UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

EMPREGO DE ANONIMATO PARA MELHORIA DE PRIVACIDADE NO CONSUMO DE SERVIÇOS EM SAAS

VINÍCIUS MAIA PACHECO

ORIENTADOR: RICARDO STACIARINI PUTTINI

TESE DE DOUTORADO EM ENGENHARIA ELÉTRICA

PUBLICAÇÃO: PPGEE.TD 073 – A/13

BRASÍLIA / DF: MAIO / 2013

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

EMPREGO DE ANONIMATO PARA MELHORIA DE PRIVACIDADE NO CONSUMO DE SERVIÇOS EM SAAS

VINÍCIUS MAIA PACHECO

TESE DE DOUTORADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR.

APROVADA POR:

RICARDO STACIARINI PUTTINI, Doutor, ENE/FT/UnB, Brasil
(ORIENTADOR)

ANDERSON CLAYTON ALVES NASCIMENTO, Doutor, ENE/UnB, Brasil
(EXAMINADOR INTERNO)

RAFAEL TIMÓTEO DE SOUSA, Doutor, ENE/UnB, Brasil
(EXAMINADOR INTERNO)

LUDOVIC MÉ, Doutor, Supélec-Rennes, França
(EXAMINADOR EXTERNO)

RICARDO MATOS CHAIM, Doutor, FGA/UnB, Brasil
(EXAMINADOR EXTERNO)

FLÁVIO ELIAS GOMES DE DEUS, Doutor, ENE/UnB, Brasil
(SUPLENTE)

DATA: BRASÍLIA/DF, 07 DE MAIO DE 2013.

FICHA CATALOGRÁFICA

PACHECO, VINÍCIUS MAIA
Emprego de Anonimato para Melhoria de Privacidade no Consumo de Serviços em SaaS [Distrito Federal] 2013.
xxii, 195p, 297 mm
(ENE/FT/UnB, Doutor, Engenharia Elétrica, 2013).

Tese de Doutorado – Universidade de Brasília, Faculdade de Tecnologia. Departamento de Engenharia Elétrica.

1. Anonimato 2. Computação em Nuvem
3. Privacidade 4. SaaS

I. ENE/FT/UnB. II. Título

REFERÊNCIA BIBLIOGRÁFICA

PACHECO, V. M. (2013). Emprego de Anonimato para Melhoria de Privacidade no Consumo de Serviços em SaaS. Tese de Doutorado, Publicação PPGEE.TD – 073 A/13, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 195p.

CESSÃO DE DIREITOS

NOME DO AUTOR: Vinícius Maia Pacheco
TÍTULO DA TESE: Emprego de Anonimato para Melhoria de Privacidade no Consumo de Serviços em SaaS.
GRAU/ANO: Doutor/2013.

Vinícius Maia Pacheco
SHIS QI 3 Conjunto 8 casa 2
CEP 70680-250 – Brasília – DF - Brasil

à minha família, especialmente à Maria Carla.

AGRADECIMENTOS

À minha família por ter suportado emocional e financeiramente a minha educação.

Em especial, à Maria Carla por ter inspirado e motivado a decisão de aceitar este desafio. Meu amor, minha vida.

Ao meu orientador, o Prof. Dr. Ricardo Staciarini Puttini, pelo constante apoio, incentivo, dedicação e amizade essenciais para o desenvolvimento deste trabalho e para o meu aperfeiçoamento como pesquisador. Além de orientador, meu grande amigo.

Ao SICS – *Swedish Institute of Computer Science* – na figura de Per Brand, por ter me recebido para o período de pesquisa na Suécia, fundamental para a concretização deste trabalho. Também, aos amigos da Suécia, especialmente ao Gustavo Azzolin, pela acolhida, amizade e discussões enriquecedoras.

Ao IBTI – *Instituto Brasília de Tecnologia e Inovação* – pela contribuição durante o período de validação experimental da tese.

A todos, os meus sinceros agradecimentos.

RESUMO

EMPREGO DE ANONIMATO PARA MELHORIA DE PRIVACIDADE NO CONSUMO DE SERVIÇOS EM SAAS

Autor: Vinícius Maia Pacheco

Orientador: Ricardo Staciarini Puttini

Programa de Pós-graduação em Engenharia Elétrica

Brasília, Maio de 2013

O objetivo desta tese é prover privacidade ao consumidor no modelo Software-as-a-Service – SaaS – da computação em nuvem. Dentro deste paradigma, as características do ambiente em nuvem, do próprio SaaS e, consequentemente, das entidades envolvidas limitam uma abordagem direta de segurança das informações sensíveis do consumidor.

Nesse sentido, iniciou-se por promover uma análise compreensiva da situação de privacidade do consumidor dentro do SaaS. O enfoque foi modular, de forma que as interações consistindo o consumo de serviço foram divididas em níveis, cada um abarcando itens de interesse específicos do consumidor. Essa primeira contribuição original da tese permitiu definir um método apropriado para se atingir a privacidade buscada. Optou-se por uma abordagem modular e baseada no anonimato, pois, no SaaS, cada nível de interação é melhor tratado por técnicas específicas de anonimato e o acesso irrestrito às informações do consumidor por parte do provedor é fundamental para o processamento do serviço.

Surgiu, assim, a segunda contribuição: a estrutura multicamadas de anonimato para SaaS. Com o enfoque modular de proteção dos itens de interesse, apresentam-se tecnologias de anonimato específicas para as características de cada camada, que atuando em conjunto proporcionam ao consumidor uma solução de privacidade completa e flexível. Pode-se selecionar as técnicas de uma camada de acordo com as demandas de privacidade específicas.

A terceira contribuição original da tese é constituída pelos esquemas práticos. Os esquemas práticos são instâncias da estrutura, apresentando um design de transações que suporta e concilia as tecnologias escolhidas para cada camada de anonimato. De destacar, dois esquemas são apresentados, um com anonimato rastreável e outro sem rastreabilidade.

Promove-se, também, uma validação experimental da tese com a implementação de um esquema prático. Verifica-se, na prática, como a análise de privacidade do SaaS, a estrutura multicamadas e o conceito de esquema de prático possibilitam a privacidade do consumidor.

ABSTRACT

EMPLOYING ANONYMITY TO ENHANCE PRIVACY IN SAAS SERVICE CONSUMPTION

Author: Vinícius Maia Pacheco

Supervisor: Ricardo Staciarini Puttini

Programa de Pós-graduação em Engenharia Elétrica

Brasília, May of 2013

The objective of this thesis is to provide privacy to the consumer of the Software-as-a-Service – SaaS – model in cloud computing. Within this paradigm, the characteristics of the cloud environment, of the actual SaaS, and of the entities involved limit a direct approach to the safety of consumers' sensitive information.

In this sense, a comprehensive analysis of the situation of consumer privacy within the SaaS was promoted. The approach was modular, so that the interactions of SaaS service consumption were divided into levels, each covering specific items of interest. This first original contribution of the thesis allowed the definition of an appropriate method to achieve the privacy sought. A modular approach based on anonymity was elected, because, in SaaS, each level of interaction is best handled by specific anonymity techniques and unrestricted access to consumer's information by the provider is critical to the processing of the service.

Thus emerged the second original contribution: a multilayered structure for SaaS anonymity. Employing a modular approach for the protection of the items of interest, specific anonymity technologies are presented, tailored to the characteristics of each layer. And, these act together to provide the consumer with a complete and flexible privacy solution. Anonymity techniques can be selected accordingly to specific privacy demands.

The third contribution consists of the proposal of practical schemes. The practical schemes are instances of the structure, with a transaction design that supports and reconciles the technologies chosen for each layer of anonymity. Of note, two schemes are presented, one with and the other without traceable anonymity.

An experimental validation of the thesis is also promoted, with the implementation of a practical scheme. With it, it is verified how the SaaS privacy analysis, the multilayered structure and the concept of a practical scheme enable consumer privacy in SaaS.

ÍNDICE

LISTA DE TABELAS

LISTA DE FIGURAS

1. INTRODUÇÃO

A computação em nuvem é uma das mais importantes mudanças de paradigma da indústria de TI na última década. Ela emergiu como um novo modelo de negócios, onde recursos remotos de TI são compartilhados em larga escala, permitindo uma alocação dinâmica e flexível, e possibilitando a consequente redução de custos. Esses recursos de TI são disponibilizados como serviços acessíveis ubiquamente, tipicamente pela Internet, e por meio de plataformas heterogêneas, variando desde celulares até estações de trabalho ou servidores.

O *Software-as-a-Service* – SaaS – é um modelo de entrega na computação em nuvem, onde um provedor de nuvem disponibiliza uma determinada aplicação como um serviço em *software*, em concordância com as características do paradigma da computação em nuvem.

Serviços SaaS se tornaram bastante comuns, oferecendo em nuvem aplicações do tipo faturamento, colaboração na criação e edição de documentos, CRM (*Customer Relationship Management*), ERP (*Enterprise Resource Planning*), e diversas outras. Uma das principais vantagens proporcionadas pelo modelo SaaS é o potencial que um consumidor de nuvem tem de usufruir imediatamente e sob demanda de um serviço em *software* de alto nível, sem os gastos de investimento envolvidos na implementação e na manutenção de um ambiente ou plataforma em *hardware* e *software*. Em estimativa publicada pelo Gartner [75], as vendas mundiais dentro do modelo SaaS em 2010 atingiram U$ 10 bilhões, com projeção para alcançar U$ 21.3 bilhões em 2015.

O consumo de serviços SaaS é feito através da troca de mensagens entre consumidor e provedor. Essas mensagens contêm os dados do consumidor e normalmente carregam informações estratégicas sobre o negócio da empresa ou indivíduo. Embora terceiros possam interceptar ou espiar essas trocas de mensagem, a criptografia tradicional pode abordar tais situações com sucesso [78], protegendo a confidencialidade dos consumidores em relação a possíveis atacantes externos. Porém, controlar a exposição de informação do consumidor em relação ao provedor dentro do paradigma SaaS é um desafio maior. O provedor precisa ter acesso aos dados para efetivamente realizar o

processamento do serviço. Um provedor que, por exemplo, disponibiliza um serviço de cálculo de folha de pagamento em modelo SaaS na nuvem precisa necessariamente ter acesso à lista dos funcionários da empresa consumidora e às demais informações necessárias para o processamento dos salários de cada um.

Ainda que existam dentro da área da criptografia os chamados criptossistemas homomórficos [108], focados em realizar operações em dados cifrados, sua aplicação é limitada na proteção de privacidade do consumidor SaaS. Excluindo-se as questões de performance ainda existentes nos criptossistemas homomórficos [69], mesmo que um provedor possa eventualmente processar os dados do consumidor sem precisar os decifrar, outros aspectos privados do consumidor serão impactados, como, por exemplo, a própria identificação do consumidor para provedor no momento do consumo, no qual apresenta-se algum tipo de credencial e demanda-se do provedor um cálculo específico em seus dados cifrados.

Nenhuma outra entidade dentro do modelo SaaS terá maior acesso às informações do consumidor do que o próprio provedor. Configura-se um cenário onde o provedor passa a ser considerado como a entidade com maior poder de ameaça e como a possível origem de mau uso das informações do consumidor. Torna-se o provedor, portanto, o foco apropriado para uma abordagem de proteção de privacidade no contexto SaaS.

Resta estabelecida a motivação da tese: possibilitar o consumo de serviços SaaS, enquanto mantendo resguardada a privacidade do consumidor em relação ao provedor. Ademais, pondera-se que ao proteger a privacidade do consumidor frente ao provedor, outras entidades do paradigma da computação em nuvem com menor poder de acesso, como espiões externos ou mesmo consumidores maliciosos, também terão seus impactos mitigados.

Buscou-se estabelecer um método objetivo para se efetivar a proteção de privacidade do consumidor SaaS. A abordagem selecionada foi primeiramente instituir uma avaliação do estado da privacidade do ambiente, considerando as particularidades da computação em nuvem e do paradigma SaaS.

Para isso, dividiu-se conceitualmente em níveis as interações entre consumidor e provedor, com cada nível possuindo itens de interesse particulares que, por sua vez, afetam a privacidade do consumidor de uma determinada maneira. Esse modo de influência pode ser classificado de acordo com os aspectos privados específicos sendo impactados. Instituem-se, assim, as dimensões da privacidade; e essas irão nortear como será a abordagem de proteção de cada item de interesse.

Os três níveis conceituais de interações entre consumidores e provedores identificados são: **contrato, troca de mensagens** e **comunicações de rede**. O primeiro nível é uma interação fora da banda e se refere ao estabelecimento de um contrato legal, especificando os termos de utilização do serviço, como condições de sigilo, SLA (*Service Level Agreement*) e tarifação. A identidade e requisitos de negócio são usualmente revelados neste momento. O segundo nível consiste na troca de mensagens durante o consumo do serviço propriamente dito. Nesta etapa, credenciais de autenticação, padrões de uso de serviço, e, no limite, os próprios dados do consumidor são expostos ao provedor. Já no o terceiro nível, o das comunicações de rede, endereços IP podem facilmente ser ligados à identidade do consumidor e à sua localização.

Os itens de interesse, neste contexto, são os dados específicos do consumidor que carregam informações sensíveis relativas à privacidade dele em cada nível de interação considerado. Seja a própria credencial do consumidor, ou mesmo a frequência com que o consumidor acessa o serviço, é de interesse do consumidor proteger a manipulação ou mesmo o conhecimento dessas informações.

Em relação à classificação objetiva de privacidade, estabelecem-se quatro dimensões para o conceito: **ID**, **localização**, **comportamento** e **conteúdo**. Um item de interesse pode, desse modo, ser classificado em relação ao impacto específico que traz à privacidade do consumidor. Por exemplo, para o caso do nível de contrato, onde existem os itens de interesse presentes no próprio contrato, como a identificação do consumidor e o SLA definido, verifica-se que essa identificação impacta a dimensão de ID e o SLA afeta a dimensão do comportamento. A identificação estabelece um elo direto para o consumidor (ID) e o SLA permite uma estimativa de como será a

característica de consumo do consumidor, revelando, por conseguinte, como será seu comportamento.

Estendendo-se a análise para todos os níveis e seus respectivos itens de interesse, obtém-se um mapeamento modular da privacidade do consumidor SaaS, relacionando quais itens de interesse pertencem a qual nível de interação e quais dimensões da privacidade são impactadas por eles.

Essa **análise de privacidade do consumo de serviços SaaS**, fundada na inter-relação entre os níveis de interação do consumo SaaS, os itens de interesse e as dimensões de privacidade afetadas, constitui a primeira contribuição original da tese. Abordar modularmente os aspectos de privacidade de todos os dados de um consumidor SaaS é um enfoque abrangente e não se restringe a apenas um aspecto de privacidade, item de interesse específico ou nível de interação particular, que é uma proposta comum em soluções de privacidade estabelecidas, como, por exemplo, a criptografia tradicional.

Nesse sentido, ao permitir uma visão compreensiva de como a privacidade do consumidor é afetada durante um consumo SaaS, a análise de privacidade possibilitou a seleção da tecnologia adequada para a proteção do consumidor.

Basicamente, como todos os itens de interesse do consumidor precisam ser necessariamente expostos ao provedor durante a operação normal do SaaS, protegê-los simplesmente com criptografia não é eficaz. Desponta, destarte, o anonimato como a tecnologia mais adequada para atingir o objetivo da tese. O anonimato permite a manutenção do acesso à determinada informação, mas garante a manutenção de privacidade durante esse contato.

Conceitualmente, o anonimato é a propriedade que uma entidade assume em determinada ocasião, onde, no evento avaliado, não é unicamente identificável dentro do conjunto de entidades a serem consideradas [103]. A Figura 1.1 resume como a motivação da tese gera a primeira contribuição original, que, por conseguinte, define o anonimato como a tecnologia mais adequada para a introdução da privacidade no consumo de serviços SaaS em nuvem.

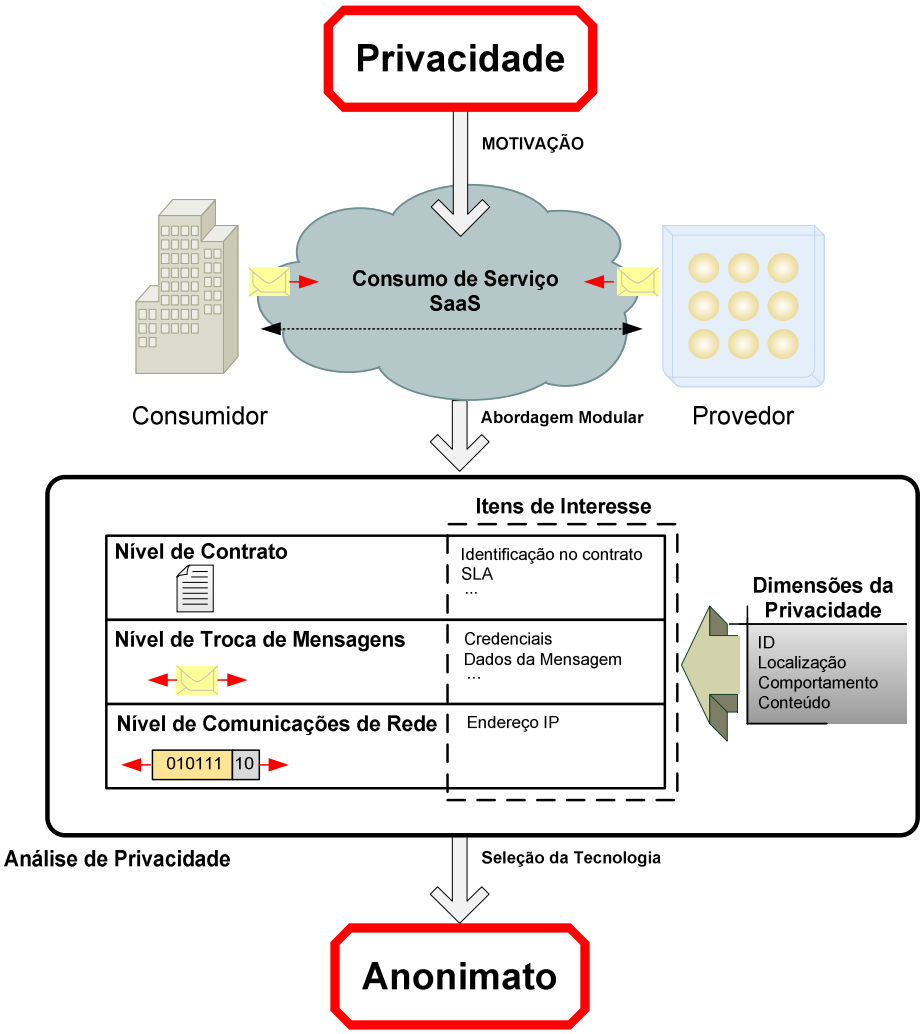

Figura 1.1 – Definição do Anonimato como Tecnologia para a Proteção da Privacidade

Com a análise de privacidade definida, é necessário agora determinar como objetivamente o anonimato poderá imprimir a privacidade no consumo de serviços SaaS. Sabe-se que o anonimato tem sido utilizado com sucesso na proteção da privacidade em geral [65], mas, até o momento, no melhor conhecimento do autor, ainda não foi especificado como ferramenta de melhoria da privacidade no ambiente de computação em nuvem, salvo por pesquisas preliminares em [67].

Assim sendo, na definição de como empregar o anonimato em SaaS, o autor estabeleceu a segunda contribuição original da tese: a **estrutura conceitual multicamadas de**

anonimato para SaaS. Nessa estrutura, inspirada no inter-relacionamento entre os níveis de interação consumidor-provedor, os itens de interesse respectivos e as dimensões de privacidade, instituem-se camadas de anonimato, onde são propostas diversas e complementares tecnologias de anonimato para resguardar a privacidade do consumidor em relação ao provedor. A estrutura é flexível, sendo possível substituir uma tecnologia específica em determinada camada por outra que seja mais adequada para o nível e dimensões de privacidade requeridas pelo consumidor. A proteção oferecida por cada tecnologia e cada camada atua independentemente das outras e, no final, contribui para a melhoria como um todo da privacidade total.

A definição das tecnologias de anonimato específicas para cada camada da estrutura foi permitida pela modularização das interações entre consumidor e provedor e pelo isolamento dos itens de interesse, juntamente com a divisão da privacidade em dimensões. Destaca-se que algumas tecnologias a serem empregadas pela estrutura multicamadas não são classificadas tradicionalmente como tecnologias de anonimato, como, por exemplo, a própria criptografia homomórfica, que será pertinente para anonimizar o item de interesse relativo aos dados da mensagem do consumidor. Porém, como aqui essas tecnologias estarão cumprindo o objetivo de anonimizar o consumidor, serão consideradas no contexto deste trabalho como tecnologias de anonimato.

Ainda neste paradigma, é também útil a consideração de dois tipos de anonimato: o de conexão e o de dados [46]. Anonimato de conexão foca na proteção das identidades do remetente e do destinatário durante as comunicações, enquanto anonimato de dados se refere à remoção da possibilidade da identificação de uma entidade a partir de seus dados.

As tecnologias que trabalham com o anonimato de conexão são diferentes daquelas que focam no anonimato de dados. Dessa maneira, pôde-se melhor selecionar as tecnologias apropriadas para cada nível de interação consumidor-provedor. Mais ainda, verificou-se que para melhor empregar o anonimato no contexto SaaS, seria mais adequado o estabelecimento da estrutura multicamadas com quatro camadas de anonimato.

As quatro camadas da estrutura são: **contrato**, **metadados**, **rede** e **conteúdo**. A camada de contrato corresponde diretamente ao nível de interação consumidor-provedor de contrato e tem como tecnologia de anonimato apropriada os chamados corretores de nuvem [8]. O tipo de anonimato aplicável aqui é o de conexão. Os corretores de nuvem não são diretamente uma tecnologia de anonimato, mas são muito úteis hoje na intermediação de vários aspectos do consumo de serviços em nuvem. Todavia, como eles também focam na intermediação de contratos entre consumidores e provedores, utilizar-se-á nesta estrutura um tipo específico de corretor que se especializará em não expor a privacidade do consumidor ao provedor no que se refere à informação sensível presente nos contratos.

A segunda camada, a de metadados, faz parte do nível de interação de troca de mensagens, e o tipo de anonimato utilizado também é o de conexão. As tecnologias de anonimato aplicáveis aqui são as de anonimato de credenciais. Especialmente nesta estrutura, propõem-se o emprego de assinaturas de grupo [38] ou de dinheiro eletrônico [34]. As assinaturas de grupo são credenciais anônimas que permitem a autenticação sem identificação, mas proporcionam o rastreamento por terceiros caso necessário. Já o dinheiro eletrônico é uma assinatura anônima que, assim como o dinheiro real, não é rastreável.

A terceira camada é a camada de rede e ela faz parte do nível de interação de comunicações de rede. O tipo de anonimato empregado também é o de conexão e as tecnologias adequadas para esse contexto são as chamadas redes Mix-Net [37]. Mix-nets são sistemas focados em possibilitar o anonimato dos endereços de rede utilizados em transações de rede, especialmente na Internet.

Finalmente, a quarta camada da estrutura é a de conteúdo. Ela corresponde à aplicação do tipo de anonimato de dados no nível de interação consumidor-provedor de troca de mensagens. As tecnologias de anonimato aqui pertinentes são a de anonimato-k [111] e a de criptossistemas homomórficos [108]. O anonimato-k é uma tecnologia que busca o anonimato de microdados por meio da supressão ou generalização das informações que possam unicamente identificar cada registro. Para o caso dos criptossistemas homomórficos, como comentado no previamente, o objetivo é alcançar a privacidade do

conteúdo em questão por meio da realização de operações matemáticas diretamente nos dados cifrados.

A partir, destarte, da concepção da estrutura multicamadas de anonimato fundou-se a terceira contribuição da tese: os **esquemas práticos para consumo anônimo de serviços SaaS**. Os esquemas práticos são instâncias da estrutura multicamadas, selecionando um conjunto de tecnologias de anonimato da estrutura (uma de cada camada) e oferendo um design de transações correspondente que suporte tais escolhas. Cada esquema prático é projetado para atender requisitos de anonimato específicos e dois esquemas são propostos e detalhados na tese. Contudo, a estrutura multicamadas pode ser utilizada como base para a definição de outros esquemas práticos, mais ajustados para eventuais requisitos diferentes de privacidade ou mesmo de performance.

O primeiro esquema proposto é o que oferece o anonimato rastreável, ou seja, o consumidor é anônimo durante o consumo do serviço, mas pode ser identificado *a posteriori*, isto é, após o consumo do serviço. Esse cenário é adequado para situações onde a possibilidade de rastreamento é um requisito formal, em caso, por exemplo, de mau uso ou de mau comportamento. Isso normalmente deriva de contextos legais, regulamentais ou mesmo políticos. A possibilidade de rastreamento deste esquema prático é fundamentada na escolha das assinaturas de grupo como tecnologia de anonimato da camada de metadados da estrutura.

O segundo esquema proposto foca no consumo não rastreável de serviços SaaS. Este esquema é recomendado para ambientes onde prevalece a condição de anonimato absoluto para o consumidor. A não rastreabilidade do consumidor é proporcionada neste esquema prático pela escolha do dinheiro eletrônico como a tecnologia de anonimato da camada de metadados da estrutura.

A Figura 1.2 mostra a concepção da estrutura multicamadas e dos esquemas práticos, baseada na análise de privacidade.

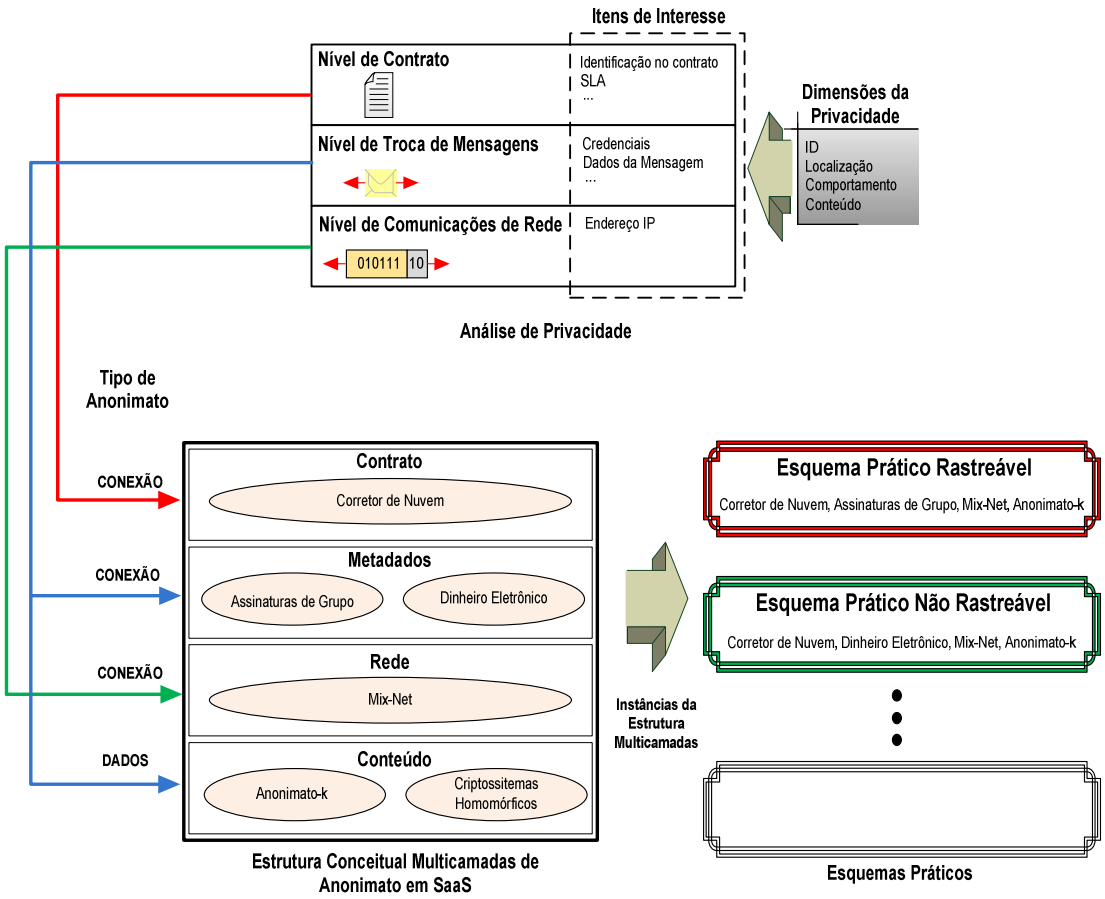

Figura 1.2 – Concepção da Estrutura Multicamadas e dos Esquemas Práticos

Por fim, com o objetivo de validar experimentalmente as contribuições originais da tese, elaborou-se uma implementação de um esquema prático, criando um produto em *software* capaz de concretizar um consumo de serviços SaaS anônimo, de forma que o consumidor possa usufruir de um serviço enquanto resguarda sua privacidade em relação ao provedor. Verificar-se-á, na prática, como a análise de privacidade do consumo de serviços SaaS, a estrutura multicamadas de anonimato e o conceito de esquema de prático permitem a proteção da privacidade do consumidor.

Considerando trabalhos relacionados, ao melhor conhecimento do autor, a análise acerca da privacidade de um consumidor interagindo com um provedor, dentro do paradigma do consumo de serviços SaaS, é inédita, bem como a concepção da estrutura multicamadas e dos esquemas práticos. Nesse contexto, e a título de comparação, o

artigo preliminar de Jensen, Schäge e Schwenk, "*Towards an anonymous access control and accountability scheme for cloud computing*" [67], não considera as interações consumidor-provedor no nível das comunicações de rede, não apresenta um esquema prático com design das transações, e oferece apenas uma proposta voltada para o anonimato rastreável. Outro artigo que se situa próximo aos aspectos abordados nesta tese é o denominado "*Providing privacy preserving in cloud computing*" [132], de Wang, Zhao, Jiang e Le. Todavia, nele, o foco é apenas o anonimato de dados em computação em nuvem, não abordando contratos, metadados e nem rede. Excluiu, portanto, o anonimato de conexão. Destaca-se, também, que em nenhum desses trabalhos correlatos foi apresentada uma validação experimental.

A tese se organiza da seguinte forma: no Capítulo 2, o modelo de consumo de serviço SaaS da computação em nuvem é descrito e a análise de privacidade é conduzida, aclarando a motivação do trabalho e fundamentando o uso do anonimato para a proteção da privacidade. Em seguida, no Capítulo 3, a segunda contribuição é apresentada e detalhada: a estrutura multicamadas de anonimato para consumo de serviços SaaS. Já nos Capítulos 4 e 5, a terceira contribuição é proposta na forma dos esquemas rastreável e não rastreável de consumo anônimo de serviços, respectivamente. No Capítulo 6, a validação experimental da tese é mostrada. E, finalmente, no Capítulo 7, as conclusões são oferecidas.

1.1. PUBLICAÇÕES

As publicações referentes a este trabalho foram focadas na evolução gradual da pesquisa, com artigos sendo produzidos na medida em que se consolidavam as contribuições. Destarte, foram elaborados, submetidos à avaliação e subsequentemente aceitos para publicação, os seguintes trabalhos, em ordem cronológica:

1.1.1. Artigo "*Untraceable Anonymous Service Consumption in SaaS*"

O primeiro artigo publicado na linha de pesquisa da tese foi o "*Untraceable Anonymous Service Consumption in SaaS*" [99].

Apresentado em abril de 2012, no Porto, em Portugal, exibiu-se nele, pela primeira vez, o conceito de uma estrutura (*framework*) multicamadas voltada para a proteção do anonimato do consumidor em relação ao provedor, considerando a privacidade no consumo de serviços SaaS da computação em nuvem. Foi, também, oferecida a descrição e detalhamento unicamente do esquema não rastreável para consumo anônimo de serviços SaaS, baseado em *E-cash*.

O evento em questão refere-se à conferência de 2012, centrada na computação em nuvem, chamado CLOSER – *International Conference on Cloud Computing and Service Science*.

1.1.2. Artigo *"SaaS Anonymous Cloud Service Consumption Structure"*

Em seguida, foi publicado o segundo artigo, denominado "*SaaS Anonymous Cloud Service Consumption Structure*" [101].

Com apresentação em junho de 2012, em Macau, na China, expandiu-se nele o conceito da estrutura multicamadas de anonimato do consumidor de serviços SaaS. Contudo, mostrou-se, ao invés do esquema não rastreável, a descrição e o detalhamento do esquema rastreável de consumo anônimo de serviços SaaS, baseado, desta vez, na tecnologia de assinaturas de grupo.

O evento em questão refere-se ao *workshop* de segurança e privacidade na computação em nuvem, da 32ª edição da conferência ICDCS – *International Conference on Distributed Computing Systems*.

1.1.3. Artigo *"Defining and Implementing Connection Anonymity for SaaS Web Services"*

Como terceiro artigo referente ao trabalho da tese, foi publicado o "*Defining and Implementing Connection Anonymity for SaaS Web Services*" [100].

Exposto em junho de 2012, em Honolulu, Havaí, consolidou-se nele a estrutura multicamadas, instituindo-se uma solução conceitual abrangente para a privacidade do consumidor de serviços SaaS. Foram ajustados aspectos teóricos com fundamentação nas revisões e apresentações dos artigos anteriores, além da expansão da aplicabilidade

da estrutura. Em relação aos esquemas práticos, as duas versões foram apresentadas, a do anonimato rastreável, com assinaturas de grupo, e a do anonimato não rastreável, com *E-cash*, ambas também com ajustes oriundos da evolução da pesquisa. Finalmente, com introdução também inédita, delineou-se ainda os *web services* referentes às implementações dos esquemas práticos.

A conferência em questão refere-se à 5ª conferência internacional de computação em nuvem do IEEE – CLOUD 2012. Considerada, nesse sentido, como o evento de maior relevância para a computação em nuvem, no contexto acadêmico e da Engenharia, a CLOUD 2012 estabelece a importância e a consolidação internacional da contribuição atestada por esta tese.

1.1.4. Livro "*Cloud Computing: Concepts, Technology & Architecture*"

De destacar, como fruto da pesquisa desenvolvida para a tese, o autor da tese foi convidado para atuar como colaborador do livro centrado na computação em nuvem: "*Cloud Computing: Concepts, Technology & Architecture*" - ISBN-13: 978-0133387520 [51].

O livro em questão ultrapassa o reconhecimento acadêmico e é voltado também para profissionais da indústria da computação em nuvem. A contribuição no livro por parte do autor da tese focou-se justamente nos aspectos de privacidade e segurança do consumo de serviços em nuvem, elevando sobremaneira a relevância e validação do trabalho realizado.

2. MODELO DE CONSUMO DE SERVIÇO EM SAAS E ANÁLISE DE PRIVACIDADE

O foco desta tese é a proteção de privacidade do consumidor durante o consumo de serviços no modelo de entrega SaaS da computação em nuvem. Para se atingir tal proteção, é necessário considerar as particularidades do paradigma da computação em nuvem e do próprio modelo SaaS.

Nesse sentido, este Capítulo mostra inicialmente uma visão geral da computação em nuvem, permitindo o entendimento das características únicas desta tecnologia. Em seguida, o modelo de consumo de serviços SaaS é apresentado.

Com base na busca da privacidade do consumidor e considerando as características específicas da computação em nuvem e do modelo SaaS, proceder-se-á, à análise da privacidade do consumidor dentro deste contexto, e ela constituirá a primeira contribuição original da tese. Essa análise de privacidade leva em conta uma divisão do consumo de serviços SaaS em níveis de interação entre consumidor e provedor, os itens de interesse referentes a cada nível e a classificação da privacidade em dimensões.

A análise privacidade deste Capítulo servirá como base conceitual para a concepção da estrutura multicamadas de anonimato para SaaS, no Capítulo 3, e para a criação dos esquemas práticos, nos Capítulos 4 e 5, que, por sua vez, instanciarão a estrutura multicamadas para oferecer um design de transações que suportará as tecnologias de anonimato propostas pela estrutura.

2.1. COMPUTAÇÃO EM NUVEM

A computação em nuvem é uma tecnologia emergente que vem remodelando significativamente a área de Tecnologia da Informação na última década [88]. Fundamentada no compartilhamento de recursos em larga escala, a computação em nuvem tem como principais objetivos a redução de custos e a agilidade organizacional. Com a computação em nuvem, empresas podem confiar parte de sua infraestrutura de TI a um provedor especializado, e, consequentemente, focar sua estratégia de operação

em seu real negócio. A empresa provedora capitaliza uma grande gama de recursos de TI, compartilhados entre seus clientes, de mão-de-obra especializada e de tecnologias para consolidação de *datacenter* para estabelecer uma arquitetura de fácil particionamento e acesso, reduzindo e padronizando custos operacionais e administrativos. Essa estrutura é, então, oferecida como um serviço aos consumidores, constituindo o modelo de negócios da computação em nuvem. Já a empresa consumidora se torna mais ágil e focada, podendo pagar menos por esse serviço especializado e tendo a possibilidade de alocar recursos dinamicamente e ubiquamente, sem implicar na aquisição de equipamentos e em sua manutenção logística.

A relação entre provedor e consumidor introduz, no entanto, novos e importantes aspectos a serem considerados e abordados. Os consumidores devem reconhecer que terão suas fronteiras de operação e segurança estendidas para a nuvem, e, portanto, deverão adotar medidas para controlar e monitorar a manipulação de seus dados. Mais ainda, como agora parte de sua infraestrutura logística de TI não estará mais em suas dependências, contratos regulando as responsabilidades legais, níveis de serviço, modelo de cobrança e governança crescem em importância e devem ser elaborados e fiscalizados [51].

Para os provedores, o foco é manter um ambiente seguro, dinamicamente escalável, com utilização mensurável e tarifável, ubiquamente acessível e facilmente compartilhado entre seus consumidores. Dessa forma, é imprescindível que tecnologias que suportem e implementem essa estrutura sejam utilizadas e desenvolvidas, como, por exemplo, segurança, virtualização [113], redes de banda larga [31], *datacenters* [121], tecnologias web, SOA (*Service Oriented Architecture*) [49], clusterização [9] e computação em *grid* [53].

2.1.1. Definição

Por se tratar de arquitetura tecnológica recente, ainda não se estabeleceu uma terminologia comum e existem diversas definições para o termo computação em nuvem [24] [62] [55]. Todavia, adota-se aqui, para efeitos de padronização e modularização, a definição do NIST (*National Institute of Standards and Technology*) [77].

Segundo o NIST em [77], a computação em nuvem é um modelo que permite acesso ubíquo, conveniente e sob demanda via rede à uma gama de recursos computacionais compartilhados, que podem ser rapidamente provisionados e disponibilizados com esforço gerencial ou interação do provedor mínimos. O modelo de computação em nuvem é composto de cinco características essenciais, quatro modelos de implantação e três modelos de serviço.

2.1.2. Características essenciais

As características essenciais descrevem como os recursos computacionais são disponibilizados como serviços no paradigma da computação em nuvem.

- **Serviço sob demanda**: um consumidor pode provisionar unilateralmente a capacidade computacional como desejado, como quantidade de processamento espaço de armazenamento, sem necessidade de interação humana com cada provedor de serviço.

- **Acesso ubíquo**: os recursos são disponibilizados via rede, especialmente pela Internet, e são acessíveis por mecanismos que promovem seu uso por plataformas heterogêneas, robustas ou leves, isto é, variando desde celulares até estações de trabalho ou servidores.

- **Compartilhamento de recursos**: os recursos computacionais do provedor são particionados de forma a servir múltiplos consumidores, utilizando modelos multiclientes, com diferentes recursos físicos ou virtuais sendo dinamicamente alocados ou realocados de acordo com a demanda dos consumidores. Estabelece-se um senso de independência de localização, onde o consumidor passa a desconhecer, ou mesmo tal informação passa a ter menos importância, a localização exata de seus recursos. Embora, porém, em uma camada de abstração maior, o consumidor talvez ainda seja capaz de especificar, por exemplo, o país, estado ou *datacenter* onde sua informação será alocada.

- **Rápida elasticidade:** recursos podem ser elasticamente provisionados, em alguns casos até automaticamente, de forma a crescer ou diminuir em

concordância com a demanda. A elasticidade é uma propriedade que se refere à capacidade de ajuste de um recurso particular, sem que outros sejam afetados, como, por exemplo, aumentar a capacidade de armazenamento sem que o processamento necessite ser alterado. Para o consumidor, os recursos disponíveis aparentam ser ilimitados e podem ser acomodados em qualquer quantidade a qualquer momento.

- **Serviço mensurável:** sistemas em nuvem automaticamente controlam e otimizam recursos realizando medições em determinado nível de abstração adequadas ao tipo de serviço sendo provisionado, como, por exemplo, armazenamento, processamento, banda, ou contas ativas de usuário. Essa utilização de recursos pode ser monitorada, controlada, e reportada, oferecendo transparência para ambos consumidores e provedores.

Não constante da definição do NIST [77], mas de menção relevante é a característica de **resiliência** da computação em nuvem. Resiliência é a capacidade que um sistema tem de se recuperar de alguma perturbação em suas condições normais de operação. As perturbações podem se referir a falhas inesperadas, intrusões, acidentes, ou mesmo a um acréscimo súbito e imprevisto de carga. A resiliência da computação em nuvem é normalmente obtida distribuindo os recursos de TI sendo oferecidos como serviços através de localidades físicas distintas, com transferência de carga automática em caso de problemas, mas se mantendo o acesso transparente e único.

2.1.3. Modelos de Implantação

Os modelos de implantação da computação em nuvem classificam como uma infraestrutura de nuvem é provisionada. E, uma infraestrutura de nuvem é o conjunto de *hardware* e *software* que possibilita a materialização das cinco características essenciais da computação em nuvem.

- **Nuvem privada**: a infraestrutura de nuvem é provisionada para uso exclusivo de uma única organização com múltiplos consumidores. Essa infraestrutura pode pertencer, ser gerenciada, e operada pela própria organização, terceiros, ou ainda

uma combinação desses; e, mais ainda, pode existir dentro ou fora da própria organização. A Figura 2.1 exemplifica esse tipo de modelo.

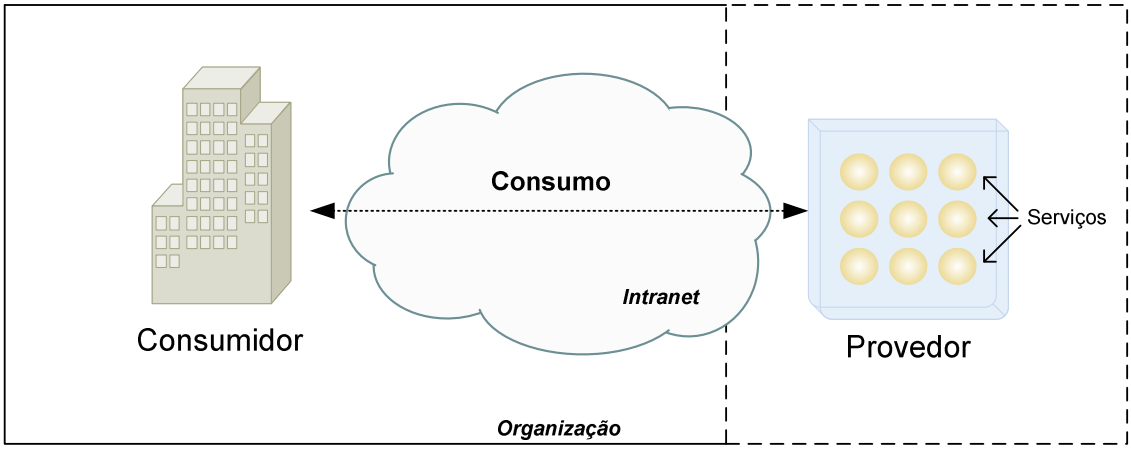

Figura 2.1 – Nuvem Privada [4]

- **Nuvem pública**: a infraestrutura de nuvem é provisionada para uso aberto ao público. Essa infraestrutura pode pertencer, ser gerenciada, ou operada por uma organização comercial, acadêmica, ou governamental, ou ainda uma combinação dessas. Todavia, a infraestrutura de nuvem pública existe sempre dentro do provedor. A Figura 2.2 mostra o modelo de nuvem pública.

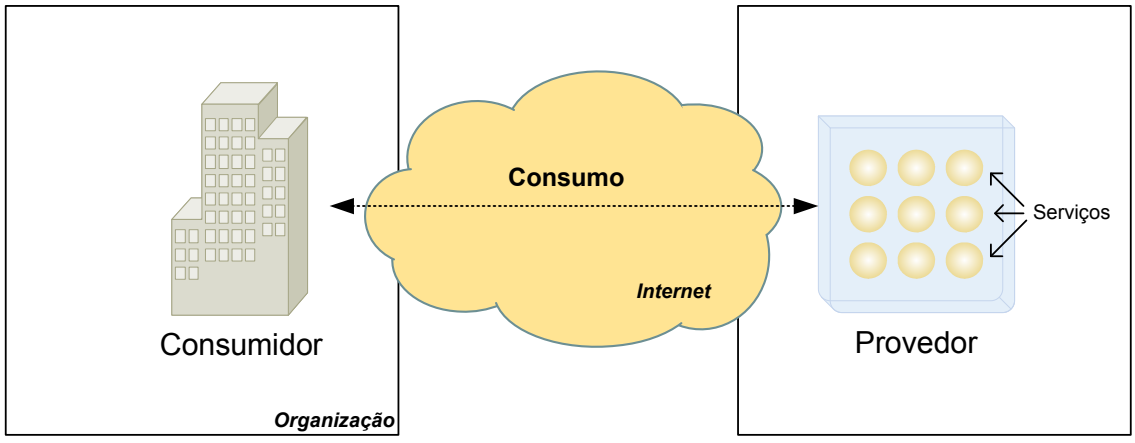

Figura 2.2 – Nuvem Pública [4]

- **Nuvem comunitária**: a infraestrutura de nuvem é provisionada para uso exclusivo de uma comunidade específica de consumidores de organizações que partilham interesses comuns. Essa infraestrutura pode pertencer, ser gerenciada, e operada por uma ou mais das organizações da comunidade, terceiros, ou ainda uma combinação desses; e, também, pode existir dentro ou fora das próprias organizações.

- **Nuvem híbrida**: a infraestrutura de nuvem é a composição de duas ou mais infraestruturas de nuvem (privada, comunitária, ou pública) que se mantêm entidades únicas, mas que compartilham tecnologia padronizada ou proprietária, permitindo portabilidade de dados e aplicações, como, por exemplo, balanceamento de carga entre nuvens.

2.1.4. Modelos de serviço

Os modelos de serviço, também conhecidos como modelos de entrega, definem a estratificação dos serviços sendo oferecidos. Estabelecem em qual nível de *hardware* e *software* o provedor operará. A Figura 2.3 apresenta o respectivo modelo de referência [51].

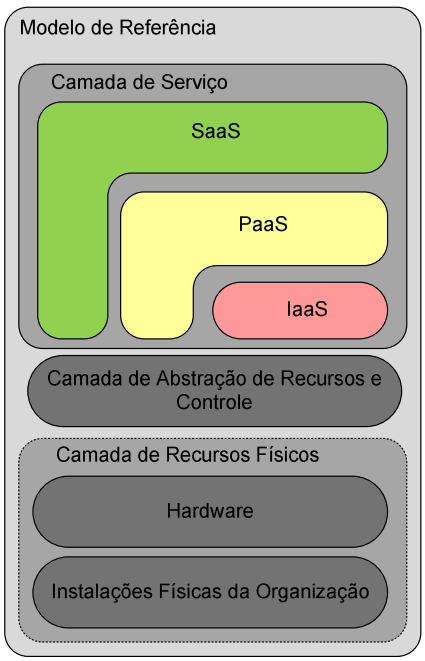

Figura 2.3 – Modelo de referência [51]

- **Infraestrutura com serviço (IaaS – *Infrastructure as a Service*)**: o recurso disponibilizado ao consumidor é a provisão de processamento, armazenamento, redes, e outros recursos computacionais fundamentais onde o consumidor é capaz de implantar e operar *softwares* arbitrários, que podem incluir sistemas operacionais e aplicações. O consumidor não gerencia nem controla a infraestrutura computacional basilar, mas tem controle sobre sistemas operacionais, armazenamento e aplicações implantadas; e possivelmente um controle limitado sobre componentes de rede específicos, como, por exemplo, *firewalls*.

- **Plataforma como serviço (PaaS – *Platfrom as a Service*):** o recurso disponibilizado ao consumidor é a capacidade de implantar, em uma infraestrutura de nuvem, aplicações criadas pelo próprio consumidor ou mesmo adquiridas, utilizando linguagens de programação, bibliotecas, serviços e ferramentas suportados pelo provedor. O consumidor não gerencia nem controla a infraestrutura computacional basilar, incluindo rede, servidores, sistemas operacionais, e armazenamento, mas tem controle sobre aplicações implantadas e possivelmente sobre configurações específicas do ambiente de aplicação.

- **Software como serviço (SaaS – *Software as a Service*):** o recurso disponibilizado ao consumidor é a utilização de aplicações de um provedor operando em uma infraestrutura de nuvem. As aplicações são acessíveis a partir de vários dispositivos clientes através de uma interface específica, como, por exemplo, navegadores ou interfaces programadas. O consumidor não gerencia nem controla a infraestrutura computacional basilar, incluindo rede, servidores, armazenamento, ou mesmo recursos individuais das aplicações. Todavia, em caráter de exceção, acesso limitado às configurações específicas de usuários em determinadas aplicações pode existir.

Alguns provedores têm introduzido também novos tipos mais especializados de modelos de entrega de serviço. Destacam-se aqueles relacionados com segurança

(*security-as-a-service*), banco de dados (*database-as-a-service*) e armazenamento (*storage-as-a-service*).

2.1.5. Atores da Computação em Nuvem

Além da definição do NIST [77] para computação em nuvem adotada nesta tese, o NIST também estabelece uma arquitetura de referência [8], focada na instituição dos atores da computação em nuvem e suas relações. São eles:

- **Consumidor**: Aquele que faz uso de um serviço de nuvem.
- **Provedor**: Aquele que disponibiliza o serviço a ser consumido.
- **Transportador**: O transportador de nuvem (*cloud carrier*) é a entidade responsável por atuar como intermediária na provisão de conectividade e transporte entre os consumidores e os provedores da computação em nuvem
- **Auditor**: O auditor de nuvem (*cloud auditor*) é a entidade responsável por realizar verificações independentes dos serviços de nuvem, com o objetivo de atestar a conformidade das especificações anunciadas pelos provedores.
- **Corretor**: Foca-se no papel da intermediação das interações entre consumidor e provedor.

2.1.6. Aspectos de Segurança Específicos da Computação em Nuvem

No ambiente de nuvem, existem aspectos de segurança específicos introduzidos pelo paradigma e que influenciam sobremaneira qualquer abordagem de segurança e privacidade. Destacam-se as fronteiras de segurança, questões legais e regulatórias, contratos e confiança [51].

2.1.6.1. Fronteiras de Segurança

O primeiro aspecto de segurança inédito introduzido pela computação em nuvem é a ampliação das fronteiras de segurança do consumidor. Agora, ao invés de lidar com uma infraestrutura de TI localizada nas próprias instalações, onde o controle é completo,

o consumidor terá incluídos em suas fronteiras de segurança equipamentos fisicamente distantes, com administração e políticas de segurança em operação não necessariamente iguais ou mesmo similares às suas.

Além disso, como um provedor tipicamente abriga outros consumidores, que compartilham os mesmos recursos, instala-se assim uma sobreposição de fronteiras de segurança, com múltiplos interesses e políticas atuando em um mesmo ambiente.

2.1.6.2. Questões Legais e Regulatórias

Em relação às questões legais e regulatórias das operações em nuvem, o consumidor deve atentar para o fato de que seus dados serão remotamente armazenados e/ou processados.

Em certas ocasiões isso poderá não estar de acordo com a regulamentação local. Se um provedor decidir mover seu *datacenter* de um país para outro, consequentemente os dados por ele processados ou armazenados serão realocados. Essa movimentação das informações do consumidor pode violar aspectos legais tanto do país de origem quanto do destino. A diretiva 95/46/EC [122] da união europeia, por exemplo, determina como informações pessoais de cidadãos europeus devem ser tratadas, ditando que registros de cidadãos europeus não podem ser transferidos para outros países sem que haja garantia formal da adequada proteção dessas informações.

2.1.6.3. Contratos

Normalmente, um contrato de serviço de nuvem tende a simplificar seus termos em favor de um documento padronizado. Esses contratos padronizados são impostos aos consumidores pelos provedores. Todos os contratos têm o mesmo nível de detalhamento e são entregues quase imediatamente, similarmente ao provisionamento sob demanda de recursos, característico da computação em nuvem.

Em nome da eficiência, SLAs, políticas de segurança, e outros aspectos específicos de contrato podem acabar sendo aglutinados e conformados em um documento voltado aos

interesses do provedor, e nem sempre será adequado às necessidades específicas de cada consumidor.

2.1.6.4. Confiança

Ao acessar serviços na nuvem, consumidores passam a confiar que seus provedores irão cumprir as obrigações estabelecidas. Os provedores tem substancial controle administrativo sobre o ambiente de nuvem, e não importando as circunstâncias terão alguma medida de responsabilidade sobre os recursos do consumidor. Os consumidores esperam que os recursos de nuvem sejam manipulados responsavelmente, tenham alta disponibilidade e sejam gerenciados como definido nos contratos. Estabelece-se aqui uma relação de confiança, fundamental para a segurança de nuvem.

Todavia, na maioria das vezes, os contratos não são detalhados o suficiente para abarcar todos os aspectos de segurança aplicáveis e, mais importante, frequentemente uma quebra de confiança será mais danosa ao consumidor do que a eventual compensação contratual poderá atenuar.

Assim sendo, hoje em dia, a utilização de um provedor confiável é um aspecto de segurança fundamental de computação em nuvem.

2.2. MODELO DE CONSUMO DE SERVIÇOS EM SAAS

Dentro dos modelos de entrega da computação em nuvem, o SaaS lidera em implantações por parte dos provedores e em adoção ou planejamento de adoção pelos consumidores. Em pesquisa realizada por *Hubspan* em 2010, mostrada na Figura 2.4 [45], das empresas que declararam estarem envolvidas com implantações de soluções em nuvem, 70% estavam focadas no modelo SaaS. Em outra pesquisa mais recente, de 2012, promovida pelo *1105 Government Information Group* em [1], o modelo SaaS se mantém como a solução mais adotada e também como a elegida para futuras implantações.

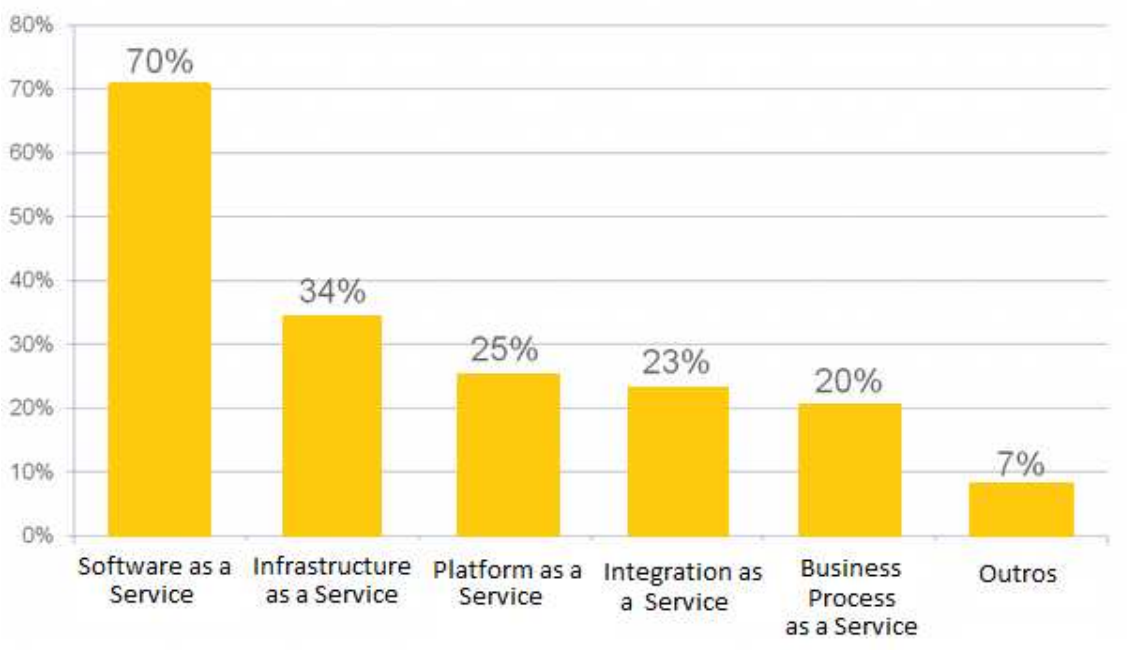

Figura 2.4 – Penetração dos modelos de entrega da computação em nuvem [45].

A adoção preferencialmente do SaaS sobre IaaS e PaaS é considerada natural, pois é o modelo que permite a manutenção da infraestrutura existente e a gradual expansão ou opção por aplicações disponíveis e já estabelecidas com sucesso [1].

Essa agilidade e a redução de custos proveniente da exclusão dos gastos de investimento envolvidos na implementação e na manutenção de um ambiente ou plataforma em *hardware* e *software* dentro do modelo SaaS não têm excluído, contudo, o fato de que a preocupação com a privacidade tem freado a adoção em massa da tecnologia por empresas e indivíduos que não podem ou escolhem não correr o risco de terem seus dados expostos a um ambiente onde não têm total controle sobre os mecanismos de segurança em operação [2].

Considerando, portanto, a introdução da privacidade em SaaS como a motivação desta tese, deve-se estabelecer como o consumo de serviços SaaS transcorre, para que, então, uma análise de privacidade possa ser feita nesse cenário e uma tecnologia de privacidade adequada possa ser selecionada.

2.2.1. Ambiente SaaS

O ambiente típico SaaS consiste em uma organização, ou indivíduo, assumindo o papel de consumidor e fazendo uso de um serviço oferecido por um provedor na nuvem. O modelo SaaS, como visto na descrição da computação em nuvem, define como uma aplicação que implementa uma funcionalidade em *software* é disponibilizada por um provedor, obedecendo as características da computação em nuvem.

Uma aplicação disponibilizada com um serviço SaaS é normalmente uma aplicação *web*, acessível via rede. O consumo desse serviço requer uma troca de mensagens, que tradicionalmente ocorre como interações ponto-a-ponto desacopladas, empregando MEPs[1] simples e tipicamente fazem uso de HTTP [52].

Sem perda de generalidade, analisam-se aqui interações individuais, tais como as realizadas por *web services*. Essas interações podem ser compostas para formar aplicações complexas, com as funcionalidades necessárias para constituir as aplicações *web* [50], a serem então disponibilizadas como serviços dentro do paradigma SaaS.

2.2.1.1. Web Service

Web service, de acordo com a definição do W3C (*World Wide Web Consortium*) [131], é um sistema em *software* projetado para suportar a interoperabilidade nas interações entre dispositivos computacionais via rede. É uma tecnologia focada na realização de conexões. Normalmente, existe um contrato de serviço que especifica exatamente como serão as interações suportadas na conexão, determinando os padrões de troca de mensagens para cada operação prevista no serviço. Esse contrato de serviço pode ser, por exemplo, do tipo WSDL [39].

O WSDL – *Web Services Description Language* – é uma linguagem baseada em XML [128], e é utilizada para descrever a funcionalidade oferecida por um *web service*.

[1] MEP (*Message Exchange Pattern*), no contexto de serviços, descreve o padrão de mensagens requeridas por um protocolo de comunicações ao estabelecer ou utilizar um canal de comunicações. Existem dois grandes padrões de troca de mensagens: pergunta-resposta e sentido único [49].

Abaixo, na Figura 2.5, apresenta-se, como exemplo, um WSDL referente a um serviço chamado *HelloService*.

HelloService

```
<?xml version="1.0" encoding="UTF-8" ?>
<definitions name="HelloService"
      targetNamespace="http://www.examples.com/wsdl/HelloService.wsdl"
      xmlns="http://schemas.xmlsoap.org/wsdl/"
      xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
      xmlns:tns="http://www.examples.com/wsdl/HelloService.wsdl"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <message name="SayHelloRequest">
    <part name="firstName" type="xsd:string"/>
  </message>
  <message name="SayHelloResponse">
    <part name="greeting" type="xsd:string"/>
  </message>
  <portType name="Hello_PortType">
    <operation name="sayHello">
      <input message="tns:SayHelloRequest"/>
      <output message="tns:SayHelloResponse"/>
    </operation>
  </portType>
  <binding name="Hello_Binding" type="tns:Hello_PortType">
  <soap:binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="sayHello">
    <soap:operation soapAction="sayHello"/>
    <input>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:examples:helloservice"
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:examples:helloservice"
        use="encoded"/>
    </output>
  </operation>
  </binding>
  <service name="Hello_Service">
    <documentation>WSDL File for HelloService</documentation>
    <port binding="tns:Hello_Binding" name="Hello_Port">
      <soap:address
        location="http://www.examples.com/SayHello/">
    </port>
  </service>
</definitions>
```

Figura 2.5 – WSDL Exemplo

O documento acima, por ser um WSDL, é basicamente composto de definições. Ele inicia por um elemento na raiz denominado de ***definitions***, que encapsula as definições propriamente ditas, sendo compostas das seguintes partes:

- ***Types***: Provê a definição dos tipos a serem empregados na descrição das mensagens trocadas. Normalmente utilizam esquemas XSD[2].

- ***Message***: Representa uma definição abstrata e tipificada dos dados sendo transmitidos.

- ***Port Type***: É um conjunto abstrato de operações suportado por um ou mais *endpoints*[3].

- ***Binding***: Institui uma especificação concreta de protocolo e formato de dados para um ***Port Type*** específico.

- ***Service***: Tem a função de agregar ***Ports***[4] relacionadas.

Embora *web services* possam ser descritos utilizando-se o WSDL, a implementação dos *web services* se divide tradicionalmente em RESTful ou baseados em SOAP.

Web services RESTful são implementados utilizando a arquitetura REST (*REpresentational State Transfer*) [109]. *Web services* RESTful utilizam o protocolo HTTP [52] para suportar suas operações e, portanto, não existe um padrão oficial para essa modalidade de serviços. O conjunto de operações possíveis são aquelas providas pelos próprios métodos HTTP, ou seja, GET, PUT, POST e DELETE.

Web services baseados em SOAP são aqueles *web services* que obrigatoriamente empregam um documento WSDL para sua descrição e que utilizam a especificação

[2] Um XSD, ou *XML Schema Definition*, é uma linguagem de esquemas XML, publicada como recomendação pelo W3C [130], com o objetivo de possibilitar a definição de uma coleção de metadados abstrata, descrevendo suas características de forma compartimentalizada, evitando problemas de ambiguidade de dados e possibilitando clareza no processamento dos dados de mensagens XML.
[3] Um *endpoint*, dentro do contexto de *web services*, se refere ao ponto de conexão onde arquivos *web* ou páginas ativas de servidores são expostas.
[4] ***Port***, dentro do contexto de *web services*, é basicamente a definição de um endpoint simples, sendo constituído da combinação de um ***Binding*** com um endereço de rede.

SOAP para a troca de mensagens. Como tanto o WSDL quanto o SOAP empregam XML, *web services* baseados em SOAP usam exclusivamente mensagens XML.

O SOAP (*Simple Object Access Protocol*) [129] é um protocolo focado na troca de informação **estruturada** entre aplicações em *software* e faz uso do XML. Estabelece-se uma forma das aplicações se comunicarem, independentemente da plataforma utilizada ou da tecnologia de implementação. É um padrão aberto e uma recomendação do W3C.

O SOAP essencialmente disponibiliza um envelope para o envio das mensagens dos *web services* e a Figura 2.6 mostra a estrutura de uma mensagem SOAP.

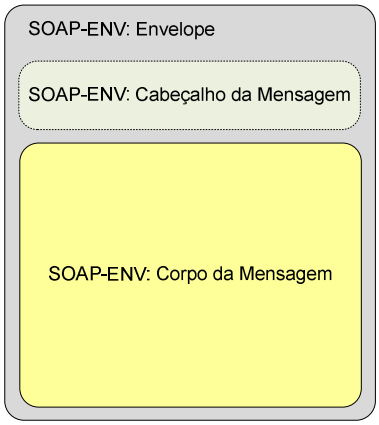

Figura 2.6 – Estrutura de Mensagem SOAP

O envelope contém um cabeçalho opcional e o corpo da mensagem, que é obrigatório. O cabeçalho carrega metadados relacionados com o processamento da mensagem, como, por exemplo, controle de transações, roteamento e segurança. Já o corpo da mensagem contém a mensagem SOAP propriamente dita.

De destacar, a especificação do SOAP permite que extensões sejam incorporadas às transações e, por exemplo, pode-se fazer uso do WS-Security [92] para a segurança das mensagens, ou do WS-ReliableMessaging [91] para a entrega confiável de mensagens.

2.2.2. Cenário Modelo de Consumo de Serviços SaaS

Considerando, então, o emprego de *web services* como apresentado acima, dentro do ambiente SaaS, define-se agora o cenário modelo de consumo de serviços SaaS em nuvem que será analisado neste trabalho. A Figura 2.7 demonstra esse cenário, e, a seguir, suas etapas são descritas.

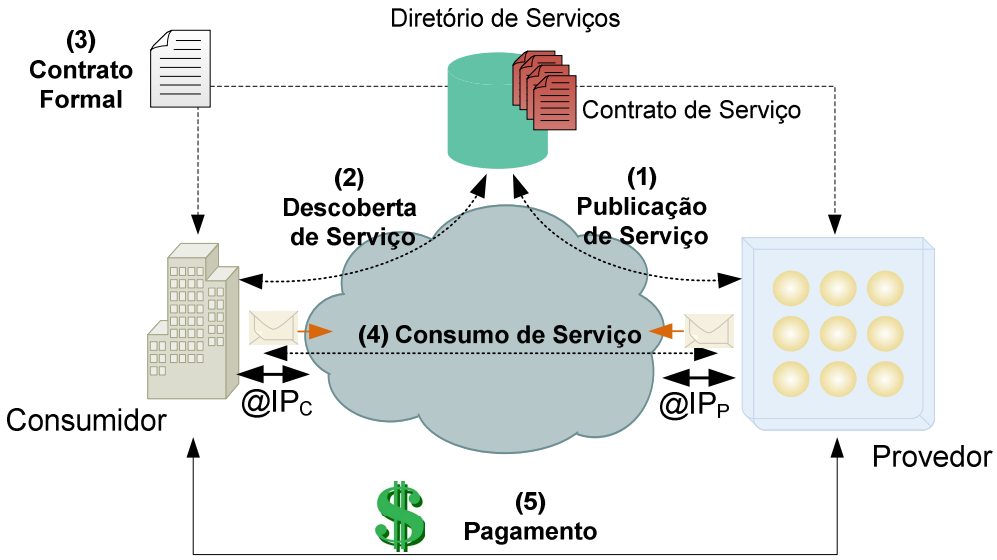

Figura 2.7 – Cenário modelo do consumo de serviço em nuvem SaaS

- **Etapa 1**: Primeiramente, o provedor do serviço publica o serviço em conjunto com as respectivas condições de uso (Figura 2.7, etapa 1). Serviços publicados são descritos nos chamados contratos de serviço, como, por exemplo, especificações WSDL. Este contrato de serviço não é o contrato formal consumidor-provedor e sim um documento que contém informação acerca das funcionalidades oferecidas pelo serviço, e dos detalhes técnicos de como acessar o serviço, tais como protocolos de transporte e endereços de rede. Mais ainda, neste documento acomodam-se também condições não operacionais de funcionamento, como políticas de provisão. Essas políticas proveem informação sobre o provedor e condições de uso do serviço, do tipo SLA e precificação. Contratos de serviço são usualmente tornados públicos através de diretórios de

serviço, que podem ser gerenciados por uma entidade terceira, ou pelo próprio provedor.

- **Etapa 2**: Em seguida, o consumidor interage com o diretório de serviços para localizar o serviço apropriado que forneça a funcionalidade desejada e avaliar as condições de consumo. Esta etapa é denominada descoberta de serviço (Figura 2.7, etapa 2). A especificação UDDI [90] provê uma estrutura para o estabelecimento de um diretório de serviços, suportando tanto a etapa da publicação de serviço quanto a da descoberta de serviço. Todavia, as etapas 1 e 2 deste cenário são opcionais, destarte não sendo essenciais para este tipo de consumo de serviços em nuvem SaaS.

- **Etapa 3**: Como próxima etapa (Figura 2.7, etapa 3), consumidor e provedor estabelecem o contrato formal. Esta etapa é realizada fora da banda, ou seja, não compõe a troca de mensagens e pode até ser realizada na forma de documento não digital. Tipicamente, entretanto, o consumidor concretiza esta etapa na plataforma de serviço do próprio provedor, concordando com um EULA (*End User License Agreement*), que é o contrato, e pagando ou pré-autorizando o pagamento do uso respectivo.

- **Etapa 4**: O consumidor, na etapa 4 da Figura 2.7, faz a requisição de uso e recebe os resultados do serviço empregando a troca de mensagens, realizando o consumo propriamente dito do serviço. Essas mensagens usualmente contêm metadados embutidos que são empregados na autenticação, autorização e contabilização do consumo do serviço.

- **Etapa 5**: Finalmente, o provedor fatura o serviço e faz a cobrança ao consumidor de acordo com a quantidade de serviço consumida e, então, recebe o pagamento devido (Figura 2.7, etapa 5).

2.3. ABORDANDO A PRIVACIDADE DO CONSUMIDOR

Existe diferença na definição dos termos privacidade e privacidade da informação. Enquanto o termo privacidade se relaciona a um conceito mais amplo da pretensão a um

domínio de influência pessoal, privacidade da informação tem seu foco em dados. Roger Clarke em [40] define privacidade da informação como:

"Privacidade da informação é o interesse que uma entidade tem em controlar, ou pelo menos significativamente influenciar, a manipulação dos dados referentes a si".

No propósito desta tese, considerando a privacidade da informação no contexto da computação em nuvem e do modelo SaaS, adotar-se-á a simplificação do termo para apenas privacidade.

Na Seção 2.1.6, foram apresentados os aspectos de segurança específicos introduzidos pelo paradigma da computação em nuvem. Fator comum entre eles é o papel de poder exercido pelo provedor. Na expansão das fronteiras de segurança, o provedor agora tem influência na manipulação dos dados do consumidor, impactando nas próprias políticas de segurança antes somente internas ao consumidor. Em relação à regulamentação das operações em nuvem, um provedor agora pode realocar sem aviso os dados de um consumidor e o sujeitar a alterações de legislações aplicáveis. Nos contratos também o provedor tem uma influência considerável, pois geralmente força contratos padrões sobre seus consumidores, que não necessariamente atendem aos requisitos específicos de segurança de cada consumidor. E, finalmente, o aspecto da confiança estabelecida com o provedor é ponto crucial da privacidade de consumidor. O dano que uma eventual quebra de confiança por parte do provedor, vazando, por exemplo, informações estratégicas sendo por ele processadas ou armazenadas, trará quase sempre muito mais prejuízos ao consumidor do que uma possível penalidade imposta por contrato possa compensá-lo.

A informação do consumidor de nuvem é, portanto, sujeita a um número crescente de ameaças, sendo que a maioria delas relaciona-se com a exposição de informações à rede, ao ambiente compartilhado do provedor, que pode admitir acesso não autorizado dos outros clientes (comumente chamados *tenants*, em inglês); e, no limite, ao próprio provedor.

Em documento [42] emitido pelo CSA – Cloud Security Alliance^{SM} – compila-se uma lista das ameaças consideradas mais relevantes na abordagem da segurança da

computação em nuvem. O objetivo desse documento, juntamente com o guia para a segurança nas áreas críticas da computação em nuvem [41], também disponibilizado pelo CSA, é auxiliar as organizações na abordagem da segurança na sua estratégia de adoção da computação em nuvem. Aqui, esses documentos atuaram como insumos na identificação das entidades envolvidas na ameaça à privacidade do consumidor.

Portanto, dentre as fontes de ameaça à privacidade do consumidor, incluindo, por exemplo, um espião de rede ou um cliente malicioso compartilhando o ambiente do provedor (*tenant*), nenhuma terá mais acesso às informações do consumidor do que o próprio provedor.

Mais do que isso, o provedor, por ser quem efetivamente processa o serviço, precisa ter acesso completo aos dados do consumidor. Mesmo, então, que uma nuvem pública esteja em uso, sujeitando o consumo de serviço a tentativas de investigação não autorizadas das comunicações em curso por terceiros, o provedor na situação normal de operação já estará em situação de controle e manipulação das informações do consumidor. E, para mitigar essas tentativas de espionagem de terceiros das comunicações, a criptografia tradicional pode ser empregada [78], tornando as comunicações entre consumidor e provedor confidenciais.

Nos últimos anos, dentro da área da criptografia, vem se desenvolvendo o uso dos criptossistemas homomórficos [108], que permitem a realização de operações matemáticas diretamente em dados cifrados. Isso poderia prover privacidade aos consumidores em relação aos provedores, mas mesmo que as questões de desempenho [69] do uso dos criptossistemas homomórficos sejam superadas, aspectos privados do consumidor continuariam sendo impactados, como, por exemplo, a própria identificação do consumidor para provedor no momento do consumo.

Assim sendo, na abordagem desta tese, considerar-se-á o provedor de nuvem como a possível origem de mau uso das informações privadas do consumidor. Ademais, prover privacidade nas interações entre consumidor e provedor irá, na maioria dos casos, também aprimorar a privacidade em relação a outras entidades menos impactantes nesse contexto.

2.4. ANÁLISE DE PRIVACIDADE DO CONSUMIDOR SAAS

Levando-se em conta os aspectos de segurança específicos do ambiente de computação em nuvem (Seção 2.1.6) e o provedor como a entidade com maior acesso às informações sensíveis do consumidor (Seção 2.3), voltou-se para o cenário modelo do consumo de serviços SaaS (Seção 2.2.2) com o objetivo de efetivar uma abordagem de proteção de privacidade para o consumidor SaaS. A escolha foi modulalizar o enfoque, de forma a tornar a solução objetiva, flexível e, mais importante, abrangente.

Nesse sentido, divide-se o cenário modelo de consumo de serviços SaaS em três níveis: **contrato**, **troca de mensagens** e **comunicações de rede**. Cada nível contém itens de interesse do consumidor que carregam a informação privada do consumidor. Ademais, cada um desses itens impacta de uma maneira diferente a privacidade do consumidor, originando, por conseguinte, a classificação da privacidade em dimensões e a associação do impacto de cada item de interesse com os aspectos específicos de privacidade sendo influenciados.

A inter-relação da tríade (**níveis de interação**, **itens de interesse** e **dimensões de privacidade)** dá origem a **análise de privacidade do consumidor SaaS**, e é a primeira contribuição original da tese. Oferece-se uma abordagem inédita, modular, flexível e abrangente da situação da privacidade de um consumidor SaaS dentro do paradigma da computação em nuvem.

Em relação aos níveis de interação entre consumidor e provedor, o primeiro nível, o de contrato, é uma interação fora da banda e se refere ao estabelecimento de um contrato legal, especificando os termos de utilização do serviço, como condições de sigilo, SLA (*Service Level Agreement*) ou tarifação. O segundo nível consiste na troca de mensagens durante o consumo do serviço propriamente dito. E, o terceiro nível se refere às comunicações de rede.

Para os itens de interesse, no nível de contrato, identificam-se os dados privados presentes no próprio contrato, que são: a identificação do consumidor, o endereço do consumidor e local de negócios e o SLA e estimativas de consumo. No nível de troca de mensagens, têm-se as credenciais empregadas para autenticação do consumidor, as

MEPs no tempo (padrão de consumo), e os próprios dados da mensagem. Já para o nível de comunicações de rede, considera-se apenas o endereço IP como item de interesse que impacta a privacidade do consumidor. Embora, no entanto, dentro de um pacote IP, a carga do pacote também inclua os dados da mensagem, considerar-se-á que o item de interesse dos dados da mensagem no nível de troca de mensagens representará essa informação e o tratamento aplicado nele será propagado naturalmente para a carga do pacote IP.

Quanto à classificação da privacidade em dimensões, estabelecem-se quatro para o conceito de privacidade: **ID**, **localização**, **comportamento** e **conteúdo**.

- **Privacidade de ID:** refere-se ao objetivo de não revelar a informação de identificação inequívoca de uma entidade. Na computação em nuvem poderá assumir várias formas, como, por exemplo, nome da empresa, credencial de acesso ou endereço de rede. Em um mercado global, pode ser do interesse estratégico de uma empresa não revelar sua identidade ao consumir serviços, principalmente no caso de nuvens públicas.

- **Privacidade de localização:** refere-se ao objetivo de não revelar a localização física de uma entidade. Na computação em nuvem manifesta-se na forma do endereço; podendo ter a granularidade exata da localização, ou parcial, do tipo, por exemplo, do país. É importante no contexto da computação em nuvem a partir do momento que leis e tributos aplicáveis podem variar de acordo com a localização do consumidor.

- **Privacidade de comportamento:** refere-se ao objetivo de não revelar a conduta de uma entidade em relação ao tempo. É importante, pois o comportamento pode revelar informações estratégicas como volume de negócios e a sazonalidade de operações críticas. Ademais, o tamanho e a penetração de uma empresa podem ser inferidos de seu comportamento.

- **Privacidade de conteúdo:** refere-se ao objetivo de não revelar o conteúdo referente a uma entidade. Na computação em nuvem, são os dados do consumo

de serviços. Carrega frequentemente informações estratégicas de uma empresa consumidora.

Na Figura 2.8 visualiza-se a divisão do consumo de serviços SaaS em níveis, seus itens de interesse e a introdução das dimensões de privacidade.

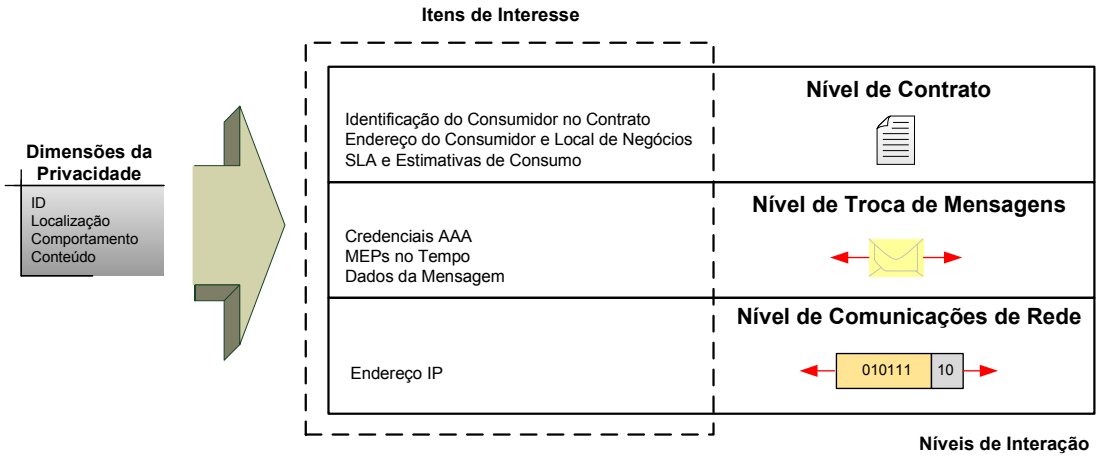

Figura 2.8 – Níveis de Interação Consumidor-Provedor, Itens de Interesse e Dimensões de Privacidade

A seguir, a análise da privacidade do consumidor SaaS é apresentada. Cada um dos níveis de interação é avaliado, demonstrando como cada uma das dimensões da privacidade é afetada pelo paradigma de consumo de serviços SaaS. Expõe-se, também, como os respectivos itens de interesse são impactados.

2.4.1. Nível de Contrato Consumidor-Provedor

O contrato consumidor-provedor contém informação que permite a identificação direta do consumidor e, tipicamente, pré-especifica o volume de serviço a ser consumido.

Para a privacidade de ID, o item de interesse é a identificação do consumidor no contrato. Informação acerca do endereço do consumidor e local de negócio também normalmente se encontram nos contratos, impactando diretamente a privacidade de localização. A privacidade de comportamento também pode ser afetada, na medida em

que um contrato pode ter seções especificando volume de serviço esperado e SLA respectivo.

A privacidade de conteúdo não é afetada, uma vez que o nível de interação do contrato consumidor-provedor é fora da banda do consumo de serviço propriamente dito.

2.4.2. Nível de Troca de Mensagens

O provedor precisa autenticar, autorizar e contabilizar as trocas de mensagem no consumo de serviço. Isso é necessário para que o provedor possa cobrar corretamente o consumidor pelo consumo do serviço, sob demanda. Autenticação, autorização e contabilização formam o processo conhecido como AAA[5] (do inglês, *Authentication, Authorization and Accounting*). O AAA é usualmente baseado em credenciais contidas nos metadados das mensagens. Exemplos destes tipos de credenciais são assinaturas e certificados digitais encapsulados em elementos de WS-Security [92] e credenciais SAML [37].

Credenciais AAA típicas estabelecem uma ligação direta para o ID do consumidor. Mais ainda, caso esta ligação possa ser executada para todas as mensagens, por meio de análise das MEPs, o provedor poderá correlacionar as mensagens de consumo no tempo, impactando não somente a privacidade de ID, como também a de comportamento.

Em relação ao conteúdo, os dados das mensagens carregam os dados do consumidor. Essa é a informação necessária para executar as computações necessárias para o efetivo consumo do serviço. Os dados das mensagens, como item de interesse, afetam a privacidade de ID, a de localização, a de comportamento e a de conteúdo. Vis-à-vis a privacidade de ID, localização e comportamento, os dados das mensagens podem ser inspecionados de forma a revelar o ID, o comportamento e a localização do consumidor, já que os dados podem carregar ligações semânticas referentes a esses aspectos. Para a privacidade de conteúdo, os dados das mensagens constituem o conteúdo propriamente dito do consumidor.

[5] Autenticação diz respeito à identificação da entidade a realizar a operação. Autorização refere-se à permissão da entidade para realizar a operação pleiteada. E, contabilização consiste no registro objetivo das operações realizadas.

2.4.3. Nível de Comunicações de Rede

As mensagens do consumo de serviço são enviadas pela rede, tipicamente via Internet, o que implica que tanto consumidor quando provedor conhecem os endereços de rede um do outro.

Endereços de rede, nesse caso endereços IPs, impactam as privacidades de ID, localização e comportamento. Os esquemas de endereçamento IP atuais conseguem ligar essa identificação ao seu proprietário facilmente, além de rapidamente também poderem mostrar a posição geográfica [18]. Uma análise de tráfego focada, por exemplo, nas comunicações TCP transportando as mensagens, pode expor o comportamento do consumidor, revelando a correlação entre os consumos sucessivos de serviços e o tempo em que ocorreram.

2.4.4. Visão Geral da Privacidade dos Níveis de Interação Consumidor-Provedor

Na Tabela 2.1 abaixo, resume-se a análise teórica da privacidade em SaaS. Apresenta-se como os itens de interesse desse ambiente são afetados pelos níveis de interação consumidor-provedor no que tange as quatro dimensões da privacidade do consumidor.

Tabela 2.1 – Resumo do Impacto nas Dimensões da Privacidade pelos Níveis de Interação Consumidor-Provedor.

Nível de Interação Consumidor-Provedor	**Itens de Interesse**	**Dimensão da Privacidade Afetada**
Contrato Consumidor-Provedor	Identificação do Consumidor no Contrato	ID
	Endereço do Consumidor e Local de Negócios	Localização
	SLA e estimativas de consumo	Comportamento
Troca de Mensagens	Credenciais AAA	ID
	MEPs no tempo	Comportamento
	Dados das Mensagens	ID, Localização, Comportamento, Conteúdo
Comunicações de Rede	Endereço IP	ID, Localização, Comportamento

Vale ressaltar que a importância dada a cada dimensão de privacidade apresentada pode variar de consumidor para consumidor. Uma organização, por exemplo, pode entender que estrategicamente precisa apenas manter privados seu ID e seu conteúdo durante o consumo de serviços SaaS, não sendo vital que seu comportamento e localização sejam mantidos resguardados.

Contudo, a análise da privacidade em SaaS desenvolvida possibilita o mapeamento modular de todas as dimensões da privacidade do consumidor sendo afetadas durante as transações. Mais ainda, essa abordagem modularizada da privacidade permitirá uma proteção abrangente e eficaz do consumidor.

3. ESTRUTURA MULTICAMADAS DE ANONIMATO PARA SAAS

Com o mapeamento fornecido pela análise de privacidade, resumido na Tabela 2.1, tem-se como todos os itens de interesse de um consumidor impactam sua privacidade durante um consumo de serviço SaaS, dentro do cenário modelo apresentado (Seção 2.2.2). Verifica-se, porém, que essas informações são necessárias para a computação propriamente dita do serviço pelo provedor. Elas não podem ser simplesmente não reveladas ou cifradas para o provedor. A criptografia pode apenas manter a confidencialidade dos dados em relação a terceiros.

Discutiu-se, também, na Seção 2.3, o fato de que atualmente existem os criptossistemas homomórficos [108], que até permitem a realização de operações matemáticas em dados cifrados (sem se entrar no mérito da performance), mas que, ainda assim, não excluíram o problema de que o consumidor ainda expõe outros itens de interesse privados ao provedor SaaS, como, por exemplo, sua identificação no contrato. A Tabela 2.1 indica os itens de interesse que também impactam a privacidade do consumidor, além dos dados das mensagens.

É necessária a seleção de uma tecnologia que possa prover privacidade, enquanto mantém disponível o acesso aos itens de interesse em questão. O anonimato é, destarte, a tecnologia que se propõe a permitir que transações ocorram, sem que a privacidade da entidade envolvida seja exposta.

Formalmente, o anonimato é um termo que se refere ao estado de não revelar informação que possa unicamente identificar uma entidade. No contexto específico da segurança da informação, o trabalho de Andreas Pfitzmann destaca-se na tratativa objetiva do assunto. Em [103], estabelece-se terminologia relevante sobre o tema e, notadamente, define-se anonimato como:

"Anonimato de uma entidade significa que essa entidade não é identificável dentro de um conjunto de entidades, o conjunto de anonimato[6]".

O anonimato será, portanto, utilizado para introduzir a privacidade no consumo de serviços SaaS. Cada item de interesse presente em cada camada de interação consumidor-provedor será abordado pelo anonimato, protegendo respectivamente as dimensões de privacidade envolvidas.

Como visto, a análise da privacidade nas interações consumidor-provedor do cenário modelo SaaS aponta para um enfoque modular de proteção de privacidade, ou seja, para a aplicação do anonimato em camadas, protegendo cada nível de interação com uma abordagem de anonimato específica.

Nesse contexto, é útil a divisão do anonimato em dois tipos: anonimato de dados e anonimato de conexão [46]. O anonimato de dados é aquele que se preocupa com a retirada da possibilidade de identificação de uma entidade a partir da análise semântica de um conteúdo, ao passo que o anonimato de conexão é aplicável na proteção das identidades de remetente e destinatário durante uma comunicação específica.

Existem tecnologias de anonimato específicas tanto para o anonimato de dados quanto para o de conexão. Portanto, idealizou-se uma estrutura conceitual em camadas, que permite a delineação e a aplicação modular de tecnologias específicas de anonimato de acordo com a demanda e as características dos itens de interesse em cada nível de interação. Institui-se, então, a segunda contribuição original da tese, a denominada **estrutura multicamadas de anonimato para SaaS**.

Foram estabelecidas quatro camadas para essa estrutura, em concordância com a análise teórica da privacidade. As camadas propostas são: **contrato**, **metadados**, **rede** e **conteúdo**. As camadas de contrato e de rede correspondem aos níveis de interação consumidor-provedor de contrato e de rede, respectivamente. Faz-se uso, em ambas, do anonimato de conexão. Já o nível de interação da troca de mensagens, devido às

[6] O conjunto de anonimato inclui todas as entidades a serem analisadas por um atacante, com o objetivo de determinar o participante da ação sob exame.

características de seus itens de interesse em relação aos tipos de anonimato, é melhor abordado quando dividido nas camadas de metadados e de conteúdo. A camada de metadados empregará o anonimato de conexão, enquanto a camada de conteúdo utilizará o anonimato de dados. A Figura 3.1 apresenta como a concepção da estrutura multicamadas é derivada da análise de privacidade do cenário modelo de consumo serviços SaaS.

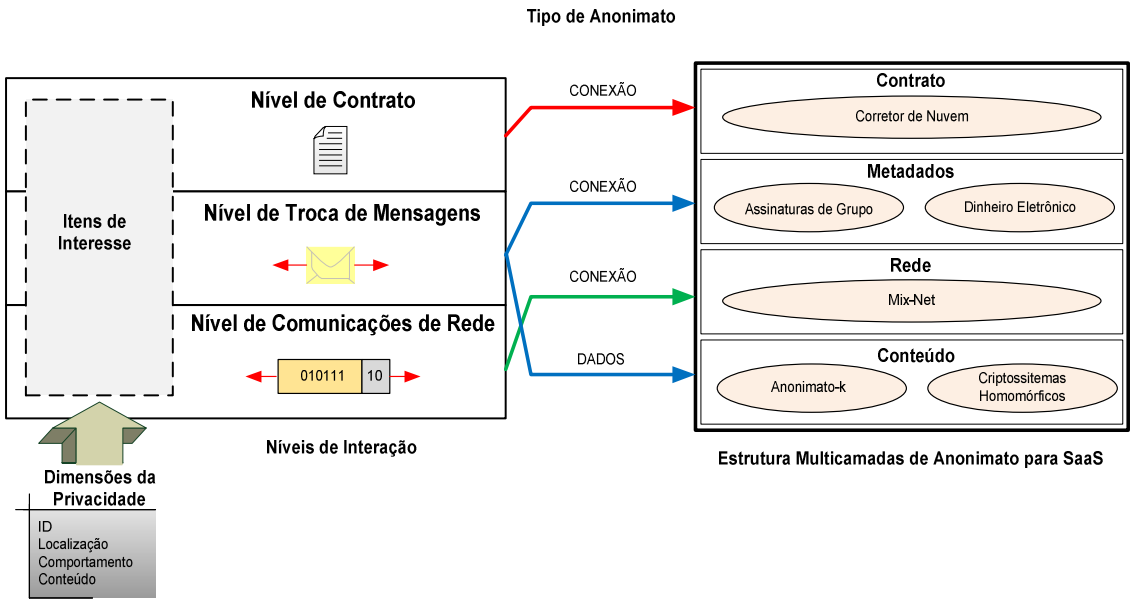

Figura 3.1 – Concepção da Estrutura Multicamadas de Anonimato para SaaS

Cada camada da estrutura usa tecnologias de anonimato específicas para proteger os respectivos itens de interesses. O projeto multicamadas é uma característica proeminente da estrutura proposta: a tecnologia de anonimato utilizada em cada camada pode ser trocada ou ajustada de forma a atender requisitos de privacidade específicos e distintos de cada consumidor. O objetivo é que o design seja flexível o bastante para poder evoluir de acordo com novas ameaças ou requisitos, sempre proporcionando a melhor solução para as condições de privacidade especificadas. Além disso, a modularidade da estrutura é relevante também para que a proteção atingida em cada camada não interfira em outra e para que, por conseguinte, a privacidade do consumidor seja abrangente e coerente com as demandas traçadas. Cada consumidor específico

selecionará as camadas importantes para si, tendo o mapeamento modular de quais tecnologias de anonimato são aplicáveis para os itens de interesse e dimensões de privacidade objetivados. Também é relevante expor que algumas tecnologias a serem empregadas pela estrutura multicamadas não são posicionadas tipicamente como tecnologias de anonimato, como, por exemplo, os corretores de nuvem, que serão empregados para anonimizar os itens de interesse do nível de interação de contrato. No entanto, como dentro da estrutura essas tecnologias estarão cumprindo o objetivo de anonimizar o consumidor, serão consideradas aqui como tecnologias de anonimato.

Em relação ao objetivo de cada camada da estrutura, a camada de **contrato** é a direcionada justamente ao nível de interação consumidor-provedor de contrato, empregando como tecnologia de anonimato os chamados corretores de nuvem [8]. Neste caso, o anonimato aplicável é o de conexão. Para a segunda camada, a de **metadados**, que faz parte do nível de interação de troca de mensagens, o tipo de anonimato empregado também é o de conexão, sendo que as tecnologias de anonimato adequadas para esse contexto são as de anonimato de credenciais. Especialmente nesta estrutura e camada, propõe-se o emprego de assinaturas de grupo [38] ou de dinheiro eletrônico [34]. Já a terceira camada, chamada de camada de **rede**, ela se refere ao nível de interação de comunicações de rede. O tipo de anonimato empregado também é o de conexão e as tecnologias adequadas para esse contexto são as chamadas redes Mix-Net [37]. Para finalizar, a quarta camada da estrutura é a de **conteúdo**. Ela corresponde à utilização do tipo de anonimato de dados para o nível de interação consumidor-provedor de troca de mensagens. As tecnologias de anonimato aqui pertinentes são a de anonimato-k [111] e de criptossistemas homomórficos [108].

A Tabela 3.1 exibe como as camadas da estrutura proposta se relacionam com os níveis de interação consumidor-provedor e com os tipos de anonimato apresentados.

Tabela 3.1 – Relação das Camadas da Estrutura com o Consumo de Serviço SaaS

Camada da Estrutura	**Itens de Interesse**	**Dimensão da Privacidade Afetada**	**Nível de Interação Consumidor-Provedor**	**Tipo de Anonimato**
Contrato	Identificação do Consumidor	ID	Contrato Consumidor-Provedor	Conexão
	Endereço do Consumidor	Localização		
	SLA e estimativas de consumo	Comportamento		
Metadados	Credenciais AAA	ID	Troca de Mensagens	Conexão
	MEPs no tempo	Comportamento		
Conteúdo	Dados das Mensagens	ID, Localização, Comportamento, Conteúdo		Dados
Rede	Endereço IP	ID, Localização, Comportamento	Comunicações de Rede	Conexão

A seguir, neste Capítulo, as camadas da estrutura de anonimato para SaaS serão detalhadas, com as respectivas e pertinentes tecnologias de anonimato descritas. A ordem de apresentação das camadas será: **conteúdo**, **rede**, **contrato** e **metadados.**

3.1. CAMADA DE ANONIMATO DE CONTEÚDO

A camada de anonimato de conteúdo se relaciona com o anonimato de dados; e na computação em nuvem pode empregar sistemas não específicos do paradigma SaaS. O objetivo é possibilitar a utilização desses dados, enquanto a privacidade do consumidor é mantida. De acordo com a Tabela 3.1, relaciona-se com a proteção do item de interesse relativo aos dados das mensagens.

Anonimato de dados, no entanto, é de utilização extremamente complexa e particular no ambiente de computação em nuvem, pois existem inúmeros tipos de dados e estruturas de dados possíveis na lógica de negócios e no projeto de serviços SaaS. Uma análise para cada tipo de serviço a ser consumido tem que ser realizada a fim de se estabelecer a técnica de anonimato de dados apropriada. No caso específico da computação em

nuvem, duas tecnologias tem se sobressaído na questão: anonimato-k e criptossistemas homomórficos [115] [89] [80].

O anonimato-k é uma técnica voltada para a anonimização dos dados em si, enquanto os criptossistemas homomórficos são técnicas de criptografia, e são focados especificadamente na possibilidade da realização de operações matemáticas diretamente em dados cifrados. Aqui, as duas tecnologias serão acomodadas na camada de anonimato de conteúdo da estrutura multicamadas, porque terão o objetivo de manter privados os itens de interesse referente aos dados do consumidor [115].

Mesmo não sendo diretamente uma técnica de anonimato, pois seu objetivo fundamental é a confidencialidade[7], a criptografia pode ser empregada com ferramenta para manter informações anônimas. Na medida em que possibilita esconder dados que possam identificar unicamente entidades, a função de anonimização é atingida. Consequentemente, tanto o anonimato-k quanto os criptossistemas homomórficos serão acomodados aqui pela camada de anonimato de conteúdo.

3.1.1. Anonimato-k

O anonimato-k, ou *k-anonymity*, busca a privacidade de microdados. Ele trabalha na definição de como um registro pode permanecer anônimo sem perder a informação nele contida ao suprimir ou generalizar os possíveis identificadores presentes no registro [111] [117]. O objetivo é a desconexão semântica do registro com seu proprietário.

Como exemplo, pode-se imaginar uma tabela de informações médicas a ser divulgada. Nela, existem os campos de CPF, Nome, Raça, Data de Nascimento, CEP, Endereço e a Condição Médica existente. Com o objetivo de anonimizar as informações e possibilitar apenas a estatística das doenças em uma região, os campos CPF e Nome são suprimidos. Porém, na lista telefônica pública, existe o CEP, o endereço e o nome dos indivíduos. Dessa forma, pode-se descobrir a condição médica de uma pessoa específica

[7] Confidencialidade, juntamente com integridade e disponibilidade, é um dos três princípios da segurança da informação. A confidencialidade dita que para uma informação ter confidencialidade ela só deve ser acessível para aquelas entidades que têm autorização para isso.

associando as informações comuns da tabela de informações médicas e da lista telefônica.

Como caso concreto, um popular provedor de vídeos em nuvem, em 2006, publicou mais de 10 milhões de avaliações de filmes feitas por mais de 500.000 de seus consumidores. E, para resguardar a privacidade desses usuários, o provedor optou por trocar seus nomes verdadeiros por números aleatórios. Porém, pesquisadores da universidade do Texas em Austin, correlacionando essas informações publicadas com as de um popular portal específico para avaliações de filmes, conseguiram obter a identificação direta de vários dos consumidores supostamente anonimizados, mas que tinham avaliações iguais em ambas as compilações de dados [87].

O anonimato-k institui então uma fundação teórica formal para o problema da quebra de privacidade na questão de divulgação de microdados, anonimizados com supressão ou generalização de dados, estabelecendo como o anonimato pode ser garantido no nível desejado aplicando a generalização ou supressão apropriadas.

Para o consumo de serviços SaaS, o anonimato-k pode ser adequado em serviços do tipo "folha de pagamento", onde a tabela a ser processada pelo provedor pode ser enviada com os nomes e matrículas dos empregados suprimidos. Um exame criterioso da lógica envolvida deve ser realizado para verificar a possibilidade de utilização da técnica.

A propósito, ressalta-se, como exemplo, que o anonimato-k já foi especificado como solução isolada de anonimato de dados para computação em nuvem em [132].

3.1.2. Criptossistemas Homomórficos

A criptografia tem sido utilizada há tempos com o objetivo de preservar a confidencialidade da informação [78]. Contudo, existe uma limitação básica da tecnologia no que se refere ao paradigma da computação em nuvem e do papel do provedor no consumo de serviços: uma informação cifrada tem que ser decifrada para que o provedor possa processá-la.

Em 1978, Rivest, Adleman e Dertouzos, em um trabalho intitulado "*On Data Banks and Privacy Homomorphisms*" [108], definiram o que chamaram de homomorfismos de

privacidade, ou *privacy homomorfisms*, que seriam funções criptográficas especiais que teoricamente poderiam realizar determinadas operações diretamente em informações cifradas, produzindo o resultado com a cifração mantida.

Seguindo a pesquisa acima, a comunidade científica passou a buscar implementações práticas da teoria, ou seja, algoritmos capazes de realizar a chamada criptografia homomórfica.

Inicialmente, conseguiu-se obter apenas criptossistemas parcialmente homomórficos, que são capazes de realizar apenas um tipo de operação em texto cifrado, ou adição ou multiplicação. Exemplos de criptossistemas parcialmente homomórficos são os criptossistemas Okamoto-Uchiyama [98], Naccache-Stern [85] e Damgård-Jurik [43].

A partir daí, intensificou-se a procura por um sistema totalmente homomórfico, e o termo "Santo Graal" [79] da criptografia foi atribuído para a possibilidade da Criptografia Totalmente Homomórfica – (FHE – *Fully Homomorphic Encryption*).

Finalmente, em 2009, Craig Gentry em [56], propôs o primeiro criptossistema totalmente homomórfico. Todavia, a performance e a utilização prática desse e de outros criptossistemas totalmente homomórficos que surgiram desde então ainda está aberta à discussão e à validação da comunidade científica [69] [116] [71] [57].

Hoje, o crescimento da computação em nuvem tem pressionado a concretização de criptossistemas totalmente homomórficos eficientes para uso geral. Contudo, em [126], por exemplo, discute-se a possibilidade de se manter a privacidade de um consumidor de nuvem mesmo com FHE. Ademais, como pode ser visto na Tabela 3.1, aponta-se que, mesmo que se consiga uma utilização ideal de FHE, somente a camada de conteúdo da estrutura multicamadas poderá estar protegida no consumo de serviços SaaS, pois ainda que os dados em si permaneçam cifrados durante o processamento do provedor, os outros itens de interesse presentes nos níveis de interação consumidor-provedor continuarão a expor a privacidade do consumidor. É neste ponto que a estrutura proposta neste Capítulo se insere, introduzindo uma abordagem abrangente para a proteção de privacidade não só para os dados do consumidor, mas para todos seus itens de interesse privados.

Similarmente ao anonimato-k, como exemplo, a utilização de criptossistemas homomórficos também já foi proposta como técnica específica para o anonimato de dados em computação em nuvem em [120].

3.2. CAMADA DE ANONIMATO DE REDE

Como visto na Tabela 3.1, o nível de interação consumidor-provedor de comunicações de rede tem um item de interesse que impacta a privacidade do consumidor: o endereço IP. Para esse item de interesse, a proteção de privacidade é realizada fazendo-se uso de técnicas para o tipo de anonimato de conexão.

Anonimizar um endereço IP, do ponto de vista de conexão, é uma matéria explorada na comunidade científica por meio dos sistemas **Mix-Net**.

3.2.1. Mix-Nets

O design Mix-Net foi idealizado por Chaum em trabalho seminal de 1981, intitulado "*Untraceable electronic mail, return addresses, and digital pseudonyms*" [37]. Nele, Chaum propôs a desconexão do destinatário e do remetente por meio do envelopamento das mensagens em camadas de criptografia de chave-pública, e do consequente envio das mesmas através de um caminho composto de "*mixers*" (embaralhadores). Cada *mixer* tem que decifrar, reter e reordenar as mensagens antes de enviá-las.

A pesquisa sobre o assunto progrediu e duas abordagens para o problema do anonimato de rede emergiram: alta latência com anonimato robusto e baixa latência com anonimato leve [47]. Sistemas buscando um anonimato robusto, com resistência a uma maior gama de modelos de adversários, acabaram por ter que aceitar valores de latência mais altos. Exemplos de sistemas criados com essas características são o **Babel** [59], o **Mixmaster** [82] e o **Mixminion** [44].

Sistemas com requisitos menos tolerantes à latência de rede aceitaram como restrição um modelo de adversário ao anonimato mais limitado. Nesse sentido, sistemas com essa característica são resistentes a ataques de análise de tráfego, mas não conseguem manter o anonimato oferecido em caso de atacantes que possam observar a rede em ambos os extremos da conexão. Em termos práticos, essa limitação acaba por ser aceitável na

maioria dos ambientes, uma vez que um atacante de rede dificilmente poderá prever e preparar um equipamento espião tanto na origem quanto no destinatário, especialmente no paradigma da Internet atual. Exemplos de sistemas com essas características são **Hordes** [70], **Crowds** [107], **Cebolla** [27] e **Tor** [47].

Ao analisar ambos os tipos de sistemas, sob a óptica da utilização em um ambiente de computação em nuvem, um funcionamento com alta latência não é considerado adequado. De fato, os sistemas de alta latência e anonimato robusto são voltados para aplicações do tipo, por exemplo, do email, onde a demora e a variação da latência não causam impactos extremos. Para o consumo de serviços SaaS, porém, por tratar-se de um ambiente ágil, interativo e dinâmico, tempos de respostas baixos são altamente desejáveis.

Em face disso, torna-se mais adequado para a camada de anonimato de rede da estrutura um sistema de baixa latência. E nesse contexto, destaca-se o **Tor** - The Second-Generation Onion Router.

3.2.1.1. Tor - The Second-Generation Onion Router

O Tor é um sistema Mix-Net, juntamente com sua respectiva implementação, para anonimato de endereços de rede, que opera com baixa latência. O Tor, especificado em [47], é uma evolução sobre o projeto "*The Onion Routing*", cujo design é detalhado e analisado em [58] [106] [118] [119]. Ele utiliza o mesmo conceito, mas suplanta as limitações do original através da introdução de encaminhamento sigiloso, controle de congestionamento, servidores de diretório, verificação de integridade, política de saída configurável, e de um desenho para serviços de localização oculta via pontos de encontro [47].

Basicamente, no Tor, o conceito consiste de uma entidade originadora de tráfego escolhendo um circuito onde cada nó, ou roteador Tor, conhece apenas seu sucessor e seu predecessor, e mais nenhum outro nó do caminho. Cada roteador se comunica com o próximo nó empregando chaves simétricas empilhadas sobre as outras comunicações, semelhantemente a uma cebola. A Figura 3.2 mostra as etapas do funcionamento do Tor. Na etapa 1, o cliente Tor entra em contato com um servidor de diretório para obter

uma lista dos nós Tor operacionais. Na etapa 2, o cliente seleciona um caminho aleatório por nós Tor para alcançar o servidor de destino escolhido. As conexões entre os nós Tor, durante essa comunicação, são cifradas. Eventualmente, na etapa 3, ao necessitar se conectar com mais um servidor de destino, o cliente Tor utiliza outro caminho aleatório [123].

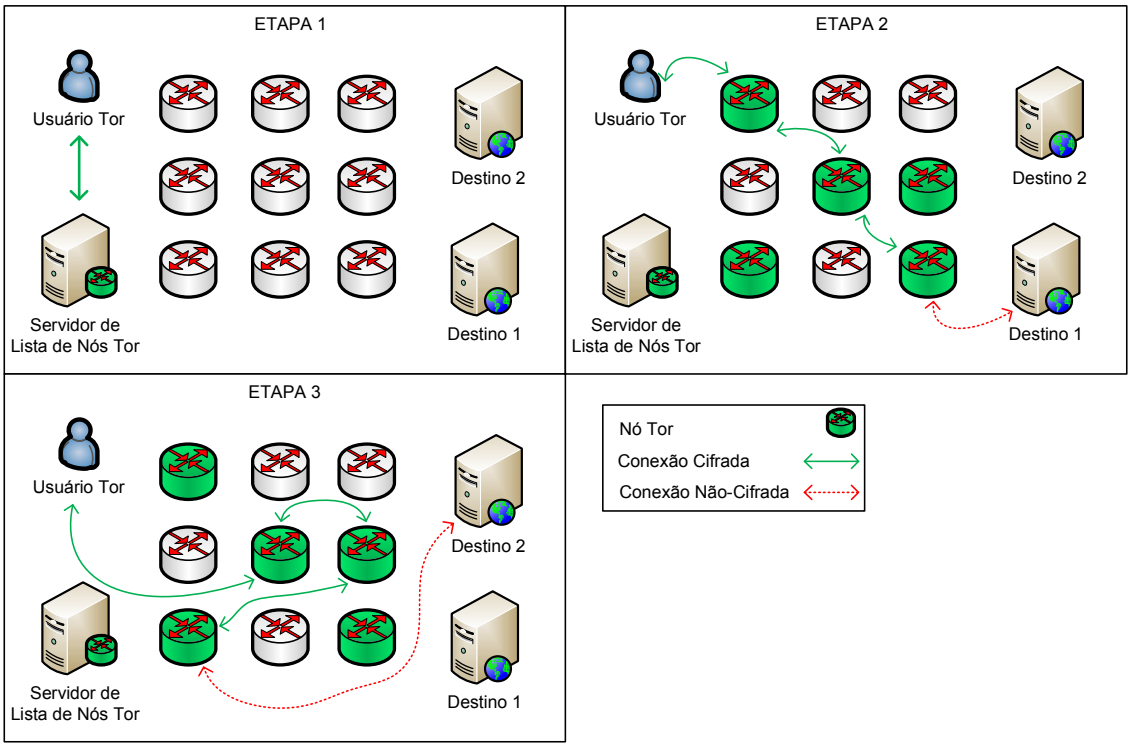

Figura 3.2 – Funcionamento do Tor [123]

Devido a grande difusão do sistema, a implementação do Tor tem experimentado extensa discussão e análise sobre sua alegada segurança. A principal preocupação é de quão resistente se mostra seu anonimato, considerando que o Tor foi projetado segundo o paradigma de baixa latência.

O modelo de ameaça do Tor estabelece que nenhum adversário tem acesso à todos os nós Tor de uma determinada comunicação. Logo, a robustez do Tor não é resistente ao chamado atacante global. O Tor pode apenas manter suas propriedades de anonimato enfrente a um ataque de rede não global.

Além disso, mesmo considerando esse modelo de ameaça limitado, detectaram-se algumas vulnerabilidades do sistema, como, por exemplo, em [84] e [14]. Em [84], Steven J. Murdoch e George Danezis demonstram que com um custo linearmente proporcional ao número de nós do Tor é possível identificar um caminho tomado por determinada comunicação, mas não sua origem. Já em [14], explora-se o projeto do Tor, que seleciona rotas com mais recursos disponíveis, para mostrar que um ataque de baixos recursos maliciosamente publicados como abundantes pode comprometer uma grande fração das requisições para criação de circuitos na rede. Porém, como o ataque em [84] não revela em si a identidade do originador e, em [14], para mitigar a vulnerabilidade exposta, uma solução baseada na verificação das alegações de recursos é proposta, uma possível adoção do Tor, no caso do consumo de serviços SaaS, não é comprometida.

Existem, também, melhorias e características recentemente propostas para o Tor. Em [112], oferecem-se ajustes para o sistema com o objetivo de melhorar sua performance e segurança. Especificamente no contexto da computação em nuvem, ressalta-se que em [83] é proposto o conceito de *dust clouds*, nós Tor temporariamente alocados como máquinas virtuais no modelo IaaS da computação em nuvem. Advoga-se que a facilidade de alocação e a efemeridade da existência de tais roteadores pode melhorar o anonimato proposto pelo Tor, especialmente considerando a facilidade e preço de alocação de recursos computacionais no paradigma da computação em nuvem. De fato, uma iniciativa conjunta em [124] da implementação Tor e da Amazon já oferece nós Tor configurados e operacionais como um serviço pago e elástico, objetivando facilitar a contribuição da comunidade, promovendo o aumento de roteadores Tor disponíveis.

Novamente, a discussão acima acerca do sistema Tor é suplementar dentro desta tese, sendo que uma eventual escolha pode ser alterada dentro da flexibilidade introduzida pela estrutura multicamadas proposta neste Capítulo. Deve-se atentar para que um sistema substituto salvaguarde os itens de interesse presentes na camada de anonimato de rede (Tabela 3.1). No entanto, o Tor tem se mostrado eficiente, com eficácia prática em anonimizar endereços IP.

3.3. CAMADA DE ANONIMATO DE CONTRATO

A Tabela 3.1 evidencia a necessidade de proteger a privacidade do consumidor relativa aos itens de interesse presentes em contratos consumidor-provedor, especificamente nas dimensões de ID, localização e comportamento.

Contratos consumidor-provedor são documentos que se relacionam diretamente com o modelo de negócios da computação em nuvem. Esses documentos atendem à necessidade de se vincular legalmente e economicamente as partes envolvidas. O contrato, contudo, representa uma significativa ligação entre consumidor e provedor e conduz à exposição da privacidade do consumidor.

Nesse sentido, contratos precisam ser anonimizados para a proteção da privacidade do consumidor, mas de maneira que os acordos estabelecidos pelo contrato possam ser preservados tanto para o consumidor quanto para o provedor. Ao consumidor devem ser garantidas as corretas condições de serviço, como SLA, e pelo outro lado, o provedor deve ser legalmente responsável por prover o serviço de acordo com as especificações técnicas, mas lhe deve ser garantida a devida compensação financeira.

Para atender à necessidade de privacidade por meio do anonimato e ao mesmo tempo às condições legais contratuais esperadas, esta estrutura multicamadas propõe a substituição do contrato direto entre consumidor e provedor por **contratos indiretos**. Destarte, para intermediar o estabelecimento dos contratos, introduz-se a entidade denominada corretor, ou *broker*. Corretores não são novidade no paradigma da computação em nuvem, e estão se tornando comuns com a proliferação dos papéis intermediários, especialmente aqueles relacionados com corretagem de segurança [104].

3.3.1. Corretores de Nuvem

Com a evolução da computação em nuvem, a integração de serviços de nuvem tornou-se demasiadamente complexa para que somente os consumidores a gerenciassem. Assim, o papel dos corretores de serviços de nuvem ganhou relevância, introduzindo um cenário onde um consumidor de nuvem pode requisitar serviços de um corretor, ao invés de contratá-los diretamente o provedor. O corretor especializa-se em gerenciar o

uso, performance, e entrega dos serviços, além de negociar relações entre provedores e consumidores.

Essencialmente, um corretor de serviços pode ser definido dentro de três categorias:

- **Intermediação de serviços**: Neste caso, um corretor de nuvem melhora um determinado serviço por meio do aprimoramento de determinada capacidade e promovendo valor agregado aos consumidores. Como exemplos, o aprimoramento pode ser na forma do gerenciamento do acesso aos serviços de nuvem, do gerenciamento de identidades, da supervisão de performance, e da melhoria de segurança.

- **Agregação de serviços**: Um corretor de nuvem pode focar na combinação e integração de múltiplos serviços de nuvem em um ou mais novos serviços. O corretor, então, promove a integração dos dados e garante a movimentação segura da informação entre o consumidor e os múltiplos provedores de nuvem.

- **Arbitragem de serviços**: A arbitragem de serviços é semelhante à agregação de serviços, com exceção do fato de que os serviços sendo agregados não são fixos. Arbitragem de serviços significa que um corretor tem a flexibilidade de selecionar serviços de múltiplas agências. Um corretor pode, por exemplo, utilizar um serviço de gradação de serviços a fim de eleger a agência com a melhor pontuação.

3.3.1.1. Corretor de Contratos SaaS - TPB

Nesta estrutura multicamadas, especificamente nesta camada de anonimato de contrato, utilizar-se-á, por conseguinte, um tipo de corretor de nuvem, da categoria de intermediação de serviços, para o estabelecimento dos contratos indiretos. Esse corretor fará o papel de uma terceira entidade no consumo de serviços, e será responsável por eliminar a necessidade do consumidor expor sua privacidade ao contratar e consumir serviços. Adota-se, a partir daqui, o termo *Third-Party Broker*, ou simplesmente **TPB**, para caracterização desse tipo de corretor de nuvem.

No contexto do consumo de serviços SaaS, o TPB, como um intermediador de serviços e, nesse caso, de contratos, realizará as seguintes tarefas:

- **Credenciamento dos Consumidores**: Para introduzir os contratos indiretos, o TPB necessita regulamentar o acesso do consumidor ao provedor, garantindo a execução correta dos processos de tarifação, faturamento e pagamento, enquanto observando os termos contratuais. Para isso, o TPB emitirá e controlará o emprego das credenciais a serem usadas nas operações AAA (identificação dos consumidores perante o provedor), e as utilizará para permitir a tarifação, faturamento e pagamento. As credenciais serão incluídas nas requisições de serviço de cada consumidor, quando contando diretamente o provedor. O modo de emprego dessas credenciais é importante, pois durante o consumo de serviço, a partir das credenciais, o provedor não deverá ser capaz de obter as informações privadas presentes nos itens de interesse da camada de anonimato de contrato (ID, endereço e SLA). E, como o TPB está exercendo a função da intermediação dos contratos, essas credenciais deverão manter a possibilidade da correta tarifação, faturamento e pagamento.

- **Tarifação e Faturamento**: O TPB atuará como intermediário na tarifação e faturamento. Com o uso dos contratos indiretos, o provedor não será capaz de conectar um consumo específico com seu respectivo contrato, onde as políticas de precificação e demais termos, como SLA, estão registrados. Essa função recairá sobre o TPB. O TPB deverá, então, individualizar o acesso de cada consumidor via seu respectivo contrato, tarifando cada acesso e compilando uma fatura que contemple além do preço atômico de cada consumo como também os termos de contrato. Devido a esse contrato, notadamente na parte de SLA, a fatura de cada consumidor específico poderá sofrer ajuste. Além disso, em alguns casos, o faturamento não ocorrerá *a posteriori*, como descrito acima, e, portanto, no evento da necessidade de faturamento imediato, com pagamento a ser realizado no momento do consumo, não existirá a necessidade da compilação de registros. As credenciais a serem emitidas nesse caso, entretanto, deverão suportar esse ambiente, caracterizando mais uma função do TPB.

- **Pagamento:** O TPB pode, também, atuar como corretor financeiro, intermediando as próprias transações monetárias entre consumidor e provedor. Essa tarefa consiste no recebimento do pagamento dos consumidores e na transferência desse para os provedores. Do ponto de vista do consumidor SaaS, o modelo mais comum em voga é o do pré-pagamento, mas modelos de pós-pagamento podem ser adotados, contanto que o sistema de credenciais em funcionamento permita esse comportamento, em concordância com a tarefa de tarifação e faturamento descrita acima.

As tarefas descritas relacionam-se com a introdução dos contratos indiretos. Elas os possibilitam e derivam-se deles. Deve-se notar que a presença do TPB promove a separação entre a informação presente no contrato consumidor-provedor (itens de interesse) e a própria troca de mensagens consistindo no consumo de serviços. O TPB tem acesso à informação de contrato, mas não observa as mensagens sendo trocadas durante o consumo propriamente dito. Enquanto isso, o provedor perde o acesso aos itens de interesse presentes no contrato, mas retém a capacidade de prover o serviço e ser pago por isso.

3.4. CAMADA DE ANONIMATO DE METADADOS

A camada de metadados foca nos itens de interesse das credenciais AAA e das MEPs no tempo, do nível de interação de troca de mensagens. Os dados das mensagens, como item de interesse, são abordados pela camada de anonimato de conteúdo.

Essa camada, juntamente com a de contrato, é voltada para características específicas do ambiente de computação em nuvem. Aplica-se o anonimato de conexão, que, por sua vez, se relaciona diretamente com os metadados de mensagem empregados na identificação dos consumidores de serviços SaaS.

Anonimizar metadados, que são basicamente as informações necessárias para a identificação do consumidor perante o provedor no momento do consumo, é basicamente anonimizar credenciais, permitindo a manutenção do processo de AAA, enquanto mantendo o requisitante anônimo.

Dessa forma, a camada de anonimato de metadados concentra-se no emprego de **credenciais anônimas**. As credenciais anônimas serão, neste contexto, emitidas pelo TPB, como visto na Seção anterior, na tarefa de credenciamento; e constituirão a tecnologia fundamental da camada de metadados. Portanto, apesar da modularidade da estrutura multicamadas, tanto a camada de contrato como a de metadados trabalham em conjunto, conciliando o emprego dos contratos indiretos com a utilização de credenciais anônimas.

Sistemas de autenticação dentro do paradigma de credenciais anônimas é um assunto que tem sido explorado consideravelmente pela comunidade científica. O conceito foi introduzido por Chaum em [36], e, a partir de então, vários sistemas foram propostos.

Com a diversidade de modelos que surgiram, múltiplas abordagens para o anonimato foram propostas. Todavia, aqui, levando-se em conta a necessidade da camada de metadados, buscaram-se sistemas que possibilitassem as seguintes premissas, sendo estas comuns à maioria dos esquemas de credenciais anônimas [5] [73]:

- **Autenticação Anônima**: O sistema de credenciais deverá permitir a autenticação, ou seja, uma garantia ao verificador de que o requisitante pertence a um grupo esperado, porém não deve unicamente identificar a entidade requisitante.

- **Impossibilidade de Correlacionamento de Autenticação**: O sistema de credenciais deverá manter o anonimato da entidade requisitante, mesmo no evento de transações sucessivas.

Dentro de tais requisitos, existem vários tipos de credenciais anônimas que podem ser utilizadas. As características mais relevantes a serem consideradas na escolha são:

- **Validação de Credenciais *online/off-line***: Refere-se ao tipo de envolvimento do TPB no momento da validação das credenciais do consumidor.

- **Rastreabilidade**: Refere-se à possibilidade de revelar a identidade anônima do consumidor em casos especiais, como, por exemplo, devido a operações ilegais.

Considerando as características acima relacionadas, o contexto da computação em nuvem e a operação conjunta com a camada de anonimato de contrato, verificou-se a adequação de dois tipos de sistemas de credenciais anônimas [30] [72]: **assinaturas de grupo** e **dinheiro eletrônico**. Esses sistemas podem oferecer a privacidade buscada, e, ao mesmo tempo, podem proporcionar opções diferentes de validação de credenciais e rastreabilidade, conforme demanda do ambiente. Obviamente, esses tipos de sistemas são apenas escolhas convenientes para a anonimização dos itens de interesse desta camada, e, como a estrutura multicamadas foi projetada para a flexibilidade, pode-se alterar a tecnologia selecionada por outra mais adaptada à situação. A condição, no entanto, que não pode ser eliminada é a anonimização dos itens de interesse afetando a privacidade do consumidor na camada de metadados (Tabela 3.1).

A seguir, detalhar-se-á as tecnologias de assinaturas de grupo e de dinheiro eletrônico, juntamente com sistemas específicos de aplicação dessas.

3.4.1. Assinaturas de Grupo

Assinaturas de grupo (*group signatures*), originalmente introduzidas por Chaum e van Heyst [38], é um sistema de assinaturas, onde qualquer membro de um determinado grupo pode assinar mensagens, mas a assinatura resultante mantém a identidade do signatário em segredo. Um verificador pode averiguar se a assinatura foi gerada por um membro legítimo do grupo, mas a assinatura não revela informação sobre qual dos membros realmente assinou a mensagem. Membros do grupo desfrutam de anonimato, significando que suas identidades são de recuperação inviável dada apenas a assinatura criada. Essa premissa possibilita uma série de aplicações. De fato, como exemplos, podem-se citar duas em uso atualmente: o *Direct Anonymous Attestation* – DAA [26] – e o *Vehicle Safety Communications* – VSC [63]. O DAA permite que um usuário de uma plataforma executando em seu *laptop* possa se autenticar para o servidor como sendo legítimo, sem revelar exatamente sua identidade, e o VSC consiste em um sistema onde carros são embutidos com transmissores de curto alcance, que possibilitam a emissão de um alerta em caso de emergência a todos os carros dentro de uma área restrita, informando a iminência de uma freada abrupta ou de uma manobra similar [76].

Em sistemas de assinatura de grupo, existe a figura de um terceiro que pode rastrear a assinatura, ou desfazer seu anonimato, utilizando um procedimento especial. A rastreabilidade é útil para inibir comportamentos inadequados. Caso contrário, membros desonestos ou maliciosos do grupo podem capitalizar seu anonimato e fraudar com impunidade.

Nos esquemas de assinaturas de grupo típicos, o terceiro é o mestre do grupo, que emite as **chaves secretas** para os membros, utilizadas para assinar, e publica a **chave pública** utilizada na operação verificação. O mestre do grupo também assume a responsabilidade de executar o rastreamento.

Anonimato e rastreabilidade são propriedades básicas de qualquer regime de assinaturas de grupo. Todavia, existem muitas variações e propriedades adicionais, como *selfless-anonymity*[8] [23] e *non-frameability*[9] [17].

Também, em determinados sistemas, um gerente de grupo pode ser empregado no lugar do mestre de grupo. Um gerente de grupo diferencia-se do mestre de grupo pelo fato de que ele irá interagir com o membro quando da emissão de sua chave secreta, mas não irá a aprender. Neste caso, somente o membro do grupo terá acesso à sua própria chave secreta.

Já a propriedade de anonimato contratual [114] estabelece que membros submetidos a uma política de utilização específica e pré-determinada podem usufruir de anonimato incondicional, ou seja, não estão sujeitos ao rastreamento de suas identidades, ao passo que caso a desobedeçam, expõem-se à autoridade rastreadora.

Em relação ao estado dos grupos, os membros podem ser estáticos, definidos no início, ou dinâmicos, com membros podendo ser adicionados depois de o grupo inicial ter sido criado.

[8] *Selfless-anonymity* é a propriedade de um sistema de assinatura de grupos que permite que um membro possa verificar se uma assinatura foi gerada por ele, mas em caso negativo não descobre mais informações acerca da mensagem.
[9] *Non-frameability* é a propriedade de um sistema de assinatura de grupos que estabelece que um adversário é incapaz de criar prova aceitável de que um membro do grupo produziu uma certa assinatura válida, se o mesmo não efetivamente o fez.

E, finalmente, alguns sistemas suportam a revogação de anonimato, onde o mestre ou o gerente do grupo pode derrogar a possibilidade de anonimato de um membro, implicando que o usuário não poderá continuar assinando suas mensagens anonimamente.

De destacar, Bellare, Micciancio e Warinschi, em [15], propuseram uma terminologia geral relacionada e utilizada amplamente nos designs do estado-da-arte da tecnologia de assinaturas de grupo. Verdadeiramente, até hoje não existe um sistema definitivo e universal de assinaturas de grupo. Os projetos mais notórios são baseados em diferentes propriedades, premissas e objetivos de desempenho.

A título de apoio à seleção de um sistema de assinaturas de grupo adequado para a utilização na estrutura multicamadas, apresentam-se, a seguir, as Tabelas 3.3 e 3.4. Respectivamente, elas comparam as propriedades relevantes para o modelo SaaS oferecidas por cada sistema e a complexidade dos parâmetros necessários para o seu funcionamento. Os sistemas comparados são os mais populares [74] e serão referenciados como descrito na Tabela 3.2:

Tabela 3.2 – Descrição dos Sistemas de Assinaturas de Grupo

Sistema	**Nome**	**Definido em**
ACJT	Ateniese-Camenisch-Joye-Tsudik	[7]
TX	Tsudik-Xu	[125]
CG	Camenisch-Groth	[28]
KY	Kiayias-Yung	[68]
AM	Ateniese-de Medeiros	[6]
FY	Furukawa-Yonezawa	[54]
BBS	Boneh-Boyen-Schacham	[21]
CL	Camenisch-Lysyanskaya	[30]
BCNSW	Bichsel-Camenisch-Neven-Smart-Warinschi	[19]
BS	Boneh-Schacham	[23]
NF	Nakanishi-Funabiki	[86]
BCNSW-VLR	Bichsel-Camenisch-Neven-Smart-Warinschi, variante VLR	[19]

Tabela 3.3 – Comparação das Propriedades dos Sistemas de Assinaturas de Grupo

Sistema	Estático	Dinâmico	Revogação
ACJT		X	X*
TX		X	X
CG	X	X*	X*
KY		X	
AM		X	
FY		X	
BBS	X		X*
CL		X	
BCNSW		X	
BS	X		VLR
NF	X		TVLR
BCNSW-VLR			VLR

X – a propriedade é fornecida; X* – a propriedade é fornecida como uma extensão separada.

Tabela 3.4 – Comparação das Complexidades dos Parâmetros dos Sistemas de Assinaturas de Grupo

Sistema	Chave Pública do Grupo	Chave Secreta de cada Membro	Assinatura	Custo de Revogação
ACJT	10	3	≈16	m
TX	8	4	≈13	m
CG	9	4	≈9	-
KY	7	3	≈12	-
AM	9	4	O(k)[10]	-
FY	10	3	O(k)	-
BBS	6	2	9	-
CL	8	4	11	-
BCNSW	4	4	5	-
BS	3	2	7	r
NF	3 + t	2	12	r
BCNSW-VLR	4	4	5	r

Todas as complexidades são dadas em elementos de grupo; k – tamanho da saída de uma função *hash*[11]. m – número de membros do grupo; r – número de membros revogados; t – número de intervalos de tempo.

[10] Emprega-se aqui a notação chamada "*Big O*", que define como a complexidade ou tamanho de uma grandeza varia, ou seja, quais variáveis e em que ordem estas impactam a função em questão. Utiliza-se colocando as variáveis e suas ordens dentro de parênteses precedidos do O maiúsculo [60].

[11] O *hash* é uma operação executada em uma determinada quantidade de informação, podendo ser de tamanho variável, que gera uma quantidade fixa de dados, não reversível à informação original, mas que varia consideravelmente com a menor alteração dos dados de entrada. É utilizada para estabelecer a integridade de dados originais.

Em relação à Tabela 3.3, verifica-se que as propriedades consideradas foram o tipo de grupos empregados, se eles são estáticos ou dinâmicos, e se o grupo suporta revogação do anonimato. Especificamente na questão de revogação, introduz-se a possibilidade do sistema ter revogação do tipo VLR (*Verifier-Local Revocation*), que determina que o sistema, para revogar o anonimato de um membro, só precisa repassar essa informação para as entidades que verificam assinaturas, e não para todos os membros do grupo. Adicionalmente, o sistema NF amplia o conceito VLR para incorporar a revogação temporária dentro do VLR, ou seja, as mensagens de revogação são enviadas somente para os verificadores de assinatura e tais revogações tem validade limitada, daí o nome da revogação ser do tipo TVLR (*Time Verifier-Local Revocation*) [74].

Para a Tabela 3.4, o objetivo é comparar a complexidade de utilização dos parâmetros necessários dentro de um sistema de assinaturas de grupo. Os parâmetros considerados foram a chave pública, a chave secreta, a assinatura e o custo de revogação. O custo de revogação corresponde ao espaço necessário para armazenar a lista de membros revogados ou o custo para a atualização da informação para as entidades envolvidas, dependendo da forma de operação do sistema. Para os demais parâmetros, o valor apresentado representa o tamanho do elemento. Esse tamanho tipicamente é influenciado pelas características criptográficas de cada sistema, que variam consideravelmente. Todavia, para facilitar a comparação, o tamanho será expresso pela quantidade de elementos de grupo em diferentes grupos cíclicos[12] necessários para representar o parâmetro em questão. Essa decisão permite a medição comparativa de tamanho utilizando um atributo comum dos sistemas [74].

A seleção de um sistema de assinaturas de grupo como o adequado para a utilização dentro da estrutura multicamadas deve observar a conveniência de cada propriedade

[12] Em álgebra, um grupo cíclico é um grupo que é gerado por um elemento único, de forma que cada elemento do grupo pode ser escrito como uma potência de um determinado elemento *g* em notação multiplicativa, ou como um múltiplo de *g* em notação aditiva. Este elemento *g* é chamado do "gerador" do grupo. Qualquer grupo cíclico infinito é isomorfo a **Z**, que são os inteiros com adição como a operação de grupo. Qualquer grupo cíclico finito de ordem n é isomorfo a **Z** / n**Z**, que são os inteiros módulo n com adição como a operação de grupo [61].

para o paradigma SaaS da computação em nuvem, juntamente com a complexidade envolvida na utilização do mesmo.

Deste modo, inicialmente em relação ao tipo de grupos, sendo estáticos ou dinâmicos, observa-se que não há restrição do ambiente SaaS, de maneira que pode-se operar em um modelo estático ou dinâmico. A utilização de grupos estáticos é possível, e talvez até preferencial, pois se pode realizar com essa propriedade uma previsão de consumidores admissíveis, permitindo ao provedor se preparar para a quantidade de solicitações a serem recebidas, evitando que eventuais SLAs estabelecidos pelos contratos indiretos sejam infringidos. Os sistemas com grupos estáticos são: CG, BBS, BS e NF.

A julgar pela revogação, a propriedade é desejável, pois o TPB, por exemplo, pode identificar que um consumidor esteja infringindo o contrato estabelecido e, então, proceder à interrupção de consumo de apenas um consumidor, caso esta sanção seja cabível. E, dentro dos tipos de revogação possíveis, a VLR é bastante adequada ao ambiente SaaS, uma vez que o comum será a existência de múltiplos consumidores e poucos provedores. Apenas os provedores tem necessidade de verificar assinaturas e, por conseguinte, torna-se indesejável enviar informações de revogações para todos os consumidores. Dos sistemas que empregam grupos estáticos, os que possuem revogação do tipo VLR são o BS e o NF.

Para concluir a seleção, a complexidade dos parâmetros é analisada. Neste quesito, o sistema BS é não só aquele que possui a menor complexidade de parâmetros dos sistemas que possuem grupos estáticos e revogação do tipo VLR, como também de todos os sistemas apresentados. Consequentemente, torna-se BS a opção para a estrutura multicamadas dentro da tecnologia de assinaturas de grupo, com vistas ao esquema prático a ser detalhado no próximo Capítulo. Tal escolha pode, porém, ser alterada e outros sistemas de assinaturas de grupo podem ser utilizados para o papel da anonimização de credenciais.

3.4.1.1. Assinaturas de Grupo com Revogação do Tipo Verificador-Local – Design de Boneh e Shacham

O modelo BS, descrito em [23], utiliza assinaturas de grupo curtas, que são tão curtas quanto assinaturas RSA padrões, e com segurança comparável. Esse modelo suporta e tem seu nome baseado na revogação do tipo verificador-local (VLR – *Verifier-Local Revocation*), onde, como detalhado acima, mensagens de revogação de anonimato são enviadas apenas para verificadores de assinatura (provedores, no caso do consumo de serviços SaaS), ao invés de ambos signatários e verificadores.

A segurança do sistema de assinaturas de grupo, no modelo de oráculo randômico [16], é baseada nas hipóteses Diffie-Hellman Forte (*Strong Diffie-Hellman – SDH – assumption*) [20] e Decisão Linear (*Decision Linear assumption*) em grupos bilineares [21]. A especificação do sistema em [23] apresenta as provas de segurança e o detalhamento matemático.

Formalmente, o BS, por ser um sistema de grupos estáticos e com revogação no modelo VLR, consiste de três algoritmos explícitos, **Keygen**, **Sign**, **Verify**, e um algoritmo implícito, o **Trace**. São eles:

- **Keygen**(***n***): Esse algoritmo aleatorizado tem como entrada um parâmetro ***n***, que é o número de membros do grupo. Ele tem como saída a chave pública do grupo ***gpk***, um vetor de ***n*** elementos de chaves de usuário ***gsk*** = (***gsk***[1], ***gsk***[2],...,***gsk***[***n***]), e um vetor de ***n*** elementos de provas de revogação ***grt***, indexado da mesma forma.

- **Sign**(***gpk***, ***gsk***[***i***], ***M***): O algoritmo de assinatura aleatorizado tem como entrada a chave de grupo ***gpk***, uma chave privada ***gsk***[***i***], e uma mensagem ***M***, e tem como saída uma assinatura σ.

- **Verify**(***gpk***, ***RL***, σ, ***M***): O algoritmo de verificação tem como entrada a chave pública do grupo ***gpk***, um conjunto de provas de revogação ***RL***, cujos elementos formam um subconjunto de elementos de ***grt***, uma suposta assinatura σ, aplicada sobre uma mensagem ***M***. A saída é ***válida*** ou ***inválida***. Uma resposta ***inválida***

pode significar que a assinatura σ ou não é válida, ou o usuário que a gerou foi revogado.

- **Algoritmo implícito Trace:** Qualquer sistema VLR possui um algoritmo implícito de rastreamento associado que, utilizando uma chave secreta de rastreamento, pode estabelecer pelo menos um membro do grupo que gerou uma assinatura. O vetor de provas de revogação, ***grt***, funciona como essa chave secreta de rastreamento. Dados o par mensagem e assinatura (M,σ), uma entidade possuindo todas as provas de revogação ***grt*** pode determinar qual membro gerou a assinatura seguindo o seguinte algoritmo:

    1. Para cada $i = 1,...,n$ executar o algoritmo de verificação em M,σ com a lista de revogação $RL = \{grt[i]\}$.

    2. Registrar como saída o índice do primeiro usuário para o qual o algoritmo de verificação resultar em ***inválida***. Registrar como saída ***falha***, se as verificações forem do tipo ***válida*** para todos os n usuários.

Efetivamente, então, como o modelo VLR precisa o parâmetro ***RL*** apenas para os verificadores de assinaturas, ele é recomendado para ambientes onde existem poucos verificadores e muitos signatários. O consumo de serviços SaaS se encaixa nesse perfil, onde os signatários seriam os consumidores e os verificadores seriam os provedores. O TPB seria o mestre do grupo, única entidade capaz de realizar a operação de rastreamento (**Trace**), uma vez que possui todas as provas de revogação ***grt*** (geradas pelo algoritmo **Keygen**).

3.4.2. Dinheiro Eletrônico

O dinheiro eletrônico (***e-cash***) é basicamente um sistema de credenciais anônimas de uso único [29], para evitar o gasto múltiplo de uma mesma representação monetária; e com propriedades semelhantes às do dinheiro real [97]. O dinheiro eletrônico tem características de privacidade adequadas à necessidade de se proteger os itens de interesse da camada de metadados (credenciais AAA e MEPs no tempo), se encaixando

modularmente na estrutura multicamadas. O objetivo é substituir o credenciamento necessário para o consumo de serviços SaaS, presente no nível de interação de troca de mensagens, pelo pagamento anônimo imediato, sem a necessidade da contabilização, tarifação e pagamento posterior.

Nesse sentido, a utilização de dinheiro eletrônico na computação em nuvem é mais adequada para cenários específicos, onde microtransações são preferidas ao invés de serviços de assinatura, contas com validade maior ou pós-pagamento. No segundo esquema a ser apresentado nos próximos Capítulos, a versão de consumo não rastreável é concebida com a utilização da tecnologia de dinheiro eletrônico como credencial anônima.

O dinheiro eletrônico foi primeiramente proposto por D. Chaum em 1982 [34], e desde a publicação desse trabalho seminal, diversos sistemas foram propostos. Os sistemas *e-cash* combinam propriedades específicas e compilam, dentre as mais comuns, aquelas mais relevantes para a motivação do modelo. O conjunto de propriedades gerais dos sistemas de dinheiro eletrônico é apresentado abaixo [97] [29], porém nem todas as propriedades estão presentes em todos os modelos, com exceção da não rastreabilidade, que é comum para todos:

- **Não rastreabilidade**: Um pagamento não pode ser rastreado ao seu pagador.
- **Independência do meio físico**: O dinheiro eletrônico é dado abstrato e pode ser transferido pela rede.
- **Transferabilidade**: Dinheiro eletrônico pode ser transferido para outros usuários.
- **Divisibilidade**: Uma vez emitido com valor de face, o dinheiro eletrônico pode ser dividido em partes menores, mas que somadas tem o mesmo valor inicial.
- **Exculpabilidade**: Uma entidade que tentar gastar o mesmo dinheiro eletrônico mais de uma vez pode ser identificada. Aqui, vale a diferenciação da

rastreabilidade, visto vez que a exposição da identidade depende exclusivamente de como a entidade se comporta, e não da decisão de um terceiro.

Um sistema básico de dinheiro eletrônico tem três participantes:

- **Pagador:** Uma entidade recebendo determinado serviço em troca de pagamento monetário.
- **Recebedor:** A entidade que recebe compensação financeira por serviço prestado.
- **Banco**: Uma instituição eletrônica que provê a infraestrutura financeira requerida para as transações de pagamento entre Pagador e Recebedor. Embora a infraestrutura completa possa incluir uma rede de instituições financeiras, corretores, companhias de cartão de crédito e bancos, conceitualmente o Banco é a entidade responsável por emitir as notas ou moedas eletrônicas reconhecíveis por ambos Pagador e Recebedor. Ou seja, tanto Pagador quanto Recebedor terão contas abertas com esse Banco, permitindo as operações de "saque" e "depósito", respectivamente.

Além disso, um sistema de *e-cash* deve suportar três protocolos básicos:

- **Saque**: É a operação na qual o Pagador adquire dinheiro eletrônico do banco.
- **Pagamento**: É a operação que corresponde à transferência de *e-cash* de Pagador para Recebedor.
- **Depósito**: Corresponde à operação na qual o Recebedor resgata o dinheiro eletrônico recebido em troca de pagamento monetário.

A ideia principal do dinheiro eletrônico é de que os protocolos de Saque e Pagamento são projetados de maneira que seja impossível identificar quando uma moeda ou nota particular tenha sido gasta. Os protocolos de Saque e Pagamento não revelam informações ao Banco sobre como as transações se procederam.

Os modelos de dinheiro eletrônico ainda podem ser classificados em *online* ou *off-line*. A definição dependerá de como o Banco está envolvido na operação de Pagamento. Caso seja necessária a interação do Banco no momento do pagamento, constitui-se o design *online*. Do contrário, se a participação ativa do Banco no instante da compra não é imperiosa, denomina-se o sistema como *off-line*.

Como dinheiro eletrônico é intrinsicamente composto de dados binários, e é computacionalmente trivial a duplicação de bits, um sistema de *e-cash* requer um mecanismo para prevenir que um Pagador gaste o mesmo dinheiro mais de uma vez. A abordagem mais utilizada para combater os múltiplos gastos consiste na manutenção pelo Banco de uma base de dados do dinheiro eletrônico já utilizado. Essa estrutura, portanto, pode ser empregada para rejeitar transações *online* envolvendo moedas ou notas já utilizadas. Para o design *off-line*, ao Recebedor é garantido que o pagamento será honrado pelo Banco, ou que um Pagador malicioso, que realize o gasto duplo de dinheiro eletrônico, será identificado, permitindo a adoção de medidas compensatórias e/ou punitivas. Se o sistema implementar a propriedade de transferabilidade do *e-cash*, a detecção de múltiplos gastos pode ser postergada, introduzindo desafios adicionais.

Especificamente para o caso do consumo anônimo de serviços SaaS, na versão não rastreável do esquema prático a ser proposto nos Capítulos seguintes, selecionar-se-á um sistema apropriado para as características desse ambiente.

Uma propriedade chave nessa seleção é justamente o tipo de presença necessária do Banco: *online* ou *off-line*. Os sistemas *online*, como [33] [36] [35], não seriam desejáveis no modelo SaaS, pois a necessidade da presença do Banco durante todos os consumos de serviços introduziria complexidade no protocolo e atraso na efetuação das transações, incompatíveis com o consumo de serviços de nuvem. Desse modo, pelo menos para escolha do sistema a ser utilizado nesta estrutura multicamadas e, consequentemente, no esquema prático não rastreável a ser proposto, a propriedade *off-line* será um requisito.

Outra propriedade relevante na escolha é a divisibilidade. Representações de dinheiro eletrônico com valores diferentes, contudo, introduzem transações que possam incluir o

"troco", em ocasiões onde o preço a ser pago é menor do que a representação empregada no pagamento. À primeira análise, a possibilidade de emissão de troco seria interessante, pois acomodaria transações mais flexíveis, com diversas opções de pagamento. Porém, em um consumo de serviço atômico, funcionando com requisição e reposta simples, o procedimento de analisar o preço do serviço e, eventualmente, enviar uma compensação respectiva, caso o Pagador tenha remetido dinheiro em excesso, também tornaria o sistema de utilização complexa dentro do paradigma do consumo de serviços SaaS. O objetivo será ter o valor de cada representação único, coincidindo com o valor a ser pago no momento do consumo. Assim, sistemas de *e-cash* com a propriedade de divisibilidade, mesmo sendo *off-line*, como [97] [96] [102], não serão considerados.

Restringindo-se, então, a seleção para sistemas *off-line* e não divisíveis (ou que não impliquem em troco), resta a verificação da complexidade das representações do dinheiro eletrônico, neste caso moedas. Buscar-se-á sistemas que minimizem esse parâmetro, pois ele terá que ser enviado juntamente com requisição de consumo e será efetivamente a credencial de autorização anônima. Em [10], promove-se uma comparação de sistemas *e-cash* provadamente eficientes. A Tabela 3.5 abaixo resume a comparação, excluindo os sistemas *online* e os que utilizam "troco"; e nela, além das complexidades das representações das moedas, são também mostradas as eficiências dos protocolos de Saque e de Pagamento.

Tabela 3.5 – Comparação de Sistemas *E-cash*

Sistema	**Brands [25]**		**Brands Modificado [11]**		**Abe [3]**		**E-cash Compacto [29]**			
							RSA		**Pareamentos**	
Eficiência	B/R[1]	P[2]	B/R	P	B/R	P	B/R	P	B/R	P
Saque[3]	2	13	4	19	6	12	10	8	15	14p
Eficiência Pagamento	7	0	9	0	11	1	NC[4]		5+6p	9+6p
Tamanho da Moeda[5]	6		9		9		≈0		≈0	

[1]Banco/Recebedor; [2]Pagador; [3]Em número de exponenciações; [4]Não Comparável; [5]Em número de elementos; p – pareamentos.

Verifica-se a adequação eficiente e conveniente do modelo de *e-cash* proposto por J. Camenisch, S. Hohenberger e A. Lysyanskaya, denominado ***E-cash* Compacto** [29]. A escolha beneficia-se das propriedades do sistema, alinhadas com um consumo não rastreável de serviços em nuvem SaaS. Apesar de ter protocolos de Saque e Pagamento um pouco menos eficientes, empregando mais exponenciações e pareamentos, o tamanho de suas moedas é notadamente menor, facilitando sobremaneira sua eventual inclusão nas mensagens de consumo. Entretanto, a flexibilidade da estrutura multicamadas pode acomodar uma seleção diferente.

3.4.2.1. *E-cash* Compacto

O design do dinheiro eletrônico compacto consiste de um sistema *e-cash off-line* de micropagamentos, não divisíveis e não transferíveis, onde o gasto múltiplo é prevenido com a possibilidade de identificação de um usuário mal intencionado (exculpabilidade), baseando-se em números seriais. A característica significativa desse design é que, em caso de gastos múltiplos, o Banco poderá computar também os números seriais de todo o dinheiro eletrônico pertencente ao usuário, juntamente com provas de propriedade. O modelo também é eficiente tanto na complexidade computacional, quanto no armazenamento requerido para o *e-cash* emitido. Não existe entidade confiável no sistema e todas as premissas de segurança são computacionais, e não baseadas em hipóteses de confiança.

O consumidor de serviços SaaS assumirá doravante o papel de Pagador, enquanto o provedor será o Recebedor, e o TPB o Banco.

O modelo de *E-cash* Compacto em [29] é composto dos protocolos descritos abaixo, e o leitor pode-se referir ao trabalho original para as provas de segurança e detalhamento matemático:

- **BKeygen(1^k, *params*)**: É o algoritmo para a geração de chaves para o Banco ***B***. Recebe como entrada um parâmetro de segurança 1^k e outros parâmetros comuns, conhecidos por ambos o Banco e o Pagador (***params***), e tem como

saída o par de chaves do Banco (pk_B, sk_B). Após sua emissão, sk_B conterá ***params*** e estes parâmetros não precisarão mais ser informados.

- **UKeygen(**1^k**, *params*)**: É o algoritmo para geração de chaves para o usuário ***U***, que tem como saída (pk_U, sk_U). Tanto consumidor como provedor empregam esse algoritmo para obtenção de suas chaves. Similarmente ao **BKeygen**, sk_U também terá ***params*** incorporado e sua divulgação explícita não será mais necessária.

- **Withdraw(*C*(**pk_B, sk_C, ***n*), *B*(**pk_C, sk_B, ***n*))**: É o protocolo que um consumidor ***C*** utiliza para sacar uma carteira ***W*** de ***n*** moedas (e-cash), ou ter como retorno uma mensagem de erro. A saída do ***B*** é a informação T_W, que permite ao Banco rastrear o consumidor, caso efetue-se o gasto múltiplo de uma mesma moeda, ou uma mensagem de erro. O Banco mantém uma base de dados ***D*** para essa operação de rastreamento, na qual se registra a informação (pk_C, T_W).

- **Spend**(***C***(***W***, pk_P), ***P***(**sk_P, pk_B, ***n***)):** É o protocolo que um consumidor ***C*** emprega para dar uma de suas moedas de sua carteira ***W*** para outro usuário, neste caso, o provedor ***P***. O provedor obtém o número serial ***S*** da moeda, e uma prova π da validade da moeda. A saída do consumidor é uma carteira atualizada ***W'***.

- **Deposit**(***P***(**sk_P**, ***S***, π, pk_B), ***B***(**pk_P, sk_B**)): É o protocolo que o provedor ***P*** usa para depositar uma moeda (***S***, π) em sua conta no Banco ***B***. Sempre que um ***P*** honesto obtém (***S***, π) executando o protocolo **Spend** com qualquer usuário (honesto ou não), existe uma garantia que essa moeda será aceita pelo Banco. O ***B*** adiciona (***S***, π) para sua lista ***L*** de moedas gastas. A saída do provedor é nada ou uma mensagem de erro.

- **Identify**(***S***, $\pi 1$, $\pi 2$): É o algoritmo que permite identificar gastadores múltiplos utilizando um número serial ***S*** e duas provas de validade da moeda, $\pi 1$ e $\pi 2$, possivelmente submetidos por um provedor malicioso. Esse algoritmo retorna a chave pública pk_U e uma prova Π_G. Se o provedor que enviou $\pi 1$ e $\pi 2$ não for

malicioso, então $\boldsymbol{\Pi_G}$ é a evidência de que $\boldsymbol{pk_U}$ é a chave pública do usuário que gastou duplamente $\boldsymbol{S}$.

- **VerifyGuilt**($\boldsymbol{S}$, $\boldsymbol{pk_C}$, $\boldsymbol{\Pi_G}$): É o algoritmo que permite verificar publicamente a prova de que o consumidor com a chave pública $\boldsymbol{pk_C}$ é culpado pelo gasto duplo da moeda $\boldsymbol{S}$

- **Trace**($\boldsymbol{S}$, $\boldsymbol{pk_C}$, $\boldsymbol{\Pi_G}$, $\boldsymbol{D}$, $\boldsymbol{n}$): É o algoritmo executado em uma chave pública de um gastador duplo $\boldsymbol{pk_C}$, em uma prova de culpa $\boldsymbol{\Pi_G}$ relativa à uma moeda $\boldsymbol{S}$, em uma base de dados $\boldsymbol{D}$ e em um tamanho de carteira $\boldsymbol{n}$, para computar os números seriais $\boldsymbol{S_1}$, . . . , $\boldsymbol{S_m}$ de todas as moedas emitidas para $\boldsymbol{C}$, juntamente com as provas de propriedade $\boldsymbol{\Pi_1}$, . . . , $\boldsymbol{\Pi_m}$ de $\boldsymbol{pk_C}$. Se o algoritmo **VerifyGuilt**($\boldsymbol{S}$, $\boldsymbol{pk_C}$, $\boldsymbol{\Pi_G}$) não executar, caso, por exemplo, $\boldsymbol{pk_C}$ seja honesto, o algoritmo não realiza nada.

- **VerifyOwnership**($\boldsymbol{S}$, $\boldsymbol{\Pi}$, $\boldsymbol{pk_C}$, $\boldsymbol{n}$): É o algoritmo que permite verificar publicamente a prova $\boldsymbol{\Pi}$ que uma moeda com número serial $\boldsymbol{S}$ pertence a um gastador duplo com chave pública $\boldsymbol{pk_C}$.

3.5. CONSIDERAÇÕES SOBRE AS CAMADAS DA ESTRUTURA

O mérito da contribuição instituída pela estrutura multicamadas de proteção da privacidade do consumidor de serviços SaaS reside na abordagem modularizada, flexível e objetiva dos itens de interesse expostos ao provedor no cenário modelo SaaS apresentado na Figura 2.7.

Os três níveis de interação presentes (Tabela 2.1) são tratados por quatro camadas conceituais que resguardam a privacidade por meio do anonimato (Tabela 3.1).

O design é flexível, já que as tecnologias de anonimato apropriadas para cada camada podem ser substituídas para acomodar outras propriedades mais adequadas a uma nova ou diferente realidade. Essencial é que a abordagem objetiva do anonimato dos respectivos itens de interesse da camada seja mantida.

Outro ponto relevante que deve ser considerado é a segurança oferecida por cada camada e tecnologia. Entretanto, o foco desta contribuição não é provar matematicamente a validade de cada sistema, ainda porque, caso exista o interesse, as provas e detalhamentos dos algoritmos podem ser consultados externamente. Especificamente, no caso, por exemplo, da camada de anonimato de metadados, onde os sistemas de assinaturas de grupo e de dinheiro eletrônico foram discutidos, juntamente com a delineação dos modelos atinentes de assinaturas de grupo com revogação do tipo verificador-local (Design de Boneh e Shacham) e do *E-cash* Compacto, tais informações técnicas podem ser obtidas em [23] e [29], respectivamente.

O que deve ser analisado de acordo com as escolhas específicas das tecnologias de anonimato é como se procedem as interações entre os sistemas. Deve verificar-se se uma tecnologia não interfere no anonimato de outra camada ao implementar seus algoritmos. Nesse sentido, no Capítulo 6, realizar-se-á uma validação experimental das contribuições originais da tese, atestando, na prática, o mérito da análise de privacidade (Seção 2.4), o conceito da estrutura multicamadas (Capítulo 3) e a operacionalidade dos esquemas práticos (Capítulos 4 e 5).

3.6. ESQUEMAS PRÁTICOS

A seguir, nos próximos Capítulos, a terceira contribuição original da tese será apresentada. Dois esquemas práticos da estrutura multicamadas serão propostos. Um permitirá o consumo anônimo de serviços SaaS rastreável (Capítulo 4), ao passo que o outro proporcionará uma versão não rastreável do anonimato (Capítulo 5).

Definem-se aqui os esquemas práticos como instâncias da estrutura multicamadas. Eles selecionam as tecnologias de anonimato mais apropriadas para o cenário objetivado e oferecem um design de transações que suporta a implementação dessas tecnologias e, consequentemente, proporciona a proteção da privacidade do consumidor. São, portanto, projetos para implantação do anonimato para o consumidor SaaS.

Por exemplo, pode-se projetar um design de transações que selecione e compatibilize para as camadas de rede, conteúdo e contrato as tecnologias de anonimato de Mix-Net, anonimato-k e corretor de nuvem, respectivamente. E, para a camada de metadados

pode-se optar pelas assinaturas de grupo, e mais especificamente, pelo sistema BS, para proporcionar a privacidade do consumidor, juntamente com a possibilidade de rastreamento de um consumidor mal intencionado.

Em outra situação, então, pode se optar pelo dinheiro eletrônico somente para a camada de metadados e, dessa maneira, obter-se-á as mesmas características de anonimato do exemplo anterior, mas será excluída a capacidade de rastreamento da identidade do consumidor.

A Figura 3.3 abaixo mostra a relação entre a estrutura multicamadas e os esquemas práticos.

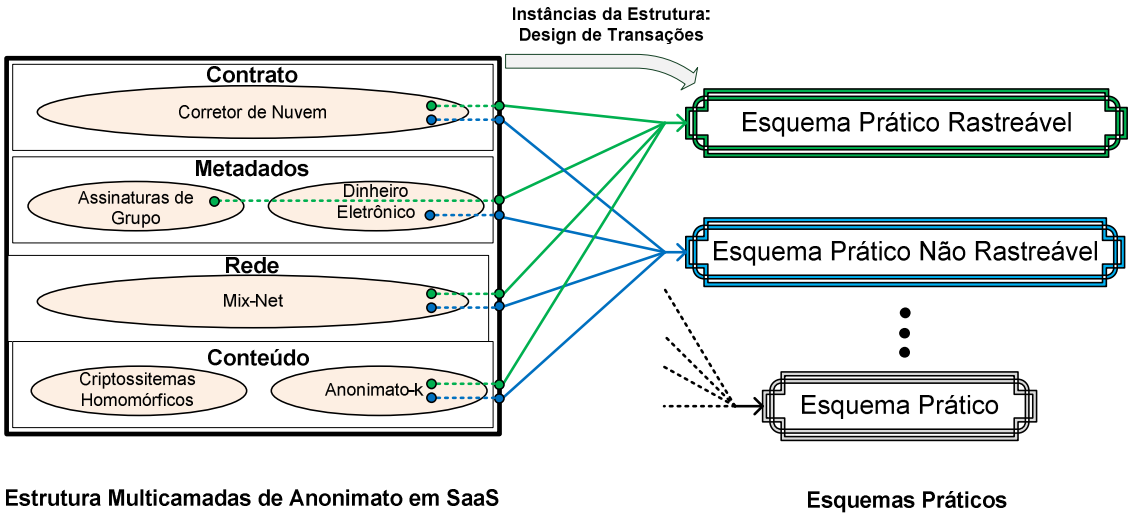

Figura 3.3 – Relação entre a Estrutura Multicamadas e os Esquemas Práticos.

4. ESQUEMA PRÁTICO PARA CONSUMO ANÔNIMO RASTREÁVEL DE SERVIÇOS SAAS

Neste Capítulo tratar-se-á da descrição de um esquema prático operando segundo o paradigma estabelecido pela estrutura multicamadas de anonimato proposta no Capítulo anterior. O foco será oferecer a possibilidade de um consumo anônimo de serviços SaaS, no cenário modelo detalhado na Figura 2.7, protegendo a privacidade do consumidor em relação ao provedor, mas suportando o rastreamento de identidade, caso haja uma necessidade legal, no evento, por exemplo, de um comportamento malicioso do consumidor.

4.1. DESCRIÇÃO

O design deste esquema prático permite a proteção efetiva de todas as dimensões de privacidade do consumidor. Empregar-se-á o anonimato de conexão nas camadas de contrato e de metadados de forma a atender às particularidades específicas do paradigma da computação em nuvem. Para as camadas de conteúdo e de rede, as tecnologias de anonimato de conexão e de dados poderão ser utilizadas em suas formas tradicionais, pois não têm seus modos de operação influenciados diretamente pelas características particulares da computação em nuvem.

Estabelecem-se os seguintes requerimentos para o esquema:

- **Contrato:** Contratos indiretos devem assegurar a política de provisão de serviço ao consumidor e o pagamento devido ao provedor, ao passo que o ID do consumidor e seu contexto de negócios permanecerão privados.

- **Metadados:** As credenciais AAA necessárias não revelarão a identidade do consumidor, mas **poderão** ser **rastreadas** em casos especiais. Além disso, um consumo de serviço realizado com uma credencial não poderá ser correlacionado no tempo com outro consumo do mesmo usuário.

- **Conteúdo e Rede:** Os dados das mensagens não poderão revelar informações sensíveis do consumidor para o provedor durante o consumo do serviço, e os

endereços IP deverão ser anonimizados, tornado a conexão anônima do ponto de vista do protocolo de rede.

De acordo com essas premissas, o consumo de serviços SaaS requer que o provedor seja capaz de autenticar o consumidor como uma entidade que concordou com a utilização dos serviços de acordo com as políticas e preços anunciados. E, o respectivo consumo de serviços deve ser medido, faturado e pago. Para, portanto, permitir que essas garantias formais continuem a existir em um cenário de consumo anônimo de serviço, a relação direta entre consumidor e provedor, estabelecida pelo contrato, deverá ser substituída por contratos indiretos. Neste esquema prático, o corretor de nuvem, do tipo de corretor de contratos SaaS (Seção 3.3.1.1), TPB, contratará com o provedor em nome do consumidor e vice-versa.

Esta abordagem de anonimato rastreável utiliza assinaturas de grupo como o sistema de credenciais anônimas (Seção 3.4.1). As assinaturas de grupo permitem que qualquer membro do grupo assine suas mensagens, mas esses mantêm suas identidades secretas. Um verificador poderá atestar que assinatura foi gerada por um membro legítimo do grupo, mas a assinatura em si não revelará informação sobre qual dos membros efetivamente a assinou. Membros do grupo gozam de anonimato, significando que suas identidades são matematicamente complexas de serem reveladas, dada apenas uma assinatura criada.

Especificamente neste esquema prático, opta-se pelo design de assinaturas de grupo com revogação do tipo verificador-local – VLR (modelo de Boneh e Shacham [23]) – Seção 3.4.1.1. A escolha baseia-se na possibilidade de rastreabilidade, simplicidade computacional e adequação das demais propriedades ao esquema prático pretendido, conforme discussão fundamentada pelas Tabelas 3.2, 3.3 e 3.4, no Capítulo anterior.

O TPB (**contratos indiretos** – **camada de contrato**) assumirá o papel mestre do grupo (**assinaturas de grupo** – **camada de metadados**), combinando as responsabilidades e funções de um Corretor de Contratos SaaS e de um mestre de grupo VLR. Em atenção às tarefas de intermediador de serviço (Seção 3.3.1.1), e as relacionando com as capacidades do mestre do grupo, o TPB será capaz de:

- **Considerando o Credenciamento dos Consumidores:** O TPB proverá a emissão, distribuição, revogação e rastreamento das credenciais:

    - **Emissão e Distribuição de Credenciais:** O conjunto completo de credenciais formará o grupo autorizado a consumir os serviços. Cada consumidor tem sua chave própria e a utiliza para assinar suas mensagens. O provedor será capaz de verificar que uma mensagem foi assinada por um membro legítimo do grupo, mas não poderá identificar unicamente esse membro.

    - **Revogação de Credenciais Emitidas:** O modelo VLR permite a revogação de credenciais anônimas, ou seja, o TPB poderá cancelar a efetividade da assinatura de um membro, o incapacitando de continuar gerando assinaturas anônimas.

    - **Revelação da Identidade de um Consumidor em Caso de Disputa ou Mau Comportamento:** Como o mestre do grupo, o TPB é capaz de expor quem assinou uma determinada mensagem, concretizando a propriedade de rastreabilidade.

- **Considerando Tarifação e Faturamento:** O TPB, por ser a única entidade capaz de expor o criador de uma assinatura, será responsável por tarifar e faturar os consumidores. Os provedores enviarão os registros de consumo anônimo de serviço (assinados) ao TPB, que por sua vez, empregando a propriedade de rastreabilidade, consolidará o consumo de cada consumidor e emitirá uma fatura respectiva.

- **Considerando o Pagamento**: Como mais uma função de intermediação de serviços, o TPB, com base na tarifação e faturamento realizados, agenciará o pagamento aos provedores, salvaguardando a privacidade dos consumidores.

Em uma operação normal fora deste esquema prático, utilizando-se esquemas de precificação típicos, como pré-pago[13], assinatura[14] ou assinatura com limitação[15], o provedor teria que contabilizar cada instância de consumo de serviço. Assim, essas medidas iriam ser passadas internamente para o sistema de tarifação e seriam utilizadas para o processamento do faturamento e pagamento. Neste esquema prático, como o consumo é anônimo para o provedor, este não tem como realizar a tarifação diretamente. Como visto acima, o provedor deverá encaminhar as assinaturas anônimas recebidas no consumo dos serviços para o TPB concretizar a tarifação, o faturamento e o pagamento. Isso, no entanto, não impedirá o provedor de prover o serviço imediatamente, pois ele terá a capacidade de verificar a associação de um consumidor com o grupo autorizado.

Para as camadas de **conteúdo** e de **rede**, onde não é necessária a atuação do TPB, as tecnologias de anonimato de dados e de conexão apropriadas para os itens de interesse dessas camadas podem ser aplicadas em suas formas padrões. Para a camada de conteúdo, utilizar-se-á aqui a técnica de **anonimato-k**, e para a camada de rede, será empregada uma rede **Mix-Net**, mais especificadamente, o Tor. Como a estrutura multicamadas de anonimato SaaS é flexível, essas escolhas podem ser alteradas e um novo esquema prático pode ser criado. O que deve ser respeitado, porém, é a manutenção da privacidade dos itens de interesse considerados e que o design de transações estabelecido permita a operação das camadas de anonimato em conjunto.

4.1.1. Considerações sobre a Limitação do Consumo de Serviço

Neste esquema prático, suportam-se os modelos de precificação pré-pago ou de assinatura limitada (“controle”). Em ambos os casos, ao consumidor são concedidos “créditos de serviço”, que são empregados para limitar a quantidade de consumo

[13] Em um esquema pré-pago, um consumidor paga antecipadamente por uma quantidade específica de uso de um serviço.
[14] Em um esquema de assinatura tradicional, um consumidor contrata uma quantidade mínima regular de uso de um serviço, a chamada “franquia”, e para a utilização que excede essa quantidade, a tarifação é complementada.
[15] Em um esquema de assinatura com limitação, comumente denominada “controle”, um consumidor contrata uma quantidade regular e limitada de uso de um serviço, e todo consumo que for demandado além dessa quantidade é recusado, até que o período de faturamento seja renovado ou até que o consumidor adquira mais acesso.

autorizado de serviço. Essa opção é útil para quando o provedor prefere controlar o uso dos consumidores, no caso, por exemplo, de possibilitar a limitação de dano a ser causado por um possível ataque, ou quando o próprio provedor tem capacidade restrita de prover o serviço.

Portanto, o TPB deverá também distribuir créditos de uso único. Esses serão distribuídos aos consumidores de acordo com a respectiva concessão de consumo. Cada vez que um usuário precisar consumir o serviço, ele incluirá o crédito na mensagem de solicitação. O provedor autenticará o consumidor anônimo e também verificará o crédito de serviço e o cancelará (registrará em uma base de dados de créditos utilizados). Para evitar a correlação de consumos subsequentes, os créditos de serviço têm que ser aleatorizados e ter distribuição uniforme. Dessa maneira, o conhecimento sobre um crédito não revela informação acerca dos outros pertencentes ao mesmo consumidor.

Os créditos de serviço podem ser gerados pelo provedor e encaminhados ao TPB, que os aleatoriza e os distribui. Alternativamente, o provedor pode informar ao TPB quantos créditos poderão ser criados, e o TPB promoverá então a criação e aleatorização dos mesmos. Em ambos os casos, quando o esgotamento dos créditos se aproximar, em função ou do fim do período em voga do modelo de assinatura com limitação ou do fim dos créditos ainda válidos dos consumidores do modelo pré-pago, o provedor poderá ou solicitar a criação de mais créditos junto ao TPB, ou emitir um novo conjunto e o enviar para o TPB efetuar a aleatorização e distribuição.

4.1.2. Design

O design completo do esquema prático rastreável é mostrado na Figura 4.1, em suas etapas.

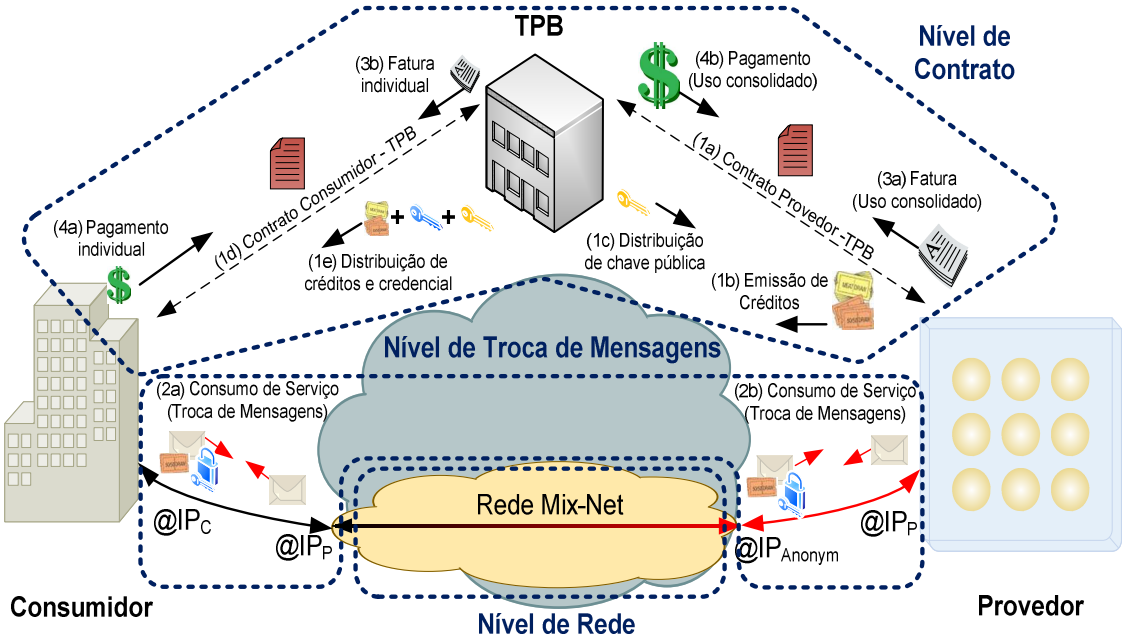

Figura 4.1 – Design do Esquema Prático Rastreável para Consumo de Serviços SaaS

Anteriormente ao consumo de serviço propriamente dito, o provedor e o TPB estabelecem seu contrato (1a), o provedor emite um conjunto de "créditos de serviço" de uso único (1b)[16], de acordo com os aspectos discutidos na Seção 4.1.1, e recebe uma chave pública para autenticação anônima dos consumidores (1c). Em um momento posterior, o consumidor firma seu contrato com o TPB (1d), e recebe sua chave secreta e a pública para assinatura (algoritmo **Sign** – Seção 3.4.1.1), além de uma coleção de créditos aleatórios para o consumo de serviço (1e).

Durante o consumo de serviço, considerando as camadas de conteúdo e rede, as tecnologias de anonimato-k e a rede Mix-Net Tor são empregadas, respectivamente. Dessa forma, antes de uma mensagem partir do consumidor em direção ao provedor, o consumidor anonimiza os dados da mensagem, resguardando, porém, a capacidade do provedor processar o serviço, e, em seguida, utiliza o Tor para anonimizar a conexão de rede (endereço IP). Além disso, cada solicitação de serviço contém um crédito de

[16] Essa etapa pode ser repetida posteriormente a qualquer momento, quando o esgotamento dos créditos se aproximar, em função ou do fim do período em voga do modelo de assinatura com limitação ou dos créditos ainda válidos dos consumidores do modelo pré-pago.

serviço e é assinada anonimamente pelo sistema VLR (2a). A assinatura (algoritmo **Sign** – Seção 3.4.1.1) utiliza como entrada as chaves recebidas do TPB e o *hash* da mensagem, que passa a ser o parâmetro ***M*** do algoritmo. Ao receber essa mensagem de requisição de serviço (2b), o provedor autentica a assinatura (algoritmo **Verify** – Seção 3.4.1.1), e checa o crédito de serviço (verifica em sua lista se o crédito não foi utilizado anteriormente). Se as duas operações tiverem sucesso, o provedor cancela o crédito (registra seu uso) e fornece o serviço.

Uma assinatura da mensagem, o *hash* da mensagem e o respectivo crédito de serviço constituem um registro de consumo, que é inserido em uma base de dados local e enviado também ao TPB, em um documento consolidado (3a). O TPB usa esses registros e a chave secreta de rastreamento de mestre do grupo para individualizar cada consumidor (algoritmo **Trace** – Seção 3.4.1.1), gerar a fatura respectiva e a enviar para o consumidor apropriado (3b). Deve-se notar que o TPB não tem acesso ao conteúdo da mensagem, apenas à assinatura, *hashes* de mensagens e créditos de serviço.

No caso do TPB intermediar também o pagamento, ele recebe a parte devida por cada consumidor (4a) e, finalmente, encaminha o pagamento consolidado para o provedor (4b). A etapa de pagamento (4) via TPB não é obrigatória, mas é necessária caso se queira manter privada a informação de quanto serviço cada consumidor está utilizando.

Observa-se que neste esquema o TPB é considerado uma entidade confiável, pois poderá expor o anonimato do consumidor. Esta característica é conveniente para um cenário onde existe um requisito legal de rastreamento e punição de comportamentos mal intencionados.

4.1.3. Considerações Práticas

Em relação aos fatores práticos envolvendo a utilização do esquema prático, discutem-se abaixo os aspectos relevantes.

4.1.3.1. Performance

Segundo o modelo de Boneh e Shacham [23], a geração de assinaturas (consumidor) requer aproximadamente oito exponenciações e duas computações de mapas bilineares.

Já a verificação da assinatura, procedimento a ser realizado pelo provedor, demanda seis exponenciações e 3 + 2 |***RL***| computações do mapa bilinear, onde |***RL***| é o número de entidades presentes no conjunto de provas de revogação.

No próprio design VLR [23], propõe-se um algoritmo mais eficiente para checar as assinaturas, cujo tempo de execução é independente de |***RL***|. Todavia, ele implica em uma leve diminuição do anonimato proporcionado pelo sistema [23]. Levando-se em conta que este esquema prático é focado no anonimato do consumidor, e que não será do interesse desse continuar utilizando os serviços caso seu anonimato tenha sido revogado, pode-se concluir que, em operação normal, o |***RL***| será zero no provedor. Assim, o desempenho para a verificação de assinaturas no provedor já será constante e independente de |***RL***|, não justificando a utilização desse outro algoritmo.

Para o caso do TPB, este sim terá |***RL***| diferente de zero, pois, na verdade, precisa de |***RL***| igual ao número de consumidores para o cálculo do algoritmo implícito **Trace** (Seção 3.4.1.1). Entretanto, como o TPB realiza essa operação em uma fatura compilada e em caráter *off-line* (Figura 4.1), não existe demanda imperiosa de otimização do desempenho computacional. Inclusive, o TPB pode capitalizar uma infraestrutura de computação em nuvem para aumentar sua performance operacional, caso necessário.

4.1.3.2. *Overhead* – Tamanho das Assinaturas

Como as assinaturas de grupo serão as credencias AAA a serem utilizadas no consumo de serviços SaaS, deve-se avaliar o tamanho das mesmas e o impacto em uma implementação em *software* deste esquema prático.

O design de Boneh e Shacham [23] emprega em seu funcionamento a fundamentação matemática de mapas bilineares. E, embora vários grupos com mapas bilineares sejam baseados em curvas elípticas, as definições usadas no contexto do modelo são abstratas e podem ser ajustadas para a utilização com mais de um tipo de curvas elípticas.

No caso de utilização da família de curvas MNT [81], como descrito em [22], o tamanho da assinatura é de 1192 bits, ou 149 bytes. Com esses parâmetros, a segurança é aproximadamente a mesma de uma assinatura RSA padrão, que tem 128 bytes.

Empregando curvas Barreto Naehrig [13], as assinaturas passam a ter 1122 bits com o mesmo nível de segurança.

4.1.3.3. Implementação

Este esquema prático oferece um design de transações que pode ser empregado em um projeto para a implementação em *software* de um serviço em nuvem que opere segundo este paradigma.

Considerando a operação conjunta das camadas de contrato e metadados, quando se usa *web services* SOAP [129] como meios para implementar serviços de nuvem, no contexto deste esquema prático, as credenciais anônimas podem ser encapsuladas diretamente pelos protocolos padrão, como WS-Security [92]. Assinaturas de grupo, no fim, são apenas assinaturas digitais.

O WS-Security é deliberadamente aberto a fim de permitir uma grande variedade de mecanismos para assinatura e encriptação. Também, as operações de assinar e de criptografar são aspectos distintos, podendo ser utilizadas juntas ou não, em concordância com os requisitos de segurança.

O emprego do WS-Security possibilita à implementação do esquema prático ser estendida para incorporar cifração em nível de mensagem, protegendo os dados e metadados enquanto o pacote trafega pela rede, ou ainda quando existe processamento intermediário da mensagem; em oposição à criptografia em nível de transporte, como TLS.

Para a camada de rede, a implementação Tor [123] pode ser empregada diretamente pelo consumidor, anonimizando o endereço IP das requisições. Já para a camada de conteúdo, a utilização do anonimato-k demandará que o *software* do consumidor incorpore a reponsabilidade de anonimizar os microdados sendo enviados ao provedor.

De destacar, no Capítulo 6, durante a validação experimental da tese, uma implementação em *software* deste esquema prático rastreável será apresentada e detalhada, fazendo uso do WS-Security.

4.2. AVALIAÇÃO DE PRIVACIDADE

Passa-se agora para uma avaliação de como o esquema prático para consumo anônimo rastreável de serviços SaaS influencia e protege a privacidade do consumidor de nuvem. A análise explana como cada camada da estrutura conceitual é tratada e operacionalizada, em especial na maneira como seus itens de interesse são abordados.

Contudo, como já aclarado durante o texto, o objetivo é uma avaliação qualitativa, não se buscando a demonstração matemática de cada premissa de anonimato. Essas podem ser obtidas referindo-se ao material teórico específico de cada tecnologia usada nas respectivas camadas da estrutura e do esquema prático.

4.2.1. Camada de Anonimato de Contrato

Com a introdução de contratos indiretos, os itens de interesse que comprometem a privacidade do consumidor presentes no contrato (Identificação do Consumidor, Endereço do Consumidor, SLA e estimativas de consumo – Tabela 3.1) não estão mais disponíveis para o provedor. Apenas o TPB terá acesso a tais informações. E, como o TPB, neste esquema prático é considerado confiável, esse acesso não viola os requerimentos propostos (Seção 4.1). Ainda assim, como delineado no design da Figura 4.1 – Seção 4.1.2, mesmo sendo confiável, o TPB não terá acesso às mensagens trocadas, sendo seu papel apenas administrativo, na geração de faturas.

Apesar do algoritmo implícito **Trace** (Seção 3.4.1.1), com funcionamento baseado no algoritmo **Verify**, precisar da entrada ***M***, chamada lá de mensagem, já se esclareceu que esse parâmetro será o *hash* da mensagem, e, portanto, não revelará o conteúdo da mensagem para o TPB. Além disso, a geração das faturas também se beneficia do *hash*, uma vez que o *hash* das mensagens terá tamanho fixo e será menor do que a informação original, simplificando o processamento do TPB.

Destarte, considerando a Tabela 3.1, os itens de interesse mantêm-se protegidos, assegurando a privacidade das dimensões de ID, localização e comportamento nesta camada.

Adicionalmente, com vias a possibilitar uma confrontação das informações de consumo de serviço geradas pelo provedor ao TPB, no sentido de uma possível desconformidade com SLA anunciado ao consumidor, pode-se adaptar o design da Figura 4.1 para incorporar um envio de requisição duplicado ao TPB, além daquele destinado ao provedor, na etapa (2a). Isso, mesmo que revele o conteúdo da mensagem ao TPB, permitirá a confrontação dos registros de consumo, podendo implicar em glosas na fatura do serviço não inicialmente detectáveis, por erro de registro do provedor, ou mesmo devido à indisponibilidade do serviço.

Contudo, em uma operação normal, seguindo estritamente o design proposto na Figura 4.1, o provedor ainda poderá infringir o SLA contratado pelo consumidor indiretamente por meio do TPB. O consumidor é anônimo para o provedor e este não conseguirá identificar e, consequentemente, mascarar um atendimento fora do SLA anunciado para um consumidor específico. Dessa forma, o TPB, ao individualizar uma fatura compilada e enviada pelo provedor, poderá verificar a conformidade com o SLA contratado e determinar algum possível desconto já no momento da emissão da fatura individual.

4.2.2. Camada de Anonimato de Metadados

Os itens de interesse da camada de metadados são as credenciais AAA e as MEPs no tempo. Com as assinaturas de grupo do modelo Boneh e Shacham, que são utilizadas nesse esquema prático, o provedor pode autenticar as requisições de serviço, mas não consegue distinguir qual consumidor assinou a mensagem, já que uma assinatura válida pode ser imediatamente reconhecida, mas ninguém, exceto o signatário e o mestre do grupo, é capaz de reconhecer qual membro do grupo a gerou. Essa condição se mantém também para mensagens sucessivas assinadas pelo mesmo consumidor. Assim, os requerimentos propostos (Seção 4.1) de metadados são cumpridos, bem como as dimensões da privacidade do consumidor de ID e de comportamento, presentes na camada de metadados da estrutura multicamadas, referentes às credenciais AAA e MEPs no tempo, são preservadas.

Um aspecto relevante a ser discutido nesta camada aplicada ao esquema prático rastreável é o estado de grupo empregado pelo design de Boneh e Shacham, no tocante à

criação dos grupos. Neste modelo específico, os grupos são estáticos, estabelecidos com o número total de membros *a priori*, como demonstrado no algoritmo **Keygen** (Seção 3.4.1.1). Todavia, essa característica não impacta negativamente o esquema prático, uma vez que não existe problema em definir uma quantidade projetada de consumidores pelo provedor, e distribuir as credenciais somente quando forem requisitadas. Mais ainda, como visto na Seção 4.1.1, o próprio provedor já realiza uma expectativa de consumo ao emitir seus créditos de serviço e os entregar para o TPB. Assim, uma abordagem eficaz é, no momento da determinação da quantidade de créditos de serviço, instituir também a quantidade de consumidores esperados, criando uma relação não rigorosa de créditos para cada consumidor. Obviamente, o próprio TPB pode atribuir mais ou menos créditos para uma credencial específica (sem ultrapassar o montante global especificado pelo provedor), conforme contratado pelo consumidor.

4.2.3. Camadas de Anonimato de Conteúdo e de Rede

A camada de anonimato de conteúdo tem como item de interesse os dados das mensagens, impactando diretamente a dimensão de privacidade de conteúdo. Indiretamente, via análise semântica dos dados, esse item de interesse pode afetar também as dimensões de ID, localização e comportamento.

Uma abordagem de anonimato de dados específica deve ser utilizada em cada serviço de nuvem, pois características diferentes das estruturas de dados da lógica implementada demandam técnicas distintas para que o anonimato dos dados conserve a possibilidade de processamento do serviço, enquanto possa manter a privacidade do consumidor. Portanto, antes do envio das mensagens ao provedor, o consumidor deverá tratar as informações potencialmente identificadoras presentes na mensagem. Uma técnica que pode ser empregada neste caso é o anonimato-k, que de fato é a tecnologia escolhida aqui. Todavia, a customização necessária para a anonimização dos microdados envolvidos via anonimato-k é um procedimento que deve ser realizado de acordo com o serviço sendo considerado.

Para o anonimato de conexão, da camada de rede, responsável pelo item de interesse do endereço IP, que por sua vez afeta as dimensões de privacidade de ID, localização e

comportamento, a técnica mais adequada é o emprego de Mix-Nets. Neste esquema prático seleciona-se o Tor para anonimizar os endereços IPs do consumidor, protegendo a privacidade de ID e de localização. Mais ainda, como o Tor emprega a troca dos endereços IP anonimizados com o tempo, a privacidade de comportamento também é resguardada.

4.2.4. Visão Geral

A Tabela 4.1 resume como as tecnologias de anonimato utilizadas pelo esquema prático rastreável protegem a privacidade do consumidor em relação ao provedor. Especifica-se, nela, o nível de acesso que o TPB e o provedor têm aos itens de interesses delimitados pelas camadas e protegidos pelo esquema (Tabela 3.1), em concordância com a avaliação de privacidade apresentada nesta Seção 4.2. Neste esquema prático, o TPB é uma entidade confiável.

Tabela 4.1 – Visão Geral da Privacidade Estabelecida com o Esquema Prático Rastreável

Camada da Estrutura	**Itens de Interesse**	**Tecnologia de Anonimato Empregada**	**Dimensão da Privacidade Protegida**	**Acesso TPB**	**Acesso Provedor**
Contrato	Identificação do Consumidor	Contratos Indiretos: *TPB*	ID	Completo	Não tem
	Endereço do Consumidor	Contratos Indiretos: *TPB*	Localização	Completo	Não tem
	SLA e estimativas de consumo	Contratos Indiretos: *TPB*	Comportamento	Completo	Não tem
Metadados	Credenciais AAA	Credenciais Anônimas: *Assinaturas de Grupo*	ID	Completo	Tem acesso, mas consumidor é anônimo
	MEPs no tempo	Credenciais Anônimas: *Assinaturas de Grupo*	Comportamento	Completo	Tem acesso, mas consumidor é anônimo
Conteúdo	Dados das Mensagens	Anonimato-k	ID, Localização, Comportamento, Conteúdo	Não tem	Tem acesso, mas consumidor é anônimo
Rede	Endereço IP	Mix-Net: *Tor*	ID, Localização, Comportamento	Não tem ao utilizado na troca de mensagens	Tem acesso, mas consumidor é anônimo

5. ESQUEMA PRÁTICO PARA CONSUMO ANÔNIMO NÃO RASTREÁVEL DE SERVIÇOS SAAS

Em continuação à apresentação dos esquemas práticos para o consumo anônimo de serviços SaaS, neste Capítulo tratar-se-á da descrição de um esquema prático não rastreável operando segundo o paradigma proposto pela estrutura multicamadas. Destarte, e diferentemente do esquema do Capítulo anterior, o objetivo será possibilitar um consumo anônimo absoluto de serviços SaaS, onde a privacidade do consumidor em relação ao provedor não poderá ser revelada com base somente no poder de uma entidade, como o TPB pode efetivar no esquema prático rastreável..

5.1. DESCRIÇÃO

O design deste esquema prático também permite a proteção efetiva de todas as dimensões de privacidade do consumidor. Semelhantemente ao esquema rastreável, se fará o uso do anonimato de conexão nas camadas de contrato e de metadados de forma a atender às particularidades específicas do paradigma da computação em nuvem. E, para as camadas de conteúdo e de rede, as tecnologias de anonimato de conexão e de dados serão utilizadas em suas formas tradicionais, já que não têm seus modos de operação influenciados diretamente pelas características particulares da computação em nuvem.

Neste esquema, os seguintes requerimentos são instituídos:

- **Contrato:** Contratos indiretos permitirão a aquisição de credenciais anônimas, que por sua vez serão o próprio pagamento a ser transferido ao provedor e trocado por serviço, sem, porém, expor a identidade do consumidor. Deve-se notar que os contratos indiretos aqui serão distintos dos presentes no esquema rastreável. Aqui terão caráter econômico, especificando a precificação e as características da provisão do serviço. Estabelecer-se-á a relação entre o Banco e o consumidor (e provedor), mas não se focará no paradigma da garantia e cobrança de SLA.

- **Metadados:** As credenciais AAA necessárias não revelarão a identidade do consumidor, nem poderão ser rastreadas. Além disso, um consumo de serviço realizado com uma credencial não poderá ser correlacionado no tempo com outro consumo do mesmo usuário.

- **Conteúdo e Rede:** Os dados das mensagens não poderão revelar informações sensíveis do consumidor para o provedor durante o consumo do serviço, e os endereços IP deverão ser anonimizados, tornado a conexão anônima do ponto de vista do protocolo de rede.

O esquema consiste basicamente na inclusão de pagamento eletrônico na própria mensagem de requisição de serviço, fazendo uso da tecnologia de dinheiro eletrônico – *E-cash* (Seção 3.4.2). O contrato indireto é estabelecido na forma da relação econômica entre banco e consumidor, e banco e provedor. Nesse contexto, o TPB assume o papel do **Banco**, agora um tipo de corretor de nuvem, no estilo exposto na Seção 3.3.1.1. Segundo esse paradigma, ao consumidor é anunciada a característica de provisão do serviço desejado e lhe é possibilitada a abertura de uma conta, recebendo o dinheiro eletrônico em troca de pagamento real (operação de Saque). Ao provedor é garantida a troca do dinheiro eletrônico recebido pelo pagamento propriamente dito (operação de Depósito). Note que o TPB se torna meramente uma entidade de processamento de pagamento eletrônico. Ele não assume nenhum outro papel no consumo de serviço e também não tem acesso às mensagens a serem trocadas. O consumidor, ou **Pagador** (Seção 3.4.2), goza de anonimato não rastreável, uma vez que credenciais *E-cash* não revelam informação acerca da identidade de seu possuidor e mesmo o Banco não consegue identificar a entidade que gastou o dinheiro eletrônico em questão. Portanto, os **Recebedores** (provedores) devem ter mecanismos e políticas de segurança eficientes em prática, uma vez que um consumidor malicioso de posse de *E-cash* válido não será sujeito à identificação. Esse esquema prático é uma opção apenas para ambientes que possam aceitar tais condições.

Então, esta abordagem de anonimato não rastreável utiliza dinheiro eletrônico como o sistema de credenciais anônimas (Seção 3.4.2). O *E-cash* permite, utilizado neste contexto, que cada mensagem seja “assinada” com um pagamento, ou seja, cada

solicitação já contém o recurso econômico necessário para subsidiá-la. Intrinsicamente, uma credencial constituída de dinheiro eletrônico é anônima e não rastreável, assim como uma moeda ou nota de dinheiro real. Serviços disponibilizados dentro deste esquema prático autenticarão seus consumidores apenas por meio do recebimento do pagamento no próprio momento do consumo do serviço.

Especificamente neste esquema prático, opta-se pelo design de dinheiro eletrônico denominado *E-cash* Compacto (modelo de J. Camenisch, S. Hohenberger e A. Lysyanskaya [29] – Seção 3.4.2.1). A escolha baseia-se na não rastreabilidade, simplicidade computacional e adequação das demais propriedades ao esquema prático pretendido, conforme discussão fundamentada pela Tabela 3.5, presente Capítulo 3.

O TPB (**contratos indiretos – camada de contrato**) assumirá o papel de Banco (**dinheiro eletrônico – camada de metadados**), combinando as responsabilidades e funções de um Corretor de Contratos SaaS e de um Banco. Em atenção às tarefas de intermediador de serviço (Seção 3.3.1.1), e as relacionando com as capacidades do Banco, o TPB será capaz de:

- **Considerando o Credenciamento dos Consumidores:** O TPB proverá a emissão e distribuição das credenciais. Ao decidir por consumir um serviço aderente a este esquema prático, o consumidor abrirá uma conta com o TPB, e receberá credenciais que serão, na verdade, representações de dinheiro eletrônico.

- **Considerando Tarifação e Faturamento:** Como cada consumo de serviço será pago imediatamente no momento de solicitação, não haverá a necessidade de registro para faturamento *a posteriori*.

- **Considerando o Pagamento**: O pagamento será realizado por meio de dinheiro eletrônico, a ser sacado em troca de pagamento real pelo consumidor, na operação de saque, e compensado por meio do intermédio do TPB, no momento do depósito do provedor.

Também da mesma forma que o esquema rastreável, para as camadas de **conteúdo** e de **rede** deste esquema não rastreável, onde não é necessária a atuação do TPB (Banco), as tecnologias de anonimato de dados e de conexão apropriadas para os itens de interesse dessas camadas podem ser aplicadas em suas formas padrões. Para a camada de conteúdo, utilizar-se-á aqui a técnica de **anonimato-k**, e para a camada de rede, será empregada uma rede **Mix-Net**, mais especificadamente, o Tor. Como a estrutura multicamadas de anonimato SaaS é flexível, essas escolhas podem ser alteradas e um novo esquema prático pode ser criado. O que deve ser respeitado, porém, é a manutenção da privacidade dos itens de interesse considerados e que o design de transações estabelecido permita a operação das camadas de anonimato em conjunto.

5.1.1. Considerações sobre o Emprego de Notas ou Moedas

Um aspecto relevante a ser considerado nesta substituição de credenciais típicas AAA por um sistema de credenciais anônimas baseado em *E-cash* é a utilização de notas ou moedas eletrônicas.

Notadamente, o sistema utilizado aqui é o *E-cash* Compacto, como já aclarado. Este sistema, por si só, já busca a diminuição do tamanho de cada representação, instituindo um cenário de micropagamentos, não divisíveis e intrasferíveis.

Caso se empregasse um sistema com a propriedade de divisibilidade, como, por exemplo, o proposto por Okamoto e Ohta em [97], o consumidor poderia sacar menos credenciais mas com valores de face maior (notas *E-cash*). No entanto, as operações de consumo teriam que prever o “troco” a ser emitido, pois a tendência seria de que uma solicitação custasse bem menos do que uma nota.

Logo, o benefício de se sacar junto ao banco quantidades menores de representações de dinheiro eletrônico, as notas, imporia um incremento considerável na complexidade do design do esquema prático, que teria que manter anônimas em todas as camadas da estrutura as trocas de mensagem consistindo na operação adicional de emissão de troco.

Ao invés disso, opta-se nesse esquema prático pela utilização das moedas suportadas pelo *E-cash* Compacto, onde o dinheiro eletrônico é sacado pelo consumidor em uma

quantidade de representações maior, com valores de face menor, escolhidas de maneira que cada moeda coincida com o valor de cada instância de consumo de serviço. Não haverá, portanto, a necessidade de emissão de troco.

Essa escolha se adapta à realidade desse esquema prático. A falta da propriedade de divisibilidade do *E-cash* Compacto não impacta nesse cenário. Mais ainda, o foco em um paradigma de micropagamentos é proveitoso para a necessidade de se realizar transações anônimas atômicas, como as do cenário modelo de consumo de serviços SaaS da Figura 2.7, foco das contribuições desta tese.

Contudo, como a estrutura multicamadas é flexível, pode-se decidir pelo emprego de um sistema de dinheiro eletrônico diferente, mais compatível com outros requisitos ou interesses de um ambiente de consumo específico.

5.1.2. Design

O design completo do esquema prático não rastreável é mostrado na Figura 5.1, em suas etapas.

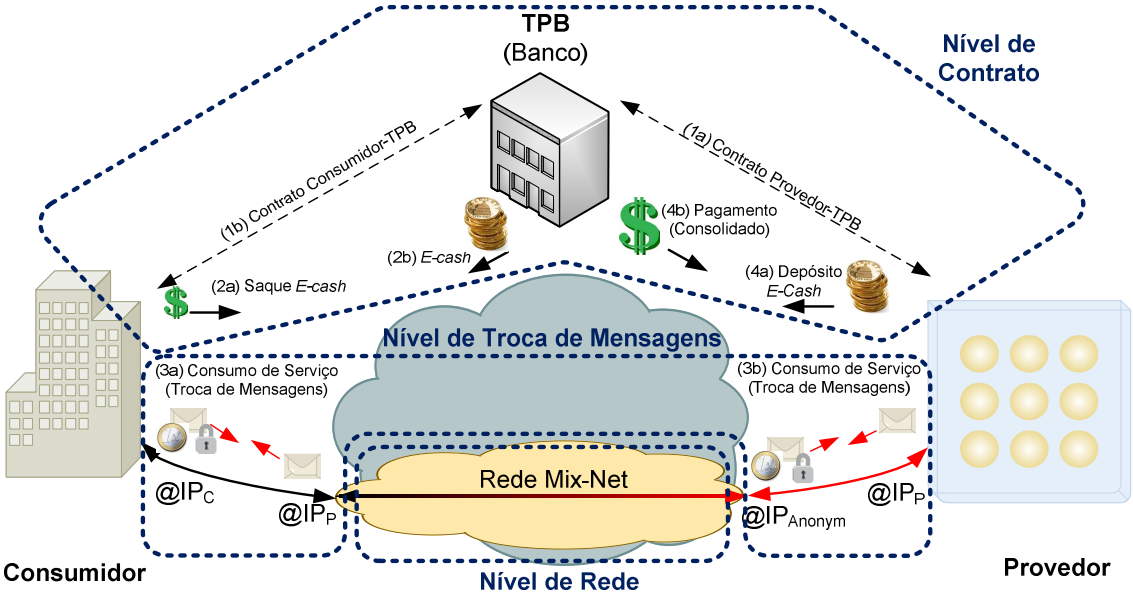

Figura 5.1 – Design do Esquema Prático Não Rastreável de Consumo de Serviços SaaS

Basicamente, no primeiro momento, o consumidor e o provedor irão contratar com o TPB, isto é, abrirão contas com o Banco, respectivamente nas etapas (1a) e (1b). Isso significa que tanto consumidor quando provedor reconhecerão as credenciais *E-cash* emitidas pelo TPB.

Antes do consumo do serviço, nas etapas (2a) e (2b)[17], o consumidor deverá obter as credenciais de dinheiro eletrônico, consistindo das representações do tipo moeda referentes à operação de saque do *E-cash* compacto (**Withdraw** – Seção 3.4.2.1). Para realizar este processo, o consumidor já realiza um pagamento real (ou à credito) junto ao Banco.

Para a etapa (3a), e considerando as camadas de conteúdo e rede, as tecnologias de anonimato-k e a rede Mix-Net Tor são empregadas, respectivamente. Dessa forma, antes de uma mensagem partir do consumidor em direção ao provedor, o consumidor anonimiza os dados da mensagem, resguardando, porém, a capacidade do provedor processar o serviço, e, em seguida, utiliza o Tor para anonimizar a conexão de rede (endereço IP). Depois disso, o consumidor insere uma moeda eletrônica na requisição de serviço e a envia para o provedor (**Spend** – Seção 3.4.2.1). O provedor, ao receber uma requisição de serviço (3b), verifica a credencial de pagamento, checa sua validade e, se o *E-cash* for legítimo, salva a credencial e fornece o serviço.

Enfim, em um momento oportuno, não necessariamente imediatamente após a provisão de serviço, o provedor entrará em contato com o TPB e trocará as credenciais de dinheiro eletrônico (4a) por pagamento real (4b), na operação de depósito (**Deposit** – Seção 3.4.2.1).

Deve-se notar que, como este esquema prático opera utilizando o *E-cash* Compacto, apresentado no contexto da tese na Seção 3.4.2.1 e descrito oficialmente em [29], nenhuma entidade é considerada confiável, nem mesmo o TPB, e todas as premissas de segurança são computacionais.

[17] A operação de saque poderá ser repetida posteriormente, sem a necessidade de executar novamente a etapa (1b).

5.1.3. Considerações Práticas

Em relação aos fatores práticos envolvendo a utilização do esquema prático, discutem-se abaixo os aspectos relevantes.

5.1.3.1. Performance e *Overhead* – Tamanho das Assinaturas

O *E-cash* Compacto foi adotado neste esquema prático devido à possibilidade de representar diversas moedas (por exemplo, 2^{ℓ} moedas) em uma carteira única, enquanto ainda mantendo linear a complexidade dos protocolos de Saque (**Withdraw** – Seção 3.4.2.1) e de Pagamento (**Spend** – Seção 3.4.2.1), junto, também, com o tamanho da carteira. Isto é, a complexidade dos protocolos de Saque e de Pagamento é $O(\ell+k)$ e a carteira pode ser armazenada em $O(\ell+k)$ bits, onde k é um parâmetro de segurança.

Vale ressaltar que a linearidade da complexidade dos protocolos e do tamanho da carteira obedece ao expoente do tamanho da carteira ℓ, como visto acima. E ℓ é notadamente menor que o número de moedas total, que é 2^{ℓ}, atestando ainda mais a eficiência do sistema.

Outro ponto importante a ser levado em conta é que o *E-cash* Compacto tem a propriedade de exculpabilidade (Seção 3.4.2). Mais ainda, o *E-cash* Compacto estende a exculpabilidade para, além de poder identificar um gastador duplo, permitir também a exposição de todas as moedas de propriedade dele (**Trace** – Seção 3.4.2.1). Tendo esse aspecto como determinante na elaboração de todos os algoritmos, o *E-Cash* Compacto em [29] oferece duas construções: uma que requer uma hipótese forte de mapa bilinear aplicável somente para curvas MNT [81], e outra que opera em tipos mais genéricos de curvas elípticas. A primeira tem as complexidades dos protocolos e o tamanho da carteira como já descrito acima, enquanto a segunda, como compensação por uma aplicabilidade mais genérica, passa a ter a complexidade dos protocolos de Pagamento e Saque de $O(\ell \cdot k)$ e $O(\ell \cdot k + k^2)$, respectivamente, ao passo que o tamanho da carteira passa a ser de $O(\ell \cdot k)$ bits. A escolha entre as construções disponíveis deve pesar esses aspectos.

5.1.3.2. Implementação

Este esquema prático oferece um design de transações que pode ser empregado em um projeto para a implementação em *software* de um serviço em nuvem que opere segundo este paradigma.

Da mesma forma que no esquema prático rastreável, e considerando a operação conjunta das camadas de contrato e metadados, quando se usa *web services* SOAP [129] como meios para implementar serviços de nuvem, as credenciais anônimas podem ser encapsuladas diretamente pelos protocolos padrão, como WS-Security [92]. Representações de *E-cash*, portanto, se tornam apenas assinaturas digitais, e a informação trocada pelo protocolo de Pagamento (**Spend** – Seção 3.4.2.1) é embutida nas extensões do WS-Security.

Neste caso, a implementação também se beneficia do fato de o WS-Security ser deliberadamente aberto a fim de permitir uma grande variedade de mecanismos para assinatura e encriptação. Também, as operações de assinar e de criptografar são aspectos distintos, podendo ser utilizadas juntas ou não, em concordância com os requisitos de segurança.

Assim sendo, o emprego do WS-Security possibilita à implementação do esquema prático ser estendida para incorporar cifração em nível de mensagem, protegendo os dados e metadados enquanto o pacote trafega pela rede, ou ainda quando existe processamento intermediário da mensagem; em oposição à criptografia em nível de transporte, como TLS.

Para a camada de rede, a implementação Tor [123] pode ser empregada diretamente pelo consumidor, anonimizando o endereço IP das requisições. Já para a camada de conteúdo, a utilização do anonimato-k demandará que o *software* do consumidor incorpore a reponsabilidade de anonimizar os microdados sendo enviados ao provedor.

5.2. AVALIAÇÃO DE PRIVACIDADE

Semelhantemente ao esquema prático rastreável, prossegue-se neste momento para uma avaliação de como o esquema prático para consumo anônimo não rastreável de serviços SaaS influencia e protege a privacidade do consumidor de nuvem. A análise explana qualitativamente como cada camada da estrutura conceitual é tratada e operacionalizada, em especial na maneira como seus itens de interesses são abordados. Para uma análise demonstrativa ou matemática de cada tecnologia empregada, deve-se remeter ao material teórico respectivo, apresentado e referenciado durante a explanação do esquema prático.

5.2.1. Camada de Anonimato de Contrato

Os itens de interesse que comprometem a privacidade do consumidor presentes no contrato (Identificação do Consumidor, Endereço do Consumidor, SLA e estimativas de consumo – Tabela 3.1) são totalmente protegidos do provedor pelos contratos indiretos. Entretanto, neste esquema prático, os contratos indiretos não tem o caráter administrativo, de um paradigma de garantia e cobrança de SLA. Os contratos indiretos aqui são de cunho econômico, onde tanto consumidor quanto provedor estabelecem contratos de abertura de conta com o TPB (Banco). Cada instância de consumo de serviço é paga atomicamente, com micropagamentos emitidos e reconhecidos pelo TPB, de acordo com esses contratos econômicos, que são, então, embutidos nas requisições. Destarte, as informações impactantes das dimensões de privacidade referentes ao contrato (ID, Localização e Comportamento) não são reveladas ao provedor. O Banco, no entanto, poderá ter acesso parcial ao ID e localização do consumidor, caso no momento de abertura da conta, as informações de pagamento, como, por exemplo, um cartão de crédito, contenham tais itens de interesse. O consumidor deverá empregar uma forma de pagamento distinta ou adaptada, e.g., cartão de crédito pré-pago anônimo[18], caso deseje não revelar essas informações ao TPB. Para o comportamento, o Banco terá

[18] Atualmente existem cartões de crédito pré-pagos que não carregam informações de identificação dos seus proprietários. Essa característica hoje é especialmente utilizada por cartões virtuais, como, por exemplo, os disponíveis pelo http://instantvcc.eu/ (Acessado em 19/04/2013).

acesso parcial, pois contabilizará a quantidade de dinheiro eletrônico emitido para um consumidor.

Este esquema prático é focado e melhor adaptado a um ambiente onde o serviço SaaS a ser oferecido não requer um contrato formal explícito assinado, como define a etapa 3 da Figura 2.7. Se um vínculo mais formal, ou até legal, for necessário entre o consumidor e provedor, nas situações, por exemplo, de limitar gerencialmente o acesso ao serviço ou para possibilitar as compensações financeiras e legais nos casos de mau uso ou desconformidades de SLA, a alternativa dos contratos indiretos do esquema rastreável é recomendada.

Todavia, este esquema prático não rastreável, com o emprego do *E-cash* Compacto (Seção 3.4.2.1), oferece um ambiente onde todas as entidades não precisam ser confiáveis. Mesmo no evento de provedores ou TPBs maliciosos, o anonimato do consumidor gerado pelas credenciais (camada de metadados) é assegurado computacionalmente, e não depende de suposições ou hipóteses baseadas em confiança [29]. Apesar de aparentar, em uma percepção inicial, ser uma configuração estrutural não atrativa, este esquema prático beneficia provedores que desejem uma opção dinâmica e segura para serviços orientados à microtransações, resistente a consumidores maliciosos. Em se alcançando uma infraestrutura de segurança tecnológica (ataques ao *software*) e provisionamento do serviço suficientemente robusta, um provedor poderá capitalizar demandas por ofertas SaaS rápidas e sem formalidades mandatórias. Consumidores poderão desfrutar de acesso imediato, anônimo e não rastreável a um serviço eficiente e seguro.

5.2.2. Camada de Anonimato de Metadados

Dentro da camada de metadados da estrutura multicamadas, os itens de interesse a serem anonimizados são as credenciais AAA e as MEPs no tempo (Tabela 3.1). Com representações de dinheiro eletrônico sendo empregadas como credenciais, dentro do sistema escolhido de *E-cash* Compacto (Seção 3.4.2.1), o provedor pode autenticar as requisições, mas não consegue distinguir qual consumidor realmente pagou pelo uso do serviço. Dinheiro eletrônico, como o real, tem intrinsicamente a propriedade de não

rastreabilidade. Essa condição também se mantém para mensagens sucessivas pagas pelo mesmo consumidor. Assim, os requerimentos propostos (Seção 5.1) de metadados são cumpridos, bem como as dimensões da privacidade do consumidor de ID e de comportamento, referentes às credenciais AAA e MEPs no tempo, são preservadas.

Em relação à possibilidade de rastreabilidade, vale a diferenciar da exculpabilidade, presente neste esquema prático. A exculpabilidade determina, conforme aclarado na Seção 3.4.2, que em um sistema de dinheiro eletrônico, caso um Pagador gaste mais de uma vez determinada moeda ou nota, este se torna passível de identificação. Mais ainda, especificamente no caso do *E-cash* Compacto, a exculpabilidade se estende para não apenas permitir a identificação do gastador duplo, mas como também possibilitar o rastreamento de todas as moedas de propriedade dessa entidade – algoritmo **Trace**, Seção 3.4.2.1. Isso, contudo, não fere o caráter de não rastreabilidade desse esquema prático. A possibilidade de expor a propriedade das credenciais só ocorre no evento específico da tentativa de gasto múltiplo. Um consumidor operando honestamente, ou mesmo um malicioso que não gaste duplamente, nunca será sujeito ao rastreamento. Atenção deve tomada, por conseguinte, porque um consumidor que vise atacar um provedor em uma área diversa da do pagamento, não poderá ser identificado, mesmo com determinação legal. Neste esquema prático, o anonimato do consumidor é absoluto.

5.2.3. Camadas de Anonimato de Conteúdo e de Rede

A camada de anonimato de conteúdo tem como item de interesse os dados das mensagens, impactando diretamente a dimensão de privacidade de conteúdo. Indiretamente, via análise semântica dos dados, esse item de interesse pode afetar também as dimensões de ID, localização e comportamento.

Uma abordagem de anonimato de dados específica deve ser utilizada em cada serviço de nuvem, pois características diferentes das estruturas de dados da lógica implementada demandam técnicas distintas para que o anonimato dos dados conserve a possibilidade de processamento do serviço, enquanto possa manter a privacidade do consumidor. Portanto, antes do envio das mensagens ao provedor, o consumidor deverá tratar as informações potencialmente identificadoras presentes na mensagem. Uma

técnica que pode ser empregada neste caso é o anonimato-k, que de fato é a tecnologia escolhida aqui. Todavia, a customização necessária para a anonimização dos microdados envolvidos via anonimato-k é um procedimento que deve ser realizado de acordo com o serviço sendo considerado.

Para o anonimato de conexão, da camada de rede, responsável pelo item de interesse do endereço IP, que por sua vez afeta as dimensões de privacidade de ID, localização e comportamento, a técnica mais adequada é o emprego de Mix-Nets. Neste esquema prático seleciona-se o Tor para anonimizar os endereços IPs do consumidor, protegendo a privacidade de ID e de localização. Mais ainda, como o Tor emprega a troca dos endereços IP anonimizados com o tempo, a privacidade de comportamento também é resguardada.

5.2.4. Visão Geral

A Tabela 5.1 resume como as tecnologias de anonimato utilizadas pelo esquema prático não rastreável protegem a privacidade do consumidor em relação ao provedor. Especifica-se, nela, o nível de acesso que o TPB (Banco) e o provedor têm aos itens de interesses delimitados pelas camadas e protegidos pelo esquema (Tabela 3.1), em concordância com a avaliação de privacidade apresentada nessa Seção 5.2.

Tabela 5.1 – Visão Geral da Privacidade Estabelecida com o Esquema Prático Não Rastreável

Camada da Estrutura	**Itens de Interesse**	**Tecnologia de Anonimato Empregada**	**Dimensão da Privacidade Protegida**	**Acesso TPB**	**Acesso Provedor**
Contrato	Identificação do Consumidor	Contratos Indiretos Econômicos: *Banco*	ID	Parcial (pagamento)	Não tem
	Endereço do Consumidor	Contratos Indiretos Econômicos: *Banco*	Localização	Parcial (pagamento)	Não tem
	SLA e estimativas de consumo	Contratos Indiretos Econômicos: *Banco*	Comportamento	Parcial (volume)	Não tem
Metadados	Credenciais AAA	Credenciais Anônimas: *E-cash Compacto*	ID	Tem acesso, mas consumidor é anônimo	Tem acesso, mas consumidor é anônimo
	MEPs no tempo	Credenciais Anônimas: *E-cash compacto*	Comportamento	Não tem	Tem acesso, mas consumidor é anônimo

Conteúdo	Dados das Mensagens	Anonimato-k	ID, Localização, Comportamento, Conteúdo	Não tem	Tem acesso, mas consumidor é anônimo
Rede	Endereço IP	Mix-Net: *Tor*	ID, Localização, Comportamento	Não tem ao utilizado na troca de mensagens	Tem acesso, mas consumidor é anônimo

6. VALIDAÇÃO EXPERIMENTAL

Neste Capítulo, leva-se a termo uma validação experimental das contribuições originais da tese. Pretende-se observar e verificar na prática aquilo que foi estabelecido pela análise de privacidade do consumidor SaaS (Seção 2.4), bem como a aplicabilidade da estrutura multicamadas (Capítulo 3), na forma da implementação de um esquema prático (Capítulos 4 e 5). Atestar-se-á, portanto, como os esquemas práticos e a estrutura multicamadas permitem a proteção de privacidade de um consumidor de serviços SaaS em nuvem, efetivamente resguardando as dimensões de privacidade de cada item de interesse presente em cada camada de anonimato (Tabela 3.1).

Para tal, criou-se como um produto efetivo da tese, uma implementação em *software* de um esquema prático. Essa implementação oferece um ambiente para consumo de serviços anonimamente, de acordo com o design teórico de seu respectivo esquema prático. Esse ambiente pode ser utilizado com quaisquer tipos de serviços SaaS em nuvem.

Optou-se pela implementação do esquema prático rastreável. A seleção considerou que este esquema já atende todas as camadas da estrutura multicamadas, podendo, assim, prover a validação buscada, além de ter maior aplicabilidade na computação em nuvem, uma vez que a garantia de rastreabilidade de consumidores mal intencionados é um cenário tecnológico e legal mais atraente para os provedores. Todavia, como a principal diferença entre os esquemas práticos rastreável e não rastreável é a utilização de assinaturas de grupo ou de dinheiro eletrônico, e as credenciais *E-Cash* exercem no contexto de autenticação a mesma função das assinaturas de grupo, uma eventual implementação do esquema prático não rastreável poderá ser gerada com adaptações simples no código da implementação do esquema rastreável.

Assim sendo, o Capítulo se organiza da seguinte forma: na primeira Seção, 6.1, o *software* da implementação é descrito, com suas funcionalidades detalhadas. Na segunda Seção, 6.2, o foco é a validação experimental, com as contribuições originais da tese sendo abordadas pela demonstração de funcionamento da implementação.

Portanto, em 6.2.1, um serviço SaaS exemplo é introduzido e seu funcionamento é utilizado para evidenciar na prática como as dimensões de privacidade são afetadas pela exposição dos itens de interesse do consumidor. Isso mostrará a relevância da análise de privacidade apresentada na Seção 2.4. Na Seção 6.2.2, introduz-se um exemplo de operação da implementação, sendo empregado para acessar o serviço SaaS exemplo da Seção anterior. Verifica-se como o esquema prático concretiza a proteção dos itens de interesse do consumidor. Finalmente, em 6.2.3, as considerações de segurança são mostradas, e a validação das contribuições da estrutura multicamadas e dos esquemas práticos é concluída. Será examinada como a avaliação de privacidade do esquema prático rastreável, da Seção 4.2, é comprovada na prática, com os dados concretos referentes aos itens de interesse sendo trocados anonimamente e protegendo as dimensões de privacidade do consumidor, de acordo com a teoria da estrutura multicamadas e o design de transações do esquema prático, que instancia a estrutura.

6.1. IMPLEMENTAÇÃO DO ESQUEMA PRÁTICO RASTREÁVEL DE CONSUMO ANÔNIMO DE SERVIÇOS SAAS

A implementação em *software* do esquema prático rastreável segue as especificações definidas no design da Seção 4.1.2 e cria uma plataforma de serviços integrados que em conjunto constituem um produto que possibilita o consumo anônimo e rastreável de serviços SaaS em nuvem. De acordo com o esquema prático rastreável do Capítulo 4, a tecnologia selecionada para a obtenção do anonimato rastreável são as assinaturas de grupo, e, nesse design, mais especificadamente, opta-se pelo modelo VLR, de Boneh e Shacham [23], descrito na Seção 3.4.1.1. Em termos das camadas de anonimato, na de contrato emprega-se o TPB, que operará em conjunto com a camada de metadados, exercendo também a função do mestre de grupo do modelo VLR. Para as camadas de anonimato de conteúdo e de rede, o anonimato-k e o Tor serão utilizados, respectivamente.

Dentro da implementação, são criados vários serviços para abarcar o funcionamento das etapas previstas no cenário modelo de consumo de Serviços SaaS, conforme detalhado na Seção 2.2.2, e ao mesmo tempo aderir ao design de transações do esquema prático rastreável de serviços, descrito na Seção 4.1.2.

6.1.1. Descrição

Para a implementação desses serviços e, consequentemente, da implementação como um todo, a escolha foi a utilização da tecnologia SOAP [129], em oposição aos serviços RESTFul [109], uma vez que, como descrito no Capítulo 2, o SOAP consiste em um protocolo focado na troca de informação estruturada XML. Dessa forma, o SOAP estabelece uma fundação modular para a introdução dos esquemas práticos, já que a utilização de um envelope (Figura 2.6) permite o emprego de funcionalidades adicionais por meio do cabeçalho, não influenciando o corpo da mensagem ou mesmo outros cabeçalhos já presentes. Exemplos são o WS-ReliableMessaging [91], para a entrega confiável de mensagens, e o próprio WS-Security [92], para a segurança das mensagens.

Adicionalmente, além do consumidor e do provedor, conforme estabelecido pelo Capítulo 4, do esquema prático rastreável, faz-se aqui uso do TPB como entidade a intermediar o consumo de serviços e a possibilitar o anonimato.

Ao todo, compõem a implementação do esquema prático rastreável, quatro *web services* e dois agentes de serviços[19], organizados segundo o seguinte ambiente:

- ***Web Service Publisher***: Disponibilizado pelo TPB com a função de permitir que um provedor publique um serviço a ser oferecido para consumo anônimo.

- ***Web Service Subscriber***: Também disponibilizado pelo TPB, mas com a função de permitir que um consumidor se subscreva para o consumo de um determinado serviço de consumo anônimo. Corresponde à assinatura do contrato indireto.

- **Agentes de serviço *Consumer-Signer* e *Provider-Verifier***: São agentes de serviço em *software* que se conectam a aplicação do consumidor e à do provedor e intermediam o consumo anônimo de serviço propriamente dito, assinando anonimamente as mensagens de requisição do consumidor e possibilitando a

[19] Um agente de serviço se difere de um *web service* pelo fato de que ele sempre age em nome de outro serviço. Agentes de serviço também não têm contratos, e os consumidores não precisam acessar esses programas diretamente. Os agentes de serviço são empregados tipicamente para interceptar mensagens sendo trocadas por outros serviços e para exercer algum processamento intermediário, como, por exemplo, autenticação de usuários, validação, conversão e balanceamento de carga [50].

verificação das mesmas no provedor, antes de permitir que o provedor processe a mensagem de consumo.

- ***Web Service Depositer***: Oferecido pelo TPB ao provedor, tem a função de possibilitar ao provedor relatar ao TPB todo consumo anônimo de serviço efetivamente realizado, capitalizando a capacidade de mestre do grupo de assinaturas de grupo do TPB para que esse tarife, fature, debite os consumidores e, finalmente, pague o provedor pelos serviços prestados.

- ***Web Service Invoicer***: Serviço também oferecido pelo TPB, mas que é direcionado ao consumidores, possibilitando a emissão de faturas detalhadas dos consumos de serviços realizados pelo próprio consumidor.

6.1.2. Produto

Os *web services* e os agentes de serviços descritos acima foram implementados na linguagem de programação JAVA e compilados em um produto a ser disponibilizado em três aplicativos especificamente desenhados para permitir aos consumidores, provedor e TPB estabelecerem um ambiente para consumo anônimo de serviços em um modelo SaaS.

Os aplicativos são: **CONSUMER-APP.jar**, **PROVIDER-APP.jar** e o **TPB-APP.jar**. O **CONSUMER-APP.jar** oferece ao consumidor uma interface para selecionar o serviço SaaS anônimo a ser consumido, se subscrever a ele, além de obter faturas. No nível operacional, o **CONSUMER-APP.jar** contém o agente de serviço *Consumer-Signer*, que intercepta as mensagens direcionadas ao serviço selecionado e as anonimiza do ponto de vista do nível de troca de mensagens. Basicamente, então, o **CONSUMER-APP.jar** é responsável por prover uma interface de acesso aos *web services Subscriber* e *Invoicer*, residentes no TPB, e por configurar e operar o agente de serviço *Consumer-Signer*.

O **PROVIDER.jar** permite ao provedor escolher um serviço SaaS para ser publicado para acesso anônimo no TPB, determinar a quantidade de serviço que poderá oferecer (créditos de serviço), e, após os consumos serem efetivados, enviar ao TPB o relatório

para que o TPB possa tarifar, faturar e realizar o pagamento para o provedor. Operacionalmente também contém um agente de serviço, que neste caso é o *Provider-Verifier*, responsável por verificar e registrar os consumos anônimos de serviço. Portanto, o **PROVIDER.jar** disponibiliza uma interface de acesso aos *web services Publisher* e *Depositer* no TPB, e configura e opera o agente de serviço *Provider-Verifier*.

Já o **TPB-APP.jar** implementa os *web services Publisher*, *Subscriber*, *Depositer* e *Invoicer*, operacionalizando o funcionamento desses e permitindo sua gerência. A Figura 6.1 permite a visualização do ambiente instituído pelo produto.

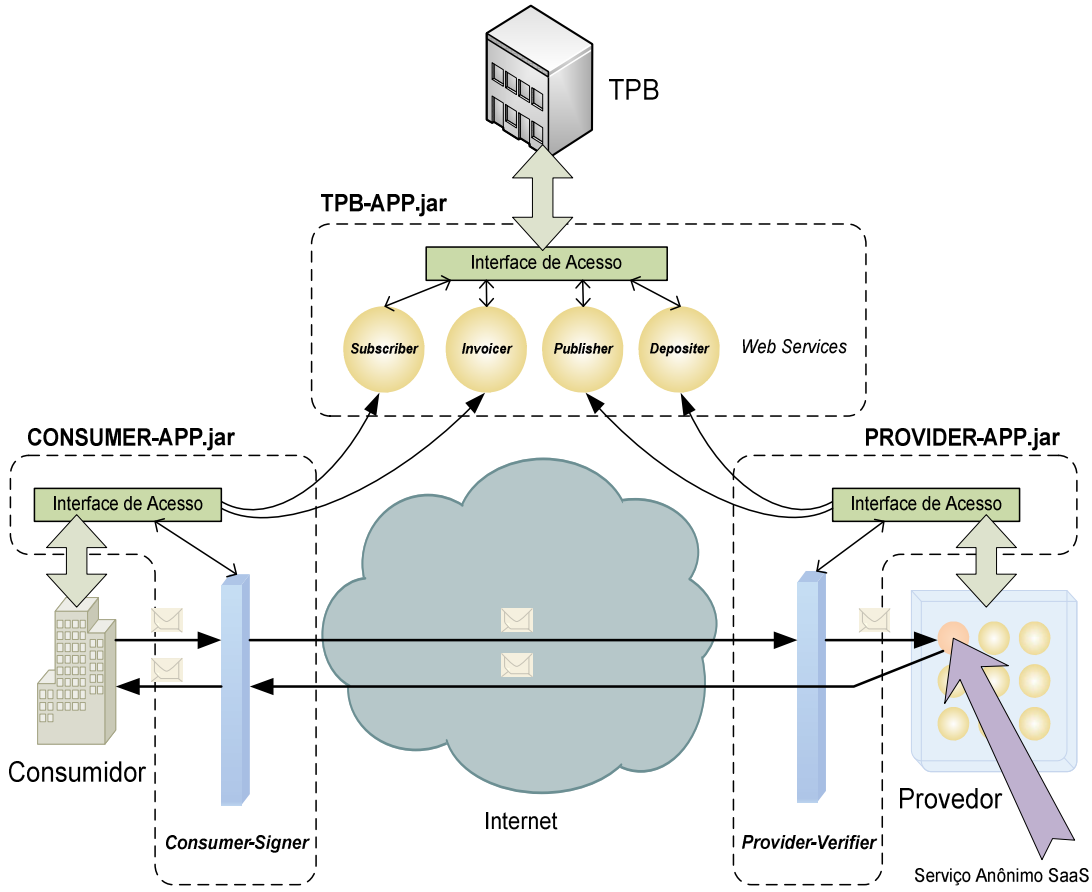

Figura 6.1 – Produto para Consumo Anônimo de Serviços SaaS

Outro ponto relevante a ser esclarecido é que, apesar do esquema prático rastreável suportar os modelos de precificação pré-pago e de assinatura com limitação (Seção

4.1.1), em favor da simplicidade, este produto operará no paradigma pré-pago, e, portanto, o consumidor adquirirá os créditos para consumo previamente às requisições de serviço e não haverá período de faturamento. O provedor controlará o momento de submeter os créditos utilizados para o TPB, determinando a execução da tarifação, faturamento e pagamento.

A seguir, a operação do produto será apresentada e as funções exercidas pelos *web services* e agentes de serviços serão descritas, separadas conforme o papel que exercem, que são: **Contratos e Credenciamento**, **Consumo de Serviço Anônimo** e **Pagamento**.

6.1.3. Contratos e Credenciamento

Responsáveis pela função estabelecimento de contratos e de credenciamento, os *web services Publisher* e *Subscriber* respectivamente permitem ao provedor publicar um serviço anônimo e ao consumidor se subscrever a esse, sem se identificar e revelar sua identidade ao provedor. A Figura 6.2 permite a visualização do contexto desses dois *web services*.

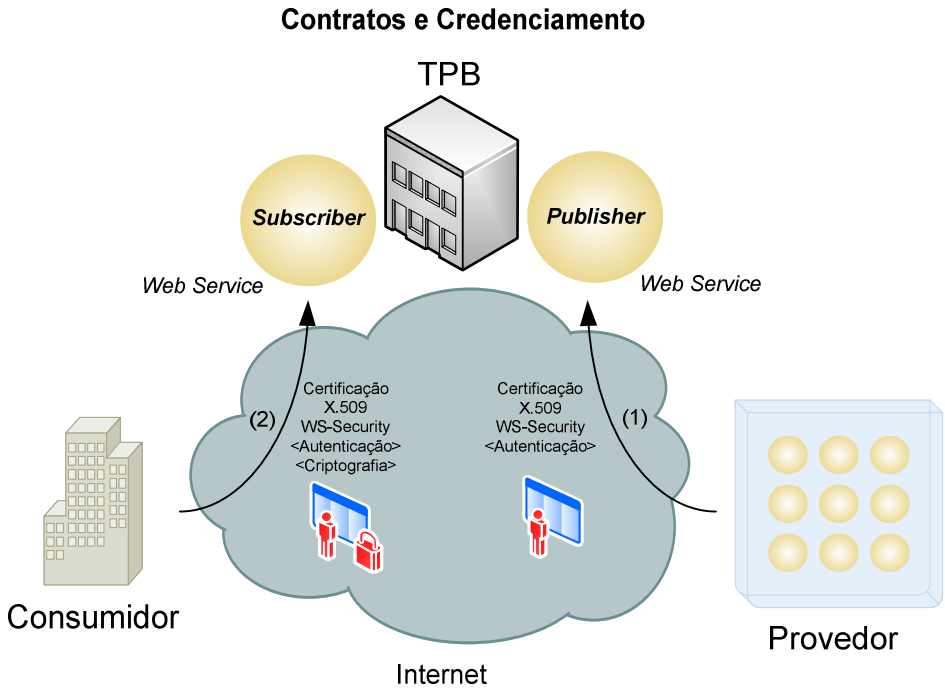

Figura 6.2 – Função de Contratos e Credenciamento

Primeiramente descrever-se-á o *web service Publisher*, pois é nele que se iniciam as interações que efetivamente possibilitarão o consumo anônimo de serviços SaaS.

6.1.3.1. *Publisher*

O *web service Publisher* corresponde à etapa 1 da Figura 2.7, referente à publicação do serviço. E, em relação ao esquema prático rastreável, mais especificadamente ao design de transações, este *web service* corresponde às etapas (1a), (1b) e (1c) da Figura 4.1, referentes ao estabelecimento do contrato indireto entre provedor e TPB, emissão de créditos de consumo e recebimento da chave pública de grupo por parte do provedor.

Assim como todos os *web services* dessa implementação, o *Publisher* é descrito na forma de um contrato WSDL. Um contrato WSDL é um documento escrito em XML [128] e que descreve a funcionalidade oferecida por um *web service*. No APÊNDICE A, apresenta-se o contrato WSDL para o *web service Publisher*.

Como se pode verificar pelo ***publisher.wsdl***, esse *web service* oferece basicamente duas operações: *publishservice* e *addservicecredit*. A Figura 6.3 apresenta uma representação gráfica do contrato ***publisher.wsdl***.

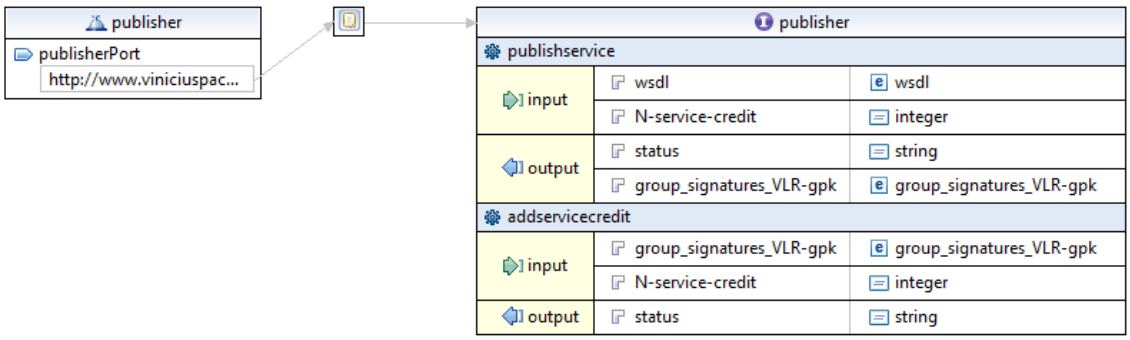

Figura 6.3 – Representação Gráfica de ***publisher.wsdl***.

Em resumo, no TPB, a operação *publishservice* recebe um contrato genérico de um provedor, no formato WSDL, para ser intermediado pelo TPB, além da quantidade de créditos que o provedor considera como adequados para o bom provisionamento desse serviço, de acordo com sua capacidade de recursos. Como resposta, o provedor recebe

um status da operação (OK ou NÃO OK) e a chave pública do grupo, para verificação das assinaturas a serem recebidas nas requisições anônimas de consumo desse serviço. Essa chave pública de grupo também passa a ser o identificador do serviço, a ser empregado para referenciar unicamente o mesmo para as três entidades (TPB, provedor e consumidor).

Internamente, no TPB, as etapas relacionadas com essa operação são:

1. Recebe as entradas (WSDL + número de créditos) do provedor;

    a. Gera as chaves privadas para cada consumidor futuro (vetor ***gsk***), a chave pública do grupo (***gpk***) e a chave secreta para rastreamento (vetor ***grt***). Estas atividades são executadas pelo algoritmo **Keygen(n)**, do sistema VLR, descrito na Seção 3.4.1.1;

    b. Cria os créditos de serviço de acordo com o número de créditos selecionado pelo provedor (entrada). A criação pega como entrada apenas o número de créditos a ser criado e gera números aleatórios como créditos;

    c. Atribui ***gpk*** como o SERVICE ID e publica o WSDL recebido em um diretório UDDI [90], conforme descrito no cenário típico SaaS, Figura 2.6;

    d. Cria um banco de dados local DB-SERVICE, com o relacionamento do SERVICE ID, as respectivas chaves privadas e de revogação e os créditos referentes ao serviço, momentaneamente não alocados para nenhum consumidor;

2. Retorna como saída para o provedor o sucesso da operação, juntamente com a chave pública do grupo ***gpk*** para que o provedor possa verificar as assinaturas anônimas a serem recebidas.

Já a segunda operação do *Publisher*, a *addservicecredit*, é acionada pelo provedor em um momento após a *publishservice*, quando o próprio provedor detecta que os créditos

de serviço necessitam ser ampliados. Dessa forma, a entrada da operação para o TPB é o SERVICE ID (***gpk***) e o número de créditos a ser adicionado ao serviço. A saída é apenas o status da operação (OK ou NÃO OK).

O TPB realiza internamente os seguintes procedimentos:

1. Recebe o SERVICE ID e o número de créditos a ser adicionado ao serviço;
    a. Gera o número de créditos adicionais e os aleatoriza;
    b. Atualiza o banco de dados DB-SERVICE, adicionando os créditos de serviço para o SERVICE ID específico;
2. Retorna como saída para o provedor o status da operação realizada.

As comunicações de rede referentes às operações presentes nesse *web service Publisher* não precisam ser criptografadas, uma vez que nenhuma informação confidencial é trocada. No entanto, é necessário possibilitar a identificação confiável do provedor para o TPB, e, portanto, a utilização de certificação digital é recomendada. Para isso, basta incorporar às mensagens trocadas o padrão ITU-T para certificação digital X.509 [64]. Nesse sentido, como os *web services* estão implementados em SOAP, a maneira para introduzir essa certificação digital de forma modular é a utilização de um cabeçalho WS-Security.

O WS-Security [92] é uma extensão ao protocolo SOAP, publicada pelo OASIS, e tem como objetivo introduzir segurança para *web services*. Descrevem-se em sua especificação três mecanismos principais: como assinar mensagens SOAP, visando à integridade da informação; como cifrar mensagens SOAP, objetivando confidencialidade; e, finalmente, como anexar *tokens* de segurança para atestar a identidade do remetente, como, por exemplo, o UsernameToken Profile [93], baseado no emprego de apenas usuário e senha. Aqui, entretanto, optar-se-á pelo emprego de um *token* do tipo X.509, definido pelo X509Token Profile [95], onde descreve-se o emprego de certificação digital dentro de uma mensagem SOAP com WS-Security. A Figura 6.2 explicita a utilização da certificação digital para o *Publisher*.

6.1.3.2. *Subscriber*

Agora, passa-se à apresentação do *web service Subscriber*, que também faz parte da função de contratos e credenciamento da implementação.

O *web service Subscriber* corresponde à etapa 3 da Figura 2.7, referente ao contrato formal do serviço. A etapa 2 da Figura 2.7, relativa à descoberta do serviço, é realizada externamente à implementação, empregando uma consulta ao diretório UDDI onde se encontra o serviço publicado. Esse diretório UDDI pode ou não ser disponibilizado pelo próprio TPB.

Em relação ao esquema prático rastreável, mais especificadamente ao design de transações, este *web service* corresponde às etapas (1d) e (1e) da Figura 4.1, referentes ao estabelecimento do contrato indireto entre consumidor e TPB e ao recebimento por parte do consumidor de sua chave secreta e a chave pública para assinatura (algoritmo **Sign** – Seção 3.4.1.1) das mensagens, além de uma coleção de créditos aleatórios (Seção 4.1.1) para o consumo de serviço. O WSDL do *Subscriber* está apresentado no APÊNDICE B.

O *web service* ***subscriber.wsdl*** comporta três operações: a *subscribeservice*, a *buyservicecredits* e a *reportabuse*. A Figura 6.4 permite a visualização gráfica do ***subscriber.wsdl.***

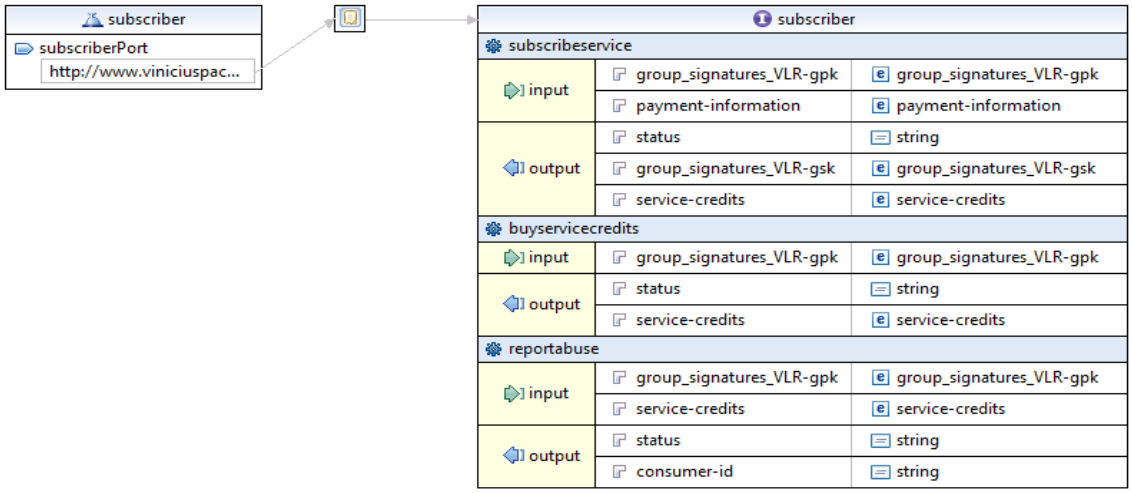

Figura 6.4 – Representação Gráfica de ***subscriber.wsdl***.

Para a operação *subscribeservice*, um consumidor que tenha previamente descoberto o serviço desejado no diretório UDDI pode, de posse do SERVICE ID (chave pública do grupo) descrito no WSDL e de sua informação para pagamento, enviar uma requisição ao TPB para se subscrever ao serviço. O TPB irá retornar ao consumidor um status da operação (OK ou NÃO OK), juntamente com a chave secreta deste consumidor e um conjunto de créditos de serviço a serem utilizados pelo próprio consumidor ao contatar o provedor.

No TPB, as etapas relacionadas com essa operação são:

1. Recebe como entrada o SERVICE ID do serviço e a informação de pagamento do consumidor.

    a. Associa internamente uma chave ***gsk*[*i*]** ao CONSUMER ID e à informação de pagamento do consumidor e insere esses dados em um banco de dados interno denominado DB-CONSUMERS. O CONSUMER ID do consumidor é obtido pela informação enviada no certificado digital X.509 presente na requisição da operação, conforme a Figura 6.2.

    b. Associa uma coleção de créditos de serviço aleatórios do serviço à chave ***gsk*[*i*]** do consumidor e atualiza essa informação no banco de dados DB-SERVICE. O conjunto total de créditos de serviço, do qual esta coleção é tirada, consta em DB-SERVICE e foi criado ou atualizado pelas operações *publishservice* e *addservicecredits* do *web service Publisher*, respectivamente.

2. Retorna ao consumidor o status da operação, juntamente com a chave secreta ***gsk*[*i*]** e seus novos créditos de serviço.

A segunda operação, chamada *buyservicecredits*, tem o propósito de permitir ao consumidor comprar novos créditos de consumo. Para isso, o consumidor envia ao TPB o SERVICE ID do serviço e o TPB retorna os novos créditos de serviço para consumo.

Dentro do TPB, os seguintes passos são tomados dentro da operação:

1. Recebe como entrada o SERVICE ID do serviço específico para o qual o consumidor deseja obter mais créditos;

    a. De posse do CONSUMER ID do consumidor, obtido na certificação digital da mensagem de requisição, e do SERVICE ID do serviço, o TPB obtém a chave secreta ***gsk[i]*** do consumidor em DB-CONSUMERS e acessa a DB-SERVICE correspondente para alocar do conjunto de créditos de serviço livres (ainda não alocados para outro consumidor) para esse consumidor;

    b. Atualiza-se a DB-SERVICE para refletir a nova alocação de créditos;

2. TPB retorna para o consumidor seus novos créditos de serviço, juntamente com o status da operação.

A última operação do *web service Subscriber* é chamada *reportabuse*. O objetivo dela é permitir a um consumidor que tenha créditos de serviço recusados pelo provedor reportar o problema e ter um rastreamento efetivado do possível consumidor malicioso. O TPB emprega seus privilégios de mestre do grupo VLR (ver Seção 3.4.1.1) para conseguir efetivar o rastreamento do consumidor associado com a assinatura recebida pelo provedor juntamente com o crédito de consumo suspeito.

A sequência de operações realizada pelo TPB é:

1. Recebe como entrada do consumidor o SERVICE ID e os seus créditos de serviço que foram recusados pelo provedor.

    a. Com o CONSUMER ID obtido da certificação digital e o SERVICE ID recebido, o TPB acessa a base correta DB-CONSUMERS e obtém a chave secreta ***gsk[i]*** do consumidor;

    b. Com a chave secreta do consumidor, o TPB consulta DB-SERVICE para verificar se os créditos reportados como de posse do consumidor

realmente estão associados a ele. Caso não estejam, um status de erro é retornado.

c. Em seguida uma base de dados chamada DB-SERVICE-BILLING-TPB é acessada. Essa base de dados é criada pelo *web service Depositer* (a ser apresentado posteriormente) e contém a informação de consumo de créditos enviada pelo provedor ao TPB. Todavia, essa informação está rastreada (***gsk***[***i***] com respectivos créditos utilizados), ao contrário daquela o provedor enviou (assinaturas anônimas com respectivos créditos utilizados), pois o TPB executa o algoritmo **Trace** (Seção 3.4.1.1) nas informações recebidas para poder depois efetivar o pagamento do provedor e a cobrança dos consumidores.

d. Na DB-SERVICE-BILLING-TPB, o TPB identifica quem utilizou os créditos reportados como recusados. Com a ***gsk*[*i*]** de quem utilizou os créditos, o TPB acessa a DB-SERVICE para verificar com que consumidor tais créditos estão associados. Caso a associação esteja violada, o TPB registra a ***grt*[*i*]** do infrator (obtida em DB-SERVICE) e obtém o respectivo CONSUMER ID em DB-CONSUMERS. O registro da infração é atualizado em uma base de dados própria chamada DB-ABUSERS-TPB, onde a respectiva chave de revogação ***grt*[*i*]** é inserida.

2. O TPB retorna para o consumidor o status da operação (ABUSE REGISTERED ou NO ABUSE REGISTERED), juntamente com o CONSUMER ID do infrator, caso o mau comportamento tenha sido comprovado.

Para todas as operações do *Subscriber*, além apenas do envio de um certificado digital para autenticação, faz-se necessário cifrar as mensagens sendo trocadas, pois em todas as operações desse *web service* informações confidenciais são trocadas. Desde a chave secreta ***gsk*[*i*]** do consumidor em *subscribeservice*, dos créditos a serem comprados pelo consumidor em *buyservicecredits* até o CONSUMER ID de um infrator em *reportabuse*, todos esses dados não devem ser acessados por terceiros.

Nesse sentido, o próprio X509Token Profile [95] utilizado no *Publisher* para autenticação (*token* X.509), pode ser utilizado para cifrar mensagens dentro do WS-Security. Assim sendo, além da autenticação, o *Subscriber* também irá fazer uso da criptografia baseada no certificado digital. A Figura 6.2 permite a visualização da utilização tanto da autenticação como da cifração dentro do *Subscriber*.

6.1.4. Consumo de Serviço Anônimo

Após o provedor publicar o serviço (WSDL enviado ao TPB) e o consumidor se subscrever a esse serviço, o consumidor passa a poder acessar diretamente o provedor, mas mantendo seu anonimato e, consequentemente, sua privacidade.

Para que o consumo anônimo de serviço possa acontecer, são inseridos dois agentes de serviço, um no consumidor e outro no provedor. O agente de serviço no consumidor, denominado *Consumer-Signer*, conecta-se à aplicação e recebe uma mensagem SOAP para ser anonimizada e destinada ao serviço selecionado. Após processá-la, o agente adiciona um cabeçalho WS-Security e envia a mensagem para o agente de serviço do provedor, o *Provider-Verifier*. Esse verifica se a mensagem é válida e repassa a mesma para o *web service* específico do provedor para que o serviço possa ser processado. Em seguida, o provedor responde com o resultado do processamento para o consumidor. A Figura 6.5 apresenta o funcionamento da função do consumo de serviço anônimo da implementação, contextualizando a operação dos dois agentes de serviço.

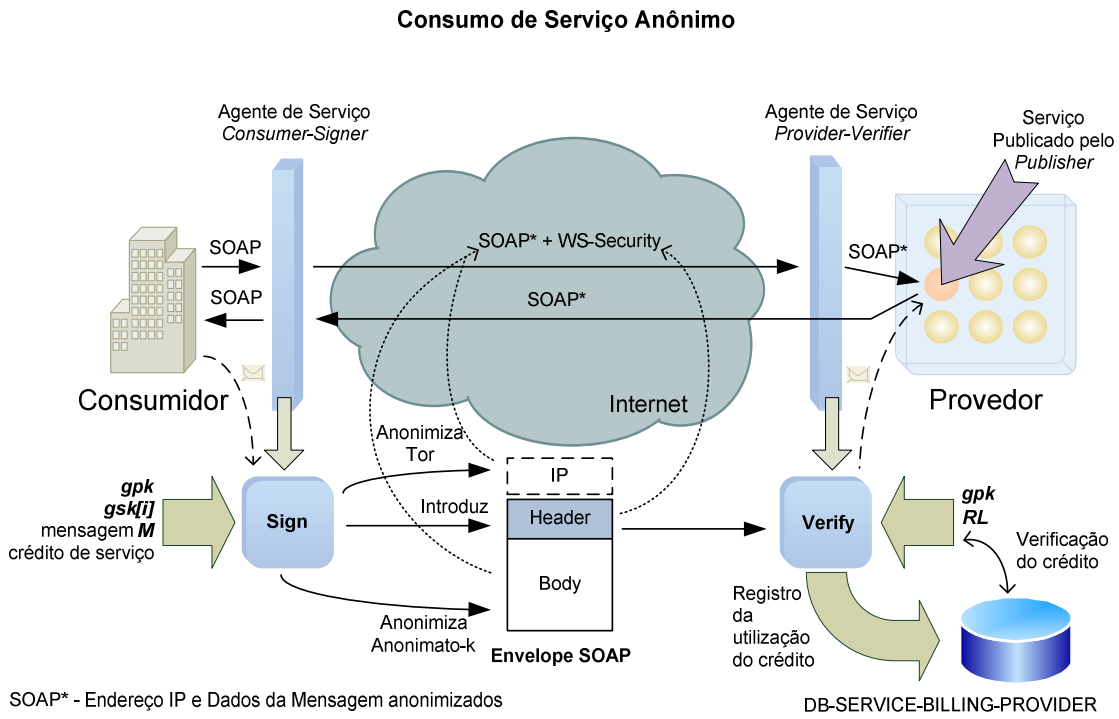

Figura 6.5 – Função de Consumo de Serviço Anônimo

6.1.4.1. *Consumer-Signer*

Passa-se, de início, a descrição do agente de serviço *Consumer-Signer*. Em termos da correspondência com as etapas do consumo típico de serviços SaaS, este agente de serviço se relaciona com a parte do consumidor da etapa 4 da Figura 2.7. E, de acordo com o design do esquema prático rastreável, com a etapa (2a) da Figura 4.1.

Para o *Consumer-Signer*, as entradas são o SERVICE ID (***gpk***), a chave secreta do consumidor (***gsk*[*i*]**), um crédito de serviço adquirido válido (não utilizado) pelo consumidor e a própria mensagem a ser enviada. A saída é um envelope SOAP, com um cabeçalho WS-Security adicionado, onde consta a assinatura anônima de grupo (***σ*** – Seção 3.4.1.1) e o crédito de serviço (camada de anonimato de metadados), além do corpo da mensagem anonimizado (camada de anonimato de conteúdo) e do IP do cabeçalho IP anonimizado (camada de anonimato de rede).

As operações internas deste agente de serviço, no consumidor, são:

1. Recebe da aplicação a mensagem SOAP a ser transmitida e busca internamente no consumidor as entradas: ***gpk***, ***gsk*[*i*]** e o crédito de serviço.

    a. Utilizando uma tecnologia de anonimato de dados, neste caso anonimato-k[20], cria identificadores numéricos aleatórios ou sumprime os elementos XML da requisição, e os substitui na mensagem, guardando em uma base de dados local a correspondência da associação, para que, na volta, a resposta possa ser reconstruída com os valores originais.

    b. Calcula-se o *hash* da mensagem original e obtêm-se o parâmetro ***M*** do algoritmo **Sign** (Seção 3.4.1.1).

    c. De posse de ***M***, ***gpk*** e ***gsk*[*i*]**, executa-se o algoritmo **Sign** (Seção 3.4.1.1) e tem-se como saída a assinatura de grupo $\boldsymbol{\sigma}$, que passa a ser a credencial anônima para acesso ao serviço.

    d. Cria-se um cabeçalho WS-Security para a mensagem SOAP, já anonimizada em seu conteúdo, e incorpora-se $\boldsymbol{\sigma}$ e o crédito de serviço a ele**.** Caso não existam mais créditos de serviço a serem utilizados, executa-se a operação *buyservicecredits* do *web service Subscriber*.

    e. Acessa-se uma tecnologia de anonimato de conexão da camada de rede, neste caso o Tor[21], e obtém-se um IP anônimo para este consumo anônimo específico do serviço.

2. Envia-se para o agente de serviço do provedor a mensagem de requisição com o cabeçalho WS-Security com a credencial anônima e o crédito de serviço, juntamente com o corpo da mensagem SOAP anonimizado, além do endereço IP do cabeçalho IP também anonimizado.

[20] Utiliza-se nesta implementação o anonimato-k, pois permite que os dados que possam identificar unicamente um registro sejam suprimidos ou generalizados. Aqui, apenas o campo que é necessário para o processamento do serviço será mantido, enquanto os outros serão suprimidos ou substituídos por valores aleatórios, sem significado semântico.

[21] Utiliza-se nesta implementação o Tor, pois ele permite a anonimização do endereço IP de origem introduzindo pouco retardo e *jitter* na transação.

6.1.4.2. *Provider-Verifier*

Segue-se, agora, com o detalhamento do agente de serviço do provedor, o *Provider-Verifier*. Esse agente se relaciona com a parte do provedor da etapa 4 da Figura 2.7, e com a etapa (2b) da Figura 4.1.

Para o *Provider-Verifier*, as entradas são a mensagem recebida do *Consumer-Signer*, a chave pública do grupo ***gpk***, obtida pela correspondência ao SERVICE ID do serviço sendo endereçado, o parâmetro ***RL*** de consumidores com anonimato revogado, e o acesso à base de dados local DB-SERVICE-BILLING-PROVIDER para a checagem do crédito sendo utilizado. A saída é a mensagem autenticada para ser entregue ao *web service* publicado e o registro de utilização do crédito de serviço na base DB-SERVICE-BILLING-PROVIDER.

Internamente, no provedor, as operações abaixo são executadas:

1. O agente de serviço recebe uma mensagem SOAP endereçada a um serviço publicado pelo provedor. De acordo com o endereço sendo buscado, obtém-se internamente a chave ***gpk*** (SERVICE ID) necessária para a verificação da assinatura e o parâmetro ***RL***, oriundo de uma base de dados local chamada DB-ABUSERS-PROVIDER. A base DB-ABUSERS-PROVIDER contém o subconjunto de consumidores com o anonimato revogado, sendo, consequentemente, o próprio parâmetro ***RL***. Registra-se como entrada também o acesso à base local DB-SERVICE-BILLING-PROVIDER relativa ao serviço em questão.

    a. De dentro do cabeçalho WS-Security recebido, retira-se a assinatura de grupo anônima $\boldsymbol{\sigma}$;

    b. Calcula-se o *hash* da mensagem recebida e, assim, estabelece-se o parâmetro ***M*** do algoritmo **Verify** (Seção 3.4.1.1);

    c. De posse de ***gpk***, ***M, RL*** e $\boldsymbol{\sigma}$, executa-se o algoritmo **Verify** e, assim, autentica-se anonimamente o consumidor;

d. Agora, acessa-se DB-SERVICE-BILLING-PROVIDER para verificar se o crédito de serviço presente no cabeçalho WS-Security é valido e ainda não foi utilizado. DB-SERVICE-BILLING-PROVIDER é uma base de dados aonde todo consumo de serviço referente a este SERVICE ID é registrado. Cada registro é composto do $\boldsymbol{M}$ calculado, da $\boldsymbol{\sigma}$ verificada, do crédito de serviço utilizado e de um *timestamp* do evento. Caso não haja registro de utilização do crédito de serviço sendo oferecido, a requisição é aceita.

2. Como saída, entrega-se a mensagem SOAP ao *web service* de destino, já com o cabeçalho processado e retirado, e realiza-se a atualização de DB-SERVICE-BILLING-PROVIDER com $\boldsymbol{M}$, $\boldsymbol{\sigma}$, crédito de serviço e *timestamp* desta requisição.

Nenhuma ferramenta de autenticação ou criptografia tradicionais, como as utilizadas pelos *web services Publisher* e *Subscriber* são empregadas aqui para os agentes de serviço, pois a privacidade a ser proporcionada ao consumidor é baseada no anonimato provido pela estrutura multicamadas de anonimato e do respectivo esquema prático rastreável instanciado e aplicado nesta implementação. O cabeçalho WS-Security aqui utilizado é ajustado para abarcar os parâmetros sendo transmitidos ($\boldsymbol{\sigma}$ e crédito de serviço). Na demonstração de funcionamento dessa implementação serão detalhados os ajustes imprimidos no WS-Security.

6.1.5. Pagamento

Para a função de pagamento dentro da implementação, são empregados os *web services Depositer* e *Invoicer*. Respectivamente, eles possibilitam ao provedor depositar junto ao TPB os créditos de consumo para receber a compensação financeira devida e ao consumidor requerer do TPB uma fatura atualizada de seus consumos de serviço anônimos. A Figura 6.6 mostra a função de pagamento da implementação.

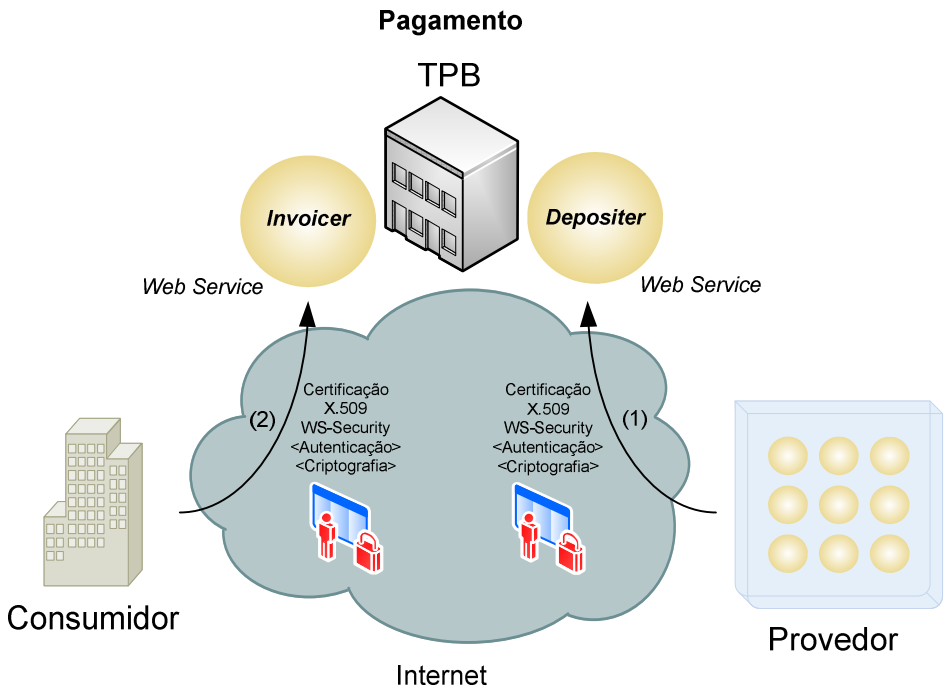

Figura 6.6 – Função de Pagamento

6.1.5.1. *Depositer*

Na função de pagamento, o primeiro *web service* a ser descrito será o *Depositer*. Ele corresponde a parte da etapa 5 da Figura 2.7, e às etapas (3a), (4a) e (4b) da Figura 4.1.

No APÊNDICE C, apresenta-se o arquivo WSDL do *Depositer*, denominado ***depositer.wsdl***, e, na Figura 6.7, mostra-se a representação gráfica do *web service*. Em ambos, verifica-se que *Depositer* disponibiliza apenas uma operação: a *depositcredits*.

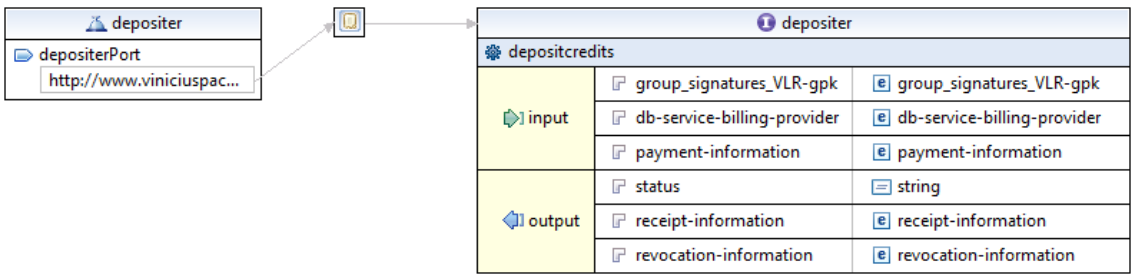

Figura 6.7 – Representação Gráfica de ***depositer.wsdl***

Para a operação *despositcredits*, o objetivo é possibilitar ao provedor depositar os créditos apresentados pelos consumidores anônimos e receber o pagamento devido pela prestação do serviço. A entrada da operação é o SERVICE ID do serviço, a base de dados do provedor chamada DB-SERVICE-BILLING-PROVIDER e a informação de pagamento para que o TPB possa creditar a conta do provedor. A saída é um recibo de crédito para o provedor, juntamente com uma mensagem de sucesso da operação, caso não tenha havido problemas. No evento do TPB ter detectado alguma utilização de crédito imprópria por algum consumidor, o TPB também irá enviar ou atualizar a lista do provedor de usuários revogados para esse serviço (parâmetro ***RL***).

No TPB, as operações envolvidas são:

1. Recebe como entrada o SERVICE ID, a respectiva base de dados do provedor DB-SERVICE-BILLING-PROVIDER e as informações de pagamento para creditar o provedor pelos consumos.

    a. Como a DB-SERVICE-BILLING-PROVIDER contém todos os registros de $\boldsymbol{M}$, $\boldsymbol{\sigma}$, crédito de serviço e *timestamp* referentes aos consumos de serviço desse SERVICE ID, o TPB, usando então seu privilégio de mestre do grupo VLR, executa o algoritmo **Trace** (Seção 3.4.1.1) em todas as assinaturas presentes e obtém a chave secreta $\boldsymbol{gsk[i]}$ relativa a cada consumo. Dessa forma, cria-se (se for a primeira vez que a operação esteja sendo executada) a base de dados local DB-SERVICE-BILLING-TPB, com os mesmos parâmetros para cada registro da DB-SERVICE-BILLING-PROVIDER ($\boldsymbol{M}$, $\boldsymbol{\sigma}$, crédito de serviço e *timestamp*). Contudo, ao invés de utilizar a assinatura anônima $\boldsymbol{\sigma}$, registra-se $\boldsymbol{gsk[i]}$.

    b. Em seguida, percorre-se a DB-SERVICE-BILLING-TPB e verifica-se junto à base DB-SERVICE se para cada crédito de serviço utilizado, a chave $\boldsymbol{gsk[i]}$ correta foi usada. Caso existam infrações (consumidor utilizou crédito de outro consumidor ou crédito não alocado), o TPB

adiciona a ***grt[i]*** do infrator (obtida em DB-SERVICE) em DB-ABUSERS-TPB.

c. Com todos os registros de consumo identificados em DB-SERVICE-BILLING-TPB, o TPB acessa DB-CONSUMERS e para todas as utilizações de crédito por cada ***gsk[i]*** específica, gera internamente uma fatura individualizada e debita a informação de pagamento correspondente.

d. Após o débito de todos os consumidores, o TPB credita o provedor, utilizando a informação de pagamento recebida e gera um recibo da operação.

2. Entrega-se, como saída, uma mensagem de sucesso, juntamente com o recibo gerado e o novo parâmetro ***RL*** a ser utilizado pelo provedor (base DB-ABUSERS-TPB atualizada). O parâmetro ***RL*** se tornará a base local do provedor DB-ABUSERS-PROVIDER.

6.1.5.2. *Invoicer*

Finalmente, apresenta-se agora o último *web service* da implementação, nomeado *Invoicer*.

O *web service Invoicer* se relaciona com a etapa 5 da Figura 2.7, e com a etapa (3b) da Figura 4.1, sendo que tem a função básica de permitir ao consumidor consultar seu extrato de consumo de serviço. O APÊNDICE D e a Figura 6.8 apresentam o arquivo WSDL do *Invoicer* e a representação gráfica do mesmo, respectivamente.

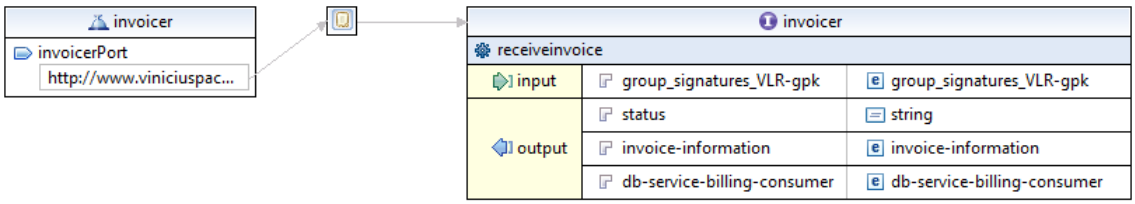

Figura 6.8 – Representação Gráfica de ***invoicer.wsdl***

Como visto acima, o *Invoicer* tem apenas uma operação definida: *receiveinvoice*. Nela, o TPB recebe como entrada de um consumidor a identificação do serviço subscrito (SERVICE ID). De posse disso, após processamento, retorna para o consumidor um relatório de seus consumos anônimos de serviço, juntamente com um extrato referente ao valor correspondente pelas transações.

Para o TPB, as etapas percorridas são:

1. Receber como entrada a chave pública do grupo ***gpk*** (SERVICE ID);

    a. De posse do CONSUMER ID, acessa-se a DB-CONSUMERS e obtém a respectiva ***gsk[i]***. O CONSUMER ID é obtido pela certificação digital da requisição.

    b. Com a ***gsk[i]***, percorre-se DB-SERVICE-BILLING-TPB e registra-se à parte todos os consumos realizados por essa chave. Separa-se esse relatório para envio como saída da operação.

    c. Compõe-se, também, um extrato de quanto corresponde o consumo presente no relatório.

2. Retorna-se ao consumidor uma mensagem de status da operação, o relatório completo de todas as requisições de consumo de serviço anônimas e um extrato financeiro correspondente aos consumos registrados.

Semelhantemente ao *web service Subscriber*, o *Depositer* e o *Invoicer* trabalham com a troca de informações confidenciais, como o relatório completo de consumo a ser enviado pelo provedor em *depositcredits* e a informação de consumo atrelada a um consumidor específico em *receiveinvoice*. Destarte, além da certificação digital, também se promove aqui a cifração das mensagens a serem trocadas. Mais uma vez, faz-se uso do X509Token Profile [95] para cifrar e certificar as mensagens dentro do WS-Security. Na Figura 6.6, visualiza-se a utilização tanto da autenticação como da cifração dentro do *Depositer* e do *Invoicer*.

6.2. DEMONSTRAÇÃO DE FUNCIONAMENTO

Com a implementação apresentada, pode-se agora demonstrar o funcionamento do *software*. A verificação da operação do ambiente possibilitará a comprovação prática da eficácia da solução proposta pela estrutura multicamadas de anonimato e pelo esquema prático rastreável para consumo anônimo de serviços SaaS. Constatar-se-á, também, como a avaliação de privacidade do esquema prático rastreável apresentada na Seção 4.2 tem validade em um paradigma concreto.

Para isso, primeiramente será introduzido um serviço SaaS exemplo a ser oferecido por um provedor. Este exemplo servirá para atestar a validade da análise de privacidade do consumidor SaaS, primeira contribuição original da tese, na Seção 2.4.

Na sequência, utilizar-se-á o produto da implementação para se prover o acesso anônimo ao serviço exemplo apresentado. Assim, verificar-se-á, na prática, como os itens de interesse do consumidor são representados e protegidos pelos seus respectivos dados concretos.

No final, na Seção 6.2.3, as considerações de segurança do funcionamento da implementação serão apresentadas e poderá se comprovar nesse ambiente as afirmações da avaliação de privacidade do esquema prático rastreável da Seção 4.2, finalizando, portanto, a validação das contribuições originais da tese, na forma da estrutura conceitual multicamadas e do instanciamento dessa, pelos esquema práticos.

6.2.1. Apresentação do Serviço SaaS Exemplo

O serviço SaaS exemplo foi concebido de forma a servir de insumo para a aplicação prática da implementação proposta na Seção 6.1 anterior, além de estabelecer uma situação de controle para comparação com a implementação. Escolheu-se o conceito de um serviço SaaS de geração de folha de pagamento. O consumidor envia para o provedor uma lista de seus funcionários, com a descrição de seus cargos; e o provedor os processa, retornando a lista calculada dos salários de cada um. Estabelece-se, assim, um padrão de troca de mensagens (MEP) do tipo requisição e resposta. A Figura 6.9 permite a visualização desse serviço.

É importante observar e contrapor a Figura 6.9 com a Figura 6.1. A diferença entre as duas é que a Figura 6.9 é basicamente a utilização de um serviço SaaS em nuvem sem privacidade do ponto de vista do consumidor, conforme a análise de privacidade da Seção 2.4 demonstrou. A Figura 6.1 é justamente como se pode, na prática, introduzir a possibilidade de privacidade por meio anonimato, concretizada pelo emprego da implementação do esquema prático rastreável, conforme a descrição do produto em *software* coloca (Seção 6.1).

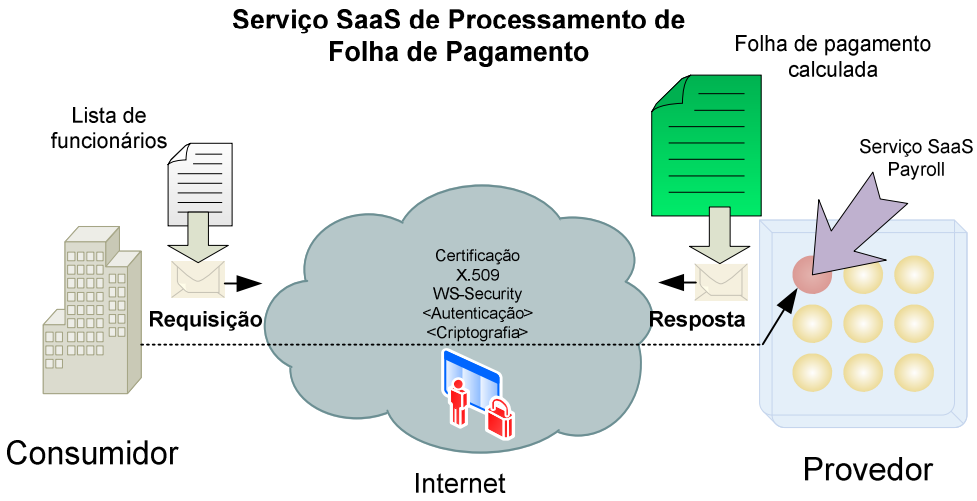

Figura 6.9 – Serviço SaaS Exemplo de Processamento de Folha de Pagamento

O provedor deste serviço exemplo concebe então o contrato WSDL específico a ele, que será denominado ***payroll.wsdl***. Nele, está determinado como um consumidor poderá requisitar o processamento de uma folha de pagamento e, no APÊNDICE E, o conteúdo do arquivo é apresentado.

Basicamente, um provedor recebe como entrada uma lista de funcionários, com cada registro consistindo de nome, cargo e lotação; e retorna para o consumidor uma lista com os mesmos registros acrescidos de um salário.

Sem a presença da solução desta tese (privacidade do consumidor SaaS em relação ao provedor por meio do anonimato), não existe forma de resguardar do provedor os itens

de interesse privados do consumidor durante as interações do consumo de serviço, como visto na análise da privacidade do consumidor SaaS na Seção 2.4.

O que tradicionalmente é empregado nesses casos é a criptografia e a assinatura digital, para garantir a confidencialidade dos dados transmitidos em relação a terceiros e a autenticidade e a integridade das mensagens, respectivamente. É essa a mesma solução utilizada nas funções de contratos e credenciamento e de pagamento da implementação apresentada na Seção 6.1 (autenticação em *Publisher* e autenticação e criptografia em *Subscriber, Depositer* e *Invoicer*).

Para manter consistência com os cenários apresentados e para permitir a comparação com a demonstração de funcionamento da implementação a ser detalhada na próxima Seção, mostra-se agora um exemplo de funcionamento deste serviço SaaS exemplo em um ambiente típico de computação em nuvem, dentro do contexto apresentado na Seção 2.2.2, utilizando o SOAP como o protocolo para a troca de mensagens e o WS-Security para a segurança das informações.

Seguindo as etapas da Figura 2.7, primeiramente o provedor publica o ***payroll.wsdl*** em um UDDI para que o consumidor possa realizar o procedimento de descoberta do serviço. Após a seleção deste serviço, o consumidor contata o provedor em sua plataforma e realiza a adesão ao contrato. Em seguida, o consumidor passa a consumir propriamente o serviço, e, como etapa final, concretiza o pagamento.

Conforme já estabelecido nesta tese (Seção 2.3), o provedor é considerado, no contexto da computação em nuvem, e, mais especificamente, no paradigma SaaS, como a entidade com maior poder de ameaça à privacidade do consumidor. E, como a estrutura multicamadas de anonimato introduz para as etapas de estabelecimento de contratos e de pagamento a intermediação por meio do TPB, possibilitando a privacidade por meio de contratos indiretos, manter-se-á o foco desta análise prática e, consequentemente, deste exemplo de serviço SaaS, nas interações constantes da etapa do consumo de serviço propriamente dito. A comparação a ser feita do consumo de serviço sem e com a utilização da implementação proposta será restrita à troca de mensagens durante apenas o consumo de serviço.

Portanto, abaixo, na Figura 6.10, apresenta-se uma troca de mensagens deste serviço SaaS na etapa do consumo de serviço, conforme a Figura 6.9. Inicialmente, todavia, mostra-se um exemplo sem o uso de autenticação e de criptografia, para que as mensagens possam ser visualizadas completamente.

Mensagem de Requisição

IP de Origem: 186.222.148.40

IP de Destino: viniciuspacheco.com

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
   <S:Header>
      <wsu:Timestamp xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
         <wsu:Created>2013-03-08T00:56:26.675 UTC</wsu:Created>
      </wsu:Timestamp>
   </S:Header>
   <S:Body wsu:Id="body">
      <funcionarios xmlns="http://www.viniciuspacheco.com">
         <funcionarios>
            <nome>Carlos Riacho</nome>
            <cargo>Analista</cargo>
            <lotacao>Departamento de Pagamento</lotacao>
         </funcionarios>
         <funcionarios>
            <nome>Ney Silva</nome>
            <cargo>Tecnico</cargo>
            <lotacao>Departamento de Orcamento</lotacao>
         </funcionarios>
         <funcionarios>
            <nome>Walter Machado</nome>
            <cargo>Tecnico</cargo>
            <lotacao>Departamento de Comunicacao</lotacao>
         </funcionarios>
         <funcionarios>
            <nome>Douglas Coelho</nome>
            <cargo>Analista</cargo>
            <lotacao>Departamento de RH</lotacao>
         </funcionarios>
      </funcionarios>
   </S:Body>
</S:Envelope>
```

Mensagem de Resposta

IP de Origem: viniciuspacheco.com

IP de Destino: 186.222.148.40

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <pagamentos xmlns="http://www.viniciuspacheco.com">
         <pagamentos>
            <funcionario>
               <nome>Carlos Riacho</nome>
               <cargo>Analista</cargo>
               <lotacao>Departamento de Pagamento</lotacao>
            </funcionario>
            <salario>2120.0</salario>
         </pagamentos>
         <pagamentos>
            <funcionario>
               <nome>Ney Silva</nome>
               <cargo>Tecnico</cargo>
               <lotacao>Departamento de Orcamento</lotacao>
            </funcionario>
            <salario>1350.0</salario>
         </pagamentos>
         <pagamentos>
            <funcionario>
               <nome>Walter Machado </nome>
               <cargo>Tecnico</cargo>
               <lotacao>Departamento de Comunicacao </lotacao>
            </funcionario>
            <salario>1350.0</salario>
         </pagamentos>
         <pagamentos>
            <funcionario>
               <nome>Douglas Coelho </nome>
               <cargo>Analista</cargo>
               <lotacao>Departamento de RH</lotacao>
            </funcionario>
            <salario>2120.0</salario>
         </pagamentos>
      </pagamentos>
   </S:Body>
</S:Envelope>
```

Figura 6.10 – Troca de Mensagens para Consumo do Serviço *Payroll* (sem autenticação e criptografia)

Verifica-se, que o provedor terá acesso ao IP real de origem e ao teor da mensagem de requisição (detalhes dos funcionários). Porém, além disso, o provedor irá precisar saber quem está demandando o serviço para que este consumidor possa ser debitado

corretamente. Ademais, é necessário também que se proteja essa troca de mensagens contra terceiros, pois não é de interesse do consumidor que sua lista de funcionários e cargos seja disponibilizada publicamente.

Portanto, a seguir, na Figura 6.11, exemplifica-se o mesmo consumo, mas com a aplicação da autenticação e da criptografia, via o WS-Security, incorporado como um cabeçalho no envelope SOAP.

Mensagem de Requisição

IP de Origem: 186.222.148.40

IP de Destino: viniciuspacheco.com

```
<?xml version="1.0" encoding="utf-8"?>
<S:Envelope xmlns:S11="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd"
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <S:Header>
      <wsu:Timestamp>
         <wsu:Created>2013-03-08T00:55:25.675 UTC</wsu:Created>
      </wsu:Timestamp>
      <wsse:Security>
         <wsse:BinarySecurityToken ValueType="wsse:X509v3"
wsu:Id="X509Token_Consumer"
EncodingType="wsse:Base64Binary">MmtJZc0rgh68hfds87frs8hdhfsdqrKh5i...
</wsse:BinarySecurityToken>
         <ds:Signature>
            <ds:SignedInfo>
               <ds:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
               <ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <ds:Reference URI="#Body">
                  <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                  <ds:DigestValue>LyLsF094hPi4wPU...</ds:DigestValue>
               </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>Hp1ZkmFZ/2kQLXDJ...</ds:SignatureValue>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:Reference URI="#X509Token_Consumer"/>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
         <xenc:EncryptedKey>
```

```
            <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:KeyIdentifier EncodingType="wsse:Base64Binary"
ValueType="wsse:X509SubjectKeyIdentifier">MIGfMa0GCSq...</wsse:KeyIden
tifier>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
            <xenc:CipherData>

<xenc:CipherValue>d2FpbmdvbGRfE0lm4byV0...</xenc:CipherValue>
            </xenc:CipherData>
            <xenc:ReferenceList>
               <xenc:DataReference URI="#Body_Encrypted"/>
            </xenc:ReferenceList>
         </xenc:EncryptedKey>
      </wsse:Security>
   </S:Header>
   <S:Body wsu:Id="Body">
      <xenc:EncryptedData
Type="http://www.w3.org/2001/04/xmlenc#Element"
wsu:Id="Body_Encrypted">
      <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes1983
cbc"/>
      <xenc:CipherData>
         <xenc:CipherValue>d2FpbmdvbGRfE0lm4byV0...
</xenc:CipherValue>
      </xenc:CipherData>
      </xenc:EncryptedData>
   </S:Body>
</S:Envelope>
```

Mensagem de Resposta

IP de Origem: viniciuspacheco.com

IP de Destino: 186.222.148.40

```
<?xml version="1.0" encoding="utf-8"?>
<S:Envelope xmlns:S11="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd"
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <S:Header>
      <wsu:Timestamp>
         <wsu:Created>2013-03-08T00:56:26.675 UTC</wsu:Created>
      </wsu:Timestamp>
      <wsse:Security>
         <wsse:BinarySecurityToken ValueType="wsse:X509v3"
wsu:Id="X509Token_Provider"
```

```
EncodingType="wsse:Base64Binary">CCA9CgAwMIIEZzIBAgIQEmtJZc0rqrKh5i...
</wsse:BinarySecurityToken>
         <ds:Signature>
            <ds:SignedInfo>
               <ds:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
               <ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <ds:Reference URI="#Body">
                  <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                  <ds:DigestValue>hPiLyLsF0944wPU...</ds:DigestValue>
               </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>Hp1ZkmFZbchm5gK...</ds:SignatureValue>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:Reference URI="#X509Token_Provider"/>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
         <xenc:EncryptedKey>
            <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:KeyIdentifier EncodingType="wsse:Base64Binary"
ValueType="wsse:X509SubjectKeyIdentifier">Ma0MIGkgtt5566533a33fGCSq...
</wsse:KeyIdentifier>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
            <xenc:CipherData>
               <xenc:CipherValue>E0lm4bmdvbGRfV0...</xenc:CipherValue>
            </xenc:CipherData>
            <xenc:ReferenceList>
               <xenc:DataReference URI="#Body_Encrypted"/>
            </xenc:ReferenceList>
         </xenc:EncryptedKey>
      </wsse:Security>
   </S:Header>
   <S:Body wsu:Id="Body">
      <xenc:EncryptedData
Type="http://www.w3.org/2001/04/xmlenc#Element"
wsu:Id="Body_Encrypted">
      <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes1983
cbc"/>
      <xenc:CipherData>
         <xenc:CipherValue>GRfE0lm4d2FpbmdvbbyV0...
</xenc:CipherValue>
      </xenc:CipherData>
      </xenc:EncryptedData>
   </S:Body>
</S:Envelope>
```

Figura 6.11 - Troca de Mensagens para Consumo do Serviço *Payroll* (com autenticação e criptografia)

Percorrem-se, agora, os campos XML das mensagens de requisição e de resposta, para apresentar a correspondência dos mesmos com as funções de autenticação e criptografia, conforme a Figura 6.9.

O envelope SOAP inicia com a delimitação do cabeçalho, no campo <*S:Header*>, e o primeiro elemento é apenas um *timestamp* de criação do cabeçalho.

Em seguida, passa-se para o próprio cabeçalho WS-Security, delimitado por <*wsse:Security*>, e seu primeiro elemento é um *token* de autenticação, definido pelo campo <*wsse:BinarySecurityToken*>. Esse *token* pode ser de vários tipos, como, por exemplo, um *ticket* Kerberos [94] ou, como será utilizado aqui, um certificado X.509 [95]. Mais ainda, podem-se definir tipos personalizados de *tokens*. Para se descobrir que tipo de *token* está incluído no <*wsse:BinarySecurityToken*>, basta verificar o atributo *ValueType*.

Nas mensagens acima, a autenticação é promovida pelo *token* X.509 incluído em <*wsse:BinarySecurityToken*>, que é um certificado. Todavia, enviar apenas o certificado anexado na mensagem, deixaria a mesma suscetível a um ataque de *replay* simples, onde um atacante copiaria um certificado válido de dentro de uma solicitação e o encaminharia como seu, burlando a autenticação. Para impedir esse comportamento, incorpora-se o uso de uma assinatura. A assinatura basicamente consiste na utilização de um *hash* cifrado com criptografia assimétrica, onde uma entidade possui um par de chaves, uma secreta e outra pública. A chave secreta, neste contexto, é utilizada para cifrar um *hash* computado a partir da mensagem a ser enviada, gerando a assinatura, e sua chave pública correspondente é a única capaz de descriptografar essa assinatura gerada, recuperando o *hash* original enviado. Localmente, a entidade receptora, para checar a assinatura, gera um outro *hash* da mensagem recebida e compara com o *hash* recuperado da assinatura. Se ambos forem idênticos, comprova-se que a mensagem foi gerada pela entidade possuidora da chave secreta relativa ao certificado e que a mensagem não foi alterada (propriedade do *hash*). Assim, o certificado encaminhado na mensagem como *token* é quem identifica a chave secreta empregada e associa a chave

pública a ser utilizada na verificação. A assinatura em si é o próximo campo do cabeçalho (<*ds:Signature*>) e referenciará este certificado. A referência usa o Id do certificado dentro da mensagem, definido pelo atributo *wsu:Id*, sendo que na requisição é X509Token_Consumer e na resposta é X509Token_Provider, por se referirem aos certificados do consumidor e do provedor, respectivamente.

O campo <*ds:Signature*> delimita, então, a assinatura em si e possui vários elementos que juntos a compõem.

O primeiro elemento é o <*ds:SignedInfo*>, e descreve em seus subelementos as características dos dados assinados. Primeiramente, em <*ds:CanonicalizationMethod*>, define-se como a informação foi tratada antes da geração do *hash* que foi depois cifrado. O processo de canonizar dados XML [127] antes de se aplicar uma função *hash* é importante para que a entidade que irá checar a assinatura, obtendo outro *hash* dos mesmos dados, possa realizar o processo nos exatos mesmos dados de origem, pois qualquer mera alteração originará *hashs* totalmente diferentes. Isso é relevante neste contexto porque, às vezes, elementos XML são descritos de forma desigual em sistemas distintos; e canonizando-se os elementos a serem utilizados como entrada para o *hash* garante-se que os elementos XML estejam exatamente iguais. Em <*ds:SignatureMethod*>, estabelece-se o algoritmo geral empregado para criar a assinatura. Neste caso, será empregado um *hash* do tipo SHA1 [48], e depois a cifração assimétrica será feita com o RSA [110]. O último subelemento de <*ds:SignedInfo*> é o <*ds:Reference*>. Nele, especifica-se qual parte do envelope SOAP foi assinada, por meio de outra referência do tipo *wsu:Id*, além de anexar o valor original dos dados assinados após a execução do *hash*. Aqui, a informação a ser assinada é o corpo da mensagem SOAP, identificado pelo Id="*Body*".

O próximo elemento da assinatura é o <*ds:SignatureValue*>. Ele encapsula a assinatura propriamente dita, após o *hash* e a criptografia assimétrica, com base no certificado referenciado.

E, o último elemento da assinatura é o <*ds:KeyInfo*>, onde se descreve a chave utilizada na cifração. Nesta assinatura, apenas se referencia o certificado incluído no próprio

cabeçalho da mensagem. Esse relacionamento é provido pelo <*wsse:SecurityTokenReference*> e aponta para o Id do certificado, sendo X509Token_Consumer na requisição e X509Token_Provider na resposta.

Resolvida a questão da autenticação pelo certificado e pela assinatura, resta o aspecto da criptografia das mensagens. Assim sendo, o campo <*xenc:EncryptedKey*>, colocado após a assinatura, se presta à função. Ele caracteriza a chave a ser empregada para cifrar a informação desejada e posicionada em qualquer local da mensagem SOAP. No entanto, assim como na assinatura, os dados a serem tornados confidenciais neste ambiente são justamente aqueles presentes no corpo da mensagem.

Por uma questão de performance, a chave encapsulada por <*xenc:EncryptedKey*> normalmente é uma chave simétrica, uma vez que a criptografia simétrica é bem mais performática do que a assimétrica. Entretanto, como a chave simétrica é única, não se pode transmiti-la anexada em claro na própria mensagem. Portanto, o método para informar a chave à outra entidade, é justamente cifrá-la com criptografia assimétrica, empregando a chave pública do destinatário. Garante-se, dessa maneira, que somente aquele que possui a chave privada par da pública usada poderá ter acesso à chave simétrica cifrada.

A chave em <*xenc:EncryptedKey*> também possui vários elementos a caracterizando. O primeiro, o <*xenc:EncryptionMethod*>, especifica o algoritmo empregado na cifragem da chave simétrica. Aqui, é o próprio RSA. Em seguida, vem o elemento <*ds:KeyInfo*>, que traz informações sobre a chave utilizada para cifrar a chave simétrica. Tanto na mensagem de requisição, quanto na de resposta, usa-se a chave pública do destinatário, e referencia-se essa chave por um apontamento via <*wsse:SecurityTokenReference*> para os certificados públicos do provedor no caso do consumidor, e do consumidor no caso do provedor. No próximo elemento, em <*xenc:CipherData*>, transmite-se a chave simétrica cifrada; e, no último, chamado <*xenc:ReferenceList*>, identifica-se a parte da mensagem que foi cifrada com essa chave A referência, mais uma vez, é feita por um Id, e, em ambas as mensagens, é Body_Encrypted.

Com campos percorridos acima, chega-se ao fim do cabeçalho WS-Security e pode-se analisar o corpo da mensagem. Nas mensagens da Figura 6.11, ele é abarcado pelo campo <*S:Body*>. No caso dos dados utilizados para gerar o *hash* e, consequentemente, a assinatura em <*ds:Signature*>, verifica-se que o Id="Body" abarca todo o corpo da mensagem. Já para a criptografia, dentro do campo <*S:Body*>, existe o elemento <*xenc:EncryptedData*> e ele carrega a informação cifrada. O atributo Id permite a referência para a chave a ser usada no processamento (Body_Encrypted, no caso), e seus elementos detalham como os dados podem ser descriptografados. Em <*xenc:EncryptionMethod*>, descreve-se o algoritmo empregado na cifragem, e, como dito antes, neste caso foi empregada a criptografia simétrica, mais especificadamente, o 3DES [12]. E, finalmente, em <*xenc:CipherData*>, dentro de <*xenc:CipherValue*>, transmite-se a informação cifrada propriamente dita.

No caso do consumo de serviço realizado pelas mensagens da Figura 6.11, e das mensagens seguintes do mesmo consumidor para o provedor, a informação privada do próprio consumidor é revelada ao provedor. Mesmo com a integridade e autenticidade permitida pela assinatura, e a confidencialidade possibilitada pela criptografia, apenas a ameaça de terceiros é mitigada. Esse paradigma foi apresentado por esta tese na Seção 2.2, onde o cenário modelo SaaS é mostrado (Seção 2.2.2), o provedor é definido como a entidade com maior poder de ameaça à privacidade do consumidor (Seção 2.3) e a análise da privacidade neste contexto é detalhada (Seção 2.4). De destacar, prover uma alternativa para essa situação foi justamente a motivação desta tese. E, considerando as afirmações da Seção 2.4, pode-se agora visualizar na prática os itens de interesse do consumidor sendo expostos ao provedor.

Abaixo, na Tabela 6.1, faz-se a correspondência dos itens de interesse do consumidor expostos ao provedor pelas interações de consumo, inicialmente apresentados pela Tabela 2.1, com os dados concretos sendo transmitidos nas mensagens da Figura 6.11.

Tabela 6.1 – Correspondência dos itens de interesse do consumidor com os dados concretos transmitidos.

Nível de Interação Consumidor-Provedor	**Itens de Interesse**	**Dimensão da Privacidade Afetada**	**Dado Concreto**
Contrato Consumidor-Provedor	Identificação do Consumidor no Contrato	ID	Dados não evidenciados pela Figura 6.11, pois são transmitidos não no consumo de serviço propriamente dito e sim na etapa prévia de contrato formal e credenciamento (etapa 3 da Figura 2.7).
	Endereço do Consumidor e Local de Negócios	Localização	
	SLA e estimativas de consumo	Comportamento	
Troca de Mensagens	Credenciais AAA	ID	<*wsse:BinarySecurityToken*> Certificado X509 identificado por Id="X509Token_Consumer".
	MEPs no tempo	Comportamento	Repetição do mesmo <*wsse:BinarySecurityToken*> nas mensagens seguintes.
	Dados das Mensagens	ID, Localização, Comportamento, Conteúdo	<*S:Body*> Corpo da mensagem descriptografado. <*xenc:EncryptedData*> com Id="Body_Encrypted" após processamento.
Comunicações de Rede	Endereço IP	ID, Localização, Comportamento	IP do consumidor: 186.222.148.40

6.2.2. Exemplo de Operação da Implementação do Esquema Prático

Com a descrição da implementação do esquema prático rastreável detalhada na Seção 6.1, e com um serviço SaaS exemplo apresentado na Seção 6.2.1, mostrando como suas transações padrão afetam concretamente a privacidade do consumidor (Tabela 6.1), pode-se agora utilizar a implementação para proteger o consumidor deste serviço SaaS enquanto estiver interagindo com o provedor, obedecendo, para isso, o design de

transações especificado pelo esquema prático rastreável para consumo anônimo de serviços SaaS (Capítulo 4).

Assim sendo, dentro da implementação, o WSDL do *Payroll* (APÊNDICE E) é, então, o arquivo a ser enviado ao TPB pelo *web service Publisher*, mais especificamente, pela operação *publishservice*. O provedor irá receber uma chave do grupo para verificar assinaturas, e terá à sua disposição, também, a possibilidade de adicionar mais créditos de serviço com a operação *addservicecredits*, caso sua capacidade permita ou os créditos sendo utilizados pelos consumidores estejam próximo ao fim.

Ainda dentro da função de contratos e credenciamento da implementação, na parte que envolve o consumidor, estes podem fazer adesão junto ao TPB para o serviço SaaS do WSDL descrito em ***payroll.wsdl***, por meio da operação *subscribeservice* do *web service Subscriber*. Ademais, pelo *Subscriber*, é possível também comprar mais créditos de serviço, quando necessário, ou mesmo reportar um possível outro consumidor malicioso que tenha utilizado um crédito de serviço deste consumidor específico.

Na função do consumo de serviço anônimo, os agentes de serviço *Consumer-Signer* e *Provider-Verifier* permitem ao consumidor do *Payroll* enviar sua lista de funcionários para o provedor e receber o processamento da mesma de forma anônima, sem ter sua privacidade impactada.

Finalmente, na função de pagamento da implementação, o provedor, empregando o *web service Depositer*, deposita os créditos anônimos de serviço e recebe o pagamento devido. Para o consumidor, com o *web service Invoicer*, garante-se a possibilidade de acompanhamento de seus gastos, com extratos personalizados.

Conforme aclarado anteriormente, por ser a influência do provedor sobre a privacidade do consumidor o ponto focal de atividade desta tese e, consequentemente, desta implementação, detalhar-se-á apenas as transações compreendendo as interações diretas entre consumidor e provedor, mais especificadamente, na função de consumo anônimo de serviço provida pelos agentes *Consumer-Signer* e *Provider-Verifier*.

Deste modo, para o serviço *Payroll*, considera-se agora que etapas referentes à função de contratos e credenciamento ocorreram conforme a Seção 6.1.3 descreveu, e, após os consumos de serviços anônimos a serem detalhados a seguir, a função de pagamento também transcorre de acordo com o delineado na Seção 6.1.5.

Apresenta-se abaixo, na Figura 6.12, uma troca de mensagens para o consumo do serviço *Payroll*, utilizando a implementação do esquema prático rastreável, intermediada pelos agentes de serviço *Consumer-Signer* e *Provider-Verifier*.

Mensagem de Requisição

IP de Origem: 186.222.148.40

IP de Origem Anonimizado: 217.172.188.138

IP de Destino: viniciuspacheco.com

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
   <S:Header>
      <wsu:Timestamp xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
         <wsu:Created>2013-03-12T13:27:37.640 UTC</wsu:Created>
      </wsu:Timestamp>
      <wsse:Security xmlns:wsse="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wsswssecurity-secext-1.0.xsd">
         <wsse:BinarySecurityToken
EncodingType="http://www.viniciuspacheco.com/xsd/service-
credits.xsd#SERVICECREDITFORMAT"
ValueType="http://www.viniciuspacheco.com/xsd/service-
credits.xsd#SERVICECREDITTYPE"
wsu:Id="Service_Credit">9316720810980024790690973849863050398340345890…
</wsse:BinarySecurityToken>
         <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:SignedInfo>
               <ds:CanonicalizationMethod
Algorithm="http://www.w3c.org/TR/2001/REC-xml-c14n-20010315"/>
               <ds:SignatureMethod
Algorithm="http://www.viniciuspacheco.com/xsd/group-
signatures.xsd#GROUPSIGNATURE"/>
               <ds:Reference URI="#Body">
                  <ds:DigestMethod
Algorithm="http://www.w3c.org/TR/2000/09/xmldsig#sha512"/>
               </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>226623542373604660…</ds:SignatureValue>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:Reference URI="#Service_Credit"/>
```

```
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
      </wsse:Security>
   </S:Header>
   <S:Body wsu:Id="Body">
      <funcionarios xmlns="http://www.viniciuspacheco.com">
         <funcionarios>
            <nome>569209349432874</nome>
            <cargo>Analista</cargo>
            <lotacao>*</lotacao>
         </funcionarios>
         <funcionarios>
            <nome>670309349438743</nome>
            <cargo>Tecnico</cargo>
            <lotacao>*</lotacao>
         </funcionarios>
         <funcionarios>
            <nome>959341049438776</nome>
            <cargo>Tecnico</cargo>
            <lotacao>*</lotacao>
         </funcionarios>
         <funcionarios>
            <nome>702417349438820</nome>
            <cargo>Analista</cargo>
            <lotacao>*</lotacao>
         </funcionarios>
      </funcionarios>
   </S:Body>
</S:Envelope>
```

Mensagem de Resposta

IP de Origem: viniciuspacheco.com

IP de Destino: 217.172.188.138

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <pagamentos xmlns="http://www.viniciuspacheco.com">
         <pagamentos>
            <funcionario>
               <nome>569209349432874</nome>
               <cargo>Analista</cargo>
               <lotacao>*</lotacao>
            </funcionario>
            <salario>2120.0</salario>
         </pagamentos>
         <pagamentos>
            <funcionario>
               <nome>670309349438743</nome>
               <cargo>Tecnico</cargo>
               <lotacao>*</lotacao>
            </funcionario>
            <salario>1350.0</salario>
```

```
         </pagamentos>
         <pagamentos>
            <funcionario>
               <nome>959341049438776</nome>
               <cargo>Tecnico</cargo>
               <lotacao>*</lotacao>
            </funcionario>
            <salario>1350.0</salario>
         </pagamentos>
         <pagamentos>
            <funcionario>
               <nome>702417349438820</nome>
               <cargo>Analista</cargo>
               <lotacao>*</lotacao>
            </funcionario>
            <salario>2120.0</salario>
         </pagamentos>
      </pagamentos>
   </S:Body>
</S:Envelope>
```

Figura 6.12 – Troca de Mensagens para Consumo do Serviço *Payroll* (com anonimato do consumidor)

Iniciando a análise da mensagem de requisição, verifica-se que o IP de origem é anonimizado pelo agente de serviço *Consumer-Signer*. Mais ainda, apesar de uma segunda mensagem de requisição não estar mostrada, o Tor [123], empregado aqui, irá promover o uso de outro IP para cada nova requisição.

Já dentro do envelope SOAP, na parte do cabeçalho (<*S:Header*>), o primeiro elemento presente é um *timestamp*, útil para o registro do tempo da criação da mensagem.

O segundo elemento do cabeçalho corresponde ao WS-Security (<*wsse:Security*>) e é onde o anonimato de credenciais, relativo à camada de anonimato de metadados da estrutura multicamadas, é posto em prática. Seguindo o design de transações do esquema prático rastreável, a credencial anônima será a assinatura de grupo.

Junto com a credencial anônima, contudo, também é enviado na mensagem de requisição um crédito de serviço, e sua inclusão na mensagem é feita justamente pelo primeiro elemento do WS-Security, dentro de <*wsse:BinarySecurityToken*>. Esse elemento, como visto anteriormente, é utilizado tipicamente para a incorporação de *tokens* para autenticação, como um certificado X.509 ou um *ticket* Kerberos. Entretanto,

também é possível a criação de *tokens* personalizados, e como este crédito de serviço servirá para validar a assinatura anônima em um consumo de serviço, similarmente, por exemplo, ao certificado X.509 empregado na Figura 6.11, ele será codificado no cabeçalho WS-Security como um *token*. A descrição dessa customização é feita pelos atributos EncodingType e ValueType. Este crédito de serviço também é identificado para a futura referência da assinatura de grupo pelo Id="Service_Credit".

A assinatura de grupo propriamente dita, que irá fazer o papel de credencial anônima, é o próximo campo do cabeçalho WS-Security. Da mesma forma que uma assinatura digital tradicional, ela será encapsulada pelo campo *<ds:Signature>*. Porém, alguns ajustes são realizados em seus elementos para acomodar o funcionamento segundo o paradigma do esquema rastreável.

O primeiro elemento da assinatura de grupo é o *<ds:SignedInfo>*, e, nele, discorre-se sobre os dados assinados. Seu primeiro elemento, o *<ds:CanonicalizationMethod>*, especifica o algoritmo de canonização imprimido nos dados a serem assinados, antes da aplicação do *hash*. Em uma assinatura digital típica, toda informação antes de ser assinada passa por uma função de *hash*. Aqui, o *hash* também é empregado, mas, conforme detalhado pelo design do esquema prático rastreável (Seção 4.1), ele serve para outro propósito também: o de permitir que o parâmetro ***M*** do algoritmo **Sign** (Seção 3.4.1.1) do modelo VLR seja não a mensagem completa e sim algo que não carregue valor semântico. O TPB que irá receber o relatório de todos os consumos (DB-SERVICE-BILLING-PROVIDER) via o *web service Depositer* não terá acesso ao conteúdo das mensagens, e quando gerar a base de dados SERVICE-BILLING-TPB, seus registros serão da forma (***M***, ***gsk*[*i*]**, crédito de serviço e *timestamp*), com ***M*** sendo o *hash*.

Em seguida, ainda dentro de *<ds:SignedInfo>*, observa-se o elemento *<ds:SignatureMethod>*. Ele é importante neste contexto de assinaturas de grupo, pois ao invés de especificar um algoritmo tradicional de assinatura digital, como o visto na Figura 6.11, de aplicação de um *hash* SHA-1 seguido de cifração RSA, aqui ele institui o emprego das assinaturas de grupo VLR, e aponta, portanto, para a descrição XSD deste procedimento. Dentro do XSD identificado, basicamente, tem-se o detalhamento

do emprego do algoritmo **Sign** (Seção 3.4.1.1) para a incorporação de uma assinatura $\boldsymbol{\sigma}$ em um cabeçalho WS-Security.

Como o último elemento de *<ds:SignedInfo>*, passa-se ao *<ds:Reference>*. Em seu atributo Id, referencia a mensagem a ser assinada, que é o corpo do envelope SOAP; e, no subelemento *<ds:DigestMethod>*, institui o algoritmo utilizado para gerar o *hash* da mensagem, e que se tornará, consequentemente, o parâmetro $\boldsymbol{M}$.

Após o *<ds:SignedInfo>*, tem-se o campo *<ds:SignatureValue>*, e ele carrega o valor propriamente dito da credencial anônima, que é justamente o parâmetro $\boldsymbol{\sigma}$.

Finalizando a assinatura do WS-Security, tem-se o campo *<ds:KeyInfo>*, que estabelece informações sobre a chave utilizada para a assinatura. No entanto, aqui a credencial anônima estará associada com o crédito de serviço para consumo. Então, referencia-se o *token* "Service_Credit" como o *<wsse:SecurityTokenReference>*.

Para o corpo da mensagem SOAP (*<S:Body>*), identificado pelo Id="Body", para referência dos dados a serem assinados pelo campo *<ds:Signature>*, verifica-se que a informação não é criptografada como aquela da requisição na Figura 6.11. Isso porque a criptografia simples não irá proteger a privacidade do consumidor em relação ao provedor, já que o provedor iria ter que descriptografar a informação para poder processá-la. Ao invés disso, emprega-se uma tecnologia de anonimato de dados, conforme proposto pela camada de anonimato de conteúdo da estrutura multicamadas. Mais especificadamente, nesta implementação opta-se pelo anonimato-k [111], onde os valores dos campos com identificadores de informações sensíveis, como o nome dos funcionários e suas lotações, são substituídos por valores aleatórios ou suprimidos (substituídos por *) e enviados ao provedor. Mantem-se, todavia, a possibilidade do provedor processar o serviço, visto que a definição do cargo é mantida. O que o provedor não será capaz de fazer é descobrir a lista de funcionários da empresa, nem onde esses trabalham.

Por fim, a mensagem de resposta, gerada pelo provedor, não precisa ser assinada anonimamente, e, por isso, não contém um cabeçalho WS-Security. O corpo da mensagem traz o valor calculado dos salários, mas retém o anonimato gerado pelos

identificadores anônimos dos campos do nome e lotação. A volta desses campos para os valores originais será realizada pelo agente de serviço do consumidor, o *Consumer-Signer*.

6.2.3. Considerações de Segurança

Como etapa final da demonstração de funcionamento da implementação do esquema prático rastreável, realizam-se considerações de segurança sobre a implementação, percorrendo-se a relação de associações dos itens de interesse e tecnologias de anonimato, propostas na avaliação de privacidade da Seção 4.2 e resumidas na Tabela 4.1, com os dados concretos responsáveis pela anonimização e consequente privacidade dos consumidores. O foco se mantém na relação entre consumidor e provedor, entretanto algumas considerações são estendidas à influência de consumidores maliciosos e de atacantes externos sobre a privacidade do consumidor nas transações estabelecidas pela implementação.

Primeiramente, apresenta-se a Tabela 6.2. Nela, mostra-se a correspondência dos itens de interesse do consumidor com a tecnologia de anonimato empregada e com o dado concreto que evidencia a eficácia da tecnologia, observados na transação de consumo da Figura 6.12.

Tabela 6.2 – Relação das tecnologias de anonimato com os dados concretos correspondentes, dentro da implementação do esquema prático rastreável.

Camada da Estrutura	**Itens de Interesse**	**Tecnologia de Anonimato Empregada**	**Dimensão da Privacidade Protegida**	**Acesso Provedor**	**Dado Concreto**
Contrato	Identificação do Consumidor	Contratos Indiretos: *TPB*	ID	Não tem	Função de Credenciamento e Contratos. *Web services Publisher* e *Subscriber*.
	Endereço do Consumidor	Contratos Indiretos: *TPB*	Localização	Não tem	
	SLA e estimativas de consumo	Contratos Indiretos: *TPB*	Comportamento	Não tem	
Metadados	Credenciais AAA	*Assinaturas de Grupo* – Modelo VLR [23]	ID	Tem acesso, mas consumidor é anônimo	*<wsse: BinarySecurity Token>* e *<ds:Signature Value>*

	MEPs no tempo	*Assinaturas de Grupo* – Modelo VLR [23]	Comportamento	Tem acesso, mas consumidor é anônimo	*<wsse: BinarySecurity Token>* e *<ds:Signature Value>* subsequentes
Conteúdo	Dados das Mensagens	Anonimato-k	ID, Localização, Comportamento, Conteúdo	Tem acesso, mas consumidor é anônimo	*<S:Body>* anonimizado
Rede	Endereço IP	Tor	ID, Localização, Comportamento	Tem acesso, mas consumidor é anônimo	217.172.188.138 e os IPs anônimos subsequentes

De início, analisa-se a questão dos itens de interesse relativos à camada de anonimato de contratos, referentes à privacidade de ID, localização e comportamento do próprio contrato. Esse itens de interesse, dentro da implementação do esquema rastreável, não são expostos ao provedor na etapa de consumo de serviço (e nem em nenhuma outra etapa), já que fazem parte, na verdade, das interações entre consumidor e TPB e provedor e TPB atinentes à função de contratos e credenciamento (Seção 6.1.3). A tecnologia de contratos indiretos, via TPB, é materializada pelas operações dos *web services Publisher* e *Subscriber*.

Quanto às considerações de segurança para o *Publisher* e o *Subscriber*, tem-se que, por não haver contato direto entre provedor e consumidor, a privacidade do consumidor é resguardada. Em relação a terceiros, especificamente para o *web service Publisher*, a espionagem das transações não possibilitará a nenhum consumidor malicioso obter acesso indevido ao serviço, pois mesmo que apenas a autenticação por certificado digital seja empregada pelo provedor e TPB, obter uma cópia da chave pública para verificação de acesso aos futuros consumos de serviço, resultado de uma possível espionagem da operação *publishservice*, só permitirá por parte de tal atacante a eventual validação de uma assinatura anônima de um consumidor legítimo, sem impossibilitar tal consumo e sem impactar em sua privacidade. A chave pública não permite a assinatura de mensagens. Ainda assim, uma verificação clandestina de assinatura não abarcará

usuários revogados, visto que o parâmetro ***RL*** não estará disponível. Atacantes externos compartilham esse contexto com os consumidores maliciosos.

Já para o *web service Subscriber*, por ter em suas transações a transferência da chave privada do consumidor e dos créditos de serviço para validar as assinaturas, nas operações *subscribeservice* e *buyservicecredits*, é necessário, além de implementar a autenticação, o emprego da criptografia. Dessa forma, além de não envolver o provedor, as transações cifradas entre consumidor e TPB não permitem a terceiros espionar e nem obter informações privadas de cada consumidor. A privacidade do consumidor é resguardada.

Passando para a camada de metadados da estrutura multicamadas, analisa-se como a implementação do esquema prático rastreável lida com as credenciais anônimas. A credencial para consumo do serviço consistirá, na prática, da assinatura de grupo $\boldsymbol{\sigma}$ e do respectivo crédito de serviço.

Para a assinatura de grupo $\boldsymbol{\sigma}$, enviada pelo elemento <*ds:SignatureValue*> da requisição, do ponto de vista do provedor, esse não conseguirá identificar o consumidor. Saberá somente que é um consumidor válido (pertencente ao grupo), caso o algoritmo **Verify** (Seção 3.4.1.1) tenha sucesso. Como pode ser comprovado pelo detalhamento matemático em [23], uma assinatura de grupo não pode ser diretamente associada a um único membro do grupo, a não ser pelo mestre do grupo, que neste caso é o TPB.

Em relação a um possível consumidor espião, ter acesso à assinatura poderá levar à comprovação de que o consumidor em voga é válido (com limitação do desconhecimento das revogações), na medida em que a chave pública do grupo, necessária para o algoritmo **Verify,** é divulgada publicamente (SERVICE ID). Porém, esse conhecimento não permite que este consumidor malicioso possa consumir serviço em nome do outro, pois a assinatura válida carrega em sua composição um *hash* da mensagem original e se for empregada na tentativa de validar outra mensagem será recusada pelo provedor durante o **Verify**. Um atacante externo também terá a mesma limitação.

Essa privacidade de ID por anonimato da assinatura de grupo, além de possibilitar o consumo de serviço anônimo com autenticação, também salvaguarda a privacidade de comportamento do consumidor, pois como a assinatura se altera por meio do *hash* da mensagem sendo transmitida (***M*** do algoritmo **Sign**), não haverá correlacionamento de consumos oriundos de um mesmo remetente. Tanto o provedor quanto consumidores maliciosos e atacantes externos estão submetidos a esta característica da solução.

Ainda na camada de metadados, resta analisar o crédito de serviço – <*wsse:BinarySecurityToken*>. Este é criado pelo TPB, e a associação de créditos válidos para cada consumidor só existe dentro do próprio TPB. Um provedor que receba uma assinatura anônima válida registra como aceito o crédito que a acompanha, caso este ainda não tenha sido apresentado. A privacidade de ID do consumidor não é violada porque esse registro não identifica o consumidor.

Um consumidor malicioso que intercepte um crédito válido de outro consumidor poderá incluí-lo em sua mensagem assinada com uma assinatura de grupo válida gerada pelo espião. Entretanto, mesmo que o provedor eventualmente aceite o crédito e preste o serviço, quando os créditos e as assinaturas correspondentes forem enviados ao TPB, na operação *depositcredits* do *web service Depositer*, o TPB descobrirá que um crédito de serviço foi utilizado por um consumidor para o qual este não estava alocado. O TPB então procederá à revogação do anonimato de tal consumidor, e tomará as medidas legais cabíveis. Ademais, o consumidor lesado, que ao tentar consumir o serviço utilizando um crédito que lhe tenha sido roubado, também poderá iniciar o processo de rastreamento acima, executando a operação *reportabuse* do *web service Subscriber*. Para atacantes externos, ter acesso a um crédito válido não possibilita o consumo de serviço, porque ele não será capaz de criar uma assinatura de grupo válida para acompanhar o crédito. Destarte, considerando o acesso do provedor, dos consumidores maliciosos e de atacantes externos, não se viola a privacidade de ID do consumidor, e, mesmo que para um consumidor malicioso possa ser possível a utilização de um crédito roubado, esse comportamento será rastreado e passível de punição.

Para a privacidade de comportamento, como o provedor não sabe a associação de créditos aleatórios para cada consumidor, não poderá identificar consumos subsequentes

de um mesmo consumidor. Apenas o TPB, durante a elaboração da fatura (algoritmo **Trace**), conseguirá obter o comportamento de consumo de cada ***gsk***[***i***], e, consequentemente, de cada consumidor. Para consumidores maliciosos e atacantes externos também não é possível, a partir de duas mensagens de um mesmo consumidor, relacioná-las a uma mesma entidade analisando o crédito de consumo. Mantém-se, desse modo, protegida a privacidade de comportamento do consumidor, na camada de anonimato de metadados.

Para a camada de anonimato de conteúdo o item de interesse em questão é os dados da mensagem, representado pelo dado concreto <*S:Body*> das mensagens mostradas Figura 6.12. As dimensões de privacidade do consumidor envolvidas são a de ID, localização, comportamento e conteúdo oriundas de uma análise semântica das informações sendo transmitidas.

O <*S:Body*> é constituído dos elementos que irão ser processados pelo provedor. O objetivo então é manter a possibilidade de processamento do provedor enquanto se anonimizam as informações. Nesta implementação do esquema prático rastreável, o corpo da mensagem é mantido privado substituindo-se os elementos que possibilitam uma identificação do consumidor (Nome e Lotação) por números aleatórios ou por * (supressão). Essa prática é suportada pela tecnologia de anonimato-k [111]. Todavia, a descrição dos cargos não é alterada e permite a computação específica deste serviço SaaS (*Payroll*). O provedor ao responder ao consumidor envia a folha de pagamento calculada para os identificadores numéricos e o agente de serviço *Consumer-Signer* desfaz o anonimato de dados, entregando para a aplicação requisitante a folha completa calculada. Para a ameaça relativa a consumidores maliciosos e a atacantes externos, o anonimato de dados se comporta da mesma maneira, e as privacidades de ID, localização, comportamento e conteúdo são resguardadas.

Enfim, observa-se a eficácia do dado concreto que implementa a privacidade de ID, localização e comportamneto da camada de anonimato de Rede. Este dado concreto é justamente os endereços IPs utilizados pelo consumidor durante as transações. A tecnologia de anonimato empregada para anonimizar o item de interesse é uma rede Mix-Net, mais especificamente, o Tor [47]. Dessa forma, ao invés de receber como IP

de origem 186.222.148.40, ao provedor é entregue inicialmente o IP 217.172.188.138. Nos consumos subsequentes do mesmo consumidor, o Tor irá gerar e usar novos IPs anônimos. As privacidades de ID, localização e comportamento são protegidas. Abaixo, na Figura 6.13, comprova-se que a identidade e localizações oriundas do IP original 186.222.148.40 são anonimizadas pelo IP 217.172.188.138. As informações obtidas abaixo são fornecidas pela ferramenta *online* de análise de IPs chamada *whatsmyipaddress*[22].

Figura 6.13 – Eficácia do Tor em anonimizar um endereço IP

Com as considerações de segurança apresentadas acima, termina-se a demonstração de funcionamento da implementação do esquema prático rastreável para consumo anônimo de serviços SaaS. Instituiu-se, então, uma prova dos conceitos estabelecidos pelas contribuições originais da tese. Validou-se experimentalmente a análise de privacidade da Seção 2.4 na Seção 6.2.1, e legitimou-se também a avaliação de privacidade da Seção 4.2 na demonstração de funcionamento da Seção 6.2.2 e com as considerações de segurança da Seção 6.2.3. Destarte, fica validada experimentalmente a aplicabilidade e a eficácia da estrutura multicamadas de anonimato e do conceito dos esquemas práticos.

[22] Acessada no endereço http://whatismyipaddress.com/, em 18/03/2013.

7. CONCLUSÕES

A motivação do trabalho desta tese foi a introdução da privacidade no consumo de serviços no modelo de entrega SaaS da computação em nuvem. Esse tipo de consumo de serviços é o que atualmente tem mais penetração no mercado e o que possui a maior intenção de implantação por parte das organizações [45] [1]. Todavia, pesquisas mostram que a preocupação mais impactante, impedindo ou pelo menos diminuindo o ritmo de adoção da tecnologia, ainda é a privacidade [2].

Buscou-se, então, oferecer uma opção para as entidades que estão obrigadas a abdicar de sua privacidade ao usufruir do modelo SaaS como consumidores, e para aquelas que ainda optam por não se submeterem a um ambiente não condizente com seus requerimentos de segurança.

Nesse sentido, para que se pudesse introduzir um método eficaz de proteção da privacidade em SaaS, foi necessária uma análise de como o consumo de serviços neste paradigma afeta os interesses de um consumidor.

Primeiramente, foram delineadas as características do ambiente da computação em nuvem, apresentando-se os aspectos de segurança específicos dessa nova arquitetura, que, por sua vez, influenciam qualquer abordagem de segurança a ser desenhada.

Em seguida, descreveu-se um cenário modelo de consumo de serviços SaaS, mostrando como acontecem as interações típicas entre consumidor e provedor. Isso possibilitou, juntamente com a caracterização inicial da computação em nuvem, a verificação de que o provedor é a entidade com maior poder de ameaça à privacidade do consumidor, ao mesmo tempo em que é a entidade que, dentro do modelo SaaS, inerentemente precisa ter acesso a todos os itens de interesse privados do consumidor, pois efetivamente deve prover o serviço em questão.

Mais ainda, nesse contexto, foi concebida a primeira contribuição original da tese, a análise de privacidade do consumidor SaaS. Nela, estabeleceu-se que em cada nível de interação consumidor-provedor (contrato, troca de mensagens e comunicações de rede) existem itens de interesse que efetivamente, ao serem expostos ao provedor, afetam a

privacidade do consumidor, cada um de maneira diferente; e essa forma de influência pode ser classificada em termos da dimensão de privacidade sendo afetada.

Com essa análise de privacidade fica destacada a limitação de privacidade do cenário modelo de consumo de serviços SaaS: manter privadas informações que são justamente a matéria prima para o provimento dos serviços. Entretanto, o anonimato se propõe exatamente a possibilitar interações, enquanto resguarda a privacidade das entidades envolvidas. E, por isso, torna-se ele a tecnologia mais adequada para a situação em questão.

Com o anonimato selecionado como a tecnologia apropriada para a abordagem da privacidade, restou a definição de como ele seria empregado. Para isso, a análise promovida das interações entre consumidor e provedor também foi pertinente, instituindo formalmente os níveis de interação, os respectivos itens de interesse e as dimensões de privacidade a serem protegidas. A abordagem modular foi preferida, isolando-se cada nível de interação e buscando-se a tecnologia apropriada para seus respectivos itens de interesse.

Surgiu a segunda contribuição original da tese: a estrutura conceitual multicamadas de anonimato para SaaS. Ela funda quatro camadas: contrato, metadados, rede e conteúdo. Cada camada é voltada para itens de interesse específicos da interação consumidor-provedor, permitindo o emprego de tecnologias de anonimato adequadas para cada situação. Apresentou-se para cada camada pelo menos uma tecnologia de anonimato que pudesse proteger os itens de interesse nela presentes. Em determinadas camadas, como, por exemplo, a de metadados, ofereceu-se mais de uma opção de tecnologia, possibilitando o anonimato dos mesmos itens de interesse, mas com características e propriedades diferentes, como a possibilidade de rastreabilidade. Neste contexto, a modularidade da estrutura também permite que, em caso de necessidades de privacidade diferentes, um consumidor possa optar por trocar a tecnologia presente em uma camada, sem influenciar na proteção em efetividade em outra camada.

A estrutura multicamadas também permitiu a terceira contribuição original desta tese, que são os esquemas práticos para consumo anônimo de serviços. Um esquema prático

é exatamente uma instância da estrutura, selecionando uma tecnologia para cada camada e apresentando um design de transações que suporta a operação dessas tecnologias em conjunto.

Foram propostos aqui dois esquemas práticos, mas um consumidor específico pode instanciar a estrutura multicamadas e efetivar o projeto de outro, mais ajustado às suas necessidades de privacidade e características de consumo, podendo inclusive abdicar da proteção de uma camada que não seja imprescindível para seus interesses. A modularidade da estrutura multicamadas, todavia, estabelece como cada item de interesse impacta a privacidade do consumidor e como ele pode ser abordado, caso esse impacto precise ser mitigado.

O primeiro esquema prático propõe o anonimato rastreável. A rastreabilidade, ou seja, a possibilidade de identificação de um consumidor anônimo em casos especiais, é permitida pela seleção da tecnologia de assinaturas de grupo para a camada de metadados. Essa propriedade concilia os interesses de privacidade do consumidor com eventuais requisitos legais de identificação de comportamentos maliciosos, que poderiam abusar do anonimato presente para lesar o provedor impunemente.

Ao alterar o uso de assinaturas de grupo na camada de metadados pelo emprego de dinheiro eletrônico, estabeleceu-se o segundo esquema prático. O dinheiro eletrônico também mantem os itens de interesse referentes à camada de metadados privados, mas, por sua vez, não admite a rastreabilidade em hipótese alguma. Apesar de esse esquema prático ser focado para ambientes bem específicos, onde o provedor deverá se resguardar de eventuais consumidores mal intencionados, o anonimato absoluto torna-se uma opção real aos consumidores mais exigentes.

A título de análise da eficácia de cada esquema, realizou-se, para cada um, uma avaliação de privacidade. Percorreram-se as camadas de anonimato e verificou-se como os itens de interesse respectivos do consumidor eram abordados e protegidos pelo design de transações. Ao final das avaliações, nas Tabelas 4.1 e 5.1, resumiu-se o efeito das tecnologias das camadas nos itens de interesse privados, delineando o impacto que tanto o provedor quanto o TPB podem reter durante a utilização dos esquemas. Para

ambos os esquemas, ao provedor, mesmo que consiga manter algum acesso às informações, especialmente no contexto do provimento do serviço, não é possível a utilização desse privilégio para afetar a privacidade dos itens de interesse protegidos. A privacidade do consumidor é mantida em relação ao provedor.

De relevância para o trabalho, no Capítulo 6 da tese, promoveu-se a validação experimental das contribuições originais da tese. Em específico, optou-se por implementar em *software* um produto que possibilitou a materialização do esquema prático rastreável para consumo anônimo de serviços SaaS. Durante o Capítulo pode-se verificar, na prática, como a análise de privacidade do cenário modelo SaaS previu corretamente como dos dados concretos que representam os itens de interesse sendo trocados entre consumidor e provedor realmente expõem a privacidade do consumidor, impactando suas dimensões de privacidade. E, mais importante, demonstrou, em um ambiente simulado, a aplicabilidade e a eficácia da estrutura multicamadas de anonimato e como o conceito dos esquemas práticos pode ser empregado para, com seus designs de transações, instanciar e concretizar a proteção de privacidade do consumidor SaaS.

A abordagem da privacidade aqui proposta é, no melhor conhecimento do autor, inédita. Abranger todos os níveis de interação entre consumidor e provedor não é enfoque utilizado pelos trabalhos relacionados. Como já se discutiu durante a Introdução, trabalhos como o artigo preliminar de Jensen, Schäge e Schwenk, "*Towards an anonymous access control and accountability scheme for cloud computing*" [67] e o "*Providing privacy preserving in cloud computing*" [132], de Wang, Zhao, Jiang e Le, focam-se em aspectos limitados da privacidade total do consumidor. O artigo "*Towards an anonymous access control and accountability scheme for cloud computing*" [67], por exemplo, não considera as interações consumidor-provedor no nível das comunicações de rede, não apresenta um esquema prático com design das transações, e oferece apenas uma proposta voltada para o anonimato rastreável. Já o de Wang, Zhao, Jiang e Le [132] é voltado apenas para o anonimato de dados da computação em nuvem, excluindo, portanto, o anonimato de conexão. De destacar, também, em nenhum desses trabalhos correlatos foi apresentada uma validação experimental, como a proposta nesta tese.

Funda-se, durante esta tese, portanto, a primeira abordagem compreensiva de todos os aspectos de privacidade envolvidos no consumo de serviços SaaS da computação em nuvem. A estrutura multicamadas mapeia todos os itens de interesse a serem protegidos, oferecendo técnicas de anonimato adequadas para cada camada conceitual, sendo modular e flexível. Ela permite a escolha ou troca de técnicas de anonimato para atender interesses diversos dos consumidores. Já os esquemas práticos instanciam a estrutura multicamadas e, com seu desenho de transações, comportam a compatibilização das técnicas de anonimato da estrutura.

No que diz respeito às publicações referentes ao trabalho realizado, três artigos já foram publicados [99] [101] [100], além de um livro conceitual, extrapolando o ambiente de pesquisa e voltado também para a indústria, com publicação para maio de 2013 [51].

Finalmente, como trabalho futuro, pretende-se comercialização do produto criado e empregado na validação experimental da tese. Buscar-se-á parcerias com provedores de serviços em nuvens públicas que queiram disponibilizar a opção de acesso anônimo a seus consumidores, capitalizando a demanda reprimida do mercado por privacidade na computação em nuvem. Espera-se que a solução possa permitir uma maior adoção da computação em nuvem, além de fortalecer a segurança dos ambientes SaaS na Internet.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] 1105 Government Information Group. (2012). “SaaS remains the biggest cloud.” Mainstream Cloud Computing, Online report, BROCADE.

[2] 1105 Government Information Group. (2012). “Security still a hurdle.” Mainstream Cloud Computing, Online report, BROCADE.

[3] Abe, M. (2001). “A secure three-move blind signature scheme for polynomially many signatures.” Proc. Eurocrypt'01, LNCS 2045.

[4] Adib, M. e Brohi, S. (2011). “Exploring the Cloud Deployment and Service Delivery Models.” International Journal of Research and Reviews in Information Sciences (IJRRIS), Vol. 1, No 3.

[5] Akagi, N., Manabe, Y. e Okamoto, T. (2008). “An Efficient Anonymous Credential System.” Financial Cryptography and Data Security. Book, pages 272 – 286. Springer-Verlag. ISBN 978-3-540-85229-2.

[6] Ateniese, G. e De Medeiros, B. (2003). “Efficient Group Signatures without Trapdoors.” ASIACRYPT 2003, vol. 2894 of LNCS, Springer, 2003, pp. 246–268.

[7] Ateniese, G. e Tsudik, G. (1999). “Group Signatures á la carte.” ACM-SIAM Symposium on Discrete Algorithms (SODA 1999), ACM, 1999, pp. 848–849.

[8] Badger, L., Bohn, R., Chu, S., Hogan, M., Liu, F., Kaufmann, V., Mao, J., Messina, J., Mills, K., Sokol, A., Tong, J., Whiteside, F. e Leaf, D. (2011). “US Government Cloud Computing Technology Roadmap, Volume II, Release 1.0 (Draft).” NIST Special Publication 500-293.

[9] Baker, M. (2000). “Cluster Computing White Paper.” Cornell University Library, CoRR, cs.DC/0004014.

[10] Baldimtsi, F., Hinterwalder, G., Rupp, A., Lysyanskaya, A., Paar, C. e Burleson, W. (2012). “Pay as you go.” Workshop on hot topics in privacy enhancing technologies, HotPETSs 2012, Vigo, Spain.

[11] Baldimtsi, F. e Lysyanskaya, A. (2012). “*On the security of one-witness blind signature schemes*.” Cryptology ePrint Archive, Report 2012/197, 2012. http://eprint.iacr.org/.

[12] Barker, W. e Barker, E. (2012). “Recommendation for the Triple Data Encryption

Algorithm (TDEA) Block Cipher.” NIST Special Publication 800-67.

[13] Barreto, P. e Naehrig, M. (2006). “Pairing-friendly elliptic curves of prime order.” B. Preneel and S. Tavares, editors, Proceedings of SAC 2005, volume 3897 of LNCS, pages 319–31. Springer-Verlag.

[14] Bauer, K., Mccoy, D., Grunwald, D., Kohno, T. e Sicker, D. (2007). “Low-Resource Routing Attacks Against Tor.” Proceedings of the 2007 ACM workshop on Privacy in electronic society.

[15] Bellare, M., Micciancio, D. e Warinschi, B. (2003). “Foundations of group signatures: formal definitions, simplified requirements, and a construction based on general assumptions.” Proc. Advances in Cryptology - Eurocrypt 2003, LNCS Vol. 2656, Springer-Verlag.

[16] Bellare, M. e Rogaway, P. (1993). “Random oracles are practical: A paradigm for designing efficient protocols.” D. Denning, R. Pyle, R. Ganesan, R. Sandhu, and V. Ashby, editors, Proceedings of CCS 1993, pages 62–73. ACM Press.

[17] Bellare, M., Shi, H. e Zhang, C. (2005). “Foundations of group signatures: The case of dynamic groups.” A. J. Menezes, editor, Proceedings of CT-RSA 2005, LNCS vol. 3376, pp. 136-153. Springer-Verlag.

[18] Beresford, A. e Stajano, F. (2003). “Location privacy in pervasive computing.” IEEE Pervasive Computing journal, Volume 2 Issue 1.

[19] Bichsel, P., Camenisch, J., Neven, G., Smart, N. e Warinschi, B. (2010). “Get Shorty via Group Signatures without Encryption.” International Conference on Security and Cryptography for Networks (SCN 2010), vol. 6280 of LNCS, Springer, 2010, pp. 381–398.

[20] Boneh, D. e Boyen, X. (2004). “Short signatures without random oracles.” C. Cachin and J. Camenisch, editors, Proceedings of Eurocrypt 2004, volume 3027 of LNCS, pages 56–73. Springer-Verlag.

[21] Boneh, D., Boyen, X. e Shacham, H. (2004). “Short group signatures.” M. Franklin, editor, Proceedings of Crypto 2004, volume 3152 of LNCS, pages 41–55. Springer-Verlag.

[22] Boneh, D., Lynn, B. e Shacham, H. (2001). “Short signatures from the Weil pairing.” J. Cryptology, 17(4):297–319, Sept. 2004. Extended abstract in Proceedings of Asiacrypt 2001.

[23] Boneh, D. e Shacham, H. (2004). “Group signatures with verifier-local revocation.” B. Pfitzmann and P. Liu, editors, Proceedings of CCS 2004, pages 168-177. ACM Press.

[24] Brand, J. (2009). “Cloud Computing: Worthy Of Definition(s).” FORRESTER REPORT, http://www.forrester.com/home#/Cloud+Computing+Worthy+Of+Definitions/fulltext/-/E-RES59588.

[25] Brands, S. (1993). “Untraceable off-line cash in wallets with observers.” CRYPTO’93, Springer-Verlag, pages 302—318.

[26] Brickell, E., Camenisch, J., Chen, L. (2004). “Direct anonymous attestation.” Proceedings of 11th ACM Conference on Computer and Communications Security, ACM Press.

[27] Brown, Z. (2002). “Cebolla: Pragmatic IP Anonymity.” Ottawa Linux Symposium.

[28] Camenisch, J. e Groth, J. (2004). “Group Signatures: Better Efficiency and New Theoretical Aspects.” International Conference on Security in Communication Networks (SCN 2004), vol. 3352 of LNCS, Springer, 2004, pp. 120–133.

[29] Camenisch, J., Hohenberger, S. e Lysyanskaya, A. (2005). “Compact E-Cash.” Lecture Notes on Computer Science, Eurocrypt 2005, pages 302-321, Springer Verlag.

[30] Camenisch, J. e Lysyanskaya, A. (2004). “Signature schemes and anonymous credentials from bilinear maps.” CRYPTO’04, LNCS, Vol.3152 56–72.

[31] Carty, G. (2002). “Broadband Networking.” McGraw-Hill/Osborne. p. 4. ISBN 007219510X.

[32] Cerf, V., Dalal, Y. e Sunshine, C. (1974). “RFC 675: Specification of Internet Transmission Control Program.” IETF Request For Comments 675.

[33] Chaum, D. (1983). “Blind signature systems.” David Chaum, editor, Advances in Cryptology — CRYPTO ’83, page 153. Plenum Press.

[34] Chaum, D. (1982). “Blind signatures for untraceable payments.” David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, Advances in Cryptology — CRYPTO ’82, pages 199– 203.

[35] Chaum, D. (1989). “Online cash checks.” Jean-Jacques Quisquater and Joos Vandewalle, editors, Advances in Cryptology — EUROCRYPT ’89, volume 434 of LNCS, pages 289–3293. Springer Verlag.

[36] Chaum, D. (1985). “Security without identification: transaction systems to make big brother obsolete.” Commun. ACM. 28(10) 1030-1044.

[37] Chaum, D. (1981). “Untraceable electronic mail, return addresses, and digital pseudonyms.” Communications of the ACM, Volume 24, Number 2.

[38] Chaum, D. e Van Heyst, E. (1991). “Group signatures.” D. W. Davies, editor, Proceedings of Eurocrypt 1991, LNCS vol. 547 pp. 257-65, Springer-Verlag.

[39] Christensen, E., Curbera, F., Meredith, G. e Weerawarana, S. (2001). “Web Services Description Language (WSDL) 1.1.” W3C Note 15, http://www.w3.org/TR/wsdl.

[40] Clarke, R. (1997). “Introduction to dataveillance and information privacy, and definitions of terms.” http://www.rogerclarke.com/DV/Intro.html.

[41] Cloud Computing Alliance^{SM} – CSA. (2011). “Security guidance for critical areas of focus in cloud computing V3.0.” https://cloudsecurityalliance.org/research/security-guidance/.

[42] Cloud Computing Alliance^{SM} – CSA. (2010). “Top threats to cloud computing V1.0.” https://cloudsecurityalliance.org/research/initiatives/top-threats-to-cloud-computing/.

[43] Damgård, I. e Jurik, M. (2001). “A Generalization, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System.” Public Key Cryptography 2001: 119-136.

[44] Danezis, G., Dingledine, R. e Mathewson, N. (2003). “Mixminion: Design of a type III anonymous remailer protocol.” 2003 IEEE Symposium on Security and Privacy, pages 2-15. IEEE CS.

[45] Dawson, M. (2010). “SaaS Solutions Lead the Pack in Cloud Implementations.” Hubspan, October 2010, http://liaison.com/blog/blank-blog-page/liaison-blog/2010/10/07/SaaS-Solutions-Lead-the-Pack-in-Cloud-Implementations.

[46] Díaz, C., Seys, S., Claessens, J. e Preneel, B. (2002). “Towards measuring anonymity.” Proc. 2nd international conference on Privacy enhancing technologies.

[47] Dingledine, R., Mathewson, N. e Syverson, P. (2004). “Tor: The second-generation onion router.” Proc. 13th USENIX Security Symposium. USENIX Association.

[48] Eastlake, D. e Jones, P. (2001). “RFC 3174: US Secure Hash Algorithm 1

(SHA1).” IETF Request For Comments 3174.

[49] Erl, T. (2005). “Service Oriented Architecture: Concepts, Technology, and Design.” Indiana: Pearson Education, pp. 171, ISBN 0-13-185858-0.

[50] Erl, T. (2009). “SOA Design Patterns.” Prentice Hall/PearsonPTR, ISBN 0-13-613516-1.

[51] Erl, T., Mahmood, Z. e Puttini, R. (2013). “Cloud Computing: Concepts, Technology & Architecture.” Prentice Hall, ISBN-13: 978-0133387520.

[52] Fielding, R., Gettys, J., Mogul, J., Nielsen, H., Masinter, L., Leach, P. e Berners-Lee, T. (1999). “RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1.” IETF Request For Comments 2616.

[53] Foster, I. e Kesselman, C. (1999). “The Grid: Blueprint for a New Computing Infrastructure.” Morgan Kaufmann Publishers, ISBN 1-55860-475-8.

[54] Furukawa, J. e Yonezawa, S. (2004). “Group Signatures with Separate and Distributed Authorities.” International Conference on Security in Communication Networks (SCN 2004), vol. 3352 of LNCS, Springer, 2004, pp. 77–90.

[55] Gartner. (2013). “Cloud Computing.” Gartner IT Glossary – Cloud Computing Definition, Defining IT Industry, http://www.gartner.com/it-glossary/cloud-computing/.

[56] Gentry, C. (2009). “Fully homomorphic encryption using ideal lattices.” STOC, pages 169–178.

[57] Gentry, C. e Halevi, S. (2011). “Implementing Gentry’s fully-homomorphic encryption scheme.” EUROCRYPT'11 Proceedings of the 30th Annual international conference on Theory and applications of cryptographic techniques: advances in cryptology Pages 129-148.

[58] Goldschlag, D., Reed, M. e Syverson, P. (1996). “Hiding Routing Information.” Proc. Information Hiding, pp.137-150.

[59] Gülcü, C. e Tsudik, G. (1996). “Mixing E-mail with Babel.” Network and Distributed Security Symposium (NDSS 96), pages 2-16. IEEE.

[60] Hardy, G. e Wright, E. (1979). “An Introduction to the Theory of Numbers.” 5th ed. Oxford, England: Clarendon Press, pp. 7-8.

[61] Herstein, I. (1996) “*Abstract algebra.*” 3rd ed. Prentice Hall, ISBN 978-0-13-

374562-7, MR 1375019.

[62] IBM. (2013). "*IBMSmartCloud*." IBMSmartCloud, http://www.ibm.com/cloud-computing/us/en/index.html.

[63] IEEE P1556 Working Group, VSC Project. (2003). "Dedicated short range communications (DSRC)."

[64] International Telecommunication Union. (2005). "Recommendation X.509 (2005)." ITU-T Information Technology - Open Systems Interconnection - The Directory: Authentication Framework.

[65] Ishai, Y., Kushilevitz, E., Ostrovsky, R. e Sahai, A. (2006). "Cryptography from anonymity." Proc. 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), pp.239-248.

[66] ITU-T. (1993). "Recommendation X.210. Basic Reference Model: Conventions for the Definition of OSI Services."

[67] Jensen, M., Schäge, S. e Schwenk, J. (2010). "Towards an anonymous access control and accountability scheme for cloud computing." Proc. 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD).

[68] Kiayias, A. e Yung, M. (2005). "Efficient Secure Group Signatures with Dynamic Joins and Keeping Anonymity Against Group Managers." Mycrypt 2005, vol. 3715 of LNCS, Springer, 2005, pp. 151–170.

[69] Lauter, K., Naehrig, M. e Vaikuntanathan, V. (2011). "Can Homomorphic Encryption be Practical?" Microsoft Research – TechReport, MSR-TR-2011-61.

[70] Levine, B. e Shields, C. (2002). "Hordes: A multicast-based protocol for anonymity." Journal of Computer Security, 10(3):213– 240.

[71] Liu, J., Lu, Y. e Koh, C. (2010). "Performance Analysis of Arithmetic Operations in Homomorphic Encryption." Purdue University – ECE Technical Reports – TR-ECE-10-08.

[72] Lysyanskaya, A. (2001). "An efficient system for non-transferable anonymous credentials with optional anonymity revocation." EUROCRYPT '01 Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology. Pages 93-118.

[73] Lysyanskaya, A. (2002). "Signature Schemes and Applications to Cryptographic

Protocol Design.” Ph.D thesis, Massachusetts Institute of Technology, Cambridge, MA, USA.

[74] Manulis, M., Fleischhacker, N., Günther, F., Kiefer, F. e Poettering, B. (2012). “Group Signatures: Authentication with Privacy.” Federal Office for Information Security - Study, Cryptographic Protocols Group, Department of Computer Science, Technische Universität Darmstadt, Germany.

[75] Mchall, T. (2011). “Gartner Says Worldwide Software as a Service Revenue Is Forecast to Grow 21 Percent in 2011.” Gartner.com. Gartner.

[76] Meiklejohn, S. (2011). “An Exploration of Group and Ring Signatures.” UCSD Research Exam.

[77] Mell, P. e Grance, T. (2011). “The NIST Definition of Cloud Computing.” NIST Special Publication 800-145.

[78] Menezes, A., Van Oorschot, P. e Vanstone, S. (1996). “Handbook of Applied Cryptography.” CRC Press.

[79] Micciancio, D. (2010). “A first glimpse of cryptography’s Holy Grail.” C. ACM, 53(3):96.

[80] Minggiang, X., Papadimitriou, P., Raissi, C., Kalnis, P. e Pung, H. (2009). “Distributed Privacy Preserving Data Collection using Cryptographic Techniques.” Stanford University, Stanford InfoLab – Technical Report.

[81] Miyaji, A., Nakabayashi, M. e Takano, S. (2001). “New explicit conditions of elliptic curve traces for FR-reduction.” IEICE Trans. Fundamentals, E84-A(5):1234–43.

[82] Möller, U., Cottrell, L., Palfrader, P. e Sassaman, L. (2003). “Mixmaster Protocol - Version 2.” Draft, http://www.freehaven.net/anonbib/cache/mixmaster-spec.txt.

[83] Mortier, R., Madhavapeddy, A., Hong, T., Murray, D. e Schwarzkopf, M. (2010). “Using dust clouds to enhance anonymous communication.” Proc. 18th International Workshop on Security Protocols, Cambridge, UK.

[84] Murdoch, S. e Danezis, G. (2005). “Low-cost traffic analysis of Tor.” Security and Privacy, 2005 IEEE Symposium.

[85] Naccache, D. e Stern, J. (1998). “*A new public key cryptosystem based on higher residues*.” CCS '98 Proceedings of the 5th ACM conference on Computer and

communications security.

[86] Nakanishi, T. e Funabiki, N. (2005). “Verifier-Local Revocation Group Signature Schemes with Backward Unlinkability from Bilinear Maps.” ASIACRYPT 2005, vol. 3788 of LNCS, Springer, 2005, pp. 533–548.

[87] Narayanan, A. e Shmatikov, V. (2008). “Robust De-anonymization of Large Sparse Datasets.” SP '08 Proceedings of the 2008 IEEE Symposium on Security and Privacy. Pages 111-125.

[88] National Security Agency (NSA). (2009). “An Overview of Cloud Computing.” The Next Wave – Vol 17, No 4, The National Security Agency’s Review of Emerging Technologies.

[89] Nin, J. e Herranz, J. (2010). “Privacy and Anonymity in Information Management Systems: New Techniques for New Practical Problems.” Springer. ISBN 978-1-84996-237-7. 198 pages.

[90] OASIS UDDI Spec Technical Committee. (2004). “UDDI Version 3.0.2.” http://uddi.org/pubs/uddi_v3.htm.

[91] OASIS Web Services Reliable Exchange (WS-Rx) Technical Committee. (2009). “Web Services Reliable Messaging (WS-ReliableMessaging) Version 1.2.” http://docs.oasis-open.org/ws-rx/wsrm/200702/wsrm-1.2-spec-os.pdf.

[92] OASIS Web Services Security Technical Committee. (2004). “Web services security: SOAP Message Security 1.1 (WS-Security 2004).” http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf.

[93] OASIS Web Services Security Technical Committee. (2006). “Web Services Security UsernameToken Profile 1.1.” https://www.oasis-open.org/committees/download.php/16782/wss-v1.1-spec-os-UsernameTokenProfile.pdf.

[94] OASIS Web Services Security Technical Committee. (2006). “Web Services Security Kerberos Token Profile 1.1.” http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-KerberosTokenProfile.pdf.

[95] OASIS Web Services Security Technical Committee. (2006). “X.509 Certificate Token Profile 1.1.” http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-x509TokenProfile.pdf.

[96] Okamoto, T. (1995). “An Efficient Divisible Electronic Cash Scheme.” Advances

in Cryptology - Crypto'95, volume 963 of LNCS, pages 438-451.

[97] Okamoto, T. e Ohta, K. (1992). “Universal Electronic Cash.” Advances in Cryptology – Crypto ’91, page 324-325, Springer-Verlag.

[98] Okamoto, T. e Uchiyama, S. (1998). “A new public-key cryptosystem as secure as factoring.” Advances in Cryptology — Eurocrypt'98, Lecture Notes in Computer Science, Volume 1403/1998, 308-318.

[99] Pacheco, V. e Puttini, R. (2012). “Untraceable Anonymous Service Consumption in SaaS.” Frank Leymann, Ivan Ivanov, Marten van Sinderen & Tony Shan, ed., 'CLOSER' , SciTePress, pp. 96-101.

[100] Pacheco, V. e Puttini, R. (2012). “Defining and Implementing Connection Anonymity for SaaS Web Services.” Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on: 479-486.

[101] Pacheco, V. e Puttini, R. (2012). “SaaS Anonymous Cloud Service Consumption Structure.” ICDCS Workshops, The Third International Conference on Security and Privacy in Cloud Computing, 2012: 491-499.

[102] Pailles, J. (1993). “New Protocols for Electronic Money.” Advances in Cryptology - Asiacrypt'92, volume 718 of LNCS, pages 263-274.

[103] Pfitzman, A. e Hansen, M. (2010). “*A terminology for talking about privacy by data minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management*.” http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_ v0.34.pdf. IETF Internet Draft.

[104] Plummer, D., Lheureux, B. e Karamouzis, F. (2010). “Defining cloud services brokerage: taking intermediation to the next level.” Gartner Research Note n. G00206187, Gartner Group.

[105] Ragouzis, N., Hughes, J., Philpott, R., Maler, E., Madsen, P. e Scavo, T. (2008). “Security Assertion Markup Language (SAML) V2.0 Technical Overview.” OASIS Committee Draft, March 2008. Document ID sstc-saml-tech-overview-2.0-cd-02 http://www.oasis-open.org/committees/download.php /27819/sstc-saml-tech-overview-2.0-cd-02.pdf.

[106] Reed, M., Syverson, P. e Goldschlag, D. (1998). “Anonymous connections and onion routing.” IEEE Journal on Selected Areas in Communications, Volume 16, Issue 4.

[107] Reiter, M. e Rubin, A. (1998). “Crowds: Anonymity for web transactions.” ACM TISSEC, 1(1):66–92.

[108] Rivest, R., Adleman, L. e Dertouzos, M. (1978). “On data banks and privacy homomorphisms.” Foundations of Secure Computation, pages 169–179.

[109] Rodriguez, A. (2008). “RESTful Web services: The basics.” IBM Technical Library Article, https://www.ibm.com/developerworks/webservices/library/ws-restful/.

[110] RSA Laboratories. (2012). “PKCS #1: v2.2 RSA Cryptography Standard.” RSA Laboratories – Public-Key Cryptography Standards (PKCS). http://www.rsa.com/rsalabs/pkcs/files/h11300-wp-pkcs-1v2-2-rsa-cryptography-standard.pdf.

[111] Samarati, P. (2001). “Protecting respondents’ identities in microdata release.” IEEE Transactions on Knowledge and Data Engineering, Volume 13, issue 6.

[112] Sander, R. e Borisov, N. (2008). “A Tune-up for Tor: Improving Security and Performance in the Tor Network.” Proceedings of the Network and Distributed Security Symposium - NDSS '08.

[113] Scarfone, K., Souppaya, M. e Hoffman, P. (2011). “Guide to Security for Full Virtualization Technologies.” NIST Special Publication 800-125.

[114] Schwartz, E., Brumley, D. e Mccune, J. (2010). “Contractual anonymity.” Proceedings of NDSS 2010.

[115] Sedayao, J. (2012). “Enhancing Cloud Security Using Data Anonymization.” Intel IT, IT@Intel White Paper. IT Best Practices, Cloud Computing and Information Security.

[116] Smart, N. e Vercauteren, F. (2011). “Fully homomorphic simd operations.” Cryptology ePrint Archive, Report 2011/133, 2011. http://eprint.iacr.org/2011/133.

[117] Sweeney, L. (2002). “K-anonymity: a model for protecting privacy.” International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10 (5), 2002, 557-570.

[118] Syverson, P., Reed, M. e Goldschlag, D. (2000). “Onion Routing access configurations.” DARPA Information Survivability Conference and Exposition (DISCEX 2000), volume 1, pages 34-40. IEEE CS Press.

[119] Syverson, P., Tsudik, G., Reed, M. e Landwehr, C. (2001). “Towards an Analysis of Onion Routing Security.” Designing Privacy Enhancing Technologies: Workshop on Design Issue in Anonymity and Unobservability, pages 96-114. Springer-Verlag.

[120] Tebaa, M., El Hajji, S. e El Ghazi, A. (2012). “Homomorphic Encryption Applied to the Cloud Computing Security.” Proceedings of the World Congress on Engineering 2012 Vol I WCE 2012, July 4 - 6, 2012, London, U.K.

[121] Telecommunications Industry Association. (2010). “Telecommunications Infrastructure Standard for Data Centers.” TIA-942.

[122] The European Parliament. (1995). “Directive 95/46/EC.” Official Journal L 281, 23/11/1995 P. 0031 – 0050, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:HTML.

[123] Tor Project Inc. (2013). “Tor Project Online.” https://www.torproject.org.

[124] Tor Project: Anonymity Online. (2012). “Tor Cloud.” Tor Cloud, https://cloud.torproject.org/.

[125] Tsudik, G. e Xu, S. (2003). “Accumulating Composites and Improved Group Signing.” ASIACRYPT 2003, vol. 2894 of LNCS, Springer, 2003, pp. 269–286.

[126] Van Dijk, M. e Juels, A. (2010). “On the impossibility of cryptography alone for privacy-preserving cloud computing.” Proceedings of the 5th USENIX Conference on Hot Topics in Security, pages 1–8.

[127] W3C. (2002). “Exclusive XML Canonicalization Version 1.0.” W3C Recommendation, http://www.w3.org/TR/xml-exc-c14n/.

[128] W3C. (2008). “Extensible Markup Language (XML) 1.0 (Fifth Edition).” W3C Recommendation, http://www.w3.org/TR/xml/.

[129] W3C. (2007). “SOAP Version 1.2.” W3C Recommendation 27 http://www.w3.org/TR/2007/REC-soap12-part0-20070427/.

[130] W3C. (2012). “W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures.” W3C Recommendation, http://www.w3.org/TR/xmlschema11-1/.

[131] W3C. (2004). “Web Services Glossary.” W3C Working Group Note 11.

[132] Wang, J., Zhao, Y., Jiang, S. e Le, J. (2009). “Providing privacy preserving in cloud computing.” International Conference on Test and Measurement 2009 -

ICTM '09, pages 213-216, Hong Kong, 10.1109/ICTM.2009.5413073.

APÊNDICES

A – WEB SERVICE PUBLISHER – *PUBLISHER.WSDL*

Publisher

```
<?xml version="1.0" encoding="UTF-8" ?>
<definitions targetNamespace="urn:publisher"
             xmlns="http://schemas.xmlsoap.org/wsdl/"
             xmlns:tns="urn:publisher"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
             xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
             xmlns:wvc="http://www.viniciuspacheco.com">
  <types>
    <xsd:schema targetNamespace="urn:publisher/types"
                elementFormDefault="qualified"/>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/wsdl.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/group_signatures_VLR-gpk.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:element name="status">
        <complexType>
            <all>
                <element name="value" type="string"/>
            </all>
        </complexType>
    </xsd:element>
    <xsd:element name="N-service-credit">
        <complexType>
            <all>
                <element name="value" type="integer"/>
            </all>
        </complexType>
    </xsd:element>
  </types>
  <message name="publish_service-request">
    <part name="wsdl" element="wvc:wsdl"/>
    <part name="N-service-credit" type="xsd:integer"/>
  </message>
  <message name="publish_service-response">
    <part name="status" type="xsd:string"/>
    <part name="group_signatures_VLR-gpk"
element="wvc:group_signatures_VLR-gpk"/>
  </message>
  <message name="add_service_credit-request">
    <part name="group_signatures_VLR-gpk"
element="wvc:group_signatures_VLR-gpk"/>
    <part name="N-service-credit" type="xsd:integer"/>
  </message>
  <message name="add_service_credit-response">
    <part name="status" type="xsd:string"/>
```

```
  </message>
  <portType name="publisher">
    <operation name="publishservice">
      <input message="tns:publish_service-request"/>
      <output message="tns:publish_service-response"/>
    </operation>
    <operation name="addservicecredit">
      <input message="tns:add_service_credit-request"/>
      <output message="tns:add_service_credit-response"/>
    </operation>
  </portType>
  <binding name="publisherSOAP11Binding" type="tns:publisher">
    <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="publishservice">
      <soap:operation style="document"
soapAction="urn:publisher/publishservice"/>
      <input>
        <soap:body use="literal" parts="wsdl"/>
        <soap:body use="literal" parts="N-service-credit"/>
      </input>
      <output>
        <soap:body use="literal" parts="status"/>
        <soap:body use="literal" parts="group_signatures_VLR-gpk"/>
      </output>
    </operation>
    <operation name="addservicecredit">
      <soap:operation style="document"
soapAction="urn:publisher/publishservice"/>
      <input>
        <soap:body use="literal" parts="group_signatures_VLR-gpk"/>
        <soap:body use="literal" parts="N-service-credit"/>
      </input>
      <output>
        <soap:body use="literal" parts="status"/>
      </output>
    </operation>
  </binding>
  <service name="publisher">
    <port name="publisherPort" binding="tns:publisherSOAP11Binding">
      <soap:address location="http://www.viniciuspacheco.com"/>
    </port>
  </service>
</definitions>
```

B – WEB SERVICE SUBSCRIBER – *SUBSCRIBER.WSDL*

Subscriber

```
<?xml version="1.0" encoding="UTF-8" ?>
<definitions targetNamespace="urn:subscriber"
             xmlns="http://schemas.xmlsoap.org/wsdl/"
             xmlns:tns="urn:subscriber"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
             xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
             xmlns:wvc="http://www.viniciuspacheco.com">
  <types>
    <xsd:schema targetNamespace="urn:subscriber/types"
                elementFormDefault="qualified"/>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/group_signatures_VLR-gpk.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/payment-information.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/group_signatures_VLR-gsk.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/service-credits.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:element name="status">
        <complexType>
            <all>
                <element name="value" type="string"/>
            </all>
        </complexType>
    </xsd:element>
    <xsd:element name="consumer-id">
        <complexType>
            <all>
                <element name="value" type="string"/>
            </all>
        </complexType>
    </xsd:element>
    </types>
  <message name="subscribe_service-request">
    <part name="group_signatures_VLR-gpk"
element="wvc:group_signatures_VLR-gpk"/>
    <part name="payment-information" element="wvc:payment-
information"/>
  </message>
  <message name="subscribe_service-response">
    <part name="status" type="xsd:string"/>
```

```
    <part name="group_signatures_VLR-gsk"
element="wvc:group_signatures_VLR-gsk"/>
    <part name="service-credits" element="wvc:service-credits"/>
  </message>
  <message name="buy_service_credits-request">
    <part name="group_signatures_VLR-gpk"
element="wvc:group_signatures_VLR-gpk"/>
  </message>
  <message name="buy_service_credits-response">
    <part name="status" type="xsd:string"/>
    <part name="service-credits" element="wvc:service-credits"/>
  </message>
  <message name="report_abuse-request">
    <part name="group_signatures_VLR-gpk"
element="wvc:group_signatures_VLR-gpk"/>
    <part name="service-credits" element="wvc:service-credits"/>
  </message>
  <message name="report_abuse-response">
    <part name="status" type="xsd:string"/>
    <part name="consumer-id" type="xsd:string"/>
  </message>
  <portType name="subscriber">
    <operation name="subscribeservice">
      <input message="tns:subscribe_service-request"/>
      <output message="tns:subscribe_service-response"/>
    </operation>
    <operation name="buyservicecredits">
      <input message="tns:buy_service_credits-request"/>
      <output message="tns:buy_service_credits-response"/>
    </operation>
    <operation name="reportabuse">
      <input message="tns:report_abuse-request"/>
      <output message="tns:report_abuse-response"/>
    </operation>
  </portType>
  <binding name="subscriberSOAP11Binding" type="tns:subscriber">
    <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="subscribeservice">
      <soap:operation style="document"
soapAction="urn:subscriber/subscribeservice"/>
      <input>
        <soap:body use="literal" parts="group_signatures_VLR-gpk"/>
        <soap:body use="literal" parts="payment-information"/>
      </input>
      <output>
        <soap:body use="literal" parts="status"/>
        <soap:body use="literal" parts="group_signatures_VLR-gsk"/>
        <soap:body use="literal" parts="service-credits"/>
      </output>
    </operation>
    <operation name="buyservicecredits">
    <soap:operation style="document"
soapAction="urn:subscriber/buyservicecredits"/>
      <input>
        <soap:body use="literal" parts="group_signatures_VLR-gpk"/>
      </input>
      <output>
```

```
        <soap:body use="literal" parts="status"/>
        <soap:body use="literal" parts="service-credits"/>
      </output>
    </operation>
    <operation name="reportabuse">
      <soap:operation style="document"
soapAction="urn:subscriber/reportabuse"/>
      <input>
        <soap:body use="literal" parts="group_signatures_VLR-gpk"/>
        <soap:body use="literal" parts="service-credits"/>
      </input>
      <output>
        <soap:body use="literal" parts="status"/>
        <soap:body use="literal" parts="consumer-id"/>
      </output>
    </operation>
  </binding>
  <service name="subscriber">
    <port name="subscriberPort" binding="tns:subscriberSOAP11Binding">
      <soap:address location="http://www.viniciuspacheco.com"/>
    </port>
  </service>
</definitions>
```

C – WEB SERVICE DEPOSITER – *DEPOSITER.WSDL*

Depositer

```
<?xml version="1.0" encoding="UTF-8" ?>
<definitions targetNamespace="urn:depositer"
             xmlns="http://schemas.xmlsoap.org/wsdl/"
             xmlns:tns="urn:depositer"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
             xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
             xmlns:wvc="http://www.viniciuspacheco.com">
  <types>
    <xsd:schema targetNamespace="urn:depositer/types"
                elementFormDefault="qualified"/>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/group_signatures_VLR-gpk.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/db-service-billing-
provider.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/payment-information.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/receipt-information.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/revocation-information.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:element name="status">
        complexType>
            <all>
                <element name="value" type="string"/>
            </all>
        </complexType>
    </xsd:element>
    </types>
  <message name="deposit_credits-request">
    <part name="group_signatures_VLR-gpk"
element="wvc:group_signatures_VLR-gpk"/>
    <part name="db-service-billing-provider" element="wvc:db-service-
billing-provider"/>
    <part name="payment-information" element="wvc:payment-
information"/>
  </message>
  <message name="deposit_credits-response">
```

```
    <part name="status" type="xsd:string"/>
    <part name="receipt-information" element="wvc:receipt-
information"/>
    <part name="revocation-information" element="wvc:revocation-
information"/>
  </message>
  <portType name="depositer">
    <operation name="depositcredits">
      <input message="tns:deposit_credits-request"/>
      <output message="tns:deposit_credits-response"/>
    </operation>
  </portType>
  <binding name="depositerSOAP11Binding" type="tns:depositer">
    <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="depositcredits">
      <soap:operation style="document"
soapAction="urn:publisher/publishservice"/>
      <input>
        <soap:body use="literal" parts="group_signatures_VLR-gpk"/>
        <soap:body use="literal" parts="db-service-billing-provider"/>
        <soap:body use="literal" parts="payment-information"/>
      </input>
      <output>
        <soap:body use="literal" parts="status"/>
        <soap:body use="literal" parts="receipt-information"/>
        <soap:body use="literal" parts="revocation-information"/>
      </output>
    </operation>
  </binding>
  <service name="depositer">
    <port name="depositerPort" binding="tns:depositerSOAP11Binding">
      <soap:address location="http://www.viniciuspacheco.com"/>
    </port>
  </service>
</definitions>
```

D – WEB SERVICE INVOICER – *INVOICER.WSDL*

Invoicer

```
<?xml version="1.0" encoding="UTF-8" ?>
<definitions targetNamespace="urn:invoicer"
             xmlns="http://schemas.xmlsoap.org/wsdl/"
             xmlns:tns="urn:invoicer"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
             xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
             xmlns:wvc="http://www.viniciuspacheco.com">
  <types>
    <xsd:schema targetNamespace="urn:invoicer/types"
                elementFormDefault="qualified"/>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/group_signatures_VLR-gpk.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/db-service-billing-
consumer.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/invoice-information.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:element name="status">
        <complexType>
            <all>
                <element name="value" type="string"/>
            </all>
        </complexType>
    </xsd:element>
    </types>
  <message name="receive_invoice-request">
    <part name="group_signatures_VLR-gpk"
element="wvc:group_signatures_VLR-gpk"/>
  </message>
  <message name="receive_invoice-response">
    <part name="status" type="xsd:string"/>
    <part name="db-service-billing-consumer" element="wvc:db-service-
billing-consumer"/>
    <part name="invoice-information" element="wvc:invoice-
information"/>
  </message>
  <portType name="invoicer">
    <operation name="receiveinvoice">
      <input message="tns:receive_invoice-request"/>
      <output message="tns:receive_invoice-response"/>
    </operation>
  </portType>
```

```
  <binding name="invoicerSOAP11Binding" type="tns:invoicer">
    <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="receiveinvoice">
      <soap:operation style="document"
soapAction="urn:invoicer/receiveinvoice"/>
      <input>
        <soap:body use="literal" parts="group_signatures_VLR-gpk"/>
      </input>
      <output>
        <soap:body use="literal" parts="status"/>
        <soap:body use="literal" parts="db-service-billing-consumer"/>
        <soap:body use="literal" parts="invoice-information"/>
      </output>
    </operation>
  </binding>
  <service name="invoicer">
    <port name="invoicerPort" binding="tns:invoicerSOAP11Binding">
      <soap:address location="http://www.viniciuspacheco.com"/>
    </port>
  </service>
</definitions>
```

E – SERVIÇO SAAS EXEMPLO PAYROLL – *PAYROLL.WSDL*

Payroll

```
<?xml version="1.0" encoding="UTF-8" ?>
<definitions targetNamespace="urn:payroll"
             xmlns="http://schemas.xmlsoap.org/wsdl/"
             xmlns:tns="urn:payroll"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
             xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
             xmlns:wvc="http://www.viniciuspacheco.com">
  <types>
    <xsd:schema targetNamespace="urn:payroll/types"
                elementFormDefault="qualified"/>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/funcionarios.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import schemaLocation="../xsd/pagamentos.xsd"
                  namespace="http://www.viniciuspacheco.com"/>
    </xsd:schema>
  </types>
  <message name="request">
    <part name="funcionarios" element="wvc:funcionarios"/>
  </message>
  <message name="response">
    <part name="pagamentos" element="wvc:pagamentos"/>
  </message>
  <portType name="payroll">
    <operation name="execute">
      <input message="tns:request"/>
      <output message="tns:response"/>
    </operation>
  </portType>
  <binding name="payrollSOAP11Binding" type="tns:payroll">
    <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="execute">
      <soap:operation style="document"
soapAction="urn:payroll/execute"/>
      <input>
        <soap:body use="literal" parts="funcionarios"/>
      </input>
      <output>
        <soap:body use="literal" parts="pagamentos"/>
      </output>
    </operation>
  </binding>
  <service name="payroll">
    <port name="payrollPort" binding="tns:payrollSOAP11Binding">
      <soap:address location="http://www.viniciuspacheco.com"/>
```

```
    </port>
  </service>
</definitions>
```