Universidade Federal do Rio de Janeiro

MÁRCIO ROBERTO GALHANO

Registro e Autenticação Remotos com Otimização do Fluxo de Mídia em Arquitetura SIP

DISSERTAÇÃO DE MESTRADO

Instituto de Matemática

Núcleo de Computação Eletrônica

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

MÁRCIO R GALHANO

Registro e Autenticação Remotos com Otimização do Fluxo de Mídia em Arquitetura SIP

Prof. Dr. Paulo Henrique de Aguiar Rodrigues
Orientador

RIO DE JANEIRO
2009.

Márcio R. Galhano

Registro e Autenticação Remotos com Otimização do Fluxo de Mídia em Arquitetura SIP

Dissertação de Mestrado submetida ao Corpo Docente do Departamento de Ciência da Computação do Instituto de Matemática, e Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para obtenção do título de Mestre em Informática.

Aprovada em 15 de maio de 2009.

Prof. Dr. Paulo Henrique de Aguiar Rodrigues, Ph.D
(Universidade Federal do Rio de Janeiro)

Profa. Dra. Luci Pirmez, Ds.c
(Universidade Federal do Rio de Janeiro)

Prof. Dr. Michael Anthony Stanton, Ph.D
(Universidade Federal Fluminense)

Galhano, Márcio R.

Registro e Autenticação Remotos com Otimização do Fluxo de Mídia em Arquitetura SIP / Márcio R. Galhano. – Rio de Janeiro: UFRJ IM, 2009.

107 f.: il.

Dissertação (Mestrado em Informática) – Universidade Federal do Rio de Janeiro. Programa de Pós-Graduação em Informática, Rio de Janeiro, BR-RJ, 2009.

Orientador: Paulo Henrique de Aguiar Rodrigues.

I. Rodrigues, P. H. de A. II. Título.

AGRADECIMENTOS

Aos companheiros de trabalho na COBAD/IBGE, que, através de apoio e estímulo, contribuíram para que eu realizasse o curso: Vânia Costa, Elineide Maria dos Santos, Verônica Santos, Valéria Roitman e Luiz Tavares.

Aos companheiros do LabVoIP, que, apesar de pequena convivência, contribuíram muito durante uma etapa crucial do mestrado: Thiago Maluf Resende, Cláudio Miceli de Farias, Paulo Vitor Barion Heckmaier, Rafael Vilardo e outros.

À minha família que amo muito, por estar sempre ao meu lado me apoiando, mesmo com minha ausência no decorrer do curso, principalmente a minha esposa Luciane L. P. Galhano e a meus sogros Luiz L. Pires e Francisca L. Pires, por nos ajudar na educação de nosso tesouro maior, nossa filha Isabela L. P. Galhano.

Em especial, ao meu orientador Paulo Henrique de Aguiar Rodrigues, pela confiança depositada e oportunidade de desenvolver este trabalho e, principalmente, por me direcionar nos momentos ímpares do curso.

Ao IM/UFRJ e NCE/UFRJ, pela infraestrutura disponível para realização do curso, e principalmente aos funcionários, prestadores de serviço e professores.

A Deus, pela luz, saúde e oportunidades.

Aos meus pais, Francisco Galhano e Dalva G. Galhano, pela educação, amor, apoio e estímulo, em todas as fases da minha vida.

RESUMO

GALHANO, Márcio Roberto. **Registro e Autenticação Remotos com Otimização do Fluxo de Mídia em Arquitetura SIP**. Rio de Janeiro, 2009. Dissertação (Mestrado em Informática) – PPGI/NCE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2009.

São apresentadas as adaptações na sinalização SIP para permitir que os usuários consigam realizar, de forma descentralizada, o registro em um serviço VoIP SIP, com a validação ocorrendo no domínio de origem através de autenticação *digest*, segundo a RFC 2617, como consequência foi obtida a otimização do caminho seguido pela mídia nas chamadas do usuário em mobilidade. Adicionalmente, a solução contempla o transporte de créditos de ligações associadas ao usuário diretamente pela sinalização SIP. A solução desenvolvida é validada com a implementação em ambiente de software livre com uso de DNS ou mecanismo de localização P2P.

Palavras-chave: Registro, Autenticação, Mobilidade, Sinalização SIP

ABSTRACT

GALHANO, Márcio Roberto. **Registro e Autenticação Remotos com Otimização do Fluxo de Mídia em Arquitetura SIP**. Rio de Janeiro, 2009. Dissertação (Mestrado em Informática) – PPGI/NCE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2009.

SIP Signalling extensions to allow SIP decentralized registration with authentication and validation in the user home domain are presented. Authentication is performed according to RFC 2617 digest mechanism and the result was from the optimization of the path followed by the media calls for the user in mobility. Additionally, user call credit information is also transferred directly by the SIP extended signaling, improving the functionality associated to the architecture. Implementation based on open software architecture, with use of DNS or P2P location mechanisms, is shown and it demonstrates the validity and usability of the proposed mechanisms.

Key-Words: SIP, Authentication, Mobility, Registration, Remote

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

2G/SMS	Second Generation/Short Messaging Services
AAA	Authentication, Authorization and Accounting
ANSI	American National Standard International
AoR	Address of Record
CDR	Call Detail Record
CHAP	Challenge Authentication Protocol
DBLP	Digital Bibliography & Library Project
DHCP	Dynamic Host Configuration Protocol
DHT	Distributed Hash Table
DNS	Domain Name System
DoS	Denial of Service
EAP	Extensible Authentication Protocol
ENUM	E.164 Number Mapping
GW	Gateway
H.248	Gateway Control Protocol (MGCP)
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Messaging Protocol
IDP	Identity Provider
IETF	Internet Engineering Task Force
IM	Instant Messenger
IMS	Internet Protocol Multimedia Subsystem
IP	Internet Protocol
IPSec	IP Security
ISP	Internet Solution Provider
ITU-T	International Telecomunication Union - Telephony
KBR	Key Based Routing
LAN	Local Area Network
LCR	Least Cost Routing
LDAP	Lightweight Directory Access Protocol
LRU	Least Recently Used

MACE	Middleware Architeture Committee for Education
MD5	Message Digest 5
MGCP	Media Gateway Control Protocol
MiME	Multipurpose Internet Mail Extensions
MitM	Man in the Middle
MSN	Microsoft Send Messaging
NAT	Network Address Translation
NGN	Next Generation Networks
P2P	Peer-to-Peer
PAP	Password Authentication Protocol
PBX	Private Branch Exchange
PRACK	Positive Retransmit Acknowledgement
PSTN	Public Switched Telephone Network
QoS	Quality of Services
RADIUS	Remote Authentication Dial In User Service
RFC	Request For Comments
RTP	Real-Time Transport Protocol
SDP	Session Description Protocol
SER	Session Express Router
SGBD	Sistema Gerenciador de Banco de Dados
SIP	Session Initiation Protocol
SIPS	Session Initiation Protocol Secure
SP	Service Provider
SQL	Session Query Language
SRTP	Secure Real-Time Transport Protocol
SSO	Single-Sign-On
UA	User Agent
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
VLAN	Virtual Local Area Network
VoIP	Voice over Internet Protocol
VPN	Virtual private Network

SUMÁRIO

Capítulo 1

Introdução

1.1 - Motivação

No contexto de um serviço de telefonia IP, em que instituições clientes usam a Internet para viabilizar comunicações por voz (via telefone comum, telefone IP ou software) com outras instituições, a disponibilização do serviço para o usuário final através de contas/senhas é atribuição das instituições que compõem o serviço. Neste cenário, encaixa-se o serviço fone@RNP da RNP, pelo qual os usuários são registrados e autenticados no domínio de origem, independentemente de onde estejam conectados. O serviço fone@RNP [Acesso: <http://www.rnp.br/voip> disponível em Março de 2009], como a maioria dos serviços VoIP padronizados atuais, utiliza preferencialmente o protocolo SIP [ROSENBERG, J.; SCHULZRINNE, H., CAMARILLO, G., et al., 2002] para a sinalização VoIP.

Mais além, um serviço SIP nacional requer que as instituições participantes possuam servidores funcionando como *proxies* de mídia e sinalização, para que apenas o tráfego vindo diretamente destes *proxies* conhecidos seja tratado no *backbone* do serviço de forma diferenciada, com habilitação de QoS. Quando o *backbone* analisa a origem do tráfego marcado, consultando o endereço IP de origem, o tráfego que não tem origem em nenhum dos endereços IPs dos *proxies* do serviço e que tem seus pacotes IP marcados de forma similar é remarcado, evitando assim que este tráfego não reconhecido receba o mesmo tratamento prioritário.

Num serviço SIP tradicional como foi descrito acima, os usuários são autenticados sempre em suas instituições de origem e o tráfego necessariamente passa pelos servidores de *proxy* das instituições dos usuários. Esta estrutura possui uma inerente ineficiência, quando um usuário em mobilidade interage com outros usuários localizados geograficamente mais próximos dele e todos conectados de forma distante da instituição origem do usuário em questão.

Um exemplo do comportamento ineficiente pode ser visto na Figura 1.1, que mostra o usuário (UA - *user agent*) B.spo.voip.br visitando o domínio rio.voip.br e interagindo com o usuário C.rio.voip.br, neste domínio. Além da validação e do registro do usuário B.spo.voip.br ocorrerem no domínio rio.voip.br, o fluxo de mídia entre os dois usuários faz uma rota longa passando pelo *proxy* de spo.voip.br. Se a condição da rede entre os domínios rio.voip.br e spo.voip.br não for adequada e estiver prejudicada, talvez, por vários fatores como taxa de erro na rede, atraso e variação de atraso elevados, é possível que a qualidade da ligação entre os usuários B.spo.voip.br e C.rio.voip.br seja ruim, apesar de ambos estarem, eventualmente, em uma mesma rede local, com todas as condições favoráveis para uma excelente comunicação de voz.

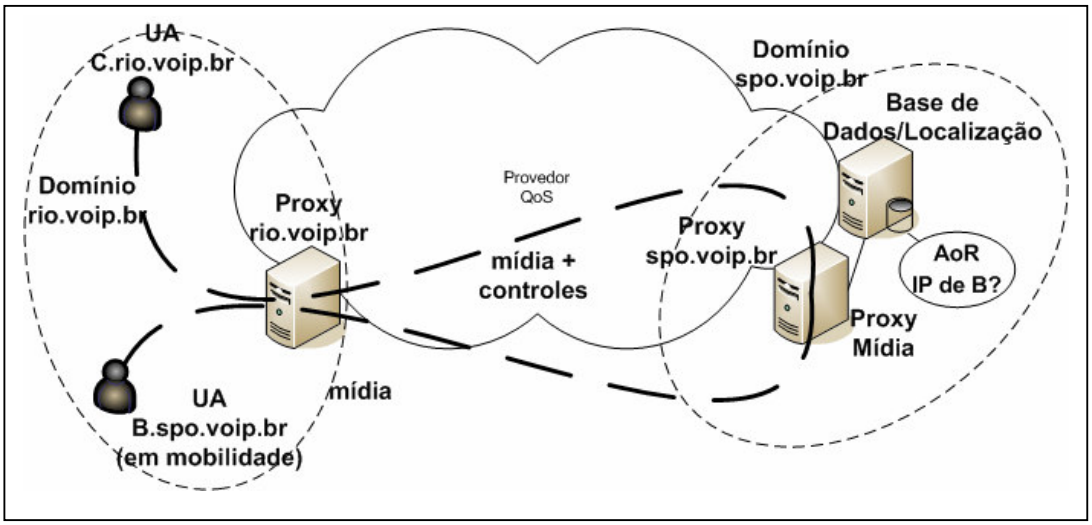


Figura 1.1 - Arquitetura SIP com proxies de mídia e sinalização

O problema de ineficiência mostrado na Figura 1.1 ocorre porque, numa arquitetura SIP tradicional operando com *proxies,* um usuário em mobilidade não é diferenciado de um usuário local, que esteja operacional na rede local de sua instituição de origem. Todo o processo de localização e direcionamento de mídia é feito com consultas à instituição de origem.

Estando o usuário em mobilidade, conectado em uma rede local em outra instituição, a sua comunicação será diretamente afetada pelas condições da rede entre a instituição de origem e sua localização física instantânea, além, obviamente, das condições da infraestrutura até o outro participante de uma chamada VoIP. Estando os dois participantes da chamada em mobilidade, a qualidade da chamada pode ser duplamente prejudicada pelo exposto acima, quando a mídia teria que passar sempre pelos *proxies* de mídia das instituições de origem.

Este trabalho apresenta uma solução de modelo de registro distribuído, que permite que o estabelecimento da mídia num diálogo SIP não precise passar pelo domínio de origem do usuário que estiver em mobilidade em uma instituição remota e, sim, pelo próprio servidor remoto, ou seja, aquele que está mais próximo do usuário/cliente, eliminando deste modo a ineficiência apresentada.

A Figura 1.2 mostra a situação desejável, com a mídia indo diretamente do usuário C.rio.voip.br ao usuário B.spo.voip.br, passando pelo *proxy* do domínio spo.voip.br. O usuário B.spo.voip.br deve se registrar no domínio rio.voip.br, mas a autenticação continua a ser feita no domínio de origem, onde se encontram as suas credenciais. Uma vez registrado e autenticado no domínio rio.voip.br, o usuário B.spo.voip.br poderá receber ou enviar requisições de estabelecimento de chamada (*INVITE*) através do *proxy* rio.voip.br, como se fosse de fato um usuário deste domínio.

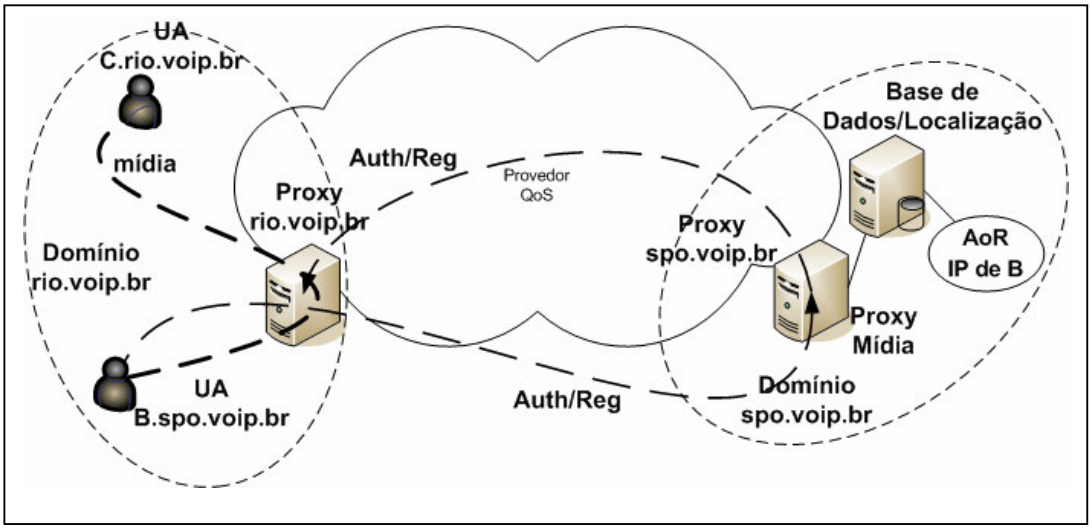


Figura 1.2 - Fluxos desejáveis em arquitetura SIP otimizada

A solução proposta evita qualquer alteração no software do cliente SIP e preserva o modelo de arquitetura tradicional, como exemplificado pelo serviço fone@RNP. Assumiu-se o conceito de federação, uma vez que instituições possuem um acordo de serviço entre si e trocam sinalização e informação relevante com seus parceiros, como é o caso hoje do próprio serviço fone@RNP. Neste serviço, solicitações de encaminhamento de chamadas sem restrição só são aceitas de parceiros conhecidos e identificados.

A fim de permitir que os usuários em mobilidade pudessem realizar ações tarifadas dentro da federação, foi incluída uma extensão do serviço que permite a transferência, para o *proxy* remoto, de informação de controle (como créditos para ligações [apendiceB], *vouchers* ou autorização para empréstimos em bibliotecas), por dentro da sinalização SIP. O domínio de um usuário em mobilidade é conhecido pelo URI (*Uniform Resource Identifier*), devendo existir uma base de confiança entre as instituições para realizar o procedimento de registro distribuído dos usuários na federação. Nota-se que uma das preocupações foi manter a sinalização padrão do protocolo SIP, garantindo a interoperabilidade.

1.2 – Trabalhos relacionados

No contexto em que se desenvolveu este trabalho, cuja área de interesse se volta para instituições que fazem uso de um serviço VoIP comum, não foram encontradas quaisquer publicações tratando da questão de otimização das rotas das chamadas para os usuários em mobilidade, que representassem uma melhora na qualidade do serviço prestado e reduzissem os recursos de infraestrutura da rede utilizados com o modelo convencional. Já, de acordo com a proposta deste estudo, os registros no serviço se darão nas localidades em que os usuários estiverem, garantindo a interoperabilidade com uma arquitetura sem modificações.

Alguns artigos, [SINGH, K. e SCHULZRINNE, H., 2005] e [BRYAN, D. A., LOWEKAMP, B. B., JENNINGS, C., 2005], abordam arquiteturas P2P para suporte ao usuário em mobilidade. Neste caso, as rotas seguidas pelas chamadas já são ótimas, devido ao processo de localização ser feito pela arquitetura P2P e as chamadas serem estabelecidas diretamente entre os clientes ou através de *proxies* próximos em que os clientes se registram. Este é o caso, por exemplo, da rede Skype [BASET, S. A., SCHULZRINNE, H., 2006], todavia nesses ambientes P2P o processo de registro e autenticação é centralizado, o que difere totalmente do cenário deste trabalho.

1.3 – Contribuição da dissertação

Esta dissertação apresenta uma solução otimizada para a comunicação em mobilidade de usuários pertencentes a uma federação de instituições. A federação é descrita como um conjunto de instituições que suportam os registros dos usuários vindos de outras instituições, além de permitir, eventualmente, que estes usuários acessem o serviço com base em permissões dadas pela instituição de origem do usuário.

A solução encaminhada aqui é única, inovadora e voltada para o protocolo SIP, visto que o SIP é o protocolo de sinalização padrão mais comum em VoIP, sobrepujando o H.323.

A solução proposta preserva a operação com uso de um cliente SIP sem modificação, e apenas estende a sinalização trocada entre os *proxies*

nos processos de registro e localização. Mesmo para os *proxies,* as extensões propostas são plenamente viáveis de serem implementadas e a viabilidade desta realização é demonstrada com a implementação para um ambiente com software livre baseado no *proxy* SIP OpenSER [IANCU, A. B., MIERLA, D. C., MODROIU, E. R., et al, 2005], um software bastante utilizado mundialmente.

1.4 – Organização da dissertação

Este trabalho está estruturado em seis capítulos. A introdução fornece uma visão geral do problema, detalhando a motivação como contribuição da dissertação. O capítulo 2 apresenta os conceitos básicos relacionados com os seguintes tópicos: SIP, VoIP e Federação. O capítulo 3 revê os principais tópicos do protocolo SIP, SIP com P2P e apresenta o software OpenSER. O capítulo 4 apresenta a solução proposta para apoio à mobilidade. O capítulo 5 apresenta os exemplos de operação e atendimento através de diagramas que demonstram as trocas das primitivas dentro dos cenários propostos. O capítulo 6 descreve o resultado da solução como conclusão e uma proposta de trabalhos futuros. Finalmente, os apêndices que descrevem as estruturas utilizadas como contribuição dentro dos cenários utilizados na proposta. O apêndice A, descreve a otimização do fluxo de mídia no modelo de arquitetura utilizado na prova de conceito. O apêndice B descreve o registro de um usuário em mobilidade, demonstrado com os logs envolvidos. O apêndice C descreve o código utilizado dentro do script de configuração do servidor SIP para validação do processo de encaminhamento das requisições de chamada.

Capítulo 2

Conceitos básicos

2.1 - VoIP (Voice over IP)

VoIP (*Voice over IP*) é uma tecnologia que vem crescendo na área de telecomunicações, visto que o seu aprimoramento é mais veloz que o da telefonia móvel.[HERSANT, O.; GUIDE, D., 2005] Fabricantes, operadoras e gerentes precisam se adaptar com urgência, pois a curva de aprendizado para gerência da tecnologia e seu aprimoramento é bastante íngreme e requer de fato uma estratégia educacional.

A tecnologia empregada em VoIP parece trivial, mas não o é . Em particular, ela é muito mais complexa do que a transmissão de TV ou de rádio na rede, porque a latência entre o locutor e o ouvinte deve permanecer muito baixa para garantir boa interatividade, ao passo que aplicações tipo *streaming* podem usar *buffers* muito grandes para compensar as grandes variações de atraso típicos da Internet.

O VoIP (*Voice over IP*) faz uso de sistemas hoje baseados na sua maioria em uma arquitetura cliente servidor, em que são necessários um serviço de localização de recurso e o estabelecimento de sessões multimídia entre usuários. Estes, por sua vez, utilizam um agente (UA – *User Agent*) que pode ser uma aplicação ou um *hardware*, um telefone IP. Estes UAs se conectam a um servidor *proxy* ou *gatekeeper,* dependendo do tipo de protocolo. E, através de um sistema de localização de domínios como DNS, é possível localizar o UA de destino, que possui um nome identificado como URN como um endereço de *email*, ou outro identificador que esteja registrado, para o estabelecimento da chamada.

Existem vários protocolos VoIP, sendo os principais o H.323, o SIP e o MGCP. O H.323 [Recommendation ITU-T, 1998] é um *framework* de recomendações muito utilizado na implementação de comunicação multimídia em redes de pacotes e tem sido padronizado pelo ITU-T. Existem muitas implementações H.323 no mercado, como o *Netmeeting* da Microsoft, *IP Phone* da

Intel e soluções desenvolvidas pela *PictureTel*. Como o H.323 foi o protocolo VoIP que se estabilizou primeiro, a partir de sua versão 2 em 1998 [H.323 – ITU-T Recommendation**,** 1998], ele foi utilizado preferencialmente em sistemas de videoconferência corporativa. O SIP foi desenvolvido pela IETF e sua estabilização somente aconteceu após a publicação da RFC 3261 [ROSENBERG, J.; SCHULZRINNE, H., CAMARILLO, G., et al., 2002], em junho de 2002. O SIP usa a mesma estrutura dos protocolos da Web, como *HTTP*, e tem muito poder de integração com a Internet atual. O SIP, diferentemente do H.323, coloca grande parte do controle no agente de usuário (UA – *User Agente*), embora ainda dependa de servidores *proxies* para funções principais. O uso do SIP é a tendência atual, principalmente depois que a Microsoft divulgou o seu uso no MSN Messenger, e a arquitetura NGN [FACCIN, S.M., 2004] definiu-o como elemento de comunicação. O uso do SIP também é padronizado nas arquiteturas IMS [M. Gomez, et al., 2005].

Finalmente o MGCP (*Media Gateway Control Protocol*) [ARANGO, M., et al., 1999], uma parceria IETF/ITU-T. O H.248 é o padrão ITU-T correspondente ao MGCP IETF, que trata especificamente de controlar, de forma centralizada, sessões entre equipamentos chamados Media Gateways, que são os softs-PBXs das redes IP. O MGCP é hoje de uso corrente nos *backbones* das operadoras.

2.2 Segurança em VoIP

VoIP é uma aplicação em rede e é vulnerável a ataques externos e internos. Sua autenticação pode ser feita nos próprios servidores VoIP ou via Radius, utilizando protocolos e algoritmos de criptografia conhecidos. Um destes protocolos de autenticação é o HTTP Digest [FRANKS, J., et al, 1999], que originalmente é um sistema baseado em *senhas* e opera sobre o paradigma de um protocolo de desafio/resposta, no qual se utiliza um valor *nonce* para compor um desafio para um destinatário. A resposta ao desafio inclui um *checksum* (por padrão, um *checksum* MD5) do nome de usuário, da senha, do valor do *nonce* fornecido, do método http e do URI que realizou a requisição. Neste procedimento a senha nunca é enviada em texto claro, garantindo sua confidencialidade.

Um protocolo de autenticação muito conhecido é o CHAP (*Challenge Handshake Authentication Protocol*) [SIMPSON, W;1996], como esquema de autenticação, assim como PAP (Password Authentication Protocol) [LLOYD,B. , SIMPSON, W., 1992] e EAP (Extensible Authentication Protocol) [SIMON, D., ABOBA, B., SIMON, D., ERONEN, P., 2008]. Eles são utilizados por um servidor RADIUS [RIGNEY, C., WILLENS, S., RUBENS, A. et al., 2000] na validação de um usuário. Este método está sendo substituído pelo uso de mecanismos externos para validação das credenciais dos usuários, como LDAP [WAHL, M.; KILLE, S. e HOWES,T., 1997], Kerberos [NEUMAN, C., HARTMAN, S., RAEBURN, K., 2005], SQL ou *Active Directory* [Microsoft AD, 2009]. Ainda podem ser utilizados certificados digitais [KENT, S., 1993] para o processo de segurança, bem como o uso de transporte seguro, pelo TLS (Transport Layer Security)[DIERKS, T.; RESCORLA, E., 2006];

O endereço lógico (IP) também pode ser utilizado para validar a autenticação. Em relação à privacidade, a mídia pode ser transmitida criptografada com uso de SRTP (RTP Seguro)[BAUGHER, M. , NASLUND, M. , CARRARA, E. , NORRMAN, K., 2004], ao nível da aplicação, ou com uso de VPN com túneis seguros, o que pode ser útil quando operando atrás de *firewalls* e NATs, uma vez que podem barrar a comunicação VoIP que faça uso dinâmico de portas para a mídia. O IPSec também é uma boa opção para transmissão da mídia de forma segura.

Nas redes internas (LANs) uma boa prática é a segmentação em VLANs (Virtual LANs), ou seja, uma para VoIP e outra VLAN para dados, para isolar a rede VoIP de ataques que possam ocorrer na rede de dados e, eventualmente, possibilitar melhor qualidade para voz, dando prioridade da VLAN de voz sobre a VLAN de dados.

2.2.1 - Ataques em VoIP

Como este assunto é abrangente e complexo, ele é abordado sobre 3 aspectos: tipos de ataques, formas básicas de segurança das comunicações e questões em relação a dispositivos específicos.

Uma forma simples de realizar um ataque *hijacking* é o registro de um usuário falso no serviço, mas com o identificador (URI) de um cliente legítimo. O resultado bem sucedido desse tipo de ataque é aquele em que o telefone registrado falso iria tocar da mesma forma que o telefone original, quando alguém chamasse o legítimo usuário do número. Qualquer tipo de autenticação iria ajudar a minimizar este tipo de violação.

Menos óbvios são certos ataques específicos em que o VoIP é suscetível e estes podem estar entre os mais perigosos. Pode-se citar, por exemplo, a alteração do campo CALLER-ID (identificador de chamadas), que serve para exibir o número de quem está chamando. Como o VoIP possui por parte do usuário uma habilidade para manipular o CALLER-ID, este pode ainda ser utilizado para ajudar a responder a uma chamada (do ponto de vista pessoal do usuário), mas não deve ser utilizado para determinar a identidade do chamador. [ENDLER, D., et al, 2008]. Neste caso específico, não existe uma "correção" real da ação a ser tomada, pois não é considerada uma falha. E até que os usuários tomem conhecimento desta e de outras questões semelhantes eles podem estar vulneráveis.

Tipo de ataque DoS pode ser o mais discutido. Ocorre quando um atacante tenta derrubar os recursos de um elemento da comunicação, tal como um *proxy* com requisições inválidas, impedindo assim a utilização legítima. Um ataque genérico DoS pode ser tão simples quanto um alto volume de requisições ICMP gerando uma inundação. Tais ataques IP, em geral são descobertos e respondidos por mecanismos de segurança que já estejam em vigor. Uma aplicação específica para ataques DoS, como a geração de um alto volume de *INVITE*s, precisa ser tratada diretamente por um elemento na aplicação do serviço. Alguns *proxies* tem lógica interna que identifica um ataque DoS comum e descarta os pacotes envolvidos. Ataques DoS podem ser minimizados, pois são bem conhecidos e se, no momento de um ataque, as

partes forem comunicadas, o conteúdo da chamada não será comprometido e nem será gerado um *overhead* de requisições.

Uma recomendação básica para segurança é garantir que a máquina que execute o *softphone* esteja tão protegida quanto a rede em que a transmissão é enviada.

O uso de VLAN específica para VoIP, colocando nesta VLAN apenas os telefones IP, pelo menos protege o VoIP dos ataques DoS (*Denial of Services*), da violação de segurança e da privacidade, originada geralmente na VLAN de dados. Os computadores atuais suportam IEEE 802.1q/p [Lidinsky, P.W., 1998], de modo que eles possam suportar VLAN diretamente, permitindo que uma aplicação de *softphone* possa fazer uso da VLAN de voz diretamente. Isso traz mais segurança para a aplicação de voz. Todavia, estendendo a VLAN de voz até os computadores, pode-se comprometer a segurança dos casos de invasão ou infração da máquina do usuário.

Um ataque do tipo MitM (*Man in the Middle*) ocorre, quando um fluxo de dados (sinalização ou mídia) é interceptado e retransmitido, ou seja, o sinal pode ser interceptado, armazenado e enviado a terceiros ou alterado, antes de voltar a ser transmitido ao destinatário pretendido inicialmente. Uma defesa efetiva contra ataques MitM é uma encriptação forte para ambos os fluxos, mídia e sinalização. O termo encriptação forte é o processo de converter uma informação comum (texto claro ou aberto), usando um algoritmo (chamado cifra) em algo não-inteligível, exceto para aqueles que possuam um conhecimento especial, geralmente referido como chave. Antes de aceitar o custo da criptografia fim-a-fim, deve-se pesar cuidadosamente o risco, que vem a ser a incerteza de um fato, o peso de uma ameaça em termos de impacto e probabilidade, como o risco da interceptação e as consequências em relação à supressão de silêncio e a geração de ruído de conforto.

Só algumas comunicações podem justificar uma proteção assim cara. Além disso, a criptografia é eficaz apenas contra a interceptação das comunicações em trânsito, mas não salvaguarda o *user agent* (terminal final) em ambas as partes de uma conversa. [SEEDORF, J.; 2006]. Algumas táticas podem ser aplicadas, como, por exemplo, a máquina que executa o *softphone* precisa estar protegida tão cuidadosamente quanto à rede em que a transmissão é enviada.

SIPS (SIP Seguro) é uma variação do SIP como um meio de criptografia de baixo custo, em que especifica o TLS (*Transport Layer Security*) descrito na RFC 2246, sobre a camada TCP para transporte de mídia de voz. Mas como as chamadas com uso do TLS podem ter falhas, na validação de certificados, o usuário deve manter o certificado válido para evitar este transtorno. Usando TLS, nem sempre a validação fim-a-fim entre os usuários é realizada de forma eficiente e isso irá gerar falha no uso do serviço. De acordo com alguns especialistas [SALSANO S.; VELTRI, L., PAPALILO, D., 2002] em SIP existem dois princípios básicos para fornecer segurança, fim-a-fim e salto-a-salto. Segurança fim-a-fim envolve os usuários finais, por exemplo, autenticação SIP, que tem por objetivo a validação das credenciais do usuário que desejar utilizar o serviço. E nas soluções salto-a-salto a confiança na segurança é fornecida pela rede, com uso de TLS (*Transport Level Security*) e/ou *Internet Protocol Security* (IPSec). [SALSANO S.; VELTRI, L., PAPALILO, D., 2002].

Outro tipo de ataque em sistemas VoIP, juntamente com a clonagem de um registro, seria tentar obter gratuitamente o serviço de um *gateway* SIP-PSTN. O *proxy* deve exigir que, para a realização de chamadas tarifadas, os usuários sejam validados por um serviço de autenticação e bilhetagem. O acesso por pessoas não autorizadas ao *script* de configuração do servidor *proxy* deve ser proibido a fim de evitar configurações maliciosas. Uma tática possível em ambientes com conexões de longa distância é tentar ignorar o *proxy* local e enviar o tráfego diretamente para o *gateway* [HANDLEY,M.; e JACOBSON,V., 1998], o que pode ser prevenido com ACL (*Access Control Lists*) nos *gateways*, garantindo que os *gateways* somente recebam encaminhamentos vindos do *proxy* e não de um usuário qualquer.

Os servidores de *proxy*, *redirect* e *registrar* compartilham vulnerabilidades comuns a qualquer dispositivo IP e esses servidores devem estar globalmente acessíveis, se o serviço de estabelecimento de chamadas for geograficamente ilimitado. Tecnicamente, podem ser colocados atrás de um *firewall* ou *gateway* de borda, mas a latência e o *jitter* impostos pela inserção de mais um hop (salto) na rede podem ser altamente indesejados. Servidores podem, contudo, executar *firewalls* na própria máquina, tal como *iptables* ou

ipchains, e tal uso é recomendado. Outra sugestão seria restringir as portas abertas para uso apenas do VoIP.[ROSENBERG, J.; et al,2002]

O uso de equipamentos de segurança de alta performance, aliados à rede de alta velocidade, permite desempenhos adequados sem comprometimento da qualidade.

Um modelo de defesa por autenticação básica é sempre possível nos servidores de *proxy* , *redirect* e *registrar,* envolvendo um segredo compartilhado entre um *user agent* (terminal final) e a base de dados utilizada por esses servidores. A melhor prática é a certeza de que a senha esteja armazenada de forma criptografada nos servidores, assim como nos *user agents*. Com isso, mesmo que um invasor tenha acesso *root* nos servidores, este não conseguirá ser capaz de clonar um registro. Para detectar este ataque, existem alguns aplicativos, que informam mudanças feitas em arquivos sensíveis, como o *Tripwire*[1]*.* Isto pode reduzir significativamente os danos, sendo recomendado para o uso no ambiente de servidores. *Tripwire* já é incluído em muitas distribuições linux, como debian, SuSe e RedHat, podendo ser obtido gratuitamente pela Internet.

Softphone VoIP são elementos mais vulneráveis em relação aos telefones IP (*hardphones IP*), pois estes últimos normalmente fazem parte de VLANS (Virtual LANs) dedicadas à voz, separadas das VLANS dos equipamentos de dados, o que é uma boa prática de implantação VoIP. Alguns telefones IP permitem acesso via *telnet* para uma fácil manutenção, gerando uma brecha na segurança. Tal recurso deve ser desativado ou seu uso deve ser minimizado, em situações em que a "escuta" seja possível. Uma senha diferente deve ser usada para cada terminal, a fim de limitar os danos no caso de uma interceptação da sessão telnet. Além disso, alguns telefones SIP podem ser vulneráveis a ataques DoS e falham em elevados níveis de tráfego *multicast* [ANDREASEN, F.; FOSTER, B., 2003]. Vale mencionar que existem UAs que possuem a característica de uso com criptografia, sem qualquer impacto no desempenho ou na qualidade, dado o poder computacional dos processadores atuais [HERSANT, O.; GUIDE, D., 2005].

1 Disponível em: <www.tripwire.com>, Acesso em: Nov/2008

2.3 – Federação

Numa federação, a gerência dos usuários fica por conta da instituição de origem. Quando um usuário visita uma outra instituição, o provedor dos recursos desta instituição visitada se preocupa apenas com o controle de acesso, sendo as permissões de uso repassadas pelo provedor da instituição federada de origem do usuário. É, então, necessário que ambas as instituições federadas confiem nas informações trocadas.

Geralmente isso é feito através do estabelecimento de políticas e regras. Esse estabelecimento de confiança conjunta é básico para existência de uma federação. Existem federações de âmbito nacional espalhadas pelo mundo: [Cantor, S.;2005]

- InCommon (EUA); [CANTOR, S.;2005]
- UK - *Access Management Federation for Education and Research* (Reino Unido); [CANTOR, S.;2005]
- CRU (França); [CANTOR, S.;2005]

O Shibboleth[2] é um projeto da Internet2/MACE, financiado pela IBM, que objetiva transferir, de maneira segura, atributos de usuário a partir do site de origem para o provedor de recursos. Shibboleth é um exemplo de tecnologia que implementa o gerenciamento de acesso e autorização por federação em relação a recursos web disponibilizados através de acessos via http.

A partir do conceito de federação, foi utilizada a mesma lógica em relação ao serviço VoIP no contexto dentro de um *backbone* entre instituições que se utilizam de um serviço SIP. Através da federação, será realizada a validação dos usuários em mobilidade do serviço para o registro do UA (*User Agent*) na instituição visitada após autenticação pela instituição de origem deste usuário e a liberação do acesso ao serviço com envio de créditos para uso do recurso disponibilizado na instituição visitada.

[2] Disponível em: <shibboleth.internet2.edu>, Acesso em: Nov/2008

Há um crescente interesse no compartilhamento de recursos entre instituições. No caso das universidades, muitas soluções empregam mecanismos simples de identificação, como o uso do endereço IP, para validação. Um sistema que permita a identificação do usuário com base no processo de autenticação, autorização e contabilização dos acessos, fornecendo compartilhamento de recursos entre as instituições com uso de SSO (*Single-Sign-On*) por seus usuários permitirá também o registro no serviço e, por sua vez, promoverá a otimização pretendida.

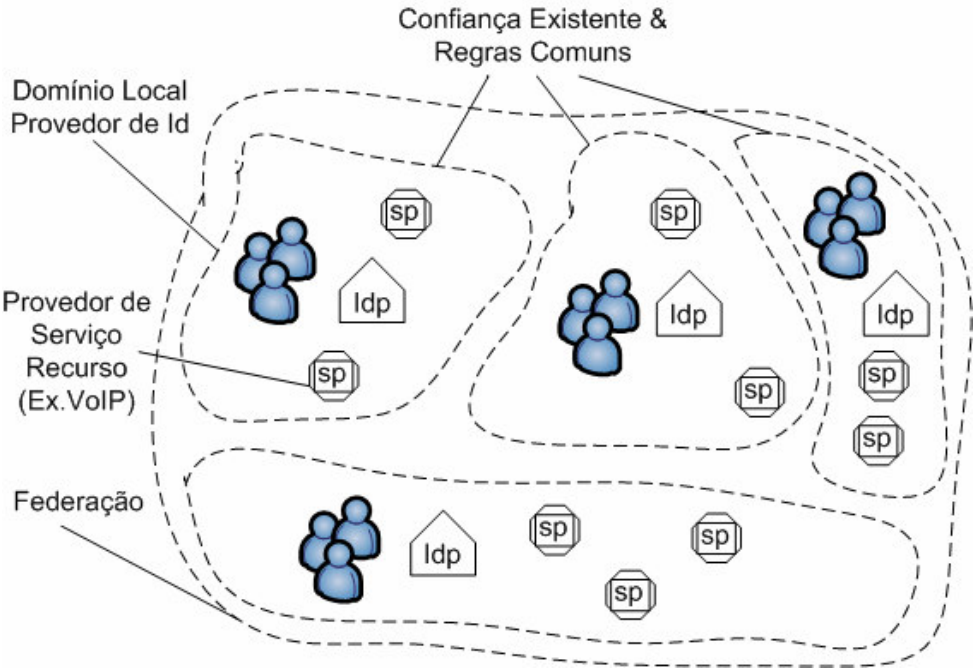


Figura 2.5 - Federação

Na Figura 2.5, o objetivo foi mostrar o ambiente de federação em que se consideram as instituições como domínios locais constituídos de um conjunto de usuários, de um provedor de serviço de recursos (sp), de um provedor de identidades (Idp). E que nessa topologia encontram-se as instituições seguindo regras comuns e determinando uma confiança entre as partes.

Capítulo 3

SIP (Session Initiation Protocol)

SIP é um protocolo baseado em texto e derivado do http e, assim como este, suporta o transporte de qualquer tipo de carga em seus pacotes, pelo uso de Mime-Types (Multipurpose Internet Mail Extensions). Voltado para o estabelecimento e controle de sessões multimídia, em geral com maior utilização na área de VoIP. Está descrito principalmente em duas RFCs, RFC2543 e RFC3261[, que também é conhecida como a versão 2 do SIP. SIP] faz uso do *Session Description Protocol* (SDP) para passar informações específicas sobre parâmetros de mídia (a voz digitalizada).

Um servidor *proxy* age como servidor por um lado (recebendo pedidos) e como cliente pelo outro lado (enviando pedidos). Um *proxy* pode passar adiante uma requisição sem nenhuma mudança para seu destino final, pode mudar alguns parâmetros antes de passar a requisição e pode até mesmo decidir enviar uma resposta gerada localmente.

De acordo com o site *http://www.internet2.edu/sip.edu,* algumas definições de quando e onde usar *proxies* SIP e os cenários em que são utilizados:

- Provedores de serviços VoIP;
- Entroncamento SIP;
- Balanceamento de carga SIP;
- SIP front-end (terminação SIP);
- Serviços empresariais;
- SIP router (LCR para multi GWs);

Um dispositivo final ativo do SIP é chamado de agente de usuário ou UA (*User Agent*). O propósito do protocolo SIP é estabelecer sessões de chamada entre os usuários. Como o próprio nome indica, um agente de usuário, ao realizar uma chamada, atua como um agente em nome de um usuário para a criação e para o encerramento de sessões de mídia.

3.1 – Métodos SIP

No protocolo SIP para estabelecimento e término das sessões são utilizadas requisições SIP, também conhecidas como métodos no protocolo. Todavia tais requisições solicitam uma ação específica a ser tomada por um UA (*user agent*) ou servidor. Os seis métodos originais no SIP são: *INVITE*, *REGISTER*, *BYE*, *ACK*, *CANCEL*, e *OPTIONS*. Os métodos *REFER*, *SUBSCRIBE*, *NOTIFY*, *MESSAGE*, *UPDATE*, *INFO* e *PRACK* foram acrescentados posteriormente e são descritos em RFCs separadas.

Nome do método	Comportamento
INVITE	Indica que o usuário está sendo convidado a participar de uma sessão multimídia. O corpo da mensagem pode conter uma descrição da sessão, utilizando-se o protocolo de descrição de sessão SDP (*Session Description Protocol*)[HANDLEY,M.; e JACOBSON,V., 1998].
ACK	Mensagem recebida como resposta final a um INVITE. A requisição ACK pode conter o SDP de descrição da sessão negociada entre ambos os clientes. Se não contiver o SDP, o usuário chamado pode assumir a descrição dada pelo primeiro INVITE, se houver.
OPTIONS	Faz uma pergunta sobre quais métodos e extensões são suportados pelo servidor e pelo usuário descrito no campo de cabeçalho <To:> . O servidor pode responder a esta pergunta com o conjunto de métodos e extensões suportado pelo usuário e por ele mesmo.
BYE	Usado para liberar os recursos associados a uma ligação e forçar a desconexão da mesma.
CANCEL	Cancela uma requisição que ainda esteja pendente, ou seja, em andamento. Uma requisição é considerada pendente, se e somente se, ela não foi atendida com uma resposta final.
REGISTER	Um cliente usa este método para registrar o "alias" (apelido) do seu endereço em algum servidor SIP, que, por aceitar registro de usuários, é chamado de serviço REGISTRAR.

Tabela 3.1.1 - Métodos da Sinalização SIP

A tabela 3.1.1 tem uma breve descrição de cada um dos métodos originais:

REFER	Usado por UA para requisitar de outro UA o acesso ao recurso URI ou URL. O recurso é identificado por um URI ou URL requerido no campo de cabeçalho *Refer-To*. Note que a URI ou URL podem ser de qualquer tipo de: sip, sips, http ou pres. Quando a URI for sip ou sips, o REFER estará sendo usado para implementar um serviço de chamada de transferência. O REFER também é usado para implementar um controle de chamada *peer-to-peer*.
SUBSCRIBE	Usado por um UA para estabelecer uma subscrição com o propósito de receber notificações (através do método NOTIFY) sobre um evento em particular. Uma subscrição com sucesso estabelece um diálogo entre o UA cliente e o servidor. O pedido de subscrição contém um *expires* (campo de cabeçalho), que indica a duração desejada desta subscrição.
NOTIFY	Utilizado por um UA para transportar uma informação sobre a ocorrência de um evento particular. Um *NOTIFY* é sempre enviado dentro de um diálogo, quando uma subscrição existe entre um assinante e o notificador.
MESSAGE	Usado para transportar mensagens instantâneas (IM) por meio do protocolo SIP. As mensagens instantâneas normalmente consistem de curtas mensagens trocadas próximas do tempo real por participantes em uma conversação.

UPDATE	Usado para modificar o estado de uma sessão, sem alterar o estado de um diálogo. Um uso possível inclui o *muting* ou colocar em espera o diálogo, suspendendo a mídia a fim de realizar um QoS ou outra negociação de atributo fim-a-fim antes do estabelecimento da sessão.
INFO	Utilizado por um UA para enviar informação de sinalização de chamada para outro UA com a qual se tenha uma sessão de mídia estabelecida. E tipicamente contém um corpo na mensagem cujo conteúdo pode ser informação de sinalização, um evento *midcall* do tipo sinalização de PSTN.
PRACK	Ë usado para receber uma confirmação (*ACK*) por respostas provisionais (1xx) de transporte confiáveis. Como exemplo, a resposta 180 Ringing pode ser critica para determinar o estado da chamada, sendo necessário uma confirmação desta resposta provisional.

Tabela 3.1.2 - Métodos adicionais da sinalização SIP

A Tabela 3.1.2 define os métodos adicionais da sinalização SIP.

Para cada requisição ou resposta, temos um grupo de cabeçalhos, divididos em: cabeçalhos gerais, com informações importantes sobre a chamada; cabeçalhos de entidade, com meta-informação sobre o corpo da mensagem; e os cabeçalhos específicos, que permitem passar informações adicionais, que não couberam na linha de *status* da requisição ou da resposta.

Quando requisições são atendidas, as respostas enviadas são identificadas por números, sendo que o primeiro dígito representa a classe de resposta desse número. São enviadas diversas mensagens provisórias antes de se enviar uma resposta definitiva. Existem seis classes possíveis de resposta: Classe 1XX, respostas temporárias ou informativas; Classe 2XX, resposta final de sucesso; Classe 3XX, redirecionamento da requisição; Classe 4XX, erros no cliente; Classe 5XX, erros do servidor; e Classe 6XX, erros globais na rede.

3.2 – Encaminhamento de chamada com autenticação

Existem duas entidades SIP relevantes para a compreensão da proposta do trabalho. A primeira é o *proxy registrar* que, nas redes SIP tradicionais, tem como objetivo resolver o *Address of Record* (AoR) para o endereço IP corrente (Contato URI) do usuário. Essa função é normalmente executada em modelo cliente-servidor, sendo o processo de localização do recurso de destino realizado por DNS, de forma centralizada. O endereço IP de um usuário pode variar, sob algumas circunstâncias, pelo uso de DHCP em rede local, ou em serviço de ISP (*Internet Service Provider*) ou em mobilidade. Os *registrars* são servidores necessários para manter a localização atual de um usuário, mapeando seu endereço SIP num endereço lógico IP. Um *proxy* registrar é um servidor que aceita pedidos *Register*, embora possa desempenhar outras funções SIP (como um *proxy*). O *proxy* de mídia é responsável em gerenciar a mídia passante pelo *proxy* SIP e o de sinalização é responsável pelo controle das requisições e respostas.

A sinalização de resposta pode ser sempre forçada a passar pelo servidor *proxy* do domínio local através da inserção de um campo *record-route()* nas primitivas processadas pelo próprio *proxy*. Utilizando uma extensão ao *proxy*, denominada mídia-router, um cliente em sua interação com outros clientes poderá ter sempre, além da sinalização, a mídia sendo roteada forçosamente por esses *proxies*, independente de sua localização física.

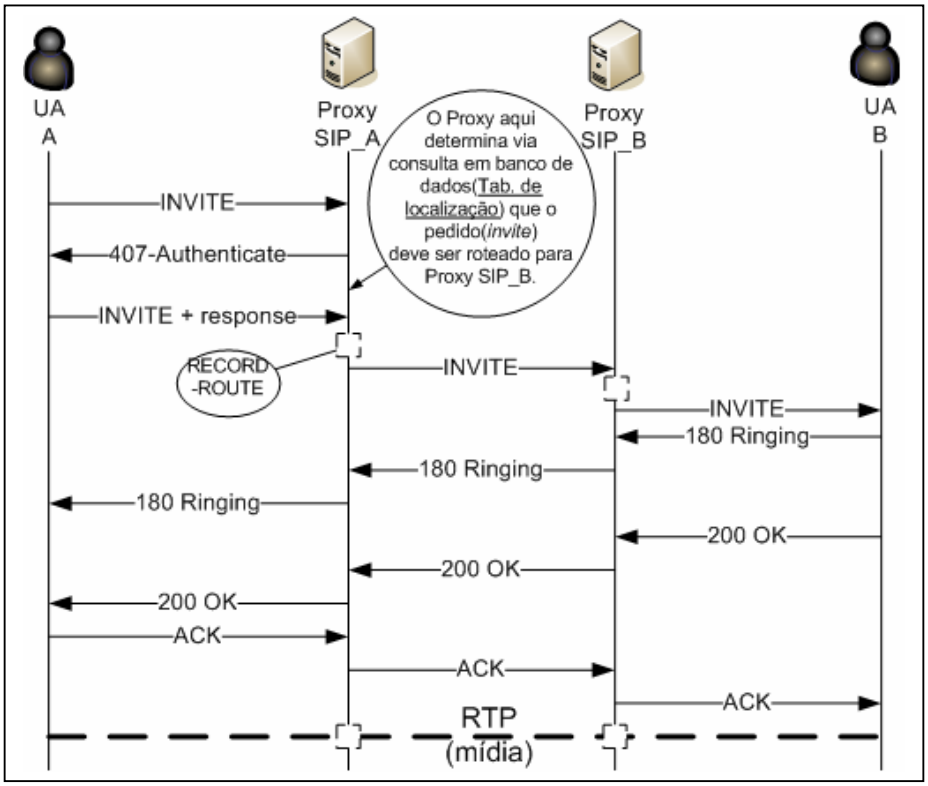


Figura 3.2.1 – Servidor Proxy

Para ilustrar o processo de encaminhamento de chamadas com autenticação e para validação do processo, a Figura 3.2.1 mostra uma chamada SIP entre dois UAs envolvendo dois servidores *proxies*. Ao solicitar uma chamada para um UA remoto através do SIP URI, o SIP fone envia uma requisição de chamada *INVITE* para o servidor *proxy* para resolução de endereços. O servidor *proxy* requer autenticação para realizar este serviço e responde com um código 407 *Proxy Authentication Required,* contendo um campo de cabeçalho *Proxy-Authenticate* com a forma de desafio . Após enviar um *ACK* para o 407, o UA poderá então reenviar o *INVITE* com o campo de cabeçalho *Proxy-Authorization* contendo as credenciais. Usando um *nonce* a partir do desafio, o chamador reenvia a requisição *INVITE* com as credenciais criptografadas a partir do nome de usuário e senha do chamador. O servidor *proxy* verifica as credenciais e, estando corretas, realiza a busca DNS sobre a requisição do URI. O *INVITE* é então encaminhado para o servidor *proxy* listado no registro DNS SRV do domínio correspondente deste serviço. O *proxy*, então, através da procura na requisição URI localiza o registro da parte chamada. O *INVITE* é encaminhado ao destino UAS (*User Agent Server) e*,

se houver um cabeçalho denominado *RECORD-ROUTE* inserido, servirá para garantir que o *proxy* estará presente nas futuras requisições por qualquer das partes. Isso porque normalmente uma mensagem SIP roteada de forma direta pode ser bloqueada por um *firewall*.

A Figura 3.2.1 mostra o *proxy* "A" repassando um *INVITE* (pedido de chamada) do usuário A para o usuário B com uso de autenticação *Digest*. A resposta 180 *Ringing* é usada para indicar que um *INVITE* foi recebido pelo usuário B e que agora está sendo alertado, sendo importante para interagir com os protocolos de telefonia, além disso, é que o corpo desta mensagem permite carregar QoS ou informação de segurança. A resposta 200 *OK* confirma a aceitação da chamada pelo usuário B. O *ACK* confirma o fim do diálogo SIP entre A e B. Após o recebimento do *ACK* a troca de mídia é estabelecida.

A segunda entidade importante no contexto do trabalho é o servidor de redirecionamento, um servidor SIP que,embora responda a requisições, não realiza encaminhamento de pedido. Como um servidor *proxy*, um servidor de redirecionamento usa um banco de dados ou um serviço de localização (exemplo do modelo SIP P2P, baseado em DHT (*Distributed Hash Table*)) [SINGH, K.; e SCHULZRINNE, H., 2006]. A informação de localização, entretanto, é enviada de volta para quem realiza a chamada dentro de uma classe de resposta (3xx) de redireção. Logo após o ACK, é concluída a transação. A Figura 3.2.2 mostra o fluxo de chamadas em que se usa o servidor de redireção, ao invés do servidor *proxy* para assistir a localização do UA B.

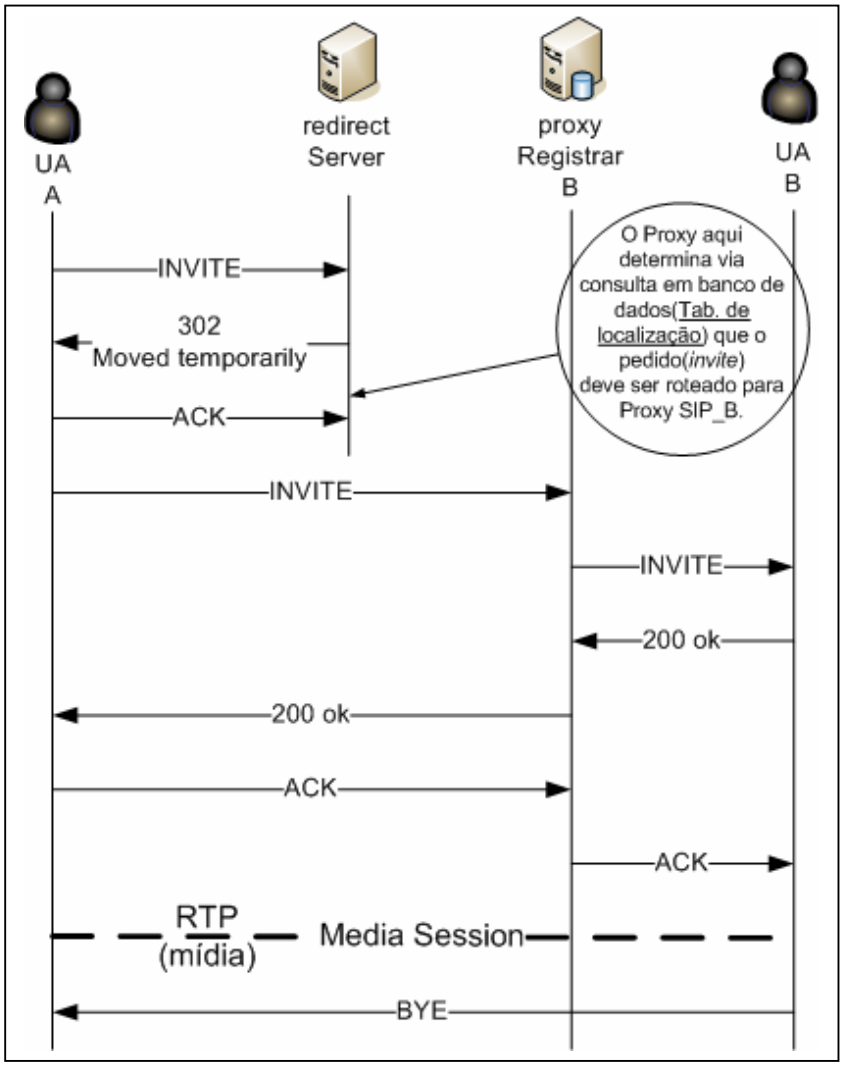


Figura 3.2.2 – Servidor de redirecionamento

O *proxy* não precisa entender um método de requisição para encaminhar um pedido. Um *proxy* trata um método desconhecido como se fosse uma *OPTIONS* e encaminha o pedido para o destino, se for possível (melhor esforço). Isso permite introduzir novas características e métodos úteis para os agentes de usuários ou UA (*user agents*), sem que se exija apoio dos *proxies* que estejam no meio do caminho. No caso de um UA receber um método sem suporte, responderá com um *501 Not Implemented*.

O SIP possui alguns aspectos relacionados ao estabelecimento e término das sessões multimídia, como localização dos usuários em que são usadas bases de dados locais ou servidores LDAP (*Ligtweigth Directory Ac-*

cess Protocol) [WAHL, M.; KILLE, S. e HOWES,T, 1997], para montar diretórios de usuários e seus perfis.

3.3 – Autenticação/ registro centralizado

O uso do serviço LDAP ou outra base de dados para troca de informações entre diferentes instituições esbarra no problema da autenticação distribuída de seus clientes, pois cada instituição constitui um domínio administrativo próprio e seu servidor pode autenticar apenas os usuários cadastrados naquele domínio.

Em mobilidade, o usuário que estiver em alguma localização remota na rede, terá que continuar a usar necessariamente o servidor local de sua instituição para se registrar. Tipicamente empregando um servidor de registro para cada domínio, os UA`s (*user agents*) ou *softphones* dos usuários dentro do domínio registram seus endereços IP`s com o servidor, permitindo que outros usuários possam se comunicar com eles.

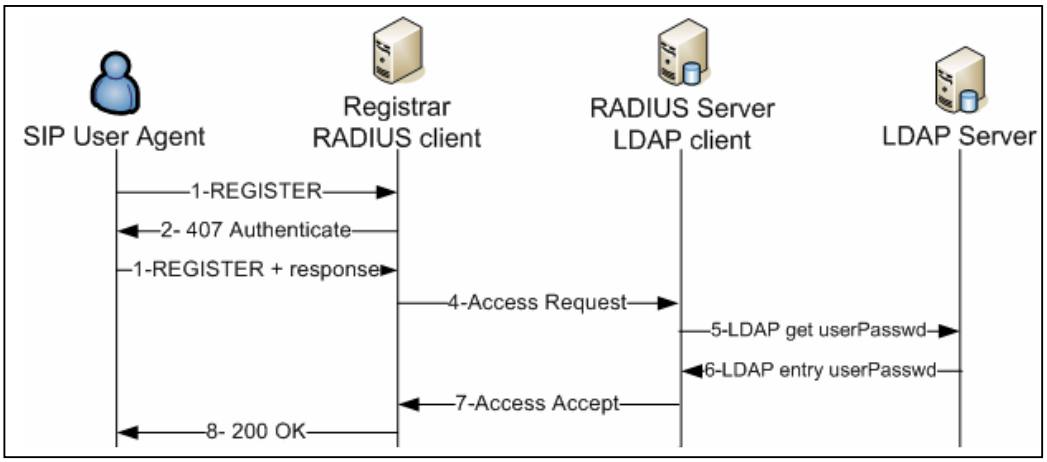


Figura 3.3.1 – Registro/Autenticação Cliente SIP

Na Figura 3.3.1 é demonstrado o registro de um UA em seu domínio e sua autenticação. Com uso de uma instância do RADIUS [RIGNEY, C., WILLENS, S., RUBENS, A. et al., 2000], a parte cliente, no servidor registrar, realiza uma requisição de acesso. Nesta requisição constam as credenciais de acesso, na forma de usuário e senha ou um certificado de segurança fornecido por um usuário. O RADIUS Server verifica se esta informação está correta, consultando uma base de dados através do LDAP [ZEILENGA, K., 2006].

O UA, utilizando um *softphone*, com objetivo de realizar registro em uma base de dados, deverá ter configurado o parâmetro de *outbound proxy* para o endereço IP ou nome do servidor registrar que por sua vez terá uma instância cliente do *RADIUS* em execução, operando como cliente RADIUS. O servidor *registrar* verificará que na requisição do registro não constam as credenciais do UA, logo será respondido ao UA, utilizando o bloco de roteamento de respostas, um desafio, solicitando suas credenciais. Isso está definido na RFC 2617. A composição da resposta dada pelo servidor registrar ao UA, de acordo com a RFC 2617, está definida como *401 Unauthorized* com o campo de cabeçalho *WWW-Authenticate* sendo composta de:

challenge = *digest-challenge*;

digest-challenge = #(*[domain]* | *nonce* | *[opaque]* | *[stale]* | *[algorithm]*)

Os significados das diretivas acima são:

domain relativo à URL do servidor sendo acessado;

nonce relativo a uma *string* de dados unicamente gerado a cada vez que uma resposta 401 é enviada. Construída com codificação base 64.

Time-stamp H(time-stamp ":" ETag ":" private-key), sendo *time-stamp* o registro de hora do servidor; ETag. o *Entity-Tag*, ou seja,o campo de cabeçalho do http da entidade que realiza a requisição, e a *private-key,* um dado apenas conhecido pelo servidor.

Com o *nonce* desta forma, o servidor recalcula a porção *hash(H),* após receber o cabeçalho de autenticação do cliente e rejeita a requisição, caso o *nonce* não corresponder com o do cabeçalho enviado no *401-Unauthorized* ou se o valor do *time-stamp* não for recente o suficiente. Desta forma, existe um limite para o tempo de validade do *nonce*. *Stale* é um *flag* booleano (verdadeiro ou falso) que informa se o *nonce* estava obsoleto. *Algorithm* é uma *string* que indica o par de algoritmos usado para produzir o *digest* e o *checksum* e, caso não esteja presente, é assumido o MD5.

O UA apresentará novamente a requisição *REGISTER*, agora com suas credenciais. O servidor *registrar,* por sua vez, realizará o encaminhamento ao servidor de autenticação *RADIUS* que validará as credenciais, consultando um servidor LDAP, conforme descrição da Figura 3.3.1 acima. Porém faltou informar neste processo que o LDAP armazena a senha do usuário em texto criptografado pelo RADIUS que utiliza um segredo compartilhado jun-

tamente com um algoritmo de *hash* MD5 para ofuscar as senhas. Na consulta do servidor RADIUS ao banco LDAP, as credenciais são decodificadas para serem comparadas com o *hash* das credenciais recebido do servidor *proxy registrar*.

Em relação a um usuário em mobilidade, o procedimento de autenticação e registro se dá remotamente e o uso de *proxy* de mídia força todo o tráfego a passar pelo domínio remoto. Para um usuário em mobilidade em uma instituição federada, o estabelecimento dos canais de mídia de forma otimizada vai requerer que este usuário possa receber ou realizar chamada utilizando uma rota ótima com outro usuário, ou seja, um diálogo. Para isso, o registro deste usuário deve ser na instituição federada visitada e o *proxy* da instituição deve atuar como *proxy* de mídia, frente ao serviço VoIP, em relação às chamadas do usuário visitante. Com isso, a rota da mídia irá passar pelo *proxy* que geograficamente é o mais indicado, por estar mais perto do UA visitante.

Numa arquitetura SIP convencional, o processo otimizado não é possível porque falta um procedimento de registro dinâmico com base na validação do usuário e um processo para autenticação no servidor mais próximo, que consulta a base de dados remota, que é onde se encontram as informações referentes a este usuário.

3.4 – SIP com P2P

Foi verificado um grande interesse atual na área de convergência do VoIP com P2P. Uma característica comum aos serviços que utilizam a solução do P2P de forma estruturada como Skype, MSN, SIPPeer [SINGH, K. e SCHULZRINNE, H., 2006] é que possuem um único ponto de registro e autenticação para todos os clientes. Isso difere totalmente de um serviço SIP convencional para uso de instituições corporativas, nas quais a administração dos usuários e a validação das credenciais são como o fone@RNP, realizadas de forma totalmente distribuída, por cada instituição.

O uso do P2P com VoIP foi motivado pelo surgimento do Skype, desenvolvido pelo mesmo grupo de criadores do KaZaa. Duas abordagens que

combinam SIP e P2P surgiram com enfoque de uso não proprietário. Na primeira abordagem, o processo de localização via DNS é substituído pelo serviço de localização do protocolo P2P, sendo chamado de *SIP-using-P2P* [SINGH, K. e SCHULZRINNE, H.; 2006]. Neste caso, o P2P é usado apenas para localização e atualização (*lookups and updates*) dos endereços IPs dos usuários SIP, similares ao serviço LDAP e banco de dados SQL usados nos *proxies* SIP existentes, através de mapeamentos via DHT (*Distributed Hash Table*). Em se tratando de DHT, as pesquisas nessa área foram motivadas originalmente, em parte, pelos sistemas *P2P,* tais como Napster[3], Gnutella[4] e Freenet[5], que aproveitam os recursos distribuídos através da Internet para fornecer uma aplicação útil, como um serviço de compartilhamento de arquivos.

Na segunda abordagem, é usado o protocolo P2P sobre o SIP (*P2P-over-SIP*) [SINGH, K. e SCHULZRINNE, H.; 2006]. Neste caso, a manutenção do protocolo P2P pode ainda apresentar dois modos: tunelar o protocolo P2P em mensagens SIP, por exemplo, no corpo das mensagens ou no cabeçalho, ou reusar a semântica de algumas das mensagens SIP como cabeçalhos para transportar informação de proximidade e localização. [SINGH, K. e SCHULZRINNE, H., 2006].

O uso de P2P com registro centralizado acaba visando ao uso do VoIP como elemento individual de comunicação, e não como solução para um serviço corporativo em rede, onde os registros são descentralizados.

O processo de localização P2P usa, em geral, uma DHT que tem por objetivo mapear unicamente qualquer item a um dos nós da rede, além de gerenciar a remoção e inserção de itens. Tabelas *hash* distribuídas aumentam a capacidade e a disponibilidade, participando o espaço de chaves entre um grupo de *peers* e replicando os dados armazenados. O uso de DHT favorece as características de descentralização, tolerância à falhas e escalabilidade.

Em relação à VoIP, o recurso a ser localizado é o usuário. No DHT, o nome do usuário fica associado ao seu endereço IP e um *hash* do nome é

[3] Acesso: <http://free.napster.com> disponível em nov/2008.
[4] Acesso:<http://www.gnutellaforums.com> disponível em nov/2008.
[5] Acesso:<http://freenetproject.org > disponível em nov/2008.

inserido no sistema, funcionando simplesmente como um mecanismo de localização.

A solução do registro distribuído para um usuário em mobilidade em um ambiente de federação, pode ser independente do processo específico de localização do URI, seja via P2P ou via DNS. De fato, nesta dissertação, a proposta é validada com uso de DNS, mas considerações sobre o uso de DHT/P2P também são feitas, ao final do capítulo 5, item 5.6.

3.5 – OpenSER

O SIP Express Router (SER) [MIERLA, D., IANCU, B., HOFMANN, A., 2001] é um servidor de voz sobre IP gratuito, baseado no protocolo SIP (*Session Initiation Protocol*, RFC3261) e voltado a aplicações de grande volume. Ele foi criado para atender infraestruturas de voz sobre IP de larga escala. O servidor mantém o registro dos usuários, configura as sessões VoIP, encaminha mensagens instantâneas e cria espaço para novas aplicações. Sua interoperabilidade comprovada garante integração simples com componentes de outros fornecedores. Isto elimina a possibilidade de ficar travado em um único fabricante.

O OpenSER [IANCU, B., MIERLA, D. C., MODROIU, 2005] é a implementação aberta do SER e é a solução de *proxy* adotada no serviço fone@RNP. O OpenSER tem um modelo flexível de *plug-ins* para novas aplicações. Terceiros podem facilmente ativar seus próprios *plug-ins* com o código do servidor e prover deste modo serviços avançados. Seu arquivo de configuração (openser.cfg) combina configurações estáticas como também provê um ambiente de programação dinâmica. Desta forma, *plug-ins,* tais como contabilização usando o protocolo RADIUS, *gateways* de SMS, *queries* ENUM [FALSTROM, P., 2000], ou agente de presença, já foram desenvolvidos e são fornecidos como recursos avançados. Outros módulos estão a caminho: controle de *firewall* , *postgres*, *drivers* de LDAP .[6]

3.5.1 – SER vs OpenSER

O SER é mantido pela IPTEL (www.iptel.org), que saiu da companhia nacional de pesquisa alemã Fhg Focus. O site da IPTEL é a fonte primária de informações sobre o SER. Houve uma ramificação chamada OpenSER[7] (www.openser.org), O seu grupo de desenvolvimento se encontra mais ativo e novas versões estão sendo lançadas. Muito da documentação do SER data

[6] Disponível em: < http://www.voip-info.org/wiki/view/openser#Features> , Acesso em: Novembro/2007.
[7] Disponível agora como Kamailio em: <http://www.voip-info.org/wiki/view/Kamailio#Features > , Acesso em: Março/2009

de 2003 e não tem sido frequentemente atualizada, depois que a empresa foi comprada pela Tekelec.

Baseado nos últimos padrões do SIP, o OpenSER inclui suporte para o servidor de registro, de proxy e de redirecionamento (Registrar, SIP Proxy e SIP *Redirect Mode*). Além disso, ele atua como um servidor de aplicações com suporte para *instant messaging* e presença incluindo *gateway* 2G/SMS e *Jabber Gateway* , uma linguagem de políticas de controle de chamadas, tradução de números de chamada, planos de discagem provados, ENUM, AAA (*Authentication, Authorization, Accounting*). OpenSER roda nas principais vertentes do Linux, Solaris e suporta ambos o Ipv6 e Ipv4. É possível manter múltiplos domínios e redundância da base de dados é suportada também.

O OpenSER foi criado com os seguintes objetivos:

- **Velocidade**

  Com o OpenSER, milhares de chamadas por segundo podem ser obtidas, mesmo em plataformas de baixo custo. Esta velocidade permite que redes sejam configuradas com um custo baixo e de simples gerenciamento, e foi obtida com uso de um código customizado em ANSI C combinado com instruções em *Assembler* e com as últimas melhorias do SIP.

- **Flexibilidade**

  O OpenSER permite que seus usuários definam o seu comportamento. Administradores podem escrever scripts em texto que determinam as decisões de roteamento SIP, o principal trabalho de um servidor *proxy*.
  S*cripts* podem ser usados para configurar numerosos parâmetros e introduzir uma lógica adicional. Por exemplo, os *scripts* podem determinar para que destinos *record routing* devem ser feitos, quem será autenticado, que transações devem ser processadas com estado e que pedidos devem ser processados pelo *proxy* ou redirecionados.

- **Possibilidade de crescimento**

  O OpenSER pode ser estendido adicionando (*linking*) novos códigos em “C”. O novo código pode ser desenvolvido de forma independente do núcleo do OpenSER e ser interligado (*linking*) em tempo de execução. O conceito é similar ao do servidor WEB apache.

- **Portabilidade**

  Por ser escrito em ANSI C, ele tem sido testado em PC/LINUX e SOLARIS. Versões para BSD e IPAQ/LINUX também existem.

- **Interoperabilidade**

  OpenSER é baseado no padrão SIP. Ele passou por testes extensivos com produtos de outros fornecedores ambos nos laboratórios da IPTEL e no SIP Interoperability Tests (SIPIT).

- **Tamanho pequeno**

  O núcleo do OpenSER é de 300KB, mas, com alguns módulos adicionais, chega até 630KB.

O projeto do OpenSER foi desmembrado e criaram-se dois novos projetos: um denominado OpenSIPS [Bogdan, I., 2007] e o outro, Kamailio [Mierla, D., 2007] com as mesmas características. Porém não foram adotados na prova de conceito por uma questão de retrabalho em recompilar todo o código fonte do serviço e instalá-lo, apenas para troca do nome do serviço.

O projeto Kamailio é a continuação do projeto OpenSER, herdando deste o espírito de abertura à comunidade e a vontade de progresso com a continuidade do trabalho de desenvolvimento, estendendo o código e mantendo a visão do OpenSER através de forte processo de consolidação, o que é crítico para garantir o melhor esforço e os melhores resultados dentro do projeto. Tal ambiente consolidado é obrigatório para projetos de grande escala a fim de ajudar a distribuir soluções profissionais para indústria.

Consiste em uma diversidade consolidada, pois o Kamailio é um projeto de padrão aberto não apenas com relação a licenças de uso, mas também como sua política em relação ao seu uso considerando as contribuições, cooperações e a comunidade envolvida. A diversidade vem a partir de grande

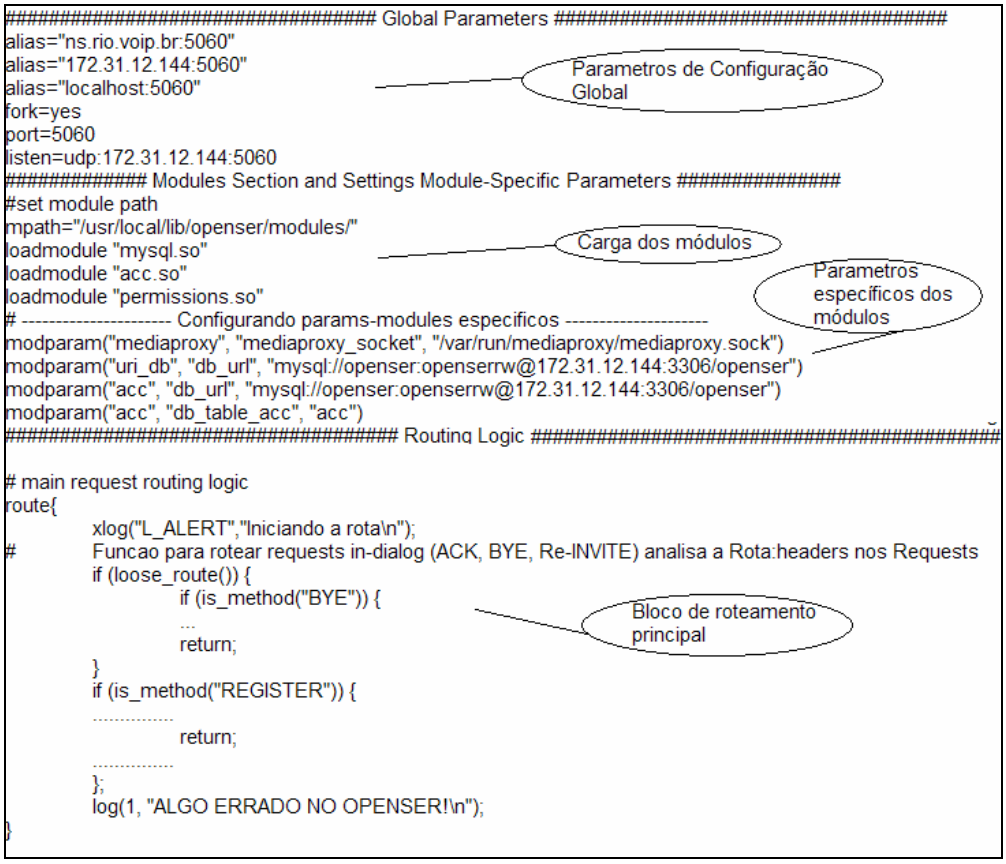
número de pessoas envolvidas no desenvolvimento do projeto, sua complexidade e na riqueza de características e funcionalidades disponíveis. Logo, para atingir um alto nível de confiança e estabilidade, toda essa diversidade precisa ser consolidada, através da visão do gerenciamento do projeto, sobre o seu design e código, o esforço do trabalho e o futuro do projeto.

3.5.2 – Arquitetura do OpenSER

Seu núcleo é responsável pela funcionalidade básica e administração das mensagens SIP, sendo que a maioria das funcionalidades é gerenciada a partir dos seus módulos. Tais módulos são usados dentro do arquivo *openser.cfg*, conforme descrito nas Figuras 3.5.2.1, 3.5.2.3 e 3.5.2.4. O código do *script openser.cfg* controla quais módulos são carregados e permite configurar parâmetros que regulam o funcionamento dos módulos, sendo o principal arquivo de configuração do OpenSER.

Possui sete seções:

- Definições globais: São o endereço IP e a porta que ele deve ouvir;
- Módulos: São a lista de bibliotecas externas que são necessárias para expor as funcionalidades que não estão disponíveis no núcleo;
- Configuração dos módulos: É o comando modparam ("nome do módulo", "parâmetro do módulo", "valor do parâmetro") que serve para passar adequadamente os parâmetros de vários módulos.
- Bloco de roteamento principal: É onde começa o processamento das mensagens em SIP, e serve para controlar como cada mensagem recebida é processada;
- Bloco de roteamento secundário: Este comando *(route*) serve para desviar a sequência.
- Bloco de roteamento de respostas: É usado para processar os *replies*, como o (200 ok);
- Bloco de roteamento de falhas: Serve para processar condições de falha como ocupado e *timeout*;




```
##################################### Global Parameters ###################################
alias="ns.rio.voip.br:5060"
alias="172.31.12.144:5060"
alias="localhost:5060"
fork=yes
port=5060
listen=udp:172.31.12.144:5060
############## Modules Section and Settings Module-Specific Parameters ###############
#set module path
mpath="/usr/local/lib/openser/modules/"
loadmodule "mysql.so"
loadmodule "acc.so"
loadmodule "permissions.so"
# ---------------------- Configurando params-modules especificos ----------------------
modparam("mediaproxy", "mediaproxy_socket", "/var/run/mediaproxy/mediaproxy.sock")
modparam("uri_db", "db_url", "mysql://openser:openserrw@172.31.12.144:3306/openser")
modparam("acc", "db_url", "mysql://openser:openserrw@172.31.12.144:3306/openser")
modparam("acc", "db_table_acc", "acc")
##################################### Routing Logic ########################################

# main request routing logic
route{
        xlog("L_ALERT","Iniciando a rota\n");
#       Funcao para rotear requests in-dialog (ACK, BYE, Re-INVITE) analisa a Rota:headers nos Requests
        if (loose_route()) {
                if (is_method("BYE")) {
                ...
                return;
        }
        if (is_method("REGISTER")) {
        ..............
                return;
        ..............
        };
        log(1, "ALGO ERRADO NO OPENSER!\n");
}
```

Figura 3.5.2.1 – Arquivo de Configuração – openser.cfg

Com o objetivo de mostrar de fato as seções do script openser.cfg, três figuras são apresentadas. A Figura 3.5.2.1 descreve a parametrização global, com definições gerais como a porta que o serviço irá escutar, além do *socket* utilizado com o protocolo de transporte UDP. Também é descrito nesta Figura 3.5.2.1 a carga dos módulos, como do SGBD (Sistema Gerenciador de Banco de Dados) MySQL, o módulo “ACC” para realizar o *accounting*, relacionados com os diálogos e transações do SIP. São passados os parâmetros específicos dos módulos utilizados, como a *string* de conexão com o banco de dados, no caso, MySQL. A última seção descrita nesta figura é a do bloco de roteamento principal, pelo qual toda e qualquer requisição para estabelecimento, seja de uma transação ou diálogo, deverá passar.

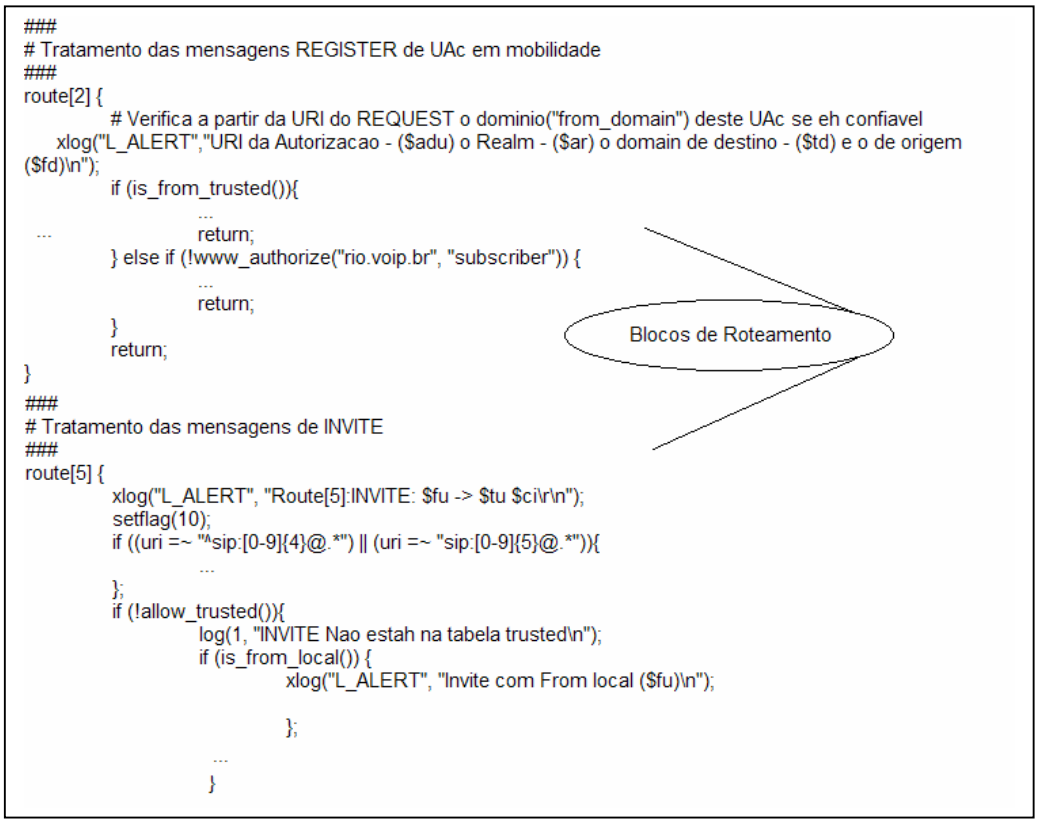


Figura 3.5.2.2 – Arquivo de configuração – openser.cfg

A Figura 3.5.2.2 mostra o detalhamento dos blocos de roteamento. Em especial, neste caso, é demonstrado o tratamento dado pelo serviço Open-SER às requisições de registro dos usuários e de encaminhamento de chamadas.

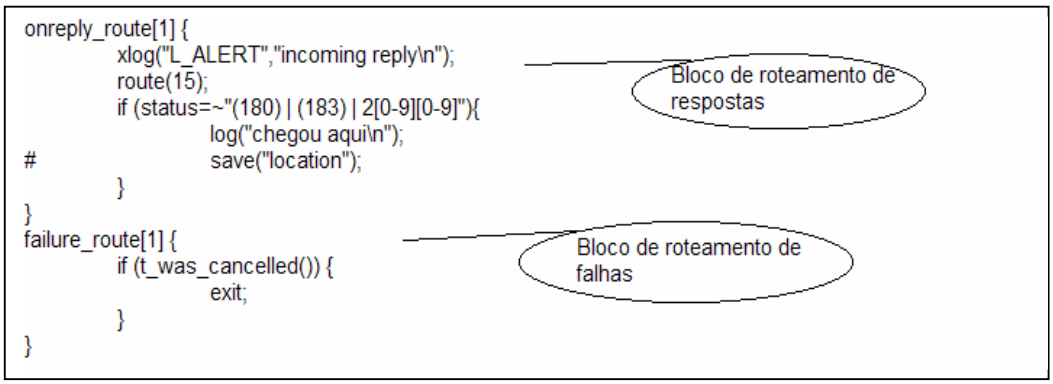


Figura 3.5.2.3 – Arquivo de configuração – openser.cfg

A Figura 3.5.2.3 mostra a parte do *script* que trata das respostas pelo servidor *proxy* nas requisições SIP, sendo que nesta figura é mostrado apenas um subconjunto de todo o tratamento dado nas respostas das requisições através do bloco de roteamento de respostas e de falhas.

Os módulos do OpenSER expõe sua funcionalidade de forma que possam ser usados dentro do arquivo openser.cfg.

Na implementação da proposta desta dissertação foi criada uma lógica para validar os domínios tidos como confiáveis no tratamento do registro dos usuários que não sejam locais, usando uma nova função *is_from_trusted()* no arquivo de configuração.

Com o UA em mobilidade sendo confiável e de um domínio remoto, o pedido *REGISTER* de um usuário em mobilidade será encaminhado ao domínio remoto com uso da função *relay($register())*, que foi modificada para tal. Esta e outras extensões são descritas no capítulo 5, que trata de uma implementação da proposta.

Capítulo 4

Solução proposta para apoio à mobilidade

4.1 - Objetivo

O problema abordado neste trabalho é definido como: o estabelecimento de rotas eficientes para mídia de voz em uma infraestrutura da rede, quando um usuário VoIP em mobilidade se registrar no domínio remoto, para conseguir realizar ou receber chamada. A Figura 1.1, exemplifica o caminho ineficiente da rota da mídia entre o UA B.spo.voip.br e o UA C.rio.voip.br, quando o usuário do domínio spo.voip.br estiver em mobilidade visitando a instituição RIO. Neste cenário, apesar do UA B.spo.voip.br e UA C.rio.voip.br estarem na mesma rede da instituição RIO, o fluxo da mídia tem que percorrer toda a rede do provedor indo até a instituição SPO e retornar à instituição RIO. Se a rede do provedor estiver com qualidade desfavorável (latência, variações de atrasos ou perdas), a comunicação entre B.spo.boip.br e C.rio.voip.br poderá ser ruim, apesar de localmente existir a condição para uma comunicação de perfeita qualidade, se o fluxo de mídia ficar confinado ao domínio rio.voip.br.

A solução seria permitir ao usuário B.spo.voip.br o registro no domínio rio.voip.br o mais próximo e permitir que o fluxo de mídia possa ficar confinado ao domínio rio.voip.br, como mostra a Figura 1.2. Cabe ressaltar, como ainda mostrado na Figura 1.2, que o processo de autenticação continua a ser feito no domínio de origem do usuário B.spo.voip.br, isto é, na instituição SPO do domínio spo.voip.br, pois é ali que se encontram as permissões e senhas associadas a este usuário.

Para um melhor entendimento, a solução proposta é explicada para três operações básicas:

- Registro do UA no domínio mais próximo;
- Estabelecimento de chamadas;
- Recebimento de chamadas;

Antes da apresentação das três operações acima, são apresentadas as novas estruturas para dar suporte à solução.

4.2 – Alterações de estruturas para suporte ao registro distribuído

Os serviços de registro e localização possuem o objetivo de permitir que os usuários realizem suas chamadas ou as recebam de qualquer instituição do programa, independente de sua localização física. O cadastro dos usuários, em uma instituição, pode ser realizado através do LDAP (serviço de diretório), ou por um SGBD (Sistema gerenciador de banco de dados). A autenticação de usuário que solicita seu registro no servidor SIP de sua instituição pode ser feita através do servidor RADIUS [LOUGHNEY, J., CAMARILLO, G., 2005] ou de outro mecanismo direto no banco de dados.

Quando do registro de um usuário em mobilidade no *proxy* local da instituição visitada, houver a necessidade de se alterar o comportamento da validação do serviço denominado *registrar* para que o usuário de outro domínio consiga realizar o registro (o que configura um procedimento não conforme com a RFC 3261) é implementado como contribuição deste trabalho. Os registros devem ser realizados de acordo com a autenticação local com validação em domínio correspondente, ou seja, em seu próprio domínio de origem.

Além dos cuidados de alteração no processo de registro no domínio remoto, há a necessidade do domínio de origem manter o estado sobre a mobilidade de seu usuário, para que, ao receber um pedido de estabelecimento de chamada (*INVITE*), o domínio de origem faça o redirecionamento para o domínio remoto (onde se encontra de fato o usuário em mobilidade). Esta alteração de comportamento no domínio origem também é um procedimento não conforme com a RFC 3261 e implementado como extensão na programação do domínio.

Num processo de registro normal pela RFC 3261, um usuário remoto se registra no domínio origem com seu IP remoto. Uma vez que o *proxy* opera como *proxy* de mídia, a chamada entrante para este usuário é vista e o

fluxo de mídia recebido pelo *proxy* é direcionado para o usuário remoto, no IP de registro.

Não existe a possibilidade do pedido de estabelecimento da chamada ser redirecionado para o outro *proxy*, o que acontece quando o *proxy* origem retorna uma resposta 3xx, pois o usuário não está sendo servido por nenhum outro *proxy* de fato. O usuário continua registrado no seu domínio origem. Em resposta a um *INVITE*, um *proxy* pode também recusar a chamada, devido a erro de cliente ou erro de servidor para usuários que não sejam locais. É proposto uma alteração no comportamento deste serviço para que o registro do usuário em mobilidade seja redirecionado para o domínio remoto onde ele se encontra de fato registrado, como descrito anteriormente, no item 4.2. O *proxy/registrar/redirect* tem, então, ao receber uma chamada para um usuário/endereço URI local, que diferenciar se o usuário está registrado localmente ou em mobilidade.

Para compreender os mecanismos propostos, será utilizado um cenário exemplo conforme mostrado na Figura 4.2.1.

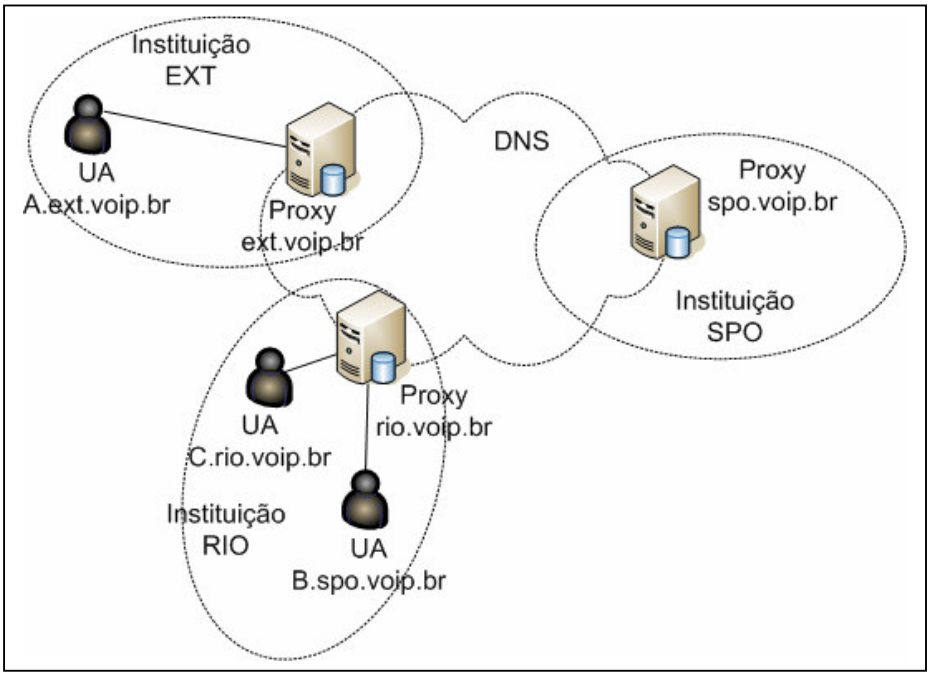


Figura 4.2.1 – Cenário da topologia utilizada

Nesta figura, o usuário B.spo.voip.br da instituição SPO, cujo domínio é spo.voip.br, está em mobilidade na instituição RIO, cujo domínio é rio.voip.br. Além das duas instituições descritas, existe uma terceira instituição denomi-

nada EXT, cujos usuários podem vir a interagir com o usuário B.spo.voip.br que está em mobilidade no domínio rio.voip.br.

Inicialmente são descritas as modificações em tabelas do modelo de dados que atendem à configuração do ambiente, para suporte à operação otimizada do usuário federativo em mobilidade no domínio rio.voip.br. As estruturas das tabelas descritas a seguir se encontram no Apêndice D, sendo que a tabela 4.2.2, descrita abaixo, se localiza no domínio *rio.voip.br*, na base de dados do serviço que também dá suporte ao registro dos usuários.

id	trusdomain	last_modified
1	spo.voip.br	2008-11-10 23:58:00
2	rio.voip.br	2008-11-10 23:59:00
3	ext.voip.br	2008-11-10 00:00:00

Tabela 4.2.2 – Tabela trusdomain comum às instituições da federação

A tabela 4.2.2, chamada *trusdomain,* deve ser criada para realizar a validação do registro dos usuários em mobilidade no domínio visitado e dar suporte à federação, na qual constam os domínios confiáveis que fazem parte do serviço VoIP. O preenchimento desta tabela deve ser realizado com a identificação dos domínios utilizados nas instituições da federação e qualquer alteração deve ser replicada para todas as instituições do serviço. Essa estrutura faz parte da contribuição proposta e foi adicionada ao modelo de dados do serviço *registrar*.

username	domain	Contact	expires	socket
B	spo.voip.br	sip:B@172.31.12.142	2009-04-08 05:31:03	udp:172.31.12.142:5060

Tabela 4.2.3 - Tabela de localização na instituição RIO

id	domain	last_modified
1	rio.voip.br	2009-04-08 04:25:00

Tabela 4.2.4 – Tabela de domínio na instituição RIO

Para o processo de localização e reconhecimento dos domínios da federação serão utilizadas as tabelas 4.2.3 e 4.2.4.

Com objetivo de verificar a localização do usuário, serão consultadas as colunas *domain* e a coluna *contact*. A coluna *domain* identifica que o UA B.spo.voip.br faz parte do domínio spo.voip.br e a coluna *contact* determina que seu endereço IP é do domínio rio.voip.br, ou seja, o UA B.spo.voip.br está em mobilidade na instituição cujo domínio é rio.voip.br. O preenchimento desta tabela é em tempo de execução do serviço, ou seja, sua inicialização ocorre na validação de localização dos usuários do serviço.

username	domain	email_address	src_ip	credit	is_mobility
A	rio.voip.br	A@spo.voip.br	172.31.12.147	30	0
B	rio.voip.br	B@spo.voip.br	172.31.12.142	30	1
21	rio.voip.br	21@rio.voip.br	172.31.12.150	NULL	NULL

Tabela 4.2.5 – Tabela de assinantes na instituição RIO

A Figura 4.2.5 mostra a tabela de assinantes, que é nativa do serviço *registrar* do domínio rio.voip.br. Todavia se propõe uma alteração na estrutura desta tabela, a fim de dar suporte à mobilidade dos usuários, com a inclusão das colunas: *src_ip*, *credit* e *ismobility*. Aqui, *src_ip* refere-se ao endereço lógico do usuário, o endereço IP associado ao UA,. e *credit* referencia os créditos dos usuários, a ser preenchido por um sistema de controle de bilhetagem, que, no caso deste trabalho, foi preenchido na transação da requisição de registro do usuário em mobilidade com a consulta na base remota, deta-

lhado na figura do diagrama 4.3.1. Vale ressaltar que ao final de um diálogo, quando acontecer um *CANCEL*, *BYE* ou caso a ligação caia, o valor resultante dos créditos utilizados será transferido de volta ao domínio de origem pelo servidor registrar, com uso de um novo método incluído no núcleo do serviço do OpenSER, o *NOTIFY_CRED*. O método *NOTIFY* já existe, porém ele é sempre disparado por um UA (*user agent*) para transportar a informação de uma ocorrência assíncrona, dentro de um diálogo (comunicação com outro UA), quando existir uma subscrição definida pelo método *SUBSCRIBE* com um servidor SIP. A subscrição existe exatamente para permitir o recebimento de notificações. Como não existe nenhum método que atenda à necessidade de interação direta entre *proxies*, foi proposto o uso desse novo método, que é semelhante ao *NOTIFY*, porém ocorre entre os servidores envolvidos no controle de chamadas de seus UAs. Veem-se maiores detalhes da atualização dos créditos no capítulo 5.

As Figuras 4.2.6 e 4.2.7 mostram como um usuário B.spo.voip.br, em mobilidade na instituição RIO, poderia receber uma chamada do usuário A.ext.voip.br, de forma otimizada. Quando o usuário A.ext.voip.br enviasse seu pedido de estabelecimento de chamada, o *INVITE* para o *proxy* do domínio spo.voip.br. Este *proxy*, consultaria as informações do registro de B.spo.voip.br e retornaria um *REDIRECT* (302 - mudança temporária), indicando que a chamada deveria, na realidade, ser enviada para o *proxy* do domínio rio.voip.br, no qual estaria registrado de fato o usuário B.spo.voip.br em mobilidade. Ao receber o *REDIRECT* com o endereço do *proxy* rio.voip.br, o *proxy* ext.voip.br reenviaria o *INVITE* ao *proxy* rio.voip.br, que repassará ao destinatário B.spo.voip.br.

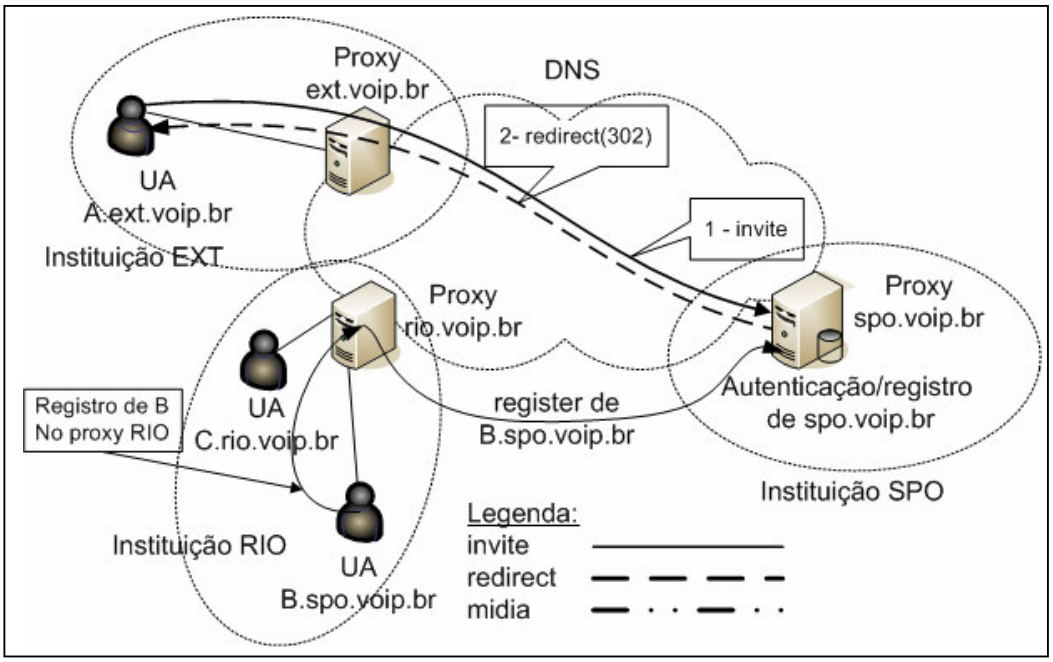


Figura 4.2.6 – Requisição *INVITE* com redirecionamento na otimização proposta

A Figura 4.2.6 representa o início da sequência da requisição *INVITE* para estabelecimento de um diálogo a partir de um usuário A.ext.voip.br para um usuário B.spo.voip.br que esteja em mobilidade na instituição RIO.

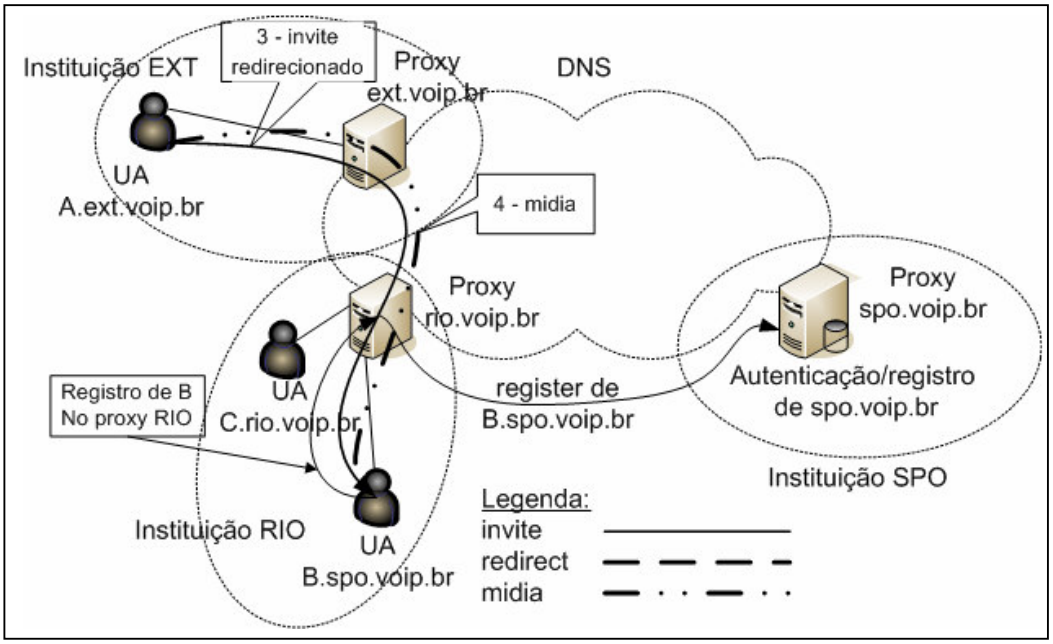


Figura 4.2.7 – Rede SIP com a otimização proposta

A Figura 4.2.7 representa a continuação da sequência referida na Figura 4.2.6 com a numeração das requisições representando o *INVITE* redirecionado e posteriormente o tráfego da mídia.

O repasse só aconteceria porque o UA B.spo.voip.br estaria registrado corretamente no domínio rio.voip.br.

Para que as operações funcionem como descritas, são necessárias modificações na forma que um usuário em mobilidade se registra, no servidor denominado *registrar* do domínio local. Esta configuração normalmente realizada pelo próprio usuário manualmente ou se utiliza um servidor de aplicações que o servidor de registro da instituição RIO validará a URI do usuário em mobilidade, usando uma nova tabela denominada *trusdomain*, descrita no item 4.2.2, que seja comum entre as instituições da federação, através de um procedimento denominado replicação de dados.

Na confirmação da autenticação com a resposta 200, ok, como está definida na RFC2617, serão transferidos para a base de dados da instituição que solicitou o registro do UA (dita remota) os valores de créditos (vide Apêndice B, Figura B09) que o UA possui em sua base de dados local. Este procedimento é uma extensão proposta nesta dissertação. Além disso, será realizada a inclusão do usuário (vide Figura 4.2.3) na tabela de localização relacionada com a tabela *subscribers* do domínio rio.voip.br, indicando que o UA B.spo.voip.br poderá ser registrado nesse domínio.

O usuário B.spo.voip.br conseguirá através da sinalização SIP se registrar no domínio rio.voip.br, fazendo com que o *proxy* deste domínio insira na base dados local toda a informação pertinente ao usuário B.spo.voip.br em mobilidade. Uma destas informações é o domínio spo.voip.br. Como parte do processo de autenticação remota, o *proxy* da instituição SPO fará a inserção na base de dados de localização do domínio spo.voip.br de uma linha indicando a nova localização do usuário B.spo.voip.br, preenchendo os campos *domain* com o endereço de origem do domínio ou a URL, e o campo *CONTACT* que representa a URI com o endereço lógico da nova localização deste UA B.spo.voip.br, pois qualquer requisição que chegue para este UA B.spo.voip.br deverá ser redirecionada para a nova localização. O teste de localização é realizado consultando a tabela de localização. E através dos

serviços de consulta de DNS (SRV, NAPTR, A) é possível resolver qualquer referência associada a dispositivos remotos.

Além das estruturas alteradas já descritas, os processos de registro distribuído e de obtenção e retorno de créditos exigem o uso de primitivas SIP com campos adicionais em novos procedimentos, que serão melhor detalhados após ser descrito o cenário em que foi feita a implementação para a prova de conceito. Este detalhamento será mostrado no capítulo 5.

4.3 – Processo de registro distribuído

A Figura 4.3.1 representa a sequência da requisição proposta na arquitetura de registro distribuído. O registro de um usuário em mobilidade é realizado na localidade que está sendo visitada, e, no domínio local do usuário em mobilidade, a tabela de localização tem uma entrada que aponta para a localidade visitada, ou seja, onde o usuário se encontra. O comportamento do usuário em mobilidade no domínio remoto é típico de um usuário de uma instituição que faça parte do serviço de telefonia IP, como fone@RNP, que esteja em visita a outra instituição e deseje realizar e receber chamadas pelo seu cliente VoIP, usando um *softphone* instalado em seu *notebook*.

Para que este usuário consiga utilizar o serviço é necessário ter navegabilidade na web, pela infraestrutura de rede da instituição visitada, registrar-se no serviço VoIP, além de possuir créditos para estabelecer chamadas. O processo de registro distribuído proposto depende da adoção da tabela *trusdomain*, para validação do registro dos usuários em mobilidade, do uso da tabela de localização descrita anteriormente e da utilização da tabela *trusted*, usada pelos arquivos de permissões e bloqueios denominados, *permissions.allow* e *permissions.deny.* Os arquivos *permissions.allow* e *permissions.deny* são nativos do OpenSER e concedem direitos para o estabelecimento e recebimento de chamadas. Através de uma função que fica no bloco de roteamento específico do tratamento de *INVITE*, as regras definidas em um dos arquivos são verificados na implementação. No caso, o arquivo *permissions.allow*, como será detalhado no Capítulo 5 na parte da definição dos diagramas que mostram o fluxo das primitivas envolvidas no estabelecimento de uma chamada.

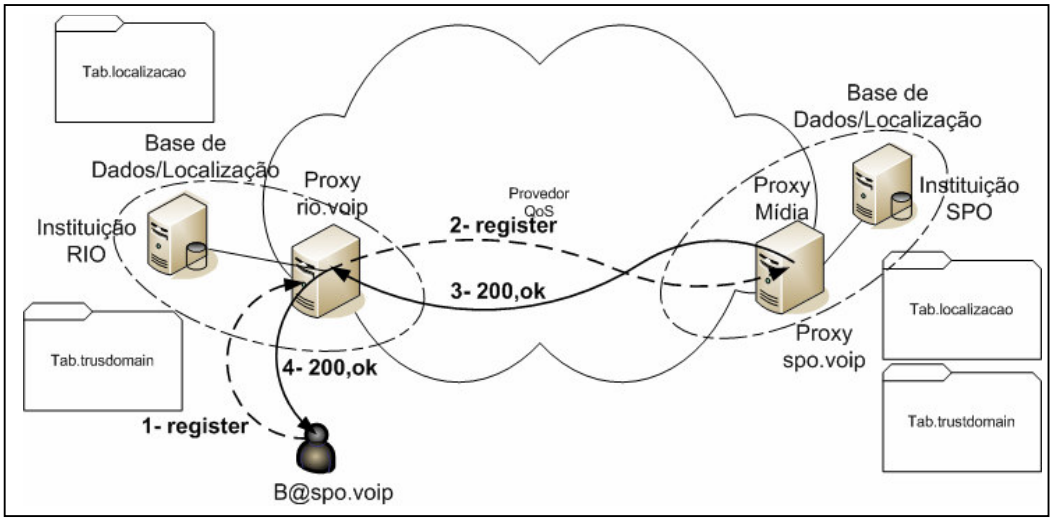


Figura 4.3.1 – Proposta da sequência do registro distribuído

A Figura 4.3.1 mostra a sequência proposta de registro de um UA em mobilidade e a requisição *REGISTER* sendo transferida ao seu domínio de origem (instituição SPO) pelo servidor *proxy* mais próximo da instituição RIO. O capítulo 5, de implementação, tratará do detalhamento desta sequência.

O procedimento de registro mantém o processo de autenticação no domínio remoto do UA, que detém as credenciais numa base de dados ou num serviço de diretórios como LDAP. Durante todo o processo de autenticação do UA, é mantida a segurança das credenciais, utilizando-se do procedimento da RFC2617, que determina o armazenamento criptografado e o uso de desafios para estabelecer a autenticação dos usuários.

4.4 – Encaminhamento de chamadas por um usuário em mobilidade

Após o processo de registro do UA de um usuário em mobilidade teremos a identificação do AoR (*Address-of-Record*) em que define a URI deste usuário com o respectivo endereço lógico a ele atribuído, armazenado na base de dados do servidor *registrar* como se fosse um UA local. Quando este usuário realizar uma requisição de chamada (*INVITE*), passará pelo procedi-

mento de validação que testará a URI como sendo local pela regra dada no arquivo *permissions.allow* .

4.5 – Recebimento de chamadas por um usuário em mobilidade

Em relação ao domínio origem do usuário que esteja em mobilidade, após processo de registro, a forma encontrada para viabilização do recebimento das chamadas para um UA em mobilidade que não faça parte do domínio local foi acrescentar na tabela *subscriber* à coluna *ismobility* com valor do tipo *booleano* (0 ou 1) no domínio remoto deste UA. Ou seja, ao tratar uma requisição em direção a este UA, é verificado o valor desta coluna. Se o valor for igual a (1), deve ser realizado o *redirect* para o domínio que está sendo visitado, pois este UA está em mobilidade; já se o valor for igual a (0), deve ser realizado o *lookup* na base local e encaminhada a requisição diretamente ao AoR (*address-of-record*) que esteja na base deste domínio origem.

Em relação ao domínio visitado em que o usuário em mobilidade está registrado, constará a informação sobre o seu realm/domínio. Quando o *proxy* receber uma chamada destinada a um usuário em mobilidade de um domínio externo, a modificação no tratamento do recebimento da chamada fará com que o *proxy* analise a presença deste usuário na sua tabela de localização, sem gerar um redirecionamento ou reenvio da chamada. A consequência é que a chamada é encaminhada ao usuário devido, como se ele fosse um usuário local daquela instituição.

4.6 – Rota ótima para a mídia

A Figura 4.6.1 mostra o caminho percorrido pela mídia numa chamada do UA ext para o usuário B.spo.voip.br, em mobilidade na instituição RIO.

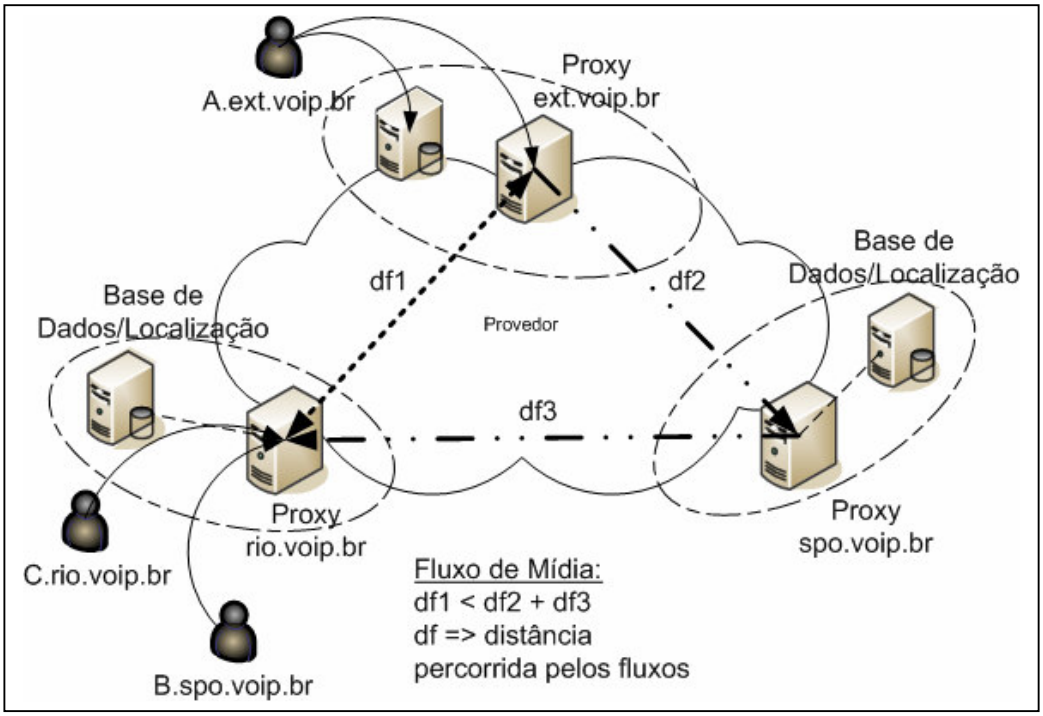


Figura 4.6.1 - Procedimento de validação do processo de encaminhamento

Como pode ser observado, o ganho de desempenho e qualidade é inerente ao fato do fluxo de mídia passar a ser enviado ao *proxy* em que o usuário se registra, evitando a passagem pelo *proxy* de origem (df1), como seria o caminho tradicional (df2+df3). Quanto mais próximos estiverem fisicamente, topologicamente os dois interlocutores, maior o impacto.

Numa comunicação VoIP, *buffers* de compensação de *jitter* são sempre utilizados nos UAs para garantir uma tolerância nos instantes de reprodução da voz, o que é essencial quando existe grande variação de *jitter*, além de diminuir as perdas características da rede de pacotes. Quanto maior a latência da rede, tanto maior será o *jitter* associado aos *buffers* de compensação, no caso do uso de algoritmos dinâmicos de compensação. [RAMJEE, R; KUROSE, J.; TOWSLEY, D.; SCHULZRINNE, H.,1994] Na *Internet*, o uso de

rotas com menor número possível de *hops* é determinante para diminuir a variação de atraso. Um aumento de atraso associado aos *buffers* de compensação de *jitter* ajuda a diminuir as perdas, mas tem impacto na qualidade das chamadas, pois afeta a interatividade. Controlar a perda e o atraso na rota da chamada é fundamental para a garantia da qualidade. E a proposta desta dissertação visa exatamente a obter rotas ótimas para o usuário em mobilidade e, consequentemente, melhor qualidade para as chamadas envolvendo este usuário. [Lustosa, L.C.G.; Rodrigues, P.H.A; David, F; Quinellato, D.G., 2005]

Convém ressaltar que o processo de encaminhamento das chamadas faz parte da etapa de sinalização e não interfere, por si, na qualidade da chamada, o que dependerá essencialmente da rota seguida pela mídia. A diminuição do trânsito pela rede é fundamental para uma garantia de melhor qualidade.

Capítulo 5

Implementação da proposta

Para ilustrar a implementação da proposta da otimização da rede SIP no contexto de interligação de instituições clientes em um serviço de telefonia IP, foram utilizadas máquinas virtuais (vmware) para a criação de uma arquitetura com três domínios e três máquinas clientes, utilizando o software de código aberto OpenSER versão 3.1 com a função de servidor SIP, atuando como servidor *Proxy,* servidor *registrar* e de redirecionamento. Utilizou-se o serviço de resolução de nomes, DNS com o BIND9 e SGBD MySQL 5.1, para o repositório de tabelas usadas nas consultas de localização, registro/autenticação e créditos de ligações.

O servidor OpenSER é apropriado para tal solução, pois é o software utilizado no serviço fone@RNP e possui uma arquitetura modular, como visto no Capítulo 3, seção 3.5, subseção 3.5.2 da arquitetura.

Neste capítulo, a implementação da proposta no cenário, descrito a partir da seção 5.3, será detalhado com a apresentação de diagramas para as três operações básicas: registro distribuído realizado por um usuário em mobilidade, estabelecimento de chamadas por um usuário em mobilidade e o recebimento de chamadas por um usuário em mobilidade.

Os seguintes domínios e usuários foram criados:

Local EXT	Local SPO	Local RIO
Domínio: ext.voip.br	Domínio: spo.voip.br	Domínio: rio.voip.br
UA(uri):A.ext.voip.br	UA(uri):B.spo.voip.br	UA(uri): C.rio.voip.br

Tabela 5.1 - Ambiente

Em cada domínio foram instalados o serviço de DNS, o servidor SGBD (MySQL) e o servidor SIP OpenSER. Para entendimento da proposta são analisados os procedimentos para as três operações fundamentais:

- Registro distribuído;

- Encaminhamento de chamadas de um usuário em mobilidade;
- Recebimento de chamadas por um usuário em mobilidade.

O cenário de implementação da validação da proposta será utilizado para uma explicação dos mecanismos envolvidos e compreensão das alterações propostas.

5.1 - Funcionamento do Proxy OpenSER

O uso de um servidor OpenSER como *proxy SIP, servidor registrar* e de redirecionamento, permite a implementação via *script* do arquivo de configuração do serviço, o *openser.cfg*, que é executado cada vez que uma nova mensagem SIP é recebida. Por exemplo, um UA (A.ext.voip.br *SIP phone*) envia um convite *INVITE* para outro UA (B.spo.voip.br *SIP phone*) para uma conversação (UA A.rio.voip.br realiza uma chamada para B.spo.voip.br). O UA A.ext.voip.br envia uma mensagem SIP *invite* para o OpenSER e o bloco de roteamento (*main route* (*block)*) principal do *openser.cfg* iniciará pelo topo e executará os comandos ali encontrados.

Dependendo da lógica do script, pode-se enviar o pedido *INVITE* a B.spo.voip.br usando a função *t_relay()*, ou enviar uma resposta de erro para A pela função *sl_send_reply()*, ou apenas descartar o pedido *INVITE*, caso o final do roteamento principal seja atingido ou um *break()* executado, o que não é recomendado. B.spo.voip.br responderá ao pedido *INVITE* com uma mensagem de *OK*. O *OK* é uma resposta direta para o *INVITE* inicial e esta mensagem é tratada pela última seção dentro de *on_reply_route[x]*. Se B.spo.voip.br não respondeu ou respondeu com um erro (de ocupado, etc), a função *failure_route[x]* seria chamada.

Finalmente A.rio.voip.br enviará um *ACK* para B.spo.voip.br para informar que tudo foi recebido e aceito. Um diálogo *INVITE* também incluem respostas provisionais (*trying*) antes da primitiva *OK.*

Então, como tudo é tratado dentro do *script openser.cfg,* todas as mensagens SIP iniciarão novas transações SIP que executarão pelo topo do bloco de roteamento principal (*main route{block}*). No exemplo acima, “A” envia um

pedido de estabelecimento de chamada (*INVITE),* iniciando uma transação que é respondida com *OK* por B.spo.voip.br.

Como o *script openser.cfg* é uma lógica de roteamento, deve-se tratar cada tipo de mensagem SIP corretamente. Os fluxos e a cada mensagem de resposta possível na transação. A transação é apropriadamente tratada pela resposta ou por falhas das rotas, para conseguir o que se deseja. A lógica do *script* configura uma total liberdade para o tratamento das mensagens SIP, o que pode ser desastroso em relação a qualquer erro aparentemente inofensivo, causando o desvio do cumprimento da RFC 3261.

5.2 – Modificações introduzidas para atender à solução

Com o objetivo de mostrar de forma clara quais foram as modificações introduzidas nas primitivas, foi realizada uma alteração no *core* do servidor OpenSER. Para a validação dos domínios nas requisições de registro (primitiva SIP: *REGISTER*) pelo UA em mobilidade, foi adicionada a função *is_from_trusted()*. Utilizada no script de configuração do serviço, openser.cfg, a função consulta a tabela *trusdomain* que possui todos os nomes dos *realms* (domínios) relacionados com a rede federada. Assim, um pedido de registro pode ser enviado ao domínio de origem do UA em mobilidade para que as credenciais sejam validadas via *digest* , conforme descrito na RFC 2617 [FRANKS, J., et al.,1999], RFC 3702 [LOUGHNEY, J.; CAMARILLO, G.; 2005].

```
mysql> select * from trusdomain;
+----+-------------+---------------------+
| id | trusdomain  | last_modified       |
+----+-------------+---------------------+
|  1 | spo.voip.br | 2008-11-10 23:58:00 |
|  2 | rio.voip.br | 2008-11-10 23:59:00 |
|  3 | ext.voip.br | 2008-11-10 00:00:00 |
+----+-------------+---------------------+
3 rows in set (0.00 sec)

mysql>
```

Figura 5.2.1 – Screenshot da tabela trusdomain

A Figura 5.2.1 mostra o conteúdo da tabela *trusdomain* criada para armazenar os domínios que deverão fazer parte de uma federação e com isso ser utilizada como repositório de realms/domínios para que sejam validados em tempo de execução, com a recepção das requisições *REGISTER* por parte dos *proxies* de registro e no processo de estabelecimento de chamadas.

```
mysql> select username, domain, contact, expires, socket from location;
+----------+-------------+--------------------------+---------------------+------------------------+
| username | domain      | contact                  | expires             | socket                 |
+----------+-------------+--------------------------+---------------------+------------------------+
| 1142     | spo.voip.br | sip:B@172.31.12.142:5060 | 2009-04-08 05:31:03 | udp:172.31.12.142:5060 |
+----------+-------------+--------------------------+---------------------+------------------------+
1 row in set (0.00 sec)

mysql> select * from domain;
+----+-------------+---------------------+
| id | domain      | last_modified       |
+----+-------------+---------------------+
|  1 | rio.voip.br | 2009-04-08 04:25:00 |
+----+-------------+---------------------+
1 row in set (0.00 sec)

mysql>
```

Figura 5.2.2 – Screenshot da tabela location e domain

A Figura 5.2.2 mostra a tabela *location* e a tabela *domain*, que são utilizadas para dar suporte ao processo de localização dos recursos, os AoRs, e validação dos UAs locais para que sejam realizados as requisições de chamadas.

```
mysql> select username, domain, email_address, src_ip, credit, ismobility from subscriber;
+----------+-------------+------------------+---------------+--------+------------+
| username | domain      | email_address    | src_ip        | credit | ismobility |
+----------+-------------+------------------+---------------+--------+------------+
| 2147     | rio.voip.br | A@rio.voip.br    | 172.31.12.147 | 30     |          0 |
| 1142     | rio.voip.br | B@spo.voip.br    | 172.31.12.142 | 30     |          1 |
| 2150     | rio.voip.br | 2150@rio.voip.br | 172.31.12.150 | NULL   |       NULL |
+----------+-------------+------------------+---------------+--------+------------+
3 rows in set (0.00 sec)

mysql>
```

Figura 5.2.3 – *Screenshot* da tabela subscriber

A Figura 5.2.3 mostra a alteração realizada na tabela *subscriber*, adicionando os seguintes atributos: src_ip, que indica o endereço lógico IP do UA registrado, *credit* indica o valor dos créditos inseridos, pela própria sinalização VoIP, na base, após consulta a instituição SPO, dita aqui como *home* do usuário em mobilidade. E um identificador booleano que informa ao servidor VoIP quem está em mobilidade.

O registro distribuído se dará após a validação na base de dados do servidor registrar por uma nova função adicionada no core do OpenSER, descrita anteriormente como *is_from_trusted()*.

O encaminhamento de chamadas por um usuário em mobilidade, é realizado após validação em um arquivo de permissões, chamado *permissions.allow,* desde que conste seu registro na tabela de assinantes do serviço *registrar* atrelado a sua localização atual. O recebimento de chamadas pelo usuário que esteja em mobilidade acontecerá no domínio onde ele estiver, desde que tal usuário tenha seu registro no domínio do serviço registrar local e sua localização também esteja definida em seu domínio de origem, através da tabela localização. Será necessário também o uso do bloco de roteamento de respostas para tratar o método 302 (de redirecionamento) e reencaminhar a requisição ao destino especificado no topo do parâmetro *record-route*.

As estruturas abaixo são usadas nas operações da proposta e o detalhamento complementar é mostrado nos apêndices.

Estrutura	Definição
is_from_trusted() – Introduzida no core do servidor proxy SIP.	Validação de um usuário de uma instituição que faça parte do serviço VoIP, através de uma base de dados, compartilhado pela federação.
relay($register()) – Função modificada para atender à requisição de registro.	Encaminhamento da requisição de registro de um usuário em mobilidade para validação de suas credenciais no seu domínio de origem.
avp_db_query() – Função existente do módulo avpops.so. Alterada apenas para realizar um encaminhamento para o próximo servidor *proxy*.	Realizar consulta na base de origem e introduzir na primitiva de resposta de confirmação (2xx) o valor de créditos de um usuário em mobilidade.
Tabela trusdomain – Introduzida na base de dados do servidor registrar para validação junto com a função is_from_trusted().	Possui os seguintes atributos: id (chave primária com autoincremento), trusdomain (unique key – uma constraint), last-modified (manter histórico-

	co).
Tabela Subscriber – tabela existente na base de dados do servidor registrar, porém introduzidas três colunas para dar suporte à validação do usuário em mobilidade.	Adicionados os seguintes atributos nesta tabela: src_ip (endereço ip de origem, dados por um dhcp local), credit (a ser preenchido após a resposta de confirmação do pedido de registro, com uso do módulo avpops.so) e ismobility (valor booleano)
Método **NOTIFY_CRED**	Utilizado por um servidor proxy/registrar para transportar informação de créditos atualizados ao servidor registrar de origem do UA em mobilidade, após a conclusão de um diálogo, envolvendo um UA em mobilidade registrado e usando um evento síncrono com o encerramento de um diálogo.
openser.cfg – Modelagem da lógica de roteamento das mensagens SIP, quando passar pelo servidor.	Script de configuração do servidor proxy SIP

Tabela 5.2.1 – Estruturas introduzidas e modificadas no serviço

5.3 – Registro distribuído

O usuário B.spo.voip.br, do domínio spo.voip.br estará se registrando no domínio rio.voip.br, da instituição RIO. No processo de registro, o *proxy/registrar rio.voip.br* encaminhará a requisição *register* do UA B.spo.voip.br em mobilidade ao proxy/registrar *spo.voip.br* de forma encapsulada através da função *relay(register)* com objetivo de realizar a autenticação na base *home* deste UA que está em mobilidade, após descoberta e validação do domínio do UA móvel como *trusted* (com uso da tabela de mesmo nome na base de dados local que conterá todos os *Proxies* SIP de uma rede federada e tida como confiável).

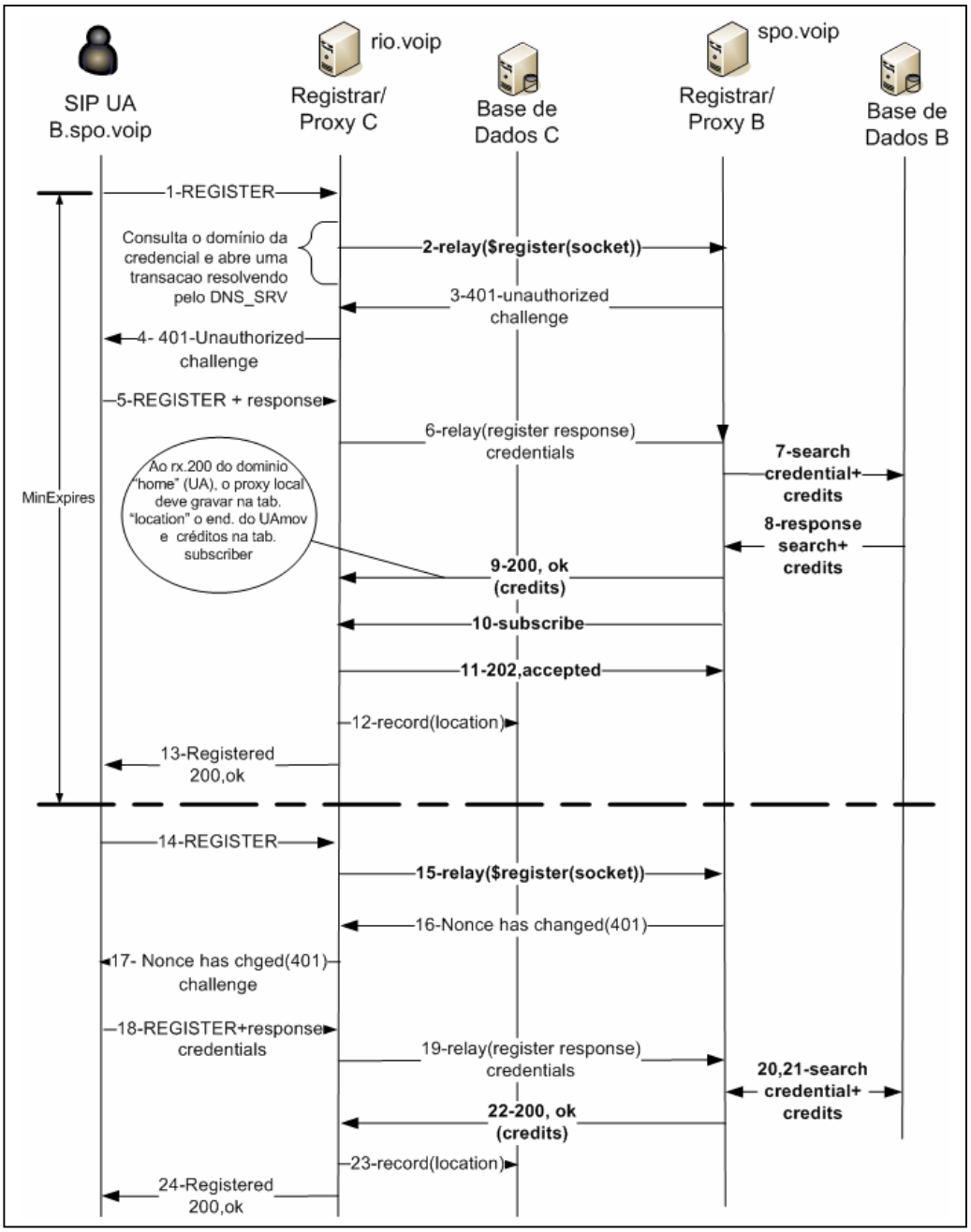


Figura 5.3.1 - Processo de registro de um UA de forma otimizada

A Figura 5.3.1 mostra o registro de um UA em mobilidade ao realizar uma requisição de registro a um servidor *proxy registrar.* O pedido de registro deverá ser tratado no bloco de roteamento do *script* do servidor. A função ***is_from_trusted()*** irá validar a URI apresentada pelo UA como sendo de uma origem confiável, comparando com a coluna denominada *trusdomain* da tabela trusdomain. Assim, será enviada a requisição *REGISTER* para o do-

mínio origem para que haja a autenticação do UA, utilizando a função modificada **relay($register())**. Requisição passará pelo processo de autenticação *Digest* [RFC 2617] e no final, com a confirmação de autenticação, os créditos serão trazidos no corpo da mensagem de confirmação, ou seja, na resposta do servidor de registro da localidade remota e gravados na tabela ***subscriber***, na coluna ***credit.*** Além das inserções do endereço IP atribuído nesse novo domínio e um flag denominado *ismobility* indicando que se trata de um usuário em mobilidade. Haverá também no processo de gravação, pela função *record(location)* a inclusão na tabela de localização do domínio de origem do usuário em mobilidade o campo ***contact*** referenciando o endereço lógico recebido. O valor MinExpires, é definido como um número inteiro de segundos em que o servidor *registrar* pode rejeitar um valor muito curto com a resposta 423 *Interval too brief*. O cliente, ao receber esse campo de cabeçalho na resposta da requisição, pode atualizá-lo para manter a transação ativa, além de definir um tempo ótimo para o registro neste servidor perdurar.

Fluxo de mensagens da Figura 5.3.1.

1- UA(B.spo.voip.br) inicia o processo de Registro de seu URI;
   UA(B.spo.voip.br) envia um *REGISTER* para o *proxy* definido no parâmetro *outbound proxy* configurado no softphone do próprio cliente, o endereço lógico do servidor *registrar* desse domínio;
2- *Proxy/registrar* (rio.voip.br) recebe o *REGISTER* e altera o valor(aumentando o valor para 1200seg) do campo de cabeçalho *MinExpires* e encaminha para o Proxy/*registrar*(spo.voip.br), usando a função modificada *relay($register())*; a alteração do *MinExpires* objetiva manter a transação da requisição do registro válida além do tempo padrão de 300 seg;
3- O servidor *Proxy/registrar* (spo.voip.br) verifica os campos de cabeçalho e solicita as credenciais, incluindo o campo *www-authenticate* , através de desafio (*challenge*);
4- O Proxy/Registrar (rio.voip.br) encaminhará este *reply* com pedido de desafio (*challenge*) ao UA(B.spo.voip.br), sendo fundamental que esse desafio seja passado intacto ao UA (B.spo.voip.br), pois ele contém o *nonce;*

5- O UA(B.spo.voip.br) responde ao desafio, usando o método *register* e preenchendo o campo *www-authorized* com suas credenciais;

6- O servidor *Proxy/registrar* (rio.voip.br) enviará esta mensagem ao *proxy/registrar* (spo.voip.br), anexando o campo de créditos, que se encontra vazio;

7- O servidor *Proxy/registrar* (spo.voip.br) validará na base de dados as credenciais do UA (B.spo.voip.br) e consultará os créditos deste;

8- O servidor *Proxy/registrar* (spo.voip.br) deverá receber um código de retorno ("$?" ou "$retcode") indicando sucesso(valor=1) junto com o valor de créditos deste UA(B.spo.voip.br);

9- O servidor *Proxy/registrar* (spo.voip.br) responde com "200 ok" mais um campo de créditos preenchido após consulta na base de origem;

10- O servidor *Proxy/registrar* (spo.voip.br) emite uma requisição *SUBSCRIBE* com o proxy/registrar (rio.voip.br) com objetivo de receber notificações do proxy/registrar que está registrando o UA B.spo.voip.br como remoto;

11- O *Proxy/registrar* (rio.voip.br) envia a resposta *202-accepted* em relação a requisição anterior;

12- O servidor *Proxy/registrar* (rio.voip.br) adicionará um registro na base de dados de localização (tabela:. *location*) e os créditos correspondentes na tabela subscriber;

13- O *Proxy/registrar* local (rio.voip.br) enfim, retornará ao UA(B.spo.voip.br) o 200,ok como confirmação do registro realizado, após o Min-Expires - Novo *REGISTER*

14- O UA(B.spo.voip.br) envia um novo *REGISTER* ao *proxy/registrar* (rio.voip.br), em virtude de o tempo de MinExpires ter expirado;

15 a 24 - O processo de registro será repetido;

5.4 – Encaminhamento de chamadas do usuário em mobilidade

Neste exemplo, o UA B.spo.voip.br em mobilidade e registrado na instituição RIO chama o UA C.rio.voip.br, usuário local do domínio rio.voip.br. Quando o UA B.spo.voip.br em mobilidade no domínio rio.voip.br realizar uma chamada (*INVITE*), o *proxy* do domínio rio.voip.br realizará o encaminhamento da chamada, pois terá na sua base de localização os AoRs (*Address of Record*) do UA B.spo.voip.br em mobilidade, após o processo de registro. O *proxy* não participará do encaminhamento. Todavia os créditos serão atualizados ao final da chamada.

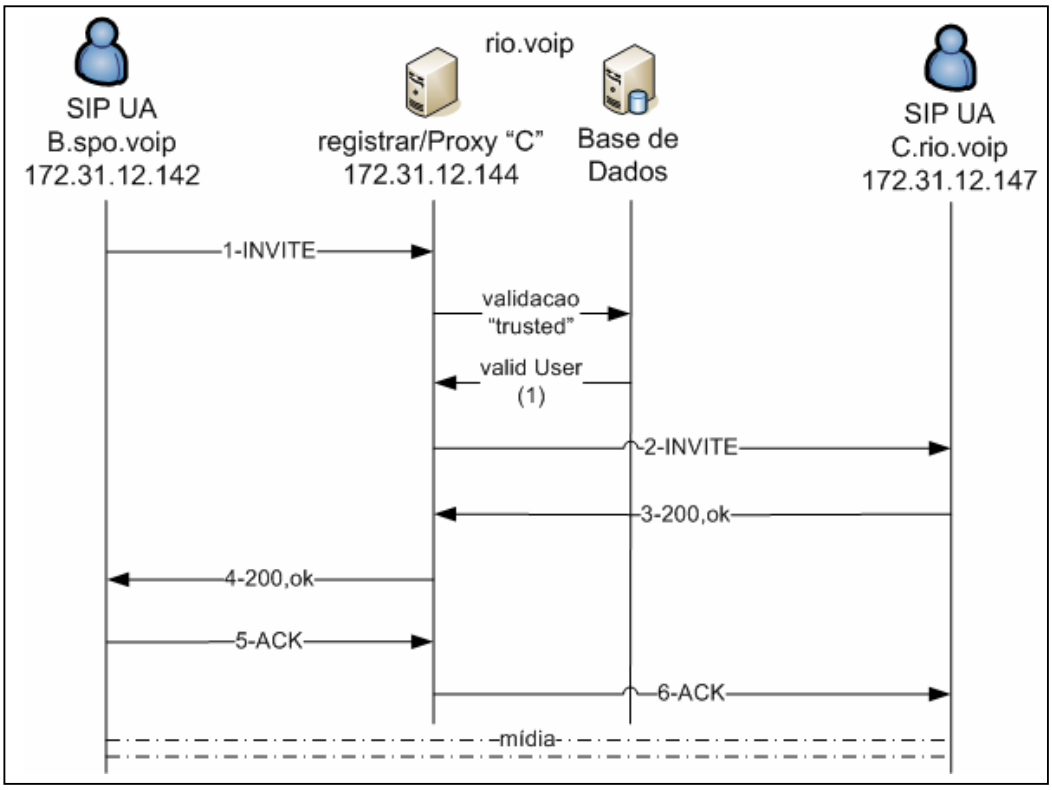


Figura 5.4.1 – Chamada Interna de UA B.spo.voip para o UA C.rio.voip

Fluxo de mensagens da Figura 5.4.1:

1) UA B.spo.voip.br inicia uma requisição *INVITE* , na qual será validado pela função *allow_trusted()* com o arquivo *permissions.allow* em que se define regras funcionando como filtro de acesso, A requisição pode

ser aceita ou não. As regras de filtro possuem uma sintaxe de escrita de acordo com :

< *source_addr, transport prot, regular expression* >

Será aceita a requisição do UA B.spo.voip.br, caso tenha na regra de permissão o endereço lógico de origem igual ao de UA B.spo.voip.br recebido pelo DHCP local, com protocolo de transporte configurado com o parâmetro *any* e/ou uma expressão regular ($fu – corresponde ao campo do cabeçalho “from” da requisição), equivalente a uma variável contendo a URI da requisição, que será testada na base de dados comparado com o campo *contact* (cujo formato é de URI) com o campo *from* da requisição;

A base de dados do domínio (spo.voip.br) responde com um código de retorno, indicando que o usuário é confiável, se for um UA em mobilidade;

2) O servidor *Proxy/registrar* (rio.voip.br) encaminhará a requisição *INVITE* , após realizar o processo de localização pelo campo de cabeçalho R-URI (Request URI) do destino;

3) O UA C.rio.voip.br, estando cadastrado na mesma base, receberá a requisição e responderá com um 200,ok;

4) O *Proxy/Registrar* (rio.voip.br), ao receber o 200,ok, o encaminhará ao UA B.spo.voip.br;

5,6) O UA B.spo.voip.br responderá com uma confirmação ACK, que será encaminhada pelo *proxy* ao UA C.rio.voip.br, indicando o fim da negociação e liberação para troca de mídia;

7) Mídia estabelecida. A partir deste instante, o tempo do fluxo de mídia será computado através do módulo ACC inserido no *proxy* OpenSER, Este módulo realiza a bilhetagem (a cada minuto diminui uma unidade na coluna *credit* da tabela *subscriber*) com auxílio da tabela de mesmo nome “acc” (nativa do serviço) que guarda os *timestamps* do diálogo;

Neste exemplo, o *proxy* da instituição RIO opera como *proxy* de mídia para a comunicação entre o UA B.spo.voip.br e o UA C.rio.voip.br.

Como proposta desta dissertação, o processamento dos créditos se dará com a finalização de um diálogo com *BYE* ou *CANCEL,* que passe pela PSTN. O tempo de fluxo de mídia como descrito no item 7 do diagrama da Figura 5.4.1, será computado e o valor resultado será retornado à base remota do UA que está em mobilidade utilizando um novo método proposto NOTIFY_CRED. O *proxy* registrar local, com uso do endereço que está em sua tabela de localização na coluna *domain*, transferirá o valor resultado que consta na coluna *credit* da tabela local *subscriber* para o *proxy* registrar do UA em mobilidade.

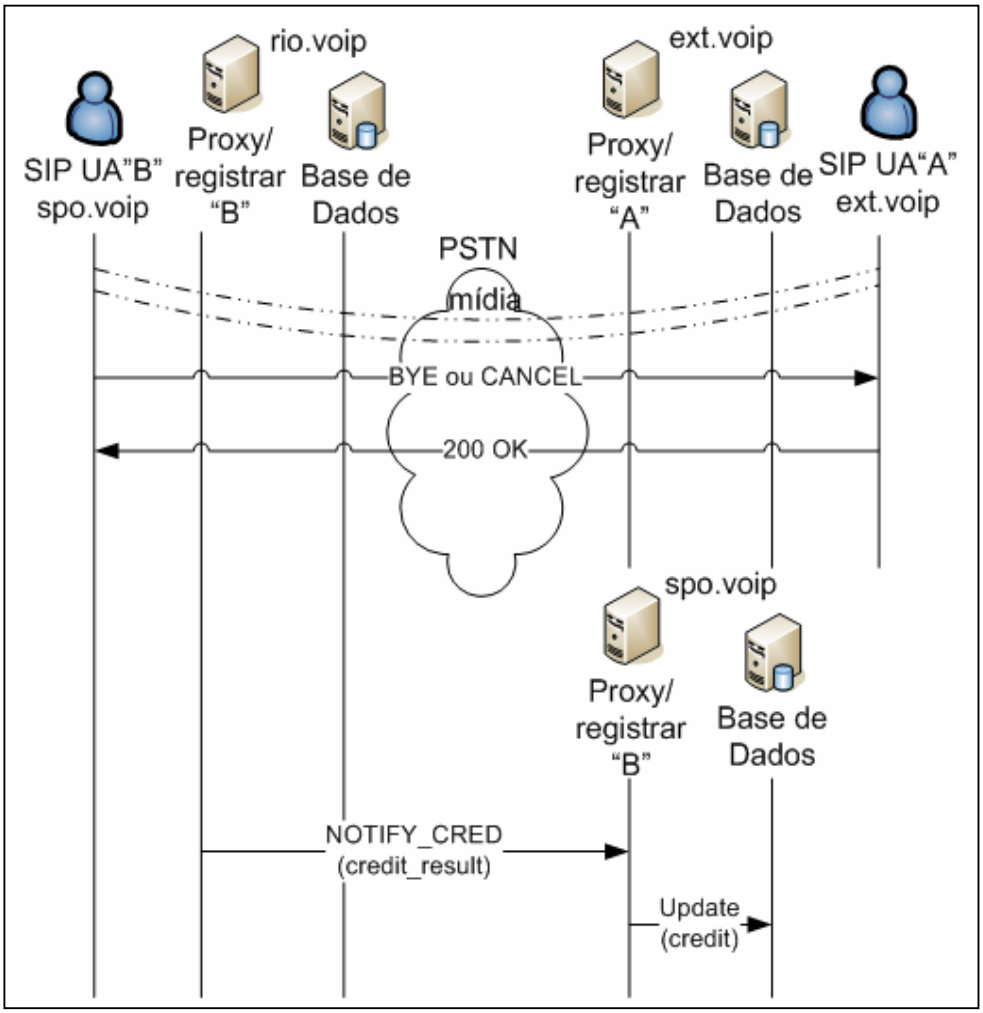


Figura 5.4.2 – Detalhamento do uso do método NOTIFY_CRED

A Figura 5.4.2 mostra, no fim de um diálogo entre os UAs B.spo.voip.br com A.ext.voip.br, as primitivas *BYE* ou *CANCEL,* liberando os recursos. O proxy do domínio rio.voip.br, através do novo método NOTIFY_CRED dispara uma mensagem para o *proxy* de origem B.spo.voip.br com o valor resultado que restou na coluna *credit* da tabela *subscriber* referente ao UA B.spo.voip.br em mobilidade. As exceções não foram tratadas, ou seja, caso o tempo gasto no diálogo supere o valor dos créditos, a ligação deveria cair ou uma mensagem seria disparada, porém isso não foi implementado. E o que acontece é que o fluxo de mídia continua e no fim do diálogo o valor transferido é zero (0), não sendo computados valores negativos.

5.5 – Recebimento de chamadas pelo usuário em mobilidade

Neste exemplo, um usuário externo UA A.ext.voip.br chama o usuário em mobilidade UA B.spo.voip.br. Feita a localização do domínio de B.spo.voip.br com uso de DNS/SRV, o pedido *INVITE* gerado pelo UA A.ext.voip.br será enviado para o domínio spo.voip.br.

Na Figura 5.5.1, o servidor *proxy* SIP do domínio spo.voip.br redireciona o pedido *INVITE* para o domínio onde esteja o UA em mobilidade, o que é um procedimento não conforme à RFC3261. O procedimento seria conforme, se fosse realizado um encaminhamento tipo *INVITE* direto pelo *proxy home* spo.voip.br para o seu UA, que está em mobilidade em outra localidade, formulando um modelo de comunicação tido como trapezoidal, sem passar pelo *proxy* local do domínio que está sendo visitado. Com a otimização, é realizado o *redirect* para o *proxy* local e este por sua vez realizará o encaminhamento ao UA móvel, com base nas informações extraídas da base de dados de localização, tabela de localização (usuários móveis) e da tabela *trusdomain* (endereços lógicos e domínios tidos como confiáveis).

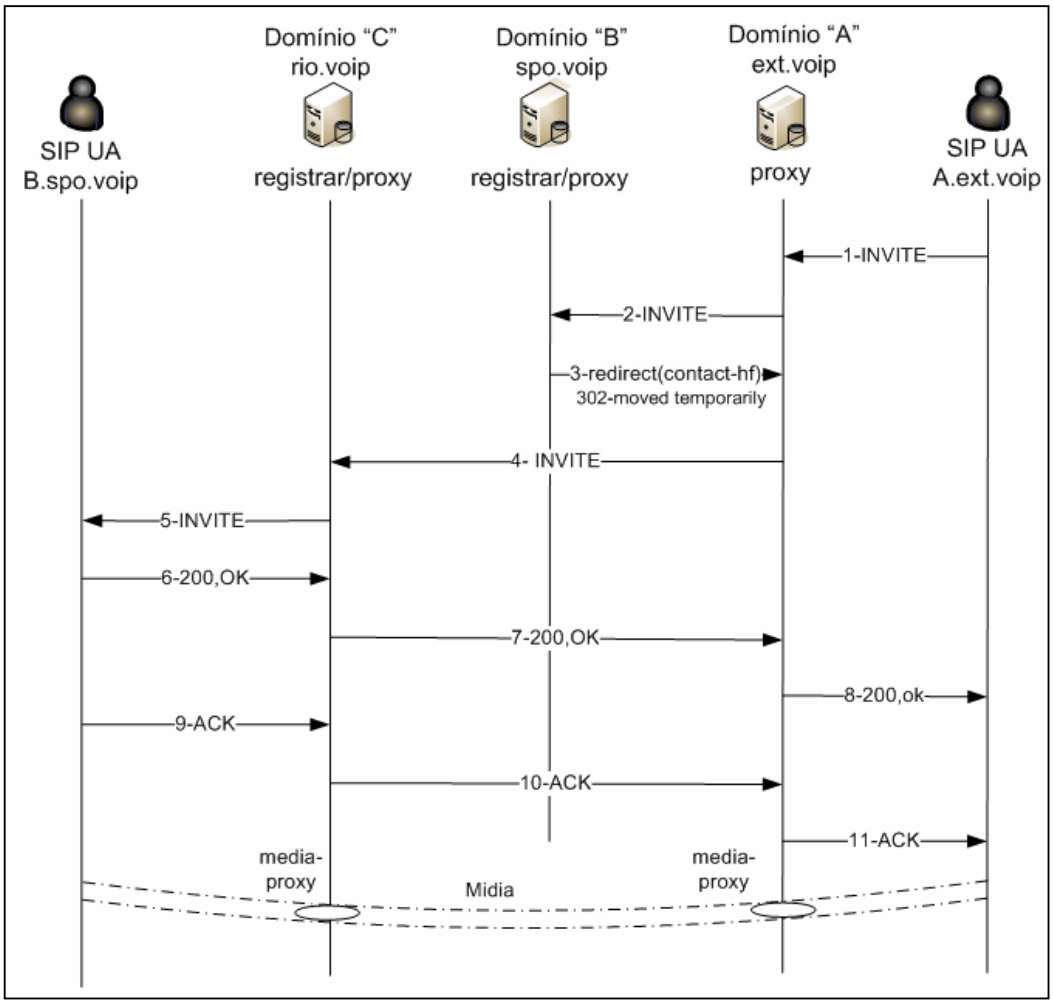


Figura 5.5.1 – Recebendo uma Chamada

Fluxo de mensagens da Figura 5.5.1:

1) Um UA A.ext.voip.br realiza uma chamada para o UA B.spo.voip.br, que está em mobilidade em outro domínio, no domínio RIO (rio.voip.br). Esta requisição é resolvida pelo DNS/SRV encaminhada para o domínio home spo.voip.br , o *proxy* B.spo.voip.br;

2) O *proxy home* do domínio spo.voip.br, ao receber o *INVITE* para o UA B.spo.voip.br, verifica se esse UA está registrado e qual é a *flag* da coluna *ismobility* da tabela *subscriber*. Sendo igual a "1", deverá realizar o redirecionamento, consultando a tabela de localização para o endereço lógico(ip addr) associado;

3) O *proxy home* do domínio B.spo.voip.br (spo.voip.br) faz uso de uma função *uac_redirect()* que, baseado em filtros (campo de cabeçalhos, ip_addr ou variáveis definidas na configuração), realiza o roteamento através do *script* de configuração do *proxy* para fazer a requisição *INVITE* ser enviada para o *proxy* do domínio rio.voip.br. Ou seja, envia um 3xx - *REDIRECT* para o *proxy* A.ext.voip.br, removendo o campo de origem através do campo de cabeçalho "*via"* e preenchendo o campo de cabeçalho *contact* com o endereço lógico associado ao *proxy* rio.voip.br, localização do UA(B.spo.voip.br);

4) O *proxy* externo, ao receber o *REDIRECT* (com o campo do cabeçalho do URI *contact* redefinido para o proxy rio.voip.br), remove o campo *contact* referente ao UA B.spo.voip.br, que é o endereço lógico (endereço IP) do *proxy* B.spo.voip.br , e envia uma nova requisição *INVITE* ao destino;

5) O *proxy* rio.voip.br, ao receber a requisição *INVITE* , validará a URI (sip: B.spo.voip.br) na base de dados na tabela localização. Pelo procedimento normal do SIP, não seria encontrado a URI do UA B.spo.voip.br, retornando um *404-Not found.* Mas, como está sendo usada uma configuração personalizada nos servidores *Proxy/registrar,* o URI (*sip:B@spo.voip.br*) será encontrado na base de dados, na tabela localização e com isso é encaminhado ao UA(B.spo.voip.br) em mobilidade pelo endereço lógico IP associado a esta URI;

6) UA B.spo.voip.br, ao receber o *INVITE,* retornará um 200,ok;

7) O *proxy* do UA B.spo.voip.br ao receber o 200,ok, realizará o *relay* para o *proxy* A.ext.voip.br pelo campo de cabeçalho "VIA";
Observe que sempre que um *proxy* processa uma primitiva, ele preenche seu endereço no campo "VIA", se quiser participar da sinalização;

8) O *proxy ext.voip.br* , ao receber o 200,ok, encaminhará ao UAext;

9) O UA B.spo.voip.br responderá com um *ACK* (negociando parâmetros da sessão);

10) O *proxy* spo.voip.br também encaminhará este ACK àquela origem do *INVITE* ;

11) O *proxy* ext, ao receber o ACK , também encaminhará ao UAext com os parâmetros de sessão disponíveis;

12) Mídia estabelecida, passando pelos *proxies* “A” (ext.voip) e “C” (rio.voip.br) que atuam como proxies de mídia;

```
$ORIGIN rio.voip.br.
$TTL 3600
;rio.voip.br.          IN      SOA     ns.rio.voip.br. root.rio.voip.br.(
@                      IN      SOA     ns.rio.voip.br. root.rio.voip.br.(
                                       1995032001 3600 3600 604800 86400);
@                      IN      NS      ns.rio.voip.br.
@                      IN      A       172.31.12.144
spo.voip.br.           IN      NS      ns.spo.voip.br.
ext.voip.br.           IN      NS      ns.ext.voip.br.
;
rio.voip.br.                   IN  A       172.31.12.144
ns.rio.voip.br.                IN  A       172.31.12.144
ns.spo.voip.br.                IN  A       172.31.12.147
spo.voip.br.                   IN  A       172.31.12.147
ns.ext.voip.br.                IN  A       172.31.12.140
ext.voip.br.                   IN  A       172.31.12.140
;
_sip._udp.rio.voip.br.         IN  SRV  0  0 5060 ns.rio.voip.br.
_sip._tcp.rio.voip.br.         IN  SRV  0  2 5060 ns.rio.voip.br.
_sip._udp.spo.voip.br.         IN  SRV  1  0 5060 ns.spo.voip.br.
_sip._tcp.spo.voip.br.         IN  SRV  1  2 5060 ns.spo.voip.br.
_sip._udp.ext.voip.br.         IN  SRV  1  0 5060 ns.ext.voip.br.
_sip._tcp.ext.voip.br.         IN  SRV  1  2 5060 ns.ext.voip.br.
;
rio.voip.br.           IN      NAPTR   0  0  "s" "SIP+D2U"  ""  _sip._udp.rio.voip.br.
rio.voip.br.           IN      NAPTR   1  0  "s" "SIP+D2T"  ""  _sip._tcp.rio.voip.br.
```

Figura 5.5.2 – Configuração DNS – domínio rio.voip.br

Na Figura 5.5.2 é descrito o arquivo de configuração do DNS, referente ao domínio *rio.voip.br,* utilizado na prova de conceito. Neste caso, que as requisições relacionadas ao serviço VoIP (consultas DNS, SRV) do protocolo SIP serão referenciadas e convertidas para o endereço lógico do domínio correspondente. A fim de facilitar os testes da implementação foram incluídos os três domínios (RIO, SPO e EXT) com seus respectivos endereços lógicos em cada DNS de localidade.

O DNS fornece dois tipos de registro relevantes para as requisições SIP: SRV [GULBRANDSEN, A., VIXIE, P., ESIBOV, L., 2000] e NAPTR [MEALLING, M., DANIEL, R., 2000], sendo que algumas implementações usam apenas registros SRV. Os registros SRV possuem o seguinte formato:

"_Service._Proto.Name [TTL] Class SRV Priority Weight Port Target"
Por exemplo:
_sip._udp.rio.voip.br. 43200 IN SRV 0 0 5060 ns.rio.voip.br."

Dentre os parâmetros acima, vale destacar a prioridade que define a ordem de consulta do *proxy*. Valores mais baixos são consultados primeiro. E o peso similar à prioridade determina a proporcionalidade da frequência em que o *proxy* será consultado. Valores mais altos mais frequentes serão as consultas.

Registros NAPTR (*The Name Authority Pointer*) fornecem um mapeamento a partir do nome de domínio para os registros SRV, com objetivo de contatar um servidor com o protocolo de transporte específico no campo de serviço do NAPTR. Em outras palavras, os registros NAPTR fornecem um mecanismo para um domínio chamado, a fim de especificar qual protocolo utilizar em uma requisição SIP. Os registros NAPTR possuem o seguinte formato:
"domain-name TTL Class NAPTR order preference flags service regexp target"
Por exemplo:
rio.voip.br IN NAPTR 0 0 "s" "SIP+D2U" "" _sip._udp.rio.voip.br.
rio.voip.br IN NAPTR 1 0 "s" "SIP+D2T" "" _sip._tcp.rio.voip.br.

Dentre os parâmetros acima, vale destacar que o IN define a classe como sempre verdadeira (utilização dos registros), ordem é "0" e a preferência também é "0", onde o parâmetro *order* define como os registros serão lidos e em que ordem. Ou seja, no exemplo, a ordem define que primeiro vai resolver pelo registro SRV (dado pelo parâmetro *target*) utilizando o protocolo de transporte *udp* e, caso haja falha, usará o *tcp. O* parâmetro preferência será utilizado, quando todos os campos *order* forem os mesmos, dando aos valores mais baixos uma precedência mais alta. Os demais parâmetros não estão sendo abordados por uma questão de relevância do objetivo deste tópico.

5.6 – Considerações do funcionamento desta implementação com o modelo P2P

Foram realizados testes com um simulador, denominado OverSIM [Baumgart, I., Heep, B., Krause, S., 2007] para emulação de um ambiente P2P com uso de um mecanismo DHT para gerenciar a entrada e saída dos recursos tidos como *proxies* da rede. O simulador OverSIM cria um *overlay* estruturado que é um conjunto de conexões lógicas entre os *proxies*. Redes *overlay* podem ser estruturadas ou não estruturadas. No P2P não estruturado, os dados, no caso os *proxies* dos domínios, seriam distribuídos randomicamente e mecanismos de *broadcasting* seriam utilizados para pesquisa dos recursos. No P2P estruturado, a topologia de rede está fortemente controlada e os recursos são colocados precisamente em locais especificados.

Não foi possível uma implementação completa com o simulador, com tratamento dos fluxos de mídia e as variações que se desdobraram para atender o problema desta dissertação que era a otimização do fluxo de mídia na mobilidade de um UA. Porém foi possível identificar que a solução proposta é viável, se fizermos a substituição do processo de procura e localização do recurso via DNS pelo modelo P2P, como mostra o artigo [BAUMGART, I., 2008] com uso do serviço denominado P2PNS e exemplificado na Figura 5.6.1 abaixo.

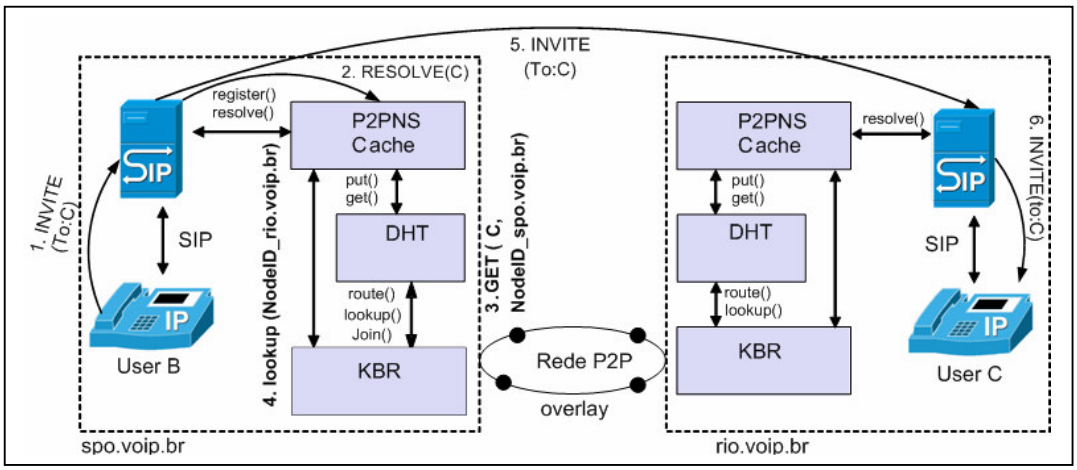


Figura 5.6.1 – Modelo P2PNS (fonte: PERCOM '08: Proc., P2PNS: A Secure Distributed Name Service for P2PSIP, 2008)

A Figura 5.6.1 mostra como um usuário do *peer* spo.voip.br estabelece uma chamada para o AoR (Address-of-Record) do usuário “C”. A princípio, o UA envia uma requisição SIP *INVITE* para o *proxy* P2PSIP local. O *proxy* local consulta P2PNS para resolver a URI de “C” (chamando a função resolve()). A camada do P2PNS primeiro recupera o *nodeID* correspondente ao AoR do “C” a partir do DHT (caso o mapeamento já não esteja em *cache*). Na próxima etapa, o nodeID obtido fica resolvido para o atual endereço IP do *peer* rio.voip.br. Finalmente o SIP *INVITE* é encaminhado para o UA via o *proxy* do *peer* rio.voip.br.

Algumas considerações são relevantes no contexto da arquitetura P2P para uso com um serviço VoIP. O modelo adotado é sempre referente a um serviço global, de preferência comercial e não visto como um modelo de serviço fechado como o do fone@RNP. E neste aspecto as arquiteturas P2P com SIP apresentam a questão da autenticação de seus usuários de forma centralizada, como descrito no artigo [SINGH, K. e SCHULZRINNE, H., 2005]. Uma citação descreve como desafio a proposta de um modelo descentralizado para autenticação dos usuários do serviço, conforme transcrição abaixo:

[...] *A reliable framework for authentication without centralized elements is a challenge*[...].[SINGH, K. e SCHULZRINNE, H., 2005]

Trata-se de uma oportunidade para pesquisas mais aprofundadas na questão de se utilizar tal modelo para explorar a descentralização da questão de autenticação, visto que a arquitetura descentralizada do P2P dá suporte ao processo de uso de rotas ótimas em relação aos fluxos de mídia das chamadas.

5.7 – Dificuldades de realização do ambiente virtualizado

Para validação da proposta foram utilizadas máquinas virtuais do tipo vmware junto com o OpenSER que operam como *proxy* de sinalização VoIP, servidor registrar e servidor de redirecionamento. Os serviços foram executados em ambiente virtualizado, porém na mesma infraestrutura de rede local. Como os testes ocorreram dentro de uma rede interna, os atrasos caracterís-

ticos são relacionados ao tempo de acesso à rede interna (LAN comutada) e entre as máquinas físicas nesta rede.

A montagem dessas máquinas virtuais com os recursos utilizados, como pouca memória RAM e capacidade de disco pequena foram um desafio para se instalar o software OpenSER como *proxy* de sinalização, mídia e redirecionamento. Foi necessário ir compilando e instalando só os módulos verdadeiramente essenciais para a solução. Os demais, como benchmark (ajuda aos desenvolvedores), cpl-c (linguagem de processamento de chamadas via xml), db_berkeley (banco de dados), dispatcher (balanceamento de carga), foram retirados do *makefile.,* com objetivo de não ser compilado e nem realizar a instalação. Desta forma, a instalação ficou customizada e de acordo com a infraestrutura.

Entre tantas dificuldades encontradas vale ressaltar o funcionamento do programa OpenSER com seus blocos de roteamento, que são tratados pelo script de configuração através de funções internas de seu núcleo principal, pois, apesar de vasta a documentação e muitas funções disponibilizadas para uma variedade de objetivos, nos testes, muitas destas funcionalidades apresentaram bugs, e não funcionavam de acordo com o esperado. Assim se fez necessário recorrer aos fóruns na internet, cujos desenvolvedores que davam suporte ao produto disponibilizavam *fixes* e *patches* para as correções.

Capítulo 6

Conclusão e propostas futuras

Neste trabalho foi apresentado o registro distribuído em uma arquitetura SIP cliente/servidor e, como conseqüência, foi obtida uma rota com um percurso mais favorável da mídia. A ineficiência descrita na introdução desta dissertação ocorre porque, em uma arquitetura SIP convencional, um usuário em mobilidade não é diferenciado de um usuário local, pois todo o processo de localização e direcionamento de mídia é realizado em consultas à instituição origem. O tráfego necessariamente passa pelos servidores de *proxy* das instituições desses usuários, quando eles são utilizados como *proxies* de mídia, o que é bastante comum. E, quando não são utilizados como proxies de mídia, possuem o comportamento tradicional em que há o modelo de comunicações tido como trapezoidal. O objetivo foi manter a mecânica da arquitetura com a autenticação e validação do registro do usuário em mobilidade, e permitir que esse usuário possa se registrar na instituição em que estiver, conseguindo que a mídia flua por essa nova localidade, tendo enfim a otimização da arquitetura.

O desenvolvimento da proposta foi contemplado no capítulo 4 e no capítulo 5 foi detalhada a troca de primitivas através de diagramas dos procedimentos principais: *REGISTER*, recebimento e envio de chamadas por parte do usuário em mobilidade.

A solução proposta foi implementada, preservando a operação normal SIP dos agentes de usuário, nos quais foram realizadas apenas modificações nos *proxies* SIP para realizar o conceito de registro distribuído e localização numa federação. A validação da proposta através de implementação comprovou a plena viabilidade e eficácia da solução. As modificações no código do *proxy* SIP OpenSER não foram substanciais e não houve impacto na *performance* do proxy, que é projetado para grande escalabilidade.

Aproveitando o uso da sinalização SIP, foi estendido o processo de registro para contemplar o repasse de créditos, que podem ser usados de forma arbitrária dentro da federação. O conceito de federação foi montado com

o uso de tabelas com a inclusão dos *proxies* federados das instituições, sendo que as primitivas de sinalização podem ser trocadas de forma confiável.

Para a prova de conceito foi usado um servidor SIP, também utilizado em um serviço com abrangência nacional, o fone@RNP. Este servidor SIP conhecido como OpenSER possui uma arquitetura modular e opera baseado em um arquivo de configuração e módulos plugáveis, de acordo com as funcionalidades esperadas para cada tipo de aplicação. Foram introduzidas modificações nessa sua estrutura, com a reutilização do módulo responsável pelo tratamento do URI dos usuários e com a inclusão de uma nova função para a validação das instituições no momento do registro desses usuários. Com a consulta numa base de dados contendo os domínios dessas instituições participantes da federação. Isso é possível através de um processo de replicação de dados, que por acaso já existe na arquitetura do fone@RNP. O uso dessa função modificada neste trabalho visa à validação, encaminhando a requisição de registro dos usuários as suas instituições de origem desde que estes usuários estejam em mobilidade.

Como consequência deste processo de registro descentralizado, com validação no domínio de origem teve um caminho otimizado para a mídia. Para dar suporte a essa solução foi incluída no modelo de dados do serviço registrar das instituições utilizadas na prova de conceito uma nova tabela, denominada *trusdomain*, visando à verificação no momento do registro dos usuários em mobilidade, além da alteração da estrutura de outras tabelas, como a de localização e a de assinantes (*subscriber*). Com objetivo de modelar a questão do registro distribuído, foi necessário montar uma lógica de roteamento baseado no *script* de configuração do serviço para realização de requisições e de respostas. Como resultado final foi minimizada a ineficiência do fluxo de mídia por um caminho ótimo entre os participantes de um diálogo, alcançando assim o objetivo proposto.

Com relação a trabalhos futuros uma proposta é a implementação de uma extensão da arquitetura P2P com uso do modelo P2PNS, cujo objetivo seria dar suporte à mobilidade dos usuários em um serviço de telefonia IP. Através desta recomendação, de uma proposta, será utilizado a validação por autenticação remota, de forma distribuída com uso de um simulador, que está baseado na ferramenta Omnet++ [Varga, A.,2009].

REFERÊNCIAS

ANDREASEN, F.; FOSTER, B.; RFC 3435 – **"Media Gateway Control Protocol"**, Janeiro de 2003.

ARANGO, M.; DUGAN, A.; ELLIOTT, I; HUITEMA, C.; PICKETT, S.; IETF, RFC 2705 – "**MGCP : Media Gateway Control Protocol"**. Outubro de 1999.

BASET, S. A., SCHULZRINNE, H. **"An Analysis of the Skype Peer-to-Peer Internet Telephone Protocol"**, 15 de setembro de 2004. IEEE Infocom 2006.

BAUGHER, M. , NASLUND, M. , CARRARA, E. , NORRMAN, K. , **RFC 3711 - The Secure Real-time Transport Protocol (SRTP),** de março de 2004, Disponível em < http://tools.ietf.org/html/rfc3711>, Acesso em Março de 2009.

BAUMGART, I., 2008, **"P2PNS A Secure Distributed Name Service for P2PSIP"**, PERCOM '08: Proceedings of the 2008 Sixth Annual IEEE International Conference on Pervasive Computing and Communications - Volume 00 (March, 2008).

BAUMGART, I., HEEP, B., KRAUSE, S., 2007. **"OverSim: A Flexible Overlay Network Simulation Framework"**. Proceedings of 10th IEEE Global Internet Symposium (May, 2007). p.79—84.

BRYAN, D. A., LOWEKAMP, B. B., JENNINGS, C., **"SOSIMPLE: A Serverless, Standards-based"**, P2P SIP Communication System (pdf), International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications (AAA-IDEA), June 2005;

CANTOR S.; "**Shibboleth architecture: protocols and profiles"**, February 2005. Disponível em: http://shibboleth.internet2.edu/docs/draft-mace-shibboleth-arch-protocols-06.pdf, acesso em janeiro de 2008.

CHOUDHURY, G. L.; COLE; R.G. Robert; **"Design and Analysis of optimal adaptive de-jitter buffers"**, 2004, Elsevier Computer Science – Computer Communications 27(6), de 2004

DIERKS, T.; RESCORLA, E.; RFC 4346 – **"The Transport Layer Security (TLS) Protocol"**; Abril de 2006.

ENDLER, D.; GHOSAL, D.; JAFARI, R.; KARLCUT, A.; KOLENKO, M.; ZAR, J.; **"Voice Security Alliance, "VoIP Security and Privacy Threat Taxonomy"**, Disponível em: <http://www.voipsa.org/Activities/taxonomy.php>, Acesso em: Novembro de 2008.

FACCIN, S.M., LALWANEY, P. e PATILI, B. (2004) **"IP multimedia services: analysis of mobile IP and SIP interactions in 3G networks"**, IEEE Communications Magazine, Vol. 8, No. 1, pp. 113-118.

FALSTROM, P.; **"RFC 2916 - E.164 Number and DNS"**, IETF., Setembro de 2000. Disponível em: http://www.ietf.org/rfc/rfc2916.txt. Acesso em março de 2009.

FRANKS, J.; HALLAM-BAKER, P.; HOSTETLER, J.; LAWRENCE, S.; LEACH, P.; "RFC 2617 – **HTTP Authentication: Basic and Digest Access Authentication"**, junho de 1999.

GOMEZ, M., Megias, J.L., Bueno, C., Brocal, C., **"Interworking between the Multimedia Messaging Service (MMS) and the 3G IP Multimedia subsystem (IMS) Instant Messaging Service"**, IEEE 16th International Symposium on Personal, Indoor and Mobile Radio communications, (2005), pp: 22742-278.

GONÇALVES, F. A.; **Livro:Telefonia IP com SIP-Abordando SIP Express Router**; 292 p, Abril de 2005. ; ISBN: 978-85-906904-1-2.

GULBRANDSEN, A., VIXIE, P., ESIBOV, L. – **"RFC 2782 - A DNS RR for specifying the location of services (DNS SRV)"**, Fevereiro de 2000, Disponível em: < http://www.ietf.org/rfc/rfc2782.txt >, Acesso em Março de 2009.

H.323 – **ITU-T Recommendation H.323, Packet-Based Multimedia Communications Systems**, 1998. Disponível em: <http://www.itu.int/rec/T-REC-H.323/e > Acesso em Março de 2009. Versão 2 do H.323. Disponível em < http://www.itu.int/rec/T-REC-H.323-199802-S/en/> Acesso em março: 2009.

HANDLEY, M.; SCHULZRINNE, H; SCHOOLER, E. e ROSENBERG, J.; **RFC 2543 – "SIP: Session Initiation Protocol"**, março de 1999. Disponível em: < http://www.ietf.org/rfc/rfc2543.txt>. Acesso em Março de 2009.

HANDLEY,M.; e JACOBSON,V.; **RFC 2327 - "SDP: Session Description Protocol"**, Internet Engineering Task Force, Abril, 1998. Disponível em: <http://www.ietf.org/rfc/rfc2327.txt> . Acesso em Março de 2009.

HERSENT, O.; GUIDE, D., PETIT, JEAN-PIERRE; 2005, Livro: **"IP Telephony: Deploying Voice-over-IP Protocols"** Ed. Pearson Education Limited (Addison-Wesley); ISBN: 978-0470023594.

IANCU, A. B., MIERLA, D. C., MODROIU, E. R., et al. "**OpenSER - Servidor SIP"** de código fonte aberto (junho/2005); Disponível em: < http://www.voip-info.org/wiki/view/Kamailio#Features> , Acesso em: Março de 2009.

IANCU, A. B., PASCU, D., et al. **"Projeto OpenSIPS – Servidor SIP com origem no OpenSER"** com código fonte aberto (2007); Disponível em:<http://www.opensips.org >, Acesso em Março de 2009.

KENT, S.; **RFC 1422 – "Privacy Enhancement for Internet Electronic Mail – Part II: Certificate-Based Key Management"**, fevereiro de 1993.

LIDINSKY, P. W., SEAMAN, M., JEFFREE, T., et al., **"802.1q – Standard IEEE"** 8/12/1998. Disponível em: <http://standards.ieee.org>, Acesso em: Março de 2009.

LLOYD,B. , SIMPSON, W. , **RFC 1334 – PPP Auhentication Protocols, Outubro de 1992**, Disponível em: <http://tools.ietf.org/html/rfc1334> Acesso em Março de 2009.

LOUGHNEY, J.; CAMARILLO, G.; **RFC 3702 – "Authentication, Autorization and Accounting – Requirements for the Session Initiation Protocol (SIP)"**, Fevereiro de 2004.

LUSTOSA, L.C.G.; RODRIGUES, P.H.A; DAVID, F; QUINELLATO, D.G. **"Arquitetura de Monitoração de Qualidade de Chamadas Telefônicas IP"**. XXIII Simpósio Brasileiro de Redes de Computadores (SBRC'2005), Fortaleza-CE, Maio de 2005.

MEALLING, M., DANIEL, R., RFC 3403 – **"Dynamic Delegation Discovery System (DDDS) - Part Three: The Domain Name System (DNS) Database",** Outubro de 2002, Disponível em <http://tools.ietf.org/html/rfc3403>, Acesso em Março de 2009.

Microsoft Active Directory Architeture (AD), Disponível em: <http://technet.microsoft.com/en-us/library/bb727030.aspx>, Acessso em Março de 2009.

MIERLA, D. C.,VAMANU, A., INAKI, C. B., et al. **"Projeto Kamailio – Servidor SIP com origem no OpenSER"** com código fonte aberto (2007); Disponível em:<http://www.kamailio.org >, Acesso em Março de 2009.

MIERLA, D., IANCU, A. B., HOFFMANN, A., et al., **"SIP Express Route - SER"**, Disponível em: <http:// http://www.iptel.org/ser_history> , Acesso em novembro de 2008.

NEUMAN, C., HARTMAN, S., RAEBURN, K., **RFC 4120 - The Kerberos Network Authentication Service (V5, Julho de 2005,** Disponível em: http://tools.ietf.org/html/rfc4120. Acesso em Março de 2009.

RAMJEE, R; KUROSE, J.; TOWSLEY, D.; SCHULZRINNE, H.; **"Adaptative Playout Mechanisms for Packetized Audio Applications in WAN"**, networks, in Proc. 13th IEEE Infocom Conference on Computer Communications (Infocom '94), pp. 680–688, Toronto, Ontario, Canada, Junho de 1994.

RIGNEY, C., WILLENS, S., RUBENS, A. et al., **"Remote Authentication Dial In User Service (RADIUS)"**, **RFC 2865**. Junho de 2000.

ROSENBERG, J.; SCHULZRINNE, H., CAMARILLO, G., et al.; **RFC 3261 - "SIP: Session Initiation Protocol"**, Junho de 2002. Disponível em: <http://www.ietf.org/rfc/rfc3261.txt>, Acesso em: Março de 2009.

SALSANO, S.,VELTRI L. e PAPALILO D. "**SIP security issues: the SIP authentication procedure and its processing load"**. IEEE Network, Volume 16, Issue 6, Nov-Dec 2002. Page(s): 38-44.

SECSIP, **"A Stateful SIP Protection System; Open Source SIP Networks Protection System"**, Disponivel em: <http://secsip.gforge.inria.fr/doku.php?id=projects>, Acesso em: Nov/2008.

SIMON, D., ABOBA, B., SIMON, D., ERONEN, P. , **RFC 5247 - Extensible Authentication Protocol (EAP) Key Management Framework,** Agosto de 2008. Disponível em < http://tools.ietf.org/html/rfc5247 >, Acesso em Março de 2009.

SIMPSON, W; **RFC 1994 –** "**PPP Challenge Handshake Authentication Protocol"**, Agosto de 1996.

SINGH, K. e SCHULZRINNE, H.; **"Using an External DHT as a SIP Location Service"**; University of Columbia, 2006, Acesso em 8/10/2007; https://mice.cs.columbia.edu/getTechreport.php?techreportID=388.

SINGH, K. e SCHULZRINNE, H.; **"SIPPeer: A SIP-based peer-to-peer Internet Telephony Client Adaptor"**; ACM Press; janeiro de 2005.

SINGH, K.; e SCHULZRINNE, H., **"Peer-to-peer Internet telephony using SIP,"** in Proc. International Workshop on Network and Operating System

Support for Digital Audio and Video (NOSSDAV), Skamania,Washington, June 2005.

VARGA, A., "**Omnet++ community site"**. [Online]. Disponível: http://www.omnetpp.org/, Acesso em março de 2009.

WAHL, M.; KILLE, S. e HOWES,T. IETF RFC 2251 – **LDAP: Lightweight Directory Access Protocol (v3)**.Dezembro de 1997.

ZEILENGA, K., **"Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map"**,RFC 4510, junho de 2006.

Apêndice A – Descrição do ambiente utilizado

O apêndice trata do detalhamento das estruturas utilizadas na prova de conceito, como os scripts de configuração do serviço, os códigos das funções criadas e/ou modificadas relacionadas com as operações de registro, estabelecimento da chamada e recebimento da chamada e da estrutura das tabelas criadas e/ou modificadas do modelo de dados do servidor proxy/ registrar/redirecionamento.

Descrição do ambiente utilizado:
Máquinas Virtuais: VMWare Server;
Servidor de Banco de Dados: MySQL 5.1;
Serviço de DNS: BIND9;
Sistema Operacional: Linux Kernel 2.6 - Ubuntu 7.04;
Servidor SIP: OpenSER versão 3.0;

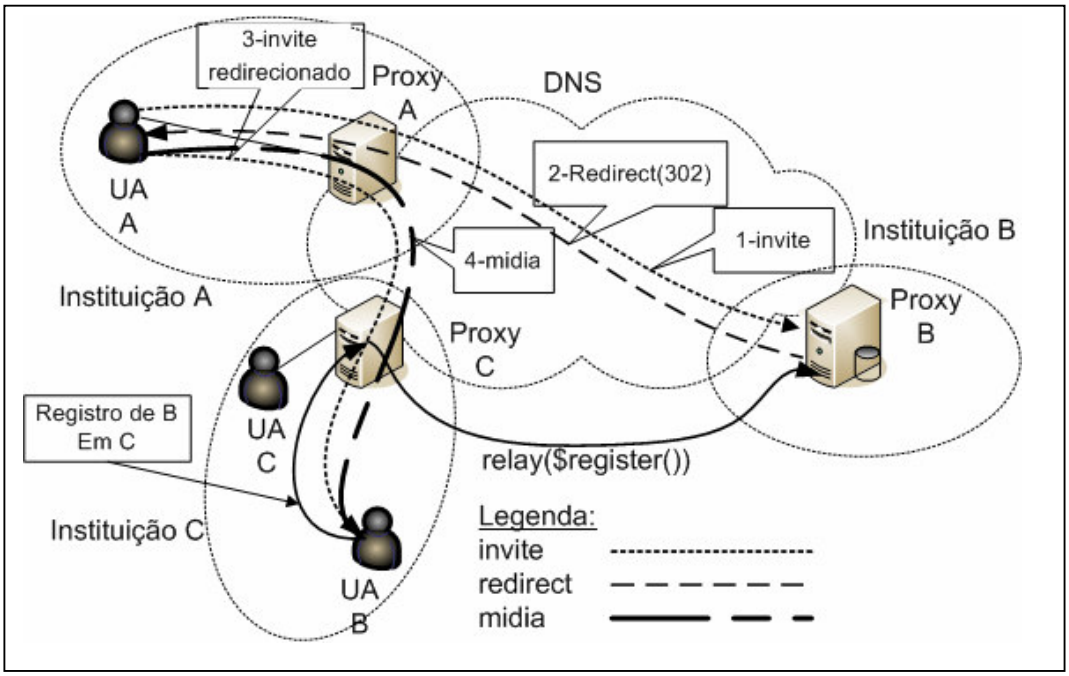


Figura A.1: Rede SIP com otimização proposta

A Figura A.1 representa a arquitetura otimizada com o encaminhamento da mídia sendo tratado pelo domínio mais próximo de onde o usuário em

mobilidade se encontra. Com isso, o objetivo de resolver a questão de ineficiência inerente ao trânsito da mídia pela rede de forma a acompanhar os controles, primitivas/métodos do protocolo SIP foram alcançados com a utilização de softwares com o código fonte aberto, seguindo a orientação GNU/GPL. Na demonstração da prova de conceito foi utilizado, o OpenSER 1.3. A partir deste ponto, os apêndices terão a função de mostrar o código utilizado na implementação desta prova de conceito.

Função criada e adicionada no *core* do servidor registrar do OpenSER, para validação dos domínios/Realms nas requisições de registro dos usuários em mobilidade: Figura A.3 - *is_from_trusted()*

```
/*
 *  * Verifica se o domain no Header From da URI eh
trusted
 *
 */
int is_from_trusted(struct sip_msg* _msg, char* _s1, char*
_s2)
{
        struct sip_uri *puri;
/*      struct str* _host;  */
        str table_trus;
        str col_id;
        str col_dom;
        str col_lastmod;

        str* _host = (str*)(&(puri->host));
        char* db_url = DEFAULT_RODB_URL;

        db_key_t keys[1];  /* Tipo da coluna .. Nome das
colunas */
        db_val_t vals[1];
        db_key_t cols[1];
        db_res_t* res = NULL;

        if ((puri = parse_from_uri(_msg)) == NULL){
                LM_ERR("Error while parsing From
header\n");       return -2;
        }

        LM_DBG(">/domain/domain.c>>1<<Antes da fc
bind_dbmod(db_url,&domain_dbf)\n");
        /* Bind the database module */
        if (bind_dbmod(db_url, &domain_dbf)){
            LM_ERR("Cannot bind to database
module!\n");
                return -1;
        }

        LM_DBG(">/domain/domain.c>>2<<Antes da fc
DB_CAPABILITY(domain_dbf,DB_CAP_QUERY)\n");
        /* Check for SELECT capability  */
        if (!DB_CAPABILITY(domain_dbf,
DB_CAP_QUERY)){
                LM_ERR("Database modules does not
provide all functions needed\n");
                return -1;
        }

        LM_DBG(">/domain/domain.c>>3<<Antes da fc
dbhandle\n");
        /* Connect to DB  */
        db_handle = domain_dbf.init(db_url);
        if (!db_handle){
                LM_ERR("failed to connect database\n");
                return -1;
        }

        LM_DBG(">/domain/domain.c>>4<<Antes das
atribuicoes da tabela\n");
        /* Prepare the date for the query */
        table_trus.s = "trusdomain";
        table_trus.len = 11;

        col_id.s = "id";
        col_id.len = 2;
        col_dom.s = "trusdomain";
        col_dom.len = 10;
        col_lastmod.s = "last_modified";
        col_lastmod.len = 13;

        keys[0] = col_id.s;
        cols[0] = col_dom.s;

        LM_DBG(">>domain.c>>5<<IF
RES_ROW_N(res)=>%d\n",RES_ROW_N(res));
        /* Estrutura dos vls das colunas - arq. db/db_val.h
*/
        VAL_TYPE(vals) = DB_STR;
        VAL_NULL(vals) = 0;

        VAL_STR(vals).s = col_dom.s;            /*
_host->s; */
        VAL_STR(vals).len = col_dom.len;        /*
_host->len; */

        LM_DBG(">>domain.c>>6<<connector
db_handle e trusdomain_table.s\n");
        if (domain_dbf.use_table(db_handle, table_trus.s)
< 0) {
                LM_ERR("Error while trying to use
trusdomain table\n");
                return -1;
        }

        LM_DBG(">>domain.c>>7<<Antes da consulta
query\n");
        if (domain_dbf.query(db_handle, keys, NULL,
vals, cols, 1, 1, 0, &res) < 0) {
                LM_ERR("Error while querying
database\n");
                domain_dbf.close(db_handle);
                return -1;
        }

        LM_DBG(">>domain.c>>8<<Antes da validacao
do fetch\n");
        /* RES_ROW_N(res) Num. de linhas no fetch */
        if (RES_ROW_N(res) <= 0 ||
RES_ROWS(res)[0].values[0].nul != 0){
                LM_DBG("no value found\n");
                if (res != NULL &&
domain_dbf.free_result(db_handle, res) < 0)
                        LM_DBG("failed to free the
result\n");
                domain_dbf.close(db_handle);
        } else if (RES_ROW_N(res) >= 1) {
                /* Teste do que estah na tab. trusdomain
com o que vem na URI */
                LM_DBG("Realm '%.*s' is
trusted\n",_host->len, ZSW(_host->s));
        }

        LM_DBG(">>domain.c>>9<<Antes da liberacao
da conexao db\n");
        /* Free the result */
        domain_dbf.free_result(db_handle, res);
        domain_dbf.close(db_handle);

        return 0;
}
```

Figura A.3: Função is_from_trusted()

Apêndice B – Cenário de registro do UA

Este apêndice trata da captura do processo de registro de um UA em mobilidade através dos encaminhamentos disparados pelos *proxies* de registro, conforme demonstra o log da Figura B.1 e o detalhamento deste log nas Figuras B.3 a B.10.

Log do Cenário 1 em relação ao registro de um UA em mobilidade:
Endereço IP do UA (domínio spo.voip.br) em mobilidade: 172.31.12.142/24
Endereço IP do Registrar (rio.voip.br): 172.31.12.144/24
Endereço IP do Registrar (spo.voip.br): 172.31.12.147

```
0.514324 172.31.12.142 -> 172.31.12.144 SIP Request: REGISTER sip:spo.voip.br
0.624737 172.31.12.144 -> 172.31.12.147 SIP Request: REGISTER sip:spo.voip.br
0.834544 172.31.12.147 -> 172.31.12.144 SIP Status: 401 Unauthorized    (0 bindings)
0.836937 172.31.12.144 -> 172.31.12.142 SIP Status: 401 Unauthorized    (0 bindings)
0.861845 172.31.12.142 -> 172.31.12.144 SIP Request: REGISTER sip:spo.voip.br
0.863010 172.31.12.144 -> 172.31.12.147 SIP Request: REGISTER sip:spo.voip.br
1.047046 172.31.12.147 -> 172.31.12.144 SIP Status: 200 OK    (1 bindings)
1.049832 172.31.12.144 -> 172.31.12.142 SIP Status: 200 OK    (1 bindings)
```

Figura B.1: Log do register por saltos

A Figura B.1 mostra o processo de registro por saltos entre as entidades envolvidas no registro, um UA em mobilidade, o servidor registrar mais próximo e o servidor registrar remoto, em que ficam demonstrados o encaminhamento e a validação do servidor registrar mais próximo do UA, em mobilidade com o seu domínio remoto (domínio *home* do UA).

```
###
# Tratamento das mensagens REGISTER de UAc em mobilidade
###
route[2] {
        # Verifica a partir da URI do REQUEST o dominio("from_domain") deste UAc se eh confiavel
    xlog("L_ALERT","URI da Autorizacao - ($adu) o Realm - ($ar) o domain de destino - ($td) e o de origem ($fd)\n");
        xlog("L_ALERT","($src_ip-0)\n"); #End.IP do UAc#
        if (is_from_trusted()){
                xlog("L_ALERT","Domain eh trusted ($fd)\n");
                t_relay($register("udp:172.31.12.147:5060"));
                t_on_reply("1");
                xlog("L_ALERT","Retorno do 147 - reply status-($rs) - reply reason-($rr)\n");
                return;
        } else if (!www_authorize("rio.voip.br", "subscriber")) {
                xlog("L_ALERT", "REGISTER ($fU)($ru) nao autorizado\n");
                log(1, "Enviando 401 - Unauthorized\n");
                www_challenge("rio.voip.br", "1");
                return;
        }

        # Verifica campo "To" username contra a tab. URI ??
        if (!check_to()) {
                xlog("L_ALERT", "REGISTER ($fU) ($ru) com username invalido\n");
                log(1, "Enviando 401 - Unauthorized (Username)\n");
                sl_send_reply("401","Unauthorized");
                return;
        }
        # Para verificacao de IP na Tab. subscriber
        if ($avp(s:src_ip) != $si) {
                sl_send_reply("403","Forbidden IP");
                return;
        }
        xlog("L_ALERT", "REGISTER ($fU) ($ru) com username valido\n");
        # Remove as Credenciais das mensagens processadas afim de que o proxy nao revele tais infos
        consume_credentials();
        # The function processes a REGISTER message. It can add, remove or modify usrloc records depending on
        # Contact and Expires HFs in the REGISTER message. On success, 200 OK will be returned listing all
contacts
        # that are currently in usrloc. On an error, error message will be send with a short description in reason
        # phrase.
        if (!save("location")) {
                xlog("L_ALERT","Nao pode salvar REGISTER, na USRLOC\n");
                sl_reply_error();
        }
        return;
}
```

Figura B.2 - Procedimento de registro com validação se a origem é confiável

A Figura B.2 mostra o uso das funções criadas/modificadas no procedimento de registro tratado pelo proxy *registrar.*

Log detalhado da Figura B.1, representando a requisição de registro do usuário em mobilidade com os servidores *registrars*.

Foram realizados oito *screenshots* representando o *handshake* entre o UA em mobilidade, o servidor registrar mais próximo (rio.voip.br) e o servidor *registrar* remoto (spo.voip.br).

```
Frame 105 (472 bytes on wire, 472 bytes captured)
    Arrival Time: Sep 20, 2008 16:35:23.458020000
    [Time delta from previous packet: 0.000006000 seconds]
    [Time since reference or first frame: 103.759749000 seconds]
    Frame Number: 105
    Packet Length: 472 bytes
    Capture Length: 472 bytes
    [Frame is marked: False]
    [Protocols in frame: eth:ip:udp:sip]
Ethernet II, Src: Vmware_e4:94:5d (00:0c:29:e4:94:5d), Dst: Vmware_ce:24:8e (00:0c:29:ce:24:8e)
    ...
Internet Protocol, Src: 172.31.12.142 (172.31.12.142), Dst: 172.31.12.144 (172.31.12.144)
    Version: 4
    Header length: 20 bytes
    ...
    Source: 172.31.12.142 (172.31.12.142)
    Destination: 172.31.12.144 (172.31.12.144)
User Datagram Protocol, Src Port: sip (5060), Dst Port: sip (5060)
    Source port: sip (5060)
    Destination port: sip (5060)
    Length: 438
    Checksum: 0x6988 [correct]
Session Initiation Protocol
    Request-Line: REGISTER sip:spo.voip.br SIP/2.0
        Method: REGISTER
        [Resent Packet: False]
    Message Header
        Via: SIP/2.0/UDP 172.31.12.142:5060;rport;branch=z9hG4bK0FD1E2887C899517FC64B083ED03DBBB
            Transport: UDP
            Sent-by Address: 172.31.12.142
            Sent-by port: 5060
            RPort: rport
            Branch: z9hG4bK0FD1E2887C899517FC64B083ED03DBBB
        From: 2115 <sip:vmwcli01@spo.voip.br>;tag=1553660366
            SIP Display info: 2115
            SIP from address: sip:vmwcli01@spo.voip.br
            SIP tag: 1553660366
        To: 2115 <sip:vmwcli01@spo.voip.br>
            SIP Display info: 2115
            SIP to address: sip:vmwcli01@spo.voip.br
        Contact: "2115" <sip:vmwcli01@172.31.12.142:5060>
            Contact Binding: "2115" <sip:vmwcli01@172.31.12.142:5060>
                URI: "2115" <sip:vmwcli01@172.31.12.142:5060>
                    SIP Display info: "2115"
                    SIP contact address: sip:vmwcli01@172.31.12.142:5060
        Call-ID: 1C5DF1047A8B10015F4A5B9346CF6184@spo.voip.br
        CSeq: 8054 REGISTER
            Sequence Number: 8054
            Method: REGISTER
        Expires: 1800
        Max-Forwards: 70
        User-Agent: X-Lite release 1105d
        Content-Length: 0
```

Figura B.3: Log do register entre UAmovel e os proxies rio.voip.br e spo.voip.br

```
Frame 108 (596 bytes on wire, 596 bytes captured)
...
   [Protocols in frame: eth:ip:udp:sip]
Ethernet II, Src: Vmware_ce:24:8e (00:0c:29:ce:24:8e), Dst: Giga-Byt_9b:5c:2e (00:0f:ea:9b:5c:2e)
...
Internet Protocol, Src: 172.31.12.144 (172.31.12.144), Dst: 172.31.12.147 (172.31.12.147)
   Version: 4
   Header length: 20 bytes
...
   Source: 172.31.12.144 (172.31.12.144)
   Destination: 172.31.12.147 (172.31.12.147)
User Datagram Protocol, Src Port: sip (5060), Dst Port: sip (5060)
   Source port: sip (5060)
   Destination port: sip (5060)
   Length: 562
   Checksum: 0xc6fb [correct]
Session Initiation Protocol
   Request-Line: REGISTER sip:spo.voip.br SIP/2.0
      Method: REGISTER
      [Resent Packet: False]
   Message Header
      Record-Route: <sip:172.31.12.144;lr=on;ftag=1553660366>
      Via: SIP/2.0/UDP 172.31.12.144;branch=z9hG4bKb886.b3d4ef94.0
         Transport: UDP
         Sent-by Address: 172.31.12.144
         Branch: z9hG4bKb886.b3d4ef94.0
      Via: SIP/2.0/UDP
172.31.12.142:5060;rport=5060;branch=z9hG4bK0FD1E2887C899517FC64B083ED03DBBB
         Transport: UDP
         Sent-by Address: 172.31.12.142
         Sent-by port: 5060
         RPort: 5060
         Branch: z9hG4bK0FD1E2887C899517FC64B083ED03DBBB
      From: 2115 <sip:vmwcli01@spo.voip.br>;tag=1553660366
         SIP Display info: 2115
         SIP from address: sip:vmwcli01@spo.voip.br
         SIP tag: 1553660366
      To: 2115 <sip:vmwcli01@spo.voip.br>
         SIP Display info: 2115
         SIP to address: sip:vmwcli01@spo.voip.br
      Contact: "2115" <sip:vmwcli01@172.31.12.142:5060>
         Contact Binding: "2115" <sip:vmwcli01@172.31.12.142:5060>
            URI: "2115" <sip:vmwcli01@172.31.12.142:5060>
               SIP Display info: "2115"
               SIP contact address: sip:vmwcli01@172.31.12.142:5060
      Call-ID: 1C5DF1047A8B10015F4A5B9346CF6184@spo.voip.br
      CSeq: 8054 REGISTER
         Sequence Number: 8054
         Method: REGISTER
      Expires: 1800
      Max-Forwards: 69
      User-Agent: X-Lite release 1105d
      Content-Length: 0
```

Nesta Figura B.4 é mostrado o salto da requisição *register* do UA, que foi encaminhado ao seu *proxy* de origem para autenticação através da função modificada *relay($register())*.

```
Frame 109 (607 bytes on wire, 607 bytes captured)
...
   [Protocols in frame: eth:ip:udp:sip]
Ethernet II, Src: Giga-Byt_9b:5c:2e (00:0f:ea:9b:5c:2e), Dst: Vmware_ce:24:8e (00:0c:29:ce:24:8e)
...
Internet Protocol, Src: 172.31.12.147 (172.31.12.147), Dst: 172.31.12.144 (172.31.12.144)
   Version: 4
   Header length: 20 bytes
...
   Source: 172.31.12.147 (172.31.12.147)
   Destination: 172.31.12.144 (172.31.12.144)
User Datagram Protocol, Src Port: sip (5060), Dst Port: sip (5060)
...
Session Initiation Protocol
   Status-Line: SIP/2.0 401 Unauthorized
      Status-Code: 401
      [Resent Packet: False]
   Message Header
      Via: SIP/2.0/UDP 172.31.12.144;branch=z9hG4bKb886.b3d4ef94.0
         Transport: UDP
         Sent-by Address: 172.31.12.144
         Branch: z9hG4bKb886.b3d4ef94.0
      Via: SIP/2.0/UDP
172.31.12.142:5060;rport=5060;branch=z9hG4bK0FD1E2887C899517FC64B083ED03DBBB
         Transport: UDP
         Sent-by Address: 172.31.12.142
         Sent-by port: 5060
         RPort: 5060
         Branch: z9hG4bK0FD1E2887C899517FC64B083ED03DBBB
      From: 2115 <sip:vmwcli01@spo.voip.br>;tag=1553660366
         SIP Display info: 2115
         SIP from address: sip:vmwcli01@spo.voip.br
         SIP tag: 1553660366
      To: 2115 <sip:vmwcli01@spo.voip.br>;tag=06ce33ac5d943c89b7ef6922ffe1dd1f.fde6
         SIP Display info: 2115
         SIP to address: sip:vmwcli01@spo.voip.br
         SIP tag: 06ce33ac5d943c89b7ef6922ffe1dd1f.fde6
      Call-ID: 1C5DF1047A8B10015F4A5B9346CF6184@spo.voip.br
      CSeq: 8054 REGISTER
         Sequence Number: 8054
         Method: REGISTER
      WWW-Authenticate: Digest realm="spo.voip.br",
nonce="48ebc076448901fb6e912dc04ff04d73d91d94b8", qop="auth"
         Authentication Scheme: Digest
         Realm: "spo.voip.br"
         Nonce Value: "48ebc076448901fb6e912dc04ff04d73d91d94b8"
         QOP: "auth"
      Server: OpenSER (1.3.0-notls (i386/linux))
      Content-Length: 0
```

Figura B.5: Log do register entre UAmovel e os proxies rio.voip.br e spo.voip.br

A Figura B.5 mostra a requisição de resposta do proxy origem do UA em mobilidade para o proxy rio.voip.br, usando o esquema de autenticação *Digest.*

```
Frame 110 (545 bytes on wire, 545 bytes captured)
...
   [Protocols in frame: eth:ip:udp:sip]
Ethernet II, Src: Vmware_ce:24:8e (00:0c:29:ce:24:8e), Dst: Vmware_e4:94:5d (00:0c:29:e4:94:5d)
...
Internet Protocol, Src: 172.31.12.144 (172.31.12.144), Dst: 172.31.12.142 (172.31.12.142)
...
   Source: 172.31.12.144 (172.31.12.144)
   Destination: 172.31.12.142 (172.31.12.142)
User Datagram Protocol, Src Port: sip (5060), Dst Port: sip (5060)
   Source port: sip (5060)
   Destination port: sip (5060)
...
Session Initiation Protocol
   Status-Line: SIP/2.0 401 Unauthorized
      Status-Code: 401
      [Resent Packet: False]
   Message Header
      Via: SIP/2.0/UDP
172.31.12.142:5060;rport=5060;branch=z9hG4bK0FD1E2887C899517FC64B083ED03DBBB
         Transport: UDP
         Sent-by Address: 172.31.12.142
         Sent-by port: 5060
         RPort: 5060
         Branch: z9hG4bK0FD1E2887C899517FC64B083ED03DBBB
      From: 2115 <sip:vmwcli01@spo.voip.br>;tag=1553660366
         SIP Display info: 2115
         SIP from address: sip:vmwcli01@spo.voip.br
         SIP tag: 1553660366
      To: 2115 <sip:vmwcli01@spo.voip.br>;tag=06ce33ac5d943c89b7ef6922ffe1dd1f.fde6
         SIP Display info: 2115
         SIP to address: sip:vmwcli01@spo.voip.br
         SIP tag: 06ce33ac5d943c89b7ef6922ffe1dd1f.fde6
      Call-ID: 1C5DF1047A8B10015F4A5B9346CF6184@spo.voip.br
      CSeq: 8054 REGISTER
         Sequence Number: 8054
         Method: REGISTER
      WWW-Authenticate: Digest realm="spo.voip.br",
nonce="48ebc076448901fb6e912dc04ff04d73d91d94b8", qop="auth"
         Authentication Scheme: Digest
         Realm: "spo.voip.br"
         Nonce Value: "48ebc076448901fb6e912dc04ff04d73d91d94b8"
         QOP: "auth"
      Server: OpenSER (1.3.0-notls (i386/linux))
      Content-Length: 0
```

Figura B.6: Log do register entre UAmovel e os proxies rio.voip.br e spo.voip.br

A Figura B.6 mostra a requisição do proxy spo.voip.br sendo repassada pelo proxy rio.voip.br para o UA que está em mobilidade, para que esse apresente suas credenciais, de acordo com o modelo de autenticação *digest.*

```
Frame 111 (713 bytes on wire, 713 bytes captured)
...
   [Protocols in frame: eth:ip:udp:sip]
Ethernet II, Src: Vmware_e4:94:5d (00:0c:29:e4:94:5d), Dst: Vmware_ce:24:8e (00:0c:29:ce:24:8e)
...
Internet Protocol, Src: 172.31.12.142 (172.31.12.142), Dst: 172.31.12.144 (172.31.12.144)
...
   Source: 172.31.12.142 (172.31.12.142)
   Destination: 172.31.12.144 (172.31.12.144)
User Datagram Protocol, Src Port: sip (5060), Dst Port: sip (5060)
...
Session Initiation Protocol
   Request-Line: REGISTER sip:spo.voip.br SIP/2.0
      Method: REGISTER
      [Resent Packet: False]
   Message Header
      Via: SIP/2.0/UDP 172.31.12.142:5060;rport;branch=z9hG4bK3ADAF0213E3D6BD8F547144C3F537EF3
         Transport: UDP
         Sent-by Address: 172.31.12.142
         Sent-by port: 5060
         RPort: rport
         Branch: z9hG4bK3ADAF0213E3D6BD8F547144C3F537EF3
      From: 2115 <sip:vmwcli01@spo.voip.br>;tag=1553660366
         SIP Display info: 2115
         SIP from address: sip:vmwcli01@spo.voip.br
         SIP tag: 1553660366
      To: 2115 <sip:vmwcli01@spo.voip.br>
         SIP Display info: 2115
         SIP to address: sip:vmwcli01@spo.voip.br
      Contact: "2115" <sip:vmwcli01@172.31.12.142:5060>
         Contact Binding: "2115" <sip:vmwcli01@172.31.12.142:5060>
            URI: "2115" <sip:vmwcli01@172.31.12.142:5060>
               SIP Display info: "2115"
               SIP contact address: sip:vmwcli01@172.31.12.142:5060
      Call-ID: 1C5DF1047A8B10015F4A5B9346CF6184@spo.voip.br
      CSeq: 8055 REGISTER
         Sequence Number: 8055
         Method: REGISTER
      Expires: 1800
      Authorization: Digest
username="vmwcli01",realm="spo.voip.br",nonce="48ebc076448901fb6e912dc04ff04d73d91d94b8",response="83085a000aa23c2792d4c5
7570d40108",uri="sip:spo.voip.br",qop=auth,cnonce="7D1F007277F0AE6E7075F1BA39C783F1",nc=00000001
         Authentication Scheme: Digest
         Username: "vmwcli01"
         Realm: "spo.voip.br"
         Nonce Value: "48ebc076448901fb6e912dc04ff04d73d91d94b8"
         Digest Authentication Response: "83085a000aa23c2792d4c57570d40108"
         Authentication URI: "sip:spo.voip.br"
         QOP: auth
         CNonce Value: "7D1F007277F0AE6E7075F1BA39C783F1"
         Nonce Count: 00000001
      Max-Forwards: 70
      User-Agent: X-Lite release 1105d
      Content-Length: 0
```

Figura B.7: Log do register entre UAmovel e os proxies rio.voip.br e spo.voip.br

A Figura B7 mostra a resposta do UA, apresentando suas credenciais ao *proxy* rio.voip.br para que seja realizada a validação.

```
Frame 112 (837 bytes on wire, 837 bytes captured)
...
Ethernet II, Src: Vmware_ce:24:8e (00:0c:29:ce:24:8e), Dst: Giga-Byt_9b:5c:2e (00:0f:ea:9b:5c:2e)
...
Internet Protocol, Src: 172.31.12.144 (172.31.12.144), Dst: 172.31.12.147 (172.31.12.147)
...
   Source: 172.31.12.144 (172.31.12.144)
   Destination: 172.31.12.147 (172.31.12.147)
User Datagram Protocol, Src Port: sip (5060), Dst Port: sip (5060)
...
Session Initiation Protocol
   Request-Line: REGISTER sip:spo.voip.br SIP/2.0
      Method: REGISTER
      [Resent Packet: False]
   Message Header
      Record-Route: <sip:172.31.12.144;lr=on;ftag=1553660366>
      Via: SIP/2.0/UDP 172.31.12.144;branch=z9hG4bKc886.7438e966.0
         Transport: UDP
         Sent-by Address: 172.31.12.144
         Branch: z9hG4bKc886.7438e966.0
      Via: SIP/2.0/UDP 172.31.12.142:5060;rport=5060;branch=z9hG4bK3ADAF0213E3D6BD8F547144C3F537EF3
         Transport: UDP
         Sent-by Address: 172.31.12.142
         Sent-by port: 5060
         RPort: 5060
         Branch: z9hG4bK3ADAF0213E3D6BD8F547144C3F537EF3
      From: 2115 <sip:vmwcli01@spo.voip.br>;tag=1553660366
         SIP Display info: 2115
         SIP from address: sip:vmwcli01@spo.voip.br
         SIP tag: 1553660366
      To: 2115 <sip:vmwcli01@spo.voip.br>
         SIP Display info: 2115
         SIP to address: sip:vmwcli01@spo.voip.br
      Contact: "2115" <sip:vmwcli01@172.31.12.142:5060>
         Contact Binding: "2115" <sip:vmwcli01@172.31.12.142:5060>
            URI: "2115" <sip:vmwcli01@172.31.12.142:5060>
               SIP Display info: "2115"
               SIP contact address: sip:vmwcli01@172.31.12.142:5060
      Call-ID: 1C5DF1047A8B10015F4A5B9346CF6184@spo.voip.br
      CSeq: 8055 REGISTER
         Sequence Number: 8055
         Method: REGISTER
      Expires: 1800
      Authorization: Digest username="vmwcli01",realm="spo.voip.br",nonce="48ebc076448901fb6e912dc04ff04d73d91d94b8",
response="83085a000aa23c2792d4c57570d40108",uri="sip:spo.voip.br",qop=auth,
cnonce="7D1F007277F0AE6E7075F1BA39C783F1",nc=00000001
         Authentication Scheme: Digest
         Username: "vmwcli01"
         Realm: "spo.voip.br"
         Nonce Value: "48ebc076448901fb6e912dc04ff04d73d91d94b8"
         Digest Authentication Response: "83085a000aa23c2792d4c57570d40108"
         Authentication URI: "sip:spo.voip.br"
         QOP: auth
         CNonce Value: "7D1F007277F0AE6E7075F1BA39C783F1"
         Nonce Count: 00000001
      Max-Forwards: 69
      User-Agent: X-Lite release 1105d
      Content-Length: 0
```

Figura B.8: Log do register entre UAmovel e os proxies rio.voip.br e spo.voip.br

A Figura B.8 mostra o proxy rio.voip.br encaminhando a requisição ao *proxy* spo.voip.br com o campo www-autheticate preenchido pelo UA em mobilidade.

Log da transferência do crédito de um usuário em mobilidade do domínio de origem para o domínio remoto, utilizando a própria sinalização SIP, que é uma novidade para transferência de valores, uma contribuição implementada, visto que, através de metodologias de bilhetagem, conhecida como CDR(*call detail record*) poderá ser computado tais valores, a fim de permitir o usuário em mobilidade ser tarifado.

```
Frame 113 (630 bytes on wire, 630 bytes captured)
...
Ethernet II, Src: Giga-Byt_9b:5c:2e (00:0f:ea:9b:5c:2e), Dst: Vmware_ce:24:8e (00:0c:29:ce:24:8e)
...
Internet Protocol, Src: 172.31.12.147 (172.31.12.147), Dst: 172.31.12.144 (172.31.12.144)
...
    Source: 172.31.12.147 (172.31.12.147)
    Destination: 172.31.12.144 (172.31.12.144)
User Datagram Protocol, Src Port: sip (5060), Dst Port: sip (5060)
...
Session Initiation Protocol
    Status-Line: SIP/2.0 200 OK
        Status-Code: 200
        [Resent Packet: False]
    Message Header
        Record-Route: <sip:172.31.12.144;lr=on;ftag=1553660366>
        Via: SIP/2.0/UDP 172.31.12.144;branch=z9hG4bKc886.7438e966.0
            Transport: UDP
            Sent-by Address: 172.31.12.144
            Branch: z9hG4bKc886.7438e966.0
        Via: SIP/2.0/UDP 172.31.12.142:5060;rport=5060;branch=z9hG4bK3ADAF0213E3D6BD8F547144C3F537EF3
            Transport: UDP
            Sent-by Address: 172.31.12.142
            Sent-by port: 5060
            RPort: 5060
            Branch: z9hG4bK3ADAF0213E3D6BD8F547144C3F537EF3
        From: 2115 <sip:vmwcli01@spo.voip.br>;tag=1553660366
            SIP Display info: 2115
            SIP from address: sip:vmwcli01@spo.voip.br
            SIP tag: 1553660366
        To: 2115 <sip:vmwcli01@spo.voip.br>;tag=06ce33ac5d943c89b7ef6922ffe1dd1f.74bd
            SIP Display info: 2115
            SIP to address: sip:vmwcli01@spo.voip.br
            SIP tag: 06ce33ac5d943c89b7ef6922ffe1dd1f.74bd
        Call-ID: 1C5DF1047A8B10015F4A5B9346CF6184@spo.voip.br
        CSeq: 8055 REGISTER
            Sequence Number: 8055
            Method: REGISTER
        Credito Disponivel:30
        Contact: <sip:vmwcli01@172.31.12.142:5060>;q=0;expires=3600
            Contact Binding: <sip:vmwcli01@172.31.12.142:5060>;q=0;expires=3600
                URI: <sip:vmwcli01@172.31.12.142:5060>
                    SIP contact address: sip:vmwcli01@172.31.12.142:5060
        Server: OpenSER (1.3.0-notls (i386/linux))
        Content-Length: 0
```

Recuperado no banco de origem do UA

Figura B.9: Log do register entre UAmovel e os proxies rio.voip.br e spo.voip.br

```
Frame 114 (568 bytes on wire, 568 bytes captured)
...
Ethernet II, Src: Vmware_ce:24:8e (00:0c:29:ce:24:8e), Dst: Vmware_e4:94:5d (00:0c:29:e4:94:5d)
...
Internet Protocol, Src: 172.31.12.144 (172.31.12.144), Dst: 172.31.12.142 (172.31.12.142)
...
   Source: 172.31.12.144 (172.31.12.144)
   Destination: 172.31.12.142 (172.31.12.142)
User Datagram Protocol, Src Port: sip (5060), Dst Port: sip (5060)
...
Session Initiation Protocol
   Status-Line: SIP/2.0 200 OK
      Status-Code: 200
      [Resent Packet: False]
   Message Header
      Record-Route: <sip:172.31.12.144;lr=on;ftag=1553660366>
      Via: SIP/2.0/UDP 172.31.12.142:5060;rport=5060;branch=z9hG4bK3ADAF0213E3D6BD8F547144C3F53
         Transport: UDP
         Sent-by Address: 172.31.12.142
         Sent-by port: 5060
         RPort: 5060
         Branch: z9hG4bK3ADAF0213E3D6BD8F547144C3F537EF3
      From: 2115 <sip:vmwcli01@spo.voip.br>;tag=1553660366
         SIP Display info: 2115
         SIP from address: sip:vmwcli01@spo.voip.br
         SIP tag: 1553660366
      To: 2115 <sip:vmwcli01@spo.voip.br>;tag=06ce33ac5d943c89b7ef6922ffe1dd1f.74bd
         SIP Display info: 2115
         SIP to address: sip:vmwcli01@spo.voip.br
         SIP tag: 06ce33ac5d943c89b7ef6922ffe1dd1f.74bd
      Call-ID: 1C5DF1047A8B10015F4A5B9346CF6184@spo.voip.br
      CSeq: 8055 REGISTER
         Sequence Number: 8055
         Method: REGISTER
      Credito Disponivel:30
      Contact: <sip:vmwcli01@172.31.12.142:5060>;q=0;expires=3600
         Contact Binding: <sip:vmwcli01@172.31.12.142:5060>;q=0;expires=3600
            URI: <sip:vmwcli01@172.31.12.142:5060>
               SIP contact address: sip:vmwcli01@172.31.12.142:5060
      Server: OpenSER (1.3.0-notls (i386/linux))
      Content-Length: 0
```

Figura B.10: Log do register entre UAmovel e os proxies rio.voip e spo.voip

A Figura B.10 mostra a confirmação do registro do UA em mobilidade no proxy rio.voip.br, com a resposta 200, ok.

Apêndice C – Procedimento de validação do processo de encaminhamento

Este apêndice tem o objetivo de demonstrar o procedimento criado no processo de encaminhamento para a requisição *INVITE* com uso da função *allow_trusted()* (nativa do serviço) que faz uso das regras definidas no arquivo *permissions.allow*.

Listada abaixo parte do script de configuração do *proxy* SIP que trata da validação da URI do usuário que está em mobilidade na realização do encaminhamento das chamadas:

```
loadmodule "permissions.so"
modparam("permissions","db_url","mysql://openser:openserrw@172.31.12.144:3306/openser")
#######
## Valido somente com a fc allow_trusted() ####
#######
modparam("permissions","db_mode",1)
modparam("permissions","source_col","src_ip")
modparam("permissions","trusted_table","trusted")
...
...
route{
...
...
        if (is_method("REGISTER")) {
                xlog("L_ALERT", "Chegou um REGISTER ($fU)($fu)($ru)($rU)($retcode)($td)\n");
                record_route();
                route(2);
                return;
        } else if (is_method("INVITE")) {
                xlog("L_ALERT", "Chegou um INVITE ($ru)\n");
                acc_db_request("INVITE", "acc");
                route(5);
                return;
...
...
        ###
        # Tratamento das mensagens de INVITE
        ###
        route[5] {
                xlog("L_ALERT", "Route[5]:INVITE: $fu -> $tu $ci\r\n");
                # Verificacao do arquivo de regras: permissions.allow
                if (!allow_trusted()){
                        log(1, "INVITE Nao estah na tabela trusted\n");
                        if (!proxy_authorize("","subscriber")) { #Dominio usado
                                proxy_challenge("","0"); # Realm serah autogerado
                                return;
                        } else if (!check_from()) {
                                sl_send_reply("403","User from=ID");
                                xlog("L_ERR", "ROUTE[5]: Error 403 $fu -> $tu\r\n");
                                return;
                        };
                } else {
                   log(1,"INVITE na tabela trusted\n");
                   xlog("L_ALERT","INVITE com From externo ($fu)\n");
                };
                if (!t_relay()) {
                   sl_send_reply("404","Nao encontrado");
                   xlog("L_ALERT","Error on t_relay()");
                } else {
                   xlog("L_ALERT","Call to t_relay() complete ok");
                }
                consume_credentials();
                return;
        }
...
...
```

Figura C.1: Procedimento de validação do processo de encaminhamento

Apêndice D – Estrutura das tabelas utilizadas no modelo de dados do Proxy

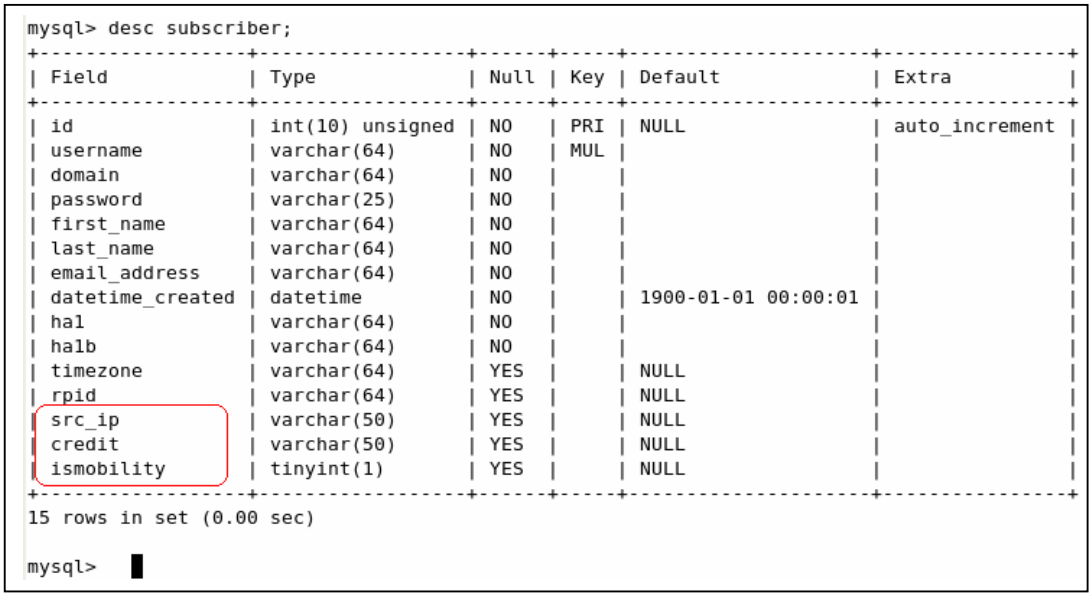
Visualização das estruturas das tabelas criadas/modificadas descritas no cap. 4 seção 4.2 como suporte ao modelo de registro e encaminhamento de chamadas.

```
mysql> desc trusdomain;
+---------------+------------------+------+-----+---------------------+----------------+
| Field         | Type             | Null | Key | Default             | Extra          |
+---------------+------------------+------+-----+---------------------+----------------+
| id            | int(10) unsigned | NO   | PRI | NULL                | auto_increment |
| trusdomain    | varchar(64)      | NO   | UNI |                     |                |
| last_modified | datetime         | NO   |     | 1900-01-01 00:00:01 |                |
+---------------+------------------+------+-----+---------------------+----------------+
3 rows in set (0.03 sec)

mysql>
```

Figura D.1: Estrutura da tabela trusdomain

```
mysql> desc location;
+---------------+------------------+------+-----+---------------------+----------------+
| Field         | Type             | Null | Key | Default             | Extra          |
+---------------+------------------+------+-----+---------------------+----------------+
| id            | int(10) unsigned | NO   | PRI | NULL                | auto_increment |
| username      | varchar(64)      | NO   | MUL |                     |                |
| domain        | varchar(64)      | YES  |     | NULL                |                |
| contact       | varchar(255)     | NO   |     |                     |                |
| received      | varchar(128)     | YES  |     | NULL                |                |
| path          | varchar(128)     | YES  |     | NULL                |                |
| expires       | datetime         | NO   |     | 2020-05-28 21:32:15 |                |
| q             | float(10,2)      | NO   |     | 1.00                |                |
| callid        | varchar(255)     | NO   |     | Default-Call-ID     |                |
| cseq          | int(11)          | NO   |     | 13                  |                |
| last_modified | datetime         | NO   |     | 1900-01-01 00:00:01 |                |
| flags         | int(11)          | NO   |     | 0                   |                |
| cflags        | int(11)          | NO   |     | 0                   |                |
| user_agent    | varchar(255)     | NO   |     |                     |                |
| socket        | varchar(64)      | YES  |     | NULL                |                |
| methods       | int(11)          | YES  |     | NULL                |                |
+---------------+------------------+------+-----+---------------------+----------------+
16 rows in set (0.01 sec)

mysql>
```

Figura D2: Estrutura da tabela location



```
mysql> desc subscriber;
+------------------+------------------+------+-----+---------------------+----------------+
| Field            | Type             | Null | Key | Default             | Extra          |
+------------------+------------------+------+-----+---------------------+----------------+
| id               | int(10) unsigned | NO   | PRI | NULL                | auto_increment |
| username         | varchar(64)      | NO   | MUL |                     |                |
| domain           | varchar(64)      | NO   |     |                     |                |
| password         | varchar(25)      | NO   |     |                     |                |
| first_name       | varchar(64)      | NO   |     |                     |                |
| last_name        | varchar(64)      | NO   |     |                     |                |
| email_address    | varchar(64)      | NO   |     |                     |                |
| datetime_created | datetime         | NO   |     | 1900-01-01 00:00:01 |                |
| ha1              | varchar(64)      | NO   |     |                     |                |
| ha1b             | varchar(64)      | NO   |     |                     |                |
| timezone         | varchar(64)      | YES  |     | NULL                |                |
| rpid             | varchar(64)      | YES  |     | NULL                |                |
| src_ip           | varchar(50)      | YES  |     | NULL                |                |
| credit           | varchar(50)      | YES  |     | NULL                |                |
| ismobility       | tinyint(1)       | YES  |     | NULL                |                |
+------------------+------------------+------+-----+---------------------+----------------+
15 rows in set (0.00 sec)

mysql>
```

Figura D3: Estrutura da tabela subscriber

A Figura D3 mostra as colunas *src_ip* e *ismobility* criadas para dar suporte à validação do processo de mobilidade no *proxy registrar* e também à coluna credit para armazenamento do valores trazidos do *proxy* de origem do UA em mobilidade.

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

CCMN - Bloco C - Cidade Universitária - Ilha do Fundão
Rio de Janeiro - RJ CEP: 21941-916
www.ppgi.ufrj.br