ALEXANDRE MACHADO DEUSAJUTE

**PROPOSTA DE UM MECANISMO DE SEGURANÇA
ALTERNATIVO PARA O SIP UTILIZANDO O
PROTOCOLO MASSEY-OMURA APERFEIÇOADO COM
O USO DE EMPARELHAMENTOS BILINEARES**

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia Elétrica.

São Paulo
2010

ALEXANDRE MACHADO DEUSAJUTE

**PROPOSTA DE UM MECANISMO DE SEGURANÇA
ALTERNATIVO PARA O SIP UTILIZANDO O
PROTOCOLO MASSEY-OMURA APERFEIÇOADO COM
O USO DE EMPARELHAMENTOS BILINEARES**

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia Elétrica.

Área de Concentração:

Sistemas Digitais

Orientador:

Paulo Sérgio Licciardi Messeder Barreto

São Paulo
2010

Para Márcia e Lucas.

AGRADECIMENTOS

Primeiramente a Deus, por ter me guardado e guiado, de diversas formas, em incontáveis vezes, me provendo sabedoria, discernimento, saúde e equilíbrio.

Ao professor Paulo Barreto, pela oportunidade dada não só de fazer um mestrado em uma das maiores e mais renomadas universidades do Brasil, mas também pelo privilégio de tê-lo como orientador, um especialista em criptografia (notadamente, em curvas elípticas e emparelhamentos bilineares). E ainda, pela paciência e disposição em me passar os ensinamentos necessários sobre emparelhamentos, curvas elípticas e outros relacionados com a criptografia em geral, além de valiosas experiências compartilhadas comigo.

Ao professor José Roberto Amazonas, pelo suporte teórico e apoio acadêmico referente ao tema VoIP e redes de computadores em geral, suporte e apoio estes dados em diversas vezes que recorri a ele.

A João Lucareski Filho, pela oportunidade de trabalhar mais diretamente com Segurança da Informação, ampliando minha visão sobre esse assunto e melhorando minhas habilidades, conhecimento e experiência no mesmo, o que acabou refletindo em passagens relevantes do presente trabalho.

Ainda que fora do âmbito acadêmico e sem relação direta com o conteúdo dessa dissertação, não posso deixar de fazer importantes agradecimentos.

À minha esposa Márcia, pelo carinho, amor, dedicação inigualáveis, e por ter agido como um verdadeiro “*firewall*”, abstraindo-me das preocupações e dificuldades do dia-a-dia ao longo desses 3 anos, o que me permitiu cumprir as diversas fases do programa de mestrado e, principalmente, compor este trabalho.

À minha mãe Milza Lúcia por todo o apoio e pela formação moral e intelectual, as quais têm sido imprescindíveis para alcançar meus objetivos pessoais, profissionais e acadêmicos.

Ao meu antigo gestor – na época, do Unibanco – Antônio Carlos Cardoso Gondim, por ter autorizado e, principalmente, incentivado a fazer o mestrado.

A José Geraldo Leite Coura, Renato Jorge Galvão Teixeira e Augusto Henrique Bernardo de Aguiar, pelos sempre produtivos debates e trocas de idéias acerca de criptografia e Segurança da Informação como um todo.

“Fortitudine Vincimus” (By endurance we conquer)

*Lema da família de sir Ernest Henry Shackleton,
explorador anglo-irlandês.*

RESUMO

Voz sobre IP (ou VoIP) vem sendo adotada progressivamente não apenas por um grande número de empresas mas também por um número expressivo de pessoas, no Brasil e em outros países. Entretanto, essa crescente adoção de VoIP no mundo traz consigo algumas preocupações tais como ameaças e riscos de segurança, sobretudo no que diz respeito à autenticidade, privacidade e integridade da comunicação. Para proteger a sessão de mídia existem protocolos muito eficientes, como o *Secure Real-time Transport Protocol* (SRTP). Mas ele depende de uma chave secreta para tornar a comunicação segura de fato. Assim, uma boa estratégia é aproveitar o processo de sinalização que estabelece a sessão de mídia e negociar uma chave secreta de sessão que seja comum às partes comunicantes. Esse processo de sinalização é realizado por tipos específicos de protocolo tais como o *Session Initiation Protocol* (SIP), um protocolo de sinalização muito importante e que vem sendo usado cada vez mais por *softphones* para comunicação na Internet. Todavia, os riscos e ameaças mencionados já existem no próprio processo de sinalização e, dentre eles, o ataque do tipo *man-in-the-middle* é o mais perigoso, devido aos prejuízos que ele pode causar. Depois de fazer uma revisão bibliográfica dos riscos e ameaças inerentes ao SIP, bem como de seus mecanismos de segurança (analisando os pontos fortes e de atenção deles), foi possível originar um novo mecanismo de segurança, o qual é apresentado neste trabalho. O mecanismo proposto usa um protocolo para troca segura de informações – o protocolo Massey-Omura – o qual, quando combinado com emparelhamentos bilineares, provê ao SIP um melhor nível de segurança em todos os aspectos (autenticidade, privacidade e integridade). Além disso, o novo mecanismo é avaliado através de uma prova de conceito, na qual utilizou-se um *softphone* SIP funcional. A análise de segurança realizada e os resultados obtidos da prova de conceito fazem do mecanismo de segurança proposto uma alternativa viável para o SIP.

Palavras-chave: VoIP. SIP. Massey-Omura. Emparelhamentos bilineares. Assinatura baseada em identidade.

ABSTRACT

Voice over IP (or VoIP) has been progressively adopted not only by a great number of companies but also by an expressive number of people, in Brazil and in other countries. However, this increasing adoption of VoIP in the world brings some concerns such as security risks and threats, mainly on the authenticity, privacy and integrity of the communication. In order to protect the media session, efficient protocols like the Secure Real-time Transport Protocol (SRTP) have been used. However, it depends on a secret key to make the communication secure. Thus, a good strategy is to take advantage of the signaling process to establish the media session, and agree on a common secret session key between the communicating parties. This signaling process is performed by specific types of protocols such as the Session Initiation Protocol (SIP), a very important signaling protocol, which has been used more and more by softphones in the Internet communication. Nevertheless, those risks and threats already exist in the own signaling process and, among them, the man-in-the-middle attack is the worst of all due to its high danger degree. After doing a bibliographical revision of the SIP security risks and threats, as well as its security mechanisms (analyzing their advantages and drawbacks), it was possible to generate a new security mechanism, which is presented in this work. The proposed mechanism uses a protocol for secure information exchange – the Massey-Omura protocol – which, when combined with bilinear pairings, provides a better security level for SIP in all its aspects (authenticity, privacy and integrity). Besides this, the new mechanism is evaluated by a proof of concept, in the which a functional SIP softphone was used. The security analysis and the results obtained from the proof of concept, make the proposed security mechanism a viable alternative for SIP.

Keywords: VoIP. SIP. Massey-Omura. Bilinear pairings. Identity-based signature.

LISTA DE FIGURAS

1	Mensagem SIP transportada por UDP em uma rede IP sobre Ethernet – adaptado de (TANENBAUM, 2003)	28
2	Exemplo de mensagem SIP típica	29
3	Processo de registro de um <i>user agent</i> – adaptado de (SINNREICH; JOHNSTON, 2006, p. 124)	32
4	Processo de autenticação junto a um servidor <i>proxy</i> SIP – adaptado de (SINNREICH; JOHNSTON, 2006, p. 131)	33
5	Modo básico de sinalização SIP: comunicação direta entre dois <i>user agents</i> – adaptado de (SINNREICH; JOHNSTON, 2006, p. 112)	34
6	Modo de sinalização SIP com servidor <i>proxy</i> – adaptado de (SINNREICH; JOHNSTON, 2006, p. 116)	35
7	Modo de sinalização SIP com mais de um servidor <i>proxy</i> envolvido – Trapezoide SIP	36
8	Modo de sinalização SIP com redirecionamento – adaptado de (DOUSKALIS, 2000, p. 78)	37
9	Utilização do S/MIME em mensagens SIP – extraído de (ROSEMBERG et al., 2002, p. 205)	40
10	O protocolo Massey-Omura – visão conceitual	58
11	Ataque do tipo <i>man-in-the-middle</i> sobre o protocolo Massey-Omura, de modo que a mensagem secreta trocada entre as partes seja descoberta sem que estas suspeitem de nada	63

12	O protocolo Massey-Omura na versão com curvas elípticas (ECMO), melhorado, em termos de segurança, com o uso de assinaturas baseadas em identidade	70
13	Diagrama de classe UML do protocolo EECMO	76
14	Visão conceitual da relação façade EECMO – protocolo EECMO – protocolo ECMO	79
15	O elemento “key-mgmt-data” idealizado para o EECMO em conformidade com a RFC 4567	84
16	O protocolo EECMO acoplado ao SIP – as partes comunicantes estão sob domínio da mesma TA – visão conceitual	86
17	O protocolo EECMO acoplado ao SIP – as partes comunicantes estão sob domínio de TAs diferentes	87
18	Arquitetura funcional do <i>softphone</i> Endurance – adaptado de (PEREA, 2008)	95
19	Arquitetura funcional do <i>softphone</i> Endurance mostrando as interações entre as camadas do <i>softphone</i> – adaptado de (PEREA, 2008)	96
20	Cenário do teste funcional realizado com o <i>softphone</i> Endurance	97
21	Interface do <i>softphone</i> Endurance para permitir configurações diversas	98
22	Interface do <i>softphone</i> Endurance para permitir comunicação	99
23	Sinalização SIP através do <i>softphone</i> Endurance	100
24	O <i>softphone</i> Endurance configurado automaticamente após a conclusão, com sucesso, de um processo de sinalização SIP utilizando o EECMO102	
25	A métrica <i>call set-up delay</i> (CSD) para o SIP – extraído de (INTERNATIONAL TELECOMMUNICATION UNION, 2007, p. 6)	108

26	A métrica <i>call set-up delay</i> (CSD) para o SIP, considerando o uso do EECMO	109
27	Exemplo de desafio (a) e resposta (b) no contexto do mecanismo HTTP Digest Authentication aprimorado pelo uso de encriptação baseada em identidade – extraído de (RING et al., 2006)	142
28	Utilização de encriptação baseada em identidade para geração de uma chave secreta comum entre dois <i>user agents</i> SIP	143
29	SAKA: mecanismo seguro para autenticação e negociação de chaves para o SIP – adaptado de (WANG; ZHANG, 2008)	145
30	(a) O esquema SAKA; (b) O esquema SAKA melhorado; (c) O esquema MAKP – extraídos de (LUO; WEN; ZHAO, 2008)	148
31	Ataque do tipo <i>man-in-the-middle</i> sobre o ECMO onde a mensagem secreta é obtida diretamente pelo atacante	159
32	Ataque do tipo <i>man-in-the-middle</i> sobre o ECMO onde a mensagem secreta é obtida indiretamente pelo atacante	160
33	Ataque do tipo <i>man-in-the-middle</i> sobre o ECMO onde o atacante consegue passar uma mensagem definida por ele mesmo, diretamente para Bob	160
34	Ataque do tipo <i>man-in-the-middle</i> sobre o ECMO onde o atacante consegue passar uma mensagem definida por ele mesmo, indiretamente para Bob	161
35	Ataque do tipo <i>man-in-the-middle</i> sobre o ECMO onde a mensagem secreta trocada entre as partes é descoberta sem que elas suspeitem de algo	162

LISTA DE TABELAS

1	Comparação dos mecanismos de segurança do SIP que constam nas RFCs 3261, 3830, 4572, 4650 e 4738 em relação a quesitos de segurança e operacionais	55
2	Tempos de processamento obtidos no teste funcional da prova de conceito (valores expressos em milissegundos)	115
3	Tempo médio de processamento despendido com as operações exclusivamente EECMO (valores expressos em milissegundos)	117
4	<i>Call Set-up Delay</i> estimado para o SIP utilizando o EECMO em situações diferentes (valores expressos em milissegundos)	118
5	Tamanho máximo dos “key-mgmt-attribute” que transitam entre os <i>user agents</i> (valores expressos em <i>bytes</i>)	119
6	Sobrecarga causada nas mensagens SIP por conta do acoplamento do EECMO (valores expressos em <i>bytes</i>)	119
7	Sobrecarga causada nas mensagens SIP por conta do acoplamento do EECMO – visão percentual específica da prova de conceito realizada .	119

LISTA DE SIGLAS

AES *Advanced Encryption Standard*

BKLS *Barreto-Kim-Lynn-Scott*

BLMQ *Barreto-Libert-McCullagh-Quisquater*

CSD *call set-up delay*

DoS *Denial-of-Service*

DHCP *Dynamic Host Configuration Protocol*

DHMAC *Diffie-Hellman with keyed-Hash Message Authentication Code*

DNS *Domain Name Service*

ECMO *Elliptic Curve Massey-Omura*

EECMO *Enhanced Elliptic Curve Massey-Omura*

IBE *Identity-Based Encryption*

IBS *Identity-Based Signature*

IETF *Internet Engineering Task Force*

HMAC *keyed-Hash Message Authentication Code*

HSM *Hardware Security Module*

HTTP *Hyper Text Transfer Protocol*

IANA *Internet Assigned Numbers Authority*

ICP *Infraestrutura de Chaves Públicas*

IKE *Internet Key Exchange*

ITU-T *Telecommunication Standardization Sector of International Telecommunications Union*

JAIN *Java Integrated Network*

JMF *Java Media Framework*

LAN *Local Area Network*

MIKEY *Multimedia Internet KEYing*

MITM *Man-In-The-Middle*

MNT *Miyaji-Nakabayashi-Takano*

MTU *Maximum Transmission Unit*

NIST *National Institute of Standards and Technology*

P2P *peer-to-peer*

PDD *post-dial delay*

PGP *Pretty Good Privacy*

PLD *Problema do Logaritmo Discreto*

PLDCE *Problema do Logaritmo Discreto para Curvas Elípticas*

PPD *post-pickup delay*

RFC *Request for Comments*

RTP *Real-time Transport Protocol*

S/MIME *Secure/Multi-purpose Internet Mail Extensions*

SDP *Session Description Protocol*

SHA-1 *Secure Hash Algorithm One*

SIP *Session Initiation Protocol*

SRTP *Secure Real-time Transport Protocol*

TA *Trusted Authority*

TCP *Transport Control Protocol*

TLS *Transport Layer Security*

ToIP *Text over IP*

UA *user agent*

UAC *user agent client*

UAS *user agent server*

UDP *User Datagram Protocol*

URI *Uniform Resource Indicator*

VoIP *Voice over Internet Protocol*

SUMÁRIO

1	Introdução	15
1.1	Objetivo	17
1.2	Motivação	18
1.3	Justificativa	19
1.4	Metodologia	20
1.5	Contribuições	22
1.6	Organização	22
2	SIP	24
2.1	Fundamentos	24
2.1.1	Apresentação	24
2.1.2	Interação com outros protocolos	25
2.1.3	Componentes fundamentais do protocolo	26
2.1.4	Endereçamento	31
2.1.5	Operação do protocolo	32
2.2	Segurança no contexto do SIP	35
2.2.1	SIP como facilitador da segurança em VoIP	35
2.2.2	Mecanismos para prover segurança ao SIP	37
2.2.3	Análise dos mecanismos de segurança	44

2.2.4	Discussão acerca dos mecanismos de segurança	50
3	EECMO	56
3.1	O protocolo Massey-Omura	56
3.1.1	Versão do protocolo com curvas elípticas – ECMO	59
3.1.2	Discussão sobre a segurança do protocolo original	61
3.2	Aplicação de assinaturas baseadas em identidade para melhorar a segurança do ECMO	65
3.2.1	Discussão sobre a segurança do novo protocolo	69
3.3	Implementação do EECMO	75
3.3.1	Cenários de TA contemplados na implementação	79
4	A segurança do SIP melhorada com o protocolo EECMO	82
4.1	Acoplamento do EECMO ao SIP	82
4.1.1	SIP e a segurança provida pelo EECMO	87
4.2	Prova de Conceito	93
4.2.1	O <i>softphone</i> Endurance	93
4.2.2	Execução da prova	95
4.2.2.1	Refinamento dos valores obtidos	104
4.2.3	Avaliação e discussão acerca do desempenho	107
4.2.3.1	Tempo de processamento	107
4.2.3.2	Sobrecarga na mensagem SIP	112
5	Conclusão	120

Referências	124
Apêndice A - Emparelhamentos bilineares	132
A.1 Teoria	132
A.2 Tipos de emparelhamento	136
A.3 Prática	138
A.4 Aplicações	139
Apêndice B - Aplicações envolvendo emparelhamentos bilineares e SIP	141
B.1 Autenticação e negociação de chaves SIP a partir de IBE	141
B.2 Autenticação e negociação de chaves SIP baseados em CL-PKC	144
B.2.1 Aperfeiçoamento do esquema	146
B.3 Assinaturas baseadas em identidade para redes SIP P2P	149
Apêndice C - Redução das possibilidades de ataque do tipo <i>man-in-the-middle</i> ao ECMO	151
C.1 Lógica da redução	152
C.2 Cenários de ataque efetivos	158
Apêndice D - Publicações do autor	164

1 INTRODUÇÃO

Com o advento das tecnologias para Internet e para conectividade de dados em alta velocidade (sobretudo nas empresas), uma nova tecnologia começou a despo-
tar, surgindo para revolucionar as comunicações telefônicas: Voz sobre IP – do inglês, *Voice over Internet Protocol* (VoIP). Essa tecnologia tem um apelo significativo tanto para empresas quanto para provedores de serviço e usuários finais, pois torna pos-
sível que a Internet e as redes de dados em geral (tais como aquelas existentes em escritórios, fábricas, universidades etc.) transformem-se em meios de transporte não apenas para chamadas de voz, mas também para outras aplicações de tempo real relacionadas com multimídia.

Ao utilizar VoIP, a conversação passa a ser convertida em pacotes de dados, os quais fluem através de redes interligadas. Dessa forma, não é necessária uma infraes-
trutura dedicada de circuitos comutados, como no caso das redes públicas de telefonia convencional – do inglês, *Public Switched Telephone Network* (PSTN). Basta simples-
mente que o meio de comunicação seja compatível com o protocolo IP (NETWORKS, 2008; WALLINGFORD, 2005).

Assim, VoIP vem sendo adotada progressivamente por um grande número de em-
presas para substituir a infraestrutura tradicional de circuitos comutados por uma nova infraestrutura de telefonia IP. Muitos provedores de serviço estão buscando melho-
rar a capacidade deles para transferência de mensagens por meios eletrônicos atra-
vés dessa nova infraestrutura, ao invés de investirem na infraestrutura tradicional. Ao

mesmo tempo, o desenvolvimento da telefonia IP provê oportunidades para a introdução de novos serviços agregados tais como áudio-conferência, trabalho colaborativo através da *web* e jogos *online* (CAO; MALIK, 2005).

Contudo, devido ao fato de utilizarem redes IP normais, as aplicações VoIP herdam os pontos fracos de segurança – conhecidos e desconhecidos – que estão associados ao protocolo IP. As chamadas realizadas com VoIP estão suscetíveis a ataques que resultam no acesso ilegal a sistemas (levando a chamadas gratuitas não autorizadas), na escuta e redirecionamento malicioso de chamadas, no *SPAM over IP Telephony* (SPIT), além de ataques de negação de serviço – do inglês, *Denial-of-Service* (DoS) – e do tipo *man-in-the-middle* (KIM; KIM, 2007; QUINTEN; MEENT; PRAS, 2007). Consequentemente, os usuários de VoIP herdam as preocupações de segurança inerentes à Internet e aos serviços disponibilizados por ela.

Do ponto de vista do usuário final de uma comunicação VoIP o que importa mesmo é a segurança do que está sendo transmitido (áudio, vídeo, texto etc.), ou seja, a segurança da mídia em si. Como a mídia é transportada pelo *Real-time Transport Protocol* (RTP) a escolha mais natural para prover segurança à transmissão é o *Secure Real-time Transport Protocol* (SRTP).

O SRTP utiliza o *Advanced Encryption Standard* (AES), reconhecido mundialmente como o algoritmo simétrico mais seguro e eficiente da atualidade. Porém, o AES depende de uma chave simétrica de criptografia. Isto significa que, antes de uma sessão de mídia poder ser estabelecida pelo RTP, e protegida pelo SRTP, é necessário que ambas as partes comunicantes compartilhem, de uma forma segura e eficiente, uma chave de sessão secreta comum.

Como antes da sessão de mídia há um processo de sinalização para o estabelecimento da comunicação, uma boa estratégia, senão a melhor, é procurar aproveitar-se desse processo de sinalização para negociar a chave secreta comum.

Ao longo de vários anos tem-se tentado fazer isso através do *Session Initiation Protocol* (SIP), um dos protocolos de sinalização VoIP mais utilizados e que vem sendo largamente adotado na Internet, até porque foi projetado pensando-se especificamente nesta rede. Porém, o SIP possui diversos problemas de segurança. A própria *Request for Comments* (RFC) 3261 – que é a RFC vigente do SIP – não é muito “otimista” quanto aos aspectos de segurança (incluindo os mecanismos de proteção previstos para o protocolo) ao afirmar que “SIP não é um protocolo fácil para aplicar segurança. O uso que o protocolo faz de intermediários, de relações de confiança com muitas variantes, e de operações usuário-à-usuário, tornam a segurança algo longe do trivial.” (ROSEMBERG et al., 2002, p. 231, tradução nossa).

Por ter sido concebido conceitualmente a partir dos protocolos *Hyper Text Transfer Protocol* (HTTP) e *Simple Mail Transfer Protocol* (SMTP), e por ser um protocolo textual e simples, o SIP oferece muitos desafios de segurança, principalmente quanto à autenticidade, confidencialidade e integridade. A carência destes serviços básicos da segurança da informação dificulta bastante utilizar o SIP para realizar uma negociação segura de chaves criptográficas, de modo que as mesmas possam ser utilizadas, depois, para proteger a sessão de mídia.

1.1 Objetivo

O objetivo deste trabalho é propor um mecanismo de segurança alternativo para o SIP – intencionalmente baseado em emparelhamentos bilineares, para mostrar a viabilidade de aplicação deste recurso criptográfico no contexto SIP – de modo que seja possível, durante o processo de sinalização para estabelecimento da sessão de mídia, realizar uma negociação segura e eficiente de chave secreta, no intuito de, posteriormente, utilizá-la para assegurar a sessão de mídia, seja através do SRTP ou de outro protocolo de segurança similar.

1.2 Motivação

Muitas RFCs complementares e trabalhos isolados – na maioria das vezes, propostas ao *Internet Engineering Task Force* (IETF) – têm surgido ao longo dos anos, visando melhorar a segurança do SIP em diversos pontos. Sobretudo para que a sinalização possa ser utilizada para negociar, de forma segura, chaves secretas de sessão de mídia.

Entretanto, os mecanismos de segurança apresentados ao longo dos anos ou não cobrem todos os pontos de segurança necessários ou dependem de processos como pré-combinar/compartilhar chaves secretas. Ou ainda, de condições externas favoráveis como a existência de uma infraestrutura de chaves públicas e certificados digitais. Ainda que justificáveis – principalmente no que diz respeito a prover autenticidade – trazem certa complexidade e preocupações adicionais de segurança a mecanismos que deveriam ser tão simples quanto o processo de sinalização. Além disso, os mecanismos de segurança propostos para o SIP, sejam os originais ou aqueles considerados evoluções, não trazem grandes inovações, seguindo uma mesma linha de pensamento com algumas pequenas melhorias apenas.

Por outro lado, há poucos trabalhos propondo a aplicação de emparelhamentos bilineares como alternativa para aperfeiçoar a segurança do SIP. E estes poucos trabalhos acabam ficando restritos ao modelo conceitual, não explorando os aspectos práticos, de tal forma que não dá para saber se eles são aplicáveis em situações reais, muito menos se é viável aplicar emparelhamentos no contexto do SIP.

Assim, a busca por um mecanismo de segurança alternativo para o SIP, que aproveitasse o processo de sinalização para negociar seguramente uma chave secreta, sem recorrer a processos ou condições externas complexas, e fazendo uso de emparelhamentos bilineares, de modo a investigar, experimentar, e mostrar a viabilidade da aplicação deles, em termos práticos, no contexto SIP, motivaram a realização do

trabalho descrito nessa dissertação.

1.3 Justificativa

As pesquisas e estatísticas apontam um cenário atual de expansão e adoção da telefonia IP. Em 2010, estima-se que cerca de 25% de todas as famílias na Europa Ocidental terão abandonado os serviços da rede pública de telefonia devido à adoção de VoIP (ELECTRICNEWS.NET, 2005). No Brasil, estima-se que existiam em 2008 aproximadamente 2 milhões de assinantes VoIP, quase o triplo do existente em 2007 – cerca de 700 mil assinantes. Este número, por sua vez, já era também quase o triplo de assinantes ao final de 2006 – 262 mil. Além disso, os provedores de chamadas VoIP provocaram uma queda no preço do minuto das chamadas internacionais da Embratel e isso deve continuar impulsionando ainda mais a procura por VoIP (TELECO, 2009).

Por outro lado, as pesquisas e previsões quanto à segurança no contexto de VoIP não são tão animadoras. É esperado que os chamados “criminosos cibernéticos” voltem-se cada vez mais para VoIP, engajados em fraudes de voz, roubo de dados, negação de serviço, execução remota de códigos maliciosos, *botnets*, *phishing* de voz, *SPAM over IP Telephony* etc. Logo, faz-se necessário a existência de sistemas de prevenção a intrusão na camada de transporte VoIP, bem como segurança fim-a-fim em telefones SIP e outros dispositivos VoIP (GEORGIA TECH INFORMATION SECURITY CENTER, 2008).

Por isso, a crescente adoção de VoIP no mundo associada à necessidade cada vez maior de segurança na telefonia IP – ainda mais em situações nas quais não há mecanismos de segurança aprimorados, como no caso da sinalização através do SIP – justificam as pesquisas e esforços para novas propostas de segurança.

No caso particular do SIP, pesquisas envolvendo a segurança do protocolo são altamente justificadas também devido a outras frentes nas quais o protocolo pode ser

utilizado: *Text over IP* (ToIP) – um meio para prover o serviço de conversação por texto em tempo real, complementando voz e vídeo sobre redes IP¹ – e redes SIP P2P (P2P SIP). Esta última é uma tendência recente e que alguns extremamente otimistas afirmam: “Redes SIP P2P farão para VoIP o que VoIP tem feito para rede pública de telefonia convencional.” (SINNREICH; JOHNSTON, 2006, p. 340, tradução nossa).

Quanto a emparelhamentos, ainda que eles já venham sendo utilizados com sucesso para resolver problemas de segurança em muitas aplicações práticas e reais, existem poucas propostas de aplicação deles no contexto SIP, o que justifica mais pesquisas nessa linha. Além disso, os trabalhos existentes carecem de uma experimentação prática, ainda que fosse um simples ensaio, de modo investigar se emparelhamentos constituem um recurso criptográfico viável ou não para o contexto SIP e para os padrões de comunicação através de VoIP.

1.4 Metodologia

A partir de RFCs e de trabalhos obtidos em portais reconhecidos como importantes fontes de referência acadêmica – além de buscas livres na Internet – foi feita uma revisão bibliográfica objetivando identificar as vulnerabilidades do SIP e avaliar os mecanismos de segurança que o mesmo dispõe para tentar resolvê-las. Além disso, pesquisou-se os trabalhos envolvendo SIP e emparelhamentos bilineares, no intuito de verificar o quanto se tinha investigado e experimentado em relação a esses dois temas combinados.

A revisão bibliográfica acabou norteando a busca por um mecanismo de segurança alternativo capaz de permitir o protocolo negociar, de maneira segura, uma chave secreta de sessão, a qual seria utilizada posteriormente para prover segurança à sessão

¹ToIP difere de mensagens instantâneas em um ponto crucial, o da conversação. Enquanto em mensagens instantâneas é possível dois comunicantes escreverem ao mesmo tempo, com ToIP, ou há uma sobreposição ou há uma interrupção de alguma das partes, como em uma conversação comum por voz.

de mídia.

Por observação, foi identificada uma coincidência na dinâmica de transições do SIP e de protocolos criptográficos de três passos, tornando-os, assim, candidatos ao mecanismo de segurança pretendido. Dentre esses protocolos, foi considerado o uso do Massey-Omura, em razão de experiência anterior com este protocolo (DEUSAJUTE; BARRETO, 2007).

Em uma nova revisão bibliográfica, dessa vez contemplando as vulnerabilidades do protocolo Massey-Omura e formas de resolvê-las, descobriu-se que o uso de uma assinatura digital poderia prover autenticidade, integridade e confidencialidade ao protocolo – e, conseqüentemente, ao SIP. Como pretendia-se experimentar a aplicação de emparelhamentos bilineares no contexto SIP, optou-se pelo uso de assinaturas baseadas em identidade, pois as mesmas utilizam emparelhamentos. Então, foi elaborado um modelo teórico do novo mecanismo de segurança proposto.

Para mostrar a viabilidade do novo mecanismo em termos de segurança, foi feita uma análise dedutiva de segurança sobre o modelo teórico elaborado. Já em termos práticos foi necessário executar algumas implementações, as quais subsidiaram a prova de conceito que foi realizada. Nessa prova, utilizou-se o *User Datagram Protocol* (UDP) como protocolo de transporte, pois o mesmo é mais apropriado para o SIP, conforme indicado em (SINNREICH; JOHNSTON, 2006, p. 111).

A partir dessa prova de conceito foram obtidos valores correspondentes ao tempo despendido com o processamento nos *endpoints*. Também foram obtidos valores referentes à sobrecarga gerada nas mensagens SIP. Houve ainda um refinamento de modo a obter valores que fossem estatisticamente mais apropriados.

Os valores refinados foram, então, analisados no intuito de determinar o impacto do mecanismo de segurança proposto no tempo de sinalização e no tamanho das mensagens SIP. Para a análise de impacto envolvendo tempo, considerou-se as métricas

call set-up delay (CSD), *post-dial delay* (PDD) e *post-pickup delay* (PPD) – para as duas últimas, ainda foram considerados os valores alvo recomendados para elas. No caso da análise envolvendo tamanho das mensagens, foram utilizadas como métricas o tamanho máximo que um datagrama UDP deve ter para transportar uma carga útil sem que ocorra fragmentação e a *Maximum Transmission Unit* (MTU) de Ethernet.

1.5 Contribuições

- Aperfeiçoamento da segurança do protocolo Massey-Omura, especificamente a versão com curvas elípticas.
- Proposta de um mecanismo de segurança alternativo para o SIP.
- Experimentação prática da aplicação de emparelhamentos bilineares no contexto do SIP, mostrando o quão viável é o emprego desse recurso criptográfico nesse contexto e em VoIP como um todo.

1.6 Organização

Além do presente capítulo introdutório, esta dissertação está organizada conforme descrito a seguir.

No Capítulo 2 é mostrada uma visão geral do SIP, abordando objetivamente os fundamentos do protocolo e indicando como os mecanismos de segurança dele podem ser utilizados para prover segurança no contexto de VoIP. O capítulo ainda discute sobre os pontos fortes e de atenção desses mecanismos.

No Capítulo 3 é apresentado o protocolo Massey-Omura, a versão elíptica do mesmo e como a segurança dessa versão pôde ser melhorada com a aplicação de emparelhamentos bilineares, evoluindo para um esquema que pudesse ser utilizado em situações específicas. O capítulo ainda descreve como a versão melhorada do

protocolo foi implementada.

O Capítulo 4 detalha como o protocolo Massey-Omura, melhorado com emparelhamentos, pôde ser acoplado ao SIP no intuito de torná-lo mais seguro. Além disso, objetivando avaliar o novo mecanismo de segurança proposto para o SIP, o capítulo mostra a prova de conceito que foi realizada, na qual utilizou-se um *softphone* SIP criado especialmente para ela. Os resultados dessa prova de conceito também são mostrados no capítulo.

No Capítulo 5 são apresentadas as conclusões e trabalhos futuros.

O documento ainda possui os seguintes Apêndices:

- Apêndice A: apresenta os fundamentos mínimos de emparelhamentos bilineares, necessários para que um leitor menos familiarizado com esse tema possa entender determinadas passagens dessa dissertação, sobretudo no Capítulo 3.
- Apêndice B: apresenta os trabalhos relacionados envolvendo emparelhamentos bilineares e SIP.
- Apêndice C: discute como a segurança da versão elíptica do protocolo Massey-Omura pode ser melhorada assumindo-se uma premissa simples. Ainda que isso não resolva todos os cenários de ataque ao protocolo, reduz bastante as possibilidades de ataques efetivos.
- Apêndice D: descreve, de maneira sucinta, as publicações do autor que tornaram-se contribuições relevantes para o meio acadêmico e para a criptologia.

2 SIP

Este capítulo mostra uma visão geral do SIP, abordando objetivamente os fundamentos¹ do protocolo e como ele pode ser utilizado como facilitador para prover segurança no contexto de VoIP.

2.1 Fundamentos

2.1.1 Apresentação

O processamento de dados em VoIP consiste das seguintes etapas segundo (BUTCHER; LI; GUO, 2007): sinalização (criação e gerenciamento de conexões ou chamadas entre os participantes da comunicação), codificação e transporte (conversão de sinais analógicos de voz em sinais digitais, compressão de dados na maioria das vezes, e empacotamento destes dados para serem transmitidos por algum protocolo específico para transporte de mídia, como o RTP), e controle de *gateway* (garantia da conversação em tempo real, mesmo que através de redes diferentes como, por exemplo, redes IP e redes convencionais de telefonia).

Na etapa de sinalização, os participantes são contatados para negociarem determinadas características de modo a efetivar o estabelecimento da sessão. Também ocorre a troca de informações sobre a mídia que estará presente na sessão a ser estabelecida. Através da sinalização, é possível ainda modificar sessões de mídia já

¹Exceto por alguma eventual citação específica que se fizer necessária, o conteúdo da seção 2.1 foi baseado em (DOUSKALIS, 2000), (PEREA, 2008), (ROSEMBERG et al., 2002), e (SINNREICH; JOHNSTON, 2006).

estabelecidas e também desfazer estas sessões.

O SIP é um protocolo que atua nessa etapa de sinalização. Ele é cliente-servidor, baseado em texto, usado para criar e controlar sessões de multimídia com dois ou mais participantes. Sob um outro ponto de vista, SIP é um protocolo fim-a-fim na camada de aplicação. Essa abordagem fim-a-fim para projeto de aplicações visando a comunicação provê grande flexibilidade e reduz o *time to market*, pois diminui a dependência da rede.

O protocolo foi criado originalmente por Henning Schulzrinne e Mark Handley, em 1996, mas foi descrito formalmente como padrão do IETF somente em 1999, na RFC 2543 (HANDLEY et al., 1999). Esta RFC foi atualizada pela RFC 3261, sendo esta a RFC oficial mais recente do SIP.

Sessões SIP podem ser estabelecidas entre participantes localizados em uma mesma rede local ou através de múltiplos segmentos de rede, já que o SIP pode trabalhar integrado com *gateways* que proveem sinalização e permitem a tradução de mídia através de segmentos de rede diferentes. SIP também possui suporte para mobilidade e pode ser utilizado em aplicativos de mensagens instantâneas, vídeo-conferências e jogos *online*.

Por isso, SIP tem alcançado significativa penetração no mercado, pois ele é ao mesmo tempo rico, robusto e simples.

2.1.2 Interação com outros protocolos

SIP é um protocolo da camada de aplicação que necessita de um protocolo para viabilizar o transporte da sinalização. As duas principais opções para isto são: *Transport Control Protocol* (TCP) ou UDP², sendo mais comum encontrar implementações SIP sobre UDP em razão da simplicidade e de velocidade (SINNREICH; JOHNSTON, 2006, p.

²Há algumas literaturas que indicam a possibilidade de uso do SIP com outros protocolos de transporte menos utilizados, como o *Stream Control Transmission Protocol* (SCTP).

111). A porta padrão utilizada pelo SIP é a 5060.

Tanto o TCP quanto o UDP utilizam o protocolo IP na camada de Internet que, por sua vez, pode utilizar qualquer recurso disponível na camada de enlace de dados e na camada física. Mas este nível de condução dos dados já passa a ser transparente para o SIP. Consequentemente, o tipo de endereçamento IP (IPv4 ou IPv6, com ou sem IPsec) passa a ser transparente também. Essa transparência com relação ao protocolo IP permite que o SIP se beneficie de outro recurso muito importante, o qual faz o mapeamento de nomes simbólicos (inteligíveis e mais amigáveis de se utilizar) em endereços IP: o *Domain Name Service* (DNS).

Como o SIP controla o início, a manutenção e o término de uma sessão de mídia, é natural que haja interação com outros protocolos utilizados durante tal sessão como o RTP e o *RTP Control Protocol* (RTCP), este último um protocolo que normalmente trabalha junto ao RTP para monitorar a qualidade das chamadas e obter dados estatísticos.

Mas em termos de interação com outros protocolos, certamente a interação com o *Session Description Protocol* (SDP) é a mais importante, como será visto adiante.

2.1.3 Componentes fundamentais do protocolo

Podemos destacar os seguintes componentes fundamentais do protocolo: os *user agents*, os servidores SIP e as mensagens SIP que são trocadas entre os *user agents*, passando por um ou mais servidores SIP.

User agents: em um processo de sinalização SIP cada uma das partes comunicantes é chamada de *user agent* (UA)³. Um UA toma uma instrução ou informação fornecida por um usuário real e atua como um agente em favor desse usuário para es-

³Também chamado de *endpoint*. Alguns autores dão a entender que *user agent* e *endpoint* são conceitualmente diferentes. Outros autores indicam que um *user agent* estaria contido em um *endpoint*. Segundo consta na página 8 da RFC 3261, *user agent* e *endpoint* são conceitualmente iguais e podem ser tratados como sinônimo um do outro.

tabelecer, modificar (quando for o caso), e desfazer sessões de mídia com outros *user agents*, os quais, por sua vez, estão representando outros usuários reais também. Um UA pode assumir o papel de cliente – *user agent client* (UAC), o *endpoint* origem – ao emitir requisições para outro UA que, neste caso, assume o papel de servidor – *user agent server* (UAS), o *endpoint* destino – e responde às requisições do cliente.

Na prática, um *user agent* é um artefato físico, tangível de alguma forma. Pode ser representado por *hardware* específico (p. ex., telefones IP) ou não, como Assistentes Pessoais Digitais – do inglês, *Personal Digital Assistant* (PDA) – e telefones celulares (p. ex., *smartphones*). O *user agent* pode ser representado também por *software* aplicativo (programas), na forma de *softphones*. É importante deixar claro que o *user agent*, quando representado por algum programa, é o próprio programa e não o *hardware* no qual o programa está instalado. Assim, por exemplo, se houver um *softphone* instalado em um PDA, o *user agent* será o programa do *softphone* e não o PDA. Essa sutil diferenciação é muito importante, principalmente quando se trata das questões de segurança fim-a-fim em VoIP.

Servidores SIP: podem ser classificados em servidores *proxy* (encaminham mensagens SIP para outros servidores SIP), servidores de redirecionamento (informam ao UAC endereços para que estes possam estabelecer diretamente uma conexão), servidores de registro (ou *registrar servers*, notificam a localização de cada servidor SIP através de DNS e registram ou atualizam a localização de usuários), servidores de localização (descobrem a localização de servidores SIP ou usuários, possuindo um banco de dados local de informações)⁴ e *gateways* (convertem pacotes da Internet em dados de circuitos comutados para trafegarem na rede de telefonia pública – e vice-versa).

Mensagens: a Figura 1 mostra esquematicamente uma mensagem SIP em relação a outros protocolos (supondo UDP para transporte em uma rede Ethernet).

⁴Em algumas configurações, o servidor de registro e o de localização correspondem a um equipa-

Figura 1: Mensagem SIP transportada por UDP em uma rede IP sobre Ethernet – adaptado de (TANENBAUM, 2003)

Uma mensagem SIP pode ser uma requisição⁵ ou uma resposta. As requisições são consideradas “verbos” no protocolo, pois elas solicitam que uma ação específica seja executada por outro *user agent*. Já uma resposta é uma mensagem gerada por um *user agent* na função de servidor para responder uma requisição feita por um *user agent* na função de cliente.

As requisições SIP mais comuns de serem utilizadas são: REGISTER (para registrar um *user agent* em um domínio SIP), INVITE (para configurar uma sessão), ACK (para confirmar o estabelecimento de uma sessão), BYE (para encerrar uma sessão), CANCEL (para cancelar uma sessão pendente ainda não estabelecida), UPDATE (para atualizar informações de uma sessão), OPTIONS (para consultar as opções disponíveis e a capacidade de um *user agent*), e MESSAGE (para enviar uma mensagem instantânea através de uma sessão estabelecida).

Já as respostas SIP, por serem numerosas, são divididas em classes: 1xx (*Provisional* ou *Informational* – requisição em progresso, mas ainda não completada), 2xx (*Success* – requisição completada com sucesso), 3xx (*Redirection* – a requisição deve

mento físico único.

⁵Em algumas literaturas é utilizado o termo “método”.

ser tentada em outro local), 4xx (*Client Error* – a requisição não foi completada por causa de um erro na mesma, pode ser tentada novamente quando a requisição estiver correta), 5xx (*Server Error* – a requisição não foi completada por causa de um erro no receptor, pode ser tentada novamente em outro local), e 6xx (*Global Failure* – a requisição falhou e não deve ser tentada novamente).

Toda mensagem SIP possui uma linha inicial, um ou mais campos de cabeçalho e um corpo, conforme mostra o exemplo da Figura 2. Note na figura que há um duplo CRLF (*carriage return – line feed*) separando os campos de cabeçalho do corpo na mensagem SIP. Por definição, esse duplo CRLF pertence ao cabeçalho SIP. O exemplo mostrado na figura é uma típica mensagem SIP (no caso, de requisição “INVITE”), contendo os principais e mais comuns campos de cabeçalho e o corpo. Por sua vez, o corpo é composto de uma mensagem SDP, que será explicada mais adiante. Note que a mensagem SDP está servindo para negociar não somente as características da sessão de mídia em si, mas também a chave secreta que proverá segurança à sessão, através de um esquema de gerenciamento de chaves denominado “eecmo”.

Cabeçalho SIP	Linha Inicial	INVITE sip:bob@ocean.com SIP/2.0
	Campos de Cabeçalho	Call-ID: 59031.85799575422 CSeq: 1 INVITE From: "Alice" <sip:alice@ocean.com>;tag=56438 To: <sip:bob@ocean.com> Via: SIP/2.0/UDP 192.168.1.111:5060;branch=z9hG4bK35560d541a3 Max-Forwards: 70 Route: <sip:192.168.1.111:5061;lr> Contact: <sip:192.168.1.111:5060> Content-Type: application/sdp Content-Length: 342
	Duplo CRLF	
Corpo SIP	Mensagem SDP	v=0 o=- 3465394895 3465394895 IN IP4 192.168.1.111 s=- a=key-mgmt:eecmo MzMwMjU2MQId4qe+8s1LjT84hbEQ+s37cAK/xEOG/zHB8EQqKj/iTQKlEyVr9z 5zTONAhzeqMnv8tx19FmmgzAnJHUM9Zp82MNm3MgvIAZVGy3qVW1aH54uZUkXa enbrCDUiDIpZ+pfUAK/YdKRYuPIj/9rmBe7jk0ryfdARS655+5d3TTab/Ek5 c=IN IP4 192.168.1.111 t=0 0 m=audio 40000 RTP/AVP 3 m=video 50000 RTP/AVP 26

Figura 2: Exemplo de mensagem SIP típica

A linha inicial identifica o tipo e a finalidade da mensagem. Se a mensagem for uma requisição, a linha é chamada de “linha de requisição”. Se for uma resposta, a linha é chamada de “linha de estado” (*status line*).

Os campos de cabeçalho proveem informações sobre a mensagem em si e sobre o corpo contido na mensagem. Uma característica comum e importante dos campos de cabeçalho é que eles são copiados literalmente na resposta que for dada a uma requisição INVITE recebida, não importa qual seja essa resposta.

O corpo corresponde à carga útil da mensagem SIP e pode conter informações da sessão de mídia, da qualidade de serviço na transmissão, e de segurança. Por padrão, essa carga útil é uma mensagem SDP.

Apesar do nome, o SDP – especificado formalmente na RFC 4566 (HANDLEY; JACOBSON; PERKINS, 2006) – não define um protocolo propriamente dito mas, ao contrário, uma linguagem para representação de parâmetros que caracterizam uma sessão de multimídia. Assim como SIP, o SDP é baseado em texto.

Uma mensagem SDP é composta por um conjunto de linhas de texto onde cada uma possui um parâmetro (representado por um único caracter) e um valor associado a este parâmetro. O tipo e o formato desse valor dependem do parâmetro associado. Esse conjunto parâmetro - valor permite descrever sessões multimídia, indicando diversas características da sessão, como informações sobre conexões, a banda de transmissão, aspectos de tempo, além de informações sobre as mídias propriamente ditas envolvidas na sessão (áudio, vídeo etc.).

Além dos parâmetros considerados padrão, o SDP tem a versatilidade de poder ser estendido através da criação de atributos. Já existem alguns atributos aceitos como oficiais, de natureza genérica, definidos em algumas RFCs complementares à RFC 4566. Além deles, é possível criar atributos proprietários, específicos de aplicações e fabricantes, para finalidades diversas, como, por exemplo, viabilizar uma negociação

segura de chaves secretas entre partes comunicantes.

O SDP trabalha no chamado esquema “oferta-resposta” (*offer-answer*) no qual a parte chamadora faz uma “oferta” das informações de mídia que ela suporta (só áudio, só vídeo, ambos, Codecs etc.) e a parte chamada “responde à oferta”, oferecendo uma “contra-proposta” contendo o que a parte chamada realmente suporta. Neste caso, a parte chamadora acata automaticamente a “contra-proposta”.

2.1.4 Endereçamento

Na arquitetura SIP, os usuários são identificados através de um tipo específico de *Uniform Resource Indicator* (URI), o SIP URI. Por ser um tipo de URI, o SIP URI está aderente às regras gerais da RFC 3986 (BERNERS-LEE; FIELDING; MASINTER, 2005). Em geral, SIP URIs identificam os recursos em uma comunicação, pois contêm informação suficiente para iniciar e manter uma sessão de comunicação entre esses recursos.

Um SIP URI usa um esquema “sip” e é composto de duas partes separadas pelo símbolo “@”: uma parte (opcional) que identifica um recurso em particular no *host* que está sendo endereçado; e outra parte que identifica o *host* em si, isto é, a fonte que está provendo o recurso. Adicionalmente, SIP URI pode conter um certo número de parâmetros que afetam a requisição.

SIP URIs podem se referir a usuários SIP (mais especificamente, à identidade pública de um usuário em um *host*), a um grupo de usuários, a servidores SIP ou, ainda, a serviços tais como e-mails e vídeo-conferências.

Exemplos de SIP URIs: “sip:alice@ocean.com”, “sip:bob@192.168.1.11”, “sip:proxy1.ocean.com”, “sip:193.53.24.3”, “sip:sales.dept@bigmart.com”, “sip:group.videoconf@ietf.com”.

2.1.5 Operação do protocolo

Essencialmente existem os seguintes momentos no ciclo de uso do SIP: registro, autenticação e sinalização para estabelecimento de uma sessão de mídia.

O registro é representado na Figura 3 e é feito através de uma requisição SIP denominada “REGISTER”. Tal requisição transporta o endereço SIP do *user agent* e, a partir deste endereço, o servidor de registros já resolve o endereço IP, armazenando tanto o endereço SIP URI quanto o endereço IP em um banco de dados, que pode ser do próprio servidor de registros ou do servidor de localização. Quando o servidor *proxy* necessitar atender a um “INVITE”, ele consulta o banco de dados do servidor de registros (ou de localização) de modo a obter o endereço IP do *user agent*.

Figura 3: Processo de registro de um *user agent* – adaptado de (SINNREICH; JOHNSTON, 2006, p. 124)

Uma vez devidamente registrado, quando o usuário necessita estabelecer uma sessão de mídia (áudio, vídeo etc.), normalmente ele se autentica junto a um servidor

SIP, que pode ser o próprio servidor de registro original (DWIVEDI, 2009) ou um servidor *proxy*, conforme mostrado na Figura 4.

Figura 4: Processo de autenticação junto a um servidor *proxy* SIP – adaptado de (SINNREICH; JOHNSTON, 2006, p. 131)

No que diz respeito à sinalização em si para estabelecimento de sessões de mídia, SIP suporta três modos de sinalização. O mais simples deles é o modo básico de sinalização, mostrado na Figura 5.

Na prática, este modo de sinalização não é muito comum. Apesar da aparente simplicidade do cenário expresso no modelo, há uma questão primordial a ser resolvida: o endereçamento. O *user agent* emissor precisaria saber exatamente o endereço IP do *user agent* receptor. Entretanto, um endereço IP não pode ser usado de forma tão simples como se fosse um número de telefone. Uma razão para isso é que, devido às limitações de endereçamento do IPv4, endereços IP na Internet geralmente são atribuídos dinamicamente através do *Dynamic Host Configuration Protocol* (DHCP). Isto

Figura 5: Modo básico de sinalização SIP: comunicação direta entre dois *user agents* – adaptado de (SINNREICH; JOHNSTON, 2006, p. 112)

acontece em toda chamada VoIP na qual, a cada sessão, o *user agent* pode receber um endereço IP diferente – mesmo em conexões de banda larga.

Por isso, são mais comuns modos de sinalização nos quais a consulta DNS (DNS *lookup*), a resolução de endereços, os redirecionamentos, a execução de pedidos de conexão, e outras tarefas extraprotocolo SIP, sejam feitos por dispositivos dedicados, tais como um servidor *proxy*. É o caso dos modos de sinalização mostrados nas Figuras 6 e 7 (este último é conhecido como “Trapezoide SIP”).

Outro ponto importante a ser observado é que, independente do número de servidores *proxy* envolvidos no processo de sinalização, eles apenas retransmitem as mensagens SIP recebidas, sem modificá-las. O máximo que um servidor *proxy* faz é ler e analisar determinados campos de cabeçalho, de modo a encaminhar corretamente a mensagem SIP, seja para outro servidor ou para um *user agent*. É importante ressaltar que *proxies* SIP não atuam no corpo de uma mensagem SIP e, conseqüentemente,

Figura 6: Modo de sinalização SIP com servidor *proxy* – adaptado de (SINNREICH; JOHNSTON, 2006, p. 116)

não atuam em mensagens SDP.

Por fim, há ainda o modo de sinalização com redirecionamento, mostrado na Figura 8.

2.2 Segurança no contexto do SIP

2.2.1 SIP como facilitador da segurança em VoIP

Do ponto de vista do usuário final de VoIP, o que importa mesmo é se a comunicação dele com a outra parte (independente do tipo de mídia – áudio, vídeo, texto etc.) tem garantias quanto à autenticidade, privacidade e integridade dos dados trafegados. Pois o *user agent* atua na sinalização, mas o usuário em si só passa a ter algum papel na comunicação quando a sessão de mídia já estiver estabelecida.

Figura 7: Modo de sinalização SIP com mais de um servidor *proxy* envolvido – Trapezoide SIP

O protocolo utilizado na sessão de mídia é o RTP (SCHULZRINNE et al., 2003). Para prover segurança a ele, foi desenvolvido o SRTP (BAUGHER et al., 2004) que consegue garantir a privacidade através do uso do AES e a autenticidade e integridade através do uso combinado de *keyed-Hash Message Authentication Code* (HMAC) com *Secure Hash Algorithm One* (SHA-1). Ainda que o SRTP tenha à disposição dele mecanismos considerados *hors concours* em termos de segurança, os mesmos dependem de elementos básicos, porém essenciais: chaves secretas. O SRTP parte do princípio de que tais chaves, associadas ou não a outros parâmetros de segurança, são fornecidas por algum esquema de gerenciamento de chaves anterior ao RTP, independente de qual seja esse esquema. Como a sinalização ocorre obrigatoriamente antes do estabelecimento da sessão de mídia, uma abordagem conveniente é aproveitar a sinalização para já providenciar as chaves secretas e, eventualmente, outros parâmetros de segurança.

Figura 8: Modo de sinalização SIP com redirecionamento – adaptado de (DOUSKALIS, 2000, p. 78)

Assim, ao longo dos anos têm surgido algumas propostas no intuito de prover segurança e robustez ao SIP, de modo que seja possível aproveitar o processo de sinalização para realizar o gerenciamento/negociação de chaves secretas e, assim, o protocolo ser usado como um facilitador na segurança para o usuário final de VoIP.

De um modo geral, as propostas têm feito uso do corpo da mensagem SIP (que é onde está a carga útil) para realizar, de alguma maneira, o acordo de chaves. Mais especificamente, usando mensagens SDP transportadas pelo SIP.

2.2.2 Mecanismos para prover segurança ao SIP

A primeira proposta, por assim dizer, foi utilizar o parâmetro “k” da mensagem SDP. Este parâmetro foi idealizado tanto para transmitir chaves de encriptação (em claro ou codificadas em Base64) quanto para obtê-las a partir de algum endereço na Internet, conforme consta na RFC original do SDP, a RFC 2327 (HANDLEY; JACOBSON, 1998).

Esta RFC tornou-se obsoleta com a publicação da RFC 4566 (que é a mais atual para o SDP), a qual assume uma posição totalmente contrária à sua antecessora, sendo categórica ao NÃO recomendar o uso do parâmetro “k=”: “Um mecanismo simples para troca de chaves é provido pelo campo chave (“k=”), embora ele seja suportado apenas para manter compatibilidade com as implementações legadas, pois o uso dele NÃO É RECOMENDADO.” (HANDLEY; JACOBSON; PERKINS, 2006, p. 19, tradução nossa).

De fato, é totalmente insensato usar um protocolo textual para transmitir chaves secretas em claro, pois ele não oferece o mínimo trabalho para um atacante capturar, por escuta simples, os dados trafegados. A própria RFC 2327 já previa a necessidade de uma camada de segurança adicional – e externa – ao SDP para se utilizar o parâmetro “k”. E isto aplica-se para qualquer proposta que faça uso do SIP/SDP para transmitir dados em claro.

Pensando nisso, tanto a RFC original do SIP – RFC 2543 – quanto a mais atual – RFC 3261 – foram publicadas contemplando alguns mecanismos de segurança, os quais não eram exclusivos do SIP, mas poderiam ser utilizados em conjunto com o protocolo para, na sinalização, tentarem assegurar a transmissão de chaves secretas:

IPSec: uma suíte de protocolos da camada de rede desenvolvidos pelo IETF para suportar a troca segura de pacotes na camada IP (modo tunelamento) e também para proteger protocolos da camada de transporte (UDP, TCP etc.). Para isso, IPSec ou necessita de uma infraestrutura de chaves públicas ou de chaves previamente negociadas, sendo que essa pré-negociação é normalmente feita pelo *Internet Key Exchange* (IKE), o qual é baseado no acordo de chaves Diffie-Hellman (PEREA, 2008).

TLS: outro mecanismo de segurança aplicável ao SIP para proteger o canal de comunicação é o *Transport Layer Security* (TLS). Assim como o IPSec, TLS também provê confidencialidade, autenticidade e integridade, com a vantagem de que pode fazer isso em camadas acima da de transporte. Pode prover autenticação mútua: ambos os *endpoints* sabem, com certeza, com quem estão se comunicando. Provê

suporte para um grande número de suítes de encriptação, admitindo variações que utilizam RSA, Diffie-Hellman, 3DES, MD5, SHA-1 etc.

SIPS URI: é um tipo especial de SIP URI e representa um mecanismo decorrente do TLS para identificar um recurso (outro *user agent*, por exemplo) que deva ser alcançado seguramente via TLS. Isso significa que todo o caminho, desde o *user agent* de origem até o *user agent* de destino, passando por servidores SIP (principalmente *proxies*), devem estar protegidos por TLS (proteção *hop-by-hop*). Exemplo de SIPS URI: “sips:bob@sea.com” (note que há o acréscimo de um “s” ao esquema “sip” do SIP URI).⁶

S/MIME: o *Secure/Multi-purpose Internet Mail Extensions* (S/MIME) é um aprimoramento, em termos de segurança, para o formato padrão de e-mail Internet MIME. O S/MIME permite tanto assinar quanto encriptar mensagens, utilizando, para isto, certificados de chaves públicas em conformidade com a versão 3 do padrão X.509. Os certificados devem ser configurados localmente nos *user agents* para serem utilizados na encriptação e na verificação de assinaturas digitais.

A Figura 9 mostra um exemplo de como mensagens SIP podem ser protegidas com o uso de S/MIME. Na figura, os asteriscos não fazem parte da mensagem SIP. Eles servem apenas para mostrar quais partes da mensagem SIP foram protegidas: um campo de cabeçalho (“Content-Type”) e todo o conteúdo da mensagem SDP que compõe o corpo. É importante esclarecer que nem todos os campos de cabeçalho podem ser protegidos, pois alguns são necessários para fins de roteamento em *proxies* SIP.

Uma assinatura S/MIME só é um mecanismo eficiente se o *user agent* receptor

⁶É comum as literaturas tratarem este esquema como um mecanismo de segurança. No entanto, entende-se que isto não é verdade, pois SIPS URI simplesmente serve para forçar o uso de TLS em todo o caminho da sinalização. Ou seja, o mecanismo de segurança em si é o TLS. Inclusive, o único ponto que poderia caracterizar o uso de SIPS URI como mecanismo de defesa é o parâmetro “senha” que existe no esquema “sips:”, mas cujo uso é desaconselhado pela própria RFC 3261 pelo fato do parâmetro ser transmitido em claro.

```

INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
             name=smime.p7m
Content-Disposition: attachment; filename=smime.p7m
                  handling=required

*****
* Content-Type: application/sdp                                *
*                                                             *
* v=0                                                         *
* o=alice 53655765 2353687637 IN IP4 pc33.atlanta.com      *
* s=-                                                         *
* t=0 0                                                       *
* c=IN IP4 pc33.atlanta.com                                   *
* m=audio 3456 RTP/AVP 0 1 3 99                             *
* a=rtpmap:0 PCMU/8000                                       *
*****

```

Figura 9: Utilização do S/MIME em mensagens SIP – extraído de (ROSEMBERG et al., 2002, p. 205)

da mensagem requerer a presença do corpo S/MIME. Caso contrário, um atacante poderia simplesmente modificar a requisição SIP e substituir o corpo assinado com S/MIME por um outro corpo de conteúdo qualquer, que o receptor não perceberia.

Tanto para encriptação quanto para validação de assinatura digital, S/MIME requer o uso de chaves públicas e certificados digitais. Assim, caso não seja possível manter um “chaveiro” de chaves públicas e correspondentes certificados digitais, o *user agent* deve estar apto para obter, dinâmica e externamente, a chave pública / certificado digital (ROSEMBERG et al., 2002; SINNREICH; JOHNSTON, 2006; TANENBAUM, 2003).

PGP: na primeira versão da especificação do SIP que consta na RFC 2543 (HANDLEY et al., 1999) foi proposto o uso do *Pretty Good Privacy* (PGP), um pacote completo para segurança de mensagens de correio eletrônico que fornece confidencialidade, autenticidade, integridade e compactação. Idealizado e desenvolvido por Phillip R. Zimmermann Jr., o PGP utiliza o *International Data Encryption Algorithm* (IDEA) de 128 bits para encriptação simétrica. O gerenciamento de chaves utiliza o RSA e a integridade

fica por conta do algoritmo MD5 (STALLINGS, 2006). A RFC 2543 possuía campos de cabeçalho específicos para viabilizar o uso do PGP, os quais foram descontinuados na RFC 3261, em razão desta propor a utilização do S/MIME.

Não foi encontrada uma explicação ou razão clara e definitiva do porquê da substituição do PGP pelo S/MIME. Quando questionado diretamente, o próprio Phil Zimmermann respondeu que não sabia a razão dessa mudança e direcionou a pergunta para Alan Johnston, um dos autores de ambas RFCs do SIP. Johnston respondeu que quando chegou a hora de revisar o SIP, foi percebido que o uso de PGP para autenticação e encriptação estava subespecificado e que também não havia implementações do SIP com suporte para PGP. Assim, o PGP deu lugar ao S/MIME. Johnston ainda opinou que não há razão por que alguém não poderia implementar os cabeçalhos da RFC 2543 se houver um sistema PGP sendo executado. Não é como no caso do HTTP Basic Authentication (1.0), o qual foi removido porque ele tinha problemas de segurança maiores (informação pessoal)⁷.

Além dos mecanismos existentes nas RFCs do SIP, outros foram propostos, dos quais destacam-se aqueles que acabaram tornando-se padrões IETF e foram idealizados objetivando assegurar o processo de negociação de chaves secretas:

MIKEY: o *Multimedia Internet KEYing* (MIKEY) prevê um esquema para realizar a negociação segura de chaves secretas de sessão e está descrito na RFC-3830 (ARKKO et al., 2004). Ele pode operar nos seguintes modos: *pre-shared key* (MIKEY-PSK), *public key* (MIKEY-PK, mas que também ficou conhecido como MIKEY-RSA) e *signed* Diffie-Hellman (MIKEY-DHSIGN). O primeiro modo é puramente baseado no estabelecimento de chaves secretas comuns realizado de forma “*offline*” (p. ex., uma cerimônia de chaves como aquelas comumente realizadas por *Security Offices* de grandes corporações). O segundo e o terceiro modos são baseados em criptografia com chaves públicas, sendo que ambos necessitam de certificados digitais por conta das assinatu-

⁷ Johnston, A. B. **PGP in the original SIP RFC – why changed ?** Mensagem recebida por adeusajute@larc.usp.br em 11 jun.2008.

ras digitais que devem constar nas mensagens. Por isso, o MIKEY foi projetado para poder transportar certificados digitais na própria mensagem.

Existem também outros dois modos, instituídos por duas RFCs complementares. A RFC 4650 (EUCHNER, 2006) descreve o modo MIKEY-DHMAC, o qual combina o uso do esquema Diffie-Hellman com um esquema HMAC. Este modo dispensa o uso de certificados digitais, mas, por outro lado, necessita de um acordo prévio de chaves entre as partes comunicantes. A RFC 4738 (IGNJATIC et al., 2006) descreve o MIKEY-RSA-R que, em linhas gerais, é o modo *public key* mas de forma reversa, isto é, o receptor (a parte chamada) é que define as condições de segurança, incluindo a chave secreta que originará a chave da sessão SRTP. Segundo os autores dessa RFC, este modo foi idealizado para cenários de comunicação onde a parte chamadora pode não conhecer a identificação correta e/ou o certificado da parte chamada.⁸

O esquema de gerenciamento/negociação de chaves previsto no MIKEY pode ser atrelado a mensagens SDP seguindo-se as orientações contidas na RFC 4567 (ARKKO et al., 2006). Esta RFC instituiu o uso de extensões específicas no SDP, visando o gerenciamento de chaves criptográficas⁹. Uma dessas extensões é o *key management attribute* (ou, abreviadamente, “key-mgmt”), o qual permite que mensagens MIKEY sejam incorporadas em mensagens SDP (e, daí, incluídas em mensagens SIP). Para ilustrar, a seguinte mensagem SDP possui uma mensagem MIKEY incorporada apropriadamente:

⁸Ainda foi proposto um outro modo que ficou conhecido como DHMAC-SAS (BARRETO; FALEIROS, 2007). Ele faz uso de *Short Authentication String* (SAS) e, essencialmente, é o modo DHMAC combinado com algumas proteções adicionais do *Media Path Key Agreement for secure RTP* (ZRTP), também idealizado por Phillip Zimmermann. Segundo os autores, o objetivo maior do DHMAC-SAS é prover uma escalabilidade segura ao DHMAC, de modo que o MIKEY possa ser usado em um ambiente com um grande número de usuários. Apesar da relevância da proposta, o DHMAC-SAS não se tornou uma RFC, ficando apenas como um *draft* IETF.

⁹Alguns autores consideram a RFC 4567 como um mecanismo de segurança baseado no SDP. O fato é que a RFC 4567 abre a possibilidade para que se proponha, de forma padronizada, esquemas para gerenciamento/negociação de chaves secretas. Mas o único esquema citado pela RFC (talvez até por falta de outras propostas) é o próprio MIKEY.

```

v=0
o=alice 2891092738 2891092738 IN IP4 w-land.example.com
s=Secret discussion
e=alice@w-land.example.com
t=0 0
c=IN IP4 w-land.example.com
a=key-mgmt:mikey AQAfGm0XflABAAAAAAAAAAAAAsAyONQ6gAAAAAGEEoo2pee4hp2
UaDX8ZE22YwKAAAPZG9uYWxkQGR1Y2suY29tAQAAAAAAQAK0JKpgaVkDaawi9whVBtBt
0KZ14ymNuu62+Nv3ozPLygwK/GbAV9iemnGUIZ19fWQUOSrzKTA9zV
m=audio 49000 RTP/SAVP 98
a=rtpmap:98 AMR/8000
m=video 52230 RTP/SAVP 31
a=rtpmap:31 H261/90000

```

Atributo “a=crypto”: a RFC 4568 (ANDREASEN; BAUGHER; WING, 2006), aproveitando-se da extensibilidade do SDP (vide subseção 2.1.3), introduziu atributos específicos do tipo “a=crypto”. O objetivo foi fazer uso destes atributos para transportar chaves criptográficas e outros parâmetros para compor o contexto de segurança de sessões SRTP. Porém, as chaves são enviadas em claro, necessitando de um mecanismo de segurança adicional, como, por exemplo, o S/MIME. Isto significa que, se por um lado não foi recomendado o uso do parâmetro “k” pela RFC original do SDP (pois a chave transitaria em claro), por outro, uma nova RFC instituiu um novo atributo que, em termos de segurança, é similar ao parâmetro “k”. Além disso, abre espaço para ataques *Bidding-Down*¹⁰.

Fingerprint: este mecanismo foi instituído pela RFC 4572, a qual especifica como estabelecer sessões seguras para transporte de mídia, orientadas à conexão sobre

¹⁰A RFC 4568 propõe o uso do esquema “oferta-resposta” do SDP para determinar o nível de segurança entre os *user agents*. Neste caso, uma das partes pode oferecer duas possibilidades de uso do AES – 128 e 256 *bits* – para a outra parte. Esta, por sua vez, vai responder sempre com o nível de segurança maior suportado (256 *bits*, por exemplo). O ataque *bidding-down* se dá quando o atacante intercepta a requisição INVITE e altera a “oferta” (já que a mensagem SDP está em claro), de modo que seja oferecida a opção de nível de segurança mais “baixa” possível. Daí, a outra parte, sem ter opções para escolher, acaba respondendo também com o nível de segurança mais “baixo”.

TLS, e tendo o SDP como facilitador. Para tanto, é definido um novo atributo SDP – “fingerprint” – que identifica o certificado digital que será utilizado para estabelecer a sessão TLS. Esse atributo “fingerprint” corresponde a um valor de resumo criptográfico (*hash*) do certificado. Durante a sinalização, as partes trocam entre si os respectivos atributos “fingerprint”, aproveitando o esquema “oferta-resposta” do SDP. Depois de estabelecida a sessão, as partes podem trocar os certificados entre si e verificar a validade dos mesmos através dos atributos “fingerprints” trocados anteriormente (LENNOX, 2006). Exemplo de uma mensagem SDP com o atributo “fingerprint”:

```
v=0
o=john 8427002738 2891092738 IN IP4 w-land.example.com
s=-
t=0 0
c=IN IP4 w-land.example.com
m=image 51221 TCP/TLS t38
a=setup:passive
a=connection:new
a=fingerprint:SHA-1
\4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
```

2.2.3 Análise dos mecanismos de segurança

Nesta subseção é feita uma análise de cada mecanismo de segurança citado na subseção anterior, objetivando apontar os pontos fortes e de atenção de cada mecanismo. Antes, porém, é necessário equalizar alguns conceitos e critérios adotados na análise:

- Não foram considerados na análise o parâmetro “k” e o atributo “a=crypto” do SDP, pois ambos, por si só, claramente, não são seguros, conforme explicado na subseção anterior.

- Como não há um consenso entre autores ou mesmo entre profissionais da segurança da informação sobre o que delimita conceitualmente segurança fim-a-fim e ponto-a-ponto, adotou-se o seguinte critério na análise:
 - Segurança ponto-a-ponto: aquela que protege apenas o trânsito dos pacotes de dados pelo meio de transmissão que liga duas partes comunicantes. Assim, a segurança ponto-a-ponto começa no momento em que os pacotes de dados adentram no meio de transmissão, continua durante a transmissão por este, e termina no momento em que os pacotes de dados chegam ao ponto final do meio de transmissão, saindo deste. Normalmente, é implementada por *hardware* ou, no máximo, por algum *software* que representa um sistema operacional ou um serviço especializado (nativo ou não) provido pelo mesmo.
 - Segurança fim-a-fim: aquela que começa (na origem) imediatamente depois de gerada a informação que deve ser protegida, continua durante a transmissão pelo meio de comunicação entre duas partes e só termina imediatamente antes do uso efetivo da informação, momento no qual ela deve ficar exposta para ser utilizada no destino. Normalmente, é implementada por *software* (camada de aplicação).
- Considerou-se como ataque *man-in-the-middle* o tipo de ataque que ocorre em uma comunicação entre duas partes (um emissor e um receptor), e é executado por alguém tomando o cuidado de não se expor. Este tipo de ataque pode ser utilizado para alterar o fluxo encriptado de várias maneiras: excluindo, retardando, substituindo ou inserindo texto encriptado adulterado. Ainda neste tipo de ataque, se o atacante quiser monitorar uma comunicação por um longo período de tempo, ele deve tentar mantê-la transparente tanto quanto for possível, uma vez que qualquer marca que ele deixar naquilo que for transmitido provavelmente levantará suspeitas (RIVEST; SHAMIR, 1984).

Análise:

- **IPSec**

Pontos fortes:

- Provê autenticidade, confidencialidade e integridade.
- Aplicável tanto ao IPv4 quanto ao IPv6, podendo, assim, ser usado em sistemas legados da Internet e em novas implementações.

Pontos de atenção:

- Não provê segurança fim-a-fim, apenas ponto-a-ponto.
- Dependente de uma infraestrutura de chaves públicas e certificados digitais ou de um acordo de chaves baseado no esquema Diffie-Hellman, o qual é vulnerável a ataques do tipo *man-in-the-middle*.

- **TLS / SIPS URI**

Pontos fortes:

- Provê autenticidade, confidencialidade e integridade.
- Permite estabelecer autenticação mútua entre as partes (cliente e servidor) através do uso de certificados digitais.
- Provê suporte para um grande número de suítes de encriptação, admitindo variações que utilizam RSA, Diffie-Hellman, 3DES, MD5, SHA-1 etc.

Pontos de atenção:

- Não provê segurança fim-a-fim, apenas ponto-a-ponto.
- A autenticação mútua através do uso de certificados digitais implica na necessidade de uma infraestrutura de chaves públicas. Além disso, na maioria dos casos, por questões de conveniência ou de restrições mesmo, não há autenticação mútua efetiva, apenas o servidor é autenticado via TLS (PEREA, 2008).

- Para o esquema SIPS URI ser realmente eficiente, tem de se garantir o uso de TLS em todo o caminho, desde o *user agent* chamador até o *user agent* chamado, independente de quantos e quais tipos de servidores SIP existam no caminho. E obter essa garantia nem sempre é fácil (ou viável) no caso de sinalização na Internet, dada a dimensão e abrangência dessa rede.
- Requer um protocolo de transporte orientado à conexão, tal como o TCP – e as implementações mais eficientes do SIP utilizam o UDP para transporte.
- Assim como seu antecessor – o SSL – o TLS é vulnerável a ataques do tipo *man-in-the-middle*.

- **S/MIME**

Pontos fortes:

- Quando corretamente implementado, provê segurança fim-a-fim.
- Provê confidencialidade e integridade.
- É o único mecanismo capaz de proteger todo o corpo de uma mensagem SIP e aqueles campos de cabeçalho que são permitidos. Ou seja, com ele é possível proteger informações que poderiam ser consideradas sensíveis e que, obrigatoriamente, devem ser enviadas através de campos de cabeçalho.

Pontos de atenção:

- Necessita de uma infraestrutura de chaves públicas previamente estabelecida, pois requer a instalação prévia dessas chaves públicas e respectivos certificados digitais nos *user agents*, incluindo os certificados-raiz de Autoridades Certificadoras, da mesma forma como acontece nos programas para navegação na Internet (*browsers*).
- A autenticidade só é garantida se o certificado digital utilizado no S/MIME

é de uma parte terceira (p. ex., uma Autoridade Certificadora reconhecida como idônea). Caso contrário, torna-se vulnerável a ataques do tipo *man-in-the-middle*: se certificados autoassinados (ou certificados que não podem ser verificados por qualquer uma das partes da comunicação) forem usados, um atacante pode, efetivamente, atacar e modificar corpos S/MIME, neutralizando assim a proteção provida pelo mecanismo à mensagem SIP (ROSEMBERG et al., 2002, p. 201 - 204).

- S/MIME pode resultar em mensagens muito longas, especialmente quando é utilizado um mecanismo de tunelamento SIP. Neste caso, quando o tunelamento S/MIME é empregado, seria recomendável o uso de TCP como protocolo de transporte – mas aí há novamente a questão de que implementações SIP são mais eficientes com o UDP.

• PGP

Pontos fortes:

- Provê segurança fim-a-fim no que diz respeito à confidencialidade e integridade.
- Simples e eficiente, pois se beneficia diretamente dos princípios da criptografia assimétrica.

Pontos de atenção:

- Depende de um pré-compartilhamento das chaves públicas de modo a garantir a autenticidade das partes comunicantes, e isto representa um problema para a escalabilidade no contexto específico do SIP. Ou, alternativamente, depende de certificados digitais e infraestrutura de chaves públicas para prover a mesma autenticidade. Neste caso, cabe a mesma ressalva feita para o S/MIME quanto aos certificados digitais a serem utilizados.

- **MIKEY**

Pontos fortes:

- Provê segurança fim-a-fim.
- Todos os modos descritos nas RFCs são capazes de prover autenticidade, confidencialidade e integridade.

Pontos de atenção:

- Os modos MIKEY-PSK e MIKEY-DHMAC implicam na necessidade de chaves secretas previamente negociadas entre as partes comunicantes. Os procedimentos externos de configuração dessas chaves são, muitas vezes, manuais, dispendiosos e demorados, prejudicando a escalabilidade do sistema.
- Os modos MIKEY-RSA, MIKEY-DHSIGN e MIKEY-RSA-R implicam na necessidade de certificados digitais e, daí, de uma infraestrutura de chaves públicas. Novamente, deve-se atentar para os certificados digitais utilizados, de modo a não sofrer ataques do tipo *man-in-the-middle*. Além disso, quando combinado com SIP, os certificados transportados pelo MIKEY tendem a deixar as mensagens SIP grandes o suficiente para serem divididas em vários pacotes de dados a transmitir. E isto pode ser um problema quando se utiliza o UDP, já que o mesmo não oferece garantia de entrega dos pacotes.

- ***Fingerprint***

Pontos fortes:

- Viabiliza o uso de certificados digitais autoassinados pelos *endpoints*, evitando-se assim o uso de uma infraestrutura de chaves públicas.

Pontos de atenção:

- É suscetível a ataques *man-in-the-middle* caso não se garanta a integridade da mensagem SDP no corpo SIP.
- Apesar de o mecanismo ter sido idealizado para poder viabilizar o uso de certificados autoassinados em sessões TLS, não há obrigação de uma parte ou da outra aceitar a comunicação via TLS. Ou seja, as partes comunicantes podem até trocar os respectivos atributos “fingerprint”, mas não há nada que as force, depois, estabelecerem uma sessão TLS.

A Tabela 1 sumariza a análise feita anteriormente, comparando os mecanismos de segurança em relação a quesitos – de segurança e operacionais – mais significativos. O atributo SDP “fingerprint” acaba destoando dos demais mecanismos por ser uma abordagem totalmente diferente. Na verdade, ele é mais um “suporte” para viabilizar outros mecanismos – como certificados autoassinados – mas que acaba dependendo de segurança adicional para se sustentar.

Pela tabela nota-se que um mecanismo de segurança apropriado para o SIP no contexto VoIP seria aquele que conseguisse prover segurança fim-a-fim com autenticidade, integridade, confidencialidade, e sem pontos de atenção de qualquer natureza.

2.2.4 Discussão acerca dos mecanismos de segurança

Antes que SIP possa ser utilizado como facilitador da segurança para VoIP, de modo que a comunicação entre dois *user agents* (e seus respectivos usuários) esteja assegurada quanto à autenticidade, confidencialidade e integridade, é preciso prover estes mesmos serviços de segurança ao próprio SIP.

Foi mostrado que o SIP possui um conjunto razoável de mecanismos de segurança à disposição do protocolo. Entretanto, o uso deles deve ser avaliado sob a ótica tanto da eficiência em termos de segurança quanto do custo-benefício no contexto em que

eles estão sendo aplicados.

A começar pela segurança ponto-a-ponto: ela é geralmente eficiente e satisfatória sob determinadas condições. Porém, prevendo-se atacantes bem próximos às aplicações (p. ex., *malwares* no caso de *softphones*), é necessário uma segurança fim-a-fim. Caso contrário, poderiam ocorrer ataques no tráfego das informações em claro, entre a saída do “túnel” SSL e o destino final – por exemplo, um *softphone*. Isto é facilmente demonstrável com navegadores de Internet (*browsers*) ao se utilizar produtos para análise de tráfego – tais como o HttpWatch, um *plug-in* que pode ser acoplado em navegadores como o “Microsoft Internet Explorer” e o “Mozilla Firefox” – que servem para visualizar o tráfego HTTP e, inclusive, HTTPS. Assim, é possível capturar dados sigilosos que tenham sido transmitidos e protegidos pelo SSL, mas que não estão mais sob tal proteção em determinado momento.

Uma vez que segurança ponto-a-ponto não é abrangente, opta-se pela segurança fim-a-fim. E o uso de certificados digitais para encriptação e autenticação, tendo por trás uma infraestrutura de chaves públicas, parece ser o melhor caminho a seguir. De fato, infraestrutura de chaves públicas e certificados digitais são, reconhecidamente, eficientes para auxiliar na segurança fim-a-fim, sobretudo nos quesitos autenticidade, confidencialidade e integridade. Tanto que são largamente adotados, subsidiando, inclusive, decisões tomadas na esfera jurídica.

Porém, demandam razoável complexidade devida aos processos extrassistêmicos associados ao ciclo de vida dos certificados (emissão, publicação, revogação e renovação), além da necessidade de se gerenciar certificados-raiz de Autoridades Certificadoras. Esta complexidade se torna aparente quando é considerado o número de usuários atrelados a uma infraestrutura de chaves públicas.

E mesmo que se pense na utilização de certificados autoassinados, mecanismos que dependam deste tipo de certificado são seriamente limitados, uma vez que não há, virtualmente, uma autoridade consolidada que possa prover, atualmente, certificados

para aplicações fim-a-fim (ROSEMBERG et al., 2002).

Assim, em uma primeira análise, usar uma infraestrutura de chaves públicas no contexto de VoIP, objetivando autenticar ambas (e quaisquer) partes comunicantes, as quais podem estar em qualquer parte do mundo, não parece ser muito prático.

Outro agravante no contexto de VoIP é que a obtenção de certificados assinados por Autoridades Certificadoras, para serem utilizados em *endpoints* comunicantes, é difícil na maioria das vezes. Por exemplo, a maioria dos *hosts* tem endereços dinâmicos de IP ou nomes de *hosts* atribuídos via DHCP. Não seria muito fácil e prático obter (e gerenciar) certificados assinados digitalmente que fossem válidos apenas no período de concessão do endereço IP (LENNOX, 2006, p. 4).

Um ponto que deve ficar claro nesta discussão é que o problema em se utilizar uma infraestrutura de chaves pública e certificados digitais não é o fato de se ter uma entidade externa ao sistema que o rege de alguma forma (tal como ocorre com as Autoridades Certificadoras). A grande maioria dos modelos de segurança (senão todos) que oferecem o serviço de autenticidade, preveem o uso de algum tipo de entidade externa. O problema está nos processos extrassistêmicos, de razoável complexidade, manuais em determinados momentos, e que podem gerar outras preocupações de segurança, além de serem difíceis de aplicar no contexto de VoIP, impactando na escalabilidade.

Outra abordagem que, apesar de simples na teoria, afetaria a escalabilidade é a necessidade de se trocar ou configurar previamente chaves secretas, pois a complexidade aumenta a medida que aumenta o número de usuários no sistema de VoIP.

Se por um lado SIP enfrenta problemas no aspecto da praticidade quanto à implementação de determinados mecanismos de segurança, por outro há uma grande preocupação com ataques do tipo *man-in-the-middle*.

Ao se analisar os principais tipos de ataque que SIP pode sofrer – vide (BUTCHER;

LI; GUO, 2007), (REZENDE, 2006), e (SINNREICH; JOHNSTON, 2006) – percebe-se que praticamente todos eles tem como agente causador “alguém tomando o cuidado de não se expor”, isto é, o próprio *man-in-the-middle*.

Uma constatação interessante é que pode ocorrer um ataque *man-in-the-middle* muito simples: a remoção de qualquer indicação de uso de um esquema de gerenciamento/negociação de chaves secretas. Por exemplo, o MIKEY faz uso de um atributo SDP (“a=”) para transportar chaves e parâmetros de segurança. A criação deste atributo SDP sempre ocorre no *user agent* origem. Porém, o *user agent* destino só tomará conhecimento de que é para utilizar o MIKEY quando ele receber o atributo “a=” na mensagem SDP. Mas, e se um ataque *man-in-the-middle* simplesmente retirar este atributo, uma vez que ele, assim como toda a mensagem SIP trafegada (campos de cabeçalho e corpo), são puramente texto e facilmente manipuláveis ?

Ou ainda, um ataque *man-in-the-middle* “inesperado”, também relacionado ao MIKEY, mas especificamente para os modos MIKEY-PSK e MIKEY-DHMAC: caso a mesma chave secreta fosse previamente combinada entre três ou mais participantes do esquema, qualquer um deles poderia se tornar um atacante. Isto porque qualquer um poderia interceptar a comunicação entre outras duas partes quaisquer e, utilizando a chave secreta comum, ou descobrir como a sessão de mídia entre essas partes será protegida (no caso do MIKEY-PSK) ou até determinar tal proteção (no caso do MIKEY-DHMAC, onde o atacante poderia aplicar um ataque *man-in-the-middle* sobre o esquema Diffie-Hellman, já que seria possível falsificar o HMAC).¹¹

• • •

¹¹ Este tipo de ataque não foi considerado na análise realizada na subseção 2.2.3 onde optou-se por considerar um atacante típico, o qual, normalmente, é alguém de fora do grupo das partes comunicantes previstas. Além disso, apesar dos modos MIKEY-PSK e MIKEY-DHMAC partirem do princípio de que duas partes comunicantes devam possuir uma chave secreta comum previamente negociada, isto não significa que todo um grupo deva proceder da mesma forma (p. ex., metade de um grupo pode usar uma chave secreta comum que é totalmente diferente da chave secreta comum da outra metade). Ou seja, dependendo da forma como for negociada previamente a chave secreta num grupo, esse tipo de ataque pode nem ocorrer. Ainda assim, é importante indicar que existe essa possibilidade de ataque *man-in-the-middle*.

Neste capítulo foram apresentados os conceitos fundamentais do SIP, indicando que o protocolo poderia ser utilizado para negociar chaves secretas e outros parâmetros de segurança que se fizerem necessários, de modo a garantir a segurança (autenticidade, confidencialidade e integridade) no momento que realmente interessa ao usuário final de VoIP: durante a sessão de mídia. Antes, porém, é necessário dotar o SIP de algum mecanismo que possa lhe prover também os serviços de autenticidade, confidencialidade e integridade.

Entretanto, os mecanismos de segurança existentes à disposição do SIP apresentados no capítulo ou não são abrangentes ou, quando são, não são práticos e demandam razoável complexidade e esforço na implementação dentro do contexto de VoIP, podendo não compensar o custo-benefício.

Outros agravantes para prover segurança ao protocolo podem estar na própria natureza textual do mesmo e na forma como o protocolo permite instituir esquemas de gerenciamento/negociação de chaves, como o MIKEY. Esse agravante pôde ser constatado em um ataque do tipo *man-in-the-middle* comentado no capítulo, o qual pode impedir qualquer tipo de negociação de chaves, se simplesmente forem retirados da mensagem SIP os atributos específicos relacionados à segurança.

Assim, para que o SIP possa realmente ser um agente facilitador para segurança em VoIP, é necessário um mecanismo de segurança que possa prover autenticidade, confidencialidade e integridade ao protocolo de forma abrangente, que evite ataques *man-in-the-middle* de qualquer natureza, e que não seja dependente de condições e processos externos difíceis de se colocar em prática no contexto de VoIP.

Tabela 1: Comparação dos mecanismos de segurança do SIP que constam nas RFCs 3261, 3830, 4572, 4650 e 4738 em relação a quesitos de segurança e operacionais

	Provê			Provê		Necessita de uma ICP	Necessita de pré-compartilhamento de chaves	Suscetível a ataques do tipo <i>man-in-the-middle</i>
	Segurança fim-a-fim	Provê Autenticidade	Provê Integridade	Provê Confidencialidade	Segurança ponto-a-ponto apenas			
IPSec		⊕	⊕	⊕	⊙			⊗ ¹
TLS / SIPS URI		⊕	⊕	⊕	⊙	⊙		⊗
S/MIME	⊕	⊕ ²	⊕	⊕		⊙		⊗ ³
PGP	⊕	⊕ ⁴	⊕	⊕	ou	⊙	⊙	⊗ ³
MIKEY-PSK	⊕	⊕	⊕	⊕			⊙	
MIKEY-RSA	⊕	⊕ ²	⊕	⊕		⊙		⊗ ³
MIKEY-DHSIGN	⊕	⊕ ²	⊕	⊕		⊙		⊗ ³
MIKEY-DHMAC	⊕	⊕	⊕	⊕			⊙	
MIKEY-RSA-R	⊕	⊕ ²	⊕	⊕		⊙		⊗ ³
Atributo SDP "fingerprint"	–	–	–	–	–	–	–	⊗ ⁵

Legenda: ⊕ = ponto forte ⊙ = ponto de atenção não-crítico – questões operacionais no contexto VoIP ou segurança parcial ⊗ = ponto de atenção crítico – risco de segurança alto

Observações:

- ¹ Quando utiliza o IKE, pois o mesmo é baseado no esquema Diffie-Hellman o qual é conhecido suscetível a ataques *man-in-the-middle*
- ² Desde que seja utilizado um certificado digital apropriado e que possa ser verificado
- ³ Quando não é utilizado um certificado digital apropriado (p. ex., um certificado digital autoassinado)
- ⁴ Desde que seja utilizado um certificado digital apropriado e que possa ser verificado, no caso de usar uma ICP
- ⁵ Caso não sejam utilizados mecanismos de segurança adicionais para proteger a mensagem SDP no corpo SIP

3 EECMO

Este capítulo apresenta um protocolo que corresponde a uma evolução do protocolo Massey-Omura. O capítulo mostra como ocorreu essa evolução, começando do protocolo Massey-Omura tradicional, passando pela versão com curvas elípticas do mesmo, e chegando em uma versão melhorada pelo uso de assinaturas baseadas em identidades¹. O capítulo descreve ainda detalhes da implementação dessa versão melhorada.

3.1 O protocolo Massey-Omura

Dentre os vários tipos de protocolos criptográficos existentes, há aqueles que são classificados como “protocolos de três passos” – do inglês, *three-pass protocol*. Eles recebem essa denominação porque o emissor e o receptor trocam entre si 3 mensagens encriptadas em uma sequência apropriada. Ou seja, não é uma troca simultânea.

Protocolos desta natureza evitam o compartilhamento ou a publicação das chaves criptográficas – tanto de encriptação quanto de deciptação – geradas localmente por cada participante. E para que funcionem adequadamente, tais protocolos requerem um esquema de encriptação comutativo, onde encriptar primeiro com a chave x_A e depois com a chave x_B é o mesmo que encriptar primeiro com a chave x_B e depois com a chave x_A . Formalmente:

¹Assinaturas baseadas em identidade utilizam emparelhamentos bilineares. Assim, recomenda-se a leitura do Apêndice A para o leitor menos familiarizado com o assunto emparelhamento, sobretudo para um melhor entendimento da seção 3.2.

Definição 1. Considere o esquema de encriptação $(\mathcal{K}, \mathcal{M}, \mathcal{E})$ onde $\mathcal{K}$ é o conjunto de chaves, $\mathcal{M}$ é o conjunto de mensagens, e $\mathcal{E} : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}$ é um mapeamento tal que, para cada $k \in \mathcal{K}$, $\mathcal{E}$ define uma permutação sobre $\mathcal{M}$. Assim, faz sentido, para cada $k \in \mathcal{K}$, falar de encriptação inversa satisfazendo $\mathcal{E}^{-1}(k, c) = m \Leftrightarrow \mathcal{E}(k, m) = c$ para todo $m, c \in \mathcal{M}$ (m é um texto em claro e c é o texto encriptado correspondente).

Um esquema de encriptação comutativo é aquele onde, para qualquer uma das duas chaves $x_A, x_B \in \mathcal{K}$ e qualquer mensagem $m \in \mathcal{M}$, o mesmo texto encriptado pode ser obtido, seja encriptando m primeiro com x_A e, então, com x_B , seja encriptando m primeiro com x_B e, então, com x_A . Logo, $\mathcal{E}(x_B, \mathcal{E}(x_A, m)) = \mathcal{E}(x_A, \mathcal{E}(x_B, m))$.

A denominação “protocolo de três passos” ficou associada a protocolos simples, representados por uma sequência de troca de 3 mensagens encriptadas, sem o compartilhamento das chaves privadas. O primeiro protocolo deste tipo foi desenvolvido por Adi Shamir (MENEZES; OORSCHOT; VANSTONE, 1996, p. 409 - 500)².

Outro protocolo simples de três passos, que já foi objeto de pesquisas, inclusive, para propor melhorias quanto à segurança dele, é o protocolo Massey-Omura para troca de informações sigilosas sobre um canal de comunicação de domínio público (MASSEY; OMURA, 1983). Pelo fato de ser de três passos, é comum haver trabalhos – tais como (HENDERSON, 1998), (MAMBO; OKAMOTO; SAKURAI, 1997), e (SAKURAI; SHIZUYA, 1998) – que confundem o protocolo Massey-Omura com o protocolo de Shamir de três passos. Ainda que, de fato, o protocolo Massey-Omura tenha tido forte influência do esquema de três passos de Shamir, o Massey-Omura é um protocolo com características próprias. Tanto que acabou sendo patentado pelos inventores dele, James L. Massey e Jimmy K. Omura. Um fato curioso é que, apesar do protocolo ter sido publicado e patentado na década de 1980, o esquema que consta no protocolo

²É importante deixar claro que existem outros protocolos que também são executados em três passos, como o protocolo de chave pública de Needham-Schroeder (MENEZES; OORSCHOT; VANSTONE, 1996, p. 508), ou uma série de esquemas do tipo “desafio - resposta”. Mas esses protocolos e esquemas geralmente visam a autenticidade, enquanto que os protocolos que ficaram conhecidos como de três passos visam, primeiramente, a confidencialidade.

já havia sido relatado – ainda que não publicamente – na década de 1970, em (WILLIAMSON, 1974).

A ideia principal do protocolo Massey-Omura pode ser abstraída da seguinte forma: se Alice quiser enviar uma mensagem M para Bob, ela encripta a mensagem com a chave dela e envia o resultado para Bob; Bob encripta o que ele recebeu com a chave dele e envia o novo resultado de volta para Alice; ela decripta a resposta de Bob (obtendo, assim, a mensagem encriptada apenas com a chave de Bob) e envia o resultado de volta para ele que, finalmente, realiza a decriptação final e recupera a mensagem original. Alice não precisa saber qual é a chave de Bob para poder comunicar-se com ele – e vice-versa.

A Figura 10 ilustra o protocolo Massey-Omura em uma abordagem mais conceitual, elaborada a partir do texto que consta em (WASHINGTON, 2008, p. 173).

Figura 10: O protocolo Massey-Omura – visão conceitual

3.1.1 Versão do protocolo com curvas elípticas – ECMO

A descrição da versão “clássica” do protocolo Massey-Omura pode ser encontrada em (KOBLOITZ, 1994, p. 100) e (HENDERSON, 1998). E, assim como aconteceu com outros protocolos para negociação de chaves considerados “clássicos” (p. ex., Diffie-Hellman e ElGamal), o Massey-Omura também ganhou uma versão com curvas elípticas que será descrita nesta subseção, conforme consta em (KOBLOITZ, 1994, p. 178) e (WASHINGTON, 2008, p. 173 - 174).

Antes, é importante esclarecer que a denominação ECMO que foi adotada nessa dissertação não é encontrada na literatura especializada – inclusive em (KOBLOITZ, 1994) e (WASHINGTON, 2008). A denominação ECMO significa *Elliptic Curve Massey-Omura* e ela procurou seguir denominações parecidas, como, por exemplo, *Elliptic Curve Diffie-Hellman* (ECDH).

Suponha que Alice deseja enviar uma mensagem secreta $\mathcal{M}$ para Bob, sobre um canal de comunicação público e sem proteção alguma, utilizando a versão com curvas elípticas do protocolo Massey-Omura. Então, as seguintes etapas devem ser cumpridas:

- **Setup:** É estabelecido um conjunto de parâmetros públicos e comuns aos participantes do protocolo. Seja $(E, \mathbb{F}_q, q, N)$ este conjunto, no qual E é uma curva elíptica definida sobre um corpo finito $\mathbb{F}_q$ (onde q é grande), de tal maneira que seja difícil resolver o problema do logaritmo discreto em $E(\mathbb{F}_q)$. Seja $N = \#E(\mathbb{F}_q)$ o número de pontos da curva.
- **Passo 1:** Alice representa³ a mensagem secreta $\mathcal{M}$ em claro como um ponto P_m da curva $E(\mathbb{F}_q)$.

Alice ainda calcula x_A como sendo a chave secreta dela e x_A^{-1} como o respec-

³Uma descrição de como fazer essa representação – e também de como desfazê-la para recuperar a mensagem em claro – consta em (WASHINGTON, 2008, p. 174).

tivo valor inverso. O valor x_A é um número inteiro aleatório $0 \leq x_A \leq N$ tal que $\text{MDC}(x_A, N) = 1$. Já o valor x_A^{-1} é calculado pelo algoritmo de Euclides (MENEZES; OORSCHOT; VANSTONE, 1996, p. 66) de tal maneira que $x_A^{-1}x_A \equiv 1 \pmod{N}$.

Por fim, Alice calcula $M_A = x_AP_m$ e envia M_A para Bob.

- **Passo 2:** Bob recebe $M_A = x_AP_m$, mas esse valor M_A não significa nada para Bob que, não conhecendo nem x_A nem x_A^{-1} , não consegue recuperar a mensagem P_m .

Então, Bob calcula a própria chave secreta x_B e o respectivo valor inverso x_B^{-1} , da mesma forma como Alice fez no Passo 1. Em seguida, Bob calcula $M_{BA} = x_B M_A$ e o envia para Alice.

- **Passo 3:** Alice multiplica o valor M_{BA} recebido de Bob por x_A^{-1} . Uma vez que $NP_m = O$ e $x_A^{-1}x_A \equiv 1 \pmod{N}$, isto resulta no ponto $M_B = x_BP_m$ da curva E , sendo este o valor que Alice deve retornar para Bob.
- **Fechamento:** Bob multiplica o valor M_B recebido de Alice pelo valor x_B^{-1} e recupera P_m , já que $x_B^{-1}x_BP_m = P_m$. Porém, Bob obtém apenas o ponto da curva que representa a mensagem $\mathcal{M}$. Para recuperar a mensagem original $\mathcal{M}$, é preciso, ainda, executar um procedimento específico para desfazer a representação de $\mathcal{M}$ como um ponto da curva elíptica E .

É importante notar que foi possível usar um esquema de encriptação comutativo devido ao fato da multiplicação de um ponto da curva elíptica por um escalar ser uma operação comutativa.

Outro ponto a esclarecer é que a patente sobre o protocolo Massey-Omura – conforme consta em (MASSEY; OMURA, 1986) – não contempla a versão com curvas elípticas, de modo que, em princípio, o ECMO poderia ser utilizado livremente em aplicações, sem preocupações legais.

3.1.2 Discussão sobre a segurança do protocolo original

O protocolo Massey-Omura é seguro contra adversários passivos, tanto no contexto “clássico” (tradicional) quanto no correspondente elíptico. Um adversário passivo é aquele capaz somente de ler informação proveniente de um canal não seguro, sem alterá-la (MENEZES; OORSCHOT; VANSTONE, 1996).

A segurança do protocolo tradicional contra adversários passivos está baseada na dificuldade de se resolver o Problema do Logaritmo Discreto (PLD). Já no correspondente elíptico, a segurança contra adversários passivos está baseada na dificuldade de se resolver o Problema do Logaritmo Discreto para Curvas Elípticas (PLDCE) que é o seguinte: dada uma curva elíptica E definida sobre um corpo finito $\mathbb{F}_q$, um ponto $P \in E(\mathbb{F}_q)$ de ordem n , e um ponto $Q \in \langle P \rangle$, encontre um inteiro $k \in [0, n - 1]$ tal que $Q = k \cdot P$. O número inteiro k é chamado “logaritmo discreto” de Q para a base P , sendo denotado por $k = \log_P Q$ (HANKERSON; MENEZES; VANSTONE, 2004, p. 153).

Informalmente falando, o PLDCE consiste em: dado um número inteiro qualquer k e um ponto P de uma curva elíptica E , onde tanto o ponto quanto a própria curva são de conhecimento de todos (isto é, são públicos), é calculado um outro ponto Q , da mesma curva, o qual corresponde à multiplicação do ponto P pelo valor k . Ou seja, $Q = kP$. A resolução do PLDCE consiste em: dado que Q e P são conhecidos, qual é o valor de k , onde k é o logaritmo em questão? Atualmente, o PLDCE é intratável para a grande maioria das curvas. Porém, em alguns casos, o PLDCE apresenta uma solução computacionalmente factível em tempo polinomial, dependendo da curva escolhida.

Portanto, o Problema do Logaritmo Discreto (seja ele no contexto tradicional, ou no contexto elíptico, com curvas apropriadas) garante que um conteúdo secreto, capturado em alguma das transições do protocolo, não seja revelado utilizando uma criptoanálise ou mesmo um ataque de força-bruta.

Porém, o protocolo Massey-Omura – e protocolos de três passos, de uma maneira geral – estão vulneráveis a ataques desferidos por adversários ativos, que são aqueles capazes não só de ler as informações, mas também de transmitir, alterar ou eliminar as mesmas em um canal não seguro (MENEZES; OORSCHOT; VANSTONE, 1996). Ataques desse tipo são conhecidos como *man-in-the-middle*, conforme enunciado na subseção 2.2.4.

Através de ataques *man-in-the-middle* aplicados nas transições do protocolo Massey-Omura, é possível capturar dados em trânsito, de modo que eles possam ser combinados para revelar o conteúdo secreto transmitido, sem a necessidade de uma criptoanálise ou de um ataque de força-bruta. O mesmo tipo de ataque poderia ser usado para tentar enganar alguma das partes comunicantes, fazendo-as pensar que estão comunicando uma com a outra, quando, na verdade, ambas estão se comunicando com um intruso, o qual acaba tirando proveito dessa situação para modificar os dados durante as transições.

Para se ter uma noção da vulnerabilidade do protocolo frente ao *man-in-the-middle*, considere o cenário de ataque mostrado na Figura 11. Este cenário é uma das possibilidades de ataque. O Apêndice C apresenta outras.

Assim, apesar do protocolo Massey-Omura já possuir certo grau de segurança, não há nada no esquema básico do Massey-Omura (tradicional ou com curvas elípticas) que o destino (digamos, Bob) possa utilizar para verificar se foi realmente a origem (digamos, Alice) que enviou a mensagem. Bob não consegue nem mesmo verificar se ele recuperou corretamente a mensagem original, a menos que ele pergunte para Alice. Além disso, mesmo que Alice e Bob pudessem verificar-se mutuamente durante as transições do protocolo, não seria qualquer tipo de verificação ou qualquer tipo de autenticação que resolveria os problemas de vulnerabilidade. E ainda, o uso de procedimentos ou de processos extrassistêmicos poderiam ser insuficientes para evitar ataques de adversários ativos.

Figura 11: Ataque do tipo *man-in-the-middle* sobre o protocolo Massey-Omura, de modo que a mensagem secreta trocada entre as partes seja descoberta sem que estas suspeitem de nada

Fica aparente, portanto, que o protocolo carece de algum recurso para garantir o serviço de autenticidade, cuja ausência é explorada pelo *man-in-the-middle* para atacar o protocolo. Uma vez provida a autenticidade, resolvem-se os problemas de segurança do protocolo, sob determinadas condições, conforme indicado no Apêndice

C. Uma assinatura digital pode ser utilizada como tal recurso.

O uso de assinatura digital para aprimorar a segurança do Massey-Omura foi proposto inicialmente em (KOBLOITZ, 1994, p. 100). E é nesta proposta que se baseia o trabalho descrito em (WINTON, 2007), onde são apresentados dois esquemas: o EMO-1 e o EMO-2. EMO significa *Enhanced Massey Omura* e tem por objetivo tanto resolver vulnerabilidades referentes à criptoanálise quanto atribuir um esquema de assinatura digital ao esquema Massey-Omura. Segundo o autor, o esquema EMO-1 provê ao protocolo Massey-Omura um nível de segurança contra criptoanálise similar ao de um sistema de chave pública RSA. Uma versão mais reforçada do esquema EMO-1 é o esquema EMO-2, o qual adiciona uma assinatura digital ao esquema EMO-1. O objetivo desta assinatura digital é fazer com que o receptor de uma mensagem encriptada com o esquema EMO-2 possa autenticar a identidade do emissor da mensagem. No entanto, no esquema EMO-2, a introdução da assinatura digital ocorre apenas na transição final do protocolo, sendo que ataques podem ocorrer na primeira e na segunda transição, conforme mostrado na Figura 11.

Pensando nisso e também seguindo a proposta feita por Koblitz, em (DEUSAJUTE; BARRETO, 2008) é proposto o uso de emparelhamentos bilineares – já que os mesmos são definidos sobre curvas elípticas – na tentativa de estabelecer pontos de verificação entre as transições do protocolo e, assim, prover o serviço de autenticidade ao mesmo. Sobretudo na primeira transição, onde procurou-se instituir um esquema de assinatura bem similar ao de Assinaturas BLS (BONEH; LYNN; SHACHAM, 2002).

Seguindo a linha de utilização de emparelhamentos, é possível melhorar o esquema proposto em (DEUSAJUTE; BARRETO, 2008), fazendo uso de assinaturas baseadas em identidade – do inglês, *Identity-Based Signature* (IBS) – as quais podem ser mais apropriadas para determinados tipos de aplicações. Este esquema que melhora a segurança do protocolo Massey-Omura através do uso de IBS, será apresentado na próxima seção.

3.2 Aplicação de assinaturas baseadas em identidade para melhorar a segurança do ECMO

O conceito de assinatura baseada em identidade é consequência de outro conceito, o de encriptação baseada em identidade, o qual foi introduzido originalmente em (SHAMIR, 1985). Na proposta de Adi Shamir, os usuários poderiam se comunicar de maneira segura e verificarem as assinaturas um do outro sem trocar chaves públicas ou privadas, sem manter chaves em diretórios e sem usar serviços de uma terceira parte para certificar a propriedade da chave pública.

Essencialmente, a ideia da encriptação baseada em identidade é aproveitar, como chave pública, algum tipo de informação pessoal, individual e exclusiva do usuário, ao invés de uma chave pública pré-calculada e que, na maioria das vezes, é difícil de memorizar, além de que constitui uma informação a mais. Procedendo dessa forma, seria possível, por exemplo, encriptar uma mensagem tendo como chave pública o endereço de e-mail do usuário. Essa abordagem elimina, em princípio, a necessidade de uma terceira parte para certificar a propriedade da “chave pública” e-mail.

Esse esquema foi proposto inicialmente de forma teórica por Shamir. Uma implementação prática e funcional foi proposta somente bem mais tarde, por Dan Boneh e Matthew Franklin, utilizando emparelhamentos bilineares (BONEH; FRANKLIN, 2001).

Um dos esquemas IBS existentes é o esquema BLMQ, descrito pela primeira vez em (BARRETO et al., 2005). Porém, a descrição dos algoritmos que compõem esse esquema, dada a seguir, é a que consta em (BARRETO et al., 2008)⁴:

- **Setup:** dado um parâmetro de segurança k esse algoritmo seleciona um número primo n de k bits, grupos $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ de ordem n , suportando um empa-

⁴Existem leves diferenças entre o esquema BLMQ apresentado no artigo original (BARRETO et al., 2005) e o apresentado em (BARRETO et al., 2008). No artigo original, o cálculo no item 3 do algoritmo “Sign” faz $(u + h)$ e não $(u - h)$. Além disso, no algoritmo “Verify”, a contraparte g^{-h} difere de g^h . Porém, estas “leves” diferenças fazem com que o esquema descrito em (BARRETO et al., 2008) propicie um certo ganho em termos de esforço computacional, já que evita-se uma operação de divisão $- 1/g^h$.

relhamento não degenerado e computável eficientemente $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, geradores $P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$ e funções de resumo criptográfico (hash) $h_0 : \mathbb{G}_T \times \{0, 1\}^* \rightarrow \mathbb{Z}_n^*$, $h_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_n^*$. Uma chave mestra $s \xleftarrow{\$} \mathbb{Z}_n^*$ também é selecionada⁵, para a qual é associada a chave pública $Q_{pub} = sQ \in \mathbb{G}_2$. O gerador $g = e(P, Q) \in \mathbb{G}_T$ é também incluído no conjunto de parâmetros públicos $params = (k, n, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P, Q, Q_{pub}, g, e, h_0, h_1)$.

- **Private-Key-Extract:** toma como entrada um identificador da entidade $A - ID_A \in \{0, 1\}^*$ – e extrai a chave privada baseada na identidade de $A - P_A = (h_1(ID_A) + s)^{-1}P \in \mathbb{G}_1$. A entidade A pode verificar a consistência dessa chave ao constatar que $e(P_A, h_1(ID_A)Q + Q_{pub}) = g$.
- **Sign:** para assinar $m \in \{0, 1\}^*$ utilizando a chave privada baseada em identidade P_A , o signatário seleciona $u \xleftarrow{R} \mathbb{Z}_n^*$ e calcula:

1. $r \leftarrow g^u$
2. $h \leftarrow h_0(r, m)$
3. $S \leftarrow (u - h)P_A$

A mensagem assinada é a tupla $(m, h, S) \in \{0, 1\}^* \times \mathbb{Z}_n^* \times \mathbb{G}_1$.

- **Verify:** dada a identidade ID_A da entidade A , a partir do recebimento de (m, h, S) o verificador calcula:

1. $r \leftarrow e(S, h_1(ID_A)Q + Q_{pub})g^h$
2. $v \leftarrow h_0(r, m)$

O verificador aceita a mensagem assinada se, e somente se, $v = h$.

Analisando o esquema BLMQ e a versão elíptica do protocolo Massey-Omura – o ECMO – verificou-se que era possível associar um esquema ao outro, originando

⁵O “\$” na expressão “ $s \xleftarrow{\$} \mathbb{Z}_n^*$ ” indica que a seleção de s é feita de forma totalmente aleatória.

um protocolo novo, o qual foi denominado *Enhanced Elliptic Curve Massey-Omura* (EECMO). Em outras palavras, o EECMO nada mais é que o protocolo ECMO apresentado na subseção 3.1.1 acrescido de uma “camada” de autenticação derivada do esquema BLMQ.⁶

Suponha um cenário onde uma parte cliente (Alice, cuja identidade ID_A é publicamente conhecida) deseja enviar uma mensagem M secreta para outra parte servidora (Bob, cuja identidade ID_B também é publicamente conhecida), através de um canal não seguro, utilizando o EECMO. Então, as seguintes etapas desse novo protocolo devem cumpridas:

Setup do criptossistema: uma Autoridade de Confiança – *Trusted Authority* (TA)⁷ – executa o algoritmo “setup” do esquema BLMQ publicando os parâmetros $(k, n, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P, Q, Q_{pub}, g, e, h_0, h_1)$, onde $\mathbb{G}_1 = \langle P \rangle$, $\mathbb{G}_2 = \langle Q \rangle$, e $\mathbb{G}_T = \langle g \rangle$. Ainda é executado o algoritmo “Private-Key-Extract” do esquema BLMQ para obter as chaves privadas baseadas em identidade dos participantes do criptossistema – no caso, P_A para Alice e P_B para Bob.

Passo 1: Alice seleciona um valor secreto x_A e calcula x_A^{-1} como o respectivo valor inverso. O valor x_A é um número inteiro aleatório $0 \leq x_A \leq n$ tal que $MDC(x_A, n) = 1$.

Alice ainda representa a mensagem secreta M em claro como um ponto P_m da curva $E(\mathbb{G}_1)$ e calcula $M_A = x_A P_m$.

Então, Alice aplica o algoritmo “sign” do esquema BLMQ sobre M_A , usando a chave privada baseada na identidade dela – P_A – e x_A como o valor u do algoritmo “sign”. Em seguida, ela envia a tupla (M_A, h_A, S_A) mais ID_A para Bob, cuja identidade – ID_B – Alice já conhece previamente.

⁶Dentre os vários esquemas IBS existentes optou-se pelo BLMQ porque, conforme consta em (BARRETO et al., 2005, p. 10), ele é o que utiliza menos cálculos de emparelhamentos, sendo, portanto, mais eficiente quando comparado com outros esquemas similares.

⁷Também conhecida como *Key Generator Bureau* (KGB) ou como *Key Generating Centre* (KGC).

Passo 2: Bob recebe a tupla (M_A, h_A, S_A) e, utilizando a identidade ID_A de Alice (também recebida), executa o algoritmo “verify” do esquema BLMQ. Se a igualdade prevista pelo algoritmo “verify” não ocorrer, então Bob interrompe o protocolo.

Caso contrário, Bob seleciona o próprio valor secreto x_B e o respectivo valor inverso x_B^{-1} da mesma forma como Alice fez no Passo 1. Em seguida, Bob calcula $M_{BA} = x_B M_A$.

Bob também executa o algoritmo “sign” do esquema BLMQ, só que sobre M_{BA} (usando a chave privada baseada na identidade dele – P_B – e x_B como o valor u do algoritmo “sign”), enviando, em seguida, a tupla (M_{BA}, h_{BA}, S_{BA}) para Alice.

Passo 3: Alice recebe a tupla (M_{BA}, h_{BA}, S_{BA}) e, utilizando a identidade ID_B de Bob que ela já possui desde o Passo 1, executa o algoritmo “verify” do esquema BLMQ. Se a igualdade prevista pelo algoritmo “verify” não ocorrer, então é Alice quem interrompe o protocolo.

Caso contrário, Alice multiplica o valor M_{BA} recebido de Bob por x_A^{-1} . Uma vez que $nP_m = O$ e $x_A^{-1}x_A \equiv 1 \pmod n$, isto resulta no ponto $M_B = x_B P_m$ da curva E , pois:

$$x_A^{-1}M_{BA} = x_A^{-1}x_B M_A = x_A^{-1}x_B x_A P_m = x_A^{-1}x_A x_B P_m = x_B P_m$$

Alice aplica novamente o algoritmo “sign”, mas sobre M_B , de modo a obter a tupla (M_B, h_B, S_B) para enviar a Bob. Dessa vez, não é necessário enviar a identidade ID_A , pois Bob já a recebeu na primeira transição.

Fechamento: Bob recebe a tupla (M_B, h_B, S_B) e novamente executa o algoritmo “verify”, utilizando a identidade ID_A de Alice que Bob detém desde o Passo 2. Se a igualdade prevista no “verify” não ocorrer então Bob interrompe o protocolo. Caso contrário, Bob multiplica o valor M_B recebido pelo valor x_B^{-1} de modo a obter P_m , já que $x_B^{-1}x_B P_m = P_m$.

Após Bob obter o ponto da curva que representa a mensagem $\mathcal{M}$, ele recupera a

mensagem original M aplicando um processo específico para desfazer a representação de M como um ponto da curva elíptica E .

É importante ressaltar que os valores secretos x_A e x_B são selecionados por Alice e Bob, respectivamente, a cada nova rodada do protocolo.

A Figura 12 mostra esquematicamente o EECMO.

3.2.1 Discussão sobre a segurança do novo protocolo

No Apêndice C, é mostrado que o simples fato de se variar os valores x_A e x_B a cada rodada do protocolo Massey-Omura, já restringe bastante as possibilidades de ataque do tipo *man-in-the-middle* ao mesmo, apesar de não resolver por completo essa vulnerabilidade. É esperado que a associação do esquema BLMQ ao protocolo ECMO, a qual originou o EECMO, resolva a vulnerabilidade, e não a aumente, seja no conjunto como um todo ou em cada componente em separado.

Primeiramente, é preciso verificar a segurança inerente ao esquema BLMQ de forma isolada. Em (BARRETO et al., 2005, seção 3.1) é mostrado que o esquema IBS BLMQ é comprovadamente seguro, sendo que a assinatura provida por ele é infalsificável frente a ataques adaptativos de mensagens escolhidas no modelo de oráculo aleatório. O problema matemático no qual se baseia o BLMQ é o *q-Strong Diffie-Hellman Problem* (q-SDHP). Considerando-se grupos bilineares $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ e geradores $P \in \mathbb{G}_1$ e $Q \in \mathbb{G}_2$, o q-SDHP nos grupos $(\mathbb{G}_1, \mathbb{G}_2)$ consiste em, dado uma $(q + 2)$ -upla $(P, Q, \alpha Q, \alpha^2 Q, \dots, \alpha^q Q)$ como entrada, encontrar um par $(c, (c + \alpha)^{-1} P)$ para algum $c \in \mathbb{Z}_p^*$.

Uma particularidade do esquema BLMQ é que ele não possui problemas relacionados ao uso de emparelhamentos assimétricos e curvas ordinárias, pois o esquema não requer operação de resumo criptográfico sobre um subgrupo cíclico de curva elíptica.

Analisando-se o processo de verificação, é possível identificar que a chave pri-

Figura 12: O protocolo Massey-Omura na versão com curvas elípticas (ECMO), melhorado, em termos de segurança, com o uso de assinaturas baseadas em identidade

vada baseada em identidade – cujo método de obtenção a partir da identidade é uma simplificação do método sugerido em (SAKAI; KASAHARA, 2003) – é fator determinante na verificação. Suponha que Alice, utilizando o esquema IBS BLMQ, obtenha a chave privada dela baseada em identidade – P_A – a partir do algoritmo “Private-

Key-Extract”. Com essa chave, Alice assina uma mensagem $\mathcal{M}$ com o algoritmo “Sign” do esquema e envia para Bob a tupla $(\mathcal{M}, h, S)$ produzida pelo algoritmo. Bob verifica a assinatura BLMQ aplicando o algoritmo “Verify” de modo a constatar que $h_0(r, \mathcal{M}) = h = h_0(g^{x_A}, \mathcal{M})$ ou simplesmente que $r = g^{x_A}$, onde x_A é um valor inteiro secreto escolhido de forma aleatória por Alice. Em outras palavras, Bob tem que calcular r fazendo uso das propriedades de emparelhamentos bilineares – vide Apêndice A, seção A.1 – e verificar se o resultado obtido é igual à g^{x_A} :

$$\begin{aligned} r &= e(S, h_1(ID_A)Q + Q_{pub})g^h = e((x_A - h)P_A, h_1(ID_A)Q + Q_{pub})g^h = e((x_A - h)(h_1(ID_A) + s)^{-1}P, h_1(ID_A)Q + sQ)g^h \\ &= e((x_A - h)(h_1(ID_A) + s)^{-1}P, (h_1(ID_A) + s)Q)g^h = e((x_A - h)P, (h_1(ID_A) + s)^{-1}(h_1(ID_A) + s)Q)g^h \end{aligned}$$

Note, nesta última passagem, que se houvesse qualquer outro $P_x = (h_1(ID_x) + s)^{-1}P$, não seria possível obter $(h_1(ID_A) + s)^{-1}(h_1(ID_A)) = 1$, derivando-se então para outro valor qualquer. Porém, se S for verdadeiramente baseado em P_A , então obtém-se $(h_1(ID_A) + s)^{-1}(h_1(ID_A)) = 1$ e, daí:

$$\begin{aligned} r &= e((x_A - h)P, Q)g^h = e((x_A - h)P, Q)e(P, Q)^h = e((x_A - h)P, Q)e(hP, Q) = \\ &= e((x_A - h)P + hP, Q) = e(x_AP - hP + hP, Q) = e(x_AP, Q) = e(P, Q)^{x_A} = g^{x_A} \end{aligned}$$

Aplicando-se h_0 sobre este resultado e sobre $\mathcal{M}$ recebido, chega-se à $v = h_0(r, \mathcal{M}) = h_0(g^{x_A}, \mathcal{M}) = h$ recebido.

Portanto, uma das premissas do esquema BLMQ é garantir tanto a integridade quanto a confidencialidade da chave privada baseada em identidade, de modo que somente o portador da identidade na qual se baseia a chave possa ter acesso à ela.

A integridade pode ser garantida aplicando-se o método para verificação de consistência da chave privada descrito no algoritmo “Private-Key-Extract”, de modo a constatar se $e(P_A, h_1(ID_A)Q + Q_{pub}) = g$. Ou seja, de posse da identidade ID_A , da chave privada baseada na mesma – P_A – e de parâmetros que a TA publica, basta calcular o emparelhamento acima e verificar se o resultado é igual ao parâmetro público g :

$$e(P_A, h_1(ID_A)Q + Q_{pub}) = e((h_1(ID_A) + s)^{-1}P, h_1(ID_A)Q + sQ) = e((h_1(ID_A) + s)^{-1}P, (h_1(ID_A) + s)Q) = e((h_1(ID_A) + s)(h_1(ID_A) + s)^{-1}P, Q) = e(P, Q) = g$$

Procedendo dessa forma seria possível evitar que o portador recebesse – e, eventualmente, armazenasse – uma chave privada que não fosse baseada na identidade legítima dele.

Quanto à confidencialidade da chave privada, não há maneiras existentes ou derivadas do próprio esquema que possam garanti-la. Por um lado, no caminho da TA até chegar ao portador, a chave pode sofrer um ataque passivo (p. ex., escuta simples) e ser capturada. Por outro, uma TA mal intencionada pode calcular a chave privada baseada na identidade de alguém e fornecê-la para um atacante, ou ela própria pode usá-la indevidamente, já que a identidade é algo público e a TA detém o conhecimento da chave mestra s .

Portanto, garantir a confidencialidade implica assumir outras duas premissas: que a TA é totalmente confiável e que a chave privada baseada na identidade de um portador chega até ao mesmo – e é mantida por ele – sem sofrer ataques que permitam capturá-la.

Assumindo-se as premissas anteriores e, decorrente delas, que um atacante não consiga obter a chave privada baseada em identidade, resta a ele tentar outras formas de ataque. Como o contexto onde o BLMQ está sendo considerado é o ECMO, nada mais natural que o atacante tente utilizar contra o EECMO as mesmas técnicas de ataque que foram utilizadas – com sucesso em alguns cenários – contra o ECMO, conforme mostrado no Apêndice C.

Se bem que, no caso do EECMO, não dá para utilizar todas as técnicas. É o caso do ataque de repetição (*replay attack*), pois, no contexto do EECMO, significaria tentar fazer um signatário – atuando como verificador – validar uma assinatura que ele mesmo gerou. Uma vez que qualquer um dos componentes da assinatura BLMQ –

h e S , adulterados ou não – possui o ID do signatário na composição deles, o “Verify” no signatário falharia sempre nessas condições. Isto porque o signatário utilizaria qualquer outro ID que não fosse o dele próprio, já que, por definição, não se espera que uma assinatura BLMQ seja verificada com a própria identidade do verificador e sim com a da outra parte. Daí, em nenhum momento ter-se-ia $(h_1(ID_A) + s)^{-1}(h_1(ID_A)) = 1$ e, por conseguinte, não haveria a redução à $r = e((x_A - h)P, Q)g^h$ de onde se chega à $r = g^{x_A}$ conforme mostrado.

Então, restaria ao atacante adulterar h e/ou S e submetê-los ao “Verify” da parte prevista para receber/validar a assinatura, no intuito de tentar burlar a verificação e, assim, neutralizar o esquema BLMQ no EECMO, reduzindo-o novamente ao ECMO.

Considere o primeiro caso, onde $S' = x_\lambda S$ e h não muda. Então, $S' = x_\lambda(x_A - h)P_A = (x_\lambda x_A - x_\lambda h)P_A$. Suponha que o atacante tente passar a assinatura BLMQ (M, h, S') pelo “Verify” do receptor da assinatura. É possível mostrar que o atacante não terá sucesso.

Para simplificar, considere o desdobramento do cálculo de r feito anteriormente, especificamente quando $r = e((x_A - h)P + hP, Q)$. Como parte-se do princípio de que o atacante não consegue acesso à P_A e leva-se em conta que “Verify” utiliza o ID_A legítimo, então $(h_1(ID_A) + s)^{-1}(h_1(ID_A)) = 1$. Assim, com $S' = x_\lambda S$, obtém-se:

$$r' = e((x_\lambda x_A - x_\lambda h)P + hP, Q) = e(x_\lambda x_A P - x_\lambda h P + hP, Q) = e((x_\lambda x_A - x_\lambda h + h)P, Q) = e(P, Q)^{(x_\lambda x_A - x_\lambda h + h)} = g^{(x_\lambda x_A - x_\lambda h + h)} \neq g^{x_A}$$

No segundo caso, $h' = x_\lambda h$ e S não muda. Novamente, partindo-se de $r = e((x_A - h)P + hP, Q)$ – neste caso, o h dentro de $(x_A - h)$ não é afetado, pois essa diferença faz parte do componente S , o qual não muda – obtém-se:

$$r' = e((x_A - h)P + x_\lambda h P, Q) = e(x_A P - h P + x_\lambda h P, Q) = e((x_A - h + x_\lambda h)P, Q) = e(P, Q)^{(x_A - h + x_\lambda h)} = g^{(x_A - h + x_\lambda h)} \neq g^{x_A}$$

Na terceira combinação possível, $S' = x_\lambda S = (x_\lambda x_A - x_\lambda h)P_A$ e $h' = x_\lambda h$. Então:

$$r' = e((x_\lambda x_A - x_\lambda h)P + x_\lambda hP, Q) = e(x_\lambda x_A P - x_\lambda hP + x_\lambda hP, Q) = e(x_\lambda x_A P, Q) = e(P, Q)^{x_\lambda x_A} = g^{x_\lambda x_A} \neq g^{x_A}$$

Nos casos anteriores, assume-se que a mensagem original que foi assinada não é alterada. A alteração afeta apenas a assinatura.

Caso o atacante mude a estratégia e tente adulterar a mensagem (não descobri-la), substituindo M por outra mensagem qualquer M_λ , ele também não teria sucesso. Para tentar essa estratégia, o atacante gera $h' = h_0(g^{x_\lambda}, M_\lambda)$. Mas não gera o respectivo $S' = (x_\lambda - h')P_\lambda$, pois o mesmo não se baseia em P_A e, conforme já mostrado, o “Verify” falharia. Então, o atacante submete a tupla (M_λ, h', S) ao “Verify” do receptor da assinatura. Analogamente ao que já foi mostrado, obtém-se $r' = e((x_A - h)P + h'P, Q) = e(x_A P - hP + h'P, Q) = g^{x_A - h + h'}$. Daí, $v = h_0(g^{x_A - h + h'}, M_\lambda) \neq h_0(g^{x_\lambda}, M_\lambda) = h'$.

Mesmo que o atacante utilizasse h e S legítimos, ainda assim ele não teria sucesso. Afinal, detectar a falta de integridade em uma tupla (M_λ, h, S) é o mínimo que se espera de qualquer esquema de assinatura digital, não importa a natureza dele.

Ressaltando que, da forma como foi definida a associação ECMO - BLMQ para originar o EECMO, o valor secreto x_A (digamos, de Alice) varia a cada assinatura realizada pelo algoritmo “Sign”. Consequentemente, os valores S_A e h_A vão variar também, já que ambos são baseados em x_A . O mesmo raciocínio aplica-se para x_B , de Bob.

Isso faz com que as assinaturas BLMQ geradas por Alice e por Bob não sejam reaproveitadas por um atacante em rodadas diferentes do EECMO. Em outras palavras, uma assinatura BLMQ capturada em uma rodada do protocolo (seja na primeira, na segunda ou na terceira transição), só pode ser utilizada na mesma rodada de execução. E somente contra o verificador esperado da assinatura, sem a possibilidade da mesma ser utilizada em ataques de repetição, conforme já explicado.

Além disso, como os componentes h e S da assinatura BLMQ não são transforma-

dos após cada transição do EECMO, muito menos voltam a transitar entre as partes comunicantes, eles não derivam outros valores que pudessem ser utilizados em novos ataques às outras transições de uma mesma rodada do protocolo.

Portanto, considerando-se que o esquema BLMQ é comprovadamente seguro – desde que assumidas as premissas anteriormente descritas – e assumindo-se que, assim como no esquema ECMO, os valores secretos x_A e x_B mudam a cada rodada do protocolo, é possível inferir que o EECMO é seguro e resolve a vulnerabilidade do protocolo Massey-Omura frente a ataques do tipo *man-in-the-middle*, tais como aqueles mostrados no Apêndice C.

3.3 Implementação do EECMO

A Figura 13 mostra o diagrama de classes UML referente ao protocolo EECMO⁸. A partir deste diagrama de classes foi gerado um pacote, na linguagem de programação Java, utilizando o *Java Development Kit 6* (JDK 1.6) e o “NetBeans IDE”, versão 6.5, uma ferramenta para desenvolvimento de aplicativos Java.

A classe “EECMO” é a classe principal e possui operações comuns ao lado cliente e ao lado servidor da comunicação. Aproveitando-se do polimorfismo, a classe possui três operações de “setup” para permitir o uso tanto do ECMO tradicional quanto do EECMO, este último nos cenários previstos na subseção 3.3.1.

A classe “EECMOClient” representa especificamente o lado cliente da comunicação e possui operações inerentes somente ao lado cliente. Os nomes das operações são autoexplicativos, mas é importante comentar rapidamente sobre a operação “start”, que só existe no lado cliente e marca efetivamente o início do protocolo. É nela que a mensagem textual secreta a ser transmitida é representada como ponto da curva elíptica. Sem equiparar a mensagem textual a um ponto da curva elíptica não é pos-

⁸Por questões de simplificação e para facilitar o entendimento, não foram incluídos no diagrama os atributos das classes.

Figura 13: Diagrama de classe UML do protocolo EECMO

sível executar os cálculos do protocolo, menos ainda as verificações baseadas em emparelhamentos. Analogamente, a classe “EECMOServer” representa o lado servidor e possui operações inerentes somente ao lado servidor. Os nomes delas também são autoexplicativos, merecendo destaque a operação “retrieveMessage” que desfaz a representação da mensagem como ponto da curva, retornando a mensagem secreta original no formato texto, em claro.

As classes apresentadas anteriormente possuem também operações específicas para o EECMO. Essas operações são totalmente dependentes das operações “sign” (assinatura) e “verify” (verificação), disponibilizadas pela classe “BLMQlbsScheme”, a qual implementa o esquema de assinatura baseada em identidade BLMQ, e cujos cálculos internos são os mesmos apresentados na seção 3.2.

As classes descritas até aqui são suficientes para criar aplicações que utilizem tanto o ECMO quanto o EECMO. Além delas, há classes específicas para tornar possível o acoplamento do EECMO ao SIP.

É o caso da classe “EECMOfromRFC4567”, que procura seguir as orientações passadas pela RFC 4567, de modo que o EECMO possa ser acoplado ao SIP de forma padronizada e interoperável. Por conta disto, a função principal desta classe é suportar o *façade*⁹ fornecido pelo pacote, *façade* este representado por outra classe, a “eecmoKeyMgmt”.

O grau de abstração das operações “eecmoStepOne”, “eecmoStepTwo” e “eecmoStepThree” que essa classe disponibiliza é bem alto, pois todas elas devolvem uma cadeia de caracteres representando a mensagem a ser trocada entre as partes comunicantes, contendo todas as informações necessárias, incluindo os componentes

⁹Um *façade* provê uma interface de alto nível que torna o subsistema em questão mais fácil de se usar, fornecendo uma interface única para um conjunto de interfaces em um subsistema (GAMMA et al., 1994). Isto não significa, entretanto, que obrigatoriamente o *façade* tenha que ser implementado como uma interface em linguagens de programação orientadas a objeto. O conceito de interface aqui é no sentido de abstração de complexidades ou funcionalidades que não precisam ser expostas. Assim, um dos muitos benefícios de se utilizar *façades* é que eles tornam uma biblioteca de *software* mais fácil de se entender e usar.

da assinatura BLMQ. Além disso, tal mensagem já é devolvida codificada em Base64, pronta para ser inserida em uma mensagem SDP de um corpo SIP. Inclusive, o primeiro argumento de cada uma das operações citadas anteriormente é justamente uma mensagem em Base64. Ou seja, a mensagem codificada em Base64 que sai de uma operação entra diretamente na outra, seguindo os passos previstos para o EECMO.

Existem também classes utilitárias, como “EECMOUtil” – a qual disponibiliza operações para serem utilizadas por outras classes – e “EECMOStatistics”. Esta foi idealizada para tratar da coleta dos dados estatísticos do protocolo, os quais são, essencialmente, os tempos de execução das principais operações. Assim, após uma execução completa do protocolo, basta acessar os atributos públicos desta classe para obter os dados estatísticos, que vão sendo coletados a medida que o protocolo é executado.

No diagrama de classes UML da Figura 13 é mostrado ainda a interface “SMSPairing”, a qual contém classes que dão suporte às outras, disponibilizando operações diversas relacionadas com corpos finitos, curvas elípticas e emparelhamentos bilineares. Essas classes foram desenvolvidas pelo professor Paulo Barreto e também subsidiaram a implementação prática que é relatada em (BARRETO et al., 2008). Os cálculos de emparelhamentos existentes nas classes utilizam o algoritmo BKLS, pois o mesmo é o mais rápido para cálculo de emparelhamentos em curvas elípticas MNT, que é a família de curvas elípticas utilizada nas mesmas classes. Mais especificamente, são utilizadas curvas MNT4. Esta família de curvas elípticas ordinárias (isto é, não supersingulares), de ordem prima, foram propostas por Miyaji, Nakabayashi e Takano, cujas iniciais dos sobrenomes dão origem ao nome da família. Eles foram os primeiros a descrever um procedimento para gerar curvas ordinárias com grau de mergulho pequeno: 3, 4 ou 6 (MENEZES, 2005). Apesar de raras, são curvas amigáveis a emparelhamentos. O “4” em “MNT4” corresponde ao grau de mergulho da curva¹⁰.

¹⁰Outra forma de interpretar esse “4” é que o grau de segurança, na prática, é 4 vezes maior que o tamanho do corpo algébrico sobre o qual a curva é definida. Por exemplo, com 256 *bits* equipara-se ao RSA 1024 *bits*. Porém, mesmo este nível de segurança do RSA já não é recomendado para muitas aplicações. Por isso, a interface “SMSPairing” oferece ainda a possibilidade de se trabalhar com 272 e

As classes foram projetadas e implementadas de modo que, se necessário, fosse possível executar apenas o protocolo ECMO tradicional. A Figura 14 deixa isso mais claro ao mostrar uma visão conceitual da relação façade EECMO (em conformidade com a RFC 4567) – protocolo EECMO – protocolo ECMO tradicional.

Figura 14: Visão conceitual da relação façade EECMO – protocolo EECMO – protocolo ECMO

3.3.1 Cenários de TA contemplados na implementação

Em um esquema IBS BLMQ normal, presume-se que os participantes do esquema estejam sob a mesma TA. Este é o cenário típico e que a implementação feita suporta. Neste caso, não há muito o que fazer quanto ao processo de *setup* do EECMO. Pois, conforme descrito na seção 3.2, o *setup* do EECMO é o mesmo do esquema BLMQ. Como presume-se que uma TA já esteja estabelecida quando os participantes começarem a comunicar-se, o *setup* do EECMO nada mais será do que inicializar os atributos das classes do protocolo com os respectivos parâmetros públicos da TA. Isto é feito por uma das operações “setup” da classe “EECMO”.

Entretanto, para aplicações reais este pode não ser o cenário existente. Ou seja, por diversas razões (p. ex., escalabilidade, segregação, localização geográfica etc.) pode ser que os participantes no mundo real estejam associados a domínios diferentes

313 *bits*, equivalentes, respectivamente, a 1088 e 1252 *bits* quando comparados ao RSA.

e, daí, a TAs diferentes também.

Estando o signatário e o verificador sob o domínio de TAs diferentes, o signatário de uma mensagem deve enviar o parâmetro Q_{pub} da TA dele ao verificador da mensagem. Além disso, é necessário também que as TAs partilhem dos mesmos valores para os demais parâmetros públicos $(k, n, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P, Q, g, e, h_0, h_1)$ formando, assim, uma Federação de TAs. Entretanto, isto não é um problema, pois, com exceção da chave mestra s (e, conseqüentemente, de Q_{pub}), os demais valores podem ser publicamente acertados previamente. Para ser mais exato, as TAs precisam pré-combinar apenas k , o tipo de emparelhamento e a ser usado, a família de curvas elípticas, e as funções de resumo criptográfico (*hashes*) h_0 e h_1 .

O valor k funciona como um seletor, isto é, dado k e uma família de curvas elípticas, amigáveis a emparelhamentos, obtém-se a curva E , seu respectivo corpo base $(\mathbb{G}_1)$, n , e p , o qual corresponde ao número de elementos existentes no corpo base de E . A partir da curva E tanto obtém-se o ponto público gerador P quanto deriva-se a curva $E2$, já que esta é uma curva definida sobre a extensão $(\mathbb{G}_2)$ do corpo base de E . De $E2$ obtém-se o gerador Q e calcula-se o emparelhamento e de P e Q , de tal forma que $e(P, Q) = g$, um valor em $\mathbb{G}_T$.

Esta estratégia de derivação é justamente o que faz um outro “setup” provido pela implementação do EECMO, para o cenário onde as partes comunicantes estão sob TAs diferentes. Neste caso, a parte cliente (signatária) deve enviar à parte servidora (verificadora) não somente o Q_{pub} da TA dele, mas também o valor de k e de e ¹¹. Ao receber os valores de k e e , é realizado o *setup* EECMO da parte servidora a partir desses valores. É como se, momentaneamente, a parte servidora estivesse sob o domínio da TA da parte cliente, apenas para fazer a verificação da assinatura BLMQ.

Para tanto, além de k e de e , a parte servidora deve utilizar também o Q_{pub} enviado

¹¹Em uma implementação mais sofisticada, poderia-se enviar também algum tipo de indicação para a família de curvas elípticas. Porém, a família é algo que pode ser pré-fixado na implementação, já que não há tantas famílias de curvas amigáveis a emparelhamentos.

pela parte cliente, já que este parâmetro não pode ser derivado, uma vez que ele está relacionado diretamente com a chave mestra (totalmente secreta) da TA do signatário. A desvantagem neste cenário é que há um cálculo a mais de emparelhamento, além daqueles previstos para verificar a assinatura BLMQ: o que produz $g = e(P, Q)$.

Portanto, a implementação realizada oferece suporte para estes dois tipos de cenários, dando mais flexibilidade para as aplicações que utilizarem o EECMO.

• • •

Este capítulo apresentou o que se pode chamar de linha evolutiva do protocolo Massey-Omura, desde sua idealização (o chamado formato “clássico” do protocolo), passando pela versão com curvas elípticas do protocolo (nitidamente suscetível a ataques do tipo *man-in-the-middle*) e chegando em uma versão melhorada.

Ainda que muitos possam afirmar que o conceito do protocolo Massey-Omura (e de protocolos de três passos em geral) não passa de uma mera curiosidade acadêmica, não tendo muita aplicação, o protocolo pode ser aplicado em situações particulares onde exista uma transição semelhante ou onde a transição de três passos sirva para garantir que o protocolo não foi interrompido após a primeira transição, estabelecendo uma forte dependência entre as partes comunicantes.

Além disso, a versão melhorada do protocolo pode ser aplicada em cenários onde, além das particularidades citadas anteriormente, haja problemas de segurança não totalmente resolvidos (p. ex., existência de vulnerabilidade a ataques do tipo *man-in-the-middle*) e que necessitem de um esquema que possa prover autenticidade, confidencialidade e integridade de forma simples, mas eficiente.

Pelo que foi descrito neste capítulo e pelo que foi visto no capítulo anterior, sobretudo na seção 2.2, o SIP representa uma ótima oportunidade para se aplicar o EECMO. Para tanto, o capítulo apresentou ainda detalhes da implementação do EECMO de modo que o mesmo possa ser acoplado ao SIP de maneira padronizada.

4 A SEGURANÇA DO SIP MELHORADA COM O PROTOCOLO EECMO

Dada a semelhança em termos de funcionamento entre o SIP e o protocolo EECMO, é possível acoplar um protocolo ao outro, originando um mecanismo de segurança alternativo aos vários mecanismos que o SIP já possui. Entretanto, na prática, este acoplamento precisa ser feito convenientemente, inclusive para que o tempo de sinalização e o tamanho da mensagem SIP permaneçam dentro de intervalos de valores aceitáveis para o contexto da Telefonia IP e, principalmente, do SIP.

Assim, este capítulo descreve como o EECMO foi acoplado ao SIP, abordando, ainda, os aspectos de segurança relacionados a esse acoplamento. O capítulo também mostra a prova de conceito que foi realizada com o novo mecanismo de segurança proposto, os valores obtidos em testes funcionais, e a análise dos mesmos.

4.1 Acoplamento do EECMO ao SIP

O acoplamento do EECMO ao SIP considerou as seguintes abordagens possíveis para o estabelecimento de uma TA no contexto de uma típica rede SIP:

- Centralizada: a TA poderia ser instituída em um servidor de autenticação de usuários, o qual poderia ou não ser um servidor SIP, ou seja, o servidor de autenticação poderia ser uma entidade à parte, dedicada.
- Federada: haveria TAs instituídas em servidores *proxies* SIP ou, o que seria mais natural, em servidores de registro SIP. Em uma Federação de TAs, nem

todas elas possuem o mesmo conjunto de parâmetros públicos, mas cada uma possui sua própria chave mestra.

Em ambas as abordagens, procurou-se fazer o acoplamento de uma maneira padronizada, seguindo as recomendações e orientações que constam na RFC 4567 (ARKKO et al., 2006).

Esta RFC descreve uma solução mais geral, na forma de um *framework*, o qual permite integrar protocolos/esquemas de gerenciamento de chaves (existentes ou futuros), e negociar os respectivos parâmetros, usando o SDP. Apesar de, até o momento, a RFC só reconhecer o MIKEY, o *framework* é genérico o suficiente para acomodar outros esquemas (PEREA, 2008). Tanto que novos esquemas podem ser registrados oficialmente junto à *Internet Assigned Numbers Authority* (IANA).

A descrição formal dos atributos SDP para acomodar esquemas de gerenciamento/negociação de chaves é feita na RFC 4567 utilizando *Augmented Backus-Naur Form* (ABNF) que, por sua vez, é compatível com a descrição BNF do SDP feita na RFC 4566. A descrição é a seguinte:

```
key-mgmt-attribute = key-mgmt-att-field ":" key-mgmt-att-value
key-mgmt-att-field = "key-mgmt" ; fixo
key-mgmt-att-value = 0*1SP prtcl-id SP key-mgmt-data
```

Note, na descrição anterior, que o elemento que permite, de fato, alguma definição proprietária é o "key-mgmt-att-value". Ele pode começar com nenhum, um ou mais de um espaço em branco ("SP") antes da identificação do protocolo/esquema. Essa identificação é dada por "prtcl-id", e deve ser registrada formalmente junto à IANA, seguindo as orientações que constam na própria RFC 4567. Por exemplo, no caso do MIKEY, "prtcl-id" é igual à "mikey". Após a identificação do protocolo/esquema, obrigatoriamente deve haver um espaço em branco. Em seguida a ele, vem o "key-mgmt-data",

cujo conteúdo, em geral, consiste de parâmetros de segurança (incluindo chaves) necessários para tornar a comunicação segura e de informações necessárias para garantir que a mensagem é autêntica. Tal conteúdo – que deve estar codificado em Base64 – depende exclusivamente do protocolo/esquema de gerenciamento/negociação de chaves que for utilizado (ARKKO et al., 2006).

Uma vez definido o “key-mgmt-att-value” proprietário do protocolo/esquema em questão, junta-se o mesmo ao valor (fixo) de “key-mgmt-att-field”, obtendo-se o “key-mgmt-attribute”. Na prática, este atributo SDP é que viabiliza o acoplamento padronizado de outros protocolos/esquemas de gerenciamento de chaves ao SIP.

Assim, foi idealizado um “key-mgmt-att-value” próprio para o EECMO. Para simplificar, não foram definidos espaços antes da identificação do protocolo, a qual foi assumida como sendo “eecmo”, mesmo não tendo sido registrada formalmente junto à IANA. Quanto ao “key-mgmt-data”, foi idealizado um pequeno cabeçalho, contendo informações de controle, e um corpo de tamanho variável, onde são acomodados os parâmetros em função do passo que estiver sendo executado no protocolo. A Figura 15 descreve o “key-mgmt-data” elaborado para o EECMO.

Figura 15: O elemento “key-mgmt-data” idealizado para o EECMO em conformidade com a RFC 4567

Exemplo de “key-mgmt-attribute” do EECMO:

a=key-mgmt:eecmo NDEwNDAwMDAzMTMDAR/ULYoE5Bvob8k139d5daGp1KjyBTqMM2m+E+roR7sm

B4kk169vUQIAczDkASGWTdEOg0fQQFvgRHbW8iFDurjddUqSF0u3fEanf0MqbeS6AUOFupz9kPN26
myRb17VsRc2xYM6M49CjLPxRFH7EjM13IoPi1fmVw==

Observações:

- As informações referentes ao parâmetro de segurança (em *bits*) e ao tipo de emparelhamento são necessárias para executar o *setup* EECMO do lado servidor, caso este esteja sob uma TA diferente da TA do lado cliente. Nesta abordagem de TAs federadas, é preciso que o lado cliente envie o Q_{pub} da TA dele para o lado servidor. Já no cenário de mesma TA, isto não é necessário. Por isto, o envio do Q_{pub} é opcional e depende de como as TAs estão instituídas.
- Da forma como foi idealizado o cabeçalho do “key-mgmt-data”, ele consegue indicar a presença ou não do Q_{pub} através da indicação do tamanho dele. Se for zero, significa que o Q_{pub} está ausente na mensagem e, portanto, os participantes da comunicação estão associados a uma mesma TA centralizada.
- Os tamanhos dos componentes h e S de uma assinatura BLMQ podem variar e, portanto, é preciso indicar o tamanho desses componentes, de modo que o receptor da mensagem possa corretamente interpretá-la e extrair dela as informações necessárias.

A interpretação mencionada anteriormente consiste em decodificar a mensagem (que está em Base64), extrair o cabeçalho e, a partir das informações contidas nele, extrair os demais componentes. Somente depois da extração dos demais componentes – a mensagem que foi assinada (M_A , M_{BA} ou M_B) e os componentes da assinatura BLMQ (h e S) – é que pode-se dar continuidade na execução do protocolo.

A Figura 16 mostra conceitualmente o protocolo EECMO acoplado ao SIP, em um cenário onde as partes comunicantes estão sob a mesma TA. A figura abstrai detalhes do tipo presença de um servidor *proxy* SIP, sinalização trapezoide etc., para facilitar

o entendimento. O objetivo da figura é mostrar o acoplamento do EECMO ao SIP, indicando quais parâmetros precisam ser enviados nas mensagens SIP (requisição ou resposta), de modo a viabilizar a execução do EECMO.

Note que a figura referencia a implementação do EECMO descrita na seção 3.3. Note ainda que a abstração obtida através do façade “eecmoKeyMgmt” – o qual segue as recomendações e orientações da RFC 4567 – torna fácil o acoplamento do EECMO ao SIP.

Figura 16: O protocolo EECMO acoplado ao SIP – as partes comunicantes estão sob domínio da mesma TA – visão conceitual

A Figura 17 é uma variação da figura anterior, mostrando o cenário onde as partes comunicantes estão sob TAs diferentes. Neste caso, não é necessário enviar o Q_{pub} na última transição, pois ele já foi enviado na primeira e foi armazenado pelo lado servidor ao realizar o *setup* EECMO.

Figura 17: O protocolo EECMO acoplado ao SIP – as partes comunicantes estão sob domínio de TAs diferentes

4.1.1 SIP e a segurança provida pelo EECMO

Na subseção 3.2.1 do Capítulo 3, mostrou-se que o esquema BLMQ de assinatura baseada em identidade, quando associado ao ECMO (formando o EECMO), resolve a vulnerabilidade do Massey-Omura quanto ao *man-in-the-middle*. Assim, o EECMO acoplado ao SIP torna este mais seguro também contra ataques do tipo *man-in-the-middle*. Além disso, a dinâmica de transições característica do Massey-Omura gera uma dependência entre as partes, até que se complete o processo de sinalização. Isto evita aqueles ataques *man-in-the-middle* particulares do SIP – mais especificamente, do modo DHHMAC do MIKEY – que foram descritos na subseção 2.2.4.

Porém, conforme mostrado também no capítulo anterior, há algumas premissas que devem ser garantidas para que o esquema BLMQ – e, consequentemente, o EECMO acoplado ao SIP – sejam, de fato, seguros:

- Garantir a confidencialidade da chave privada baseada em identidade. Isto significa garantir que:
 1. A TA seja totalmente confiável e não consiga forjar (seja em proveito próprio ou de terceiros) a chave privada.
 2. A chave privada baseada em identidade seja de conhecimento único e exclusivo do portador legítimo dela.
- Garantir a integridade da chave privada baseada em identidade.

Felizmente, no mundo real (inclusive no contexto SIP), há meios para garantir tais premissas.

A TA é uma entidade extremamente simples. Essencialmente, ela armazena uma chave mestra previamente criada e, mediante solicitações, provê chaves privadas baseadas em identidade, através do algoritmo “Private-Key-Extract”. A criação da chave mestra pode tanto ocorrer automática e sigilosamente (p. ex., um coprocessador criptográfico pode gerá-la sem que seja exibido o valor obtido) quanto através de um rito processual conhecido como “cerimônia de chaves”. Existem algumas formas de se executar esse rito e tais formas normalmente são ditadas por políticas e procedimentos de segurança. Mas, de um modo geral, um sistema geraria uma chave mestra e a dividiria em partes. Cada parte seria entregue a um portador, chamado de custodiante. Em seguida, cada custodiante, secretamente, inseriria a parte da chave sob custódia dele no repositório definitivo da chave mestra. Quando todas as partes estiverem inseridas no repositório definitivo, elas são unidas sistemicamente, de modo a formar uma chave mestra única e íntegra. Depois, cada parte da chave mestra que estava em poder de um custodiante diferente é armazenada de forma segura – por exemplo, dentro de envelopes lacrados guardados em cofres com dispositivos de abertura altamente seguros – para algum eventual uso futuro. Todo esse processo normalmente é auditado e tem testemunhas (em princípio, idôneas, assim como os custodiantes).

Este rito processual é geralmente utilizado em sistemas críticos onde, no caso de uma perda total e irreversível da chave mestra, por qualquer motivo, a cerimônia possa ser realizada novamente, na qual novos custodiantes receberiam as partes da chave mestra que foram armazenadas de forma segura, conforme explicado.

Além disso, este rito poderia viabilizar a instalação de uma TA distribuída, isto é, uma TA que logicamente é uma só, mas que fisicamente está em locais geográficos diferentes. Neste caso, cada repositório físico local da TA possuiria a mesma chave mestra. E a inserção dessa chave mestra em cada local físico diferente poderia ser feita por cerimônias de chaves.¹

Quanto ao repositório da chave mestra, este é um assunto bem resolvido atualmente, pois, normalmente, uma chave mestra seria armazenada em algum tipo de *Hardware Security Module* (HSM). E os HSMs atuais atendem às recomendações que constam na norma FIPS PUB 140-2 (INFORMATION TECHNOLOGY LABORATORY OF THE NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, 2001), inclusive no nível 4. Apenas para dar uma noção do nível de segurança destes dispositivos, se um HSM que atende ao FIPS PUB 140-2 nível 4 sofrer uma tentativa de violação, todo o conteúdo interno dele se autodestrói.²

Uma vez que a TA passe a ser confiável, a próxima premissa a se garantir é que as chaves privadas baseadas em identidades geradas pela TA sejam entregues somente aos donos legítimos e guardadas por estes de forma segura.

Quanto à guarda da chave, existem diversas possibilidades e pode-se fazer um comparativo com a guarda de outros tipos de chaves privadas. Por exemplo, assim como acontece com a chave privada de um certificado digital ICP-Brasil do tipo A3,

¹Note que TA distribuída e Federação de TAs são conceitos diferentes. Em uma Federação de TAs, as TAs também estão espalhadas mas cada qual com sua respectiva chave mestra.

²Atualmente já está em elaboração um *draft* que seria uma evolução do FIPS PUB 140-2: o 140-3. Nesta atualização, o NIST introduz uma nova seção sobre segurança relacionada com *software*, um nível adicional de segurança (nível 5) e novos requisitos para *Simple Power Analysis* (SPA) e *Differential Power Analysis* (DPA).

chaves privadas baseadas em identidade poderiam ser armazenadas em cartões inteligentes (*smartcards*) ou *tokens* criptográficos. Em ambos os casos ainda há uma senha de proteção chamada comumente de *Personal Identification Number* (PIN). Seguindo a analogia, o componente *S* da assinatura BLMQ seria gerado internamente no *chip* ou no *token*, de modo a não expor, de forma alguma, a chave privada baseada em identidade, a qual estaria também dentro do dispositivo criptográfico.

Uma alternativa menos sofisticada seria guardar a chave privada baseada em identidade localmente no computador pessoal, encriptada, por exemplo, por AES, cuja chave simétrica poderia ser um PIN ou algo derivado dele. Entretanto, neste caso, medidas adicionais de segurança devem ser tomadas, principalmente quanto ao risco de ataques por programas maliciosos (*malwares*), capaz de capturar tanto o arquivo encriptado contendo a chave privada baseada em identidade, quanto o próprio PIN, transmitindo-os, em seguida, para uma entidade externa (via Internet, por exemplo). Normalmente, *softwares* do tipo antivírus minimizam esse risco.

Já a entrega da chave privada ao legítimo portador pode não ser algo tão simples de se proteger. Atualmente, a maneira mais apropriada de se obter informações pela Internet seria através de serviços *web* – do inglês, *web services*. Estes já possuem padrões bem definidos de segurança, como o WS-Security. Entretanto, o WS-Security na sua modalidade mais segura faz uso de certificados digitais X.509 em ambas as pontas, para prover uma segurança fim-a-fim. E o ideal seria ter uma solução para o SIP que não dependesse de certificados digitais pelos motivos já expostos no Capítulo 2.

Uma alternativa para o contexto SIP seria determinar que a execução do algoritmo “Private-Key-Extract” só ocorresse no momento do registro do *user agent*. Em outras palavras, a chave privada baseada em identidade seria fornecida somente quando o registro do *user agent* fosse feito junto a um servidor SIP de registro. Porém, ainda há um certo risco em se proceder assim, pois há ataques específicos nesse caso, tais

como o sequestro de registro (*Registration Hijacking*), em que um atacante intercepta uma requisição REGISTER legítima de um *user agent* e, alterando o campo de cabeçalho “Contact”, registra-se junto a um servidor SIP de registro. (SINNREICH; JOHNSTON, 2006). Dessa forma, o registro do *user agent* ocorre sem problemas, mas o endereço (SIP URI) para contato é o do atacante.

Ainda que um atacante não soubesse quando um *user agent* fosse fazer um registro (reduzindo a possibilidade de ataque a uma tentativa), e ainda que a comunicação fosse protegida por SSL (para garantir a confidencialidade e integridade no trânsito das informações), o atacante poderia simplesmente fazer-se passar por um *user agent* legítimo, personificando-o. Ou seja, a maior preocupação no momento do registro é quanto à autenticidade. Isto implicaria na necessidade de um processo de registro que conseguisse verificar e atestar a identidade do *user agent* registrante.

Há algumas maneiras de se fazer isso de forma sistêmica, mas as formas mais seguras tendem a usar processos com intervenção humana para validar a identidade. Embora isso possa trazer certa complexidade ao contexto SIP, seria em apenas um momento.

É importante ressaltar que compensaria avaliar a relação risco-benefício no caso das soluções sistêmicas, porque, mesmo que um atacante consiga a chave privada baseada na identidade de alguém, isto não significa facilidade de ataque no contexto SIP. De posse de uma chave privada legítima, um atacante poderia forjar uma assinatura BLMQ fazendo-se passar por alguém que ele não é. Para tornar a farsa maior, nada impede o atacante de adulterar o campo de cabeçalho “From” colocando um SIP URI que não é dele e, sim, de quem ele roubou a chave privada. Quem receber a mensagem SIP dificilmente desconfiará de algo, achando que está sendo contatado legitimamente. Porém, para realizar este ataque, provavelmente o atacante teria que estar entre o *user agent* atacado e o *proxy* SIP do mesmo. Caso contrário, devido às propriedades de roteamento inerentes aos *proxies* SIP, a resposta do *user agent*

atacado iria para o *user agent* legítimo.

Por fim, a garantia da integridade de uma chave privada P_A , baseada em uma identidade ID_A , pode ser feita com um cálculo de emparelhamento apenas – $e(P_A, h_1(ID_A)Q + Q_{pub}) = g$. O momento de se fazer isso pode ser qualquer um. Partindo do princípio de que a chave privada seja fornecida no momento do registro, a integridade da mesma poderia ser verificada antes dela ser armazenada localmente pela primeira vez.

Até aqui foi falado da segurança do EECMO para o SIP. Entretanto, é importante destacar os seguintes pontos que reforçam ainda mais a segurança e são decorrentes mais da natureza do SIP:

- O EECMO, da maneira como foi acoplado ao SIP, não teria problemas com relação a ataques do tipo *bidding-down*, pois mesmo que um atacante decodificasse o “key-mgmt-attribute” do EECMO e alterasse o parâmetro de segurança e/ou o tipo de emparelhamento, o único efeito que isso surtiria seria provocar uma interrupção no protocolo já na primeira transição. E apenas no caso de TAs federadas, onde o parâmetro de segurança “k” determina o *setup* no lado servidor. Isto porque, conforme já explicado, para o protocolo funcionar é preciso haver um sincronismo quanto aos parâmetros públicos de ambos os lados participantes da comunicação.
- Quando um usuário de um UAC decide iniciar uma chamada para outro usuário de um UAS, ele já sabe exatamente qual o SIP URI (ou seja, a identidade) do destino. Dada a forma como foi proposto o EECMO, obrigatoriamente o *ID* a ser utilizado no “Verify” do passo 3 será o mesmo com o qual se iniciou a chamada. Analogamente, ao extrair, no passo 2, a identidade da origem que vem no campo de cabeçalho “From”, o UAS aproveitará essa identidade no “Verify” tanto do

próprio passo 2 quanto do fechamento. O risco para segurança seria se um atacante conseguisse forjar esse campo de cabeçalho – fazendo-se passar pelo UAC legítimo – e, ao mesmo tempo, tivesse posse da chave privada baseada em identidade do UAC legítimo. Já foi indicado que no contexto SIP isto não é tão simples de ser feito.

- A própria natureza de uma rede SIP implica em o atacante ter que se posicionar adequadamente entre o *user agent* e os vários tipos de servidores SIP para que ataques surtam algum efeito, conforme já mencionado. Uma possibilidade – muito usual – seria a implantação de agentes do atacante no próprio *user agent* atacado. Se o *user agent* estiver em um computador pessoal, medidas adicionais de segurança – tais como os antivírus, já mencionados – minimizariam bastante o risco.

4.2 Prova de Conceito

4.2.1 O *softphone* Endurance

Para realizar a prova de conceito, construiu-se um *softphone*³ SIP visando realizar um teste funcional completo, de modo a avaliar o EECMO na prática.

O *softphone* construído foi baseado essencialmente no *softphone* SIP que consta em (PEREA, 2008). Entretanto, foi necessário fazer algumas modificações no mesmo, de modo não só a incorporar o EECMO, mas também acrescentar algumas facilidades para que fosse possível visualizar a troca de mensagens SIP durante o processo de sinalização e fazer algumas medições necessárias. Ainda que menos importante, também foram feitas melhorias na interface com o usuário. Essa versão melhorada do *softphone* recebeu a denominação “Endurance”.

³Programas de *software* que implementam aplicações de comunicação multimídia são tipicamente chamados de *softphone*. O conceito de *softphone* é bem genérico, e não necessariamente implica na utilização de sinalização através do SIP. Há, sim, *softphones* SIP, mas há também *softphones* que utilizam outros protocolos de sinalização, como o H.323 (PEREA, 2008).

Assim como no caso do EECMO, o Endurance foi implementado utilizando o JDK 1.6 e o Netbeans IDE 6.5. Entretanto, foi necessário ainda utilizar o *framework* JAIN SIP para viabilizar toda a parte de sinalização SIP, o JAIN SDP para o tratamento de mensagens SDP e o *Java Media Framework* (JMF) para tratar as questões de multimídia.

JAIN SIP é uma especificação de *Application Programming Interface* (API) Java para o SIP. Ele é desenvolvido em *Java 2 Standard Edition* (J2SE) e provê, aos desenvolvedores de aplicações, interfaces padronizadas de serviços SIP que são funcionalmente compatíveis com a RFC 3261. Mais especificamente, JAIN SIP disponibiliza recursos para construir e tratar mensagens SIP, utilizar subcamadas de transação e transporte, além de permitir acesso aos diálogos SIP. Assim, JAIN SIP oferece acesso às funcionalidades nas subcamadas de diálogo, transporte e transação mais acesso parcial à subcamada central do SIP.

Analogamente, JAIN SDP é uma interface que permite codificar e decodificar mensagens SDP. Dentre as interfaces, há aquelas que permitem criar a mensagem SDP em si e outras que a tratam linha a linha.

Quanto ao JMF, ele corresponde a uma API para tratamento de aplicações Java baseadas em mídia. Ela permite que programadores Java desenvolvam código para capturar, apresentar, armazenar e processar mídias baseadas em tempo. Além disso, ela pode ser estendida para suportar tipos adicionais de mídia e executar processamento personalizado. Adicionalmente, JMF define uma API RTP opcional que permite a transmissão e recepção de fluxos RTP (PEREA, 2008).

Atualmente, as implementações de JAIN SIP e JAIN SDP são controladas pelo *National Institute of Standards and Technology* (NIST). Já o JMF tem a implementação suportada tanto pela Sun Microsystems quanto pela IBM. Apesar de JAIN SIP não ser um padrão IETF, ele é parte de um esforço para criar um conjunto de APIs para o desenvolvimento de serviços para vários protocolos de Internet e de telefonia, provendo

uma interface padrão para pilhas SIP proprietárias (SINNREICH; JOHNSTON, 2006). Portanto, pode-se afirmar que ele é um padrão de fato – ainda que não de direito – sendo largamente aceito no mercado.

Uma vez construído o *softphone*, o próximo passo foi incorporar nele o protocolo EECMO, utilizando o pacote Java que representa a implementação do EECMO descrita na seção 3.3 do Capítulo 3, sobretudo o façade EECMO representado pela classe “eecmoKeyMgmt”.

A Figura 18 mostra a arquitetura geral do Endurance. A Figura 19 é semelhante, porém detalha as interações entre as camadas que compõem o *softphone*.

Figura 18: Arquitetura funcional do *softphone* Endurance – adaptado de (PEREA, 2008)

4.2.2 Execução da prova

Para a execução do teste funcional, além da construção de um *softphone* SIP que incorporasse o EECMO, foi necessário utilizar um servidor SIP que pudesse ser tanto *proxy* como servidor de registro. Como o objetivo maior do teste funcional era

Figura 19: Arquitetura funcional do *softphone* Endurance mostrando as interações entre as camadas do *softphone* – adaptado de (PEREA, 2008)

a comunicação fim-a-fim, e não a avaliação do funcionamento de um servidor SIP, aproveitou-se o *proxy* SIP que consta em (PEREA, 2008), o qual tem ainda a função de servidor de registro. Este *proxy* SIP é bem simples, atuando essencialmente como um repassador de mensagens SIP para viabilizar o estabelecimento da sessão.

Os objetivos desse teste funcional foram:

1. Comprovar a viabilidade de acoplamento do EECMO ao SIP, testando o esquema de negociação de chaves proposto em uma situação real de sinalização entre dois *user agents*.
2. Avaliar o eventual impacto ocorrido na sinalização devido ao aumento do tamanho das mensagens SIP trocadas entre os *user agents*, por conta do *façade* EECMO.
3. Avaliar também o eventual impacto ocorrido na sinalização decorrente do tempo de processamento adicional gasto com o EECMO.

O cenário do teste realizado é mostrado na Figura 20: dois computadores interligados por um roteador *wireless* em uma rede local. Os computadores utilizados foram do tipo *notebook*, sendo que um deles era um Intel-Pentium de 1,66 Ghz com 504 *Mbytes* de memória RAM e o outro um Intel-Celeron de 2 Ghz e 2 *Gbytes* de RAM. Ainda que este último seja um pouco mais robusto, atualmente ele pode ser considerado um modelo convencional de mercado. O protocolo de transporte utilizado foi o UDP, pois ele é mais apropriado para comunicações *user agent – proxy* SIP (SINNREICH; JOHNSTON, 2006, p. 111).

Figura 20: Cenário do teste funcional realizado com o *softphone* Endurance

Este cenário não é um cenário “Trapezoide SIP” (que é o cenário mais comum no contexto de uma rede extensa e dispersa como é a Internet), porém é o suficiente para os objetivos apresentados anteriormente. Essencialmente, a diferença é que ao invés de se conectar em domínios diferentes (como normalmente acontece com usuários do Skype, GoogleTalk, MSN etc.), ambos *user agents* conectam-se ao mesmo servidor de domínio. Assim, o cenário poderia representar, por exemplo, o ambiente corporativo de grandes empresas.

O teste funcional como um todo consistiu de várias rodadas, onde, em cada uma, foram realizadas as seguintes ações:

1. O *user agent* chamador (cujo usuário é “Alice” e tem endereço SIP URI igual à “alice@ocean.com”) realiza as configurações necessárias no *softphone* Endurance, através da aba “Configuração”, mostrada na Figura 21.

Figura 21: Interface do *softphone* Endurance para permitir configurações diversas

2. Uma vez devidamente configurado, Alice liga o *softphone*. Nesse momento, ocorre o processo de registro do *softphone* junto ao servidor *proxy* SIP “ocean.com”, isto é, o registro junto ao domínio “ocean.com”.

Na prática, este processo só seria necessário uma única vez, no primeiro acesso. Os demais acessos seriam realizados não com registro mas sim com autenticação junto ao servidor de domínio. Porém, abstraiu-se este nível de complexidade fazendo o registro a cada rodada do teste, devido à simplicidade do servidor SIP utilizado e que foi descrito anteriormente. É importante notar que este fato não influencia nos objetivos do teste, sobretudo na avaliação do impacto causado pelo tempo de processamento, uma vez que as medições de tempo realizadas

ocorreram somente durante o processo de sinalização.

3. Após registrado, Alice escolhe quem deseja chamar e inicia a chamada, através do botão “Chamar” da aba “Comunicação” do *softphone*, conforme mostra a Figura 22. Ainda através desta figura, é possível notar que o Endurance permite comunicação através de áudio, vídeo e mensagens escritas.

Figura 22: Interface do *softphone* Endurance para permitir comunicação

Neste ponto começa a ser executado o EECMO, primeiro fazendo-se o *setup* do lado cliente e depois executando o método “*eecmoStepOne*” do façade, de modo a preparar a mensagem EECMO (codificada em Base64) a ser atrelada ao atributo “*a=key-mgmt*” do corpo SDP da mensagem SIP “*INVITE*”. A Figura 23 mostra detalhes da sinalização, destacando o *INVITE* gerado e enviado para o *user agent* chamado.

4. O *user agent* chamado (cujo endereço SIP URI igual à “*bob@ocean.com*”) no-

Figura 23: Sinalização SIP através do *softphone* Endurance

tifica ao usuário dele, “Bob”, que há uma chamada em andamento. Bob, ao decidir atender a chamada, utiliza o botão “Atender” do *softphone* existente na aba “Comunicação”, conforme mostrado na Figura 22 (no caso, o botão está desabilitado, pois a figura ilustra a condição do *softphone* referente ao lado que iniciou a chamada). Neste momento, é executado o *setup* no lado servidor e, em seguida, o método “eecmoStepTwo” do façade. Uma das ações realizadas por este método é verificar a assinatura BLMQ que consta na mensagem recebida. Se ela estiver íntegra, o método também gera uma mensagem EECMO a ser atrelada ao corpo SDP da resposta SIP “200 OK”.

Enquanto a resposta “200 OK” é preparada, o *user agent* chamador já recebe respostas SIP para indicar a tentativa de conexão com o *user agent* chamador (“100 Trying”) e que o mesmo está sendo alertado da chamada (“180 Ringing”).

5. Deste ponto em diante, não há mais intervenção humana. Ao receber a resposta

“200 OK”, o lado cliente executa o método “eecmoStepThree”, o qual também verifica a assinatura BLMQ que consta na mensagem recebida, e, estando tudo correto, prepara a mensagem final EECMO, que será enviada ao lado servidor no corpo SDP da mensagem “ACK”.

6. Finalmente, o *user agent* chamado recebe a mensagem ACK e executa o método “eecmoFinishing” do façade. Este verifica a assinatura BLMQ que consta na mensagem recebida e, estando correta e íntegra, recupera a mensagem secreta na forma de ponto da curva elíptica. Depois, transforma-a em texto novamente, exibindo a mensagem secreta recuperada na aba “Configuração” (Figura 24).

Assim, conclui-se o processo de sinalização SIP usando o esquema de negociação de chaves EECMO. Daí em diante, a mensagem secreta pode ser utilizada, por exemplo, como chave simétrica para o AES do protocolo SRTP, de modo a proteger a sessão de mídia que é estabelecida logo em seguida à conclusão da sinalização.

Conforme comentado, o teste funcional necessitou de várias rodadas de execução das ações anteriormente descritas, no intuito de avaliar casos de uso diferentes. Os casos de uso surgiram em decorrência da variação do parâmetro de segurança (em *bits*), do tipo de emparelhamento, e do fato das partes comunicantes estarem ou não sob uma mesma TA. Especificamente, utilizou-se os parâmetros de segurança 256, 272 e 313 *bits*, os quais equiparam-se, na prática, ao RSA de 1024, 1088 e 1252 *bits* de segurança, respectivamente. Quanto ao tipo de emparelhamento, variou-se entre Eta e Ate. Essas variações resultaram em doze cenários, cada qual sendo uma rodada de teste

Essa versatilidade para gerar casos de uso foi possível graças aos recursos de configuração do *softphone* mostrados na Figura 21, os quais também permitiram que fosse executada uma rodada sem segurança habilitada, para fins de comparação. As-

Figura 24: O *softphone* Endurance configurado automaticamente após a conclusão, com sucesso, de um processo de sinalização SIP utilizando o EECMO

sim, antes de cada rodada, os *softphones* instalados em cada um dos computadores utilizados no teste (vide Figura 20) eram configurados de modo a possuir o mesmo parâmetro de segurança e o mesmo tipo de emparelhamento, daí equalizando os parâmetros públicos das TAs que regem cada *softphone*. Este artifício foi utilizado porque não foi construída e estabelecida uma TA propriamente dita para a realização do teste funcional. A TA foi apenas simulada. Isto é, ao iniciar o *softphone* pelo botão “Ligar”, realizava-se o *setup* da TA a partir do parâmetro de segurança e do tipo de emparelhamento, aproveitando a estratégia de derivação descrita na subseção 3.3.1.

Também ao ligar o *softphone*, simulava-se a obtenção da chave privada baseada na identidade de cada *user agent*, isto é, baseada no SIP URI do *user agent*. O SIP URI, muitas vezes, acaba refletindo a identidade pública de um usuário real (PEREA, 2008).

Quanto à indicação de que os *softphones* comunicantes estavam ou não sob a mesma TA, de modo a simular tanto a abordagem centralizada quanto a abordagem federada, foi utilizada uma caixa de seleção específica (*checkbox*) existente na aba “Comunicação” do *softphone*. Esta indicação era feita sempre pelo *user agent* chamador. Em uma situação real não seria preciso fazer esta indicação explícita, caso houvesse algum fator delimitador de domínios de TAs. Para simular a existência de outra TA, foi utilizada uma classe que possuía dois métodos de “setup” de TA, cada qual gerando um valor secreto s diferente e, conseqüentemente, valores Q_{pub} diferentes também.

Os artifícios descritos para configurar e simular TAs – e também para obter a chave privada baseada em identidade – não invalidam o teste funcional, pois, em uma situação real, provavelmente aconteceria algo similar, conforme discutido na subseção 4.1. Ou seja, ao registrar o *softphone* ou ao autenticar o usuário dele, a TA seria acessada para prover tanto os parâmetros públicos dela quanto a chave privada baseada em identidade.

Ao serem executadas as rodadas, essencialmente apenas os *softphones* estavam ativos. Isto é, não haviam outros aplicativos em execução no computador, com exceção, claro, do sistema operacional, do ambiente de execução Java e do programa que representava o *proxy* SIP, o qual era bem simples e não consumia praticamente recurso algum do computador.

A cada rodada de execução, o *softphone* gerava valores relacionados ao tamanho das mensagens SIP e ao tempo de processamento⁴. Os valores eram gerados através dos recursos de coleta de estatísticas tanto do EECMO quanto do próprio *softphone*, sendo exibidos – para serem posteriormente planilhados – na aba “Estatísticas” do Endurance.

⁴Uma característica não considerada na obtenção dos valores durante a sinalização foi o tipo de mídia selecionado (só áudio, áudio e vídeo etc.), pois o teste funcional foi feito focando o processo de sinalização e não a sessão de mídia. Tanto que eventuais atrasos gerados por conta da configuração multimídia nos *user agents* foram desconsiderados.

Como havia diferença na capacidade de processamento entre os computadores utilizados no teste funcional, foram necessários dois ciclos completos de rodadas, de modo a inverter a localização dos *softphones* de “Alice” e de “Bob”. Assim, o primeiro ciclo foi executado com o *softphone* “de Bob” no computador Intel-Pentium e o *softphone* “de Alice” no computador Intel-Celeron. No segundo ciclo, inverteu-se.

A Tabela 2 mostra os valores planilhados, após os mesmos terem sido coletados⁵. Os valores correspondem às médias aritméticas dos dois ciclos de rodadas realizados.

4.2.2.1 Refinamento dos valores obtidos

Na Tabela 2 são mostrados os valores obtidos a partir de dois ciclos apenas de execução do *softphone* Endurance, cada qual consistindo de 12 rodadas referentes aos 12 casos de uso gerados. Para fins estatísticos, o ideal mesmo seria realizar bem mais que dois ciclos para obter-se o tempo médio de cada uma das operações, tanto das exclusivamente SIP (preparação de cabeçalho e corpo da mensagem SIP, envio da mensagem etc.) quanto das exclusivamente EECMO. Como há intervenção humana no processo, seria bem complexo automatizar um teste funcional envolvendo diretamente o *softphone*, no qual fossem executados muitos ciclos de rodadas.

Devido à forma como o EECMO foi acoplado ao SIP, não há uma dependência significativa entre o conjunto de operações exclusivamente SIP e o conjunto de operações exclusivamente EECMO, e que possa fazer com que o tempo de processamento despendido com um conjunto influencie no outro. Isto porque o EECMO é acoplado ao SIP através de um atributo SDP, ficando transparente para a operação “Construção do corpo SIP”, a qual considera o “key-mgmt-attribute” do EECMO apenas como mais um atributo a incluir na mensagem SDP – e todas as mensagens SIP consideradas no teste possuem uma mensagem SDP no corpo delas. Assim, é possível tratar as operações de cada um dos conjuntos de maneira apartada e, daí, executar testes funcionais

⁵Na tabela, os valores referentes aos tempos totais gastos tanto no UAC quanto no UAS levam em conta o uso do EECMO.

separados para obter tempos médios que fossem, estatisticamente, mais realistas.

No caso das operações exclusivamente SIP, tal teste, de certa forma, já foi feito. Levando-se em conta que as variações realizadas para gerar os casos de uso não influenciam nas operações do SIP, que a Tabela 2 foi construída a partir de dois ciclos completos, cada qual composto por 12 casos de uso, e que as mensagens SIP de “INVITE”, “200 OK” e “ACK” são estruturalmente semelhantes quanto ao tipo e à forma, pode-se afirmar que já foi feito um teste funcional correspondente a 72 iterações, um número bem razoável para se calcular uma média aritmética aceitável. Seguindo esse raciocínio e trabalhando-se os valores da tabela, obtém-se os seguintes tempos médios de processamento para as operações SIP: 0.19 ms para “Preparação do cabeçalho SIP”, 3.31 ms para “Preparação da mensagem SDP” e 3.18 ms para “Conexão e envio” da mensagem. A operação “Construção do corpo SIP” corresponde, na prática, a um simples comando Java e, provavelmente devido a isso, não despendeu tempo de processamento algum – mesmo que extremamente pequeno – em nenhuma das iterações. Ainda trabalhando-se os valores da tabela, é possível obter os seguintes tempos médios de processamento em cada etapa⁶: 5.75 ms para o “INVITE”, 7.75 ms para o “200 OK” e 6.42 ms para o “ACK”.

É importante notar que em uma requisição “ACK” normal, sem o EECMO, não há um corpo SDP. O corpo SDP foi introduzido no “ACK” apenas para transportar o “key-mgmt-attribute” do EECMO. Entretanto, para requisições “INVITE” e respostas “200 OK”, o corpo SDP serve tanto para transportar o “key-mgmt-attribute” do EECMO quanto para viabilizar o esquema “oferta-resposta”. Em outras palavras, mesmo sem o EECMO, o corpo SDP continuaria sendo necessário no “INVITE” e no “200 OK”. Quanto aos cabeçalhos SIP, eles são preparados da mesma forma, estando ou não o EECMO presente.

Por isso, os tempos médios calculados anteriormente para as operações exclusi-

⁶Neste caso, considerando-se dois ciclos, cada qual com 12 rodadas, o que equivale a 24 iterações.

vamente SIP são aceitáveis também para o caso de sinalização sem EECMO. No caso específico do “ACK”, supondo o mesmo sem um corpo SDP, o cálculo do tempo médio de processamento da operação “Preparação da mensagem SDP” levaria em conta 48 iterações (número ainda aceitável para fins estatísticos), resultando em 1.91 ms. No caso do tempo médio para a etapa “ACK”, refazendo os cálculos, resultaria em 4.13 ms. Ambos valores estão razoavelmente próximos dos respectivos valores no caso do EECMO estar presente.

Quanto às operações exclusivamente EECMO, o raciocínio anterior não pode ser empregado, em razão dos diferentes casos de uso gerados. Assim, foi preciso fazer um teste funcional isolado e específico para o EECMO. Por se tratar de um teste envolvendo o protocolo apenas, esse teste pôde ser feito através de um programa comum, sem um *softphone* atrelado. Este programa (desenvolvido em Java, utilizando o JDK 1.6) foi construído de tal forma que houvesse uma sequência de chamadas dos métodos do façade EECMO na qual as mensagens trocadas entre os métodos estivessem codificadas em Base64. Isto porque, essas mesmas sequência e dinâmica na troca de mensagens são as que existem no *softphone* Endurance, usado no teste funcional da prova de conceito, descrito nessa seção.

O teste consistiu na execução de cada um dos casos de uso, uma única vez. Porém, para cada caso de uso, as operações exclusivamente EECMO foram executadas 100 vezes (iterações), após as quais se obtia a média aritmética dos valores. Devido a esse número de iterações, optou-se por realizar o teste em apenas um dos computadores utilizados na prova de conceito, o de menor capacidade de processamento – Intel-Pentium de 1,66 Ghz com 504 *Mbytes* de memória RAM (mesmo assim, ele pode ser considerado convencional para os padrões atuais de mercado). O *Java Runtime Environment* (JRE) utilizado foi o 6.

Os valores obtidos nesse teste são mostrados na Tabela 3 e foram coletados através de operações existentes na classe “EECMOSTatistics” da implementação do

EECMO.

4.2.3 Avaliação e discussão acerca do desempenho

4.2.3.1 Tempo de processamento

Trabalhando-se apropriadamente os valores da Tabela 2 é possível constatar que o tempo de processamento despendido com as operações do EECMO corresponde, em média, a 98% do tempo de processamento total de todas as operações (operações exclusivamente SIP mais as operações exclusivamente EECMO). Mesmo considerando-se os valores refinados em 4.2.2.1, obter-se-ia, certamente, um percentual parecido. Ou seja, em uma comparação pura e simples, o EECMO aumenta bastante o tempo de sinalização. A questão é se, mesmo com esse aumento, o tempo de sinalização ainda fica dentro de padrões aceitáveis de telefonia e de comunicação humana.

A recomendação Y.1531 do *Telecommunication Standardization Sector of International Telecommunications Union* (ITU-T) (INTERNATIONAL TELECOMMUNICATION UNION, 2007) define uma importante métrica para o SIP, o CSD. Segundo a Y.1531, o CSD corresponde ao tempo decorrido do momento que o UAC envia uma mensagem “INVITE” até a recepção, por parte do UAS, da mensagem “ACK” correspondente. O intervalo de tempo que inicia-se no momento da recepção do “INVITE”, onde o UAS é notificado de que há uma chamada para o usuário dele, e termina no momento em que a chamada é atendida, não é considerado no CSD. Também não é considerado o intervalo de tempo existente entre a recepção do “200 OK” no UAC e o envio do “ACK” pelo mesmo. Tomando como base a Figura 25, a definição do CSD poderia ser expressa da seguinte forma⁷: $CSD = (t6 - t1) - (t3 - t2) - (t5 - t4)$.

Para se avaliar o impacto do EECMO no CSD é preciso considerar os tempos de processamento despendidos com as operações do protocolo. Além disso, parte das operações do EECMO ocorre entre a recepção do “200 OK” e o envio do “ACK” (Passo

⁷Na figura, “DNS” é o servidor de DNS, “LS” é o servidor SIP de localização, e “PS” é um *proxy* SIP.

Figura 25: A métrica *call set-up delay* (CSD) para o SIP – extraído de (INTERNATIONAL TELECOMMUNICATION UNION, 2007, p. 6)

3). Portanto, a definição oficial do CSD dada pela Y.1531 precisará ser “flexibilizada” e adaptada. Outra questão a ser tratada nessa “flexibilização” são os valores de referência para representar determinados intervalos de tempo, uma vez que a Y.1531 também não traz esses valores ou mesmo outras métricas como referência.

Assim, é possível, ao mesmo tempo, simplificar o desenho da Figura 25 e incluir as considerações anteriores, resultando em um novo desenho, mostrado na Figura 26.

Os valores σ_1 , σ_2 , e σ_3 correspondem aos tempos de processamento despendidos com as mensagens SIP de “INVITE”, “200 OK” e “ACK”, respectivamente. Estes valores já foram abordados e analisados sob a ótica de médias aritméticas em 4.2.2.1, mostrando-se valores bem irrelevantes no geral, ainda mais quando se utiliza o EECMO. Portanto, para simplificar o raciocínio do CSD considerando o EECMO, estes valores não serão considerados.

Os valores δ_1 , δ_2 , δ_3 , e δ_4 , correspondem aos tempos de processamento despen-

Figura 26: A métrica *call set-up delay* (CSD) para o SIP, considerando o uso do EECMO dados nos passos 1, 2, 3, e fechamento do EECMO, respectivamente.

O PDD – ou *post-selection delay* – e o PPD – ou *answer signal delay* (ASD) – são duas importantes métricas no contexto da telefonia. Eles foram definidos originalmente em (INTERNATIONAL TELECOMMUNICATION UNION, 1999). No contexto SIP utilizando-se UDP, o PDD é definido como sendo o intervalo de tempo entre a primeira mensagem “INVITE” enviada pelo terminal chamador e primeira mensagem “180 Ringing” (no caso bem sucedido) recebida do terminal chamado. Já no caso do PPD, como o SIP é um protocolo de três transições, a definição adaptada para o contexto SIP difere um pouco da original. Isto porque, quando há o atendimento por parte do UAS, é enviada a resposta “200 OK”, e assim que esta é recebida pelo UAC, o mesmo deve enviar um “ACK”. Logo, a definição do PPD para o contexto SIP deve incluir a requisição “ACK”. O PPD é particularmente crítico porque se o valor for alto, as primeiras palavras da conversação serão perdidas, já que os canais de mídia ainda não foram completamente estabelecidos (SCHEINER, 2005, p. 68 - 70). Tanto o PDD quanto o PPD são influenciados pela latência da rede.

Uma vez definidos os principais componentes de tempo mostrados na Figura 26, é possível adaptar a definição do CSD dada pela recomendação X.1531, e gerar uma definição mais apropriada para o contexto SIP com EECMO: $CSD' = \delta_1 + PDD + \delta_2 + \delta_3 + PPD + \delta_4$ (lembrando que os valores σ_1 , σ_2 , e σ_3 foram desconsiderados por serem irrelevantes).

Os valores dos δ_i podem ser estimados a partir dos valores refinados que constam na Tabela 3. Já no caso dos valores de PDD e PPD, eles precisariam ser obtidos semelhantemente como ocorreu com os valores δ_i . Como o foco do presente trabalho não era estressar a avaliação do desempenho, optou-se por reaproveitar algum estudo feito anteriormente e de onde fosse possível extrair valores de referência tanto para o PDD quanto para o PPD. Após pesquisas, chegou-se em (SCHEINER, 2005), no qual é relatado um experimento completo e bem abrangente, envolvendo métricas tais como PDD e PPD.

Neste trabalho, os valores foram obtidos a partir de simulações realizadas em laboratório, mas usando *round-trip time* (RTT) realistas para emular a latência de rede. O trabalho considerou chamadas em uma *Local Area Network* (LAN), além de chamadas locais (CL), nacionais interestaduais (CNI), internacionais intracontinentais (CIIA) e internacionais intercontinentais (CIIE), variando-se desde situações ideais até situações adversas, como a existência de *jitter*. O trabalho ainda analisou as métricas simulando desde redes de baixa velocidade até redes de banda larga. Foram avaliados cenários com um e com dois *proxies* SIP (“Trapezoide SIP”).

Todas essas variantes resultaram em diversos valores de PDD e PPD. Destes, tomou-se os seguintes valores de referência para o PDD: 20 ms para LAN, 150 ms para CL, 310 ms para CNI, 400 ms para CIIA, e 750 ms para CIIE. Para o PPD: 10 ms para LAN, 150 ms para CL, 300 ms para CNI, 400 ms para CIIA, e 740 ms para CIIE. Com exceção dos valores de referência para LAN, os demais valores foram obtidos levando-se em conta um *jitter* de 10%. Isto para um cenário com apenas um *proxy*

SIP.

Para um cenário “Trapezoide SIP”, tomou-se como valores de referência para o PDD: 10 ms para LAN, 210 ms para CL, 390 ms para CNI, 640 ms para CIIA, e 1350 ms para CIIE. Para o PPD: 10 ms para LAN, 200 ms para CL, 390 ms para CNI, 630 ms para CIIA, e 1380 ms para CIIE.

Utilizando os valores de referência tomados anteriormente, bem como os valores que constam na linha “Total Geral” da Tabela 3, e aplicando-se a fórmula definida para CSD’, chega-se na Tabela 4. A tabela corresponde a um cruzamento envolvendo os casos de uso do EECMO e os diversos tipos de chamadas, tanto no cenário com um *proxy* SIP quanto no cenário “Trapezoide”. Este cruzamento pôde ser realizado porque o EECMO é executado apenas nos *user agents*, acrescentando tempo de processamento apenas às operações normais já existentes no SIP. Outro fato que torna esse cruzamento ainda mais aceitável é que há uma coincidência entre (SCHEINER, 2005) e a prova de conceito realizada no presente trabalho: as mensagens SIP de “INVITE”, “200 OK” e “ACK” daquele possuem tamanhos bem próximos dos tamanhos finais (isto é, com sobrecarga) das respectivas mensagens geradas para transportar o “key-mgmt-attribute” do EECMO no “pior” caso – 313 *bits*, TAs diferentes. Ou seja, as mensagens SIP utilizadas em ambos experimentos têm tamanhos semelhantes.

Embora a Y.1531 seja uma recomendação do ITU-T, a mesma não define um valor alvo⁸ para o CSD. O máximo que ela faz é indicar que o estabelecimento bem sucedido de uma chamada deve ocorrer “dentro de um tempo máximo específico, T_m (a ser especificado).” (INTERNATIONAL TELECOMMUNICATION UNION, 2007, p. 3, tradução nossa). Assim, como o ITU-T ainda não definiu um valor alvo para o CSD, os valores da Tabela 4 foram comparados com valores alvo definidos por outra recomendação aceita atualmente no contexto da telefonia e, inclusive, utilizada em outros trabalhos que também estudaram o desempenho do SIP (CURCIO; LUNDAN, 2002; MEYER; PORTMANN, 2008): a

⁸Esse valor alvo deve ser entendido como um limite superior.

recomendação E.721 (INTERNATIONAL TELECOMMUNICATION UNION, 1999). Nela, os valores alvo na média são⁹: 3 segundos para conexões locais (p. ex., LANs), 5 segundos para chamadas dentro do mesmo país e 8 segundos para chamadas internacionais. Os respectivos valores alvo 95% são 6, 8 e 11 segundos.

Na comparação realizada, a maioria dos valores que constam na tabela estão bem abaixo dos valores alvo recomendados pelo ITU-T. E mesmo os valores mais altos do EECMO, quando considerados em chamadas internacionais, ainda ficam razoavelmente abaixo do valor alvo correspondente.

Em uma análise subjetiva, sob a perspectiva comportamental humana, a espera de um segundo a mais do que se está acostumado (para o caso de uma chamada internacional no caso de uso mais complexo do EECMO), não é algo tão impactante. E isto no ponto de vista do usuário chamador. O usuário chamado tem uma espera ligeiramente menor, já que, do ponto de vista dele, a chamada só iniciou a partir do momento em que ele foi notificado pelo *user agent* dele, através de, por exemplo, um som de telefone tocando. Neste momento, já se passou δ_1 e parte do PDD.

4.2.3.2 Sobrecarga na mensagem SIP

Para avaliar a sobrecarga (*overhead*) causada nas mensagens SIP, primeiro foram coletados os tamanhos de cada “key-mgmt-attribute” do EECMO que transitou entre os *user agents* durante o teste funcional utilizando o *softphone* Endurance. Os tamanhos dos “key-mgmt-attribute” variaram em função do parâmetro de segurança (em *bits* – pois dele derivam-se os demais parâmetros de curva e as próprias curvas, cujo tamanho do corpo base varia) e do fato dos *user agents* estarem ou não sob a mesma TA, já que, no caso de não estarem, há a necessidade de transmitir também o valor de Q_{pub} , ainda que somente na duas primeiras transições¹⁰.

⁹Levando-se em conta uma rede telefônica em condições normais de tráfego.

¹⁰O tipo de emparelhamento não influencia no tamanho das mensagens SIP, uma vez que, no EECMO, os cálculos envolvendo emparelhamentos são utilizados apenas para verificar valores e não para gerar valores que transitam entre os *user agents*.

A Tabela 5 mostra o tamanho máximo que cada “key-mgmt-attribute” pode assumir. Diz-se tamanho máximo, pois, como o tamanho do componente h da assinatura BLMQ varia (devido a ele, na prática, ser um inteiro longo e devido também às particularidades do cálculo dele), os tamanhos dos “key-mgmt-attribute” podem ser ainda menores do que os valores mostrados na tabela.

A Tabela 6 mostra a sobrecarga causada nas mensagens SIP de “INVITE”, “200 OK” e “ACK”, cujos tamanhos típicos (em *bytes*) são aqueles mostrados na coluna “Sem Segurança” da tabela. As demais colunas são formadas por valores que correspondem a soma de cada valor da Tabela 5 e o tamanho (fixo) da mensagem SIP correspondente, ainda acrescentando-se mais três *bytes* por conta do atributo SDP “a=” e do retorno de linha (CRLF)¹¹.

A Tabela 7 permite ter uma melhor noção da sobrecarga ao apresentar uma visão percentual. É importante ressaltar que essa visão percentual é específica do teste funcional realizado, isto é, a sobrecarga causada nas mensagens SIP não varia em função dos tamanhos típicos das mensagens, e sim em função do parâmetro de segurança e da forma de estabelecimento da TA, conforme já explicado. Em outras palavras, qualquer que seja o tamanho de uma mensagem SIP, a sobrecarga causada nela, devido ao uso do EECMO, é dada pelos valores – fixos – apresentados na Tabela 5.

Os tamanhos mostrados na Tabela 6 ficaram bem abaixo dos 1300 *bytes*, que, no caso do SIP, corresponde ao tamanho máximo que um datagrama UDP pode ter para transportar uma mensagem SIP sem que ocorra fragmentação. A fragmentação seria um problema não só porque aumentaria o tempo processamento mas também por conta da não garantia de entrega dos pacotes, no caso do UDP. É importante esclarecer que a limitação do segmento UDP em 1300 *bytes* é mais uma conveniência, de modo que a mensagem SIP fique totalmente contida dentro da MTU, a qual, no caso

¹¹ Para facilitar a visualização da sobrecarga gerada, não levou-se em conta os tamanhos dos cabeçalhos do pacote IP, do pacote UDP e do quadro da camada de enlace. Os tamanhos desses cabeçalhos, quando somados, totalizam algo em torno de 56 bytes.

de Ethernet, corresponde a 1500 *bytes*. Para *wireless local area network* (WLAN), os valores de MTU são maiores ainda.

Assim, a sobrecarga causada pelo EECMO nas mensagens SIP não é suficiente para causar impacto significativo na transmissão dos dados pela rede.

Esta constatação é reforçada ainda pelo experimento relatado em (SCHEINER, 2005), onde foi avaliada a quantidade de banda necessária para que o SIP funcione adequadamente sem impactar o PDD e o PPD. Como os tamanhos das mensagens SIP utilizadas no experimento são bem próximos dos tamanhos das mensagens SIP sobrecarregadas pelo EECMO, pode-se reaproveitar o estudo feito em (SCHEINER, 2005, p. 112 - 114) e as respectivas conclusões tiradas. Estas indicam que, com uma conexão do tipo *Assymmetric Digital Subscriber Line* (ADSL), SIP comporta-se como se estivesse em uma rede local.

• • •

Neste capítulo foi apresentado como o EECMO pode ser acoplado ao SIP de forma padronizada. A sobrecarga gerada nas mensagens SIP de “INVITE”, “200 OK” e “ACK” não causa impacto significativo a ponto de, por exemplo, fazer com que seja necessário mais de um datagrama UDP para transportá-las. Quanto ao tempo de processamento, ainda que em uma comparação direta entre sinalizações com e sem segurança a diferença de tempo pareça demasiadamente grande, ela ainda está num patamar aceitável tanto para os padrões internacionais de telecomunicações quanto para o comportamental humano. Levando-se em conta a segurança provida pelo novo mecanismo (a qual também foi discutida nesse capítulo), o SIP melhorado pelo EECMO oferece um considerável custo-benefício.

Tabela 2: Tempos de processamento obtidos no teste funcional da prova de conceito (valores expressos em milissegundos)

Operação	256 Eta		256 Ate		272 Eta		272 Ate		313 Eta		313 Ate	
	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.
INVITE												
EECMO												
Setup (cliente)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.50	0.00	0.00
Obt. privateKey (x_A)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.50	0.00
Start/embedded msg.	1.00	0.50	2.00	2.50	9.00	9.00	1.00	0.50	4.50	1.00	1.00	11.00
Obt. M_A	33.50	34.00	33.00	33.00	35.00	35.00	34.50	42.50	55.00	55.00	55.50	46.50
Sign $M_A - \text{BLMQ}$	44.00	43.50	35.00	35.50	55.00	47.00	55.00	48.00	59.50	67.50	60.00	67.50
Prep. key-mgmt-attr. 1	0.50	0.00	0.50	0.50	0.50	0.00	0.50	0.00	0.00	0.00	0.50	0.50
Prep. cabeçalho SIP	0.00	0.00	0.00	0.00	0.50	0.50	0.00	0.00	0.00	0.00	0.50	0.50
Prep. mensagem SDP	2.50	3.00	3.50	1.50	2.00	1.50	2.00	1.50	1.50	1.50	1.50	4.00
Constr. do corpo SIP	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Conexão e envio INVITE	8.50	0.50	1.50	1.00	0.50	8.50	1.00	0.50	9.50	0.50	1.00	8.50
Tempo total sem EECMO	11.00	3.50	5.00	2.50	3.00	10.50	3.00	2.00	10.50	2.00	3.00	13.00
Tempo total com EECMO	90.00	81.50	75.50	74.00	102.50	101.50	93.50	93.00	130.50	126.50	120.50	138.50
200 OK												
EECMO												
Parse key-mgmt-attr. 1	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.50	0.00
Verify $M_A - \text{BLMQ}$	170.00	139.50	160.00	139.00	215.50	183.50	195.50	176.50	254.00	213.50	236.00	204.50
Setup (servidor)	0.00	139.50	0.00	99.50	0.00	160.00	0.50	156.50	0.50	184.50	0.00	211.00
Obt. privateKey (x_B)	0.00	0.50	0.00	0.50	0.00	0.00	0.50	0.00	0.50	0.00	0.00	0.00
Obt. M_{BA}	18.00	25.00	23.50	25.00	27.50	20.00	27.00	28.50	39.00	43.50	39.50	40.00
Sign $M_{BA} - \text{BLMQ}$	28.00	32.00	28.00	29.50	38.50	39.00	40.00	38.50	46.00	52.00	54.00	60.50
Prep. key-mgmt-attr. 2	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.50	0.00	0.00	0.50	3.00
Prep. cabeçalho SIP	0.00	0.50	0.00	0.50	0.00	0.00	0.00	0.00	0.50	0.50	0.50	0.00
Prep. mensagem SDP	1.50	1.00	9.00	0.50	8.50	1.00	8.00	1.00	8.50	10.00	8.50	9.00
Constr. do corpo SIP	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Conexão e envio 200 OK	5.50	1.50	1.00	1.00	1.00	0.50	1.00	1.00	10.50	1.00	1.50	0.50
Tempo total sem EECMO	6.00	3.00	10.00	2.00	9.50	1.50	9.00	2.00	19.50	11.50	10.50	9.50
Tempo total com EECMO	223.00	339.50	221.50	295.50	291.00	404.50	272.50	402.50	360.00	505.00	341.00	528.50

(continua)

(conclusão)

Operação	256 Eta		256 Ate		256 Ate		272 Eta		272 Ate		272 Ate		313 Eta		313 Eta		313 Ate		313 Ate	
	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.
ACK																				
EECMO																				
Parse key-mgmt-attr. 2																				
Verify M_{BA} – BLMQ	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Obt. M_B	177.00	183.50	161.50	173.50	197.00	217.00	197.00	217.00	195.00	206.00	274.50	274.50	274.50	274.50	246.50	270.00	246.50	270.00	246.50	270.00
Sign M_B – BLMQ	17.50	25.00	19.00	25.50	27.50	29.00	27.50	29.00	29.00	27.00	39.50	39.50	39.50	40.50	39.00	32.00	39.00	32.00	39.00	32.00
Prep. key-mgmt-attr. 3	28.50	29.00	28.50	27.50	40.50	45.00	40.50	45.00	41.50	40.00	45.50	45.50	45.50	52.50	53.50	54.50	53.50	54.50	53.50	54.50
Prep. cabeçalho SIP	0.50	0.50	0.50	0.50	0.00	3.50	0.00	3.50	0.00	0.00	0.50	0.50	0.50	0.00	0.00	0.50	0.00	0.50	0.00	0.50
Prep. mensagem SDP	0.00	0.50	0.00	0.00	0.50	0.00	0.50	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.50	0.50	0.00	0.50	0.00	0.50
Constr. do corpo SIP	1.00	0.50	15.00	1.00	0.50	1.00	0.50	1.00	1.00	1.00	4.00	4.00	4.00	0.50	1.00	1.00	4.00	1.00	1.00	1.00
Conexão e envio ACK	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Tempo total sem EECMO	9.50	3.00	3.00	4.50	2.00	3.00	2.00	3.00	2.50	1.50	12.50	12.50	12.50	2.00	1.50	2.00	1.50	2.00	1.50	2.00
Tempo total com EECMO	10.50	4.00	18.00	5.50	3.00	4.00	3.00	4.00	3.50	3.00	16.50	16.50	16.50	2.50	3.00	3.50	3.00	3.50	3.00	3.50
Tempo total com EECMO	234.00	242.00	227.50	232.50	268.00	298.50	268.00	298.50	269.00	276.00	376.50	376.50	376.50	370.00	342.00	360.50	342.00	360.50	342.00	360.50
Fechamento																				
EECMO																				
Parse key-mgmt-attr. 3																				
Verify M_B – BLMQ	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Obt. M (ponto)	138.00	130.00	168.00	139.50	170.50	172.50	170.50	172.50	166.00	166.50	226.00	226.00	226.00	238.50	217.50	230.00	217.50	230.00	217.50	230.00
Obt. msg. em claro	32.00	30.50	17.50	26.00	29.00	27.00	29.00	27.00	30.50	28.00	56.50	56.50	56.50	62.00	49.50	58.50	49.50	58.50	49.50	58.50
Tempo total com EECMO	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	8.50	0.50	8.50	0.50	8.50	0.50
Tempo total com EECMO	170.00	161.00	185.50	165.50	199.50	199.50	199.50	199.50	196.50	194.50	282.50	282.50	282.50	300.50	275.50	289.00	275.50	289.00	275.50	289.00
TOTAIS																				
Tempo total gasto no UAC	324.00	323.50	303.00	306.50	370.50	400.00	370.50	400.00	362.50	369.00	507.00	507.00	507.00	496.50	462.50	499.00	462.50	499.00	462.50	499.00
Tempo total gasto no UAS	393.00	500.50	407.00	461.00	490.50	604.00	490.50	604.00	469.00	597.00	642.50	642.50	642.50	805.50	616.50	817.50	616.50	817.50	616.50	817.50

Tabela 3: Tempo médio de processamento despendido com as operações exclusivamente EECMO (valores expressos em milissegundos)

Operação	256 Eta		256 Ate		272 Eta		272 Ate		313 Eta		313 Ate	
	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.	Mesma TA	TA Dif.
PASSO 1												
Setup (cliente)	0.00	0.00	0.00	0.15	0.00	0.00	0.00	0.16	0.16	0.00	0.15	0.15
Obt. prvKey (x_A)	0.00	0.00	0.00	0.15	0.00	0.00	0.00	0.16	0.16	0.00	0.15	0.15
Start/embedded msg.	4.51	3.46	2.34	2.48	2.34	2.19	7.04	2.66	6.52	4.82	12.51	12.51
Obt. M_A	21.23	22.42	22.53	22.08	27.74	29.08	27.18	28.41	38.44	39.07	38.12	40.45
Sign $M_A - \text{BLMQ}$	30.03	30.27	28.75	31.09	37.21	39.07	37.81	41.58	50.90	53.37	51.40	52.36
Prep. key-mgmt-attr. 1	0.16	0.48	0.32	0.16	0.15	0.16	0.16	0.00	0.47	0.30	0.31	0.16
Total	55.93	56.63	53.94	56.11	67.44	70.50	72.19	72.97	96.65	100.21	94.65	105.78
PASSO 2												
Setup (servidor)	0.15	104.27	0.15	98.17	0.00	136.91	0.15	122.63	0.00	179.25	0.15	163.13
Obt. prvKey (x_B)	0.15	0.00	0.15	0.16	0.00	0.47	0.15	0.00	0.00	0.16	0.15	0.15
Parse key-mgmt-attr. 1	0.00	0.00	0.16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Verify $M_A - \text{BLMQ}$	133.27	133.78	126.86	126.71	173.15	173.09	157.52	156.89	220.96	220.61	204.11	204.23
Obt. M_{BA}	23.47	21.69	21.38	21.55	29.86	29.68	29.05	28.89	40.48	40.93	40.75	40.54
Sign $M_{BA} - \text{BLMQ}$	29.18	29.47	29.72	30.12	37.66	37.36	37.82	37.03	51.69	50.13	50.29	51.31
Prep. key-mgmt-attr. 2	0.32	0.32	0.32	0.00	0.00	0.32	0.16	0.16	0.16	0.31	0.15	0.32
Total	186.54	289.53	178.74	276.71	240.67	377.83	224.85	345.60	313.29	491.39	295.60	459.68
PASSO 3												
Parse key-mgmt-attr. 2	0.00	0.16	0.00	0.00	0.15	0.16	0.00	0.00	0.00	0.00	0.00	0.00
Verify $M_{BA} - \text{BLMQ}$	135.94	145.60	127.50	135.96	178.09	188.58	161.71	172.50	231.72	247.51	216.60	229.46
Obt. M_B	22.82	21.54	22.81	21.86	29.11	28.73	28.90	29.02	39.85	38.61	39.33	39.46
Sign $M_B - \text{BLMQ}$	29.70	29.34	28.93	30.05	37.39	36.89	37.84	36.42	51.43	52.52	51.94	52.25
Prep. key-mgmt-attr. 3	0.62	0.16	0.61	0.00	0.31	0.00	0.32	0.45	0.46	0.31	0.00	0.31
Total	189.08	196.80	179.85	187.87	245.05	254.36	228.77	238.39	323.46	338.95	307.87	321.48
FECHAMENTO												
Parse key-mgmt-attr. 3	0.00	0.00	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Verify $M_B - \text{BLMQ}$	133.72	134.89	126.40	125.30	172.93	174.24	157.63	157.40	221.38	220.48	204.54	203.85
Obt. M (ponto)	22.38	21.15	21.68	21.90	29.06	29.03	29.21	28.73	39.12	39.24	39.67	37.66
Obt. msg. em claro	0.31	0.32	0.32	0.46	0.31	0.16	0.00	0.15	0.32	0.32	0.16	0.16
Total	156.41	156.71	148.40	147.81	202.30	203.43	186.84	186.28	260.82	260.04	244.37	241.67
TOTAL GERAL	587.96	699.67	560.93	668.50	755.46	906.12	712.65	843.24	994.22	1190.59	942.49	1128.61

Tabela 4: *Call Set-up Delay* estimado para o SIP utilizando o EECMO em situações diferentes (valores expressos em milissegundos)

	$\Sigma \hat{o}_i$	Cenário com um proxy					Cenário "Trapezoide SIP"				
		LAN	CL	CNI	CIIA	CIIIE	LAN	CL	CNI	CIIA	CIIIE
256 bits – Eta – mesma TA	587.96	617.96	887.96	1197.96	1387.96	2077.96	607.96	997.96	1367.96	1857.96	3317.96
256 bits – Eta – TAs diferentes	699.67	729.67	999.67	1309.67	1499.67	2189.67	719.67	1109.67	1479.67	1969.67	3429.67
256 bits – Ate – mesma TA	560.93	590.93	860.93	1170.93	1360.93	2050.93	580.93	970.93	1340.93	1830.93	3290.93
256 bits – Ate – TAs diferentes	668.50	698.50	968.50	1278.50	1468.50	2158.50	688.50	1078.50	1448.50	1938.50	3398.50
272 bits – Eta – mesma TA	755.46	785.46	1055.46	1365.46	1555.46	2245.46	775.46	1165.46	1535.46	2025.46	3485.46
272 bits – Eta – TAs diferentes	906.12	936.12	1206.12	1516.12	1706.12	2396.12	926.12	1316.12	1686.12	2176.12	3636.12
272 bits – Ate – mesma TA	712.65	742.65	1012.65	1322.65	1512.65	2202.65	732.65	1122.65	1492.65	1982.65	3442.65
272 bits – Ate – TAs diferentes	843.24	873.24	1143.24	1453.24	1643.24	2333.24	863.24	1253.24	1623.24	2113.24	3573.24
313 bits – Eta – mesma TA	944.22	974.22	1244.22	1554.22	1744.22	2434.22	964.22	1354.22	1724.22	2214.22	3674.22
313 bits – Eta – TAs diferentes	1190.59	1220.59	1490.59	1800.59	1990.59	2680.59	1210.59	1600.59	1970.59	2460.59	3920.59
313 bits – Ate – mesma TA	942.49	972.49	1242.49	1552.49	1742.49	2432.49	962.49	1352.49	1722.49	2212.49	3672.49
313 bits – Ate – TAs diferentes	1128.61	1158.61	1428.61	1738.61	1928.61	2618.61	1148.61	1538.61	1908.61	2398.61	3858.61

Tabela 5: Tamanho máximo dos “key-mgmt-attribute” que transitam entre os *user agents* (valores expressos em *bytes*)

	256 bits Mesma TA	256 bits TAs Difer.	272 bits Mesma TA	272 bits TAs Difer.	313 bits Mesma TA	313 bits TAs Difer.
key-mgmt-attribute 1	163	251	171	263	195	303
key-mgmt-attribute 2	163	251	171	263	195	303
key-mgmt-attribute 3	163	163	171	171	195	195

Tabela 6: Sobrecarga causada nas mensagens SIP por conta do acoplamento do EECMO (valores expressos em *bytes*)

	Sem segurança	256 bits Mesma TA	256 bits TAs Difer.	272 bits Mesma TA	272 bits TAs Difer.	313 bits Mesma TA	313 bits TAs Difer.
INVITE	497	663	751	671	763	695	803
200 OK	519	685	773	693	785	717	825
ACK	323	489	489	497	497	521	521

Tabela 7: Sobrecarga causada nas mensagens SIP por conta do acoplamento do EECMO – visão percentual específica da prova de conceito realizada

	256 bits Mesma TA	256 bits TAs Difer.	272 bits Mesma TA	272 bits TAs Difer.	313 bits Mesma TA	313 bits TAs Difer.
INVITE	33%	51%	35%	54%	40%	62%
200 OK	32%	49%	34%	51%	38%	59%
ACK	51%	51%	54%	54%	61%	61%

5 CONCLUSÃO

Nesta dissertação foi apresentada a proposta de um mecanismo de segurança alternativo para o SIP, o qual baseia-se em uma evolução do protocolo Massey-Omura tradicional. Esta evolução, denominada EECMO, utiliza um esquema de assinatura baseada em identidade (viabilizada por emparelhamentos bilineares) para resolver, sob determinadas condições e premissas, as vulnerabilidades do protocolo original. O presente trabalho mostrou ainda que o uso de emparelhamentos é algo viável dentro do contexto SIP, não só teoricamente mas também em termos práticos, desde que o emprego deles seja feito convenientemente.

A semelhança entre a dinâmica de transições característica dos protocolos de três passos (como é o caso do Massey-Omura) e o típico processo de sinalização SIP propiciou o acoplamento natural dos protocolos. Tão natural que não foi necessário alterar o comportamento de um ou de outro, mesmo o Massey-Omura recebendo uma “camada” adicional de segurança para capacitá-lo com dois serviços básicos de segurança que ele não dispunha: autenticidade e integridade. Isto permitiu criar um novo esquema de negociação de chaves que aproveitasse o processo de sinalização SIP para estabelecer uma chave secreta comum entre as partes de modo a proteger a sessão de mídia, que é o que realmente importa ao usuário final de VoIP. É importante ressaltar que esse acoplamento foi feito seguindo recomendações e padrões aceitos mundialmente, de modo a produzir um mecanismo que oferecesse interoperabilidade.

Além de permitir a negociação de chaves, outros benefícios trazidos pela interde-

pendência entre as partes comunicantes, associada ao esquema de assinatura baseada em identidade, são: a detecção de intrusos na comunicação ainda no processo de sinalização e a possibilidade de distinguir se o que um UAS recebeu, no final da sinalização, é uma mensagem válida (ainda que não seja inteligível – p. ex., uma chave secreta de sessão em hexadecimal) ou é qualquer mensagem sem sentido, em uma tentativa de ataque do tipo DoS. Em ambos os casos, evitar-se-ia o estabelecimento de uma sessão de mídia para, somente nela, descobrir que o valor recebido não serve para nada, pois não representa chave secreta alguma.

Outros tipos de protocolos e esquemas de negociação de chaves com menos transições não oferecem tais benefícios já que, normalmente, a estratégia deles ou é transmitir a chave em uma transição única (susceptível a várias formas de ataque *man-in-the-middle*, incluindo as que simplesmente retiram os elementos que formam o mecanismo de segurança utilizado) ou é gerar o valor final da chave a partir de cálculos matemáticos realizados ao término da segunda transição. Em outras palavras, a terceira transição acaba sendo uma forma de garantir o esquema todo. Tanto que a estratégia da terceira transição é adotada mesmo em protocolos que não são categorizados como de três passos. É o caso do esquema proposto em (LUO; WEN; ZHAO, 2008), apresentado na subseção B.2.1 do Apêndice B.

Embora o esquema proposto necessite de uma entidade externa, a TA, esta é bem simples e pode ser colocada em operação sem grandes esforços, inclusive no que diz respeito à segurança. E não há processos complexos atrelados à existência da TA. Ainda com relação às TAs, é sempre importante ressaltar a flexibilidade do mecanismo proposto (especificamente no contexto SIP), o qual funcionaria adequadamente e com bom desempenho, seja com uma TA centralizada ou uma Federação de TAs.

A viabilidade tanto do mecanismo proposto quanto do emprego de emparelhamentos bilineares no contexto SIP foi comprovada através de uma prova de conceito realizada com um *softphone* construído seguindo normas e padrões de aceitação mundial.

A partir da prova de conceito e do refinamento dos valores obtidos nela, foi possível avaliar o desempenho do mecanismo proposto. Quanto ao tempo de processamento, o uso de emparelhamentos, de fato, aumentou consideravelmente o tempo de sinalização. Entretanto, mesmo no caso de uso considerado mais complexo (313 *bits* com TAs federadas) e no cenário de comunicação cujo tempo de resposta tende a ser o mais longo (chamadas internacionais intercontinentais), os tempos de processamento ficaram razoavelmente abaixo dos valores limites recomendados. Além disso, em uma avaliação subjetiva, o emprego de emparelhamentos não geraria desconforto ou outras reações comportamentais indesejáveis, já que acrescentou-se, no máximo, algo em torno de um segundo de espera. Quanto à sobrecarga gerada nas mensagens SIP, ela não foi grande o suficiente a ponto de provocar divisões de datagramas. Ao contrário, a sobrecarga gerada ainda deixa um espaço razoável nos mesmos. E, também, verificou-se que a sobrecarga causada nas mensagens não foi suficiente para impactar o tempo de transmissão dos pacotes pela rede.

No que diz respeito a trabalhos futuros, eles poderiam tanto explorar variações na forma de construção do EECMO quanto na aplicabilidade dele ainda no contexto SIP.

Quanto à construção, outras famílias de curvas (amigáveis a emparelhamentos) poderiam ser testadas e o desempenho novamente avaliado. Também poderiam ser avaliados outros algoritmos para cálculo de emparelhamentos, ainda que nos testes realizados não houve diferença significativa entre os emparelhamentos Eta e Ate. Ou mesmo avaliar o comportamento e o desempenho de outros níveis de segurança, além de 313 *bits*.

Em termos de aplicabilidade ainda para SIP, como a dinâmica de transições é semelhante a uma dinâmica do tipo desafio-resposta, o EECMO poderia ser utilizado no processo de autenticação do *user agent* junto a um servidor *proxy* SIP. Em outras palavras, o EECMO poderia ser adaptado de tal maneira que fosse uma alternativa também para o mecanismo de HTTP Digest Authentication nativo do SIP. Uma expansão

do EECMO no contexto SIP poderia ser tentada também, no intuito de evitar ataques DoS provocados por requisições SIP do tipo “BYE” e “CANCEL”. Nesta modalidade de ataque, específica do SIP, atacantes enviam tais tipos de requisição que fazem com que a sessão que estiver estabelecida seja encerrada imediatamente. Lembrando que o SIP não somente cria mas também controla e encerra sessões de multimídia estabelecidas entre dois ou mais participantes. Neste caso, quando uma parte recebesse um “BYE” ou um “CANCEL” bastaria utilizar o algoritmo “Verify” do BLMQ para autenticar a origem da requisição SIP. A mensagem assinada poderia ser qualquer uma, inclusive a própria identidade de quem enviou a mensagem.

Por fim, dois trabalhos futuros relevantes poderiam ser realizados. Um deles seria experimentar o conjunto SIP – EECMO em redes SIP P2P, uma vez que o mecanismo de segurança proposto independe da existência de servidores SIP. Ao contrário, ele foi concebido no intuito de ser fim-a-fim, sem intermediários. O outro trabalho proposto seria repetir o experimento descrito em (SCHEINER, 2005), mas já contemplando o EECMO e em um ambiente mais próximo ainda da realidade, como o Planet Lab¹.

¹<http://www.planet-lab.org/>

REFERÊNCIAS

- AL-RIYAMI, S. S.; PATERSON, K. G. *Certificateless Public Key Cryptography*. 2003. Cryptology ePrint Archive, Report 2003/126. Disponível em "<http://eprint.iacr.org/2003/126>". Acesso em: 2 de maio de 2010.
- ANDREASEN, F.; BAUGHER, M.; WING, D. *Session Description Protocol (SDP) Security Descriptions for Media Streams*. [S.l.], Jul 2006. Disponível em "<http://www.faqs.org/rfcs/rfc4568.html>". Acesso em: 2 de maio de 2010.
- ARKKO, J. et al. *RFC 3830 - MIKEY: multimedia Internet keying*. [S.l.], Ago 2004. Disponível em "<http://www.faqs.org/rfcs/rfc3830.html>". Acesso em: 2 de maio de 2010.
- _____. *Key management extensions for session description protocol (SDP) and real time streaming protocol (RTSP)*. [S.l.], Jul 2006. Disponível em "<http://www.faqs.org/rfcs/rfc4567.html>". Acesso em: 2 de maio de 2010.
- BARBOSA, M.; FARSHIM, P. Certificateless signcryption. In: *ACM Symposium on Information, Computer and Communications Security - ASIACCS'2008*. Tóquio, Japão: Springer, 2008. (Lecture Notes in Computer Science).
- BARRETO, A.; FALEIROS, A. *MIKEY DHHMAC-SAS: the new MIKEY transportation mode*. [S.l.], Jun 2007. Proposta feita ao IETF - 'draft-barreto-ietf-dhhmac-sas-00.txt' - já expirado. *Draft* original disponível em "<http://ietfreport.isoc.org/all-ids/draft-barreto-ietf-dhhmac-sas-00.txt>". Acesso em: 2 de maio de 2010.
- BARRETO, P. S. L. M. et al. Toward efficient certificateless signcryption from (and without) bilinear pairings. In: *VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSeg'2008*. Gramado/RS, Brasil: [s.n.], 2008. p. 115 – 125.
- _____. *Efficient Pairing Computation on Supersingular Abelian Varieties*. 2004. Cryptology ePrint Archive, Report 2004/375. Disponível em "<http://eprint.iacr.org/2004/375>". Acesso em: 2 de maio de 2010.
- _____. Efficient algorithms for pairing-based cryptosystems. In: *Advances in Cryptology - CRYPTO 2002*. [S.l.]: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2442), p. 354 – 368.
- _____. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In: *Advanced in Cryptology - Asiacrypt'2005*. [S.l.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3788), p. 515–532.

BAUGHER, M. et al. *RFC 3711 - the secure real-time transport protocol (SRTP)*. [S.l.], Mar 2004. Disponível em "<http://www.faqs.org/rfcs/rfc3711.html>". Acesso em: 2 de maio de 2010.

BERNERS-LEE, T.; FIELDING, R.; MASINTER, L. *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. [S.l.], Jan 2005. Disponível em "<http://www.faqs.org/rfcs/rfc3986.html>". Acesso em: 2 de maio de 2010.

BLAKE, I. F.; SEROUSSI, G.; SMART, N. P. *Advances in Elliptic Curve Cryptography*. 1st. ed. [S.l.]: Cambridge University Press, 2005. London Mathematical Society Lecture Note Series Collection.

BONEH, D.; FRANKLIN, M. Identity-based encryption from the weil pairing. In: . [S.l.]: Springer-Verlag, 2001. (Lecture Notes in Computer Science, v. 2139), p. 213 – 229.

BONEH, D.; LYNN, B.; SHACHAM, H. Short signatures from the weil pairing. In: *Advances in Cryptology - Asiacrypt'2001*. [S.l.]: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2248), p. 514 – 532.

_____. Short signatures from the weil pairing. *Journal of Cryptology*, Springer New York, v. 17, n. 4, p. 297 – 319, September 2004.

BURNETT, A.; DUFFY, A.; DOWLING, T. *A Biometric Identity Based Signature Scheme*. 2004. Cryptology ePrint Archive, Report 2004/176. Disponível em "<http://eprint.iacr.org/2004/176>". Acesso em: 2 de maio de 2010.

BUTCHER, D.; LI, X.; GUO, J. Security challenge and defense in voip infrastructures. *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*, v. 37, n. 6, p. 1152 – 1162, 2007.

CAO, F.; MALIK, S. Security analysis and solutions for deploying ip telephony in the critical infrastructure. *Workshop of the 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks*, p. 171 – 180, 2005.

CHA, J. C.; CHEON, J. H. An identity-based signature from gap diffie-hellman groups. In: *Public Key Cryptography - PKC 2003*. [S.l.]: Springer Berlin / Heidelberg, 2003. (Lecture Notes in Computer Science, v. 2567), p. 18 – 30. Disponível em "<http://citeseer.ist.psu.edu/cha02identitybased.html>". Acesso em: 2 de maio de 2010.

CHEN, L.; KUDLA, C. *Identity Based Authenticated Key Agreement Protocols from Pairings*. 2002. Cryptology ePrint Archive, Report 2002/184. Disponível em "<http://eprint.iacr.org/2002/184>". Acesso em: 2 de maio de 2010.

CURCIO, I. D. D.; LUNDAN, M. Sip call setup delay in 3g networks. In: *Seventh International Symposium on Computers and Communications (ISCC'02)*. Washington, DC, USA: IEEE Computer Society, 2002. p. 835–840.

DENT, A. W. *A Survey of Certificateless Encryption Schemes and Security Models*. 2006. Cryptology ePrint Archive, Report 2006/211. Disponível em "<http://eprint.iacr.org/2006/211>". Acesso em: 2 de maio de 2010.

DEUSAJUTE, A. M.; BARRETO, P. S. L. M. Authenticated massey-omura encryption with minimal effort from bilinear pairings. Artigo submetido ao SBSeg - Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - 2007. 2007.

_____. The sip security enhanced by using pairing-assisted massey-omura signcryption. In: *Actas de la X Reunión Española sobre Criptología y Seguridad de la Información - RECSI*. Salamanca, Espanha: [s.n.], 2008. p. 51 – 62.

DIFFIE, W.; HELLMAN, M. E. New directions in cryptography. *IEEE Transactions on Information Theory*, v. 22, n. 06, p. 644 – 654, 1976.

DOUSKALIS, B. *IP Telephony - The Integration of Robust VoIP Services*. 1. ed. New Jersey, USA: Prentice Hall PTR, 2000.

DUURSMA, I.; LEE, H. Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$. In: *Advances in Cryptology - ASIACRYPT 2003*. [S.l.]: Springer Berlin / Heidelberg, 2003. (Lecture Notes in Computer Science, v. 2894), p. 111 – 123.

DWIVEDI, H. *Hacking VoIP - Protocols, Attacks and Countermeasures*. San Francisco, CA, USA: No Starch Press, 2009.

ELETRICNEWS.NET. *Mobile and VoIP to inherit the earth*. 2005. Disponível em "http://www.theregister.co.uk/2005/06/27/rising_mobile_voip_revenues/". Acesso em: 2 de maio de 2010.

EUCHNER, M. *RFC 4650 - HMAC-authenticated Diffie-Hellman for multimedia Internet keying (MIKEY)*. [S.l.], Set 2006. Disponível em "<http://www.faqs.org/rfcs/rfc4650.html>". Acesso em: 2 de maio de 2010.

FREY, G.; MÜLLER, M.; RÜCK, H.-G. The tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory*, v. 45, n. 5, p. 1717 – 1719, Jul 1999.

GAMMA, E. et al. *Design Patterns: elements of Reusable Object-Oriented Software*. [S.l.]: Addison-Wesley Professional, 1994.

GEORGIA TECH INFORMATION SECURITY CENTER. *Emerging Cyber Threats Report for 2009*. [S.l.], Out. 2008. Disponível em "<http://www.gtiscsecuritysummit.com/pdf/CyberThreatsReport2009.pdf>". Acesso em: 2 de maio de 2010.

HANDLEY, M.; JACOBSON, V. *RFC 2327 - SDP: Session Description Protocol*. [S.l.], Abr. 1998. Disponível em "<http://www.faqs.org/rfcs/rfc2327.html>". Acesso em 2 de maio de 2010.

HANDLEY, M.; JACOBSON, V.; PERKINS, C. *RFC 4566 - SDP: session description protocol*. [S.l.], Jul. 2006. Disponível em "<http://www.faqs.org/rfcs/rfc4566.html>". Acesso em: 2 de maio de 2010.

HANDLEY, M. et al. *RFC 2543 - SIP: session initiation protocol*. [S.l.], Mar 1999. Disponível em "<http://www.faqs.org/rfcs/rfc2543.html>". Acesso em: 2 de maio de 2010.

HANKERSON, D.; MENEZES, A. J.; VANSTONE, S. *Guide to Elliptic Curve Cryptography*. 1st. ed. [S.l.]: Springer, 2004.

HENDERSON, M. Applications of linearised and sub-linearised polynomials to information security. In: *Information Security and Privacy*. [S.l.]: Springer Berlin / Heidelberg, 1998. (Lecture Notes in Computer Science, v. 1438), p. 227 – 237.

HESS, F. Efficient identity based signature schemes based on pairings. In: *Selected Areas in Cryptography*. [S.l.]: Springer Berlin / Heidelberg, 2003. (Lecture Notes in Computer Science, v. 2595), p. 310 – 324.

HESS, F.; SMART, N. P.; VERCAUTEREN, F. *The Eta Pairing Revisited*. 2006. Cryptology ePrint Archive, Report 2006/110. Disponível em "<http://eprint.iacr.org/2006/110>". Acesso em: 2 de maio de 2010.

IGNJATIC, D. et al. *RFC 4738 - MIKEY-RSA-R: An Additional Mode of Key Distribution in Multimedia Internet KEYing (MIKEY)*. [S.l.], Nov 2006. Disponível em "<http://www.faqs.org/rfcs/rfc4738.html>". Acesso em: 2 de maio de 2010.

INFORMATION TECHNOLOGY LABORATORY OF THE NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. *FIPS PUB 140-2 - Security Requirements for Cryptographic Modules*. [S.l.], Mai. 2001. Disponível em <http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf>. Acesso em: 2 de maio de 2010.

INTERNATIONAL TELECOMMUNICATION UNION. *ITU-T Recommendation E.721 - Network grade of service parameters and target values for circuit-switched services in the evolving ISDN*. [S.l.], Mai. 1999. Disponível em "<http://www.itu.int/rec/T-REC-E.721-199905-I/en>". Acesso em: 2 de maio de 2010.

_____. *ITU-T Recommendation Y.1531 - SIP-based call processing performance*. [S.l.], Nov. 2007. Disponível em "<http://www.itu.int/rec/T-REC-Y.1531-200711-I/en>". Acesso em: 2 de maio de 2010.

JOUX, A. A one round protocol for tripartite diffie-hellman. In: *Algorithmic Number Theory: Fourth International Symposium*. [S.l.]: Springer-Verlag, 2000. (Lecture Notes in Computer Science, v. 1838), p. 385 – 393.

KHALILI, A.; KATZ, J.; ARBAUGH, W. A. Toward secure key distribution in truly ad-hoc networks. In: *SAINT-W '03: Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*. Washington, DC, USA: IEEE Computer Society, 2003. p. 342.

KIM, I.; KIM, K. Secure session management mechanism in voip service. In: *Lecture Notes in Computer Science - Frontiers of High Performance Computing and Networking ISPA 2007 Workshops*. [S.l.]: Springer Berlin / Heidelberg, 2007. v. 4743, p. 96 – 104.

KOBLITZ, N. Elliptic curve cryptosystems. *Math Comp.*, v. 48, n. 177, p. 203 – 209, Jan 1987.

_____. *A Course in Number Theory and Cryptography*. Second. [S.l.]: Springer-Verlag, 1994. Graduate Texts in Mathematics.

KRAWCZYK, H.; RABIN, T. *Chameleon Hashing and Signatures*. 1998. Cryptology ePrint Archive, Report 1998/010. Disponível em "<http://eprint.iacr.org/1998/010>". Acesso em: 2 de maio de 2010.

LEE, C.-H.; HAN, K.-S.; LEE, Y.-H. Efficient resource registration and location scheme in p2p-sip, using id-based signature. In: *10th International Conference on Advanced Communication Technology - ICACT'2008*. Gangwon-Do, Coréia do Sul: [s.n.], 2008. v. 3, p. 1823 – 1827.

LEE, E.; LEE, H.-S.; PARK, C.-M. *Efficient and Generalized Pairing Computation on Abelian Varieties*. 2008. Cryptology ePrint Archive, Report 2008/040. Disponível em "<http://eprint.iacr.org/2008/040>". Acesso em: 2 de maio de 2010.

LENNOX, J. *Connection-Oriented Media Transport over the Transport Layer Security (TLS) Protocol in the Session Description Protocol (SDP)*. [S.l.], Jul 2006. Disponível em "<http://www.faqs.org/rfcs/rfc4572.html>". Acesso em: 2 de maio de 2010.

LOU, D. et al. Personalized service mobility and security in sip-based communications. In: *13th IEEE International Conference on Networks / IEEE 7th Malaysia International Conference on Communication*. [S.l.: s.n.], 2005. v. 1, p. 113 – 117.

LUO, M.; WEN, Y.; ZHAO, H. An enhanced authentication and key agreement mechanism for sip using certificateless public-key cryptography. p. 1577 – 1582, Nov 2008.

MALONE-LEE, J. Signcryption with non-interactive non-repudiation. *Designs, Codes and Cryptography*, Springer, v. 37, n. 1, Out 2005.

MAMBO, M.; OKAMOTO, E.; SAKURAI, K. Protection of data and delegated keys in digital distribution. In: *Information Security and Privacy*. [S.l.]: Springer Berlin / Heidelber, 1997. (Lecture Notes in Computer Science, v. 1270), p. 271 – 282.

MASSEY, J. L.; OMURA, J. K. A new multiplicative algorithm over finite fields and its applicability in public key cryptography. In: *Advances in Cryptology - Eurocrypt'83*. Udine, Italy: [s.n.], 1983.

_____. *United States patent 4567600: method and apparatus for maintaining the privacy of digital messages conveyed by public transmission*. [S.l.], Jan 1986. Disponível em "<http://www.freepatentsonline.com/4567600.html>". Acesso em: 2 de maio de 2010 (para acessar e visualizar as figuras citadas na página web é necessário criar uma conta de usuário, de forma gratuita e com dados mínimos).

MATSUDA, S. et al. Optimised versions of the ate and twisted ate pairings. In: *The 11th IMA International Conference on Cryptography and Coding*. [S.l.]: Springer-Verlag, 2007. (Lecture Notes in Computer Science, v. 4887), p. 302 – 312. Disponível em "<http://eprint.iacr.org/2007/013>". Acesso em: 2 de maio de 2010.

- MCCULLAGH, N.; BARRETO, P. S. L. M. *Identity-Based Signcryption*. 2004. Cryptology ePrint Archive, Report 2004/117. Disponível em "<http://eprint.iacr.org/2004/117>". Acesso em: 2 de maio de 2010.
- MENEZES, A. J. An introduction to pairing-based cryptography. Notas de conferências, Santander, Espanha. Disponível em "<http://www.cacr.math.uwaterloo.ca/~ajmenezes/publications/pairings.pdf>". Acesso em: 2 de maio de 2010.
- MENEZES, A. J.; OKAMOTO, T.; VANSTONE, S. A. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions of Information Theory*, v. 39, n. 5, p. 1639 – 1646, Set 1993.
- MENEZES, A. J.; OORSCHOT, P.; VANSTONE, S. *Handbook of Applied Cryptography*. [S.l.]: CRC Press, 1996. Disponível em "<http://www.cacr.math.uwaterloo.ca/hac/>". Acesso em: 2 de maio de 2010.
- MEYER, B.; PORTMANN, M. Practical performance evaluation of peer-to-peer internet telephony using sip. In: *8th International Conference on Computer and Information Technology Workshops (CIT Workshops 2008)*. Washington, DC, USA: IEEE Computer Society, 2008. p. 204 – 209.
- MILLER, V. S. Use of elliptic curves in cryptography. In: *Advances in Cryptology - CRYPTO '85*. [S.l.]: Springer Berlin / Heidelberg, 1986. (Lecture Notes in Computer Science, v. 218), p. 417 – 426.
- _____. The weil pairing, and its efficient calculation. *Journal of Cryptology*, n. 17, p. 235 – 261, Ago 2004.
- NETWORKS, N. *VoIP Technologies: A Comprehensive Guide to Voice over Internet Protocol (VoIP)*. [S.l.]: Nortel Press, 2008.
- PEREA, R. M. *Internet Multimedia Communications Using SIP - A Modern Approach Including Java Practice*. Burlington, MA, USA: Morgan Kaufmann Publishers, an imprint of Elsevier, Inc., 2008.
- QUINTEN, V. M.; MEENT, R. V.; PRAS, A. Analysis of techniques for protection against spam over internet telephony. In: *Dependable and Adaptable Networks and Services*. [S.l.]: Springer Berlin / Heidelberg, 2007. v. 4606, p. 70 – 77.
- REZENDE, A. R. S. *Análise das vulnerabilidades e ataques ao protocolo SIP*. Uberlândia - MG: [s.n.], 2006. Trabalho de Conclusão de Curso. Disponível em "http://si.uniminhas.br/TFC/monografias/Monografia_SIP.pdf". Acesso em: 2 de maio de 2010.
- RING, J. et al. A new authentication mechanism and key agreement protocol for sip using identity-based cryptography. In: *Asia Pacific Information Technology Security Conference - AusCERT 2006*. Gold Coast, Australia: [s.n.], 2006. Disponível em "<http://eprints.qut.edu.au/archive/00004422/>". Acesso em: 2 de maio de 2010.

RIVEST, R.; SHAMIR, A. How to expose an eavesdropper. *Communications of the ACM*, v. 27, n. 4, p. 393 – 394, 1984.

ROSEMBERG, J. et al. *RFC 3261 - SIP: session initiation protocol*. [S.l.], Jun 2002. (atualização da RFC-2543). Disponível em "<http://www.faqs.org/rfcs/rfc3261.html>". Acesso em: 2 de maio de 2010.

SAKAI, R.; KASAHARA, M. *ID based cryptosystems with pairing on elliptic curve*. 2003. SCIS'03, Hamamatsu, Japão. Disponível em: "<http://eprint.iacr.org/2003/054>". Acesso em: 2 de maio de 2010.

SAKURAI, K.; SHIZUYA, H. A structural comparison of the computational difficulty of breaking discrete log cryptosystems. *Journal of Cryptology*, n. 11, p. 29 – 43, 1998.

SCHEINER, L. N. *Análise de Desempenho do Protocolo SIP na Sinalização de Chamadas de Voz sobre IP*. Dissertação (Mestrado) — Pontifícia Universidade Católica do Rio de Janeiro - PUC-RIO, Ago. 2005. Dissertação de mestrado disponível em "http://www.maxwell.lambda.ele.puc-rio.br/Busca_etds.php?strSecao=resultado&nrSeq=7065@1". Acesso em: 2 de maio de 2010.

SCHNORR, C. P. Efficient signature generation by smart cards. *Journal of Cryptology*, v. 4, n. 3, p. 161 – 174, 1991.

SCHULZRINNE, H. et al. *RFC 3550 - RTP: a transport protocol for real-time applications*. [S.l.], Jul 2003. Disponível em "<http://www.faqs.org/rfcs/rfc3550.html>". Acesso em: 2 de maio de 2010.

SCOTT, M.; COSTIGAN, N.; ABDULWAHAB, W. *Implementing Cryptographic Pairings on Smartcards*. 2006. Cryptology ePrint Archive, Report 2006/144. Disponível em "<http://eprint.iacr.org/2006/144>". Acesso em: 2 de maio de 2010.

SHAMIR, A. Identity-based cryptosystems and signature schemes. In: *Advances in Cryptology - CRYPTO'84*. [S.l.]: Springer Berlin / Heidelberg, 1985. (Lecture Notes in Computer Science, v. 196), p. 47 – 53.

SINNREICH, H.; JOHNSTON, A. B. *Internet communications using SIP - Delivering VoIP and Multimedia Services with Session Initiation Protocol*. 2nd. ed. Indianapolis/IN - USA: Wiley Publishing, Inc., 2006.

STALLINGS, W. *Cryptography and network security*. 4th. ed. [S.l.]: Pearson Prentice-Hall, 2006.

TANENBAUM, A. S. *Redes de Computadores*. 4a.. ed. [S.l.]: Editora Campus, 2003.

TELECO. *Estatísticas do mercado de VOIP*. 2009. Disponível em "http://www.teleco.com.br/voip_estatis.asp". Acesso em: 2 de maio de 2010.

VERCAUTEREN, F. *Optimal Pairings*. 2008. Cryptology ePrint Archive, Report 2008/096. Disponível em "<http://eprint.iacr.org/2008/096>". Acesso em: 2 de maio de 2010.

WALLINGFORD, T. *Switching to VoIP*. [S.l.]: O'Reilly Media, 2005.

WANG, F.; ZHANG, Y. A new provably secure authentication and key agreement mechanism for sip using certificateless public-key cryptography. *Comput. Commun.*, Butterworth-Heinemann, Newton, MA, USA, v. 31, n. 10, p. 2142 – 2149, 2008. Versão completa disponível em "<http://eprint.iacr.org/2007/220>". Acesso em: 2 de maio de 2010.

WASHINGTON, L. C. *Elliptic Curves: number theory and cryptography*. 2. ed. Maryland, USA: Chapman and Hall/CRC - an imprint of the Taylor and Francis Group, 2008.

WILLIAMSON, M. J. *Non-secret encryption using a finite field*. [S.l.], Jan 1974. Disponível em "<http://www.cesg.gov.uk/site/publications/media/secenc.pdf>". Acesso em: 2 de maio de 2010.

WINTON, R. Enhancing the massey-omura cryptosystem. *Journal of Mathematical Sciences and Mathematics education*, Tarleton State University, USA, v. 2, n. 1, Fev 2007. Disponível em "<http://www.msme.us/2007-1-3.pdf>". Acesso em: 2 de maio de 2010.

ZHAO, C.-A.; ZHANG, F.; HUANG, J. *A Note on the Ate Pairing*. 2007. Cryptology ePrint Archive, Report 2007/247. Disponível em "<http://eprint.iacr.org/2004/247>". Acesso em: 2 de maio de 2010.

ZHENG, Y. Digital signcryption or how to achieve cost (signature and encryption) $\ll$ cost(signature) + cost(encryption). In: *Advances in Cryptology - CRYPTO'97*. [S.l.]: Springer-Verlag, 1997. (Lecture Notes in Computer Science, v. 1294), p. 165 – 179.

APÊNDICE A - EMPARELHAMENTOS BILINEARES

Ainda que emparelhamentos bilineares seja um assunto relativamente recente na criptografia¹, muito já se produziu, se discutiu, e se propôs usando esse tema como base, de modo que seria inviável abordar, ainda que resumidamente, todos os aspectos envolvidos com emparelhamentos. Portanto, este apêndice apresenta os fundamentos mínimos necessários para dar suporte teórico ao leitor menos familiarizado com o tema.

A.1 Teoria

Definição 2. *Seja n um número inteiro positivo. Sejam $\mathbb{G}_1$ e $\mathbb{G}_2$ grupos abelianos aditivos com elemento identidade 0 . Seja ainda $\mathbb{G}_T$ um grupo cíclico multiplicativo de ordem n , com identidade multiplicativa 1 . Suponha que $\mathbb{G}_1$ e $\mathbb{G}_2$ têm expoente n (isto é, $nP = 0, \forall P \in \mathbb{G}_1, \mathbb{G}_2$).*

Um emparelhamento bilinear é uma função (mapeamento) da forma $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ satisfazendo as seguintes propriedades básicas:

- *Bilinearidade*²: $\forall P, P' \in \mathbb{G}_1$ e $\forall Q, Q' \in \mathbb{G}_2$,

¹O primeiro trabalho que tratou emparelhamentos como um recurso criptográfico para prover segurança foi (Joux, 2000), enquanto que a proposta de uso de curvas elípticas na criptografia data da década de 1980 (Koblitz, 1987; Miller, 1986), e a origem da própria criptografia assimétrica data da década de 1970 (Diffie; Hellman, 1976).

²Bilinear nesse contexto quer dizer que o mapeamento é linear em cada componente P e Q .

$$e(P + P', Q) = e(P, Q)e(P', Q)$$

$$e(P, Q + Q') = e(P, Q)e(P, Q')$$

- *Não degeneração:* $\forall P \in \mathbb{G}_1$, com $P \neq 0$, existe algum $Q \in G_2$ tal que $e(P, Q) \neq 1$.
1. Analogamente, $\forall Q \in \mathbb{G}_2$, com $Q \neq 0$, existe algum $P \in G_1$ tal que $e(P, Q) \neq 1$.
- *Computabilidade:* existe um algoritmo eficiente para calcular $e(P, Q)$, $\forall P, Q \in \mathbb{G}_1 \times \mathbb{G}_2$.

(BLAKE; SEROUSSI; SMART, 2005).

As propriedades básicas possuem, cada uma, um grau de relevância no contexto de uso de emparelhamentos. A “Bilinearidade” é essencial para se definir as aplicações, qualquer que seja a natureza delas (negociação de chaves, encriptação/decriptação, verificação de assinatura etc.). A “Não degeneração” deve existir, pois não faz sentido utilizar-se, para aplicações criptográficas, emparelhamentos degenerados onde $e(P, Q) = 1$. Quanto à “Computabilidade”, ela é imprescindível, pois, embora algum tipo de emparelhamento intratável possa ser útil em uma análise teórica (p. ex., para provar que existe um processo finito para calcular algo, mesmo que leve tempo exponencial), ele não seria aproveitável em uma área aplicada como a de criptografia. Em outras palavras, quando um emparelhamento bilinear não é implementável computacionalmente, ele acaba não servindo para aplicação alguma na prática.

Além das propriedades básicas, existem outras, também relevantes:

1. $e(P, O) = e(O, Q) = 1$, onde O nesse contexto é o ponto no infinito da curva elíptica associada ao emparelhamento.
2. $e(aP, bP) = e(P, P)^{ab} = e(abP, P) = e(P, abP) \quad \forall a, b \in \mathbb{Z}$.
3. $e(-P, Q) = e(P, Q)^{-1} = e(P, -Q)$.

4. Se $e(P, Q) = 1$, $\forall Q \in \mathbb{G}_1$ então $P = O$, onde O nesse contexto é o ponto no infinito da curva elíptica associada ao emparelhamento.

Um corolário importante derivado das propriedades anteriores é o seguinte:

Corolário 1. Se $e : \mathbb{G}_0 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ é um emparelhamento bilinear, então

$$e(cP, Q) = e(P, cQ) \quad (\text{A.1})$$

para todo $P \in \mathbb{G}_0$, $Q \in \mathbb{G}_1$, e $c \in \mathbb{Z}$.

Prova: especializando a condição de bilinearidade $e(aP, bQ) = e(P, Q)^{ab}$ para $a = c$ e $b = 1$ obtém-se $e(cP, Q) = e(P, Q)^c$, e, analogamente, tomando-se $a = 1$ e $b = c$ obtém-se $e(P, cQ) = e(P, Q)^c$. Então, $e(cP, Q) = e(P, cQ)$. $\square$

A definição matemática em si de emparelhamento bilinear é um tanto genérica, pois não especifica $\mathbb{G}_1$ e $\mathbb{G}_2$. Simplesmente trata-os como grupos abelianos aditivos, isto é, quaisquer grupos abelianos, finitos, $\mathbb{G}_1$ e $\mathbb{G}_2$, com uma operação binária de natureza aditiva definida sobre eles que satisfaçam as propriedades básicas enunciadas anteriormente, podem servir de base para a construção de emparelhamentos bilineares.

Vendo desta forma, pode parecer fácil encontrar grupos abelianos aditivos $\mathbb{G}_1$ e $\mathbb{G}_2$ em condições propícias para se construir emparelhamentos. Mas o fato é que, até o momento, apenas curvas elípticas conseguiram oferecer, inerentemente, tais grupos nas condições especificadas. Isto porque, curvas elípticas, por definição, estão baseadas em corpos e as propriedades algébricas de corpos englobam as propriedades algébricas de grupos abelianos. Por isso é que os emparelhamentos existentes estão definidos sobre curvas elípticas.

Emparelhamentos sobre curvas elípticas mapeiam um par de pontos das curvas envolvidas em um elemento de um grupo multiplicativo associado a um corpo finito. Em outras palavras, é uma função que recebe como entrada um par de valores $(P, Q) \in$

$\mathbb{G}_1 \times \mathbb{G}_2$ (onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são curvas elípticas) e produz como saída um outro valor, definido em $\mathbb{G}_T$.³

Por exemplo: $e : E(\mathbb{F}_q) \times E(\mathbb{F}_{q^k}) \rightarrow F_{q^k}^*$ é um emparelhamento cujo primeiro argumento (ponto) pertence à curva E definida sobre o corpo $\mathbb{F}_q$, o segundo argumento pertence à mesma curva, só que agora definida sobre F_{q^k} , e a saída é um valor pertencente à $F_{q^k}^*$.

O número $q = p^m$ é denominado ordem do corpo $\mathbb{F}$ e corresponde ao número de elementos no corpo. Já o número p é chamado de característica do corpo $\mathbb{F}$, e acaba-se adotando a mesma denominação para identificar a característica da curva. Quando $p = 2$, o corpo $\mathbb{F}$ é chamado de corpo binário. Quando $p \geq 3$ e $m = 1$, $\mathbb{F}$ é chamado de corpo primo. E quando $p \geq 3$ e $m \geq 2$, tem-se a extensão do corpo $\mathbb{F}$.

Um conceito importante relacionado com emparelhamentos é o definido a seguir:

Definição 3. *Seja E uma curva elíptica definida sobre o corpo $\mathbb{F}_q$ e seja P um ponto desta curva, ou seja, $P \in E(\mathbb{F}_q)$, tal que P tem ordem⁴ prima n . Suponha ainda que o Máximo Divisor Comum entre n e q seja igual à 1, isto é, $\text{MDC}(n, q) = 1$. Então, o grau de mergulho – do inglês, embedding degree – de $\langle P \rangle$ é o menor inteiro positivo k tal que $n \mid q^k - 1$.*

Se o grau de mergulho k é pequeno, então há a possibilidade de que algoritmos subexponenciais conhecidos para resolução do PLD em $\mathbb{F}_{q^k}$ sejam mais rápidos que o método Pollard ρ para resolução do PLDCE em $\langle P \rangle$. Esse é o caso de curvas supersingulares desde que $k \in \{1, 2, 3, 4, 6\}$ para estas curvas. Entretanto, é possível esperar que $k \approx n$ para a maioria das curvas elípticas – e foi provado ser esse o caso para curvas elípticas de ordem prima sobre corpos primos (MENEZES, 2005).

³Em algumas situações onde $\mathbb{G}_1 = \mathbb{G}_2$ (isto é, utiliza-se uma única curva elíptica para se definir o emparelhamento), P e Q são pontos da mesma curva.

⁴A ordem n de um ponto P de uma curva elíptica E é o menor inteiro positivo n tal que $nP = O$, onde O é o ponto no infinito da curva E .

A.2 Tipos de emparelhamento

Enquanto a fundamentação teórica permite que se proponha aplicações baseadas em emparelhamentos bilineares, conhecer os tipos de emparelhamento existentes é importante para materializar e tornar tangíveis as aplicações propostas, pois o tipo de emparelhamento influencia na aplicação implementada, sobretudo, na eficiência nos cálculos.

Os primeiros tipos de emparelhamento que surgiram foram o de Weil e o de Tate⁵. Estes tipos de emparelhamento colaboraram para introduzir o conceito de emparelhamentos bilineares na criptografia, ainda que, em um primeiro momento, objetivando prejudicar a segurança e não provê-la. Isto porque, tais tipos de emparelhamento foram utilizados para reduzir o PLDCE ao PLD simplesmente, em grupos multiplicativos de corpos finitos associados a determinadas classes de curvas. O ganho nessa redução foi que o tempo de resolução do problema passava a ser polinomial (ou seja, factível). No caso, o emparelhamento de Weil foi utilizado para viabilizar o ataque MOV (MENEZES; OKAMOTO; VANSTONE, 1993), enquanto que o emparelhamento de Tate foi utilizado para o ataque FR (FREY; MÜLLER; RÜCK, 1999). Estes ataques despertaram o interesse pelo uso de emparelhamentos na criptografia, os quais passaram a ser utilizados de forma benéfica desde então.

Outros emparelhamentos foram propostos, mas não como totalmente novos, e sim derivados, especificamente, do emparelhamento de Tate: o emparelhamento Eta e o emparelhamento Ate.

O emparelhamento Eta foi proposto por Paulo Barreto, Steven Galbraith, Colm Ó hÉigeartaigh e Michael Scott. Ele é essencialmente o emparelhamento de Tate, mas calculado de maneira bem mais eficiente (BARRETO et al., 2004). Isto é, sob certas condições, ele é computacionalmente mais rápido que o emparelhamento de

⁵Os nomes destes tipos de emparelhamento devem-se aos seus respectivos autores: André Weil (1906 – 1998), matemático francês, e John Terence Tate, matemático norte-americano.

Tate. O resultado final é elevado a uma potência diferente, mas pode ser trivialmente modificado para produzir exatamente o mesmo valor.

A proposta generaliza e otimiza os resultados de outros dois autores – Iwan Duursma e Hyang-Sook Lee – para cálculo do emparelhamento de Tate em curvas elípticas supersingulares de característica 3 (DUURSMA; LEE, 2003). É interessante notar que, após os resultados bem-sucedidos decorrentes da aplicação nessa classe de curvas, resultados eficientes de cálculos também foram obtidos em curvas de característica 2 e em curvas hiperelípticas.

Após o emparelhamento Eta, foi proposto o emparelhamento Ate. E, apesar dos autores deste tipo de emparelhamento – Florian Hess, Nigel Smart e Frederik Vercauteren – declararem em (HESS; SMART; VERCAUTEREN, 2006) que trata-se de um novo emparelhamento, na acepção da palavra trata-se de outra derivação, uma vez que o emparelhamento Ate pode ser considerado uma versão mais genérica do emparelhamento Eta e, por conta disso, também é derivado do emparelhamento de Tate⁶.

De fato, essa nova proposta generaliza o emparelhamento Eta, pois não se restringe apenas às curvas supersingulares, podendo também ser utilizado em curvas ordinárias. Além disso, os resultados obtidos mostram que o algoritmo para cálculo do emparelhamento Ate é mais eficiente que o algoritmo para cálculo do Eta.

Outros trabalhos ainda propuseram aperfeiçoamentos para o emparelhamento Ate, visando reduzir o tamanho do laço principal do algoritmo de Miller: (MATSUDA et al., 2007), emparelhamentos Ate_i (ZHAO; ZHANG; HUANG, 2007) e o emparelhamento $R\text{-}ate$ (LEE; LEE; PARK, 2008). Tais variações têm em comum o fato do tamanho do laço principal ser ao menos $\log_2 r / \varphi(k)$, onde k é o grau de mergulho. Ou seja, $\log_2 r / \varphi(k)$ pode ser considerado um limite inferior.

⁶A denominação “Ate” deve-se a duas razões. Primeira: por ser mais rápido que o emparelhamento de Tate, o “T” se perdeu (“Tate” sem o “T” fica “Ate”). Segunda: o emparelhamento Ate corresponde ao emparelhamento Eta mas com os argumentos em ordem inversa; e “Ate” é “Eta” escrito ao contrário.

Recentemente, foi introduzido o conceito de emparelhamentos ótimos, que alcançam tal limite inferior, isto é, são emparelhamentos que podem ser calculados usando $\log_2 r / \varphi(k)$ iterações básicas de Miller, onde cada iteração corresponde a uma execução do laço principal do algoritmo de Miller (VERCAUTEREN, 2008).

Formalmente: seja $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ um emparelhamento bilinear, não degenerado, onde o número de elementos em $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ é r e o corpo base de G_T é $\mathbb{F}_{q^k}$. Então, $e(\cdot, \cdot)$ é chamado de emparelhamento ótimo, se ele puder ser computado em $\log_2 r / \varphi(k) + \epsilon(k)$ iterações básicas de Miller, com $\epsilon(k) \leq \log_2 k$.

Em (VERCAUTEREN, 2008) ainda é descrito um procedimento totalmente automático para construção de emparelhamentos Ate ótimos em famílias parametrizadas de curvas elípticas amigáveis a emparelhamentos. Além disso, o autor conjectura que, sob determinadas condições específicas favoráveis a uma eficiência computacional, é possível obter emparelhamentos superóticos que requerem menos de $\log_2 r / \varphi(k)$ iterações básicas de Miller para serem calculados.

É importante mencionar que também foram propostos alguns tipos de emparelhamento que não assumem valores inteiros e, por conta disso, não são utilizados na prática para fins criptográficos: o emparelhamento Cassels-Tate (que assume valores nos números racionais – mais especificamente em $\mathbb{Q}/\mathbb{Z}$) e o emparelhamento de Néron-Tate (que assume valores nos números reais – $\mathbb{R}$).

A.3 Prática

A notação $e(\cdot, \cdot)$ representa emparelhamentos bilineares como funções matemáticas, as quais, por sua vez, são representadas por fórmulas para que se chegue a um resultado final. No entanto, tais “fórmulas” não são tão triviais de se calcular, necessitando de algoritmos específicos, dependentes do tipo de emparelhamento em questão, e das curvas elípticas sobre as quais o emparelhamento é definido.

Assim, a implementação eficiente de aplicações práticas envolvendo emparelhamentos deve considerar primeiramente a escolha adequada das curvas elípticas – e para isso, existem algumas famílias de curvas disponíveis. Isto porque nem todas as curvas são amigáveis a emparelhamentos, isto é, o cálculo de emparelhamentos não é realizado eficientemente nelas. Por isso, podem haver curvas boas para se aplicar criptografia com curvas elípticas, mas nem todas são boas para se calcular emparelhamentos. As que são consideradas amigáveis são as de menor grau de mergulho e com ordem prima grande. Como são raras, normalmente são construídas e não descobertas.

Depois, há a questão do algoritmo de cálculo a ser empregado. Nessa linha de pesquisa surgiram diversos trabalhos e propostas nos últimos anos, objetivando a evolução dos algoritmos de cálculos. Um exemplo clássico disto diz respeito ao algoritmo de Miller, que calcula tanto o emparelhamento de Tate quanto o de Weil em tempo polinomial (MILLER, 2004), e que foi aperfeiçoado no algoritmo BKLS (BARRETO et al., 2002). Este algoritmo também é o mais rápido para cálculo de emparelhamentos em uma família específica de curvas elípticas denominada MNT.

Graças a esta evolução, emparelhamentos já podem ser tratados como “caixas-pretas” no que diz respeito à construção de aplicações reais, para as quais já existem algumas ferramentas e recursos disponíveis – geralmente, de forma gratuita.

A.4 Aplicações

Aplicar emparelhamentos bilineares é, essencialmente, fazer uso das propriedades matemáticas dos mesmos, objetivando propor soluções – inovadoras ou alternativas – para determinados problemas no âmbito da criptografia assimétrica. Por estar em um escopo mais matemático, no qual se trabalha apenas com a notação $e(\cdot, \cdot)$, pode-se dizer que uma aplicação proposta começa por um modelo conceitual. A preocupa-

ção nesta etapa não é com tipos de emparelhamentos ou com a forma eficiente de implementá-los. A preocupação maior é que seja estabelecido um modelo conceitual comprovadamente seguro e não vulnerável (ainda que sob determinadas condições aceitáveis). Na próxima etapa, de materialização dos modelos conceituais em aplicações reais e palpáveis, o foco são as questões envolvendo desempenho, viabilidade técnica etc.

Assim, emparelhamentos vêm sendo largamente aplicados e abrindo, inclusive, linhas de pesquisa inovadoras, das quais destacam-se: assinaturas curtas (BONEH; LYNN; SHACHAM, 2004), assinaturas “camaleão” – do inglês, *chameleon signatures* – (KRAWCZYK; RABIN, 1998), cifrassinaturas (BARRETO et al., 2005; MALONE-LEE, 2005; MCCULLAGH; BARRETO, 2004; ZHENG, 1997), infraestruturas de chaves públicas sem certificados digitais (AL-RIYAMI; PATERSON, 2003; DENT, 2006), encriptação baseada em identidade – do inglês, *Identity-Based Encryption* (IBE) – e assinatura baseada em identidade (CHA; CHEON, 2003).

É possível encontrar a materialização de modelos de segurança baseados em emparelhamentos em redes *wireless* (KHALILI; KATZ; ARBAUGH, 2003), em biometria (BURNETT; DUFFY; DOWLING, 2004), em tecnologias emergentes como cartões inteligentes (SCOTT; COSTIGAN; ABDULWAHAB, 2006) e *Radio-Frequency Identification* (RFID), dentre outros casos de uso possíveis.

E há, ainda, outras tecnologias que não foram praticamente exploradas no que diz respeito ao uso de emparelhamentos, tais como telefonia celular, serviços *web* e VoIP.

APÊNDICE B - APLICAÇÕES ENVOLVENDO EMPARELHAMENTOS BILINEARES E SIP

Este apêndice apresenta uma compilação dos trabalhos publicados envolvendo emparelhamentos bilineares e SIP, descrevendo-os objetivamente. Esses trabalhos podem ser considerados, até certo ponto, trabalhos relacionados em relação à proposta de mecanismo de segurança alternativo para o SIP, apresentada nessa dissertação.

B.1 Autenticação e negociação de chaves SIP a partir de IBE

A primeira proposta envolvendo SIP e emparelhamentos bilineares da qual se tem conhecimento fez uso de encriptação baseada em identidade (IBE) para aprimorar a segurança do SIP em um contexto de telefonia móvel (LOU et al., 2005).

Outra proposta utilizando IBE foi feita em (RING et al., 2006) tanto para melhorar o mecanismo de autenticação quanto para propiciar uma negociação segura de chaves. Para melhorar o mecanismo de autenticação “HTTP Digest Authentication”, a encriptação baseada em identidade foi inserida no processo de desafio-resposta do mecanismo. Conforme exemplifica a Figura 27, a mudança é bem pequena. Essencialmente, apenas introduz, na mensagem de autenticação existente no mecanismo Digest, um novo campo para conter uma assinatura.


```
WWW-Authenticate: IdentityBased
  realm = "qut.com",
  nonce = "dcd98b7102dd2f0e8b11d0f600bfb0c093",
  opaque = "5ccc069c403ebaf9f0171e9517f40e41",
  time = "12-12-2005 03:24:12",
  signature = "base64(s(realm|nonce|opaque|time|ClientID,TA_Priv) "
```

a.

```
Authorization: IdentityBased
  username = "client@qut.com",
  realm = "qut.com",
  nonce = "dcd98b7102dd2f0e8b11d0f600bfb0c093",
  opaque = "5ccc069c403ebaf9f0171e9517f40e41",
  response = "base64(e(username|realm|nonce|opaque,A,B_Priv) "
```

b.

Figura 27: Exemplo de desafio (a) e resposta (b) no contexto do mecanismo HTTP Digest Authentication aprimorado pelo uso de encriptação baseada em identidade – extraído de (RING et al., 2006)

IBE também é utilizado pelos autores como base de um mecanismo para negociação de chaves e, daí, estabelecer uma chave secreta comum entre os participantes. Após gerada essa chave comum, a mesma pode ser utilizada com o protocolo SRTP para derivar as quatro chaves necessárias para a encriptação e para a verificação de integridade da sessão de mídia. A Figura 28 ilustra o emprego de IBE para gerar uma chave secreta comum.

Este esquema parte do princípio de que uma TA já distribuiu as chaves privadas S_A e S_B aos *user agents* Alice e Bob, respectivamente. De posse destas chaves privadas, cada *user agent* gera uma assinatura digital (σ_A e σ_B) para garantir a integridade das mensagens trocadas. Além disso, os *user agents* geram valores particulares a e b temporários de modo a calcular $T_A = aP$ e $T_B = bP$, onde P , publicamente conhecido, é um ponto de uma curva elíptica adequadamente escolhida. Os valores σ_A , σ_B , T_A e T_B são transportados como atributos de uma mensagem SDP em um corpo SIP.

Para gerar a chave secreta comum, Alice calcula $K_{AB} = \hat{e}(S_A, T_B) \hat{e}(Q_B, aP_{Pub(TA_2)})$ (onde $P_{Pub(TA_2)}$ é a chave pública da TA do domínio do *user agent* Bob) e $MK_A = H(A||B||\tau_{r_A}||K_{AB})$. Analogamente, Bob calcula $K_{BA} = \hat{e}(S_B, T_A) \hat{e}(Q_A, bP_{Pub(TA_1)})$

Figura 28: Utilização de encriptação baseada em identidade para geração de uma chave secreta comum entre dois *user agents* SIP

(onde $P_{Pub(TA_i)}$ é a chave pública da TA do domínio do *user agent* Alice) e $MK_B = H(A||B||\tau_{r_B}||K_{BA}) = MK_A$. Assim, $MK_B = MK_A = MK$ é a chave secreta comum. Note que H denota uma função de resumo criptográfico segura e τ_{r_A} e τ_{r_B} denotam a concatenação de mensagens enviadas entre as duas partes.

Ainda que os autores afirmem que este esquema para negociação de chaves provê irretratabilidade, isto não é possível pois há, no mínimo, duas entidades no esquema que conhecem a chave privada do *user agent*: o próprio *user agent* e a TA. Além disso,

o esquema parte da premissa de que as chaves privadas já foram distribuídas pela TA aos *user agents*. Mas este processo de distribuição não pode ser assumido como já existente de forma tão simplória. Além de poder ser um ponto de vulnerabilidade, a distribuição de chaves privadas deve ser feita de forma adequada para não gerar um trabalho tão dispendioso quanto o trabalho de geração e distribuição de chaves em uma infraestrutura de chaves públicas, acabando por não compensar o custo-benefício do esquema.

Apesar das restrições e pontos de atenção (além do fato de que a proposta não foi implementada pelos autores para comprovação na prática), o esquema é uma iniciativa interessante, principalmente porque aproveita o próprio processo de sinalização SIP, sem criar novas mensagens e sem alterar campos de cabeçalho. Além disso, ao invés de criar um modelo de segurança totalmente novo, foi feito uso de um modelo já existente: o protocolo probabilisticamente seguro de Chen-Kudla (CHEN; KUDLA, 2002). Em outras palavras, os autores tiveram a iniciativa de adaptar um protocolo existente ao processo de sinalização do SIP, de modo que o grau de segurança do protocolo fosse transferido para a sinalização SIP.

B.2 Autenticação e negociação de chaves SIP baseados em CL-PKC

Na proposta mostrada na seção anterior, os próprios autores de (RING et al., 2006) indicam que uma das preocupações é a confiabilidade das TAs envolvidas e a garantia de que elas não “conspirariam” para burlar o esquema proposto. Tal garantia normalmente só seria alcançada por processos extrassistêmicos tais como contratos judiciais com implicações legais. Outra restrição relacionada com TAs são os cenários P2P (*peer-to-peer*): segundo os autores, o esquema não é adequado para cenários SIP P2P (p. ex., mensagens instantâneas), pois neste tipo de sinalização não existem – e fica difícil instituir – TAs.

Motivados por resolver estas restrições e também se aproveitando do protocolo probabilisticamente seguro de Chen-Kudla, em (WANG; ZHANG, 2008) é proposto um novo mecanismo de autenticação / protocolo para negociação de chaves para o SIP, usando criptografia de chaves públicas sem certificados digitais (CL-PKC – *Certificateless Public-key Cryptography*), uma das aplicações possíveis de emparelhamentos bilineares. A principal vantagem de um esquema CL-PKC é ter uma infraestrutura de chaves públicas, mas sem se preocupar com certificados digitais e com as ações inerentes aos mesmos (emissão, publicação, revogação, renovação).

O mecanismo denominado pelos autores de SAKA – *Secure Authentication and Key Agreement* – é mostrado na Figura 29.

Figura 29: SAKA: mecanismo seguro para autenticação e negociação de chaves para o SIP – adaptado de (WANG; ZHANG, 2008)

K_{AB} e K_{BA} são calculados da seguinte maneira: $K_{AB} = \hat{e}(S_A, T_B)\hat{e}(Q_B, aY_B)$; $K_{BA} = \hat{e}(S_B, T_A)\hat{e}(Q_A, aY_A)$. Daí, chega-se à chave compartilhada $SK_{BA} = H(K_{BA}) = SK_{AB} = H(K_{AB})$.

Os valores S_A , S_B , Q_A , Q_B , Y_A , Y_B , indicados anteriormente, e os valores P_A , P_B , são produzidos pelos algoritmos do processo de inicialização típico de um esquema CL-PKC e que está detalhadamente descrito em (WANG; ZHANG, 2008, p. 5). Cada entidade SIP, *user agent* ou servidor, tem uma identidade própria (p. ex., SIP URI) e estabelece as respectivas chaves públicas e privadas utilizando os algoritmos citados. A função de resumo criptográfico (*hash*) H é previamente estabelecida, em comum acordo, pelas partes.

A proposta deixa a desejar no que diz respeito a como incorporar o esquema no processo de sinalização SIP. O máximo que os autores indicam é que seria aceitável inserir dois novos atributos SDP para viabilizar o esquema.

Por outro lado, uma proposta de segurança para o SIP baseada em CL-PKC é extremamente bem-vinda, pois pode abrir novas frentes de pesquisa nesta linha, assim como aconteceu com o próprio conceito de CL-PKC ao inspirar trabalhos inovadores, como (BARBOSA; FARSHIM, 2008) e (BARRETO et al., 2008).

B.2.1 Aperfeiçoamento do esquema

Muito pouco tempo depois, é publicado um outro trabalho (LUO; WEN; ZHAO, 2008), o qual revisa e critica aquilo que consta em (WANG; ZHANG, 2008), sob diversos aspectos de segurança. Um desses aspectos é o que trata da vulnerabilidade do SAKA ao ataque de substituição de chave – do inglês, *Key Replacement Attack*. Este ataque refere-se a um adversário que pode substituir a identidade pública do usuário destino, a qual passa a ser a chave pública do mesmo, por uma identidade qualquer escolhida pelo adversário. O prejuízo causado por este tipo de ataque ao SAKA é que as partes comunicantes geram uma chave secreta final diferente da esperada.

Além deste problema, em (LUO; WEN; ZHAO, 2008) são apresentados outros, sempre provando a argumentação deles de alguma maneira, mostrando, de fato, as vulnerabilidades do SAKA.

Porém, os autores não ficam apenas nas críticas e propõem um esquema que consiste no aperfeiçoamento do SAKA. Este esquema aperfeiçoado foi denominado pelos autores de MAKP – *Mutually Authenticated Key Agreement Protocol*. No artigo, os autores primeiro melhoram o SAKA, acrescentando novos elementos e novos cálculos – para ser mais específico, duas multiplicações de ponto da curva. Nesta melhora também se aproveita o protocolo probabilisticamente seguro de Chen-Kudla, através do qual se chega não somente à autenticação mútua mas também a uma chave mestra compartilhada entre as partes, as quais não se conhecem previamente.

O MAKP em si corresponde a essa versão melhorada do SAKA, mais um fluxo adicional para prover uma autenticação explícita¹. A Figura 30 mostra toda a “linha evolutiva” do SAKA até chegar no MAKP.

Apesar de ser um artigo interessante e muito bem escrito, sobretudo no que diz respeito às formulações matemáticas, às argumentações e às provas formais, o artigo dedica pouco espaço ao SIP em si, preocupando-se mais em comparar o MAKP com o SAKA. Assim, a proposta do MAKP também deixa a desejar no que diz respeito a detalhar como incorporar o esquema no processo de sinalização SIP.

Mesmo não detalhando praticamente nada sobre a interligação MAKP – SIP, há um exemplo bem ilustrativo do uso do esquema no protocolo de sinalização. O exemplo consiste em Alice fazendo uma chamada para Bob, sendo que ambos não se conhecem pessoalmente, apesar de Alice conhecer a identidade SIP de Bob (no caso, o SIP URI de Bob).

Passo 1: Alice escolhe uma chave privada transitória $a \in \mathbb{Z}_q^*$, calcula $T_A = aP^2$ e prepara a mensagem SIP de “INVITE”, incluindo a chave pública dela (P_A), o identificador da sessão (ts) e T_A . Essa mensagem é enviada para o servidor *proxy* de Alice, que o encaminha para o domínio de segurança de Bob.

¹ Analisando o esquema, o fluxo se comporta como um “reforço” para garantir a autenticação.

² P , no caso, é um dos parâmetros publicados pela TA prevista no esquema.

Figura 30: (a) O esquema SAKA; (b) O esquema SAKA melhorado; (c) O esquema MAKP – extraídos de (LUO; WEN; ZHAO, 2008)

Passo 2: Bob recebe o “INVITE” do *proxy* SIP dele. Se Bob decidir aceitar a chamada ele também escolhe uma chave privada transitória $b \in \mathbb{Z}_q^*$ e calcula $T_B = bP$, $KB_1 = \hat{e}(S_B, T_A) \hat{e}(Q_A, bY_A)$, $KB_2 = (ID_A || ID_B || bT_A || x_B X_A)$, $SK_1 = H_2(T_A || T_B || KB_1 || KB_2)$ e $MAC_1 = MAC_{SK_1}(ts || ID_A || ID_B || T_A || T_B || P_A || P_B)$, colocando estes valores calculados na resposta “200 OK”.

Passo 3: ao receber a mensagem “200 OK” de Bob, Alice calcula $T_A = aP$, $KA_1 = \hat{e}(S_A, T_B) \hat{e}(Q_B, aY_B)$, $KA_2 = (ID_A || ID_B || aT_B || x_A X_B)$, $SK_2 = H_2(T_A || T_B || KA_1 || KA_2)$ e $MAC_2 = MAC_{SK_2}(ts || ID_A || ID_B || T_A || T_B || P_A || P_B)$. Então, Alice verifica se $MAC_1 = MAC_2$, de modo a autenticar Bob. Caso a igualdade não se verifique, Alice encerra a

sinalização. Caso contrário, Alice responde à Bob com uma requisição “ACK” onde ela inclui MAC_2 .

Passo 4: depois que a mensagem “ACK” é recebida, Bob verifica MAC_2 da mesma forma como Alice verificou MAC_1 no Passo 3.

Passo 5: Alice pode agora calcular $SK_{AB} = H(T_A || T_B || KA_1 || KA_2)$ e Bob pode calcular $SK_{BA} = H(T_A || T_B || KB_1 || KB_2) = SK_{AB}$.

Finalmente, ambas as partes podem agora utilizar SK_{AB} com o protocolo padrão SRTP para proteger a sessão de mídia VoIP.

B.3 Assinaturas baseadas em identidade para redes SIP P2P

Recentemente houve esforços para propor uma arquitetura SIP ponto-a-ponto (*peer-to-peer* SIP – P2P-SIP), a qual faz uso de uma rede ponto-a-ponto ao invés de servidores SIP de registro e de localização. Um dos principais problemas em redes SIP P2P é a autenticação de usuários sem um servidor de autenticação (função que normalmente pode ser feita por um servidor SIP *proxy* ou de registro).

De modo a autenticar recursos em uma rede SIP P2P, uma assinatura pode ser anexada ao final da mensagem SIP, sendo esta assinatura criada sob o conceito de uma infraestrutura de chaves públicas convencional. Entretanto, um modelo de assinatura desta natureza em redes SIP P2P requer verificação de autoridades de certificação durante o roteamento de mensagens SIP, além de manutenção de certificados digitais nos nós intermediários da rede para o procedimento de autenticação. Isto normalmente causa um razoável atraso na sinalização, a medida que novos usuários se juntam à rede P2P. E atrasos na sinalização de comunicações multimídia em tempo real pode ser um problema significativo.

Por outro lado, a falta de segurança pode ser um problema ainda maior. Sem

mecanismos de defesa, atacantes podem gerar identidades falsas e se tornarem nós (“pontos”) no sistema, a partir dos quais eles poderiam interferir no roteamento de mensagens e na manutenção da rede. Eles poderiam também mascarar outros usuários válidos (ou mesmo serviços) gerando respostas falsas.

Assim, em (LEE; HAN; LEE, 2008) é proposto um modelo de segurança para redes SIP P2P usando assinatura baseada em identidade, especificamente o esquema de assinatura baseado em identidade de Hess (HESS, 2003). A proposta compreende a fase de registro de um recurso e a fase de localização (e uso) do mesmo. As assinaturas baseada em identidade (que neste caso, pode ser um endereço de e-mail ou o SIP URI) são anexadas no final das mensagens SIP.

De forma resumida, o esquema funciona desta maneira: inicialmente, um usuário cria um par de chaves pública e privada. Então, o usuário assina o recurso com uma chave privada e registra o recurso assinado na rede P2P. No processo de localização do recurso, o usuário pega o recurso assinado da rede P2P e verifica esse recurso com a chave pública.

Objetivando analisar, sobretudo, a latência na rede, os autores realizaram diversas simulações, nas quais os resultados mostraram a eficiência e escalabilidade dos processos de registro e localização de recursos. Especialmente, a medida que usuários cresciam na rede SIP P2P, o esquema de assinatura baseado em identidade tinha baixo custo de consultas (*lookup*) quando comparado ao modelo baseado em infraestrutura de chaves públicas convencional.

A proposta só não é muito clara quanto a como integrar de forma prática as assinaturas, chaves etc. no contexto de mensagens SIP e campos de cabeçalho. Por outro lado, mesmo sendo razoavelmente simples, a proposta é um tanto inovadora. É importante, pois pode estimular novas aplicações de emparelhamentos neste contexto de segurança para redes SIP P2P, as quais tendem a crescer nos próximos anos, não tanto por causa do SIP, mas mais pelas vantagens do modelo P2P.

APÊNDICE C - REDUÇÃO DAS POSSIBILIDADES DE ATAQUE DO TIPO *MAN-IN-THE-MIDDLE* AO ECMO

Devido ao fato do Massey-Omura ser um protocolo de três passos, há mais dados em trânsito entre as partes comunicantes. Este fato mais a própria característica de três passos acaba amplificando a já conhecida vulnerabilidade do protocolo frente a ataques do tipo *man-in-the-middle*, uma vez que o atacante pode ter mais valores “em mãos” com que trabalhar e atacar, e mais de uma chance de ataque, na mesma rodada do protocolo.

Além disso, nas várias literaturas estudadas que tratam da definição do protocolo Massey-Omura, tudo indica que os valores secretos gerados pelas partes comunicantes não mudam a cada rodada do protocolo. Uma evidência disso é o seguinte trecho: “Em segundo lugar, é importante que, depois de um usuário tal qual B ou C ¹ tenha decifrado várias mensagens P , e conheça vários pares (P, P^{e_A}) , ele não possa usar essas informações para determinar e_A . Isto é, suponha que Bob pudesse resolver o problema do logaritmo discreto em F_q^* , daí determinando, a partir de P e P^{e_A} , qual deve ser o valor de e_A . Neste caso, ele poderia rapidamente calcular $d_A = e_A^{-1} \bmod (q - 1)$ e então interceptar e ler todas as mensagens futuras que Alice enviasse, sejam elas destinadas ou não a ele.” (KOBELITZ, 1994, p. 100, tradução nossa). O texto dá a entender

¹B seria Bob e C o *man-in-the-middle* no contexto colocado pelo autor.

que o valor secreto escolhido por Alice (e_A , segundo o autor) é o mesmo em qualquer rodada do protocolo.

Em (DEUSAJUTE; BARRETO, 2008), ao contrário, afirma-se explicitamente que tais valores secretos devem ser escolhidos, de forma aleatória, a cada rodada do protocolo. E não há prejuízo algum em se proceder desta maneira, caso não se atrele o protocolo a uma infraestrutura de chaves públicas, na qual as chaves privada e pública não podem mudar em um curto período de tempo.

Essa simples condição de que os valores secretos devem mudar a cada rodada do protocolo, mitiga bastante os riscos decorrentes da criptoanálise e de ataques do tipo *man-in-the-middle*. Neste último caso, o atacante teria apenas a rodada do protocolo em que ele estiver agindo para tentar conseguir algo, de nada adiantando guardar valores para serem reaproveitados em outras rodadas. Isto permite reduzir as possibilidades deste tipo de ataque a um número pequeno de cenários efetivos na prática, os quais são discutidos nesse apêndice.

C.1 Lógica da redução

Além da premissa de que os valores secretos do protocolo mudam a cada rodada do mesmo, assumiu-se ainda as seguintes condições na análise apresentada nesta seção:

- O atacante pode atuar na primeira, na segunda e/ou na terceira transição, fazendo uso, principalmente, de ataques de substituição, combinados ou não, com ataques de repetição – do inglês, *replay attacks*. O ataque pode ocorrer em mais de uma das transições do protocolo, mas desde que em uma mesma rodada do mesmo – devido à premissa estabelecida.
- Não foi considerada a possibilidade de recuperação direta dos valores secretos em trânsito, uma vez que isto seria impraticável em apenas uma rodada do

protocolo, pois implicaria em resolver o PLDCE.

- Não foram considerados os cenários de ataque que provocassem a interrupção arbitrária do protocolo, impedindo que os valores enviados por uma parte chegassem até a outra em alguma das transições, e, assim, o protocolo não fosse executado por completo pelo menos até a terceira transição. Este tipo de ataque pode ser entendido como um caso particular de DoS.

Optou-se por não considerar esses cenários de ataque por dois motivos. Primeiro: devido à característica de três passos do protocolo Massey-Omura, ficaria fácil, na prática, a detecção deste tipo de cenário de ataque. Por exemplo, poderia ser utilizado um recurso de *timeout*: se uma parte enviasse algo para outra, e esta não respondesse após um determinado período de tempo, então o protocolo seria reinicializado. Segundo: há diversas outras maneiras – mais simples até, e em outros níveis, como camada de transporte, camada de rede etc. – de um atacante gerar um ataque de DoS.

- Assumiu-se a execução das transições do protocolo na devida ordem, conforme definição do mesmo. Em outras palavras, o atacante não poderia, por exemplo, solicitar que a origem executasse o passo 2, sem antes a mesma ter executado o passo 1.

Assuma que uma parte origem – denominada “Alice”, cujo valor secreto é x_A , e o respectivo valor inverso é x_A^{-1} – deseja enviar uma mensagem secreta $\mathcal{M}$ para outra parte destino – denominada “Bob”, cujo valor secreto é x_B , e o respectivo valor inverso é x_B^{-1} – sobre um canal não seguro.

Por ser um canal não seguro, sem qualquer tipo de segurança adicional que garanta ao menos confidencialidade e integridade no que transitar pelo canal, ele está sujeito a ataques do tipo *man-in-the-middle* executados por um intruso, denominado “MITM”, cujo valor secreto é x_λ , e o respectivo valor inverso é x_λ^{-1} . Assuma ainda $\mathcal{M}_\lambda$

como sendo uma mensagem qualquer que MITM pode gerar e incluir na transmissão.

Dado este contexto, há diversos cenários de ataque possíveis, sendo que nem todo cenário resulta em um ataque efetivo. Isto dependerá do resultado final obtido.

Na primeira transição, MITM pode atacar o valor M_A enviado por Alice de várias formas, gerando os seguintes valores possíveis de serem enviados a Bob posteriormente²:

$$M_{A_1} = M_A$$

$$M_{A_2} = x_\lambda M_A$$

$$M_{A_3} = x_\lambda \mathcal{M}_\lambda$$

$$M_{A_4} = \mathcal{M}_\lambda$$

Para cada um desses valores possíveis que chegasse até Bob, em rodadas diferentes do protocolo, seriam gerados os respectivos valores M_{BA_i} correspondentes à $x_B M_{A_i}$:

$$M_{BA_1} = x_B M_{A_1} = x_B M_A$$

$$M_{BA_2} = x_B M_{A_2} = x_B x_\lambda M_A$$

$$M_{BA_3} = x_B M_{A_3} = x_B x_\lambda \mathcal{M}_\lambda$$

$$M_{BA_4} = x_B M_{A_4} = x_B \mathcal{M}_\lambda$$

Na segunda transição, de Bob para Alice, MITM poderia interceptar e manipular valores M_{BA_i} , gerando outros valores possíveis de serem utilizados. Manipular valores nessa transição significa apenas multiplicar os valores possíveis M_{BA_i} tanto por x_λ

²Apesar de MITM poder utilizar também x_λ^{-1} para gerar outras possibilidades de M_{A_i} , isto não traria ganho significativo, pois utilizar x_λ^{-1} significa que, depois, ele teria que utilizar x_λ para tentar decifrar algo. Só que isto é o mesmo que aplicar x_λ (como, de fato, é feito por MITM), e tentar decifrar algo com x_λ^{-1} . O mesmo raciocínio aplica-se às demais transições, de modo a não ter que usar sempre x_λ e x_λ^{-1} em todas as possibilidades.

Ressaltando que esta flexibilidade de se trabalhar alternando entre x_λ e x_λ^{-1} só é possível devido ao Massey-Omura possuir um esquema de encriptação comutativo, conforme a Definição 1 dada no Capítulo 3.

quanto por x_λ^{-1} . Isto porque não faria sentido algum multiplicar os M_{BA_i} por algum dos valores obtidos na primeira transição, pois isso acrescentaria mais “camadas de valores” e evitaria qualquer tipo de ganho futuro para o MITM.

Porém, ainda assim, MITM poderia usar algum dos valores gerados na primeira transição.

O novo conjunto de valores possíveis seria :

$$M_{XA_1} = M_{A_1} = M_A$$

$$M_{XA_2} = M_{A_2} = x_\lambda M_A$$

$$M_{XA_3} = M_{A_3} = x_\lambda M_\lambda$$

$$M_{XA_4} = M_{A_4} = M_\lambda$$

$$M_{XA_5} = M_{BA_1} = x_B M_A$$

$$M_{XA_6} = M_{BA_2} = x_B x_\lambda M_A$$

$$M_{XA_7} = M_{BA_3} = x_B x_\lambda M_\lambda$$

$$M_{XA_8} = M_{BA_4} = x_B M_\lambda$$

$$M_{XA_9} = x_\lambda M_{BA_1} = x_\lambda x_B M_A$$

$$M_{XA_{10}} = x_\lambda M_{BA_2} = x_\lambda x_B x_\lambda M_A = x_\lambda^2 x_B M_A$$

$$M_{XA_{11}} = x_\lambda M_{BA_3} = x_\lambda x_B x_\lambda M_\lambda = x_\lambda^2 x_B M_\lambda$$

$$M_{XA_{12}} = x_\lambda M_{BA_4} = x_\lambda x_B M_\lambda$$

$$M_{XA_{13}} = x_\lambda^{-1} M_{BA_1} = x_\lambda^{-1} x_B M_A$$

$$M_{XA_{14}} = x_\lambda^{-1} M_{BA_2} = x_\lambda^{-1} x_B x_\lambda M_A = x_B M_A$$

$$M_{XA_{15}} = x_\lambda^{-1} M_{BA_3} = x_\lambda^{-1} x_B x_\lambda M_\lambda = x_B M_\lambda$$

$$M_{XA_{16}} = x_\lambda^{-1} M_{BA_4} = x_\lambda^{-1} x_B M_\lambda$$

Note que de M_{XA_1} à M_{XA_4} , MITM estaria reaproveitando os valores capturados (manipulados ou não) na primeira transição. De M_{XA_5} à M_{XA_8} , ele estaria apenas re-passando o que Bob gera. Já de M_{XA_9} à $M_{XA_{12}}$ e de $M_{XA_{13}}$ à $M_{XA_{16}}$, MITM estaria manipulando os valores devolvidos por Bob, multiplicando-os por x_λ e x_λ^{-1} , respectivamente.

Neste ponto, MITM poderia se comportar de duas maneiras: personificar Bob na segunda transição, enviando cada um dos valores M_{XA_i} para Alice, ou personificar Alice na terceira transição, enviando cada um dos valores M_{XA_i} para Bob.

No primeiro caso, no qual MITM se passaria por Bob, e enviaria algum dos valores possíveis gerados na segunda transição para Alice, esta, seguindo o protocolo ECMO, geraria algum dos seguintes valores possíveis:

$$M_{B_1} = x_A^{-1} M_{XA_1} = x_A^{-1} M_A = x_A^{-1} x_A \mathcal{M} = \mathcal{M}$$

$$M_{B_2} = x_A^{-1} M_{XA_2} = x_A^{-1} x_\lambda M_A = x_A^{-1} x_\lambda x_A \mathcal{M} = x_\lambda \mathcal{M}$$

$$M_{B_3} = x_A^{-1} M_{XA_3} = x_A^{-1} x_\lambda \mathcal{M}_\lambda$$

$$M_{B_4} = x_A^{-1} M_{XA_4} = x_A^{-1} \mathcal{M}_\lambda$$

$$M_{B_5} = x_A^{-1} M_{XA_5} = x_A^{-1} x_B M_A = x_A^{-1} x_B x_A \mathcal{M} = x_B \mathcal{M}$$

$$M_{B_6} = x_A^{-1} M_{XA_6} = x_A^{-1} x_B x_\lambda M_A = x_A^{-1} x_B x_\lambda x_A \mathcal{M} = x_B x_\lambda \mathcal{M}$$

$$M_{B_7} = x_A^{-1} M_{XA_7} = x_A^{-1} x_B x_\lambda \mathcal{M}_\lambda$$

$$M_{B_8} = x_A^{-1} M_{XA_8} = x_A^{-1} x_B \mathcal{M}_\lambda$$

$$M_{B_9} = x_A^{-1} M_{XA_9} = x_A^{-1} x_\lambda x_B M_A = x_A^{-1} x_\lambda x_B x_A \mathcal{M} = x_\lambda x_B \mathcal{M}$$

$$M_{B_{10}} = x_A^{-1} M_{XA_{10}} = x_A^{-1} x_\lambda^2 x_B M_A = x_A^{-1} x_\lambda^2 x_B x_A \mathcal{M} = x_\lambda^2 x_B \mathcal{M}$$

$$M_{B_{11}} = x_A^{-1} M_{XA_{11}} = x_A^{-1} x_\lambda^2 x_B \mathcal{M}_\lambda$$

$$M_{B_{12}} = x_A^{-1} M_{XA_{12}} = x_A^{-1} x_\lambda x_B \mathcal{M}_\lambda$$

$$M_{B_{13}} = x_A^{-1} M_{XA_{13}} = x_A^{-1} x_\lambda^{-1} x_B M_A = x_A^{-1} x_\lambda^{-1} x_B x_A \mathcal{M} = x_\lambda^{-1} x_B \mathcal{M}$$

$$M_{B_{14}} = x_A^{-1} M_{XA_{14}} = x_A^{-1} x_B M_A = x_A^{-1} x_B x_A \mathcal{M} = x_B \mathcal{M}$$

$$M_{B_{15}} = x_A^{-1} M_{XA_{15}} = x_A^{-1} x_B \mathcal{M}_\lambda$$

$$M_{B_{16}} = x_A^{-1} M_{XA_{16}} = x_A^{-1} x_\lambda^{-1} x_B \mathcal{M}_\lambda$$

Note que, além de M_{B_1} (onde já se expõe a mensagem $\mathcal{M}$ em claro), só interessará ao MITM os valores onde, além da mensagem em si (seja ela $\mathcal{M}$ ou $\mathcal{M}_\lambda$), estejam presentes apenas ou x_λ ou x_λ^{-1} . Pois os únicos cálculos úteis que MITM poderá fazer na transição três (que é a final para ele) envolverão ou x_λ ou x_λ^{-1} . Quaisquer outros valores capturados e/ou manipulados nas outras transições que MITM possa querer aplicar não resultarão em algo aproveitável. Mesmo que MITM passe para Bob algum dos valores possíveis, manipulados ou não nesta terceira transição, ou Bob acabará recuperando normalmente $\mathcal{M}$ (p. ex., se MITM simplesmente repassar $M_{B_{14}}$), ou Bob recuperará um valor que não faz sentido algum.

Assim, dos valores M_{B_i} devolvidos por Alice, além de M_{B_1} , MITM só conseguirá aproveitar M_{B_2} . Pois $x_\lambda^{-1} M_{B_2} = x_\lambda^{-1} x_\lambda \mathcal{M} = \mathcal{M}$.

No segundo caso, no qual MITM personificaria Alice na terceira transição, e enviaria M_{XA_i} para Bob, este, executando o fechamento do ECMO, obterá um dos seguintes valores finais:

$$M_1 = x_B^{-1} M_{XA_1} = x_B^{-1} M_A = x_B^{-1} x_A \mathcal{M}$$

$$M_2 = x_B^{-1} M_{XA_2} = x_B^{-1} x_\lambda M_A = x_B^{-1} x_\lambda x_A \mathcal{M}$$

$$M_3 = x_B^{-1} M_{XA_3} = x_B^{-1} x_\lambda \mathcal{M}_\lambda$$

$$M_4 = x_B^{-1} M_{XA_4} = x_B^{-1} \mathcal{M}_\lambda$$

$$M_5 = x_B^{-1} M_{XA_5} = x_B^{-1} x_B M_A = x_B^{-1} x_B x_A \mathcal{M} = x_A \mathcal{M}$$

$$M_6 = x_B^{-1} M_{XA_6} = x_B^{-1} x_B x_\lambda M_A = x_B^{-1} x_B x_\lambda x_A \mathcal{M} = x_\lambda x_A \mathcal{M}$$

$$M_7 = x_B^{-1} M_{XA_7} = x_B^{-1} x_B x_\lambda \mathcal{M}_\lambda = x_\lambda \mathcal{M}_\lambda$$

$$M_8 = x_B^{-1} M_{XA_8} = x_B^{-1} x_B \mathcal{M}_\lambda = \mathcal{M}_\lambda$$

$$M_9 = x_B^{-1} M_{XA_9} = x_B^{-1} x_\lambda x_B M_A = x_B^{-1} x_\lambda x_B x_A \mathcal{M} = x_\lambda x_A \mathcal{M}$$

$$M_{10} = x_B^{-1} M_{XA_{10}} = x_B^{-1} x_\lambda^2 x_B M_A = x_B^{-1} x_\lambda^2 x_B x_A \mathcal{M} = x_\lambda^2 x_A \mathcal{M}$$

$$M_{11} = x_B^{-1} M_{XA_{11}} = x_B^{-1} x_\lambda^2 x_B M_A = x_\lambda^2 \mathcal{M}_\lambda$$

$$M_{12} = x_B^{-1} M_{XA_{12}} = x_B^{-1} x_\lambda x_B \mathcal{M}_\lambda = x_\lambda \mathcal{M}_\lambda$$

$$M_{13} = x_B^{-1} M_{XA_{13}} = x_B^{-1} x_\lambda^{-1} x_B M_A = x_B^{-1} x_\lambda^{-1} x_B x_A \mathcal{M} = x_\lambda^{-1} x_A \mathcal{M}$$

$$M_{14} = x_B^{-1} M_{XA_{14}} = x_B^{-1} x_B M_A = x_A^{-1} x_B x_A \mathcal{M} = x_A \mathcal{M}$$

$$M_{15} = x_B^{-1} M_{XA_{15}} = x_B^{-1} x_B \mathcal{M}_\lambda = \mathcal{M}_\lambda$$

$$M_{16} = x_B^{-1} M_{XA_{16}} = x_B^{-1} x_\lambda^{-1} x_B \mathcal{M}_\lambda = x_\lambda^{-1} \mathcal{M}_\lambda$$

Procedendo desta forma, MITM teria sucesso apenas com os valores M_8 e M_{15} , pois Bob receberia uma mensagem falsa, acreditando ser verdadeira e proveniente de Alice.

C.2 Cenários de ataque efetivos

Conforme visto na seção anterior, o fato dos valores secretos mudarem a cada rodada do protocolo – conforme proposto em (DEUSAJUTE; BARRETO, 2008) – é crucial para reduzir-se os cenários de ataque. Caso contrário, cada um dos valores possíveis gerados em uma rodada do protocolo, por captura simples ou por manipulação, poderia ser reaproveitado de diversas maneiras, nas mais variadas combinações, em outras rodadas.

Mesmo limitando-se os cenários de ataque possíveis, nem todos produzem algo aproveitável no final, dentro das condições que foram estipuladas no início da seção anterior. Os cenários de ataque realmente efetivos, isto é, que trazem algum benefício

para o atacante, são aqueles mostrados nas Figuras 31, 32, 33 e 34.

Figura 31: Ataque do tipo *man-in-the-middle* sobre o ECMO onde a mensagem secreta é obtida diretamente pelo atacante

Note porém, na Figura 31, que basta comparar o que Alice recebe na segunda transição com o valor de M_A que foi enviado na primeira transição para evitar esse cenário de ataque. Uma comparação análoga pode ser feita para evitar o cenário de ataque mostrado na Figura 33, comparando o que Bob enviou na segunda transição com o que ele recebeu na terceira transição. Ou seja, pelo menos nestes pontos, não é necessário nenhum recurso criptográfico sofisticado.

Por outro lado, as Figuras 32 e 34 representam cenários de ataque similares aos mostrados nas Figuras 31 e 33, mas que dão um pouco mais de veracidade aos ataques, de modo que uma simples comparação não é suficiente para evitá-los.

Ainda assim, esses cenários, de forma isolada, podem não trazer muito ganho para o atacante, dependendo do contexto e caso o atacante interrompa a comunicação.

Por exemplo, em um contexto específico de comunicação VoIP, interrompê-la faria

Figura 32: Ataque do tipo *man-in-the-middle* sobre o ECMO onde a mensagem secreta é obtida indiretamente pelo atacante

Figura 33: Ataque do tipo *man-in-the-middle* sobre o ECMO onde o atacante consegue passar uma mensagem definida por ele mesmo, diretamente para Bob

Figura 34: Ataque do tipo *man-in-the-middle* sobre o ECMO onde o atacante consegue passar uma mensagem definida por ele mesmo, indiretamente para Bob

com que aquele participante que foi “desligado” tente reiniciá-la. Ainda nesse contexto, mesmo que o atacante interrompesse o processo de sinalização e passasse uma mensagem representando a chave secreta para encriptar a sessão de mídia, o máximo que poderia acontecer seria uma comunicação segura com o próprio atacante.

Entretanto, mesmo reduzindo-se os cenários de ataque efetivos, de quatro para dois (supondo que sejam tomadas as medidas simples de comparação descritas anteriormente), ainda assim, estes dois únicos cenários, quando combinados adequadamente, geram uma possibilidade de ataque singular, independente de contexto, e que atinge o objetivo maior do *man-in-the-middle*: descobrir a mensagem secreta que Alice envia para Bob, sem interromper a comunicação e, daí, sem que eles suspeitem de que um intruso interferiu nela e descobriu a mensagem enviada. É isto que mostra a Figura 35.

Se, porventura, Alice e Bob, de alguma forma extrassistêmica (p. ex., um tele-

lidade e a integridade de uma posterior comunicação entre Alice e Bob, o atacante poderá atuar nessa comunicação como um adversário passivo, não alterando o fluxo ou modificando-o, apenas decifrando-o, já que ele tem conhecimento da chave criptográfica secreta que Alice e Bob estão utilizando.

APÊNDICE D – PUBLICAÇÕES DO AUTOR

1. DEUSAJUTE, A. M.; BARRETO, P. S. L. M. **The SIP Security Enhanced by using Pairing-assisted Massey-Omura Signcryption.** In: *Actas de la X Reunión Española sobre Criptología y Seguridad de la Información - RECSI*. Salamanca, Espanha, 2008. p. 51–62.

Neste artigo é feito um primeiro ensaio sobre um novo mecanismo de segurança alternativo para o SIP. Apesar do artigo não utilizar a denominação “EECMO”, o novo mecanismo proposto apresentado no artigo também foi baseado no protocolo Massey-Omura – já variando-se os valores secretos selecionados aleatoriamente a cada rodada do protocolo – e também utilizou emparelhamentos bilineares.

Entretanto, o esquema indicava a necessidade de uma infraestrutura de chaves públicas. Além disso, apesar do artigo mostrar como acoplar o esquema ao SIP em uma situação real, não foi realizada prova de conceito ou qualquer outra experimentação prática. Estudos posteriores mostraram que um acoplamento mais eficiente – tanto em termos de desempenho quanto em termos de sobrecarga nas mensagens SIP – seria obtido através da RFC 4567. Tais estudos acabaram sendo relatados na presente dissertação.

Ainda assim, o artigo acaba se destacando de outros trabalhos similares não somente por ser uma contribuição relevante aos poucos trabalhos publicados envolvendo emparelhamentos e SIP, mas também por investigar e ensaiar aspectos práticos de implementação, em um nível razoável de detalhamento.

2. BARRETO, P. S. L. M.; DEUSAJUTE, A. M.; CRUZ, E. S.; PEREIRA, G. C. F.; SILVA, R. R. **Toward Efficient Certificateless Signcryption from (and without) Bilinear Pairings**. In: *VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais – SBSeg'2008*. Gramado/RS, Brasil, 2008. p. 115–125.

Este trabalho apresenta um novo esquema de cifrassinatura – do inglês, *signcryption* – denominado BDCPS¹. O esquema corresponde a uma integração de outros esquemas: IBS BLMQ (o qual foi mostrado no Capítulo 3), assinaturas convencionais Schnorr (SCHNORR, 1991), cifrassinatura Zheng (ZHENG, 1997) e criptosistema de chaves públicas sem certificado digital de Al-Riyami e Paterson (AL-RIYAMI; PATERSON, 2003).

Apesar do novo esquema ser derivado do conceito de encriptação e assinatura baseadas em identidade, o esquema provê irretratabilidade. Isto porque, mesmo que a TA (também presente no esquema) ainda forneça a chave privada baseada em identidade (como no esquema IBS BLMQ), a chave pública está relacionada a uma outra chave privada, a qual, por sua vez, é composta pela chave privada baseada em identidade e por um valor secreto que somente o possuidor da chave pública conhece.

Outra vantagem do esquema é que ele é mais eficiente quando comparado a outros esquemas similares, sobretudo pelo fato de que a (única) operação de emparelhamento é transferida da fase de verificação da assinatura para a fase de verificação da chave pública. Portanto, na prática, como a verificação de chaves públicas tende a ser realizada uma única vez, gastar-se-ia mais com processamento uma única vez também, ao passo que verificações de assinaturas ocorrem com mais frequência.

¹Essa denominação corresponde às letras iniciais do último sobrenome de cada um dos autores.