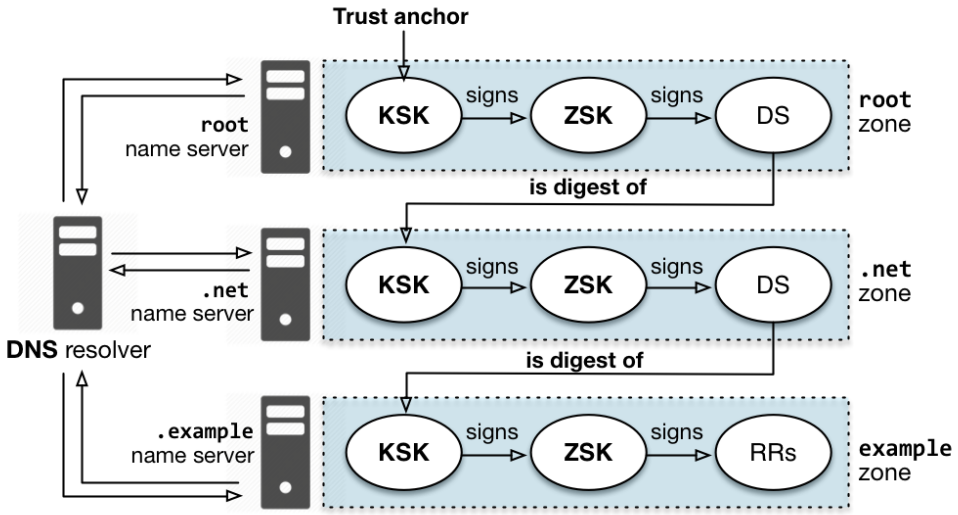

Figura 2-13 – DNSSEC: cadeia de confiança para o domínio `example.net`.

Como mencionado no anteriormente, o DNSSEC permite garantir a autenticidade e integridade das respostas dadas pelo DNS. Sendo o DNS uma plataforma de fácil acesso e fortemente utilizada por todos os *endpoints* na Internet, novos tipos de RR surgiram como DANE [25], inicialmente concebido para a *Web,* e DKIM [26], SPF [27], DMARC [28] concebidos para autenticação no serviço de email.

2.3.2 Autenticação no serviço de email

Por forma a que as mensagens de email cheguem aos destinatários com a garantia, de que efetivamente vêm da fonte que informam vir (autenticidade e não-repudiação), e também, que não foram alvo de falsificação durante a comunicação (integridade), levou à criação de mecanismos para validação da autenticação, integridade não repudiação das mensagens de email (DKIM [26]), assim como mecanismos autorização no envio de

mensagens de email entre servidores (SPF [27]) e definição de comportamento de servidores na validação destes mecanismos (DMARC [28]).

- *DomainKeys Identified Mail* (DKIM) [26] permite um servidor SMTP, na receção de uma mensagem vinda de outro servidor SMTP, verificar se esta foi enviada efetivamente pelo servidor remetente (autenticidade e não-repudiação), e ainda, verificar que não foi modificada em trânsito (integridade).
- *Sender Policy Framework* (SPF) [27] permite a publicação, num registo de recurso via DNS, de um conjunto de endereços IP públicos que estão autorizados a enviar mensagens de email num dado domínio.
- *Domain-based Message Authentication, Reporting, and Conformance* (DMARC) [28] permite a publicação de políticas, que permite os destinatários tomar decisões de aceitação ou rejeição de uma determinada mensagem, com base em DKIM [26] e SPF [27].

Cada uma destes mecanismos, utiliza o DNS para publicar o seu conteúdo, através de registos de recurso (RR) nos ficheiros de zona dos respetivos domínios, e relacionam-se na seguinte forma [Figura 2-14]:

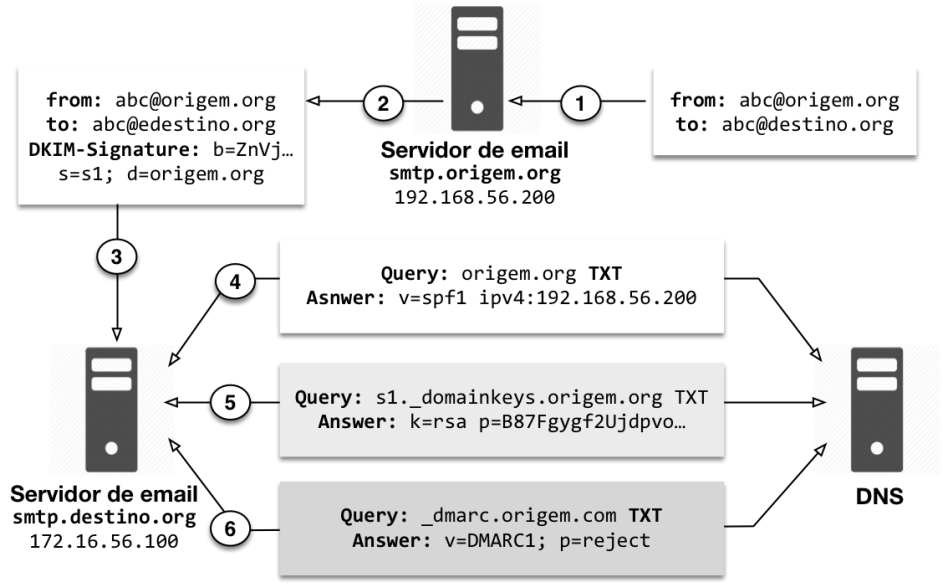

Figura 2-14 – **Autenticação de email.** Utilização de SPF, DKIM e DMARC para autenticar a origem (`smtp.origem.org`).

1. Uma mensagem de email chega até ao servidor de email `smtp.origem.org`.
2. O servidor `smtp.origem.org` assina digitalmente a mensagem, incluindo-a na mensagem um campo no cabeçalho *DKIM-Signature* com mais informações

relacionadas com a assinatura, e a origem (`smtp.origem.org`).

3. O servidor de email destinatário `smtp.destino.org` realiza um pedido por SPF ao DNS, para verificar se `smtp.origem.org` está na lista de servidores autorizados.
4. O servidor de email destinatário `smtp.origem.org` realiza um pedido por DKIM ao DNS, para obter a chave pública utilizada na assinatura (pertencente a `origem.org`) .
5. Por fim, realizada um pedido por DMARC ao DNS, para determinar a correta ação que deve tomar, caso a validação da assinatura, contida no cabeçalho de email (*DKIM-Signature*), falhe.

Outras propostas com base em publicações no DNS para reforçar a segurança no serviço de email, com foco na confidencialidade, utilizando os métodos tradicionais de criptografia para utilizadores finais no serviço de email (*OpenPGP* [14] e S/MIME [15]).

2.3.3 Publicação de chaves públicas *OpenPGP*

O objetivo principal do registro de recurso OPENPGPKEY [29] é parar os ataques passiva contra emails em texto simples (MitM). Embora possa também impedir alguns ataques ativos (como o *upload* de chaves públicas para servidores de chaves desonestos), este registo de recurso não substitui a verificação de chaves públicas *OpenPGP* via *web* de assinaturas de confiança, ou manualmente através de uma verificação de impressões digitais.

O registro de recurso OPENPGPKEY [29] consiste na associação de uma chave pública *OpenPGP* a um endereço de email. Este registo é baseado em DANE [25] e deve ser armazenado no ficheiro de zona do domínio ao qual o endereço de email pertence. É composto da seguinte forma [Figura 2-15]:

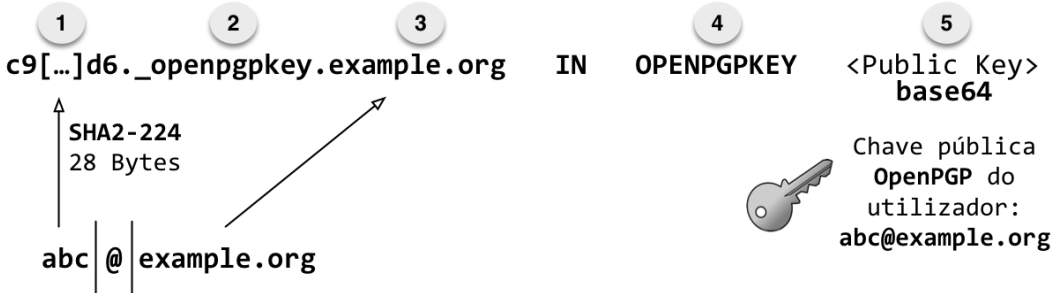

Figura 2-15 – Registo de recurso `OPENPGPKEY`

(1) Síntese da parte local da conta de email do utilizador (`abc`, antes convertida para UTF-8;
(2) Indicação do tipo do RR (`OPENPGPKEY`);
(3) Domínio da conta de email do utilizador (`example.org`);
(4) Tipo de RR (`OPENPGPKEY`);
(5) Chave pública *OpenPGP* do utilizador.

Através desta proposta, é possível qualquer utilizador através de um pedido ao DNS por este tipo de RR (`OPENPGPKEY`) obter uma chave pública *OpenPGP*, de qualquer utilizador que possua, publicado no DNS, este RR. A implementação deste registo de recurso fica mais segura se os ficheiros de zona onde devem ser inseridos implementar DNSSEC.

2.3.4 Publicação de certificados S/MIME

Algumas pessoas querem autenticar a associação de um certificado S/MIME remetente com o remetente sem confiar numa âncora de confiança configurada. Dado que o administrador de DNS para um nome de domínio está autorizado a dar informações de identificação sobre a zona, faz sentido permitir que um administrador possa também fazer uma ligação de autoridade entre as mensagens de email venham do nome de domínio e um certificado S/MIME, que possa ser utilizado por alguém autorizado a enviar mensagens de emails a partir desses servidores. A maneira mais fácil de fazer isso é usar o DNS.

O registro de recurso SMIMEA [30] consiste na associação de um certificado de utilizador (que contêm o endereço de email do utilizador) a um domínio. Este registo é baseado em DANE [25] e deve ser armazenado no ficheiro de zona do domínio ao qual o endereço de email pertence. É composto da seguinte forma [Figura 2-16]:

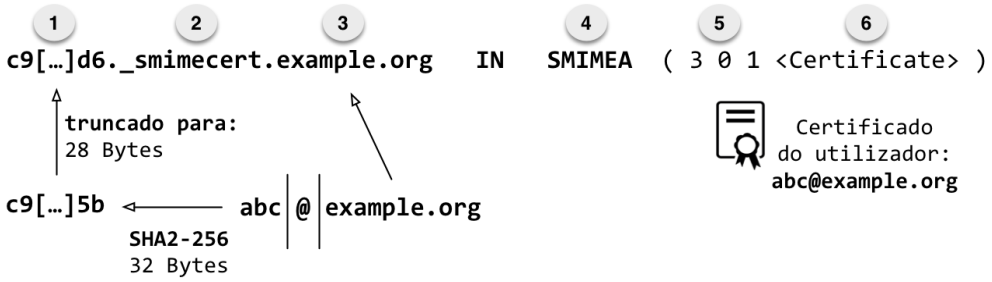

Figura 2-16 – Registo de recurso `SMIMECERT`.

(1) Síntese da parte local da conta de email do utilizador (`abc`, antes convertida para UTF-8, e no final truncada);
(2) Indicação do tipo do RR (`SMIMECERT`);
(3) Domínio da conta de email do utilizador (`example.org`)*;*
(4) Descrição dos dados de associação do certificado: *Certificate Usage*, *Selector* e *Matching Type* indicam a forma como os dados do certificado são representados;
(5) Tipo de RR (`SMIMEA`);
(6) Dados de associação do certificado, que são definidos através do conteúdo dos seletores **(4)**.

Através desta proposta, é possível qualquer utilizador através de um pedido ao DNS por este tipo de RR (`SMIMEA`) validar a correspondência de um certificado S/MIME a um determinado utilizador. De igual forma ao `OPENPGPKEY`, A implementação deste RR fica mais segura se os ficheiros de zona onde devem ser inseridos implementar DNSSEC.

2.4 Filtro de email

Um filtro de email, normalmente designado por "*milter"* [31], consiste num protocolo entre o MTA e um programa externo, que é utilizado para permitir ao programa externo, a observação e influência sobre o fluxo de email, afetando a disposição da mensagem ou alterando o conteúdo da mensagem.

Um "*milter*" oferece inúmeras funcionalidades, tais como:

- Modularizar as ações de filtragem e de política
- Distribuir a carga de filtragem e de políticas por entre diferentes máquinas (para não estancar gestão de filas das máquinas)
- Reforçar políticas sobre diversos MTAs numa localização central.
- Colecionar informação de tráfego, através de diversos MTAs numa localização central.

Um cliente SMTP conecta-se ao MTA, consequente do envio de uma mensagem de email, que transitou até esse MTA. O MTA estabelece uma nova conexão para cada aplicação "*milter"* nele configurada.

A cada etapa do processo do SMTP, o MTA passa informação relevante para a aplicação *"milter"*, como o cabeçalho ou o corpo da mensagem de email. Cada filtro aceita e analisa a informação fornecida e retorna um código de estado (*CONTINUE, REJECT, ACCEPT, TEMPFAIL, DISCARD*) [Figura 2-17]. Cada filtro pode também fazer uso de algumas funções úteis para obter informação adicional do MTA, podendo entre elas, solicitar alterações de cabeçalho ou corpo.

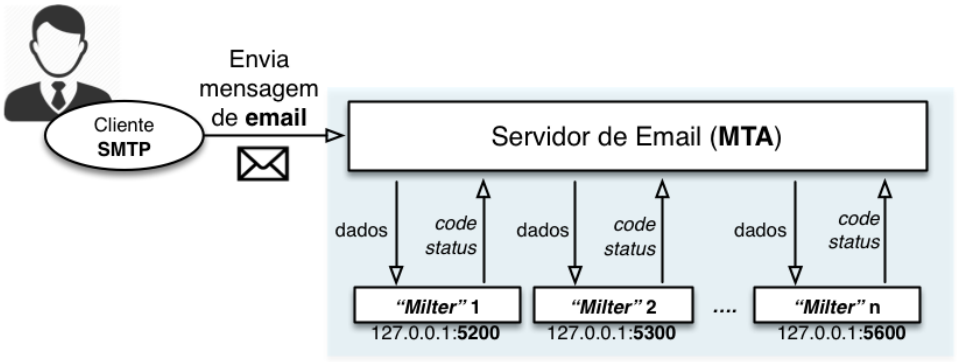

Figura 2-17 - Configuração do MTA para diversas aplicações "*milter*".

A conexão entre o MTA e a aplicação "*milter*" é estabelecida através de *socket*, a aplicação "*milter*" fica à escuta num determinado num determinado porto. A cada conexão de um cliente SMTP com o MTA, é gera uma nova conexão a cada uma das aplicações "*milter*" configuradas nesse MTA [Figura 2-16].

Através de aplicações "*milter*" além de ser possível bloquear emails indesejados (filtragem), é possível ainda verificar autenticidade ou assinar digitalmente emails, através das abordagens referidas anteriormente (DKIM, SPF e DMARC).

2.5 Conclusão

Foi introduzido o protocolo SMTP, assim como as suas fragilidades a nível de privacidade e segurança. Essas fragilidades podem ser contrariadas com mecanismos criptográficos ponto-a-ponto (S/MIME e *OpenPGP*), embora ambos apresentem alguns problemas a nível de usabilidade e gestão de chaves.

A importância do DNS no incremento da segurança no serviço de email, levou à introdução do seu funcionamento geral, assim como dos mecanismos de segurança por ele suportados, como o DNSSEC, DKIM, SPF e DMARC.

Novas propostas surgiram baseadas em DNS, sendo que duas delas visam a publicação de chaves públicas *OpenPGP* e certificados S/MIME, apresentando uma oportunidade para melhorar o processo de gestão de chaves destes protocolos. Por fim, foi apresentado o funcionamento geral dos filtros de email, para a compreensão do processamento sobre as mensagens de email em trânsito nos servidores de email.

As propostas de âmbito cientifico analisadas apresentam soluções válidas [28, 29, 30], mas as infraestruturas necessárias para o seu funcionamento, os novos métodos criptográficos nelas empregues, e a garantia de confidencialidade (criptografia simétrica) em detrimento de propriedades como a autenticidade, integridade, e não repúdio, tornam estas propostas distantes da industria, que necessita de mecanismos de fácil integração com sistemas já existentes, e de rápida aprendizagem por parte dos utilizadores e seus intervenientes.

Posto isto, e analisando todos os conteúdos descritos neste capitulo, decidiu-se desenhar um mecanismo que faz uso DNS como suporte para a publicação de chaves públicas *OpenPGP* e certificados S/MIME [29, 30]. Utilizando filtros de email, colocados servidores de email, é possível estes consultarem automaticamente o DNS, e obter chaves públicas e certificados de destinatários e remetentes, realizando operações criptográficas por eles.

Capítulo 3 Análise de Requisitos e Desenho

3.1 Descrição

Este trabalho consiste no desenvolvimento de um mecanismo que permita a criação de um canal seguro (*MTA-to-MTA*, *MTA-to-User* e *User-to-MTA*) no serviço de email. Este mecanismo deve ser capaz de descobrir as chaves criptográficas pública ou privada tanto de remetentes como de destinatários no trânsito de uma mensagem de email, e operando criptograficamente pelos utilizadores.

Esta descoberta deve ser feita através de um filtro de email, colocado nos servidores de email, que para além da descoberta de chaves públicas e certificados é responsável pelas operações criptográficas sobre mensagens de email, intercetadas por ele. As chaves públicas e certificados devem ser publicadas em registos de recurso (RR) nos ficheiros de zona DNS, e as chaves privadas armazenadas localmente de forma segura. Este mecanismo permite adicionar automatismo e transparência às operações criptográficas, que tardam em ser fortemente adotados por utilizadores finais devido à sua complexidade de utilização.

Para possibilitar a integração deste mecanismo com o utilizador, são utilizados os principais protocolos de criptográfica ponto-a-ponto (S/MIME e *OpenPGP)* utilizados no serviço de email. Esta integração é essencial para a concretização dos canais seguros de *MTA-to-User* e *User-to-MTA*, onde o utilizador faz parte da solução.

Este trabalho incluirá uma plataforma de gestão de chaves que possibilite realizar a conversão de chaves públicas PGP ou certificados S/MIME para RRs, e a sua publicação no DNS. Deve ainda incluir operações como consultar, atualizar e remover RRs. Esta plataforma pode ser acedida pelo administrador do domínio onde o filtro de email está configurado.

Este mecanismo, será desenvolvido numa primeira fase como um protótipo num ambiente de teste, de modo a validar esta opção para a AnubisNetworks e comunidade científica.

3.2 Especificação de Requisitos

Nesta secção serão descritos os requisitos funcionais e não funcionais. Os requisitos funcionais devem definir as funções dos componentes do projeto, em conformidade com os casos de uso, enquanto que os requisitos não funcionais devem definir o quão bem as aplicações têm de desempenhar as suas funções.

3.2.1 Requisitos Funcionais

O filtro de email deve suportar as seguintes **características** e **funcionalidades**:

- **Descoberta de chaves públicas ou certificados:** ao intercetar uma mensagem, deve ser capaz de descobrir as chaves públicas de um destinatário, caso esta exista;
- **Capacidade de decidir entre decifrar a mensagem ou não decifrar a mensagem:** deve ser capaz de verificar se num determinado MTA, a mensagem deve ser decifrada pela aplicação, ou deve ser decifrada pelo utilizador;
- **Cifrar a mensagem:** deve ser capaz de cifrar a mensagem, após descobrir a chave pública ou certificado do destinatário;
- **Decifrar a mensagem:** deve ser capaz de decifrar uma mensagem de email, caso esta venha corretamente cifrada;
- **Assinar a mensagem:** deve ser capaz de assinar a mensagem, se a chave privada estiver junto do MTA;
- **Verificar assinaturas:** deve ser capaz de verificar a assinatura de uma mensagem de email, caso esta venha corretamente assinada;

O filtro de email deve ainda suportar os seguintes **requisitos específicos de projeto:**

- **Operações criptográficas devem ser feitas através dos protocolos S/MIME ou *OpenPGP*;**
- **A descoberta de chaves públicas deve ser feita através do DNS:** Através de registos de recurso (RR) publicados no DNS;

- **Suportar diferentes tipos de RRs**: suportar RRs do tipo OPENPGPKEY [29] e SMIMECERT [30], o primeiro para chaves públicas *OpenPGP*, e o segundo para certificados de chave pública S/MIME.

A plataforma de apoio na gestão, conversão e publicação de chaves públicas deve suportar as seguintes **características** e **funcionalidades**:

- **Conversão de chaves públicas *OpenPGP* e certificados de chave pública S/MIME em RRs:** deve ser possível converter uma chave pública *OpenPGP* ou um certificado S/MIME válido, em RRs [29,30];
- **Publicação de RR no DNS:** deve permitir a publicação dos RRs, corretamente convertidos, em ficheiros de zona DNS;
- **Operações de gestão de chaves públicas e certificados:** deve permitir o administrador realizar operações de obter, remover e atualizar chaves públicas *OpenPGP* e certificados S/MIME;

A plataforma de gestão de chaves deve ainda suportar os seguintes **requisitos específicos de projeto:**

- **Deve ter um estilo arquitetural do tipo *RESTful web service*[2]**

3.2.2 Requisitos Não Funcionais

As aplicações devem suportar os seguintes requisitos não funcionais:

- **Transparência:** todas as ações realizadas pelo filtro de email, devem ser o mais transparentes possíveis na ótica do utilizador final:
    - Num canal seguro *MTA-to-MTA*, a transparência deve ser total em todos os processos.
    - Num canal seguro *MTA-to-User* e *User-to-MTA*, a transparência deve ser total para um dos utilizadores finais.

[2] `http://docs.oracle.com/javaee/6/tutorial/doc/gijqy.html`

- **Autonomia:** todas as ações realizadas pelo filtro de email têm de ser automáticas (sem intervenção humana).
- **Usabilidade:** as operações na plataforma de apoio na gestão devem ser simples e compreensíveis.
- **Desempenho:** A carga introduzida pelo filtro de email em situações de carga típica deve permitir o normal funcionamento do serviço de email.

3.3 Casos de Uso do Filtro de Email

A possibilidade de o filtro de email criar diferentes canais seguros, *MTA-to-MTA*, *MTA-to-User e User-to-MTA,* tem como objetivo incidir em dois tipos de sistemas. Num canal seguro *MTA-to-MTA*, os utilizadores de email não estão capacitados a utilizar sistemas criptográficos, entregando a terceiros essa responsabilidade (MTA).

Num canal seguro *MTA-to-User* e *User-to-MTA*, ou o utilizador destinatário ou remetente está capacitado a utilizar sistema criptográficos.

3.3.1 *MTA-to-MTA*

Assumindo a existência de um utilizador A pertencente a um certo domínio sediado num servidor de email denominado de remetente, e um utilizador B pertencente a um outro domínio sediado num servidor de email denominado destinatário, onde o utilizador A envia um email para o utilizador B.

A mensagem de email deve ser cifrada ou assinada pelo servidor de email remetente, e decifrada ou verificada a sua assinatura pelo servidor de email destinatário [Figura 3-1].

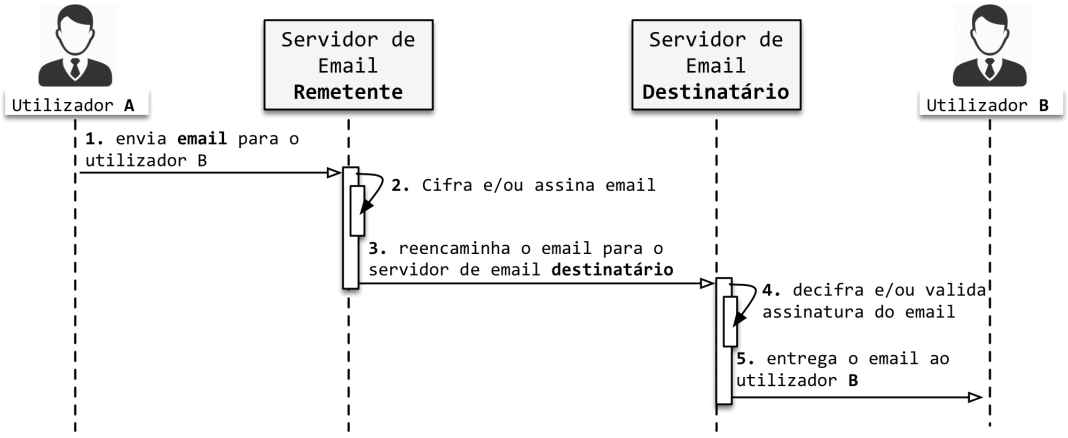

Figura 3-1 – Canal seguro *MTA-to-MTA*.

3.3.2 *MTA-to-User*

Assumindo a existência de um utilizador A pertencente a um certo domínio sediado num servidor de email denominado de remetente, e um utilizador B pertencente a um outro domínio sediado num servidor de email denominado destinatário, onde o utilizador A envia um email para o utilizador B.

A mensagem de email deve ser cifrada ou assinada pelo servidor de email remetente, e decifrada ou verificada a sua assinatura pelo utilizador destinatário [Figura 3-2].

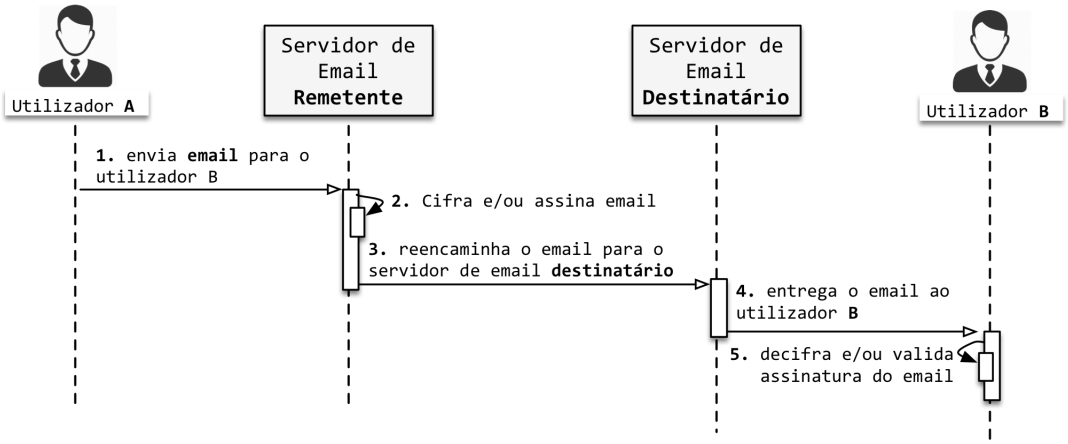

Figura 3-2 - Canal seguro *MTA-to-User*.

3.3.3 *User-to-MTA*

Assumindo a existência de um utilizador A pertencente a um certo domínio sediado num servidor de email denominado de remetente, e um utilizador B pertencente a um outro domínio sediado num servidor de email denominado destinatário, onde o utilizador A envia um email para o utilizador B.

A mensagem de email deve ser cifrada ou assinada pelo utilizador remetente (utilizador A), e decifrada ou validada a sua assinatura pelo servidor de email destinatário [Figura 3-3].

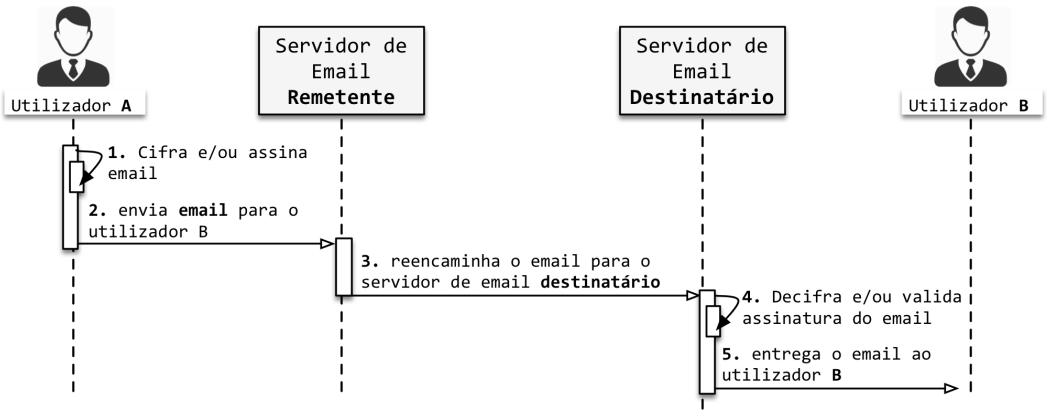

Figura 3-3 - Canal seguro *User-to-MTA*.

3.4 Casos de Uso da Plataforma de Gestão de Chaves

Como dito anteriormente, o filtro de email deve descobrir automaticamente as chaves públicas, através do DNS. O trabalho incluirá uma plataforma de gestão de chaves que possibilite fazer a conversão de chaves públicas ou certificados para RRs e a sua publicação em ficheiros de zona de um dado domínio no DNS. Deve fornecer as operações de publicar, atualizar, remover e obter chaves públicas ou certificados. Esta plataforma pode ser acedida pelo administrador do domínio onde o filtro de email está configurado.

Assumindo a existência de um administrador responsável pela administração de um determinado domínio. Acedendo à plataforma de gestão de chaves, podem-se suceder os seguintes casos de uso: publicar, atualizar, remover e obter uma chave pública ou certificado.

3.4.1 Publicar chave pública ou certificado

O admnistrador pretende publicar uma chave pública ou certificado [Figura 3-4].

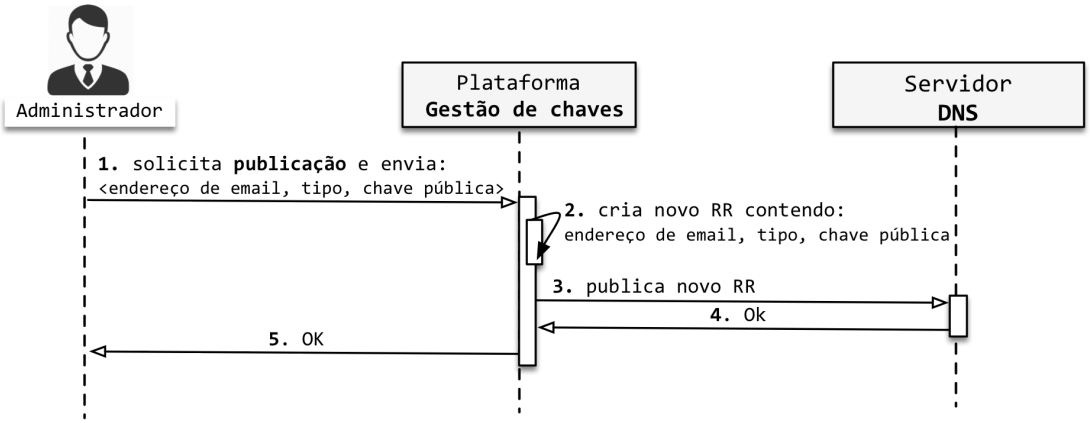

Figura 3-4 – Publicar chave pública ou certificado.

3.4.2 Atualizar chave pública ou certificado

O administrador pretende atualizar uma chave pública ou certificado [Figura 3-5].

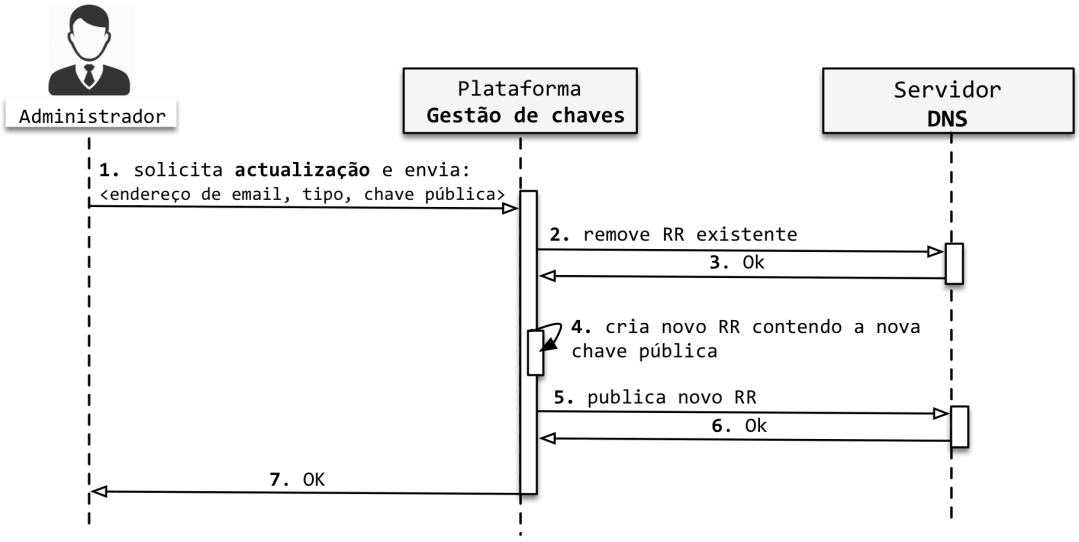

Figura 3-5 –Atualizar chave pública ou certificado.

3.4.3 Remover chave pública ou certificado

O administrador pretende remover uma chave pública ou certificado [Figura 3-6].

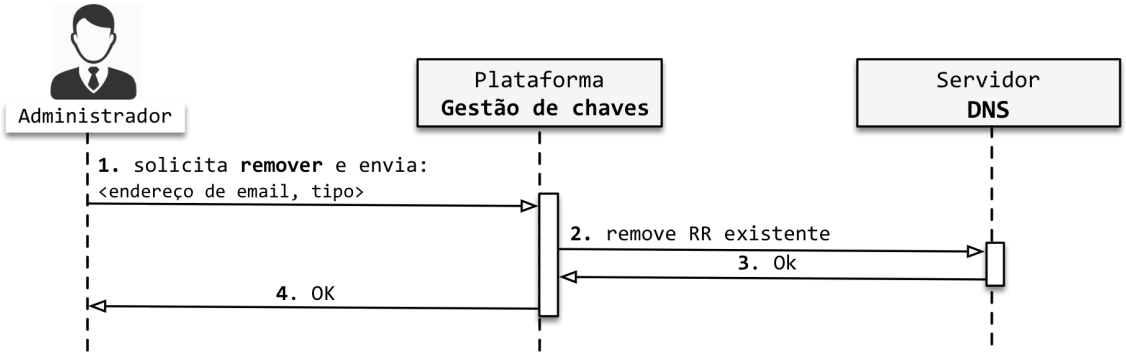

Figura 3-6 –Remover chave pública ou certificado.

3.4.4 Obter chave pública ou certificado

O admnistrador pretende obter uma chave pública ou certificado [Figura 3-6].

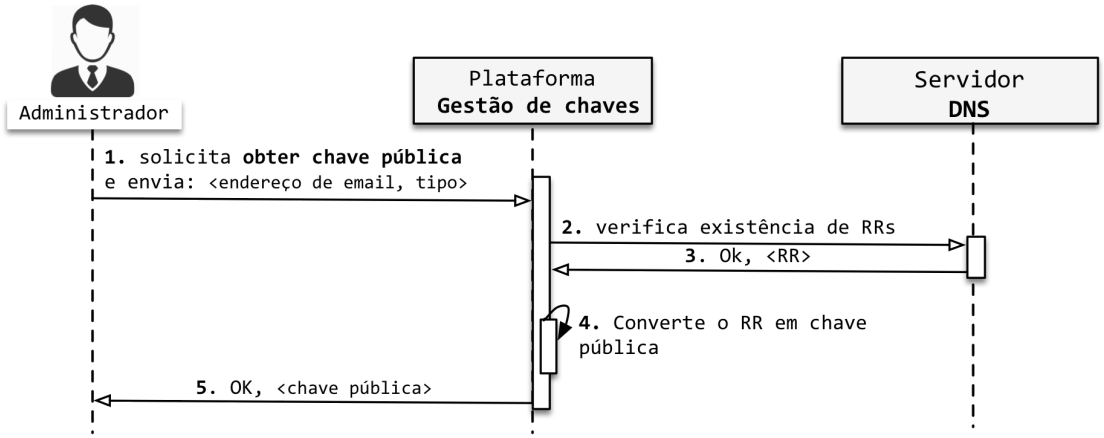

Figura 3-7 – Obter chave pública ou certificado

3.5 Arquitetura do Filtro de Email

Atendendo à definição dos requisitos funcionais e não funcionais, o filtro de email foi desenhado de forma a que esses requisitos fossem cumpridos. Como dito anteriormente, os filtros de email, funcionam junto dos servidores de email, captando eventos de cada um dos mais recebidos e enviados por este. Este filtro de email deve ser responsável pelas operações criptográficas realizadas sobre as mensagens de email, assim co-

mo as tomadas de decisão, ou seja, se deve ou não operar criptograficamente sobre uma determinada mensagem de email.

Além do filtro de email e de forma aos requisitos anteriormente definidos serem cumpridos, em cada servidor de email deverão existir os seguintes componentes, com as respetivas funcionalidades [Figura 3-8]:

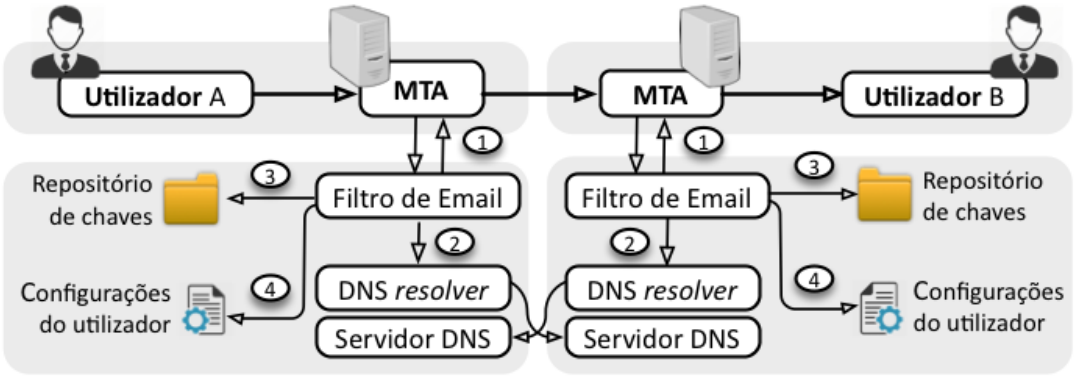

Figura 3-8 – Arquitetura do Filtro de Email. **(1)** na chegada de uma mensagem de email, esta é enviada para o filtro de email. **(2)** o filtro de email através do *DNS Resolver* consulta a chave pública ou certificado. **(3)** o filtro de email consulta a existência de chaves privadas. **(4)** o filtro consulta a ação que vai tomar para aquele destinatário (operar ou não).

- Repositório de chaves: armazena um *KeyRing OpenPGP* com as chaves privadas e públicas de utilizadores, assim como certificados S/MIME e respetivas chaves privadas;
- DNS *Resolver:* resolve pedidos de chaves públicas ou certificados de utilizadores de outros domínios;
- Servidor DNS: armazena as chaves públicas e certificados em ficheiros de zona dos domínios administrados nesse servidor, tornando-os públicos para filtros de email e utilizadores;
- Ficheiros de configurações do utilizador: dão informação sobre se para um determinado utilizador, o filtro de email pode operar criptograficamente pelo utilizador, ou seja, deve definir se o canal seguro vai ser *MTA-to-MTA*, *MTA-to-User* e *User-to-User*.

Postos estes componentes essenciais, é preciso definir como é realizada a tomada de decisão do filtro de email a nível de operações criptográficas.

3.5.1 Operações criptográficas

Como definido anteriormente, o filtro vai agir criptograficamente, dependendo das respostas que o DNS *Resolver*, o repositório de chaves, e as configurações do utilizador retornarem**.**

Sendo definido um cenário, onde um utilizador A envia uma mensagem de email para um utilizador B, onde o utilizador A pertencente ao servidor de email de domínio A, e o utilizador B pertencente. Deste modo o servidor de email de domínio A é designado de remetente, e o servidor de email de domínio B é designado de destinatário.

O filtro de email vai agir criptograficamente da seguinte forma no servidor de email remetente [Tabela 3-1]:

Servidor de Email Remetente			**Operação criptográfica**
DNS *Resolver*	**Repositório de Chaves**	**Configurações do utilizador**	
O **utilizador B** tem chave pública no DNS ?	O **utilizador A** tem chave privada no repositório de chaves ?	O filtro de email pode operar criptograficamente pelo **utilizador A**?	
Sim	Não	Sim	Mensagem é **cifrada**
Não	Sim	Sim	Mensagem é **assinada**
Sim	Sim	Sim	Mensagem é **assinada** e **cifrada**
Não	Não	Sim	Não é realizada qualquer operação criptográfica

Tabela 3-1 - Operações criptográficas realizadas pelo servidor de email remetente

De notar para que o filtro de email possa operar criptograficamente para um utilizador, as configurações do utilizador têm de dar permissão, levando a que as operações criptográficas não ocorram devido:

- ao DNS *Resolver* não encontrar a chave pública do utilizador B e o repositório de chaves não conter a chave privada do utilizador A, ou
- às configurações do utilizador não darem permissão ao filtro de email para operar criptograficamente.

O filtro de email vai agir criptograficamente da seguinte forma no servidor de email destinatário [Tabela 3-2]:

Servidor de Email Destinatário			Operação criptográfica
DNS *Resolver*	**Repositório de Chaves**	**Configurações do utilizador**	
O **utilizador A** tem chave pública no DNS ?	O **utilizador B** tem chave privada no repositório de chaves ?	O filtro de email pode operar criptograficamente pelo **utilizador A**?	
Não	Sim	Sim	Mensagem é **decifrada**
Sim	Não	Sim	Assinatura da mensagem é **verificada**
Sim	Sim	Sim	Mensagem é **decifrada** e a assinatura da mensagem é **verificada**
Não	Não	Sim	Não é realizada qualquer operação criptográfica

Tabela 3-2 - Operações criptográficas realizadas pelo Servidor de email destinatário

De notar para que o filtro de email possa operar criptograficamente para um utilizador, as configurações do utilizador têm de dar permissão, levando a que as operações criptográficas não ocorram devido:

- ao DNS *Resolver* não encontrar a chave pública do utilizador A e o repositório de chaves não conter a chave privada do utilizador B, ou
- às configurações do utilizador não darem permissão ao filtro de email para operar criptograficamente.

Os ficheiros de configurações de utilizador definem se uma mensagem de email pode ser alvo de uma operação criptográfica por parte do filtro de email. Isto leva à criação de diferentes canais seguros na sua longitude. Num canal de longitude mais curta, *MTA-to-MTA*, os ficheiros de configurações de utilizador dão informação ao filtro de email para operar criptograficamente nos servidores de email remetente e destinatário.

A possibilidade de obter um canal seguro *MTA-to-User* ou *User-to-MTA*, oferece maior segurança, e uma maior longitude no canal seguro. Num canal *MTA-to-User*, os ficheiros de configurações de utilizador dão informação ao filtro de email para operar criptograficamente no servidor de email remetente (cifrando ou assinando a mensagem), não dando informação para operar criptograficamente no servidor de email destinatário, (seguindo cifrada ou assinada até ao utilizador destinatário). Num canal *User-to-MTA*, os ficheiros de configurações de utilizador dão informação ao filtro de email para não operar criptograficamente no servidor de email remetente (a mensagem vem cifrada ou assinada do utilizador remetente), dando informação para operar criptograficamente no servidor de email destinatário, (decifrando ou validando a assinatura). Estes dois canais

só podem ser concretizados se os utilizadores estiverem capacitados a realizar operações criptográficas.

A Figura 3-9 exemplifica o envio de uma mensagem de email para vários utilizadores, onde para o utilizador `B_1` e `B_3` é concretizado um canal seguro *MTA-to-MTA* e para o utilizador `B_2` é concretizado um canal seguro *MTA-to-User*. O MTA destinatário é que toma a decisão quanto à conclusão do canal seguro, na consulta das configurações do utilizador (3), DNS *resolver* (4), e repositório de chaves (4).

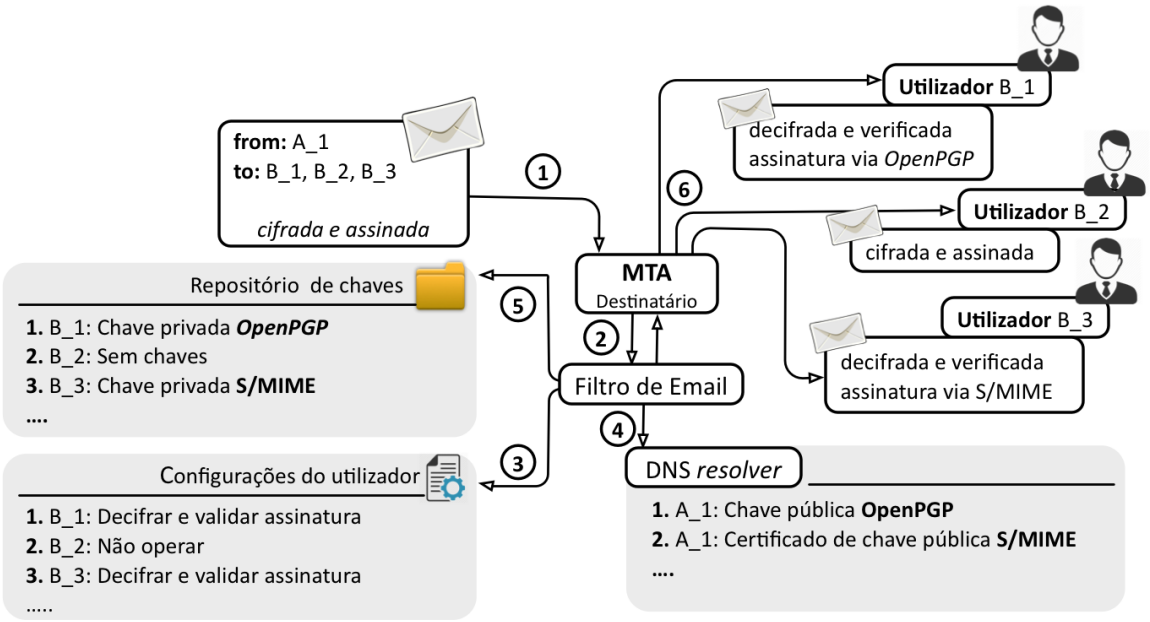

Figura 3-9 – **Exemplo de um canal seguro *MTA-to-MTA* e *MTA-to-User* (1)** chega, ao MTA destinatário, uma mensagem de email cifrada e assinada do utilizador `A_1` para os utilizadores `B_1`, `B_2` e `B_3` **(2)** O MTA destinatário reencaminha a mensagem para o filtro de email **(3)** são verificadas as configurações do utilizador para os destinatários **(4)** são consultadas as chaves públicas e certificados de chave pública ao DNS *resolver* **(5)** são consultadas as chaves privadas dos destinatários **(6)** o utilizador `B_1` recebe a mensagem de email via canal seguro *MTA-to-MTA*, o utilizador `B_2` via canal seguro *MTA-to-User* e o utilizador `B_3` via canal seguro *MTA-to-MTA*.

Além das diferentes longitudes do canal seguro, uma mensagem para múltiplos utilizadores deve possibilitar a operação com os dois protocolos, *OpenPGP* e S/MIME para os vários destinatários e ainda, diferentes combinações de operações criptográficas [Tabela 3-1 e 3-2] sobre os dois protocolos utilizado para ambos.

3.5.2 Repositório de chaves

Como definido nos requisitos, o filtro de email deve ser capaz de descobrir chaves públicas utilizadas pelos protocolos *OpenPGP* e S/MIME. Como tal, deve suportar as operações descritas anteriormente para cada um destes tipos.

Além das chaves públicas e certificados estarem publicados no DNS, é necessário a existência de um repositório de chaves, essencialmente privadas, nos servidores de email. O repositório de chaves (colocado em cada servidor de email) deve conter os seguintes componentes e a arquitetura [Figura 3-10]:

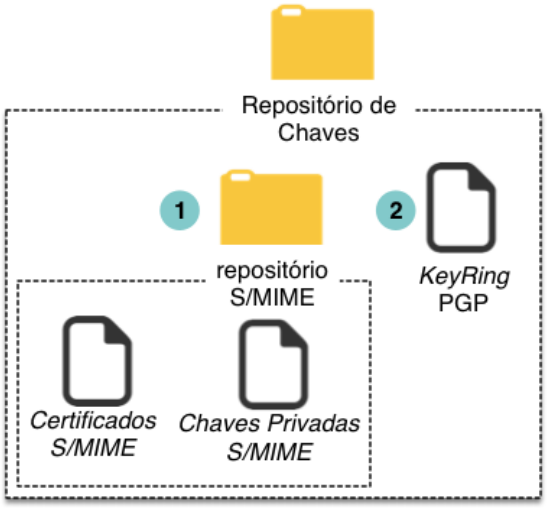

Figura 3-10 –Arquitetura do repositório de chaves

1. Um repositório para certificados S/MIME e respetivas chaves privadas;
2. Um *KeyRing OpenPGP* responsável por armazenar as chaves públicas e privadas *OpenPGP*.

O caminho deste repositório de chaves deve ser fornecido ao filtro de email no inicio da sua execução. A consulta ao repositório realizada pelo filtro de email assume a arquitetura descrita.

3.5.3 Reinjeção de mensagens de email

Os filtros de email processam eventos das mensagens emails á saída dos servidores de email que enviam para outros servidores de email, e à chegada na receção de uma mensagem de email vinda de outro servidor de email, levando a que, uma mensagem de

email para múltiplos utilizadores seja processada uma única vez, sendo depois distribuída para todos utilizadores. Atendendo ao facto de os utilizadores de um mesmo domínio poderem ter diferentes tipos de chaves (S/MIME e *OpenPGP*) diferentes tipos de permissões, resultando em diferentes canais seguros (*MTA-to-MTA*, *MTA-to-User e User-to-MTA*), leva à necessidade de processar individualmente, para cada utilizador, uma mensagem de email que é destinada a múltiplos utilizadores. Para esse fim, foi criado um **mecanismo de reinjeção de mensagens de email**.

Este mecanismo consiste em, numa determinada execução o filtro de email para uma determinada mensagem de email, efetuar um envio para o mesmo servidor de email onde o filtro de email está a ser executado, simulando deste modo o envio por parte um utilizador [Figura 3-11].

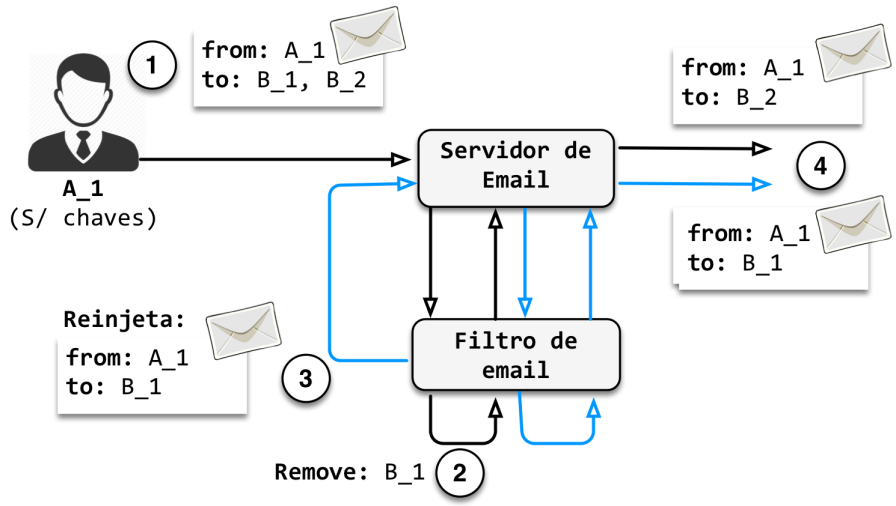

Figura 3-11 - Mecanismo de reinjeção de mensagens de email

Considerando o envio de uma mensagem de email do utilizador `A_1`, para os utilizadores `B_1` e `B_2`, onde o utilizador remetente não possui nenhum tipo de chaves, e os destinatários possuem chaves, *OpenPGP* e S/MIME respetivamente.

Neste caso, o mecanismo de reinjeção resulta nos seguintes passos [Figura 3-11]:

1. Envio da mensagem de email para múltiplos utilizadores, com diferentes tipos de chaves.
2. Quando os eventos da mensagem são enviados para o filtro de email, um dos destinatários é removido da mensagem (neste caso, o `B_1` é removido). O processamento realizado pelo filtro passa a ser igual ao do envio de `A_1` para `B_2`, ou seja, realizando uma operação criptográfica com S/MIME.

3. Na mesma execução de `A_1` para `B_2`, é reinjetada uma nova mensagem de email para o mesmo servidor de email do `A_1` para o `B_1`, sendo este processado também de forma individual, ou seja, realizando uma operação criptográfica com *OpenPGP*.
4. Seguem duas mensagens de email individuais, sendo uma operada sobre OpenPGP para o utilizador `B_1`, e outra operada sobre S/MIME para o utilizador `B_2`.

Deste modo é possível tratar individualmente para cada utilizador, uma mensagem que se destina a múltiplos utilizadores.

Como consequência, este mecanismo vai aumentar o número de processamentos do filtro de email [Figura 3-11], sendo relevante perceber esse aumento. Esse aumento é apresentado na seguinte tabela [Tabela 3-3] resultando nas seguintes funções [Figura 3-12].

Destinatários			Número de processamentos do filtro de email		
n	***m***	***p***	**MTA Remetente**	**MTA Destinatário**	**Total** MTA Remetente + MTA Destinatário
0	0	1	1	1	**2**
0	1	0	1	1	**2**
1	0	0	1	1	**2**
0	1	1	2	2	**4**
1	1	1	3	3	**6**
1	1	2	3	3	**6**
1	2	2	3	4	**7**
2	2	2	3	5	**8**
....	....	....	....	....	....

p	Número de utilizadores sem chaves
m	Número de utilizadores com chaves S/MIME
n	Número de utilizadores com chaves *Open*PGP

Tabela 3-3 – Número de execuções do filtro de email em função do tipo e número de destinatários

MTA Remetente

3, se $p > 0$ & $m > 0$ & $n > 0$
2, se $p = 0$ & $m > 0$ & $n > 0$ || $p > 0$ & $m = 0$ & $n > 0$ || $p > 0$ & $m > 0$ & $n = 0$
1, se $p = 0$ & $n = 0$ & $m > 0$ || $p > 0$ & $n = 0$ & $m = 0$ || $p = 0$ & $n > 0$ & $m = 0$

MTA Destinatário

$n + m + 1$, se $p > 0$
$n + m$, se $p = 0$

Total
(MTA Remetente + MTA Destinatário)

$n + m + 4$, se p > 0 & m > 0 & n > 0
$n + m + 3$, se $p > 0$ & $m = 0$ & $n > 0$ || $p > 0$ & $m > 0$ & $n = 0$
$n + m + 2$, se $p = 0$ & $m > 0$ & $n > 0$ || $p > 0$ & $n = 0$ & $m = 0$
$n + m + 2$, se $p = 0$ & $n = 0$ & $m > 0$ || $p = 0$ & $n > 0$ & $m = 0$

Figura 3-12 – Funções para número de execuções do filtro de email.

De notar que as funções no total, crescem em função do n e m, ou seja, se o numero de utilizadores com chaves *OpenPGP* e S/MIME aumentarem, o numero total de execuções do filtro de email também vai aumentar. De referir, que para utilizadores sem chaves, não existe aumento, pois a variável de p não entra nas funções, sendo que não é necessário reinjeção visto não haver operação para utilizadores sem chaves.

Por exemplo, um envio para de um utilizador para 30 utilizadores, onde 5 (p) não têm chaves, 14 têm chaves S/MIME (m), e 11 têm chaves *Open*PGP (n), resulta num número total de execuções de 29 execuções ($n + m + 4$, para $m = 14$ e $n = 11$).

É um aumento bastante significativo, de 2 execuções para 29 execuções, mas tendo em conta que grande maioria dos envios de emails não são um grande número de utilizadores, e ainda, que não é suposto uma organização com este sistema, os seus utilizares utilizarem dois tipos de chaves, este aumento acaba por não ter grande preocupação para o sistema.

3.5.4 Prioridade no tipo de chave

Com visto anteriormente, os utilizadores podem ter diferentes tipos de chaves (S/MIME e *OpenPGP*), e o filtro de email deve operar para esses mesmos tipos. Quando por exemplo, um destinatário possui duas chaves, uma S/MIME e uma *OpenPGP* o filtro de email precisa tomar uma decisão quanto à chave que vai utilizar. Foi por isso definido que o filtro de email é inicializado com uma prioridade no tipo da chave, sendo esta prioridade fator de decisão para os seguintes cenários:

- O destinatário possui chave S/MIME e o remetente possui chaves *OpenPGP:* se prioridade for S/MIME, a operação feita é com a chave S/MIME, para as condições verificadas.
- O destinatário possui duas publicações, uma S/MIME e outra *OpenPGP*: se a prioridade for *OpenPGP*, a operação feita é com a chave *OpenPGP*.

Estes cenários assumem-se pouco prováveis, tendo em conta que organizações que façam utilização do filtro de email, escolheriam apenas um dos protocolos para os seus utilizadores. Apenas se adoção deste mecanismo fosse global (por todos os fornecedores de serviço de email) esta prioridade seria essencial, para o correto funcionamento do filtro de email.

3.6 Arquitetura da Plataforma de Gestão de Chaves

Como dito anteriormente, para possibilitar o filtro de email encontrar as chaves públicas (através do DNS) o trabalho incluirá uma plataforma de gestão de chaves que possibilite fazer a conversão de chaves públicas ou certificados para RRs, a sua publicação nos ficheiros de zona do respetivo domínio no DNS, assim como atualizar, revogar e obter alguma chave pública ou certificado. Esta plataforma deve administrada pelo administrador do domínio onde o filtro de email esta configurado.

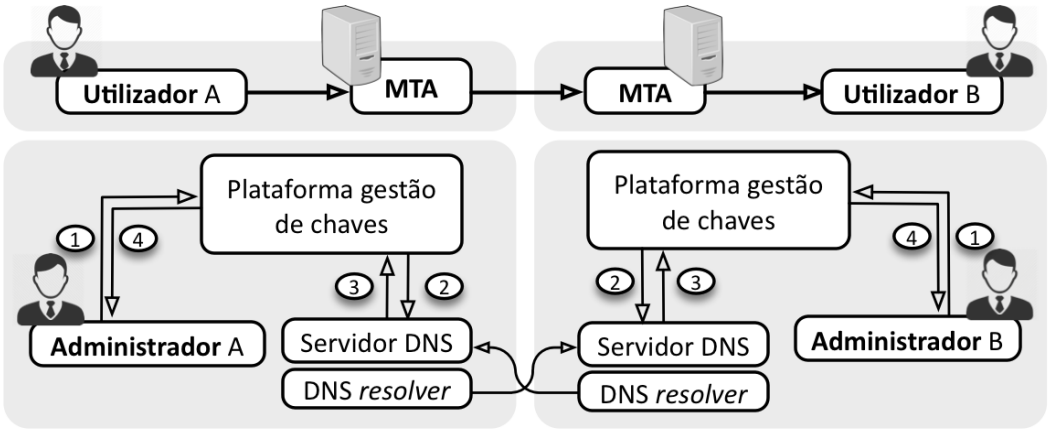

Figura 3-13 – Arquitetura da plataforma de gestão de chaves. **(1)** administrador faz um pedido à plataforma; (**2)** a plataforma realiza pedido de consulta ou actualização no servidor DNS; **(3)** O servidor DNS responde para o pedido solicitado; **(4)** a resposta é tratada, encapsulada numa resposta e entregue ao administrador

Em cada servidor de email serão incluídos os seguintes componentes, com a respectiva funcionalidade [Figura 3-13]:

- Plataforma de Gestão de Chaves: plataforma na qual o administrador pode aceder para realizar operações de gestão de chaves.

- Servidor de DNS: contendo os ficheiros de zona onde são publicadas as chaves públicas e certificados, administrado no servidor de email.

As interações entre administrador e a plataforma de Gestão de Chaves são do tipo pedido/resposta, onde as operações são feitas remotamente no servidor de email, através de um serviço *RESTful*[3], permitindo operações do tipo de GET, PUT, POST e DELETE, associadas às operações de obter, publicar, atualizar e remover chaves públicas *OpenPGP* e certificados S/MIME.

3.7 Conclusão

Ao longo deste capitulo foram apresentados os requisitos funcionais, não funcionais, assim como os requisitos específicos do projeto, tanto para o filtro de email como para a plataforma de gestão de chaves. Posteriormente, foram apresentados os casos de uso descrevendo as interações do utilizador com as componentes do projeto, terminando com o desenho do sistema.

Os filtros de email operam pelos utilizadores, realizando operações criptográficas que irão gerar uma carga de processamento adicional ao servidor de email. É necessário validar que o processamento adicionado pelo filtro de email não compromete o normal funcionamento do serviço de email. É também relevante perceber em que cenários pode este sistema ser aplicado.

Para provar que os requisitos funcionais e não funcionais são cumpridos, no capitulo **4,** serão apresentados os detalhes de implementação de cada um dos componentes, os resultados da avaliação experimental, terminando com a sua análise e respetivas conclusões.

[3] `http://docs.oracle.com/cd/E41633_01/pt853pbh1/eng/pt/tibr/concept_UnderstandingRESTServiceOperations.html`

Capítulo 4 Implementação e Avaliação Experimental

Neste capítulo, são descritos os detalhes da implementação do projeto, nomeadamente do filtro de email e da plataforma de gestão de chaves. São apresentados os principais desafios enfrentados, bem como a justificação das decisões tomadas ao longo do seu desenvolvimento. Será também descrito todo o ambiente criado para o desenvolvimento e testes de todo o projeto, terminando com uma avaliação dos resultados obtidos.

4.1 Linguagem de programação

Foram consideradas as seguintes opções em termos de linguagem de programação: *Java*, *C*, *Python* e *Perl*. A linguagem escolhida foi o *Python* devido ao facto de existirem algumas iniciativas relacionadas já desenvolvidas em *Python* [29]. As linguagens de C e *Perl* não foram consideras, pois ambas podiam comprometer os objetivo traçados, no que toca ao tempo de aprendizagem para *Perl*, e à complexidade de *C*. À semelhança do filtro de email, a linguagem de programação escolhida para a plataforma de gestão de chaves foi *Python*. Poderia ter sido implementada em *Java*, mas visto que a implementação do filtro de email foi inicialmente planeada em *Python*, optou-se por utilizar a mesma linguagem de programação. Assim, a utilização das mesmas bibliotecas, para as operações em comum, tanto no filtro de email como na plataforma de gestão de chaves, estão em conformidade.

4.2 Implementação do filtro de email

Nesta secção serão descritos alguns detalhes de implementação do filtro de email, assim como alguns pontos chave para o funcionamento do filtro de email (Operações criptográficas via *OpenPGP* e S/MIME, campos de controlo, reinjeção de mensagens de email, *XCLIENT*).

4.2.1 Descoberta de Chaves Criptográficas

Para o filtro de email descobrir as chaves públicas *OpenPGP* e certificados S/MIME, é necessário o DNS *Resolver* conter a lista dos Servidores DNS dos domínios que têm chaves e certificados publicados [Figura 4-1]. Assim, a descoberta de uma chave pública ou certificado ocorre da seguinte forma:

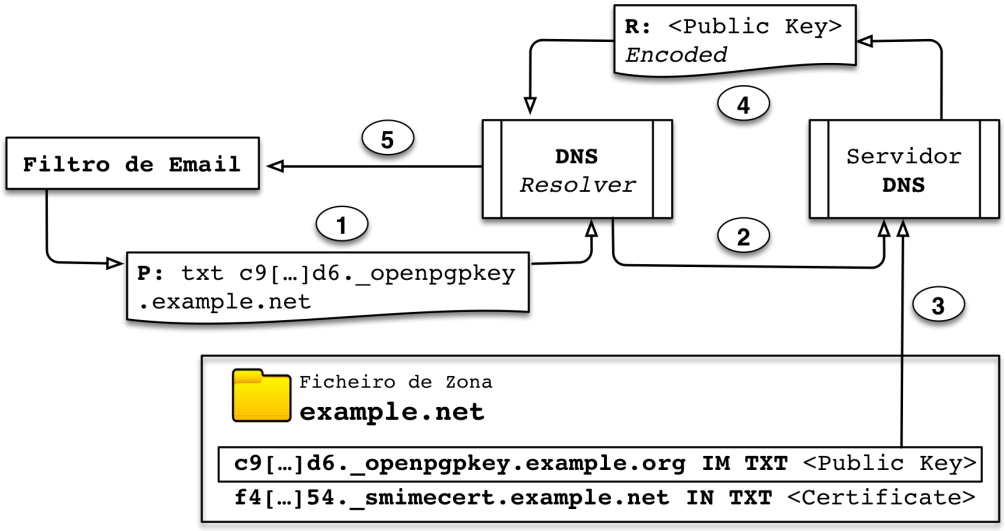

Figura 4-1 - Descoberta de chaves públicas *OpenPGP* e certificados S/MIME

1. O filtro de email prepara o **pedido** ao DNS *Resolver* [29, 30], na Figura 4-1 um pedido de uma chave pública *OpenPGP*, do utilizador `user1@example.net`.
2. O DNS *Resolver* procura o servidor DNS responsável por gerir o nome de domínio `example.net`, reencaminhando-lhe o pedido do filtro de email.
3. O Servidor DNS, consulta o ficheiro de zona do nome de domínio `example.net`, verificando a existência do registo de recurso (RR).
4. O conteúdo do RR, uma chave pública *OpenPGP* codificada em base 64, é entregue o DNS *Resolver*.
5. O DNS *Resolver* entrega o conteúdo do RR ao filtro de email, que deve descodificar a chave pública *OpenPGP*, e importa-la para um *KeyRing*.

Adicionados os servidores DNS que contêm chaves públicas *OpenPGP* e certificados S/MIME ao DNS *Resolver* do servidor de email onde se encontra em execução o filtro de email, é possível realizar a descoberta de chaves públicas e certificados

S/MIME quer de destinatários (para cifra), quer de remetentes (para verificar assinaturas).

De forma ao filtro de email descobrir as chaves privadas *OpenPGP* e S/MIME, é necessário a existência de um repositório de chaves. Este repositório deve ter a hierarquia desenhada no Capitulo 3, sendo que os caminhos e nomes das chaves devem ser os seguintes [Figura 4-2]:

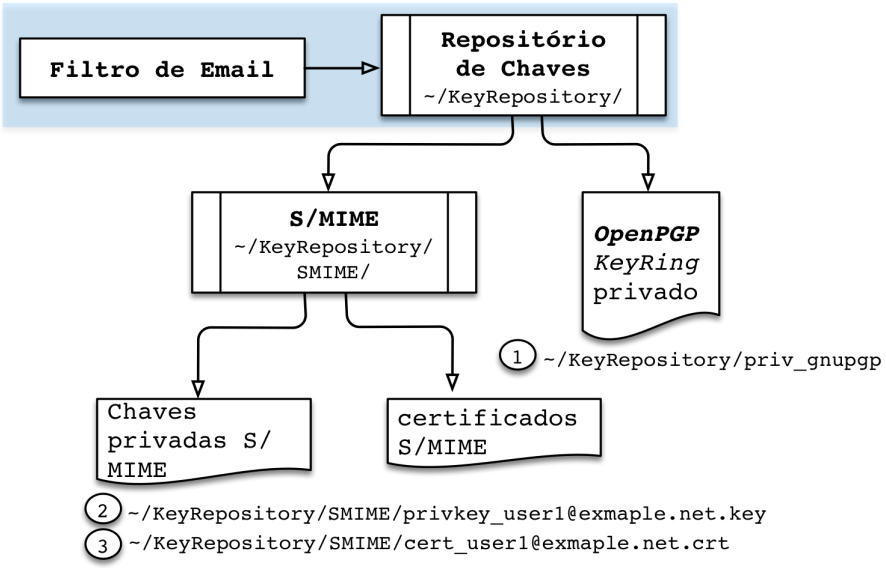

Figura 4-2 - Descoberta de chaves privadas *OpenPGP* e S/MIME

1. `~/KeyRepository/priv_gnupgp`: o *KeyRing* de chaves privadas *OpenPGP* deve estar dentro do repositório de chaves.
2. `~/KeyRepository/SMIME/cert_X.cert`: os certificados de chave pública S/MIME devem estar dentro do repositório de chaves, numa pasta designada `SMIME`. O nome do certificado deve ser `cert_X.cert`, sendo o `X` o endereço de email do utilizador.
3. `~/KeyRepository/SMIME/privkey_X.key`: as chaves privadas S/MIME devem estar dentro do repositório de chaves, numa pasta designada `SMIME`. O nome da chave privada deve ser `privkey_X.key`, sendo o `X` o endereço de email do utilizador.

Adicionado esta hierarquia no repositório de chaves, e adicionando também as respetivas chaves privadas dos utilizadores, é possível realizar a descoberta de chaves privadas S/MIME e *OpenPGP* quer de destinatários (para decifrar), quer de remetentes (para assinar).

Este mecanismo de descoberta vai adicionar o automatismo e transparência definidos nos requisitos não funcionais, pois o processo é feito sem a interação do utilizador.

4.2.2 Operações Criptográficas via *OpenPGP*

As operações criptográficas via *OpenPGP* no filtro de email, assumem a existência de chaves públicas de utilizadores em ficheiros de zona no DNS, assim como um *KeyRing* privado, sediado no servidor de email responsável por gerir as chaves públicas e privadas dos utilizadores de domínios geridos nesse servidor de email.

As operações criptográficas, nomeadamente cifra e assinatura exigem ações diferentes sobre a mensagem de email, devido à mensagem poder ser em **texto simples** ou composta por **múltiplas partes**. Pretende-se mostrar os algoritmos que permitem as operações criptográficas para estes dois tipos de mensagem de email.

Algoritmo 1.1. Mensagem de email cifrada via *OpenPGP*

```
 0: body ← ... corpo da mensagem de email
 1: have_pgp_users ← ... indicador da existência de destinatários OpenPGP
 2: gnupg_pub ← ... keyring de chaves públicas de destinatários OpenPGP
 3: is_multipart ← ... indicador do tipo de mensagem de email
 4: have_pgp_priority ← ... indicador da prioridade do servidor de email para OpenPGP
 5: is_pgp_encrypted_message ← ... indicador do estado criptográfico da mensagem para
 6:                                OpenPGP
 7:
 8: if have_pgp_users == True and is_pgp_encrypted_message == False and
 9:                                    have_pgp_priority == True then:
10:    if is_multipart == False then:
11:       encrypted_body ← pgp_encrypt(gnupg_pub, body)
12:       replace_body(encrypted_body)
13:    else:
14:       to_encrypt ← construct_multipart_body(body, header)
15:       encrypted_body ← pgp_encrypt(gnupg_pub, to_encrypt)
16:       body_to_replace,content_type ← construct_replace_body(encrypted_body)
17:       replace_body(body_to_replace)
18:       change_header(content_type)
19:    end if
20:    add_control_headers()
20:    accept()
21: end if
```

As operações de cifra *OpenPGP* só ocorrem nos servidores de email remetentes, e como tal, para ocorrerem, é necessário o filtro de email (*i*) encontrar as chaves públicas *OpenPGP* dos destinatários (via DNS), (*ii*) a mensagem não se encontrar cifrada e (*iii*) o

filtro de email estiver configurado para operar criptograficamente pelo utilizador sobre *OpenPGP* (linha 8 e 9).

Se a mensagem de email se encontrar em texto simples, o corpo da mensagem é cifrado integralmente (linha 11 e 12), e adicionados os campos de controlo ao cabeçalho da mensagem (linha 20). No entanto, caso a mensagem de email seja composta por múltiplas partes, o processo de cifra torna-se mais complexo, devido à necessidade de construir um novo corpo da mensagem, onde a mensagem a ser cifrada será inserida [Figura 4-3].

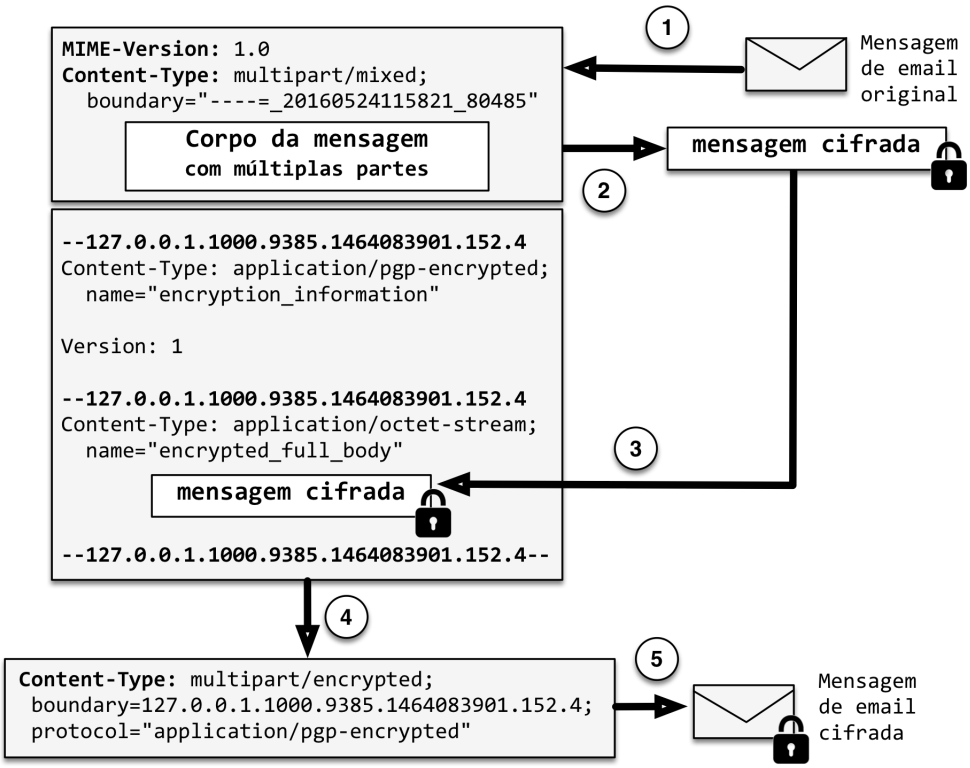

Figura 4-3 – Mensagem de email com múltiplas partes cifrada via *OpenPGP*

O processo de cifra de uma mensagem de email com múltiplas partes resulta nos seguintes passos [Figura 4-3]:

1. Extração dos campos *Content-Type* e *MIME-Version*, anexando-os ao corpo da mensagem (linha 14);
2. O conteúdo gerado em **(1)** é cifrado via *OpenPGP* (linha 15);
3. O conteúdo cifrado é adicionado a um novo corpo de mensagem composto por duas partes (versão e mensagem cifrada), que cumpre as especificações indicadas no RFC3156[4] (linha 16);

[4] `https://tools.ietf.org/html/rfc3156`

4. É gerado um novo *Content-Type,* substituindo o atual no cabeçalho da mensagem de email. Este campo contem o delimitador para o novo corpo da mensagem, a **negrito** na Figura 4-3 (linha 18);
5. A mensagem segue assim até ao servidor de email destinatário cifrada.

Após a chegada da mensagem cifrada ao servidor de email destinatário, esta deve ser decifrada.

Algoritmo 1.2. Mensagem de email decifrada via *OpenPGP*

```
 0: encrypted_body ← ... corpo da mensagem de email cifrado
 1: have_pgp_users ← ... indicador da existência de destinatários OpenPGP
 2: gnupg_priv ← ... keyring de chaves privadas de destinatários OpenPGP
 3: is_multipart ← ... indicador do tipo de mensagem de email
 4: have_pgp_priority ← ... indicador da prioridade do servidor de email para OpenPGP
 5: is_pgp_encrypted_message ← ... indicador do estado criptográfico da mensagem para
 6:                                OpenPGP
 7:
 8: if have_pgp_users == True and is_pgp_encrypted_message == True then:
 9:    if is_multipart == False then:
10:  body ← pgp_decrypt(gnupg_priv, encrypted_body)
11:       replace_body(body)
12:    else:
13:  body_to_decrypt, content_type ← get_body_to_decrypt(encrypted_body)
14:  body ← pgp_decrypt(gnupg_priv, body_to_decrypt)
15:       change_header(content_type)
16:       replace_body(body)
17:    end if
18:    add_control_headers()
19:    accept()
20: end if
```

As operações de decifra via *OpenPGP* só ocorrem nos servidores de email destinatários, e como tal, para ocorrerem, é necessário o filtro de email (*i*) encontrar a chave privada do destinatário no *KeyRing* de chaves privadas *OpenPGP* (no repositório local do servidor de email) e (*ii*) verificar que a mensagem se encontra cifrada (linha 8).

Se a mensagem de email cifrada estiver em texto simples, o corpo da mensagem é decifrado integralmente (linha 10 e 11), e adicionados os campos de controlo ao cabeçalho da mensagem (linha 18). Mas se a mensagem de email seja composta por múltiplas partes, o processo de decifra torna-se mais complexo [Figura 4-4]:

1. Extração dos campos *Content-Type* e *MIME-Version* do cabeçalho, e do corpo da mensagem cifrado (linha 13);
2. O corpo da mensagem extraído em **(1)** é decifrado via *OpenPGP* (linha 14);
3. O *Content-Type* e *MIME-Version* extraídos em (**2)**, substituem os atuais;

4. O corpo da mensagem decifrado substitui o corpo da mensagem cifrado.

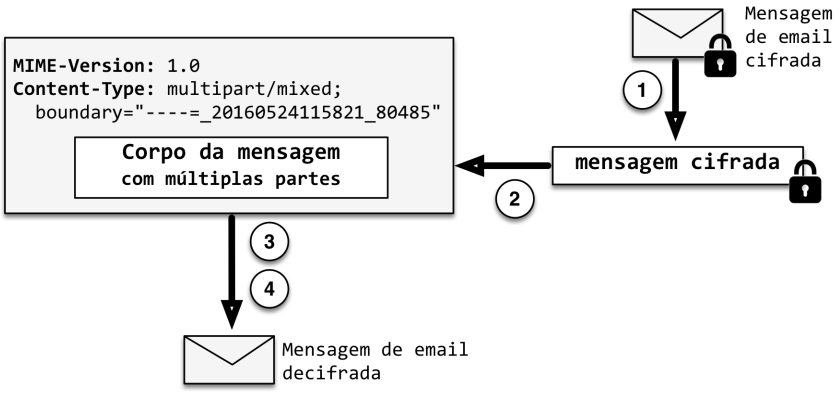

Figura 4-4 – Mensagem de email com múltiplas partes decifrada via *OpenPGP*

As operações de assinatura via *OpenPGP* no filtro de email ocorrem nos servidores de email remetentes, e como tal, para ocorrerem, é necessário o filtro de email (*i*) encontrar a chave privada do remetente no *KeyRing* de chaves privadas *OpenPGP* (no repositório de chaves do servidor) (*ii*) a mensagem não se encontrar cifrada nem assinada e (*iii*) estar configurado para dar prioridade a operações criptográficas via *OpenPGP* (linha 9 e 10).

Algoritmo 1.3. Mensagem de email assinada via *OpenPGP*

```
 0: body ← ... corpo da mensagem de email
 1: is_multipart ← ... indicador do tipo de mensagem de email
 2: sender_have_pgp_priv_key ← ... indicador da existência de chave privada OpenPGP do
 3:                                remetente
 4: gnupg_priv ← ... keyring de chaves privadas de destinatários OpenPGP
 5: have_pgp_priority ← ... indicador da prioridade do servidor de email para OpenPGP
 6: is_pgp_encrypted_message ← ... indicador do estado criptográfico da mensagem para
 7:                                OpenPGP
 8:
 9: if sender_have_pgp_priv_key == True and is_pgp_encrypted_message == False
10:                                and have_pgp_priority == True then:
11:    if is_multipart == True then:
12:       body ← Content_type + body
13:    else:
14:       body ← Content_type + Content_Transfer_Encoding + body
15:    end if
16:    signature ← pgp_sign(gnupg_priv, body)
17:    signed_body, content_type ← construct_signed_body(header,body,signature)
18:    change_header(content_type)
19:    replace_body(signed_body)
20:    add_control_headers()
21:    accept()
22: end if
```

Se a mensagem de email estiver em texto simples, o corpo da mensagem assim como os campos do cabeçalho de *Content-Type* e *Content-Transfer-Encoding* são extraídos (linha 14), e este conjunto é assinado [Figura 4-5] (linha 16). Esta assinatura é inserida num novo corpo de mensagem, contendo todo o corpo da mensagem original, gerando um corpo de mensagem assinado (linha 17). O campo *Content-Type* é substituído por um novo campo correspondente à nova mensagem de email (linha 18). O corpo da mensagem assinado substitui o antigo (linha 19). Por fim são adicionados os campos de controlo ao cabeçalho da mensagem (linha 20).

No caso de a mensagem de email ser composta por múltiplas partes, o processo é literalmente diferente. Na extração inicial, apenas é extraído o corpo da mensagem e o campo do cabeçalho de *Content-Type*. Todo o restante processo ocorre de igual forma [Figura 4-5].

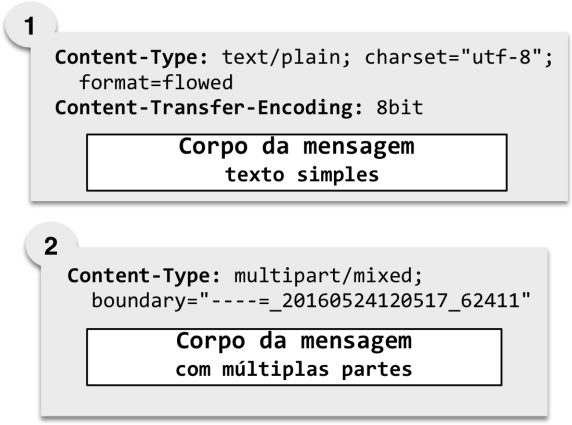

Figura 4-5 – Conteúdo da mensagem a ser assinado via *OpenPGP*.
(1) mensagem de email em texto simples; **(2)** mensagem de email com múltiplas partes.

Após a chegada da mensagem assinada ao servidor de email destinatário, a sua assinatura deve ser verificada e validada.

Algoritmo 1.4. Verificação e validação de uma mensagem de email assinada via *OpenPGP*

0: *signed_body* ← ... corpo da mensagem de email
1: *sender_have_pgp_pub_key* ← ... indicador da existência de chave publica *OpenPGP* do
2: remetente
3: *gnupg_pub* ← ... *keyring* de chaves públicas de destinatários *OpenPGP*
4: *have_pgp_priority* ← ... indicador da prioridade do servidor de email para *OpenPGP*
5: *is_pgp_signed_message* ← ... indicador do estado criptográfico da mensagem para
6: *OpenPGP*
7: *is_pgp_encrypted_message* ← ... indicador do estado criptográfico da mensagem para
8: *OpenPGP*

```
 9:
10: if sender_have_pgp_priv_key == True and  is_pgp_encrypted_message == False and
11:        is_pgp_signed_message == True then:
12:    signature ← extract_signature(signed_body)
13:    signed_data, is_multipart ← extract_signed_data(signed_body)
14:    if pgp_verify(gnupg_pub, signature, signed_data) == True then:
15:       if is_multipart == True then:
16:    content_type, body ← split_multipart_message(signed_data)
17:          change_header(content_type)
18:       else:
19:          content_type, content_transfer_encoding, body ←
20:                                divide_multipart_message(signed_data)
21:          change_header(content_type, content_transfer_encoding)
22:       end if
23:       replace_body(body)
24:       add_control_headers()
25:  accept()
26:    else:
27:       reject()
28:    end if
29: end if
```

As operações de verificação e validação de assinaturas *OpenPGP* ocorrem nos servidores de email destinatários, e como tal, para ocorrerem, é necessário o filtro de email (*i*) encontrar a chaves pública *OpenPGP* do remetente (via DNS), (*ii*) a mensagem não se encontrar cifrada (*iii*) a mensagem estar assinada (linha 10 e 11).

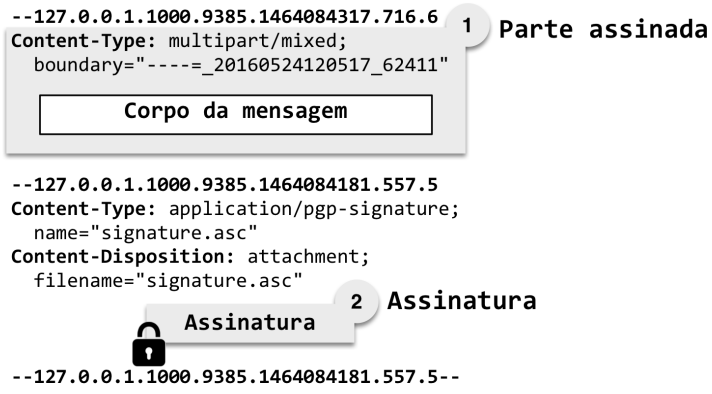

Figura 4-6 – Mensagem de email com múltiplas partes assinada via *OpenPGP*.
(1) conteúdo assinado **(2)** assinatura da mensagem.

Após verificar estas condições, os dados assinados e a assinatura devem ser extraídos do corpo da mensagem [Figura 4-6] (linha 12 e 13). A assinatura deve ser verificada e validada com a chave pública *OpenPGP* do remetente, os dados assinados e a assinatura (linha 14). Se não houver nenhum erro, a operação de verificação é validada, caso contrário a mensagem de email é rejeitada pelo filtro de email (linha 27).

Após a verificação e validação da assinatura, se os dados assinados corresponderem a uma mensagem de email composta por múltiplas partes, o campo *Content-Type* e o corpo da mensagem assinado devem ser extraídos dos dados assinados, e substituem os atuais. Se os dados assinados correspondem a uma mensagem de email em texto simples, os campos *Content-Type* e *Content-Transfer-Encoding* devem ser extraídos dos dados assinados, e substituem os atuais. Por fim, e para ambos os casos, são adicionados os campos de controlo.

4.2.3 Operações criptográficas via S/MIME

As operações criptográficas com S/MIME, assumem a existência de certificados de chave pública de utilizadores publicados em DNS, assim como o certificado de chave pública e respetiva chave privada, sediados no servidor de email responsável por gerir estes elementos de utilizadores de domínios geridos nesse servidor de email.

As operações criptográficas via S/MIME para as operações de cifra e assinatura, não sofrem alterações no caso de a mensagem de email ser em **texto simples** ou composta por **múltiplas partes**, ao contrário do que acontece nas operações criptográficas via *OpenPGP*.

Algoritmo 2.1. Mensagem de email cifrada via S/MIME

```
0: body ← ... corpo da mensagem de email
1: header ← ... corpo da mensagem de email
2: have_smime_users ← ... indicador da existência de destinatários S/MIME
3: smime_certificates ← ... certificados de chaves públicas S/MIME de destinatários
4: have_smime_priority ← ... indicador da prioridade do servidor de email para S/MIME
5: is_smime_encrypted_message ← ... indicador do estado criptográfico da mensagem para
6:                                 S/MIME
7:
8: if have_smime_users == True and is_smime_encrypted_message == False and
9:                                    have_smime_priority == True then:
10:    prefix ← extract_prefix_to_encrypt(header)
11:    body, headers_to_change ← smime_encrypt(body, prefix,
12:                                            smime_certificates)
13:    change_header(headers_to_change)
14:    replace_body(body)
15:    add_control_headers()
17:    accept()
18: end if
```

As operações de cifra S/MIME só ocorrem nos servidores de email remetentes, e como tal, para ocorrerem, é necessário o filtro de email (*i*) encontrar os certificados de

chave pública S/MIME dos destinatários (via DNS), (*ii*) a mensagem não se encontrar cifrada e (iii) e estar configurado para operar criptograficamente pelo utilizador sobre S/MIME (linha 8 e 9).

Após a passagem nesta condição, é realizada uma extração dos campos do cabeçalho da mensagem de email, onde são extraídos, caso existam, o campo *Content-Type*, *Content-Disposition*, *Content-Transfer-Encoding, MIME-Version* (linha 10). Estes campos, juntamente com corpo da mensagem, são cifrados, resultando num novo corpo de mensagem cifrado e em novos campos (linha 11 e 12). Os novos campos substituem os antigos, assim como o novo corpo da mensagem cifrado (linha 13 e 14). Por fim são adicionados os campos de controlo (linha 15).

Nas mensagens de texto simples, são extraídas as entradas de cabeçalho *Content-Type*, *Content-Disposition* e *Content-Transfer-Encoding*. Se a mensagem for composta por múltiplas partes, apenas o *Content-Type* e *MIME-Version* são extraídos.

Após a chegada da mensagem cifrada ao servidor de email destinatário, esta deve ser decifrada.

Algoritmo 2.2. Mensagem de email decifrada via S/MIME

```
 0: body ← ... corpo da mensagem de email
 1: header ← ... corpo da mensagem de email
 2: have_smime_users ← ... indicador da existência de destinatários S/MIME
 3: smime_certificate ← ... certificado de chave pública S/MIME do destinatário
 4: smime_priv_key ← ... chave privada S/MIME do destinatário
 5: is_smime_encrypted_message ← ... indicador do estado criptográfico da mensagem para
 6:                                 S/MIME
 7:
 8: if have_smime_users == True and is_smime_encrypted_message == True then:
 9:       prefix ← extract_prefix_to_encrypt(header)
10:  body, headers_to_change ← smime_decrypt(prefix, body, smime_priv_key,
11:            smime_certificate)
12:       change_header(headers_to_change)
13:       replace_body(body)
14:       add_control_headers()
15:       accept()
16: end if
```

As operações de decifra S/MIME só ocorrem nos servidores de email destinatários, e como tal, para ocorrerem, é necessário o filtro de email (*i*) encontrar a chave privada e o respetivo certificado S/MIME do destinatário (no repositório local), (*ii*) verificar que a mensagem se encontra cifrada (linha 8).

Após a passagem nesta condição, é realizada uma extração dos campos do cabeçalho da mensagem de email, onde são extraídos, caso existam, o campo *Content-Type*, *Content-Disposition*, *Content-Transfer-Encoding, MIME-Version* (linha 9). Estes cam-

pos, juntamente com o corpo da mensagem cifrada, são decifrados com a chave privada e respetivo certificado de chave pública S/MIME do destinatário. Esta operação resulta num novo corpo de mensagem (decifrado) e novos campos de cabeçalho (linha 10 e 11). Os novos campos substituem os antigos, assim como o novo corpo da mensagem decifrado (linha 12 e 13). Por fim são adicionados os campos de controlo (linha 14).

No processo de decifra, tanto nas mensagens de email em texto simples como nas compostas por múltiplas partes, todos os cabeçalhos são extraídos os cabeçalhos de *Content-Type*, *Content-Disposition*, *Content-Transfer-Encoding* e *MIME-Version*.

Algoritmo 2.3. Mensagem de email assinada via S/MIME

```
 0: body ← ... corpo da mensagem de email
 1: header ← ... corpo da mensagem de email
 2: sender_have_smime_priv_key ← ... indicador da existência de chave privada S/MIME do
 3:          remetente
 4: sender_smime_certificate ← ... certificado de chave pública S/MIME do remetente
 5: sender_smime_priv_key ← chave privada S/MIME do remetente
 6: have_smime_priority ← ... indicador da prioridade do servidor de email para S/MIME
 7: is_smime_encrypted_message ← ... indicador do estado criptográfico da mensagem para
 8:                                  S/MIME
 9:
10: if sender_have_smime_priv_key == True and is_smime_encrypted_message == False
11:                                        and have_smime_priority == True then:
12:     prefix ← extract_prefix_to_sign(header)
13:  body, headers_to_change ← smime_sign(prefix, body,
14:                                sender_smime_priv_key, smime_certificate)
15:     change_header(headers_to_change)
16:     replace_body(body)
17:     add_control_headers()
18:     accept()
19: end if
```

As operações de assinatura S/MIME só ocorrem nos servidores de email destinatários, e como tal, para ocorrerem, é necessário o filtro de email (*i*) encontrar a chave privada e o respetivo certificado S/MIME do remetente (no repositório local), (*ii*) verificar que a mensagem não se encontra cifrada e (*iii*) e estar configurado para operar criptograficamente pelo utilizador sobre S/MIME (linha 10 e 11).

Após a passagem nesta condição, é realizada uma extração dos campos do cabeçalho da mensagem de email, onde são extraídos, caso existam, o campo *Content-Type*, *Content-Disposition*, *Content-Transfer-Encoding, MIME-Version* (linha 12). Estes campos, juntamente com o corpo da mensagem são assinados, com a chave privada e respetivo certificado de chave pública S/MIME do remetente. Esta operação resulta num novo corpo de mensagem (assinado) e novos campos de cabeçalho (linha 13 e 14). Os

novos campos substituem os antigos, assim como o novo corpo da mensagem assinado (linha 15 e 16). Por fim são adicionados os campos de controlo (linha 17).

Algoritmo 2.4. Verificação e validação de uma mensagem de email assinada via S/MIME

```
 0: body ← ... corpo da mensagem de email
 1: header ← ... corpo da mensagem de email
 2: sender_smime_certificate ← ... certificado de chave pública S/MIME do remetente
 3: sender_have_smime_certificate ← ... indicador da existência de um certificado de
 4:                                    chave pública S/MIME do remetente
 5: is_smime_signed_message ← ... indicador do estado criptográfico da mensagem para
 6:                             S/MIME
 7:
 8: if sender_have_smime_certificate == True and is_smime_signed_message == True
 9: then:
10:    prefix ← extract_prefix_to_verify_signature(header)
11:    is_valid, body, headers_to_change = smime_verify(prefix, body,
12:                                                     sender_smime_certificate)
13:    if is_valid == True then:
14:       replace_body(body)
15:       change_header(headers_to_change)
16:       add_control_headers()
17:       accept()
18:    else:
19:       reject()
20: end if
```

As operações de verificação e validação de assinaturas S/MIME só ocorrem nos servidores de email destinatários, e como tal, para ocorrerem, é necessário o filtro de email encontrar (*i*) encontrar o certificado de chave pública S/MIME do remetente (via DNS), (*ii*) verificar que a mensagem se encontra assinada (linha 8 e 9).

Após a passagem nesta condição, é realizada uma extração dos campos do cabeçalho da mensagem de email, onde são extraídos, caso existam, o campo *Content-Type*, *Content-Disposition*, *Content-Transfer-Encoding, MIME-Version* (linha 10). Estes campos, juntamente com o corpo da mensagem assinado, são verificados quando à validade da assinatura digital com o certificado de chave pública S/MIME do remetente. Esta operação resulta num novo corpo de mensagem e novos campos de cabeçalho (linha 11 e 12). Se houver algum erro, a mensagem de email será rejeitada pelo filtro de email (linha 19)

Se a assinatura da mensagem for corretamente verificada e validada (linha 13), os novos campos substituem os antigos, assim como o novo corpo da mensagem (linha 14 e 15). Por fim são adicionados os campos de controlo (linha 16).

4.2.4 Campos de controlo

Por forma a serem identificadas as operações criptográficas realizadas sobre as mensagens de email, assim como o estado em que a mensagem se encontra, foram definidas as seguintes entradas de cabeçalho de email que devem ser adicionadas após as operações criptográficas:

- **X-Anubis-Auto-Discovery-State:** responsável por indicar em que estado se encontra a mensagem, podendo esse estado ser: cifrado, decifrado, assinado ou verificada assinatura ou texto em claro [Figura 4-7 e 4-8].

- **X-Anubis-Auto-Discovery-Encryption:** responsável por indicar informação relevante para o entendimento de como o processo de cifra/decifra foi realizado. Esta entrada deve conter a seguinte informação [Figura 4-7 e 4-8]:
  - *hostname* do servidor de email que realizou a operação;
  - modo de criptográfico aplicado (S/MIME ou *OpenPGP*);
  - estado do conteúdo da mensagem;
  - utilizadores, para os quais, foi realizada a operação criptográfica;
  - identificador da chave pública utilizada na operação (*fingerprint*);
  - tempo de execução do filtro de email para a operação criptográfica;
  - validação DNSSEC ao obter a chave pública no DNS;
  - data e hora da operação criptográfica.

- **X-Anubis-Auto-Discovery-Signature:** responsável por indicar informação relevante para o entendimento de como o processo de assinatura/verificação foi realizado. Esta entrada deve conter informação como [Figura 4-7 e 4-8]:
  - *hostname* do servidor de email que realizou a operação;
  - modo de criptográfico aplicado (S/MIM ou *OpenPGP*);
  - estado do conteúdo da mensagem;
  - utilizador, pelo qual, foi realizada a operação criptográfica;
  - identificador da chave pública utilizada na operação (*fingerprint*);
  - tempo de execução do filtro de email para a operação criptográfica;
  - validação DNSSEC ao obter a chave pública no DNS;
  - data e hora da operação criptográfica.

Figura 4-7 – Campos adicionados ao cabeçalho no servidor de email remetente
1 – Estado da mensagem depois das operações do filtro de email;
2 – Detalhes de operações relativamente a operações de assinatura;
3 – Detalhes de operações relativamente a operações de cifra.

Figura 4-8 – Campos adicionados ao cabeçalho no servidor de email destinatário.
1 – Estado da mensagem depois das operações do filtro de email;
2 – Detalhes de operações relativamente a operações de validação de assinatura;
3 – Detalhes de operações relativamente a operações de decifra.

A adição destes campos no cabeçalho das mensagens de email, deve ser realizada a cada execução do filtro de email, ou seja, devem ser adicionados tanto no servidor de email remetente como no destinatário. Estes campos permitem, ao analisar o cabeçalho de uma determinada mensagem de email: (*i*) perceber que dados levaram à tomada de decisão dos filtros de email nos respetivos filtros de email (prioridade no protocolo, modo, utilizador para quem foi o filtro operou), e (*ii*) detetar o que gerou uma possível falha (excesso de tempo de execução, comparação dos *fingerprints* das chaves utilizadas nos).

4.2.5 *XCLIENT*

Quando um servidor SMTP anuncia suporte para o comando *XCLIENT* [31], um cliente SMTP pode enviar informações, que substitui um ou mais atributos de sessão relacionada ao cliente. Para isso é necessário configurar o *XCLIENT* [31] no servidor SMTP, autorizando apenas determinados endereços IP, nomeadamente endereços IP dos clientes e servidores SMTP que são supostos se ligarem ao servidor com o *XCLIENT* [31] configurado.

```
#XClient - Postifx Configuration

smtpd_authorized_xclient_hosts = 172.16.56.100 192.168.56.200 127.0.0.1
                                 (1)           (2)
```

Figura 4-9 – Configuração do *XCLIENT* no servidor de email.
(1) Endereço IP do próprio servidor de email; **(2**) Endereço IP de outro servidor de email

Foi definido nos servidores de email remetente e destinatário o suporte para o comando *XCLIENT*, configurado para que cada um deles, autorizarem o próprio endereço IP e o endereço IP do outro servidor [Figura 4-9]. Este mecanismo é bastante importante, para a reinjeção de mensagens de email (Capitulo 3). Pois quando uma mensagem de email é reinjetada no próprio servidor de email, é preciso que a próxima execução do filtro de email resultante da reinjeção mantenha os mesmos atributos da execução corrente. Supondo que é enviada uma mensagem entre dois utilizadores de domínios diferentes, de `example.net` (remetente) para `example.org` (destinatário), e supondo que há uma injeção em cada um dos servidores de email, o *XCLIENT* deve fornecer a seguintes alterações relativamente aos atributos de sessão [Figura 4-10 e 4-11]:

```
1
Received: from 127.0.0.1 (ns [127.0.0.1])
      by mail.example.net (Postfix) with SMTP id 129D66504
      for <user1@æxample.org>; Tue, 19 Apr 2016 12:10:06 +0100 (WEST)

                          XClient

2
Received: from mail.example.net (mail.example.net [192.168.56.200])
      by mail.example.net (Postfix) with SMTP id 129D66504
      for <user1@æxample.org>; Tue, 19 Apr 2016 12:10:06 +0100 (WEST)
```

Figura 4-10 – *XCLIENT* no servidor de email remetente.
(1) Configuração do filtro de email sem *XCLIENT*; **(2)** Configuração do filtro de email com *XCLIENT*

1. **Reinjeção no próprio servidor de email remetente:** assume os atributos de sessão do servidor de email remetente (`example.net`), nomeadamente o endereço IP, passando de *localhost* para o seu endereço IP exterior, e o nome

do servidor de email para o exterior, passando de `ns` para `mail.example.net` [Figura 4-10].

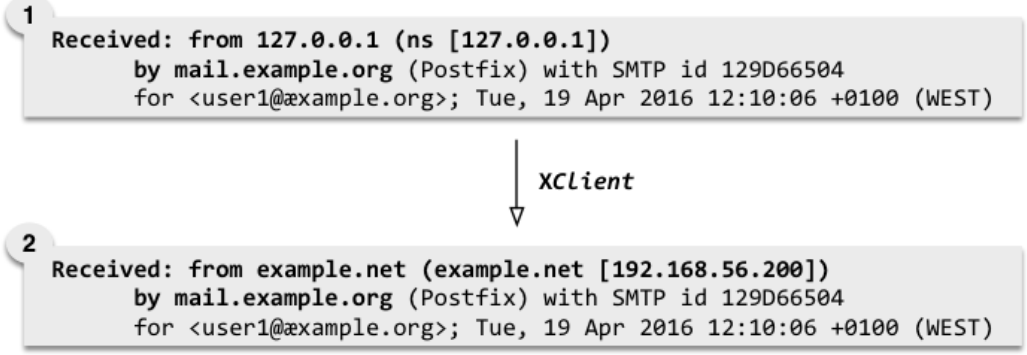

Figura 4-11 – *XCLIENT* no servidor de email destinatário.
(1) Execução do filtro de email sem *XCLIENT;* **(2)** Execução do filtro de email com *XCLIENT*

2. **Reinjeção no próprio servidor de email destinatário:** assume os atributos de sessão do servidor de email remetente (`example.net`), nomeadamente o endereço IP, passando de *localhost* para o seu endereço IP do servidor de email remetente (`example.net`) para o exterior, e o nome do servidor de email para o exterior, passando de `ns` para `mail.example.net` [Figura 4-9].

A adição do *XCLIENT* permite tornar o mecanismo de reinjeção de mensagens de email por parte do filtro de email mais transparentes e mais seguras, pois as reinjeções via localhost são transformadas em reinjeções que assumem os atributos de sessão do servidor de email remetente [Figura 4-10 e 4-11].

4.3 Implementação da Plataforma de Gestão de Chaves

Nesta secção serão descritos alguns detalhes de implementação da plataforma de gestão de chaves, assim como alguns pontos chave para o seu normal funcionamento, nomeadamente o formato dos pedidos e respostas entre administrador e plataforma. As interações são do tipo pedido/resposta, onde as operações são feitas remotamente no servidor de email através de um serviço *RESTful,* permitindo operações do tipo de GET, PUT, POST e DELETE, associadas às operações de obter, publicar, atualizar e remover chave pública ou certificado. O serviço *RESTful* escuta no porto 5000. Relativamente à autenticação e autorização no acesso á plataforma, e com o fim de simplificar a solução, a plataforma é um serviço local, onde só pode ser acedida por clientes a correr no servidor onde se encontra a plataforma.

4.3.1 Obter chave pública *OpenPGP* ou certificado S/MIME

A obtenção de uma chave pública *OpenPGP* ou certificado de chave pública S/MIME é feita através de uma operação GET ao serviço *RESTful*. Todo o fluxo para esta operação ocorre da seguinte forma, onde o administrador pretende obter um certificado S/MIME do utilizador `user1@example.org` [Figura 4-12]:

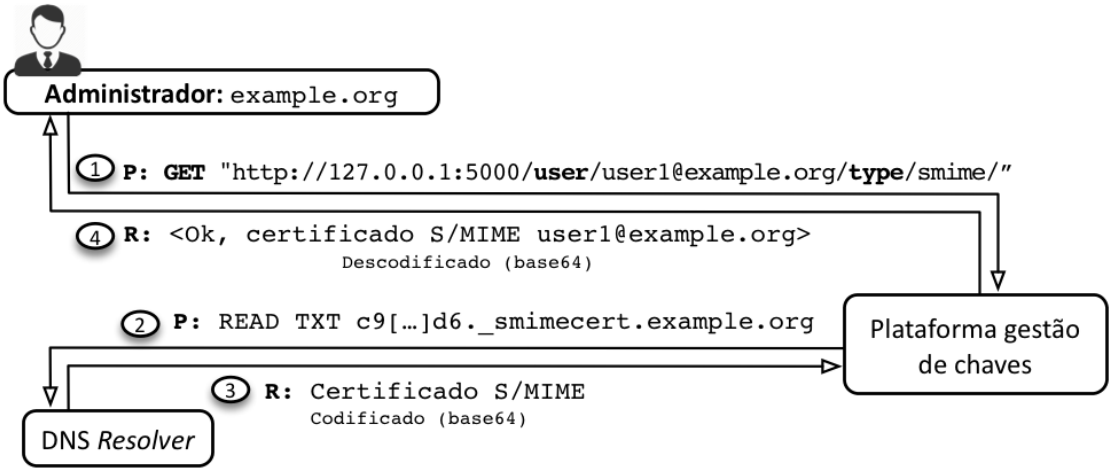

Figura 4-12 – Fluxo de um pedido GET ao serviço *RESTful* de um certificado S/MIME.

1. O administrador do domínio `example.org` realiza um pedido GET com o seguinte conteúdo: o utilizador que pretende obter a chave (**`user`**`/user1@example.org`) e o tipo (**`type`**`/smime`);
2. A plataforma converte o conteúdo em (**1**) num pedido por um TXT RR ao DNS *Resolver* [29, 30];
3. O DNS *Resolver* retorna o conteúdo do RR, nomeadamente um certificado S/MIME ou uma chave pública *OpenPGP*, ambos codificados em base 64;
4. A plataforma descodifica a conteúdo do RR. Se a operação ocorreu corretamente retorna ao administrador `<OK, certificado S/MIME>`. Caso contrário retorna `<Error, null>`.

Na obtenção de uma chave pública *OpenPGP*, o fluxo é idêntico ao exemplificado para um certificado S/MIME, sendo que o pedido à plataforma da gestão de chaves e o pedido ao DNS resolver sofrem pequenas alterações [Figura 4-13].

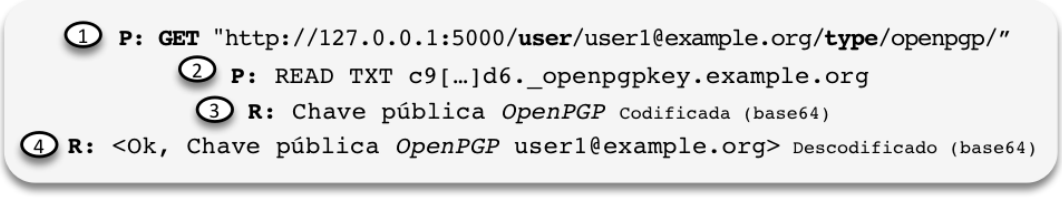

Figura 4-13 – Fluxo de um pedido GET ao serviço *RESTful* de uma chave pública *OpenPGP*.

4.3.2 Publicar chave pública *OpenPGP* ou certificado S/MIME

A publicação de uma chave pública *OpenPGP* ou certificado de chave pública S/MIME é feita através de uma operação PUT ao serviço *RESTful*. Todo o fluxo para esta operação ocorre da seguinte forma, onde o administrador pretende publicar um certificado S/MIME do utilizador `user1@example.org` [Figura 4-14]:

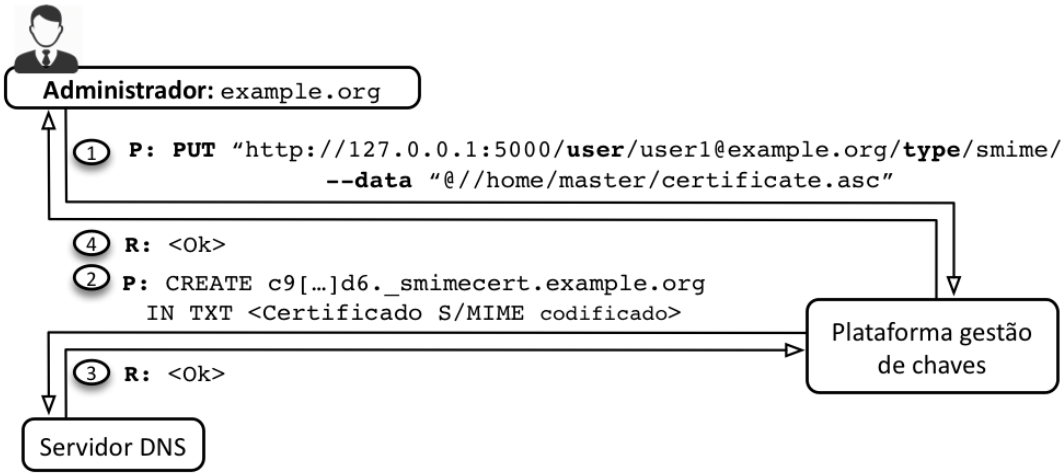

Figura 4-14 - Fluxo de um pedido PUT ao serviço *RESTful* de um certificado S/MIME.

1. O administrador do domínio `example.org` realiza um pedido *PUT* com o seguinte conteúdo: o utilizador que pretende obter a chave (**`user`**`/user1@example.org`), o tipo (**`type`**`/smime`) e o certificado **(`--data`**);
2. A plataforma cria um TXT RR com o conteúdo em **(1)** alocando-o no ficheiro de zona `example.org` no servidor DNS [29, 30];
3. O Servidor de DNS retorna um `<OK>`;
4. Se a operação ocorreu corretamente retorna ao administrador `<OK>`. Caso contrário retorna `<Error>`.

Na publicação de uma chave pública *OpenPGP*, o fluxo é idêntico ao exemplificado para um certificado S/MIME, sendo que o pedido à plataforma da gestão de chaves e o pedido ao DNS resolver sofrem pequenas alterações [Figura 4-15].

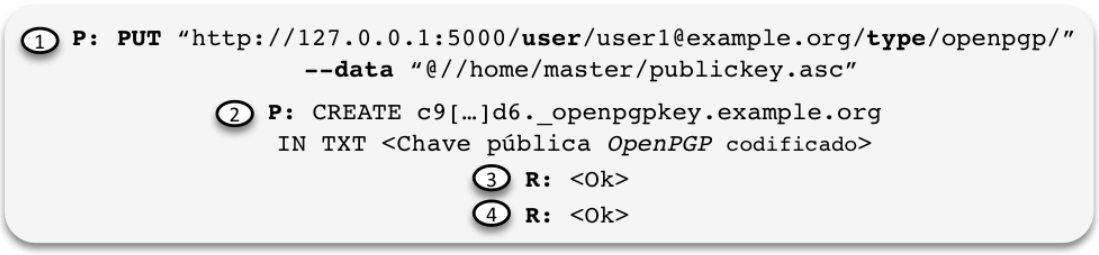

Figura 4-15 – Fluxo de um pedido PUT ao serviço *RESTful* de uma chave pública *OpenPGP*.

4.3.3 Atualizar chave pública *OpenPGP* ou certificado S/MIME

A atualização de uma chave pública *OpenPGP* ou certificado de chave pública S/MIME é feita através de uma operação POST ao serviço *RESTful*. Todo o fluxo para esta operação ocorre da seguinte forma, onde o administrador pretende atualizar um certificado S/MIME do utilizador `user1@example.org` [Figura 4-16]:

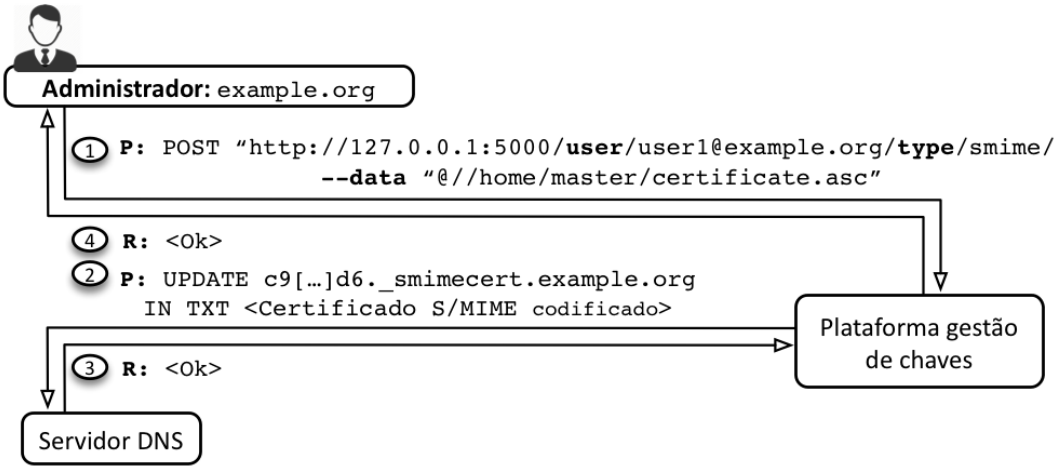

Figura 4-16 – Fluxo de um pedido POST ao serviço *RESTful* de um certificado S/MIME.

1. O administrador do domínio `example.org` realiza um pedido *POST* com o seguinte conteúdo: o utilizador que pretende obter a chave, (**`user`**`/user1@example.org`), o tipo (**`type`**`/smime`) e o certificado (**`--data`**);
2. A plataforma cria um TXT RR com o conteúdo de (**1**), substituído o antigo TXT RR no ficheiro de zona `example.org` no servidor DNS por este. [29, 30];
3. O Servidor de DNS retorna um `<OK>`;
4. Se a operação ocorreu corretamente retorna ao administrador `<OK>`. Caso contrário retorna `<Error>`.

Na atualização de uma chave pública *OpenPGP*, o fluxo é idêntico ao exemplificado anteriormente [Figura 4-16], sendo que o pedido à plataforma da gestão de chaves e o pedido ao DNS *resolver* sofrem pequenas alterações [Figura 4-17].

Figura 4-17 – Fluxo de um pedido POST ao serviço *RESTful* para uma chave pública *OpenPGP*.

4.3.4 Remover chave pública *OpenPGP* ou certificado S/MIME

A remoção de uma chave pública *OpenPGP* ou certificado de chave pública S/MIME é feita através de uma operação DELETE ao serviço *RESTful*. Todo o fluxo para esta operação ocorre da seguinte forma, onde o administrador pretende remover um certificado S/MIME do utilizador `user1@example.org` [Figura 4-18]:

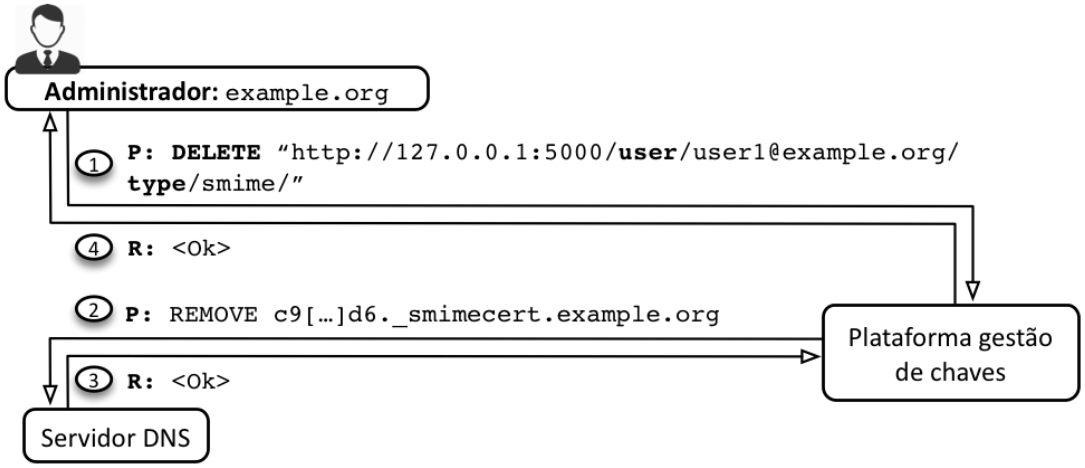

Figura 4-18 – Fluxo de um pedido DELETE ao serviço *RESTful* de um certificado S/MIME.

1. O administrador do domínio `example.org` realiza um pedido *DELETE* com o seguinte conteúdo: o utilizador que pretende remover a chave (**`user`**`/user1@example.org`) e o tipo (**`type`**`/smime`);
2. A plataforma cria um pedido com o conteúdo de **(1)** para remover o TXT RR no ficheiro de zona `example.org` no servidor DNS [29, 30];
3. O Servidor de DNS retorna um `<OK>`;
4. Se a operação ocorreu corretamente retorna ao administrador `<OK>`. Caso contrário retorna `<Error>`.

Na remoção de uma chave pública *OpenPGP*, o fluxo é idêntico ao exemplificado anteriormente [Figura 4-16], sendo que o pedido à plataforma da gestão de chaves e o pedido ao DNS *resolver* sofrem pequenas alterações [Figura 4-19].

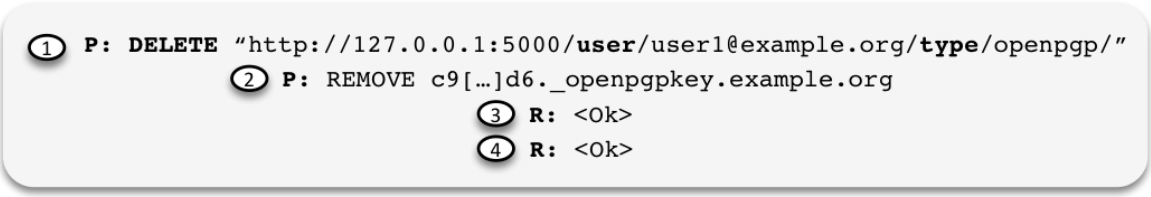

Figura 4-19 – Fluxo de um pedido DELETE ao serviço *RESTful* de uma chave pública *OpenPGP*.

4.4 Avaliação Experimental

4.4.1 Plataforma de teste

De forma a simular um ambiente de teste o mais próximo possível de um ambiente real, onde se possa simular o envio e receção de email (clientes de diferentes domínios), o email passar por dois servidores de email (remetente e destinatário), a proposta solução foi implementada sobre duas máquinas virtuais com o sistema operativo *Lubuntu* 15, onde foi instalado e configurado em cada servidor de email um servidor SMTP (*Postfix*), *MailStore* (*Dovecot*) e serviço de *webmail* (*Squirrelmail*) para o normal funcionamento do serviço de email. Cada máquina virtual foi transformada num Servidor de DNS (*Bind9*). Cada servidor foi configurado para um único domínio (`example.net` e `example.org`). As máquinas virtuais foram configuradas com 2GB de RAM 1333 MHz DDR3, e um núcleo de um processador Intel i510 2,3GHz.

Em cada servidor de email, foram criadas 6 contas de utilizador de email (do `user1` a `user6`) para o seu respetivo domínio (`example.net` e `example.org`). 2 utilizadores com chaves pública e privada *OpenPGP* (2048 bits), 2 utilizadores certificados de chave pública S/MIME e respetiva chave privada de vários tamanhos (2048 bits) e 2 utilizadores sem qualquer tipo de chaves. Todas as chaves públicas e certificados foram colocadas nos ficheiros de zona nos servidores de DNS, e as chaves privadas colocadas no repositório de chaves.

4.4.2 Resultados

A plataforma de gestão de chaves, sendo uma plataforma auxiliar para a publicação e gestão de chaves públicas *OpenPGP* e certificados de chave pública S/MIME, apenas foi testada tendo em conta os aspetos funcionais. Foram validadas todas as operações quanto ao seu correto funcionamento.

O filtro de email foi testado tendo em conta aspetos funcionais e não funcionais. Relativamente aos aspetos funcionais foi testada a tomada de decisão quanto às operações criptográficas. Para testar todas as operações, foram feitas várias combinações de envios entre os utilizadores criados para criar todas as combinações possíveis. Estas combinações resultam nas seguintes tabelas:

Envio de uma mensagem de um utilizador **sem chaves** para:	*Propriedades de segurança garantidas*			
	Confidencialidade	**Integridade**	**Autenticidade**	**Não Repúdio**
utilizador **sem chaves**				
utilizador **c/ chaves *OpenPGP***	√			
utilizador **c/ certificado S/MIME**	√			

Tabela 4-1 – Propriedades de segurança garantidas no envio de mensagens de email de um utilizador sem chaves para um utilizador sem chaves, com chave *OpenPGP* e com certificado S/MIME.

Envio de uma mensagem de um utilizador com **chaves *OpenPGP*** para:	*Propriedades de segurança garantidas*			
	Confidencialidade	**Integridade**	**Autenticidade**	**Não Repúdio**
utilizador **sem chaves**		√	√	√
utilizador **c/ chaves *OpenPGP***	√	√	√	√
utilizador **c/ certificado S/MIME**		√*	√*	√*

* - Assumindo que a prioridade do servidor de email remetente é *OpenPGP*

Tabela 4-2 – Propriedades de segurança garantidas no envio de mensagens de email de um utilizador com chave *OpenPGP* para um utilizador sem chave, com chave *OpenPGP* e com certificado S/MIME.

Envio de uma mensagem de um utilizador com **certificados S/MIME** para:	*Propriedades de segurança garantidas*			
	Confidencialidade	**Integridade**	**Autenticidade**	**Não Repúdio**
utilizador **sem chaves**	√			
utilizadores **c/ chaves *OpenPGP***	√*			
utilizadores **c/ certificado S/MIME**	√	√	√	√

* - Assumindo que a prioridade do servidor de email remetente é *OpenPGP*

Tabela 4-3 – Propriedades de segurança garantidas no envio de mensagens de email de um utilizador com certificado S/MIME para um utilizador sem chave, com chave *OpenPGP* e com certificado S/MIME.

As Tabelas 4-1, 4-2 e 4-3 representam as propriedades de segurança garantidas em cada uma das possíveis combinações, sendo que:

- **Confidencialidade:** corresponde à mensagem de email ser cifrada no servidor de email remetente e decifrada no servidor de email destinatário.
- **Autenticidade, Integridade e Não Repúdio:** corresponde à mensagem de email ser assinada no servidor de email remetente e verificada e validada a sua assinatura no servidor de email destinatário.

Ambas podem ser garantidas conjuntamente (cifra e assinatura), e para os diferentes tipos de chaves (*OpenPGP* e S/MIME).

Os aspetos não funcionais foram testados nas máquinas virtuais acima indicadas (`example.org` e `example.net`) para um servidor de email remetente e destinatário. Relativamente a aspetos não funcionais foram realizados dois tipos de testes, o primeiro onde (*i*) foram medidos os aumentos de tamanho das mensagens de email após cada operação criptográfica para mensagens de vários tamanhos, e o segundo onde (*ii*)

foram medidos os tempos de execução do filtro de email para cada operação criptográfica sobre mensagens de email de vários tamanhos.

Para (*i*) foram enviadas mensagens de email de 1.754, 780.481, 2.120.925, 4.272.513 *Bytes*, sendo a primeira em texto simples e as restantes compostas por duas partes, sendo uma de texto simples e outra um anexo de uma imagem em formato JPEG de 570Kbs, 1,5Mb e 3,1Mb respetivamente. Para cada um dos envios, foi recolhido o tamanho da mensagem de email depois de ser alvo de um processo criptográfico, para perceber de que forma o tamanho da mensagem cresce quer pelo protocolo, (S/MIME e *OpenPGP*) quer pela operação criptográfica (cifra ou assinatura) utilizada.

Operações criptográficas	*Tamanho das mensagens de email (Bytes)*			
Sem operação	**1.754**	**780.481**	**2.120.925**	**4.272.513**
Cifra *OpenPGP*	2.256 **(+28,6%)**	802.685 **(+2,85%)**	2.196.699 **(+3,57%)**	4.462.789 **(+4,45%)**
Assinatura *OpenPGP*	3.186 **(+81,6%)**	781.878 **(+0,17%)**	2.122.322 **(+0,07%)**	4.273.910 **(+0,03%)**
Cifra e Assinatura *OpenPGP*	3.371 **(+92,2%)**	803.777 **(+2,99%)**	2.197.785 **(+3,62%)**	4.463.873 **(+4,48%)**
Cifra S/MIME	3.631 **(+107,0%)**	1.074.438 **(+37,7%)**	2.917.553 **(+37,5%)**	5.875.979 **(+37,5%)**
Assinatura S/MIME	5.021 **(+186,3%)**	783.713 **(+0,41%)**	2.124.156 **(+0,15%)**	4.275.744 **(+0,08%)**
Cifra e Assinatura S/MIME	7.722 **(+340,3%)**	1.078.505 **(+38,2%)**	2.921.621 **(+37,8%)**	5.880.045 **(+37,6%)**

Tabela 4-4 – Aumento do tamanho das mensagens de email após operações criptográficas.

Recolhidos e analisados os dados relativos ao aumento do tamanho das mensagens de email após cada operação criptográfica para mensagens de vários tamanhos [Tabela 4-4], foi possível chegar às seguintes conclusões:

Quanto menor o tamanho da mensagem, maior é o aumento do tamanho da mensagem após uma operação criptográfica: na primeira coluna [Tabela 4-4], é possível verificar que a percentagem de aumento para qualquer operação criptográfica é superior à percentagem nas restantes colunas. Isto deve-se ao facto da mensagem enviada na primeira coluna ter um tamanho original reduzido (1.754 Bytes), e a adição de uma assinatura digital ou dos consequentes campos de cabeçalho, aumenta substancialmente o tamanho da mensagem para *OpenPGP*, e muito acentuadamente para S/MIME.

Com o aumento do tamanho das mensagens de email, o aumento adicionado pelas assinaturas digitais torna-se menor: sendo o tamanho da assinatura constante independentemente do tamanho da mensagem de email (foram utilizadas assinaturas de 160 bits e o algoritmo SHA1), é possível verificar nas linhas correspondentes às operações de assinatura [Tabela 4-4], a percentagem a diminuir acentuadamente.

Comparativamente com *OpenPGP*, as operações criptográficas via S/MIME adicionam maior aumento no tamanho das mensagens de email: tanto o *OpenPGP* como S/MIME, utilizam criptografia assimétrica. Logo, tendo em conta que foram utilizadas chaves com os mesmos tamanhos para ambos os protocolos, o aumento adicionado pelas operações criptográficas, não pode ser responsável pelas diferenças tão acentuadas.

No protocolo *OpenPGP*, além da operação criptográfica, a mensagem é antes comprimida, e depois codificada. O formato de compressão utilizado é o *zip*, sendo que a sua eficiência na compressão depende do tipo dos dados que são comprimidos. A codificação é feita para base64, aumentando o tamanho da mensagem de email em 33,3%. No protocolo S/MIME, além da operação criptográfica, a mensagem de email é depois codificada. A codificação é feita para base64, , aumentando o tamanho da mensagem de email em 33,3%.

Para tornar mais claro o impacto destas duas operações, foram medidos os aumentos e reduções causados pela codificação e compressão, para as mensagens com múltiplas partes identificadas acima [Tabela 4-5 e 4-6].

Tamanho da mensagem (*Bytes*)	**780.481**	**2.120.925**	**4.272.513**
Compressão zip (*Bytes*)	**581.837**	**1.593.867**	**3.239.202**
Compressão zip (redução)	-25,46%	-24,85%	-24,18%
Codificação base64 (*Bytes*)	**785.992**	**2.153.119**	**4.375.765**
Codificação base64 (aumento)	+35,09%	+35,09%	+35,09%
Aumento total	+0,70%	+1,51%	+2,41%

Tabela 4-5 – Impacto da operação de codificação (base64) no protocolo S/MIME.

Tamanho da mensagem (*Bytes*)	**780.481**	**2.120.925**	**4.272.513**
Codificação base64 (*Bytes*)	**1.054.370**	**2.865.146**	**5.771.677**
Codificação base64 (aumento)	+35,09%	+35,09%	+35,09%
Aumento total	+35,09%	+35,09%	+35,09%

Tabela 4-6 – Impacto das operações de compressão (zip) e codificação (base64) no protocolo *OpenPGP*.

Analisando os dados da Tabela 4-5 e 4-6 é possível verificar para as várias mensagens de email para o protocolo S/MIME têm um aumento de 35,09%, devido é codificação em base64. Este aumento não acontece para o protocolo *OpenPGP*, porque antes da codificação, as mensagens de email são comprimidas, levando a que o impacto do aumento gerado pela codificação seja minimizado. No pior cenário de compressão *OpenPGP*, onde a redução consequente seja mínima, o aumento do tamanho das mensagens de email aproximar-se-á do aumento do tamanho das mensagens de email através do protocolo S/MIME (+35,09%).

Para (*ii*), à semelhança de (*i*) foram enviadas mensagens de email de 1.754, 780.481, 2.120.925, 4.272.513 *Bytes*, sendo a primeira em texto simples e as restantes compostas por duas partes, sendo uma de texto simples e outra um anexo de uma imagem em formato JPEG de 570Kbs, 1,5Mb e 3,1Mb respetivamente. Para cada um dos envios, foi recolhido o tempo de execução do filtro de email para cada operação criptográfica no servidor de email remetente e destinatário. Para cada combinação foram recolhidos 10 tempos de execução e feita a sua média.

Operações criptográficas	*Média do tempo de execução do filtro de email (segundos)*			
Sem operação	**0,023 s**	**0,028s**	**0,031 s**	**0,046 s**
Cifra *OpenPGP*	**0,093 s**	**0,154 s**	**0,395 s**	**0,583 s**
Assinatura *OpenPGP*	**0,147 s**	**0,110 s**	**0,232 s**	**0,279 s**
Cifra e Assinatura *OpenPGP*	**0,212 s**	**0,229 s**	**0,495 s**	**0,760 s**
Cifra S/MIME	**0,040 s**	**0,296 s**	**2,538 s**	**12,192 s**
Assinatura S/MIME	**0,033 s**	**0,495 s**	**4,261 s**	**19,177 s**
Cifra e Assinatura S/MIME	**0,031 s**	**0,683 s**	**5,721 s**	**28,537 s**

Tabela 4-7 – Média do tempo de execução do filtro de email para operações criptográficas de cifra, assinatura e ambas para *OpenPGP* e S/MIME.

Operações criptográficas	*Média do tempo de execução do filtro de email (segundos)*			
Sem operação	**0,023 s**	**0,018 s**	**0,019 s**	**0,027 s**
Decifra *OpenPGP*	**0,133 s**	**0,182 s**	**0,325 s**	**0,755 s**
Verificação de assinatura *OpenPGP*	**0,458 s**	**0,290 s**	**0,579 s**	**0,522 s**
Decifra e verificação de assinatura *OpenPGP*	**0,442 s**	**0,492 s**	**0,634 s**	**1,353 s**
Decifra S/MIME	**0,080 s**	**0,096 s**	**0,169 s**	**0,398 s**
Verificação de assinatura S/MIME	**0,053 s**	**0,090 s**	**0,196 s**	**0,562 s**
Decifra e verificação de assinatura S/MIME	**0,064 s**	**0,085 s**	**0,251 s**	**0,789 s**

Tabela 4-8 – Média do tempo de execução do filtro de email para operações criptográficas de decifra, validação de assinatura e ambas para *OpenPGP* e S/MIME.

Calculadas e analisadas as médias dos tempos de execução do filtro de email para cada operação criptográfica sobre mensagens de email de vários tamanhos [Tabela 4-7 e 4-8], foi possível chegar às seguintes conclusões:

Com o aumento do tamanho das mensagens, o tempo de execução do filtro tende em aumentar: sendo criptografia, quanto maior for um bloco de dados a ser cifrado ou assinado, maior será o tempo de processamento.

O tempo de execução do filtro de email para operações de cifra ou assinatura através de S/MIME é maior que através de *OpenPGP*: para perceber o porque do tempo de execução do filtro de email para as operações S/MIME ser bastante inferior, foram medidos os tempos de execução das funções da biblioteca responsável pelas operações criptográficas através de S/MIME (*M2Crypto*) [Tabela 4-9].

É possível verificar na Tabela 4-9 que com o aumento do tamanho das mensagens de email, a percentagem de tempo de execução ocupado pelas funções da biblioteca tende em aumentar, e em ocupar a grande parte do tempo de execução. Isto demonstra que a diferença de tempos de execução do *OpenPGP* e S/MIME deve-se às funções da biblioteca S/MIME (*M2Crypto*), onde possivelmente, não se encontram tão otimizadas como as funções da biblioteca *OpenPGP* (*gnupg*).

Operação Criptográfica	*Operações da biblioteca* SMIME (*M2Crypto)*	*Percentagem do tempo de execução das funções da biblioteca* SMIME *para mensagens de email de vários tamanhos*			
		1.754 *Bytes*	780.481 *Bytes*	2.120.925 *Bytes*	4.272.513 *Bytes*
Assinatura S/MIME	*sign*	8,68%	29,80%	21,39%	24,08%
	write	7,10%	30,62%	26,28%	41,49%
	encrypt	0,61%	32,94%	48,64%	33,99%
	write	0,14%	0,50%	0,21%	0,05%
	Total	**16,54%**	**93,86%**	**96,52%**	**99,60%**
Cifra S/MIME	*sign*	0,09%	43,67%	49,13%	45,66%
	write	0,06%	43,33%	47,44%	52,64%
	Total	**0,15%**	**86,99%**	**96,57%**	**98,30%**
Cifra e Assinatura S/MIME	*encrypt*	0,81%	79,36%	94,47%	95,22%
	write	0,19%	1,64%	0,38%	0,29%
	Total	**1,00%**	**81,00%**	**94,85%**	**95,52%**

Tabela 4-9 – Percentagem do tempo de execução para as funções da biblioteca SMIME.

O tempo de execução do filtro de email para operações de decifra, validação de assinatura e ambas através de S/MIME, aproxima-se do tempo de execução através de *OpenPGP*: nas operações de cifra ou assinatura, a percentagem de tempo de execução utilizado pelas funções da biblioteca S/MIME ocupa quase a totalidade do tempo de execução, onde esse tempo é muitas vezes superior ao tempo de execução para as mesmas operações através de *OpenPGP*. Se para as operações de decifra e validação de assinatura, os tempos de execução são mais próximos, significa essas operações da biblioteca S/MIME encontram-se mais otimizadas.

Tendo em conta que o DNS é utilizado como meio para publicar chaves públicas OpenPGP e certificados S/MIME é necessário o perceber qual o impacto no seu normal funcionamento, nomeadamente no (*i*) tempo de resolução de chaves púbicas OpenPGP e certificados S/MIME, (*ii*) e qual o impacto desta resolução, no normal funcionamento do DNS, ou seja, na resolução de nomes.

Para (*i*) foram criados ficheiros de zona com várias quantidades de chaves públicas *OpenPGP* (OPENPGPKEY) e certificados S/MIME (SMIMEA). Para as várias quantidades, o tempo de resolução dos registos de recurso por parte do DNS não sofrem qualquer alteração demorando 2 ms [Tabela 4-10].

Registo de Recurso: **OPENPGPKEY**					
Número de chaves no mesmo ficheiro de zona	10	100	1.000	10.000	100.000
Tempo de resolução	2ms	2ms	2ms	2ms	2ms
Registo de Recurso: **SMIMEA**					
Número de chaves no mesmo ficheiro de zona	10	100	1.000	10.000	100.000
Tempo de resolução	2ms	2ms	2ms	2ms	2ms

Tabela 4-10 - Tempo de resolução do DNS para os registos de recurso OPENPGPKEY e SMIMEA

Para (*ii*) a resolução de nomes (do nome `example.org`) foi medida sem carga adicional e sobre vários 4 níveis de carga adicional, relativamente a pedidos de resolução de RRs OPENPGPKEY e SMIMEA [Tabela 4-11].

- **Nível de carga 1:** um processo a resolver múltiplos RRs OPENPGPKEY em simultâneo;
- **Nível de carga 2:** um processo a resolver múltiplos RRs OPENPGPKEY e outro processo a resolver múltiplos RRs SMIMEA em simultâneo;
- **Nível de carga 3:** dois processos a resolverem múltiplos RRs OPENPGPKEY e outro processo a resolver múltiplos RRs SMIMEA em simultâneo;
- **Nível de carga 4:** dois processos a resolver múltiplos RRs OPENPGPKEY e outros dois processos a resolver múltiplos RRs SMIMEA em simultâneo;

Nível de Carga	Tempo de resolução `example.net`
Sem carga	4,4ms
1	7,5ms
2	16,5ms
3	17,6ms
4	17,9ms

Tabela 4-11 - Tempo de resolução de um nome para vários níveis de carga

É possível verificar que o tempo de resolução aumenta, mas tende em estabilizar [Tabela 4-11]. Tendo em conta o ambiente onde foi testada a resolução não simula a resolução recursiva pelos vários servidores com autoridade pelas várias zonas e domínios, sendo a resolução feita entre as máquinas virtuais, a resolução torna-se mais rápida (por exemplo, uma resolução de `google.com` demora aproximadamente 26 ms). Visto que os mecanismos de DKIM, SPF e DMARC geram a resolução dos respetivos RRs na receção de cada mensagem de email, e o filtro de email tem um custo semelhante a nível do número de resoluções ao DNS, indica que a carga adicionada ao DNS pode ser suportada por este.

Em suma, o aumento do tamanho das mensagens e o tempo de execução do filtro de email não trazem impacto para o normal funcionamento do serviço de email. A média do tamanho das mensagens de email enviadas e recebidas é de ~27Kb para mensagens sem anexos e de ~530Kb com anexos [32]. O tamanho das mensagens de email com anexo, assemelham-se com os testados (762Kb com anexo), nomeadamente na segunda coluna de cada tabela, apresentando tempos de execução do filtro inferiores a 0,7s para qualquer operação, e um aumento máximo de 38,2%. Tendo em conta que o tempo máximo que uma mensagem pode estar em processamento no filtro de email no manuseamento do seu cabeçalho e corpo é de 300s, o tempo máximo nos testes realizados é ~28s (cifra S/MIME).

A única preocupação situa-se no aumento do tamanho das mensagens de email S/MIME (37,6% para mensagens de 4272513 Bytes). Sendo que este aumento aparenta estar a descer à medida que o tamanho da mensagem de email aumenta, pode levar à rejeição de emails nos servidores de email destinatários, se por exemplo, estes estiverem configurados para receber apenas mensagens de tamanho máximo 10 Mb. Neste caso, uma mensagem enviada com 9 Mb pelo remetente, ao sofrer um aumento de 37,6% no tamanho original passaria a 12,38 Mb, e rejeitada pelo destinatário. Com isto, o filtro de email deve limitar a sua ação para mensagens de email cujo o tamanho das mensagens de email, acrescido do aumento gerado pelas operações criptográficas, esteja abaixo dos 10Mb (valor mínimo de entre os principiais fornecedores de serviços de email). Deve também dar prioridade à utilização do protocolo *OpenPGP*, que apresenta melhores resultados, tanto a nível desempenho como no aumento que gera no tamanho das mensagens de email.

Capítulo 5 Conclusão e Trabalho Futuro

Neste trabalho, é apresentada uma solução que permite de forma automática e transparente, garantir diversas propriedades de segurança no serviço de email, entre as quais, a confidencialidade para diferentes longitudes na comunicação entre utilizadores, nomeadamente *MTA-to-MTA*, *MTA-to-User* e *User-to-MTA*.

Foi demonstrado que tanto os tempos de execução do filtro de email como o aumento do tamanho das mensagens, para tamanhos médios de mensagens de email, não traz impacto negativo para o normal funcionamento do serviço de email. Esta solução, foi inicialmente pensada para ser integrada em organizações, mas a forma como foi desenhada, com mecanismos como a prioridade no modo criptográfico, injeção de mensagens de email, e ainda o suporte dos principais protocolos de criptografia ponto-a-ponto no serviço de email (*OpenPGP* e S/MIME), pode ser também adotada pelos grandes fornecedores de serviço de email (*Gmail*, *Hotmail*).

No futuro, o comportamento do filtro de email poderá ser avaliado num ambiente de email real, avaliando de uma forma mais precisa os impactos do aumento das mensagens de email, assim como o tempo de execução do filtro de email. Poderá igualmente ser avaliada, a sua integração com os mecanismos de segurança atualmente existentes (DKIM, SPF e DMARC), com foco no desempenho e na procura de possíveis conflitos

Os pedidos ao DNS de chaves públicas *OpenPGP* e certificados S/MIME podem ser feitos sobre DNSSEC. O DNSSEC aumenta a segurança nos pedidos e respostas ao DNS, mas por outro lado, vai adicionar mais tempo execução nos processos de validação. Era importante perceber o impacto a nível de tempo de execução que o DNSSEC adicionaria ao filtro de email.

A plataforma de gestão de chaves poderá ser estendida a utilizadores finais. Ao invés de terem de consultar o DNS por via manual, uma opção legitima para qualquer utilizador, a criação de uma plataforma que permita realizar estas operações de forma transparente, iria possibilitar criar um canal seguro *User-to-User*. Nesta plataforma os utilizadores finais poderiam consultar as chaves públicas e certificados de utilizadores destinatários, e assim cifrar e decifrar as mensagens de email, sem interferência do filtro de email.

Por fim, e para contrariar as consequências do aumento do tamanho das mensagens de email, poderia ser estudada a possibilidade de dividir as mensagens de email em partes, caso o seu tamanho depois de filtrada, exceda o tamanho máximo permitido pelos fornecedores de email ou pelas organizações que façam uso deste mecanismo. Outra possibilidade poderia passar por notificar os remetentes, com uma reinjeção, caso a mensagem, depois de filtrada, exceda o tamanho máximo permitido.

Bibliografia

[1] W. Diffie and M. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory VOL. IT-22, No. 6. November 1976

[2] R.L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM, Volume 21 Issue 2. Feb. 1978

[3] Loren M Kohnfelder - Towards a Practical Public-key Cryptosystem. May, 1978

[4] W. Stalling, L. Brown - Computer Security: Principles and Practice - Second Edition

[5] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280. May 2008. https://tools.ietf.org/html/rfc5280.

[6] S. Young, D. Aitel. The Hackers Handbook – The Strategy behind breaking into and defending networks. 2004

[7] J. B. Postel. Simple Mail Transfer Protocol. RFC 821, Aug. 1982. https://tools.ietf.org/html/rfc821.

[8] J. Klensin. Simple Mail Transfer Protocol. RFC 5321, Oct. 2008. http://tools.ietf.org/html/rfc5321.

[9] N. Freed, N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. RFC 2024. https://tools.ietf.org/html/rfc2045.

[10] J. Klensin, N. Freed, M. Rose, E. Stefferud, D. Crocker. SMTP Service Extensions. RFC 1651. Nov. 1995. http://tools.ietf.org/html/rfc5321.

[11] P. Hoffman. SMTP service extension for secure SMTP over transport layer security. RFC 3207, Feb. 2002. http://www.ietf.org/rfc/rfc3207.

[12] T. Dierks, E. Rescorla. The Transport Layer Security (TLS) Protocol. RFC 4492. Aug. 2008. http://tools.ietf.org/html/rfc5321.

[13] J. Hoffman. ISPs removing their customers email encryption. https://www.eff.org/deeplinks/2014/11/starttls-downgrade-attacks.

[14] J. Callas, H. Finney, D. Shaw, R. Thayer. OpenPGP message format. RFC 4880. Nov. 2007. http://tools.ietf.org/html/rfc4880.

[15] B. Ramsdell, S. Turner. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. RFC 5751. Jan. 2010. https://tools.ietf.org/html/rfc5751.

[16] L. Cranor, G. Simson. Why Johnny Can't Encrypt - A Usability Evaluation of PGP 5.0. 8th USENIX Security Symposium (Washington, D.C., Aug. 23–36, 1999)

[17] Scott Ruoti, Jeff Andersen, Daniel Zappala, Kent Seamons. Why Johnny Still, Still Can't Encrypt: Evaluating the Usability of a Modern PGP Client

[18] Simon L. Garfinkel, Robert C. Miller. Johnny 2: A User Test of Key Continuity Management with S/MIME and Outlook Express. *SOUPS '05 Proceedings of the 2005 symposium on Usable privacy and security*

[19] I. Dacosta, A. Put, B. Decker. EmailCloak: A Practical and Flexible Approach to Improve Email Privacy. 2014 9th International Conference on Availability, Reliability and Security

[20] J. Yeh, F. Zeng, T. Long. P2P email encryption by an identity-based one-way group key agreement protocol. 2014 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS)

[21] B. Adida, S. Hohenberger and Ronald L. Rivest - Lightweight Encryption for Email. *USENIX Association Berkeley, CA, USA 2005*

[22] P. Mockapetris. DOMAIN NAMES - CONCEPTS AND FACILITIES. RFC 1034. *Nov. 1987.* http://tools.ietf.org/html/rfc1034.

[23] P. Mockapetris - DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION - RFC 1035. Nov. 1987. http://tools.ietf.org/html/rfc1035.

[24] R. Arends, R. Austein, M. Larson, D. Massey, S. Rose. DNS Security Introduction and Requirements. RFC 4033. Mar. 2005. http://tools.ietf.org/html/rfc4033.

[25] P. Hoffman, J. Schlyter. The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA. RFC 6698. Aug. 2012. http://tools.ietf.org/html/rfc6698.

[26] D. Crocker, T. Hansen, and M. Kucherawy. Domain Keys Identified Mail (DKIM) signatures. RFC 6379. Sept. 2011. https://tools.ietf.org/html/rfc6376.

[27] S. Kitterman - Sender policy framework (SPF) for authorizing use of domains in email. RFC 7208. Apr. 2014. http://tools.ietf.org/html/rfc7208.

[28] M. Kucherawy and E. Zwicky - Domain-based message authentication, reporting, and conformance (DMARC). RFC 7489. *Mar. 2015*. https://tools.ietf.org/html/rfc7489.

[29] P. Wouters. Using DANE to Associate OpenPGP public keys with email Addresses. May. 02, 2016. https://datatracker.ietf.org/doc/draft-ietf-dane-openpgpkey/

[30] P. Hoffman, J. Schlyter - Using Secure DNS to Associate Certificates with Domain Names for S/MIME Aug. 27, 2015. https://datatracker.ietf.org/doc/draft-ietf-dane-smime/

[31] Milter – Portfix before-queue Milter support**.** http://www.postfix.org/MILTER_README.html

[32] THE RADICATI GROUP, INC. Email Statistics Report, 2009-2013. May 2009. http://www.radicati.com/wp/wp-content/uploads/2009/05/email-stats-report-exec-summary.pdf