CRISTINA KLIPPEL DOMINICINI

UMA ABORDAGEM CENTRADA NO USUÁRIO PARA COMPARTILHAMENTO
E GERENCIAMENTO DE DADOS ENTRE APLICAÇÕES *WEB*

São Paulo
2012

CRISTINA KLIPPEL DOMINICINI

UMA ABORDAGEM CENTRADA NO USUÁRIO PARA COMPARTILHAMENTO
E GERENCIAMENTO DE DADOS ENTRE APLICAÇÕES *WEB*

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

São Paulo
2012

CRISTINA KLIPPEL DOMINICINI

UMA ABORDAGEM CENTRADA NO USUÁRIO PARA COMPARTILHAMENTO
E GERENCIAMENTO DE DADOS ENTRE APLICAÇÕES *WEB*

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

Área de Concentração:
Sistemas Digitais

Orientadora: Prof^ª. Dra. Tereza Cristina
Melo de Brito Carvalho

São Paulo
2012

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, de agosto de 2012.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Dominicini, Cristina Klippel

Uma abordagem centrada no usuário para compartilhamento e gerenciamento de dados entre aplicações web / C.K.

Dominicini. -- ed.rev. -- São Paulo, 2012.

216 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1. Internet 2. Protocolos de comunicação 3. Segurança de redes I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II. t.

À minha família e ao meu marido.

AGRADECIMENTOS

A Deus pela proteção durante esta caminhada.

À minha orientadora, Prof^ª. Dra. Tereza Cristina Melo de Brito Carvalho, pelas oportunidades de crescimento oferecidas, orientação e conselhos que me ajudaram a trilhar o meu caminho durante a elaboração desta dissertação.

Aos meus pais, Waldir e Eugélica, por todo o amor, carinho e dedicação. Eles são a minha maior inspiração.

Ao meu marido, amigo e companheiro, Rodolfo, pelo apoio e encorajamento incondicionais ao longo dessa jornada. Esta é uma conquista conjunta.

Aos meus amigos e a minha família que me apoiaram e caminharam ao meu lado.

Ao amigo Rony Sakuragui, pelo apoio essencial e pelas idéias valiosas para a evolução desta dissertação.

Ao Prof. Dr. Marcos Simplício, pela ajuda nas discussões para concepção deste trabalho.

Ao Prof. Dr. Wilson Ruggiero, pelo incentivo durante o desenvolvimento deste trabalho no LARC.

A todos os amigos do LARC, que me ajudaram direta ou indiretamente no desenvolvimento desta dissertação.

À Ericsson Telecomunicações do Brasil e Ericsson Research da Suécia, pelo suporte financeiro e cooperação conjunta nos projetos de pesquisa do LARC.

RESUMO

Nos últimos anos, tornou-se muito mais fácil para os usuários criarem e disseminarem seus dados na *Web*. Neste contexto, uma grande quantidade de dados dos usuários é criada e utilizada de forma distribuída entre várias aplicações *web*. Essa distribuição de dados criou novas necessidades para o usuário, que agora precisa gerenciar dados que estão hospedados em diferentes *sites* e compartilhar dados entre *sites* de forma protegida. Entretanto, os mecanismos existentes na *Web* hoje não conseguem atender a essas necessidades e o usuário se vê obrigado a utilizar mecanismos que consomem muito tempo, causam replicação de dados, não são suficientemente seguros ou fornecem apenas um controle limitado sobre os seus dados. Assim, o objetivo deste trabalho é propor uma arquitetura centrada no usuário para gerenciamento de dados na *Web* e compartilhamento desses dados com aplicações *web*. A proposta dessa arquitetura é baseada em uma especificação detalhada dos requisitos do sistema usando um método específico para o ambiente *Web*. Para assegurar que o sistema proposto é seguro, é realizada uma análise para identificação das vulnerabilidades e das ameaças do sistema e geração um plano de mitigação. A viabilidade técnica de implementação da arquitetura proposta é mostrada por meio da implementação de um protótipo como prova de conceito. Ao final, mostra-se por meio desse protótipo que o sistema cumpre os objetivos e requisitos propostos.

Palavras-chave: *Web*. Internet. Gerenciamento e compartilhamento de dados. Segurança de informação. Protocolos para *Web*.

ABSTRACT

In recent years, it became much easier for users to create and disseminate their data on the Web. In this context, a large amount of user data is created and used in a distributed way across multiple web applications. This data distribution has created new needs for the user, who now needs to manage data that is hosted on different sites and to securely share data between sites. However, existing mechanisms on the Web cannot meet these needs and the user is forced to use mechanisms that are time consuming, cause data replication, are not sufficiently secure or provide limited control over their data. The objective of this thesis is to propose a user-centric architecture for data management in the Web and data sharing across web applications. The proposed architecture is based on a detailed system requirements specification, which uses an appropriate method for the Web environment. To ensure system security, a threat and vulnerability analysis is performed and a mitigation plan is generated. The technical feasibility of implementing the proposed architecture is shown through the implementation of a proof of concept prototype. Finally, this prototype helps to show that the system meets the proposed objectives and requirements.

Keywords: Web. Internet. Data management and sharing. Information security. Web protocols.

LISTA DE ILUSTRAÇÕES

Figura 1. Silos de dados na <i>Web</i> (Adaptado de (SUN, S.-T.; BEZNOSOV, KONSTANTIN, 2009)).....	21
Figura 2. Componentes básicos de um SCGD	29
Figura 3. Arquitetura do sistema descrito em (LI; LUO, 2009) (Adaptado de (LI; LUO, 2009)).....	34
Figura 4. Arquitetura <i>Mashup</i> (HSU; CHEN, 2009).....	37
Figura 5. Arquitetura UMA (Adaptado de (MACHULAK <i>et al.</i> , 2010))	38
Figura 6. Arquitetura PeCMan (Adaptado de (DAENEN <i>et al.</i> , 2010))	41
Figura 7. Arquitetura do sistema OpenIDemail (Adaptado de (SUN; HAWKEY; BEZNOSOV, 2009)).....	42
Figura 8. Serviços de compartilhamento em redes sociais (Adaptado de (KO <i>et al.</i> , 2010))	43
Figura 9. Fluxo de atividades do método NDT (<i>Navigational Development Techniques</i>) (Adaptado de (ESCALONA; TORRES; MEJÍAS, 2002))	52
Figura 10. Atores do sistema.....	56
Figura 11. Casos de Uso do sistema para o gerenciamento de conta	57
Figura 12. Casos de Uso do sistema para o compartilhamento de recursos.....	58
Figura 13. Arquitetura OpenID.....	67
Figura 14. Fluxo de operação OpenID.....	68
Figura 15. Fluxo de mensagens do protocolo OAuth (Adaptado de (HAMMER-LAHAV; RECORDON; HARDT, 2012))	72
Figura 16. Atualização de um <i>token</i> de acesso expirado (Adaptado de (HAMMER-LAHAV; RECORDON; HARDT, 2012))	75
Figura 17. Portal de gerenciamento do Sistema PDS.....	79
Figura 18. Armazenando recursos do Usuário	80
Figura 19. Recuperando recursos do Usuário	81
Figura 20. Arquitetura do Sistema PDS.....	82
Figura 21. Arquitetura do Servidor PDS	84
Figura 22. Lista de recursos em formato JSON (<i>JavaScript Object Notation</i>)	90
Figura 23. <i>Token</i> do Sistema PDS.....	92

Figura 24. Exemplo de <i>Handle</i>	92
Figura 25. Fluxo de Autenticação OpenID	95
Figura 26. Fluxo de mensagens para compartilhamento de dados	96
Figura 27. Processo de análise de ameaças	114
Figura 28. Ambiente de testes	141
Figura 29. Arquitetura MVC (<i>Model-View-Controller</i>) (Adaptado de (SILBERSCHATZ; KORTH; SUDARSHAN, 2010))	142
Figura 30. Página inicial do portal de gerenciamento	144
Figura 31. Página de <i>login</i> do portal de gerenciamento	144
Figura 32. <i>Login</i> usando Provedor de Identidade OpenID	145
Figura 33. Página de <i>login</i> bem sucedido	145
Figura 34. Interface de gerenciamento de conta do Usuário	146
Figura 35. Interface de gerenciamento de recursos	147
Figura 36. Criação de recursos	148
Figura 37. Interface de gerenciamento de autorizações	149
Figura 38. Interface de gerenciamento de histórico	150
Figura 39. Estado inicial dos recursos	151
Figura 40. Estado inicial das autorizações	152
Figura 41. Página de login do <i>site</i> do Hospital Saint Peter	152
Figura 42. <i>Login</i> usando Provedor de Identidade OpenID	153
Figura 43. <i>Login</i> bem sucedido	153
Figura 44. Requisição de acesso de leitura	154
Figura 45. Página de <i>login</i> do Servidor PDS	155
Figura 46. Requisição de autorização para acesso de leitura	156
Figura 47. Autorização do Usuário para acesso de leitura	157
Figura 48. Acesso de leitura bem sucedido	158
Figura 49. Autorização para acesso de leitura	159
Figura 50. Requisição de acesso de escrita	160
Figura 51. Requisição de autorização para acesso de escrita	161
Figura 52. Acesso de escrita bem sucedido	162
Figura 53. Autorizações concedidas pelo Usuário	163
Figura 54. Criação de recursos pelo Provedor de Serviço	164
Figura 55. Página de registro	165

Figura 56. Confirmação de domínio	166
Figura 57. Registro bem-sucedido	166
Figura 58. Mensagens HTTP capturadas na máquina do Usuário para um acesso de leitura	167
Figura 59. Mensagens HTTP capturadas no Servidor PDS para um acesso de leitura ...	169
Figura 60. Mensagens HTTP capturadas na máquina do Usuário para um acesso de escrita	171
Figura 61. Mensagens HTTP capturadas no Servidor PDS para um acesso de escrita ...	173
Figura 62. Estrutura do banco de dados do Servidor PDS	176

LISTA DE TABELAS

Tabela 1. Comparação entre os trabalhos relacionados em relação às características desejáveis de um SGCD	48
Tabela 2. Descrição dos objetivos do sistema.....	53
Tabela 3. Descrição dos requisitos de armazenamento do sistema.....	55
Tabela 4. Descrição dos atores do sistema	56
Tabela 5. Descrição dos requisitos funcionais para os Casos de Uso de gerenciamento de contas	59
Tabela 6. Descrição dos requisitos funcionais para os Casos de Uso de compartilhamento de recursos	60
Tabela 7. Descrição das frases do sistema	61
Tabela 8. Descrição dos requisitos interativos do sistema.....	62
Tabela 9. Descrição dos requisitos não funcionais do sistema	63
Tabela 10. Matriz de rastreabilidade	65
Tabela 11. Notações para descrição das entidades do sistema	98
Tabela 12. Notações para descrição dos parâmetros das mensagens do protocolo	98
Tabela 13. Mensagens do protocolo do Sistema PDS para acesso de leitura	100
Tabela 14. Mensagens do protocolo do Sistema PDS para acesso de escrita	101
Tabela 15. Ativos do Sistema PDS	117
Tabela 16. Categoria Usuário: ameaças do Sistema PDS	119
Tabela 17. Categoria Usuário: relação das ameaças com as vulnerabilidades e ativos..	120
Tabela 18. Categoria Provedor de Serviço: ameaças do Sistema PDS.....	121
Tabela 19. Categoria Provedor de Serviço: relação das ameaças com as vulnerabilidades e ativos	122
Tabela 20. Categoria Servidor PDS: ameaças do Sistema PDS	123
Tabela 21. Categoria Servidor PDS: relação das ameaças com as vulnerabilidades e ativos	125
Tabela 22. Categoria Provedor de Identidade: ameaças do Sistema PDS.....	127
Tabela 23. Categoria Provedor de Identidade: relação das ameaças com as vulnerabilidades e ativos	127
Tabela 24. Definições do Nível de Probabilidade - $P()$ (STONEBURNER; GOGUEN; FERINGA, 2002)	128

Tabela 25. Definições do Nível de Impacto – I() (STONEBURNER; GOGUEN; FERINGA, 2002)	129
Tabela 26. Matriz de Nível de Risco – R() (STONEBURNER; GOGUEN; FERINGA, 2002)	129
Tabela 27. Escala de risco e ações necessárias (STONEBURNER; GOGUEN; FERINGA, 2002)	130
Tabela 28. Análise de risco.....	131
Tabela 29. Medidas para mitigação das ameaças.....	133
Tabela 30. Plano de mitigação do sistema	137
Tabela 31. Mensagens HTTP capturadas na máquina do Usuário para um acesso de leitura	168
Tabela 32. Mensagens HTTP capturadas no Servidor PDS para um acesso de leitura...	170
Tabela 33. Mensagens HTTP capturadas na máquina do Usuário para um acesso de escrita	172
Tabela 34. Mensagens HTTP capturadas no Servidor PDS para um acesso de escrita...	174
Tabela 35. Mapeamento entre requisitos de armazenamento e entidades do banco de dados.....	177
Tabela 36. Cumprimento de requisitos funcionais.....	179
Tabela 37. Cumprimento de requisitos interativos.....	180
Tabela 38. Cumprimento de requisitos não funcionais	181
Tabela 39. Objetivo 01 - Gerenciamento de contas dos Usuários.....	198
Tabela 40. Objetivo 02 - Autenticação de Usuários usando um Provedor de Identidade	198
Tabela 41. Objetivo 03 - Compartilhamento de recursos do Usuário com aplicações <i>web</i>	198
Tabela 42. Objetivo 04 - Proteção dos recursos do Usuário	198
Tabela 43. Objetivo 05 - Implantação do sistema em larga escala na Internet	199
Tabela 44. Requisito de Armazenamento 01 - Conta do Usuário.....	199
Tabela 45. Requisito de Armazenamento 02 - Recursos do Usuário.....	199
Tabela 46. Requisito de Armazenamento 03 - Autorizações.....	200
Tabela 47. Requisito de Armazenamento 04 - Histórico.....	200
Tabela 48. Requisito de Armazenamento 05 - Sites registrados.....	201
Tabela 49. Ator 01 - Usuário.....	201

Tabela 50. Ator 02 - Provedor de Serviço	201
Tabela 51. Ator 03 - Provedor de Identidade	202
Tabela 52. Requisito Funcional 01 - Autenticar Usuário.....	202
Tabela 53. Requisito Funcional 02 - Consultar recursos.....	203
Tabela 54. Requisito Funcional 03 - Criar recursos	203
Tabela 55. Requisito Funcional 04 - Remover recursos	204
Tabela 56. Requisito Funcional 05 - Consultar histórico	205
Tabela 57. Requisito Funcional 06 - Remover atividades.....	205
Tabela 58. Requisito Funcional 07 - Consultar autorizações.....	206
Tabela 59. Requisito Funcional 08 - Remover autorizações.....	207
Tabela 60. Requisito Funcional 09 - Requisitar autorização	208
Tabela 61. Requisito Funcional 10 - Ler recurso.....	209
Tabela 62. Requisito Funcional 11 - Escrever recurso.....	209
Tabela 63. Requisito Funcional 12 - Autenticar Provedor de Serviço.....	210
Tabela 64. Requisito Funcional 13 - Registrar Provedor de Serviço	211
Tabela 65. Frase 01 - Recuperação de recursos por tipo	211
Tabela 66. Frase 02 - Recuperação de atividade por período	212
Tabela 67. Frase 03 - Recuperação de autorização por Provedor de Serviço	212
Tabela 68. Frase 04 - Recuperação de Provedor de Serviço por URL	212
Tabela 69. Frase 05 - Recuperação de recursos por <i>tag</i>	212
Tabela 70. Requisito Interativo 01 - Informação sobre os recursos do Usuário.....	213
Tabela 71. Requisito Interativo 02 - Informação sobre as atividades.....	213
Tabela 72. Requisito Interativo 03 - Informação sobre as autorizações do Usuário.....	214
Tabela 73. Requisito Interativo 04 - Informação sobre uma requisição de Autorização	214
Tabela 74. Requisito Não Funcional 01 - Confidencialidade e integridade.....	215
Tabela 75. Requisito Não Funcional 02 - Compartilhamento de tipos de dados genéricos	215
Tabela 76. Requisito Não Funcional 03 - Autorização explícita do Usuário	215
Tabela 77. Requisito Não Funcional 04 - Uso de navegador <i>web</i> como agente do Usuário	215
Tabela 78. Requisito Não Funcional 05 - Uso de padrões abertos e amplamente adotados	216

Tabela 79. Requisito Não Funcional 06 - Disponibilidade	216
Tabela 80. Requisito Não Funcional 07 - Complexidade baixa para as aplicações <i>web</i> .	216

LISTA DE ABREVIATURAS E SIGLAS

ABAC	<i>Attribute Based Access Control</i>
API	<i>Application Programming Interface</i>
ARP	<i>Address Resolution Protocol</i>
CBAC	<i>Credential Based Access Control</i>
CSRF	<i>Cross-Site Request Forgery</i>
CWE	<i>Common Weakness Enumeration</i>
DNS	<i>Domain Name System</i>
GBA	<i>Generic Bootstrapping Architecture</i>
HTTP	<i>HyperText Transfer Protocol</i>
HTTPS	<i>HyperText Transfer Protocol Secure</i>
IP	<i>Internet Protocol</i>
JSON	<i>JavaScript Object Notation</i>
MVC	<i>Model View Control</i>
NDT	<i>Navigational Development Techniques</i>
NIST	<i>National Institute of Standards and Technology</i>
OP	<i>OpenID Provider</i>
PDS	<i>Personal Data Sharing</i>
PeCMan	<i>Personal Content Manager</i>
RBAC	<i>Role Based Access Control</i>
RDF	<i>Resource Description Framework</i>

RP	<i>Relying Party</i>
RPC	<i>Remote Procedure Call</i>
SAML	<i>Security Assertion Markup Language</i>
SGCD	<i>Sistema de Gerenciamento e Compartilhamento de Dados</i>
SGI	<i>Sistema de Gerenciamento de Identidade</i>
SSL	<i>Secure Sockets Layer</i>
SSO	<i>Single Sign-On</i>
UA	<i>User Agent</i>
UMA	<i>User-Managed Access</i>
URI	<i>Uniform Resource Identifier</i>
URL	<i>Uniform Resource Locator</i>
W3C	<i>World Wide Web Consortium</i>
XACML	<i>eXtensible Access Control Markup Language</i>
XML	<i>eXtensible Markup Language</i>
XRDS	<i>eXtensible Resource Descriptor Sequence</i>
XRI	<i>eXtensible Resource Identifier</i>

SUMÁRIO

1	INTRODUÇÃO	20
1.1	Descrição do Problema	20
1.2	Objetivo	23
1.3	Método.....	24
1.4	Contexto do Projeto de Pesquisa.....	25
1.5	Organização do Trabalho	26
2	CONCEITOS E TRABALHOS RELACIONADOS.....	27
2.1	Definições e Arquitetura.....	27
2.2	Trabalhos Relacionados.....	30
2.2.1	Gerenciamento de Identidade	30
2.2.2	Autorização e Controle de Acesso	31
2.2.3	Compartilhamento e Gerenciamento de Dados na Web.....	34
2.3	Características Desejáveis.....	44
2.4	Comparação entre os trabalhos relacionados.....	47
2.5	Resumo do Capítulo.....	48
3	ANÁLISE DE REQUISITOS.....	50
3.1	Escopo.....	50
3.1.1	Requisitos Gerais	50
3.2	Conceitos e Método.....	51
3.2.1	Método NDT	52
3.3	Análise de Requisitos usando o Método NDT.....	53
3.3.1	Atividade 1: Definição dos Objetivos	53
3.3.2	Atividade 2: Definição dos Requisitos de Armazenamento	54
3.3.3	Atividade 3: Definição dos Atores	56

3.3.4	Atividade 4: Definição dos Requisitos Funcionais	56
3.3.5	Atividade 5: Definição dos Requisitos Interativos.....	60
3.3.6	Atividade 6: Definição dos Requisitos Não Funcionais.....	62
3.3.7	Matriz de Rastreabilidade	64
3.4	Resumo do Capítulo.....	64
4	TECNOLOGIAS HABILITADORAS.....	66
4.1	OpenID.....	66
4.1.1	Arquitetura.....	67
4.1.2	Protocolo.....	67
4.2	OAuth	69
4.2.1	Arquitetura.....	70
4.2.2	Escopo de Utilização do Protocolo OAuth.....	70
4.2.3	Conceitos.....	71
4.2.4	Protocolo.....	72
4.2.5	Token de Atualização	74
4.3	Considerações de Segurança.....	75
4.4	Resumo do Capítulo.....	76
5	PROPOSTA DA ARQUITETURA DO SISTEMA PDS.....	77
5.1	Requisitos.....	77
5.2	Visão Geral do Sistema PDS	78
5.2.1	Interface de Gerenciamento	79
5.2.2	Interface de Compartilhamento.....	79
5.3	Arquitetura do Sistema PDS.....	81
5.4	Arquitetura do Servidor PDS	83
5.4.1	Módulos de Extensão	86
5.5	Estruturas de dados	88
5.5.1	Representação de Tipos de Recursos	88

5.5.2	Representação do Escopo da Requisição de Acesso	89
5.5.3	Lista de Recursos	89
5.5.4	Token.....	90
5.6	Protocolo de Comunicação do Sistema PDS.....	93
5.6.1	Condições para Execução do Protocolo	93
5.6.2	Autenticação OpenID e Single Sign-On.....	93
5.6.3	Fluxo Simplificado do Protocolo de Compartilhamento	95
5.6.4	Mensagens do Protocolo de Compartilhamento	97
5.7	Considerações sobre o Sistema PDS	101
5.7.1	Análise da Arquitetura.....	102
5.7.2	Adequação do Protocolo OAuth para o Sistema PDS.....	104
5.8	Resumo do Capítulo.....	106
6	EXEMPLOS DE CENÁRIOS DE USO DO SISTEMA PDS.....	107
6.1	Redes Sociais	107
6.1.1	Exemplo de Uso	108
6.2	Saúde Móvel.....	109
6.2.1	Exemplo de Uso	110
6.3	Resumo do Capítulo.....	111
7	ANÁLISE DE VULNERABILIDADES E AMEAÇAS DO SISTEMA PDS.....	112
7.1	Conceitos	112
7.2	Trabalhos Relacionados sobre Análise de Ameaças	113
7.3	Método e Classificação	114
7.4	Caracterização do Atacante	115
7.5	Modelagem de Ameaças.....	115
7.5.1	Caracterização do sistema.....	116
7.5.2	Identificação de ativos	116
7.5.3	Determinação de Ameaças e Vulnerabilidades.....	118

7.6	Análise de Riscos	128
7.7	Plano de Mitigação	133
7.8	Resumo do Capítulo.....	138
8	PROTÓTIPO DO SISTEMA PDS.....	139
8.1	Estrutura Geral do Protótipo	139
8.1.1	Componentes.....	139
8.1.2	Exemplo de Testes.....	140
8.1.3	Implementação do Servidor PDS.....	141
8.1.4	Bibliotecas	143
8.2	Interface com o Usuário	143
8.2.1	Interface do Portal de Gerenciamento.....	143
8.2.2	Interface de Compartilhamento.....	151
8.3	Registro do Provedor de Serviço.....	165
8.4	Fluxo de mensagens	167
8.4.1	Acesso de Leitura.....	167
8.4.2	Acesso de Escrita	171
8.5	Considerações sobre Segurança	175
8.6	Considerações sobre o Cumprimento dos Requisitos.....	175
8.6.1	Requisitos de Armazenamento	175
8.6.2	Requisitos Funcionais e Interativos	178
8.6.3	Requisitos Não Funcionais.....	180
8.7	Resumo do Capítulo.....	181
9	CONSIDERAÇÕES FINAIS	183
9.1	Contribuições	183
9.2	Considerações sobre o Sistema PDS	185
9.3	Trabalhos Futuros.....	186
9.4	Publicações	189

REFERÊNCIAS	190
APÊNDICE A – MODELOS COMPLETOS DO MÉTODO NTD PARA A ESPECIFICAÇÃO DE REQUISITOS DO SISTEMA.....	198

1 INTRODUÇÃO

O advento da *Web 2.0* representou uma grande mudança na forma como as aplicações *web* são criadas e utilizadas, pois o conteúdo *web* deixou de ser estritamente publicado pelos provedores de serviço e passou a ser interativo e gerado pelos próprios usuários (O REILLY, 2007) (MURUGESAN, 2007) (NICKULL; HINCHCLIFFE, 2009). Um exemplo dessa mudança é o crescimento de serviços que encorajam os usuários a interagirem e produzirem conteúdos, tais como *wikis*, *blogs* e redes sociais. Como resultado, tornou-se muito mais fácil para os usuários criarem seu próprio conteúdo e disseminarem seus dados na Internet. De fato, os *websites* que fornecem serviços relacionados à manipulação ou visualização de dados produzidos pelos usuários são muito populares, representando pelo menos metade dos dez *websites* mais visitados (ALEXA, 2012).

Neste contexto, uma grande quantidade de dados dos usuários, potencialmente valiosos e confidenciais, é criada e utilizada de forma distribuída entre várias aplicações *web* (MACHULAK *et al.*, 2010). Essa distribuição de dados criou novas necessidades para o usuário, tais como: pesquisar e editar dados que estão hospedados em diferentes *sites*; criar coleções de dados heterogêneos provenientes de diferentes *sites*; e compartilhar dados entre *sites* de forma protegida (GEAMBASU *et al.*, 2008). Entretanto, os mecanismos existentes na *Web* hoje não conseguem atender a todas essas necessidades (LI; LUO, 2009), impedindo que o usuário se beneficie de forma completa das funcionalidades oferecidas pela *Web 2.0* ao mesmo tempo em que mantém o controle de seus dados.

1.1 Descrição do Problema

Atualmente, grande parte dos serviços na *Web* segue um modelo conhecido como “centrado em *websites*”. Neste modelo, cada serviço *web* controla seu próprio conjunto de usuários e seu próprio contêiner ou silo de dados de forma isolada. A Figura 1 ilustra esse modelo que também é conhecido como “jardins murados” (MOSTARDA *et al.*,

2009), pois as aplicações *web* não oferecem nenhuma alternativa para o compartilhamento de dados com outras aplicações.

Figura 1. Silos de dados na Web (Adaptado de (SUN, S.-T.; BEZNOSOV, KONSTANTIN, 2009))

Um dos problemas dessa abordagem é que os dados dos usuários estão distribuídos por várias aplicações, nas quais são armazenados de forma independente e ficam confinados em silos de dados (GEAMBASU *et al.*, 2008). Como consequência, os usuários têm que manter cópias separadas e potencialmente inconsistentes de seus dados, identidades e políticas de controle de acesso para cada serviço que utilizam na Web (SUN; HAWKEY; BEZNOSOV, 2009) (TOOTOONCHIAN *et al.*, 2008). Assim, os usuários não têm uma visão centralizada dos seus dados e não conseguem executar operações de gerenciamento sobre eles, tais como pesquisa e edição de conteúdo, de forma integrada.

Outro problema está relacionado ao compartilhamento de dados do usuário entre aplicações *web*. Na maioria dos casos, quando os usuários querem transportar seus dados entre aplicações, eles não encontram um mecanismo de compartilhamento adequado e precisam fazer a transferência manual por meio do *download* de seus dados de uma aplicação seguido pelo *upload* desses dados em outra aplicação. Para tentar resolver esse problema, um conjunto restrito de serviços fornece mecanismos para compartilhamento na forma de APIs (*Application Programming Interfaces*) que expõem a outras aplicações *web* um conjunto de funções de acesso aos dados do usuário. Contudo,

esses mecanismos são, em geral, limitados a um pequeno conjunto de dados, concentrados em nichos de aplicação (por exemplo, redes sociais) e incompatíveis entre si, não permitindo que eles sejam facilmente reutilizados para compartilhar recursos entre diferentes domínios de aplicação (GEAMBASU *et al.*, 2008). Além disso, tais mecanismos nem sempre dispõem de níveis adequados de segurança, privacidade, flexibilidade e usabilidade.

A seguir são descritos alguns cenários que exemplificam os problemas discutidos.

- **Cenário 1:** Alice visita um *website* de edição de fotos para fazer uma montagem de fotos para o aniversário de casamento de seus pais. Suas fotos de família estão espalhadas em diferentes *sites* como Flickr, Google Docs e Facebook. O problema é que não existe uma interface centralizada que permita que Alice visualize todas as suas fotos e recupere uma seleção dessas fotos. Para conseguir fazer isso, Alice precisa fazer *download* e *upload* de cada uma das fotos no *website* de edição para, então, poder manipular as suas fotos e fazer a montagem.
- **Cenário 2:** João possui uma conta em um *website* de rede social, onde mantém uma extensa lista de seus contatos, que inclui amigos, familiares e colegas de trabalho. João, então, decide criar uma nova conta em um *website* de *microblog*, mas percebe que não existe um mecanismo para reutilizar a sua lista de contatos existente no seu *website* de rede social. Para que possa utilizar esse novo serviço, João terá que manualmente entrar todos os seus contatos no *website* de *microblog*. Além de inconveniente, João terá que manter atualizadas duas cópias separadas da mesma lista de contatos.
- **Cenário 3:** Carlos é desenvolvedor e criou uma aplicação que descobre as preferências de um usuário a partir dos seus arquivos pessoais e encontra pessoas com gostos similares. Entretanto, enfrenta sérios problemas de adesão, pois os usuários não estão dispostos a fazer *upload* de todos os seus arquivos em um novo serviço. Para tentar resolver esse problema, criou mecanismos que permitem que o usuário busque seus arquivos em serviços conhecidos como Gmail, Twitter e Facebook. Contudo, esses serviços oferecem mecanismos limitados para acesso aos arquivos dos usuários e as APIs são incompatíveis entre si, exigindo que Carlos crie componentes de *software* diferentes para integração com cada um dos serviços.

Em resumo, quando o usuário precisa gerenciar ou compartilhar seus dados entre aplicações *web*, ele se vê obrigado a utilizar mecanismos que consomem muito tempo, causam replicação de dados, são imprecisos ou fornecem apenas um controle limitado de suas preferências de compartilhamento. Logo, é possível afirmar que existe a necessidade de um sistema ou mecanismo na *Web* que possa lidar com as questões apontadas aqui sobre gerenciamento e compartilhamento de dados centrado no usuário entre aplicações *web*.

Dentro desse escopo, este trabalho concentra-se em investigar e responder às seguintes perguntas:

- Como permitir que o usuário gerencie seus dados na *Web* de forma centralizada?
- Como permitir que o usuário compartilhe seus dados com aplicações *web* de forma fácil e segura?
- Como equilibrar a proteção dos dados do usuário com a necessidade de permitir que uma aplicação *web* manipule os dados criados por outra aplicação?

1.2 Objetivo

Este trabalho tem como objetivo definir uma arquitetura inovadora **centrada no usuário** para **gerenciamento de recursos protegidos** do usuário na *Web* e **compartilhamento** desses recursos com **aplicações *web***.

As palavras-chave da definição do objetivo são explicadas a seguir:

- **Centrado no usuário:** o usuário deve ser capaz de determinar qual dado é compartilhado, com quem ele é compartilhado e o escopo do compartilhamento. Nenhum dado pode ser compartilhado sem o consentimento do usuário.
- **Recursos protegidos:** correspondem aos dados dos usuários que são protegidos contra acesso não autorizado pelo sistema.
- **Gerenciamento de recursos protegidos na Web:** o sistema deve permitir que o usuário organize e visualize seus recursos na *Web* por meio de um serviço centralizado.

- **Compartilhamento de recursos com aplicações *web*:** o sistema deve permitir que o usuário compartilhe seus recursos com aplicações *web* por meio de um gerenciador centralizado de autorização, que toma as decisões de compartilhamento baseadas nas preferências do usuário.

Para cumprir esse objetivo, este trabalho define e propõe o protocolo e a arquitetura do **Sistema PDS** (*Personal Data Sharing*). Esse sistema é proposto considerando-se as características do ambiente *Web* e inclui o usuário como parte central do seu modelo, de forma que o usuário possa gerenciar e compartilhar seus recursos na *Web*.

1.3 Método

O método utilizado neste trabalho é de pesquisa aplicada, abrangendo revisão bibliográfica da área, estudo dos diversos trabalhos correlatos, especificação e projeto de uma arquitetura inovadora para gerenciamento e compartilhamento de dados do usuário com aplicações *web* e implementação de um protótipo da arquitetura especificada como prova de conceito.

Inicialmente, foi realizada uma revisão bibliográfica da área, visando identificar os conceitos básicos relacionados a segurança de informação, sistemas de gerenciamento de identidade e autorização. Depois, estudaram-se os trabalhos relacionados com o objetivo de fazer um levantamento do estado da arte na área de gerenciamento e compartilhamento de dados do usuário com aplicações *web*. Nessa fase, foram comparados e avaliados os trabalhos na literatura de forma a identificar as características desejáveis para um sistema desse tipo e entender os pontos positivos e limitações dos sistemas existentes.

A partir do levantamento dos trabalhos correlatos, foi desenvolvida uma especificação detalhada dos requisitos do sistema usando o método NDT (*Navigational Development Techniques*)¹. Com base nesta especificação de requisitos foi proposta uma arquitetura para um sistema de gerenciamento e compartilhamento de dados do usuário com

¹ O método NDT (*Navigational Development Techniques*) é uma técnica para especificar e analisar os requisitos de Aplicações *Web* (ESCALONA; ARAGON, 2008) (ESCALONA; TORRES; MEJÍAS, 2002).

aplicações *web*. Depois, foi realizada uma análise de ameaças do sistema proposto com o objetivo de detectar potenciais vulnerabilidades em relação aos serviços de segurança do sistema. Posteriormente, utilizaram-se os conhecimentos obtidos nas etapas anteriores para implementar um protótipo como prova de conceito da arquitetura proposta. Ao final, este protótipo foi avaliado para determinar se os objetivos e os requisitos propostos foram cumpridos.

1.4 Contexto do Projeto de Pesquisa

O tema de compartilhamento de dados do usuário na *Web* começou a ser investigado e desenvolvido em um projeto em conjunto com a Ericsson Research, o Ericsson Labs² e o Laboratório de Arquitetura e Redes de Computadores (LARC) do Departamento de Engenharia de Computação e Sistemas Digitais (PCS) da Escola Politécnica desta universidade. Nesse projeto, foi desenvolvido um sistema de gerenciamento de identidade baseado no protocolo OpenID (OPENID FOUNDATION, 2007), que permite que um usuário seja autenticado em diferentes *sites* na Internet por meio do cartão SIM em seu telefone celular.

A arquitetura proposta nesse projeto era capaz de comprovar a identidade do usuário para aplicações na Internet e compartilhar dados pessoais de registro do usuário (dados simples em formato de texto, tais como nome, e-mail, endereço e data de nascimento). Contudo, percebeu-se a necessidade de compartilhar dados do usuário de maior complexidade, que dependiam da lógica de negócios de cada provedor de serviço que usava o sistema de gerenciamento de identidade. A partir da identificação dessa demanda, foi realizado um estudo dos trabalhos relacionados nesta área e verificou-se que nenhuma das propostas encontradas na literatura atendia completamente às necessidades e limitações existentes no ambiente típico da interação entre usuários e *sites* para compartilhamento de dados na *Web* (como será visto no Capítulo 2). Posteriormente, uma arquitetura para compartilhamento de dados do usuário com aplicações *web* foi especificada e projetada, conforme será mostrado nos Capítulos 3 e 5.

² O Ericsson Labs é um setor da Ericsson Research que suporta pesquisa com tecnologias e soluções inovadoras. O seu portal na Internet encontra-se em <https://labs.ericsson.com/>

Após o término desse projeto, iniciou-se outro projeto, também dentro do contexto de uma parceria entre a Ericsson Research e o LARC, na área de Saúde Móvel (uso de dispositivos móveis para aplicações na área de saúde). Nesse projeto, um dos problemas de pesquisa a serem resolvidos foi o compartilhamento de dados médicos do paciente entre instituições médicas no ambiente *Web*. Assim, a arquitetura que já havia sido especificada e projetada para atender a um cenário genérico de compartilhamento de dados foi implementada para aplicação no cenário de Saúde Móvel, conforme será mostrado no Capítulo 6.

1.5 Organização do Trabalho

Este documento está organizado da seguinte forma. O Capítulo 2 apresenta uma visão geral dos sistemas de gerenciamento e compartilhamento de dados na *Web* e descreve os trabalhos relacionados. No Capítulo 3, é realizada uma ampla análise dos requisitos que um sistema desse tipo deve atender e apresentada a especificação de requisitos usada na concepção de sua arquitetura. Por sua vez, o Capítulo 4 descreve as principais tecnologias que formam a base da arquitetura PDS (*Personal Data Sharing*), o sistema proposto neste trabalho. Com isso, o Capítulo 5 descreve a arquitetura e o protocolo do Sistema PDS. O Capítulo 6 mostra, então, dois cenários de uso que exemplificam como aplicações com diferentes lógicas de negócios podem se beneficiar do Sistema PDS. Em seguida, o Capítulo 7 realiza uma análise de ameaças do Sistema PDS com o objetivo de detectar potenciais vulnerabilidades e propor um plano com medidas de mitigação. Depois, o Capítulo 8 descreve a implementação de um protótipo do Sistema PDS como prova de conceito da arquitetura proposta e verifica se os requisitos anteriormente definidos são atendidos. Por fim, o Capítulo 9 apresenta a avaliação geral do sistema desenvolvido, as contribuições deste trabalho e as sugestões de trabalhos futuros.

2 CONCEITOS E TRABALHOS RELACIONADOS

Inicialmente, este capítulo apresenta os conceitos relacionados a Sistemas de Gerenciamento e Compartilhamento de Dados na *Web*. Em seguida, são apresentados os resultados da investigação realizada sobre trabalhos relacionados nas áreas de gerenciamento de identidade, autorização, gerenciamento de dados centrado no usuário e compartilhamento de dados entre aplicações *web*. Com base nessa investigação, são identificadas as características desejáveis para um Sistema de Gerenciamento e Compartilhamento de Dados na *Web* que servirá de suporte à especificação dos requisitos gerais para esses sistemas, tema do próximo capítulo.

2.1 Definições e Arquitetura

Por ser um campo de pesquisa recente (conforme será mostrado na seção 2.2.3 de trabalhos relacionados), não existe uma nomenclatura padronizada para referenciar sistemas que permitem o compartilhamento e gerenciamento de recursos do usuário entre aplicações *web*. Por isso, no decorrer deste trabalho, será utilizado o termo **Sistema de Gerenciamento e Compartilhamento de Dados (SGCD)** para designar tais sistemas.

A seguir são listadas as principais definições utilizadas no contexto deste trabalho:

- **Recurso ou recurso protegido:** dados contidos em um sistema (por exemplo, na forma de arquivos) que são de propriedade de um usuário.
- **Acesso:** capacidade e meios de comunicação e/ou interação com um sistema, para fins de utilização de seus recursos visando a manipulação de informações e/ou aquisição de conhecimento das informações que o sistema contém (SHIREY, 2000).
- **Controle de acesso:** proteção dos recursos do sistema contra acesso não autorizado; processo através do qual a utilização dos recursos do sistema é

regulada de acordo com uma política de segurança e só é permitida para entidades autorizadas pelo sistema de acordo com essa política (SHIREY, 2000).

- **Autorização:** direito ou permissão concedida a uma entidade do sistema (e.g., um processo automatizado, um subsistema, uma pessoa ou um grupo de pessoas) para acessar um recurso do sistema (SHIREY, 2000).
- **Token:** estrutura de dados que contém informações sobre uma autorização de acesso a um ou mais recursos protegidos do sistema e que concede a uma entidade que tem posse do *token* a capacidade de executar operações sobre esses recursos.

Embora não exista uma definição padrão para SGCDs, é possível identificar quatro componentes básicos que fazem parte da arquitetura da maioria dos trabalhos correlatos: **Usuário**, **Cliente**, **Servidor de Autorização** e **Servidor de Recursos**. De acordo com as necessidades de cada arquitetura, o **Servidor de Autorização** e o **Servidor de Recursos** podem ser a mesma entidade ou entidades separadas.

Além desses quatro componentes básicos, um SGCD deve possuir mecanismos para autenticar o **Usuário** que podem ser implementados individualmente em cada um dos componentes (**Cliente**, **Servidor de Autorização** e **Servidor de Recursos**) ou podem ser implementados por um componente adicional, chamado **Servidor de Autenticação**.

Os componentes de um SGCD são descritos a seguir e mostrados na Figura 2.

- **Usuário:** é o proprietário dos recursos protegidos que habilita ou não o acesso a esses recursos. **Gerencia** seus recursos que estão armazenados **no Servidor de Recursos**, **controla** as políticas de controle de acesso contidas no **Servidor de Autorização** e **delega**, de forma indireta por meio do **Servidor de Autorização**, o acesso aos seus recursos para o **Cliente**.
- **Cliente ou Provedor de Serviço:** é uma aplicação que faz requisições de **acesso** a recursos protegidos em nome do proprietário desses recursos e com a sua autorização.
- **Servidor de Autorização:** é o servidor que **protege** os recursos contidos no **Servidor de Recursos** e emite *tokens* de acesso que **autorizam** o **Cliente** a

acessar esses recursos, após obter autorização do **Usuário** para compartilhamento.

- **Servidor de Recursos:** é o servidor que **hospeda** os recursos protegidos do **Usuário**, sendo capaz de responder às requisições do **Cliente** para o acesso a recursos protegidos quando o mesmo apresenta *tokens* de acesso válidos emitidos pelo **Servidor de Autorização**.
- **Servidor de Autenticação** (*opcional*): é um serviço que **autentica** o **Usuário** e se comunica com os outros componentes para entregar uma comprovação de autenticação. Com o uso de tal serviço, os componentes não precisam implementar seus próprios mecanismos de autenticação e o **Usuário** só precisa se autenticar em uma única entidade.

Figura 2. Componentes básicos de um SCGD ³

³ A Figura 2 é uma adaptação de uma figura originalmente criada por (MACHULAK *et al.*, 2010), em que é proposta uma arquitetura para gerenciamento e compartilhamento de dados na *Web*. Para que a figura pudesse representar um SGCD genérico que não se retrinxisse a uma solução específica, foram realizadas algumas modificações na figura original.

2.2 *Trabalhos Relacionados*

O compartilhamento e o gerenciamento de recursos centrados no **Usuário** requerem **mecanismos de autenticação**, que permitam que o **Usuário** seja capaz de controlar suas próprias identidades, e **de autorização**, que permitam que o **Usuário** controle as políticas que definirão as decisões de acesso aos seus recursos (SUN; BEZNOSOV, 2009). Assim, esta seção apresenta, inicialmente, alguns dos principais trabalhos relacionados nas áreas de Gerenciamento de Identidades e Autorização. Posteriormente, é realizada uma revisão bibliográfica dos trabalhos que realizam gerenciamento e compartilhamento de recursos do **Usuário** na *Web*, de modo a compreender as soluções existentes e as limitações que os **Usuários** encontram ao tentar gerenciar e compartilhar seus recursos. Esta seção dará suporte à especificação de requisitos e à definição de uma arquitetura para um SGCD que serão discutidas nos próximos capítulos.

2.2.1 *Gerenciamento de Identidade*

Um dos desafios para compartilhar recursos do **Usuário** entre diferentes Provedores de Serviço é manter identificadores de **Usuário** diferentes para cada provedor. Em geral, os **Usuários** acabam mantendo múltiplos identificadores e senhas em vários locais, o que leva a senhas fracas e reutilização da mesma senha em diferentes serviços (FLORENCIO; HERLEY, 2007). Os Sistemas de Gerenciamento de Identidade (SGIs) podem solucionar este problema, pois permitem a reutilização de uma identidade de **Usuário** em múltiplos provedores (3G AMERICAS, 2009)(BERTINO; TAKAHASHI, 2010). Um SGI autentica seus **Usuários** e, em seguida, fornece asserções de identidade que podem ser verificados pelos Provedores de Serviços, atuando como uma entidade confiável para a autenticação dos **Usuários**.

Dentre as diversas tecnologias de SGIs (EL MALIKI; SEIGNEUR, 2007), destacam-se: ID/WSF Liberty Alliance (LIBERTY ALLIANCE, 2002), Shibboleth (MORGAN *et al.*, 2004) baseado em SAML (CANTOR *et al.*, 2005), Microsoft InfoCard/CardSpace (BERTOCCI; SERACK; BAKER, 2007) e OpenID (OPENID FOUNDATION, 2007). Com exceção do

OpenID, nenhuma dessas tecnologias tem sido amplamente adotada no ambiente *Web* da Internet (SAKURAGUI, 2011). Os motivos apontados para esta baixa adoção são: complexidade da tecnologia, padrões fechados, problemas de usabilidade, centralização da identidade por SGIs não confiáveis por parte dos **Usuários** e exigência de relações de confiança pré-estabelecidas entre Provedores de Serviço e a organização que mantém o SGI (DHAMIJA; DUSSEAULT, 2008)(BERTINO; TAKAHASHI, 2010).

Em especial, o OpenID é um protocolo aberto, simples e descentralizado, que visa fornecer uma identidade digital única ao **Usuário** que possa ser reutilizada para efetuar *login* em diferentes *websites* empregando-se apenas um navegador *web*. Nos últimos anos, o OpenID surgiu como uma solução viável para a infra-estrutura de identidade centrada no **Usuário** em escala de Internet (RECORDON; REED, 2006) e vem sendo amplamente adotado. De fato, em dezembro de 2009 foram estimadas mais de um bilhão de contas OpenID habilitadas na Internet e cerca de nove milhões de *sites* com suporte a essa tecnologia (KISSEL, 2009).

Por sua ampla adoção, simplicidade e interoperabilidade com os sistemas atuais, o OpenID será a solução para gerenciamento de identidade utilizada neste trabalho. O protocolo OpenID será discutido com mais detalhes no Capítulo 4.

2.2.2 Autorização e Controle de Acesso

Soluções tradicionais de controle de acesso tomam decisões de autorização baseadas no relacionamento do requisitante com a organização ou o proprietário do recurso, como o esquema de Controle de Acesso Baseado em Papéis (*Role Based Access Control* - RBAC) (SANDHU *et al.*, 1996). Neste esquema, permissões são atribuídas a papéis, enquanto os **Usuários** são associados a esses papéis e ganham as permissões referentes aos papéis que possuem. Uma limitação desta abordagem está relacionada à delegação dinâmica de acesso, pois um papel não pode ser atribuído a uma entidade se ela não é conhecida previamente pelo sistema (GOMI, 2011). Essa limitação impede o uso de esquemas semelhantes para um ambiente descentralizado como a *Web* em que as entidades não possuem uma relação previamente estabelecida e conhecida por todos os componentes do sistema.

Esquemas de autorização mais avançados usam atributos (informações sobre o requisitante, o ambiente ou os próprios recursos) ao invés da identidade dos sujeitos para determinar as políticas de acesso a recursos, resultando no modelo de Controle de Acesso Baseado em Atributo (*Attribute Based Access Control* - ABAC). O XACML⁴ (*eXtensible Access Control Markup Language*) (OASIS, 2005), um padrão bem conhecido para expressar e trocar políticas de controle de acesso de uso geral, é um exemplo de um modelo de ABAC. O XACML pode ser combinado com o padrão SAML⁵ (*Security Assertion Markup Language*) para criação de um sistema de autorização completo em que o XACML fornece as funcionalidades de definição e avaliação de políticas e o SAML fornece as funcionalidades de transporte, proteção e gerenciamento de asserções de autorização. O ponto negativo do uso de XACML e SAML é a grande complexidade para definir e manter as políticas de autorização (3G AMERICAS, 2009). Em um sistema em que as políticas são definidas por administradores de sistema, essa limitação pode ser contornada pelo uso de interfaces de administração específicas, mas em um sistema centrado no **Usuário**, em que o **Usuário** deve definir suas políticas, essa limitação torna-se particularmente importante.

Considerando mecanismos de autorização para sistemas distribuídos, onde o controle de acesso pode ser descentralizado, uma abordagem importante é o Controle de Acesso baseados em Capacidades (*Capability Based Access Control*) (LEVY, 1984). Este esquema associa um identificador não forjável, denominado capacidade, com um conjunto de direitos de acesso a um ou mais recursos. O detentor de uma capacidade só precisa transmitir o identificador para um verificador para obter acesso aos recursos correspondentes. Segundo (KOSHUTANSKI, 2009), este esquema também é citado na literatura como Controle de Acesso Baseado em Credenciais (*Credential Based Access Control* - CBAC) e é o modelo mais adequado para aplicação de autorizações distribuídas. Em (GONG, 1989), (BULL; GONG; SOLLINS, 1992) e (HAYTON, 1996) podem ser encontrados exemplos desses sistemas. Com base nos sistemas baseados em credenciais ou capacidades, surgiram os sistemas de Gerenciamento de Confiança (WEEKS, 2001) que, além da preocupação básica com o gerenciamento de credenciais, abordam também

⁴ O XACML (*eXtensible Access Control Markup Language*) é um padrão baseado em XML, que define uma linguagem para expressar políticas de controle de acesso e um modelo de processamento para avaliar solicitações de autorização de acordo as políticas definidas.(OASIS, 2005).

⁵ O SAML (*Security Assertion Markup Language*) é um padrão aberto baseado em XML para troca de dados de autenticação e autorização entre um provedor de identidade e um provedor de serviço (CANTOR *et al.*, 2005).

as políticas de segurança e as relações de confiança entre as entidades (YAO, 2003). Exemplos desses sistemas podem ser encontrados em (ELLISON, C. *et al.*, 1999), (BLAZE *et al.*, 1999) e (LI; MITCHELL; WINSBOROUGH, 2002).

Os sistemas citados fornecem mecanismos flexíveis para determinar políticas de acesso e derivar as decisões de autorização. Entretanto, em um sistema de compartilhamento de recursos centrado no **Usuário** para *Web*, as políticas devem ser criadas pelos próprios **Usuários**, que normalmente não possuem habilidades técnicas especiais, e devem ser usadas em diferentes domínios de aplicação. Dessa forma, existem questões importantes para se resolver em relação à usabilidade, flexibilidade e interoperabilidade das soluções de autorização (SUN; BEZNOSOV, 2009). Sistemas de controle de acesso e autorização, que se baseiam nos esquemas citados, vêm sendo extensivamente pesquisados para tentar solucionar essas questões, tais como (BALFANZ, 2003), (BAUER; SCHNEIDER; FELTEN, 2002), (BHATTI; BERTINO; GHAFOR, 2004), (FENG *et al.*, 2004) e (EMIG *et al.*, 2007). Entretanto, esses sistemas não são adequados para o ambiente da *Web* 2.0 em que os **Usuários** precisam de mais poder sobre as decisões de acesso aos seus recursos (MACHULAK *et al.*, 2010).

Neste contexto, uma proposta que tem recebido muita atenção da comunidade *Web* e conseguido grande adoção é o sistema OAuth. O OAuth é um protocolo de autorização aberto e centrado no **Usuário** que permite o compartilhamento de recursos entre diferentes aplicações *web* sem que haja divulgação das credenciais do **Usuário**. A versão 1.0 do protocolo OAuth foi publicada como RFC 5849 (HAMMER-LAHAV, 2010) e rapidamente tornou-se o padrão da indústria para autorização na *Web*. Posteriormente, a versão 2.0 do protocolo (HAMMER-LAHAV; RECORDON; HARDT, 2012) foi criada para solucionar problemas de escalabilidade e complexidade da primeira versão e, apesar de ainda não ser um padrão oficial, já se encontra estável e já foi adotada por vários dos principais nomes da indústria digital, como Facebook, Google e Twitter.

Por sua ampla adoção, simplicidade e interoperabilidade com os sistemas atuais, o OAuth 2.0 será a solução para autorização utilizada neste trabalho. O protocolo OAuth será discutido com mais detalhes no Capítulo 4.

2.2.3 Compartilhamento e Gerenciamento de Dados na Web

Esta seção contém uma revisão bibliográfica dos trabalhos que realizam gerenciamento e compartilhamento de recursos do **Usuário** na *Web*. Tal revisão será empregada como base para avaliar as soluções existentes e compreender suas limitações e pontos fortes. Além dos mecanismos de gerenciamento de identidade e autorização, vistos nas seções anteriores, um Sistema de Gerenciamento e Compartilhamento de Dados também deve definir um protocolo para compartilhamento de recursos e uma interface com o **Usuário** para gerenciamento de recursos.

2.2.3.1 Li et al

Em (LI; LUO, 2009), os autores descrevem um SGCD com foco na experiência do **Usuário**. A arquitetura proposta (Figura 3) possui três componentes: o **Usuário** que precisa compartilhar recursos entre duas aplicações *web*; uma aplicação *web* que armazena os dados do **Usuário**, chamada Provedor de Dados; e uma aplicação *web* que requisita dados do **Usuário**, chamada Consumidor de Dados.

Figura 3. Arquitetura do sistema descrito em (LI; LUO, 2009) (Adaptado de (LI; LUO, 2009))

Quando um **Usuário** precisa compartilhar dados que estão no Provedor de Dados com o Consumidor de Dados, ele notifica o Provedor de que um conjunto de dados deve ser compartilhado. Em seguida, o **Usuário** recomenda que o Consumidor solicite os dados ao Provedor. O Consumidor faz a solicitação ao Provedor, que entrega os dados após autenticar o Consumidor. Ao final, o **Usuário** recebe uma notificação de que os dados foram compartilhados com sucesso.

A interface do sistema usa RDF⁶ (*Resource Description Framework*) para construir os modelos de dados, permitindo que o **Usuário** “copie” os dados do serviço Provedor e “cole” no serviço Consumidor. Na verdade, o que é realmente copiado é um trecho do texto RDF. O Consumidor irá processar esse texto RDF, extrair informações desse texto e enviá-las para o Provedor com uma requisição de dados. Dessa forma, o compartilhamento torna-se intuitivo para o **Usuário**, que não precisa entender o modelo de dados por trás do compartilhamento.

A abordagem usada nesse trabalho possui algumas limitações. A primeira delas é relacionada à autenticação do serviço Consumidor no serviço Provedor para aquisição de *tokens* de acesso; o artigo não explica como a autenticação é feita e como ela pode ser relacionada ao pedido de autorização do **Usuário**. Esta limitação é particularmente importante quando os serviços não são conhecidos previamente. A segunda limitação é relacionada à política de autorização de acesso que o **Usuário** define para os seus dados; o trabalho não trata como são definidos os direitos de acesso e como Provedor deve gerenciar os recursos protegidos do **Usuário**. Além disso, este esquema não possui nenhuma centralização e obriga as aplicações a implementarem ambos os componentes consumidor e provedor para que elas sejam capazes de trocar dados entre si, ora para manipular dados de outra aplicação, ora para atender a requisições de acesso. Por fim, a API de troca de informações entre consumidor e provedor não é discutida em detalhes e não define exatamente como as etapas de comunicação acontecerão.

2.2.3.2 Hsu et al

Em (HSU; CHEN, 2009), os autores propõem um sistema de arquivos baseado em *mashups* (um *website* que combina conteúdo de um ou mais *websites*) para oferecer um serviço de armazenamento de dados dos **Usuários**. O trabalho defende que o armazenamento de dados deve ser separado das demais funcionalidades das aplicações *web*, livrando-as de desenvolver e administrar seus próprios mecanismos. Além disso, os **Usuários** ganham mais controle sobre seus dados uma vez que seus arquivos são administrados independentemente das aplicações *web*.

⁶ O RDF (*Resource Description Framework*) é um arcabouço especificado pelo W3C (*World Wide Web Consortium*) para representação de informações na *Web* (KLYNE; CARROLL; MCBRIDE, 2004). Esse arcabouço permite descrever recursos na *Web* por meio da adoção de padrões de metadados.

O serviço possui dois componentes: um servidor, que contém o sistema de arquivos, e um cliente, que é executado no navegador *web* do **Usuário**. O servidor fornece armazenamento de dados persistente e expõe uma API de acesso para o cliente, que acessa o servidor usando o protocolo RPC (*Remote Procedure Call*) por meio de HTTP. As aplicações *web* interagem apenas com o cliente que disponibiliza uma API JavaScript. Dessa forma, o cliente basicamente traduz e redireciona as requisições originadas nas aplicações *web* para o servidor de sistema de arquivos. A interface do cliente permite que o **Usuário** autorize o acesso a seus arquivos por meio de um navegador de arquivos utilizando um esquema baseado em capacidades (conforme explicado na seção 2.2.2). A proposta central do trabalho é integrar o serviço de sistema de arquivos com uma ou mais aplicações *web* em um *mashup* no navegador *web* do **Usuário**. Isso é conseguido utilizando uma aplicação *web* chamada integrador que cria conteúdo a partir de vários fornecedores de conteúdo. Neste caso, o integrador irá conectar aplicações *web* com o serviço do sistema de arquivos e intermediar a comunicação entre eles.

A Figura 4 mostra a janela do navegador *web* do **Usuário** quando ele visita o *site* do integrador para um exemplo de *mashup* que integra aplicações para manipulação de fotos. Neste exemplo, uma aplicação oferece edição de fotos (*editor.com*) e outra aplicação fornece organização de fotos (*organizer.com*), mas nenhuma delas armazena os dados do **Usuário**, pois ele optou por armazenar suas fotos no serviço *filesystem.com*, que contém o código do cliente para o sistema de arquivos. O integrador cria três quadros em que são carregadas as páginas das aplicações *editor.com*, *organizer.com* e *filesystem.com* e cria canais de comunicação entre essas aplicações. No passo 1, a aplicação *editor.com* faz a requisição ao cliente do serviço de sistema de arquivos para abrir um arquivo. O cliente, então, exibe um navegador de arquivos onde o **Usuário** escolhe qual arquivo deseja compartilhar (passo 2) e, se o **Usuário** permitir o compartilhamento, envia uma requisição ao servidor (passo 3). No passo 4, o servidor devolve ao cliente uma capacidade para esse arquivo. Por fim, o cliente entrega a capacidade para a aplicação *editor.com* que passará a ter acesso ao arquivo.

Figura 4. Arquitetura Mashup (HSU; CHEN, 2009)

A principal limitação desse trabalho diz respeito à construção de *mashups*, pois, apesar de alguns trabalhos mostrarem avanços nessa área (WANG *et al.*, 2007)(CRITES; HSU; CHEN, 2008), ainda existem desafios importantes em relação à segurança e usabilidade de *mashups* que não foram resolvidos (SMETTERS, 2008). Em especial, o integrador deve ser capaz de lidar com a complexidade de diferentes aplicações *web* a cada vez que o **Usuário** precisar compartilhar um arquivo com um *website* diferente. Além disso, o sistema não estabelece mecanismos de revogação de autorização de acesso por parte do **Usuário**.

2.2.3.3 Protocolo UMA

Em (MACHULAK *et al.*, 2010), os autores apresentam um SGCD com base no protocolo UMA (*User-Managed Access*). A proposta permite que os **Usuários** controlem o acesso de aplicações *web* aos seus recursos protegidos, distribuídos entre diversos *websites*, por meio de um gerenciador centralizado de autorização que toma decisões com base em suas preferências. Os componentes do sistema são o **Usuário**, um gerenciador de autorização, vários *hosts* que armazenam de forma distribuída os recursos e um serviço requisitante, conforme mostra a Figura 5.

Figura 5. Arquitetura UMA (Adaptado de (MACHULAK *et al.*, 2010))

O esquema de controle de acesso utilizado, chamado Controle de Acesso Baseado em Reivindicações (*Claim Based Access Control*), é um tipo de ABAC em que não é necessário conhecer previamente as identidades dos serviços requisitantes, pois as políticas são definidas com base em atributos que deverão ser posteriormente comprovados pelo serviço no momento do pedido de um *token* de autorização. Essas informações podem ser, por exemplo, uma asserção de identidade assinada por uma terceira parte confiável, uma promissória concordando com os termos de uso ou um conjunto de informações pessoais, como nome e endereço.

Na primeira parte do protocolo UMA, o **Usuário** deve registrar os *Hosts* onde seus recursos estão armazenados no Servidor de Autorização, mapear estes recursos, e criar as políticas de acesso. Na segunda parte, o Serviço Requisitante pede a um *Host* para acessar um recurso protegido do **Usuário** e o *Host* responde com a localização do servidor de autorização. Então, o Serviço requisita um *token* de acesso ao Servidor de Autorização, que decide se entregará o *token* com base nas políticas previamente definidas pelo **Usuário** e pode solicitar que o requisitante forneça alguma prova confirmando determinadas informações contidas na política. Na terceira parte, o Serviço Requisitante entrega o *token* de acesso recebido para o *Host*, que poderá validá-lo localmente ou submetê-lo para validação no Servidor de Autorização. Se o *token* for válido, o recurso é entregue ao Serviço Requisitante.

Se, por um lado, o UMA é flexível e abrange um grande número de cenários de compartilhamento, por outro lado, o seu protocolo é complexo e adiciona um custo considerável na comunicação do sistema. Além disso, o sistema não especifica a

linguagem adotada para a definição de políticas e como um **Usuário** médio na *Web* conseguirá descrever políticas que reflitam as suas preferências de compartilhamento. Outro problema é que as decisões de autorização do sistema baseiam-se apenas em políticas previamente definidas pelo **Usuário** sem possibilitar a criação dinâmica de políticas no momento do compartilhamento.

2.2.3.4 *Menagerie*

O trabalho descrito em (GEAMBASU *et al.*, 2008) utiliza uma abordagem semelhante ao sistema de arquivos Linux para representar os dados dos **Usuários**, que estão distribuídos em várias aplicações *web*, em uma estrutura hierárquica de pastas dentro do diretório raiz do **Usuário**. Ele suporta o compartilhamento de arquivos por meio de um esquema de Controle de Acesso Baseado em Capacidades (conforme explicado na seção 2.2.2), ou seja, *tokens* compostos por um identificador global único e por um conjunto de direitos de acesso a um conjunto de recursos.

Como prova de conceito, foi desenvolvido um protótipo chamado **Menagerie** que fornece um sistema de arquivos virtual que indexa os dados de aplicações *web* heterogêneas, permitindo que cada aplicação exporte o seu conjunto de arquivos em um espaço de nomes (*namespace*) próprio. Esses arquivos são combinados em uma estrutura hierárquica no sistema de arquivos do **Menagerie** que é capaz de identificar univocamente cada um deles. Dessa forma, o **Usuário** pode montar este sistema de arquivos virtual em seu computador local e manipulá-lo como um sistema de arquivos convencional para compartilhamento de seus arquivos com aplicações *web* ou com outros **Usuários** do **Menagerie**.

Apesar de oferecer muitos recursos para o gerenciamento de dados distribuídos dos **Usuários**, o **Menagerie** não suporta que uma aplicação *web* inicie um pedido de permissão de acesso a um arquivo do **Usuário**, pois as entregas de capacidades de acesso a arquivos são sempre iniciadas pelo **Usuário** via interface do **Menagerie**. Além disso, para que o **Usuário** possa indexar os arquivos de uma aplicação *web* no seu sistema de arquivos virtual, essa aplicação é obrigada a implementar o componente cliente do sistema de arquivos **Menagerie** e aceitar que o **Usuário** compartilhe os seus recursos com outras aplicações usando capacidades geradas pelo sistema **Menagerie**.

2.2.3.5 PeCMan

Seguindo conceitos similares aos descritos no Menagerie, um grupo de pesquisadores da Alcatel-Lucent Bell Labs criou uma ferramenta chamada **PeCMan** (*Personal Content Manager*) (DAENEN *et al.*, 2010) que permite que o **Usuário** gerencie, indexe e compartilhe os seus recursos na Internet, como, por exemplo, fotografias, e-mail, documentos de texto, ou qualquer dado endereçável na Internet.

A solução apresenta as seguintes vantagens em relação ao Menagerie: o maior aprofundamento das questões relacionadas à representação de metadados, permitindo que o **Usuário** identifique seu recurso com *tags* e realize buscas; o suporte à agregação de dados não apenas entre aplicações *web*, mas também com os dados contidos em qualquer tipo de dispositivo pessoal na internet (e.g., celulares e computadores pessoais); o uso de protocolos abertos amplamente utilizados na *web* como OpenID, para autenticação e *Single Sign-On*, e OAuth, para autorização e compartilhamento.

O servidor **PeCMan** comporta-se como um cliente do protocolo OAuth (como será visto no Capítulo 4) e, uma vez autorizado pelo **Usuário**, recebe um *token* (para leitura, escrita ou remoção) do serviço *web* que armazena um determinado dado do **Usuário** (e.g., uma planilha do Google Docs). Dessa forma, o servidor **PeCMan** pode fazer *upload*, *download* e remoção de arquivos em serviços *web* de terceiros, funcionando como uma unidade virtual de gravação e leitura em que o sistema endereça cada recurso individual por URLs, sem armazenar de fato o recurso. Para conseguir se comunicar com os serviços de terceiros, ele implementa *proxies* que funcionam como tradutores para as APIs de compartilhamento das aplicações mais comumente usadas na internet, como Google Docs, Flickr e YouTube.

O sistema **PeCMan** é composto por quatro componentes: cliente, servidor, *driver* do dispositivo e aplicações *web*, conforme mostrado na Figura 6. O cliente representa a camada de apresentação para o **Usuário** e pode ser um portal *web* ou uma aplicação para PC ou telefone celular. O servidor é o *gateway* de acesso para os recursos do **Usuário**. O *driver* do dispositivo fornece acesso ao sistema de arquivos local dos dispositivos conectados para armazenar e recuperar recursos.

Figura 6. Arquitetura PeCMan (Adaptado de (DAENEN *et al.*, 2010))

O sistema **PeCMan** possibilita que o **Usuário** gerencie seus recursos na Internet em uma interface centralizada e compartilhe seus recursos com outros **Usuários**, que também precisam estar registrados no sistema. Entretanto, uma das maiores limitações do **PeCMan** é não permitir que o **Usuário** compartilhe seus recursos com aplicações *web*. Outra limitação está relacionada ao gerenciamento de autorizações, pois, embora o sistema permita indexar recursos armazenados em aplicações *web*, os *tokens* de autorização são gerenciados individualmente por essas aplicações e não pelo **PeCMan**, que, nesse sentido, é apenas um consumidor de *tokens*.

2.2.3.6 OpenIDemail

Em (SUN; HAWKEY; BEZNOSOV, 2009), os autores propõem uma abordagem para o compartilhamento de recursos na *Web*. As principais idéias por trás de sua abordagem são: reutilização de contas de e-mail existentes para identificação global, extensão de listas de contatos baseadas em e-mail para incluir as noções de confiança e delegação, e utilização de URLs secretos para compartilhamento de recursos.

A Figura 7 mostra os componentes do sistema **OpenIDemail**: o **Usuário (U)**, os Provedores OpenIDemail (**O**), o Provedor OpenPolicy (**P**), o Provedor de Serviço (**S**) e o Destinatário (**D**).

Figura 7. Arquitetura do sistema OpenIDemail (Adaptado de (SUN; HAWKEY; BEZNOSOV, 2009))

A abordagem tem dois componentes principais: o Provedor OpenIDemail e o Provedor OpenPolicy. Um Provedor OpenIDemail é um provedor de e-mail que é estendido para suportar o protocolo OpenID (como será visto no Capítulo 4) e um serviço de mapeamento de e-mails para identificadores OpenID, de forma que os **Usuários** da *Web* possam usar o seu e-mail para fazer *login* em múltiplas aplicações na Internet. Um Provedor OpenPolicy é um provedor de políticas de controle de acesso que oferece serviços para o **Usuário** organizar as suas políticas e para as aplicações *web* tomarem decisões de autorização. Para especificação das políticas é utilizada uma linguagem que combina os conceitos de RBAC, confiança e delegação. Nessa arquitetura, os Provedores OpenPolicy e os Provedores de Serviço implementam OpenID, por isso, uma vez que o **Usuário** se autenticou no seu Provedor OpenIDemail, os outros componentes são acessados de forma transparente.

A seguir, descreve-se o fluxo de mensagens do sistema **OpenIDemail** para um exemplo em que um **Usuário (U)** deseja compartilhar um de seus recursos, que está armazenado em um Provedor de Serviço (**S**), com um **Usuário Destinatário (D)**. Inicialmente, **U** autentica-se no seu Provedor OpenIDemail (**O**) e, depois, informa a **S** qual recurso deve ser compartilhado. **S**, então, gera um URL secreto para esse recurso e redireciona **U** para o seu Provedor OpenPolicy (**P**) para definição de políticas de controle de acesso baseadas em papéis de acordo com a lista de contatos de **U**. Depois da definição de políticas, **P** envia um e-mail contendo o URL secreto criado por **S** para cada contato (da

lista de contatos de **U**) que exerce o papel especificado na política (e.g., grupo que inclui todos os membros da família). Para acessar o recurso compartilhado, o Destinatário (**D**), que recebeu o e-mail, deve apresentar o URL para **S** depois de autenticar-se no seu respectivo Provedor OpenIDemail. Quando recebe a requisição, **S** faz uma consulta a **P** para saber se **D** pode realmente acessar o recurso de acordo com as políticas de **U**. Se **P** retornar uma asserção positiva, **S** entrega o recurso para **D**.

Essa arquitetura foca apenas no compartilhamento de recursos entre **Usuários**. Dessa forma, o sistema **OpenIDemail** não oferece mecanismos para que uma aplicação *web* requisiute acesso a um conjunto de recursos do **Usuário** ou que o **Usuário** salve seus recursos em aplicações *web*.

2.2.3.7 Serviços de Compartilhamento em Redes Sociais

Em redes sociais os dados do **Usuário** podem ser classificados em três tipos (KO *et al.*, 2010): dados de identidade, que descrevem quem o **Usuário** é na *Web Social*, incluindo a identidade, informações de perfil, e política de privacidade; dados do grafo social, que representam quem o **Usuário** conhece na *Web Social*, incluindo as conexões de amizade (e.g., família e amigos); e dados de conteúdo, que representam o que o **Usuário** tem na *Web Social*, incluindo mensagens, fotos, vídeos e outros dados criados como resultado de várias atividades sociais na Internet. Esses tipos de dados são mostrados na Figura 8, assim como uma classificação que divide as interfaces de programação (APIs – *Application Programming Interfaces*) de compartilhamento em redes sociais em: autenticação, autorização, publicação de atividades e aplicações sociais.

Figura 8. Serviços de compartilhamento em redes sociais (Adaptado de (KO *et al.*, 2010))

Diversas redes sociais, como, por exemplo, Google, Yahoo, Facebook, MySpace e Twitter, disponibilizam APIs para permitir a aplicações de terceiros acessar, processar, escrever e agregar os dados do **Usuário** (KO *et al.*, 2010). A plataforma do Facebook usa o protocolo OAuth 2.0 para autenticação e autorização e a API proprietária *Graph* para leitura e escrita de dados que permite requisitar informações sobre conexões entre **Usuários**, fotos, preferências e atividades. Além disso, a plataforma fornece *plugins* que permitem que *websites* de terceiros se integrem ao Facebook de maneira simples, com poucas linhas de HTML; o *plugin* mais importante é o botão de “Like” que permite que os **Usuários** compartilhem páginas *web* com amigos. Já o Google possui uma plataforma, chamada Google Friend Connect, que usa o protocolo OpenID para autenticação e o protocolo OAuth para autorização. Embora tenha objetivos semelhantes, a abordagem do Google difere da do Facebook, pois usa padrões abertos, como o OpenSocial⁷ e Activity Streams⁸, e provedores descentralizados, já que seus serviços permitem integração com diferentes provedores de OpenID, como o Yahoo, provedores de rede social, como o Twitter, e agregadores de atualizações, como o Plaxo Pulse.

Contudo, essas APIs expõem apenas um pequeno conjunto de funções de acesso a um conjunto limitado de recursos do **Usuário** com foco específico em tipos de dados fixos relacionados a redes sociais, como lista de contatos, atividades, fotos e vídeos. Dessa forma, aplicações não relacionadas a redes sociais, ou que desejem acessar tipos de dados diferentes dos dados pré-definidos pela API, não possuem uma opção para compartilhar dados. Além disso, as APIs são, em geral, incompatíveis entre si, não permitindo que elas sejam facilmente reutilizadas para compartilhar recursos distribuídos em diferentes domínios de aplicação.

2.3 Características Desejáveis

Com base no estudo dos trabalhos relacionados (Seção 2.2) e nos benefícios e limitações que tais soluções oferecem, foi levantado um conjunto de características desejáveis para um SGCD (Sistema de Gerenciamento e Compartilhamento de Dados) de forma a endereçar o problema de gerenciamento e compartilhamento de dados do **Usuário**

⁷ OpenSocial Foundation website: <http://www.opensocial.org/>

⁸ Activity Streams website: <http://activitystrea.ms/>

entre aplicações *web* descrito na seção 1.1. A discussão dessas características servirá de base para analisar os trabalhos relacionados e ajudará a definir os requisitos de um SGCD para *Web* no próximo capítulo. A seguir são apresentadas essas características e suas respectivas descrições.

Centrado no Usuário

A idéia central do paradigma centrado no **Usuário** é que o **Usuário** deve ter o controle das transações que envolvem a divulgação de sua identidade e de seus recursos (BHARGAV-SPANTZEL *et al.*, 2007). Em um SGCD centrado no **Usuário**, ele é considerado o dono dos recursos e deve autorizar explicitamente o compartilhamento deles. Assim, o **Usuário** determina os recipientes de recursos e define suas preferências de compartilhamento.

Existência de mecanismos de Single Sign-On

A técnica de *Single Sign-On* (SSO) permite que o **Usuário** se autentique apenas uma vez e tenha seu *login* realizado automaticamente nos serviços utilizados posteriormente (PASHALIDIS; MITCHELL, 2003). Essa abordagem evita que o **Usuário** precise criar e manter várias contas para autenticação e resolve o problema de segurança gerado pela reutilização de senhas em *sites* diferentes (IVES; WALSH; SCHNEIDER, 2004). Dessa forma, a utilização de mecanismos de SSO possibilita o acesso seguro e fácil a múltiplos serviços na *Web* (DAENEN *et al.*, 2010).

Delegação de permissão de acesso

Quando uma aplicação *web* requisita acesso aos recursos protegidos do **Usuário**, este não deve ser forçado a fornecer as suas credenciais na aplicação *web* que armazena os recursos. Ao invés disso, o **Usuário** deve poder delegar uma permissão de acesso que define um período de tempo durante o qual a permissão é válida e um conjunto de recursos que a aplicação requisitante pode acessar. É importante que o **Usuário** possa delegar exatamente a permissão que é suficiente e necessária para que a aplicação possa desempenhar a tarefa desejada. Outra característica importante é que essa permissão deve poder ser revogada a qualquer momento pelo **Usuário**.

Gerenciamento centralizado

Na *Web* atual, os recursos do **Usuário** e as respectivas políticas de controle de acesso estão espalhados em diversos serviços. Neste contexto, o **Usuário** não tem uma visão global de seus recursos e perde o controle sobre eles (HSU; CHEN, 2009). Assim, é importante permitir que o **Usuário** gerencie seus recursos e políticas de forma centralizada, independentemente de esses recursos e políticas serem armazenados de forma centralizada ou distribuída; por exemplo, recursos armazenados de forma distribuída podem ser indexados e mostrados em uma lista para o **Usuário**.

Proteção dos serviços de segurança

A segurança de um sistema está relacionada aos serviços de confidencialidade, integridade e disponibilidade (BISHOP, 2002). Em um SGCD, é fundamental implementar as melhores práticas de segurança para assegurar que somente as entidades autorizadas pelo **Usuário** poderão ler ou modificar os recursos protegidos e que o sistema estará disponível para acesso aos recursos.

Suporte ao compartilhamento de tipos de dados genéricos

Atualmente, a maioria dos mecanismos existentes na *Web* para compartilhamento de recursos do **Usuário** é restrita a tipos de dados relacionados a um nicho de aplicação (por exemplo, as APIs que compartilham apenas tipos fixos estritamente relacionados a redes sociais). Para suportar serviços com diferentes lógicas de negócio que precisam manipular diferentes tipos de dados, é importante que um SGCD permita o compartilhamento de qualquer tipo de dado (e.g., imagens, dados cifrados, lista de contatos e formatos binários).

Compartilhamento integrado ao fluxo do serviço web

É importante que as aplicações *web* possam iniciar requisições de compartilhamento de forma integrada ao fluxo do seu próprio serviço. Assim, o **Usuário** pode receber requisições de compartilhamento enquanto utiliza um serviço *web* e não precisa estar previamente conectado a um serviço de compartilhamento para autorizar o acesso aos seus recursos.

Complexidade baixa para as aplicações web

Para facilitar a adoção de novos desenvolvedores, é importante que as operações mais complexas, em termos de implementação, relacionadas ao gerenciamento e compartilhamento de recursos do **Usuário** sejam realizadas por um serviço especializado, de forma que o sistema ofereça complexidade baixa para as aplicações *web*. Assim, o desenvolvedor de uma aplicação pode se concentrar na lógica de negócios do seu serviço ao invés de ter que implementar mecanismos custosos para o gerenciamento e compartilhamento de recursos.

Uso de padrões abertos e amplamente adotados

O uso da infra-estrutura *Web* já existente, assim como a adoção de um conjunto de padrões abertos e amplamente utilizados, tornam mais fácil atrair novos provedores e **Usuários**, cujos serviços já são baseados nesses padrões.

Protocolos de comunicação leves

A escolha dos protocolos de comunicação de um SGCD é particularmente importante, pois os mesmos não podem adicionar um *overhead* de comunicação e processamento que inviabilize sua implantação em larga escala na *Web*.

Liberdade para que o Usuário escolha o seu SGCD

É importante que o **Usuário** possa escolher livremente as entidades responsáveis por armazenar seus recursos protegidos e controlar suas autorizações de compartilhamento, garantindo-se que essas entidades sejam confiáveis do ponto de vista do **Usuário**.

2.4 Comparação entre os trabalhos relacionados

A Tabela 1, a seguir, compara os trabalhos relacionados discutidos na seção 2.2.3 em relação às características desejáveis para um SGCD levantadas na seção 2.3.

Tabela 1. Comparação entre os trabalhos relacionados em relação às características desejáveis de um SGCD

	Li et al	Hsu et al	UMA	Menagerie	PeCMan	OpenIDemail	Facebook	Google
Centrado no usuário	✓	✓	✓	✓	✓	✓	✓	✓
Existência de mecanismos de <i>Single Sign-On</i>	✗	✗	✗	✗	✓	✓	✓	✓
Delegação de permissão de acesso	✓	✓	✓	✓	✓	✓	✓	✓
Gerenciamento centralizado	✗	✓	✓	✓	✗	✓	✓	✓
Proteção dos serviços de segurança	✗	✓	✓	✓	✓	✓	✓	✓
Suporte ao compartilhamento de tipos de dados genéricos	✓	✗	✓	✓	✓	✓	✗	✗
Compartilhamento integrado ao fluxo do serviço web	✓	✓	✓	✗	✗	✗	✓	✓
Complexidade baixa para as aplicações web	✗	✓	✗	✗	✗	✗	✓	✓
Uso de padrões abertos e amplamente adotados	✗	✗	✗	✗	✓	✗	✗	✓
Protocolos de comunicação leves	✓	✗	✗	✓	✓	✗	✓	✓
Liberdade para que o usuário escolha o seu SGCD	✓	✓	✓	✗	✗	✓	✗	✗

Legenda	
✓	Satisfaz
✗	Não satisfaz

Conforme a Tabela 1 mostra, não existe uma solução adequada para compartilhar dados dos **Usuários** entre aplicações *web* que atenda a todas as características levantadas na Seção 2.3. Este trabalho trata desse problema e propõe uma arquitetura, apresentada no Capítulo 5, de um SGCD que visa atender a todas as características descritas.

2.5 Resumo do Capítulo

Este capítulo discutiu o problema de gerenciamento e compartilhamento de dados na *Web* apresentado no capítulo de introdução. O objetivo principal foi oferecer uma base de conhecimento para entender o problema e como resolvê-lo.

Com esse objetivo, inicialmente, o capítulo definiu os principais conceitos e uma arquitetura genérica de um Sistema de Gerenciamento e Compartilhamento de Dados

(SGCD). Essa discussão inicial mostrou que um SGCD precisa de mecanismos de gerenciamento de identidade e de autorização para executar as suas funcionalidades principais. Assim, foi apresentada uma revisão bibliográfica nessas áreas para definir mecanismos adequados para o sistema em questão.

Depois, foi realizada uma análise dos trabalhos relacionados que realizam gerenciamento e compartilhamento de recursos do **Usuário** na *Web*. Com base nesse estudo e nos benefícios e limitações que tais soluções oferecem, levantaram-se as características desejáveis para um SGCD.

A discussão dessas características, feita de maneira mais informal neste capítulo, será utilizada como base para realizar a análise de requisitos de um SGCD para *Web* apresentada no próximo capítulo.

3 ANÁLISE DE REQUISITOS

O Capítulo 2 descreveu a arquitetura básica, as definições e as características desejáveis de um SGCD (Sistema de Gerenciamento e Compartilhamento de Dados) para *Web*. Com base nessas características, este capítulo apresenta uma análise detalhada dos requisitos de um SGCD para *Web*, que orientarão o desenvolvimento dos capítulos que se seguem.

3.1 *Escopo*

O sistema sob estudo será representado pelos componentes **Servidor de Autorização** e **Servidor de Recursos** de acordo com a arquitetura descrita na seção 2.1, pois estes dois componentes apresentam as principais funcionalidades de um SGCD. Esses componentes podem ser entidades separadas, mas para modelagem de requisitos a comunicação entre eles será considerada interna ao sistema.

3.1.1 *Requisitos Gerais*

Seguem alguns requisitos gerais para o sistema a ser especificado:

- O sistema deve ser desenvolvido para o ambiente *web* e deve considerar as características específicas deste ambiente.
- O sistema deve incluir o **Usuário** como parte central do seu modelo de forma que um **Usuário** com conta no sistema possa organizar e visualizar seus recursos na Internet por meio de um serviço centralizado.
- O sistema deve permitir o compartilhamento de dados do **Usuário** com aplicações *web* com base nas preferências de compartilhamento do **Usuário**. Dessa forma, não será levado em consideração o compartilhamento entre **Usuários** diretamente, embora o sistema possa ser utilizado como infraestrutura para implementação de tal funcionalidade.

3.2 Conceitos e Método

O padrão IEEE Standard 610.12 (RADATZ, 1990) define requisitos como:

- Uma condição ou capacidade necessária para um **Usuário** solucionar um problema ou atingir um objetivo;
- Uma condição ou capacidade que precise ser atendida ou estar presente em um sistema, ou componente do sistema, para satisfazer um contrato, norma, especificação ou outro documento imposto formalmente.

A Análise de Requisitos é definida como o processo de levantamento, entendimento, especificação e validação dos requisitos dos **Usuários** e clientes (ESCALONA; KOCH, 2004). Esta análise, quando aplicada a sistemas de *software*, resulta na especificação das características operacionais do *software*, indica a interface do *software* com os outros elementos do sistema e estabelece restrições a que o *software* deve satisfazer, usando uma combinação de formas textuais e diagramas para facilitar o entendimento e verificação do sistema quanto à correção, completeza e consistência (PRESSMAN, 2006).

Os métodos tradicionais de análise de requisitos não conseguem capturar todos os requisitos de aplicações *web*, pois elas são diferentes de outras categorias de software. Por exemplo, as aplicações *web* possuem diferentes tipos de *stakeholders* (e.g., analistas, clientes, **Usuários**, designers e especialistas em *marketing*) e têm aspectos especiais de projeto (e.g., estrutura de navegação, foco no conteúdo, evolução contínua, concorrência, carga imprevisível) (PRESSMAN, 2006). De forma geral, os métodos de análise de requisitos para aplicações *web* devem ajudar a entender o conteúdo a ser entregue pela aplicação, a função a ser fornecida para o **Usuário** final e os modos de interação que cada classe de **Usuários** irá requerer à medida que a navegação pela aplicação *web* ocorre.

Dessa forma, é importante seguir um processo sistemático de análise de requisitos que seja capaz de capturar os requisitos específicos dos sistemas desenvolvidos para *Web*. Para atender a essa necessidade, existem vários métodos especializados de Análise de Requisitos em Aplicações *Web*, tais como OOHDM (*Object Oriented Hypermedia Design Model*), WSDM (*Web Site Design Method*), SOHDM (*Scenario-based Object Oriented*

Hypermedia Design Methodology), UWE (*UML-based Web Engineering*), WebML (*Web Modeling Language*), OOH (*Object-Oriented Hypermedia Method*), OOWS (*Object Oriented Web Solutions*) e NDT (*Navigational Development Techniques*) (VALDERAS; PELECHANO, 2011). Para este estudo, escolheu-se o método NDT, pois ele fornece os mecanismos necessários para descrever apropriadamente todos os requisitos de uma aplicação *web*, abrangendo a captura de requisitos de dados, funcionais, não funcionais e navegacionais (VALDERAS; PELECHANO, 2011) (ESCALONA; KOCH, 2004).

3.2.1 Método NDT

O método NDT (*Navigational Development Techniques*) é uma técnica para especificar e analisar os requisitos de Aplicações *Web* (ESCALONA; ARAGON, 2008) (ESCALONA; TORRES; MEJÍAS, 2002). Esse método baseia-se em seis atividades principais: definir os objetivos, definir os requisitos de armazenamento, definir os atores do sistema, definir os requisitos funcionais, definir os requisitos interativos e definir os requisitos não funcionais. A Figura 9 mostra essas atividades e representa as relações de precedência entre elas por meio de setas.

Figura 9. Fluxo de atividades do método NDT (*Navigational Development Techniques*) (Adaptado de (ESCALONA; TORRES; MEJÍAS, 2002))

3.3 Análise de Requisitos usando o Método NDT

Esta seção descreve como cada atividade do método NDT foi aplicada ao SGCD sob estudo para análise de requisitos do sistema. O processo para especificar os objetivos e requisitos é baseado em modelos (*templates*) na forma de tabelas que descrevem os requisitos de forma estruturada. Dessa forma, cada atividade do processo gera uma tabela para cada requisito ou objetivo.

Para facilitar o entendimento dos requisitos, esta seção mostrará somente uma tabela para cada atividade com as informações mais importantes para o processo de especificação de requisitos. As tabelas detalhadas segundo os modelos definidos pelo método NTD podem ser encontradas no **Apêndice A**.

3.3.1 Atividade 1: Definição dos Objetivos

A *Tabela 2* mostra a descrição dos objetivos do sistema sob estudo segundo um modelo composto por um identificador que dá nome a cada objetivo e uma descrição.

Tabela 2. Descrição dos objetivos do sistema

Identificador	Nome	Descrição
OBJ-01	Gerenciamento de Contas dos Usuários	Gerenciar contas de Usuários (visualizar/criar/editar recursos, visualizar/criar/editar autorizações, fornecer histórico de atividades).
OBJ-02	Autenticação de Usuários usando um Provedor de Identidade	Usar um Sistema de Gerenciamento de Identidade para autenticar os Usuários .
OBJ-03	Compartilhamento de recursos do Usuário com Aplicações <i>Web</i>	Compartilhar recursos do Usuário com Aplicações <i>Web</i> (requisição de autorização ao Usuário , entrega de <i>tokens</i> , entrega de recursos).
OBJ-04	Proteção dos recursos do Usuário	O sistema deve ser seguro de forma a proteger os recursos do Usuário contra acesso não autorizado.
OBJ-05	Implantação do sistema em larga escala na Internet	O sistema deve ser possível de ser implantado em larga escala na Internet.

3.3.2 Atividade 2: Definição dos Requisitos de Armazenamento

De acordo com os objetivos definidos na fase anterior, devem-se definir os requisitos de armazenamento de acordo com as informações que serão armazenadas no sistema e criar modelos compostos por um identificador que dá nome ao requisito de armazenamento, os objetivos associados e uma descrição dos dados a serem armazenados. A *Tabela 3* mostra a descrição dos requisitos de armazenamento do sistema. Maiores detalhes sobre os modelos para cada um dos requisitos de armazenamento podem ser encontrados no **Apêndice A**.

Tabela 3. Descrição dos requisitos de armazenamento do sistema

Identificador	Nome	Objetivos Associados	Descrição	Dados Armazenados
RA-01	Conta do Usuário	<ul style="list-style-type: none"> – OBJ-01 – OBJ-02 	O sistema deve armazenar as informações sobre os Usuários .	<ul style="list-style-type: none"> – Identificador do Usuário – Identificador do Provedor de Identidade do Usuário – Data de criação da conta
RA-02	Recursos do Usuário	<ul style="list-style-type: none"> – OBJ-01 – OBJ-03 	O sistema deve armazenar as informações sobre os recursos dos Usuários .	<ul style="list-style-type: none"> – Identificador do recurso – Identificador do Usuário – Tipo – Descrição – Criador – <i>Tags</i> – Caminho do diretório de armazenamento – Nome do arquivo – Tamanho do Arquivo – Arquivo – Data de criação
RA-03	Autorizações	<ul style="list-style-type: none"> – OBJ-01 – OBJ-03 – OBJ-04 	O sistema deve armazenar as informações sobre as autorizações de compartilhamento dos Usuários .	<ul style="list-style-type: none"> – Identificador da autorização – <i>Site</i> autorizado – Recursos compartilhados – Escopo de leitura – Escopo de escrita – Token – Tipo de token – Data de expiração – Data de criação
RA-04	Histórico	<ul style="list-style-type: none"> – OBJ-01 	O sistema deve armazenar informações sobre o histórico de atividades da conta do Usuário (por exemplo, acessos a recursos, criação de recursos e criação de autorizações).	<ul style="list-style-type: none"> – Identificador do Usuário – Atividade – Descrição – Data de criação
RA-05	Sites registrados	<ul style="list-style-type: none"> – OBJ-03 – OBJ-04 	O sistema deve armazenar informações de registro dos <i>sites</i> incluindo os segredos de autenticação.	<ul style="list-style-type: none"> – URL – Descrição – Chave – Segredo – Data de expiração – Data de criação

3.3.3 Atividade 3: Definição dos Atores

Nesta atividade são definidos os atores, ou seja, as entidades que interagem/usam o sistema. A partir do funcionamento do sistema descrito na seção 2.1, é possível definir os seguintes três atores: **Usuário**, **Provedor de Serviço** e **Provedor de Identidade**, conforme mostrado na Figura 10.

Figura 10. Atores do sistema

Para cada um dos atores existe um modelo composto por um identificador que dá nome ao ator, os objetivos associados e uma descrição. A *Tabela 4* mostra esses modelos para os atores do sistema.

Tabela 4. Descrição dos atores do sistema

Identificador	Nome	Objetivos Associados	Descrição
AT-01	Usuário	<ul style="list-style-type: none"> – OBJ-01 – OBJ-02 – OBJ-03 – OBJ-04 	O Usuário que possui uma conta no sistema.
AT-02	Provedor de Serviço	<ul style="list-style-type: none"> – OBJ-03 – OBJ-04 	O Provedor de Serviço que quer acessar os recursos de um Usuário do sistema.
AT-03	Provedor de Identidade	<ul style="list-style-type: none"> – OBJ-02 	O Provedor de Identidade que comprova a identidade de um Usuário do sistema.

3.3.4 Atividade 4: Definição dos Requisitos Funcionais

Os requisitos funcionais são as capacidades ou funcionalidades que o sistema deve exibir para resolver o problema. O método NDT sugere que seja utilizada a técnica de Casos de

Uso (MILES; HAMILTON, 2006) para levantamento dos requisitos funcionais. Casos de uso descrevem como os atores vão interagir com a aplicação *web* para realizar uma ação específica. As figuras *Figura 11* e *Figura 12* mostram os Casos de Uso desenvolvidos para descrição dos requisitos funcionais do sistema para o gerenciamento de contas dos **Usuários** e para o compartilhamento de recursos do **Usuário**.

Figura 11. Casos de Uso do sistema para o gerenciamento de conta

Figura 12. Casos de Uso do sistema para o compartilhamento de recursos

Para cada requisito funcional, deve ser criado um modelo composto por um identificador que dá nome ao requisito, os objetivos associados e uma descrição. A *Tabela 5* mostra esses modelos para os requisitos funcionais dos Casos de Uso de gerenciamento de contas e a *Tabela 6* mostra os modelos para os requisitos funcionais dos Casos de Uso de compartilhamento de recursos. Maiores detalhes sobre os modelos para cada um dos requisitos funcionais e a descrição completa dos casos de uso podem ser encontrados no **Apêndice A**.

Tabela 5. Descrição dos requisitos funcionais para os Casos de Uso de gerenciamento de contas

Identificador	Nome	Objetivos Associados	Descrição	Atores
RF-01	Autenticar Usuário	– OBJ-02 – OBJ-04	O Usuário autentica-se junto ao seu Provedor de Identidade que, por sua vez, entrega uma asserção para o sistema comprovando a identidade do Usuário .	– AT-01: Usuário – AT-02: Provedor de Identidade
RF-02	Consultar recursos	– OBJ-01	O Usuário visualiza os seus recursos armazenados no sistema.	– AT-01: Usuário
RF-03	Criar recursos	– OBJ-01	O Usuário cria recursos na sua conta no sistema.	– AT-01: Usuário
RF-04	Remover recursos	– OBJ-01	O Usuário visualiza os seus recursos armazenados no sistema e pode remover alguns (ou todos) eles.	– AT-01: Usuário
RF-05	Consultar histórico	– OBJ-01	O Usuário visualiza as atividades ocorridas na sua conta do sistema (por exemplo, acessos a recursos, criação de recursos e criação de autorizações).	– AT-01: Usuário
RF-06	Remover Atividades	– OBJ-01	O Usuário remove atividades do seu histórico no sistema.	– AT-01: Usuário
RF-07	Consultar Autorizações	– OBJ-01 – OBJ-03 – OBJ-04	O Usuário visualiza as suas autorizações de compartilhamento aprovadas no sistema, válidas ou expiradas.	– AT-01: Usuário
RF-08	Remover Autorizações	– OBJ-01 – OBJ-03 – OBJ-04	O Usuário visualiza as suas autorizações de compartilhamento aprovadas no sistema, válidas ou expiradas, e pode remover algumas (ou todas) elas. Ao remover uma autorização, o Usuário removerá o acesso concedido a um Provedor de Serviço.	– AT-01: Usuário

Tabela 6. Descrição dos requisitos funcionais para os Casos de Uso de compartilhamento de recursos

Identificador	Nome	Objetivos Associados	Descrição	Atores
RF-09	Requisitar autorização	– OBJ-03 – OBJ-04	Um Provedor de Serviço faz uma requisição ao sistema para acesso a recursos do Usuário . O sistema, por sua vez, exibe a requisição ao Usuário e pergunta se ele autoriza o compartilhamento dos dados. Em caso positivo, o Provedor de Serviço recebe uma autorização.	– AT-01: Usuário – AT-02: Provedor de Serviço
RF-10	Ler Recurso	– OBJ-03 – OBJ-04	Um Provedor de Serviço lê recursos do Usuário no sistema.	– AT-02: Provedor de Serviço
RF-11	Escrever Recurso	– OBJ-03 – OBJ-04	Um Provedor de Serviço escreve recursos do Usuário no sistema.	– AT-02: Provedor de Serviço
RF-12	Autenticar Provedor de Serviço	– OBJ-03 – OBJ-04	O Provedor de Serviço autentica-se junto ao sistema usando uma credencial (chave+segredo) previamente registrada.	– AT-02: Provedor de Serviço
RF-13	Registrar Provedor de Serviço	– OBJ-03 – OBJ-04	Registro de uma nova conta de um Provedor de Serviço e entrega de credenciais de autenticação.	– AT-02: Provedor de Serviço

3.3.5 Atividade 5: Definição dos Requisitos Interativos

Os requisitos interativos definem as telas em que as informações são mostradas para o **Usuário** e como este pode navegar de uma tela para outra. Essas telas são chamadas de nós de navegação.

O primeiro passo na definição de requisitos interativos é a definição dos critérios de recuperação ou pesquisa dos dados específicos (descritos nos requisitos de armazenamento). No método NDT, esses critérios são denominados frases e são representados por modelos que contém um identificador, os objetivos relacionados e o enunciado do critério de recuperação, conforme mostrado na Tabela 7. Maiores detalhes sobre os modelos para cada uma das frases podem ser encontrados no **Apêndice A**.

Tabela 7. Descrição das frases do sistema

Identificador	Nome	Objetivos Associados	Atores	Descrição
FR-01	Recuperação de Recursos por Tipo	– OBJ-01 – OBJ-03	– AT-01: Usuário	O sistema deve mostrar todos os recursos que são do tipo pesquisado.
FR-02	Recuperação de Atividade por Período	– OBJ-01	– AT-01: Usuário	O sistema deve mostrar todas as atividades com data de criação menor que a data final pesquisada e maior que a data inicial pesquisada.
FR-03	Recuperação de Autorização por Provedor de Serviço	– OBJ-01	– AT-01: Usuário	O sistema deve mostrar todas as autorizações que foram concedidas a um Provedor de Serviço que possui o URL pesquisado.
FR-04	Recuperação de Provedor de Serviço por URL	– OBJ-01 – OBJ-03 – OBJ-04	– AT-01: Usuário	O sistema deve mostrar informações de registro do Provedor de Serviço que possui o URL pesquisado.
FR-05	Recuperação de Recursos por <i>Tag</i>	– OBJ-01	– AT-01: Usuário	O sistema deve mostrar todos os recursos cuja <i>tag</i> contém a <i>tag</i> pesquisada.

Depois de definidas as frases, devem-se definir os requisitos interativos. Para cada requisito interativo, deve ser criado um modelo que descreve as frases utilizadas para recuperar as informações, como a informação será apresentada ao **Usuário** e os nós em que a informação é mostrada, conforme mostrado na *Tabela 8*. Maiores detalhes sobre os modelos para cada um dos requisitos interativos podem ser encontrados no **Apêndice A**.

Tabela 8. Descrição dos requisitos interativos do sistema

Identificador	Nome	Objetivo Associado	Atores	Descrição	Frases	Entrada	Saída
RI-01	Informação sobre os recursos do Usuário	OBJ-01	AT-01: Usuário	O sistema deve mostrar informações sobre os recursos do Usuário .	FR-01 FR-05	RI-02 RI-03	RI-02 RI-03
RI-02	Informação sobre as atividades	OBJ-01	AT-01: Usuário	O sistema deve mostrar informações sobre as atividades realizadas na conta do Usuário .	FR-02	RI-01 RI-03	RI-01 RI-03
RI-03	Informação sobre as autorizações do Usuário	OBJ-01 OBJ-03	AT-01: Usuário	O sistema deve mostrar informações sobre as autorizações de compartilhamento concedidas pelo Usuário .	FR-03	RI-01 RI-02	RI-01 RI-02
RI-04	Informação sobre uma requisição de autorização	OBJ-01 OBJ-03 OBJ-04	AT-01: Usuário	O sistema deve mostrar informações sobre o Provedor de Serviço que requisita um acesso e quais são os recursos que estão sendo requisitados.	FR-01 FR-04	Não tem entradas	Não tem saídas

O modelo da *Tabela 8* também mostra como o **Usuário** pode navegar entre os nós, por meio de uma lista de saídas (nós acessíveis a partir do nó atual) e entradas (nós a partir dos quais é possível acessar o nó atual). Por exemplo, o **Usuário** pode acessar o nó representado pelo requisito RI-01 na *Tabela 8* a partir dos nós RI-02 e RI-03 e pode voltar para os nós RI-02 e RI-03 a partir do nó RI-01.

3.3.6 Atividade 6: Definição dos Requisitos Não Funcionais

A última atividade é a definição dos requisitos não funcionais. Enquanto os requisitos funcionais descrevem o que o sistema faz, os requisitos não funcionais descrevem como

bem o sistema executa suas funções (VERMA, 2009). Esse tipo de requisito age de forma a restringir o sistema, como requisitos de portabilidade, de segurança e de desempenho. Para cada requisito não funcional, deve ser criado um modelo composto por um identificador e uma descrição, conforme mostrado na *Tabela 9*. Maiores detalhes sobre os modelos para cada um dos requisitos não funcionais podem ser encontrados no **Apêndice A**.

Tabela 9. Descrição dos requisitos não funcionais do sistema

Identificador	Nome	Objetivos Associados	Descrição
RNF-01	Confidencialidade e integridade	– OBJ-04	O sistema deve implementar as melhores práticas de segurança para assegurar que somente as entidades autorizadas pelo proprietário dos recursos poderão ler ou modificar os recursos protegidos.
RNF-02	Compartilhamento de tipos de dados genéricos	– OBJ-03	O sistema deve suportar o compartilhamento de qualquer tipo de dado. O sistema não precisa entender as representações dos dados, mas deve permitir que Provedores de Serviço com diferentes lógicas de negócio possam usar o sistema para obter acesso aos dados dos Usuários .
RNF-03	Autorização explícita do Usuário	– OBJ-04	Toda requisição de autorização de acesso a recursos protegidos deve ser explicitamente autorizada pelo Usuário .
RNF-04	Uso de navegador <i>web</i> como agente do Usuário	– OBJ-05	O sistema deve ser compatível com as tecnologias existentes nos navegadores <i>web</i> e não deve exigir que o Usuário instale nenhuma aplicação em seu dispositivo, além do navegador, para utilizar o sistema.
RNF-05	Uso de padrões abertos e amplamente adotados	– OBJ-05	O sistema deve ser construído sobre a infraestrutura de Internet já existente e padrões abertos, de forma que seja mais fácil atrair provedores e Usuários .
RNF-06	Disponibilidade	– OBJ-01 – OBJ-03 – OBJ-05	O sistema deve implementar as melhores práticas de segurança para assegurar que os serviços de gerenciamento e compartilhamento de dados estejam disponíveis para acesso dos Usuários do sistema.
RNF-07	Complexidade baixa para as aplicações <i>web</i>	– OBJ-03 – OBJ-05	A arquitetura do sistema deve prever que as operações mais complexas relacionadas ao compartilhamento de recursos sejam executadas pelo próprio sistema de forma a oferecer a menor complexidade possível para os Provedores de Serviço.

3.3.7 Matriz de Rastreabilidade

A matriz de rastreabilidade oferece uma visão gráfica do método NTD e permite verificar se todos os objetivos do sistema foram realmente alcançados na especificação de requisitos. A Tabela 10 mostra a matriz de rastreabilidade do sistema.

3.4 Resumo do Capítulo

Este capítulo especificou os requisitos de um Sistema de Gerenciamento e Compartilhamento de Dados (SGCD) para *Web* de acordo com as características desejáveis para este tipo de sistema, levantadas no Capítulo 2.

A técnica escolhida para análise de requisitos foi o método NDT (*Navigational Development Techniques*), pois ele é capaz de capturar requisitos específicos do ambiente *Web*. Primeiramente, definiram-se os objetivos, os requisitos de armazenamento, e os atores do sistema. Depois, definiram-se os requisitos funcionais, interativos e não funcionais. Ao final, a matriz de rastreabilidade do método NTD mostrou as relações de cada requisito com os objetivos do sistema.

Esse conjunto de requisitos servirá de base para o projeto do SGCD proposto neste trabalho. O próximo capítulo introduzirá as tecnologias que serão usadas no sistema proposto e o Capítulo 5 descreverá a arquitetura e o protocolo do sistema propriamente dito.

Tabela 10. Matriz de rastreabilidade

	OBJ-01	OBJ-02	OBJ-03	OBJ-04	OBJ-05
RA-01	X	X			
RA-02	X		X		
RA-03	X		X	X	
RA-04	X				
RA-05			X	X	
RF-01		X		X	
RF-02	X				
RF-03	X				
RF-04	X				
RF-05	X				
RF-06	X				
RF-07	X		X	X	
RF-08	X		X	X	
RF-09			X	X	
RF-10			X	X	
RF-11			X	X	
RF-12			X	X	
RF-13			X	X	
RI-01	X				
RI-02	X				
RI-03	X		X		
RI-04	X		X	X	
RNF-01				X	
RNF-02			X		
RNF-03				X	
RNF-04					X
RNF-05					X
RNF-06	X		X		X
RNF-07			X		X

4 TECNOLOGIAS HABILITADORAS

Conforme mostrado na seção 2.2, um SGCD (Sistema de Gerenciamento e Compartilhamento de Dados) para *Web* requer mecanismos de gerenciamento de identidade e de autorização. As seções 2.2.1 e 2.2.2 discutiram soluções para esses mecanismos e mostraram que este trabalho adota as tecnologias OpenID e OAuth, pois elas são amplamente adotadas na *Web*, simples e interoperáveis com os sistemas atuais.

Este capítulo descreve a arquitetura e o protocolo das tecnologias OpenID e OAuth para que seja possível entender o papel que elas desempenham no SGCD proposto nesta dissertação, que será apresentado no próximo capítulo.

4.1 OpenID

O OpenID é um protocolo de *Single Sign-On* aberto, leve, com base em HTTP e descentralizado, que visa fornecer uma identidade digital única ao **Usuário** de forma que ele possa reutilizar esta identidade para efetuar *login* em diferentes *websites* usando apenas um navegador *web*.

O protocolo OpenID (OPENID FOUNDATION, 2007) é especificado pela OpenID Foundation⁹, que inclui algumas das principais empresas da indústria de identidade digital, como Facebook, Microsoft, Yahoo e Google. Nos últimos anos, o arcabouço OpenID surgiu como uma solução viável para a infraestrutura de identidade centrada no **Usuário** em escala de Internet (RECORDON; REED, 2006).

No OpenID, o identificador de **Usuário** é definido por um URI (*Uniform Resource Identifier*) ou um XRI (*Extensible Resource Identifier*) e deve ser único; por exemplo, o identificador *bob.idprovider.com* define a identidade única controlada por pelo **Usuário** "Bob" em "IDProvider.com".

⁹ The OpenID Foundation website: <http://openid.net/>

4.1.1 Arquitetura

Os principais componentes da arquitetura OpenID são mostrados na Figura 13 e descritos a seguir (OPENID FOUNDATION, 2007).

Figura 13. Arquitetura OpenID

- **UA** (*User-Agent* - Agente do **Usuário**): o navegador *web* do **Usuário** final que implementa HTTP/1.1.
- **OP** (*OpenID Provider* - Provedor de OpenID ou Provedor de Identidade): um servidor de autenticação que fornece asserções possibilitando que um **Usuário** final prove que ele controla um identificador.
- **RP** (*Relying Party* - Parte Confiante ou Provedor de Serviço) uma aplicação *web* que requisita provas de que o **Usuário** final controla um identificador.

O protocolo OpenID não determina o método que o **OP** usará para autenticar o **Usuário**, deixando a cargo do implementador escolher a solução de autenticação mais adequada para garantir a segurança do sistema.

4.1.2 Protocolo

O fluxo de operação para uma autenticação típica com OpenID é mostrado na Figura 14 e descrita a seguir (OPENID FOUNDATION, 2007).

Figura 14. Fluxo de operação OpenID

1. O **Usuário** submete o seu identificador OpenID para o **RP**.
2. O **RP** começa o processo de descoberta do **OP** usando o identificador na procura das informações necessárias para iniciar as solicitações de autenticação. Os métodos utilizados para descoberta podem ser: resolução XRI¹⁰, protocolo Yadis¹¹ ou descoberta com base em HTML (OPENID FOUNDATION, 2007).
3. Depois de ter descoberto o **OP**, o **RP** pode gerar e armazenar um segredo compartilhado com o **OP** (opcional).
4. O **RP** envia uma solicitação de autenticação para o **OP**, redirecionando o navegador *web* do **Usuário** para o URL do provedor.
5. O **Usuário** autentica-se junto ao **OP**.
6. O **OP** redireciona o navegador do **Usuário** para um URL de retorno no *site* **RP**, juntamente com um conjunto de parâmetros que descrevem a asserção feita sobre a identidade do **Usuário** e uma assinatura digital.
7. O **RP**, em seguida, verifica a validade da assinatura usando o segredo compartilhado previamente definido com o **OP** (passo 3), ou por meio da interação direta com o **OP**. Se a verificação for bem sucedida, o **Usuário** tem seu *login* realizado no **RP**.

¹⁰ O protocolo XRI (*eXtensible Resource Identifier*) fornece um padrão de sintaxe e formato para identificadores abstratos e estruturados, que são independentes de domínio, localização, aplicação e transporte. Mais informações em: <http://docs.oasis-open.org/xri/2.0/specs/cd02/xri-resolution-V2.0-cd-02.html>

¹¹ O protocolo Yadis é um protocolo de comunicação para descoberta de serviços. Mais informações em: <http://infogrid.org/trac/wiki/Yadis>

4.2 OAuth

O OAuth é um protocolo de autorização para *Web* que permite que os **Usuários** compartilhem seus recursos protegidos (e.g., fotos, arquivos de texto e contatos) entre diferentes aplicações sem que haja divulgação das credenciais do **Usuário** em cada aplicação.

Por exemplo, um **Usuário** pode permitir que um serviço de impressão acesse suas fotos privadas armazenadas em um serviço de armazenamento de fotos, sem compartilhar o seu nome de **Usuário** e senha com o serviço de impressão. Em vez disso, o **Usuário** autentica-se diretamente no serviço de armazenamento e autoriza que este emita credenciais de delegação para o serviço de impressão. De posse dessas credenciais de delegação, o serviço de impressão acessa as fotos do **Usuário**.

A versão 1.0 do protocolo OAuth foi publicada em dezembro de 2007 e rapidamente tornou-se o padrão da indústria para a delegação de acesso baseado na *Web*. Uma revisão (OAuth 1.0 Revisão A) foi publicada em 2008 para corrigir um problema de segurança e, em abril de 2010, o protocolo OAuth 1.0 foi publicado como RFC 5849 (HAMMER-LAHAV, 2010). Em maio de 2010, o *IETF OAuth Working Group* publicou o primeiro rascunho do protocolo OAuth 2.0 que já se encontra na revisão de número 25 (HAMMER-LAHAV; RECORDON; HARDT, 2012).

O protocolo OAuth 2.0 não é compatível com as versões anteriores, apesar de manter a arquitetura global estabelecida pelas outras versões, e foi criado para solucionar problemas de escalabilidade e complexidade da primeira versão (HAMMER-LAHAV, [S.d.]). Apesar de ainda não ser um padrão oficial, a versão 2.0 já se encontra estável e já foi adotada por grandes nomes da indústria digital como Facebook, Google e Twitter. No decorrer deste trabalho, será utilizada a versão 2.0 do protocolo OAuth (HAMMER-LAHAV; RECORDON; HARDT, 2012), pois ela atende melhor os requisitos do sistema proposto do que a versão 1.0.

4.2.1 Arquitetura

A arquitetura do protocolo OAuth inclui quatro componentes básicos conforme descrição a seguir:

- **Proprietário do Recurso ou Usuário:** uma entidade capaz de permitir o acesso a um recurso protegido. Quando o proprietário do recurso é uma pessoa, ela é referida como **Usuário** final.
- **Cliente:** uma aplicação que faz solicitações de acesso a recursos protegidos em nome do **Proprietário do Recurso** e com a sua autorização.
- **Servidor de Autorização:** o servidor que emite *tokens* de acesso para o **Cliente** após autenticar com sucesso o **Proprietário do Recurso** e obter a sua autorização.
- **Servidor de Recursos:** o servidor que armazena os recursos protegidos, capaz de aceitar e de responder às solicitações de acesso a recursos protegidos associadas a *tokens* de acesso.

O **Servidor de Autorização** e o **Servidor de Recursos** podem ser o mesmo servidor ou entidades separadas. No último caso, um único **Servidor de Autorização** pode emitir *tokens* de acesso aceitos por múltiplos **Servidores de Recursos**.

4.2.2 Escopo de Utilização do Protocolo OAuth

O protocolo OAuth 2.0 fornece uma estrutura bastante flexível, extensível e com muitas funcionalidades de segurança para que o **Usuário** delegue acesso aos seus recursos protegidos. Isso permite que o desenvolvedor escolha diferentes modos de utilização do protocolo para atender aos requisitos de sua aplicação. Por exemplo, podem-se escolher diferentes tipos de Clientes, *tokens* e concessões de autorização.

Os modos de utilização do protocolo OAuth 2.0, de acordo com a especificação (HAMMER-LAHAV; RECORDON; HARDT, 2012), são apresentados a seguir:

- **Tipo de Cliente:** o protocolo OAuth permite três tipos de perfis de Cliente: Aplicação *Web*, Aplicação Baseada em Agente do **Usuário** (por exemplo, o **Usuário** baixa um código que executa no seu navegador *web*) e Aplicação Nativa.
- **Modo de Autorização:** a forma como o Cliente obtém a autorização do **Usuário** pode ser de quatro tipos: Código de Autorização, Implícito, Credenciais do Proprietário de Recursos e Credenciais do Cliente.
- **Tipo de Token:** o protocolo OAuth especifica dois tipos de *tokens*: *Token Bearer* (JONES; HARDT; RECORDON, 2012) e *Token MAC* (HAMMER-LAHAV, 2012). Detalhes sobre esses tipos de *tokens* serão discutidos na seção 5.5.4.

Este capítulo atém-se a descrever os modos de utilização do protocolo OAuth 2.0 que serão usados no sistema proposto (Capítulo 5), que são: Tipo de Cliente “**Aplicação Web**”, Modo de Autorização “**Código de Autorização**” e Token “**Bearer**”. As justificativas para as escolhas dos modos de utilização serão discutidas na seção 5.7.2.

4.2.3 Conceitos

Para entendimento do fluxo do protocolo é necessário entender os seguintes conceitos:

- **Credenciais do Cliente:** é um conjunto de informações, que inclui um identificador e atributos de autenticação, que o **Cliente** usa para se identificar ao interagir com o **Servidor de Autorização**. Essas credenciais são obtidas, geralmente, por meio de um processo de registro junto ao **Servidor de Autorização**.
- **Código de Autorização:** é um código entregue para o **Cliente** quando o **Usuário** final autoriza o acesso a seus recursos protegidos. Este código é emitido pelo **Servidor de Autorização** e entregue ao **Cliente** por meio de redirecionamento do **Agente do Usuário** (e.g., navegador *web*), como será mostrado na descrição do protocolo. Posteriormente, o **Cliente** comunica-se diretamente com o **Servidor de Autorização** e usa este código para obter um *token* de acesso.

- **Token de acesso:** é um *token* emitido pelo **Servidor de Autorização** que representa uma autorização de acesso para um **Cliente** específico e inclui um escopo e um período de duração. Posteriormente, o **Cliente** entrega o *token* de acesso para o **Servidor de Recursos** para conseguir acesso a um recurso ou um conjunto de recursos.

4.2.4 Protocolo

A Figura 15 descreve o fluxo de mensagens do protocolo OAuth segundo o escopo definido na seção 4.2.2.

Figura 15. Fluxo de mensagens do protocolo OAuth (Adaptado de (HAMMER-LAHAV; RECORDON; HARDT, 2012))

1. O **Cliente** inicia uma requisição de acesso a um recurso protegido do **Usuário**, redirecionando o **Agente do Usuário** para o **Servidor de Autorização**, tal como definido na RFC2616 (FIELDING *et al.*, 1999). O **Cliente** inclui na requisição o seu identificador, o escopo solicitado e um URL de redirecionamento para o qual o **Servidor de Autorização** irá enviar o **Agente do Usuário** de volta depois que o acesso for permitido (ou negado).
2. O **Servidor de Autorização** autentica o **Proprietário do Recurso** (por meio do **Agente do Usuário**) e verifica se ele concede ou nega a solicitação de acesso feita pelo **Cliente**.
3. Se o **Proprietário do Recurso** conceder acesso, o **Servidor de Autorização** redireciona o **Agente do Usuário** de volta para o **Cliente**, usando o URL de redirecionamento fornecido anteriormente, incluindo um Código de Autorização no URL.
4. O **Cliente** solicita um *token* de acesso ao **Servidor de Autorização**. Essa requisição inclui as credenciais de **Cliente** para autenticação, o código de autorização recebido na etapa anterior e o URL de redirecionamento usado para obter o Código de Autorização.
5. O **Servidor de Autorização** valida as informações recebidas e, se válidas, responde de volta com um *token* de acesso e, opcionalmente, um *token* de atualização (esse tipo de *token* será discutido na seção 4.2.5).
6. O **Cliente** apresenta o *token* de acesso para o **Servidor de Recursos** e solicita o recurso protegido.
7. O **Servidor de Recursos** valida o *token* de acesso para garantir que ele é válido e que seu escopo abrange o recurso solicitado. Os métodos utilizados pelo **Servidor de Recursos** para validar o *token* de acesso não são definidos pela especificação do protocolo OAuth, mas podem envolver uma interação com o **Servidor de Autorização** (HAMMER-LAHAV; RECORDON; HARDT, 2012).
8. Se o *token* de acesso for válido, o **Servidor de Recursos** entrega o recurso protegido ao **Cliente**.

4.2.5 Token de Atualização

Um *token* de atualização é usado para obter um novo *token* de acesso quando o *token* de acesso atual torna-se inválido ou expira, de forma que não seja necessário iniciar o processo de autorização novamente. Esse tipo de *token* é emitido pelo **Servidor de Autorização** para um **Cliente** e enviado junto com o *token* de acesso.

É importante observar que a emissão de um *token* de atualização é opcional, de acordo com a política do **Servidor de Autorização**. Além disso, ao contrário de *tokens* de acesso, os *tokens* de atualização são usados apenas com **Servidores de Autorização** e nunca são enviados para **Servidores de Recursos**.

A Figura 16 mostra como acontece o fluxo para troca de um *token* de atualização por um *token* de acesso. Os passos da Figura 16 são descritos a seguir.

(Passos 1 e 2) Inicialmente, o **Cliente** apresenta um código de autorização (obtido anteriormente) e solicita um *token* de acesso ao **Servidor de Autorização**. Depois de validar a requisição, o **Servidor de Autorização** entrega um *token* de um acesso e um *token* de atualização ao **Cliente**.

(Passos 3 a 6) Com isso, o **Cliente** solicita um recurso protegido para o **Servidor de Recursos**, apresentando o *token* de acesso. O **Servidor de Recursos** valida o *token* de acesso e, se válido, entrega o recurso. Os passos (3) e (4) repetem-se até que o *token* de acesso expira e o **Cliente** recebe uma mensagem de erro.

(Passos 7 e 8) Então, o **Cliente** apresenta o *token* de atualização ao **Servidor de Autorização** e solicita um novo *token* de acesso. O **Servidor de Autorização** valida a requisição e, se válida, entrega um novo *token* de acesso ao **Cliente** (e, opcionalmente, um novo *token* de atualização). Com isso, o **Cliente** pode voltar a executar o passo 3.

Figura 16. Atualização de um *token* de acesso expirado (Adaptado de (HAMMER-LAHAV; RECORDON; HARDT, 2012))

4.3 Considerações de Segurança

Os protocolos OpenID e OAuth apresentam algumas vulnerabilidades a alguns ataques conhecidos, tais como *Phishing*¹² e *Cross-Site Request Forgery*¹³ (*CSRF*), porém existem medidas que podem ser tomadas para torná-los seguros (TSYRKLEVICH; TSYRKLEVICH, 2007)(DOMINICINI *et al.*, 2010) (LODDERSTEDT; MCGLOIN; HUNT, 2012). As próprias especificações destes protocolos sugerem o uso de conexão segura SSL (*Secure Sockets Layer*) em algumas fases dos protocolos, evitando-se, assim, muitos riscos potenciais. Uma análise detalhada sobre a segurança do SGCD proposto nesta dissertação, que integra os protocolos OpenID e OAuth, será realizada no Capítulo 7.

¹² O ataque de *Phishing* é uma técnica de engenharia social, em que um atacante se disfarça como uma entidade legítima em quem a vítima confia para solicitar que esta revele alguma informação confidencial (THE MITRE CORPORATION, 2012).

¹³ *Cross-Site Request Forgery* (*CSRF*) é um ataque que induz um **Usuário** final a executar ações maliciosas em uma aplicação *web* em que ele está autenticado no momento do ataque (OWASP, 2012).

4.4 *Resumo do Capítulo*

Este capítulo descreveu a arquitetura e o protocolo das tecnologias OpenID e OAuth, que serão usados como base para o SGCD (Sistema de Gerenciamento e Compartilhamento de Dados) para *Web* proposto nesta dissertação. Ambos os protocolos foram desenvolvidos com foco no ambiente *Web* e foram escolhidos por sua ampla adoção, simplicidade e interoperabilidade com os sistemas atuais.

O próximo capítulo descreverá o sistema SGCD proposto e como as tecnologias descritas neste capítulo são usadas para habilitar as funcionalidades desse sistema relativas a gerenciamento de identidades, *Single Sign-On* e autorização.

5 PROPOSTA DA ARQUITETURA DO SISTEMA PDS

No Capítulo 2, foram descritos os conceitos e a arquitetura geral de um SGCD (Sistema de Gerenciamento e Compartilhamento de Dados) para *Web*, de forma independente de tecnologias. Em seguida, no Capítulo 3 foram especificados os requisitos que um SGCD genérico deve atender para permitir que um **Usuário** gerencie e compartilhe seus recursos na *Web*. Com base nessa especificação de requisitos, este capítulo propõe a arquitetura e o protocolo de um SGCD para *Web*, denominado **Sistema PDS (*Personal Data Sharing*)**.

O **Sistema PDS** utiliza os protocolos OpenID e OAuth, conforme descritos no Capítulo 4, para autenticação do **Usuário** e para autorização de acesso aos recursos protegidos do **Usuário**, respectivamente. Esse sistema possibilita que aplicações *web* acessem os recursos do **Usuário**, após a sua autorização, permitindo que tal **Usuário** defina configurações de compartilhamento que melhor se ajustem aos seus interesses. Além disso, o sistema permite que o **Usuário** gerencie seus recursos na *Web* de forma centralizada.

Inicialmente, este capítulo apresenta uma descrição geral do **Sistema PDS** e discute como o problema de gerenciamento e compartilhamento de dados é tratado no sistema proposto. Depois, são descritos a arquitetura, os componentes e o protocolo da solução. Por fim, é feita uma análise da arquitetura e do protocolo do **Sistema PDS**.

5.1 Requisitos

O **Sistema PDS** (*Personal Data Sharing*) deve atender a todos os requisitos levantados na Especificação de Requisitos do Capítulo 3, relacionados a seguir:

- **Requisitos de Armazenamento:** RA-01 (*Conta do Usuário*), RA-02 (*Recursos do Usuário*), RA-03 (*Autorizações*), RA-04 (*Histórico*) e RA-05 (*Sites registrados*).
- **Requisitos Funcionais:** RF-01 (*Autenticar Usuário*), RF-02 (*Consultar recursos*), RF-03 (*Criar recursos*), RF-04 (*Remover recursos*), RF-05 (*Consultar histórico*), RF-

06 (*Remover atividades*), RF-07 (*Consultar autorizações*), RF-08 (*Remover autorizações*), RF-09 (*Requisitar autorização*), RF-10 (*Ler recurso*), RF-11 (*Escrever recurso*), RF-12 (*Autenticar Provedor de Serviço*) e RF-13 (*Registrar Provedor de Serviço*).

- **Requisitos Interativos:** RI-01 (*Informação sobre os recursos do Usuário*), RI-02 (*Informação sobre as atividades*), RI-03 (*Informação sobre as autorizações do Usuário*) e RI-04 (*Informação sobre uma requisição de autorização*).
- **Requisitos Não Funcionais:** RNF-01 (*Confidencialidade e integridade*), RNF-02 (*Compartilhamento de tipos de dados genéricos*), RNF-03 (*Autorização explícita do Usuário*), RNF-04 (*Uso de navegador web como agente do Usuário*), RNF-05 (*Uso de padrões abertos e amplamente adotados*), RNF-06 (*Disponibilidade*) e RNF-07 (*Complexidade baixa para as aplicações web*).

5.2 Visão Geral do Sistema PDS

As funcionalidades que o **Sistema PDS** oferece ao **Usuário** podem ser separadas em dois grupos principais: **gerenciamento de recursos na Web** e **compartilhamento de recursos com aplicações web**. No primeiro grupo, o **Usuário** acessa um *site* do portal de gerenciamento da sua conta PDS. No segundo grupo, o **Usuário** está originalmente em um *site* de um Provedor de Serviço, que o redireciona para a interface de compartilhamento do serviço PDS para requisitar acesso (de leitura ou escrita) aos seus recursos protegidos.

Esta seção mostra uma visão geral da interface do **Sistema PDS** considerando o ponto de vista do **Usuário**. O objetivo é facilitar o entendimento sobre o modo de operação do sistema ao mostrar um conjunto de telas simplificado que exemplifica como o **Usuário** interage com esse sistema quando quer gerenciar ou compartilhar seus recursos. Essa visão inicial será complementada no Capítulo 8, que mostrará com detalhes, por meio da descrição das funcionalidades de um protótipo desse sistema, a sua interface com o **Usuário**.

5.2.1 Interface de Gerenciamento

A interface de gerenciamento do **Sistema PDS** é representada por um *site* ou portal onde o **Usuário** pode gerenciar seus **recursos**, **autorizações** e **atividades**. A Figura 17 mostra um exemplo de como seria a página inicial desse portal de gerenciamento. Na função de **gerenciamento de recursos**, o **Usuário** pode visualizar, adicionar, remover ou alterar itens do seu conjunto de recursos. Na função de **gerenciamento de autorizações**, o **Usuário** pode administrar suas configurações de compartilhamento, ou seja, visualizar quais autorizações de compartilhamento foram concedidas e revogá-las, se necessário. Na função de **gerenciamento de atividades** o **Usuário** pode acompanhar o histórico de atividades realizadas na sua conta PDS como, por exemplo, criação de recursos, remoção de recursos e criação de autorizações de compartilhamento.

Figura 17. Portal de gerenciamento do Sistema PDS

5.2.2 Interface de Compartilhamento

O protocolo de compartilhamento do **Sistema PDS** trabalha com redirecionamentos HTTP de forma que o navegador *web* do **Usuário** é redirecionado do *site* de um Provedor de Serviço para o *site* do serviço PDS para autorizar uma requisição de acesso (de leitura ou escrita) à sua conta PDS. Esta seção apresenta dois exemplos de compartilhamento: um acesso de escrita e um acesso de leitura.

No exemplo da Figura 18, o **Usuário** acessa o *site* X, que representa um serviço de edição de fotos, no seu navegador *web*. Nesse *site*, o **Usuário** edita um álbum de fotos da sua

viagem de férias (“Fotos_Italia”) e, ao final, escolhe salvar esse álbum na sua conta PDS. Com isso, o *site X* redireciona o navegador *web* do **Usuário** para o *site* do serviço PDS com uma requisição de acesso de escrita para recursos do tipo “Álbum de Fotos”. Se o **Usuário** aprovar a requisição, o álbum de fotos é salvo pelo *site X* na conta PDS do **Usuário**. Por fim, o navegador do **Usuário** é redirecionado de volta para o *site X*.

Figura 18. Armazenando recursos do Usuário

A Figura 19 mostra um exemplo em que um *site Y* recupera um recurso do **Usuário**. Neste exemplo, o **Usuário** acessa o *site Y*, que oferece as opções de criar um novo álbum de fotos manualmente ou buscar um álbum existente na sua conta PDS. O **Usuário**, então, opta por buscar o recurso na sua conta PDS e é redirecionado para o *site* do serviço PDS com uma requisição de acesso de leitura para recursos do tipo “Álbum de Fotos”. No *site* do PDS, o **Usuário** visualiza uma lista de todos os álbuns que possui na sua conta e seleciona o álbum de fotos criado pelo *site X* no exemplo anterior (“Fotos_Italia”) para compartilhar com o *site Y*. Quando o **Usuário** submete a aprovação da requisição, o álbum de fotos é entregue ao *site Y* e o navegador do **Usuário** é redirecionado de volta para o *site Y*.

Figura 19. Recuperando recursos do Usuário

É importante destacar que o *site Y* tem acesso aos recursos do **Usuário** previamente criados pelo *site X*, com o consentimento do **Usuário**, mas nenhum dos dois *sites* tem acesso às credenciais do **Usuário** no **Sistema PDS**. As próximas seções explicarão com mais detalhes como acontecem os fluxos de leitura e escrita no **Sistema PDS**.

5.3 Arquitetura do Sistema PDS

Os componentes da arquitetura do **Sistema PDS** (*Personal Data Sharing*) são mostrados na Figura 20 e descritos a seguir.

Figura 20. Arquitetura do Sistema PDS

- **Usuário:** representa o proprietário dos recursos que acessa o sistema por meio de um navegador *web* padrão.
- **Provedor de Identidade:** é o serviço que fornece a infra-estrutura de segurança necessária para autenticação, gerenciamento de identidade e *Single Sign-On*. A tecnologia utilizada para compor o **Provedor de Identidade** é o protocolo OpenID (discutido no Capítulo 4). Dessa forma, o **Usuário** deve possuir uma conta em um **Provedor de Identidade** que suporte OpenID e pode escolher o provedor que melhor atender às suas necessidades. Uma vez que o **Usuário** fez o *login* no seu **Provedor de Identidade**, o **Usuário** pode acessar automaticamente os serviços dos outros componentes da arquitetura, **Servidor PDS** e **Provedores de Serviço**, sem precisar fazer *login* novamente ou criar uma nova conta, pois esses componentes funcionam como clientes (*Relying Parties*) no protocolo OpenID.
- **Provedor de Serviço:** é uma aplicação *web* que faz requisições para obter a autorização do **Usuário** para o acesso a seus recursos protegidos. Essa aplicação implementa o componente *Relying Party* do protocolo OpenID e o componente Cliente do protocolo OAuth, explicados no Capítulo 4.

- **Servidor PDS:** é o componente em que estão concentradas as principais funcionalidades para gerenciamento de recursos e autorizações do **Usuário**. Por ser o componente mais complexo da arquitetura PDS, a seção 5.4, a seguir, descreverá esse componente individualmente.

5.4 Arquitetura do Servidor PDS

Esta seção detalha a arquitetura do **Servidor PDS**, que emprega os protocolos OpenID e OAuth, explicados no Capítulo 4. Nesta arquitetura, o **Servidor de Autorização** e o **Servidor de Recursos**, do protocolo OAuth, são combinados na mesma entidade de forma a obter centralização tanto do gerenciamento dos recursos protegidos quanto do gerenciamento de autorização. Além disso, o **Servidor PDS** implementa o componente *Relying Party* do protocolo OpenID.

Os componentes da arquitetura do **Servidor PDS** são detalhados na Figura 21 e descritos a seguir. O componente **Provedor de Identidade** (vide Figura 20) foi omitido da figura para efeito de simplificação.

Figura 21. Arquitetura do Servidor PDS

- **Interface de Serviço:** é a interface com os Provedores de Serviço para compartilhamento de dados que usa métodos HTTP padrão para enviar requisições e respostas utilizando uma **API**. É também por meio dessa interface que ocorre o **Registro** dos **Provedores de Serviço**.

- **API:** é um conjunto de funções de acesso usadas pelos **Provedores de Serviço** para ler e escrever um determinado recurso ou um conjunto de recursos da conta de um **Usuário**.
- **Registro:** é um *site* em que os **Provedores de Serviço** se registram por meio de um formulário e recebem suas credenciais que serão usadas posteriormente para autenticá-los junto ao **Servidor PDS** conforme exigido pelo protocolo OAuth (seção 4.2.3).
- **Relying Party OpenID:** é o componente cliente da arquitetura OpenID que se comunica com o **Provedor de Identidade** para comprovar a identidade do **Usuário**.
- **Servidor de Autorização OAuth:** é o componente da arquitetura OAuth responsável por emitir *tokens* de acesso para o **Provedor de Serviço** após autenticar o **Usuário** e obter a sua autorização.
- **Servidor de Recursos OAuth:** é o componente da arquitetura OAuth responsável por proteger os recursos do **Usuário** e responder às solicitações de acesso a esses recursos usando *tokens* de acesso emitidos pelo **Servidor de Autorização**.
- **Repositório:** é o componente de banco de dados que armazena informações a respeito das contas dos **Usuários**, dos recursos do **Usuário** e dos metadados associados a eles, dos serviços registrados, das políticas e *tokens* associados a cada recurso e dos *logs* das atividades.
- **Camada de Acesso e Apresentação:** é responsável por recuperar dados do banco de dados e apresentar ao **Usuário** por meio do Portal. Inclui também os mecanismos de navegação e busca de conteúdo específico dentro do conjunto de recursos pessoais do **Usuário**. A busca é facilitada pela inclusão de *tags* associados pelo **Usuário** aos seus recursos.
- **Portal:** é um *website* que funciona como a camada de apresentação do serviço PDS para o **Usuário**, onde ele pode gerenciar de forma centralizada o seu repositório pessoal de recursos. Inclui as seguintes funcionalidades: administrar preferências de compartilhamento (i.e., adicionar, remover e alterar autorizações de compartilhamento), acompanhar histórico de atividades (e.g., lista de *websites* com quem compartilhou dados) e manipular (i.e., adicionar, remover e alterar) um determinado recurso ou um conjunto de recursos.

5.4.1 Módulos de Extensão

Os componentes mostrados na Figura 21 compõem a arquitetura básica do **Servidor PDS** que permite o gerenciamento e compartilhamento de recursos do **Usuário**. Esta arquitetura básica pode ser estendida por meio de módulos de extensão que, apesar de não estarem diretamente ligados às funcionalidades principais do **Sistema PDS** podem executar funcionalidades importantes para os **Usuários** e **Provedores de Serviço**. Os módulos de extensão do **Servidor PDS** são *Proxies*, *Renderizadores* e *Indexador*, conforme descrito a seguir.

5.4.1.1 Proxies

Para facilitar que novos **Usuários** comecem a utilizar o **Sistema PDS**, é necessário criar mecanismos de integração com as soluções de compartilhamento existentes. Com esse intuito, devem ser construídos componentes de *software*, chamados **Proxies**, que traduzem uma API de compartilhamento de um serviço específico para as funções da API padrão do **Servidor PDS**, tornando o serviço compatível com o **Sistema PDS**. Assim, o **Usuário** pode importar seus dados de outros serviços para a base de dados no **Servidor PDS**.

Para cada novo serviço adicionado, um novo **Proxy** deve ser implementado, pois cada um desses serviços possui uma API de compartilhamento diferente. Exemplos de serviços que seriam bons candidatos para implementação de **Proxies** são: Gmail, Yahoo Mail, Facebook, Twitter, e Google Docs.

5.4.1.2 Renderizadores

Na arquitetura básica do **Servidor PDS**, o **Usuário** não visualiza o recurso propriamente dito, mas apenas informações sobre os recursos (por exemplo, tamanho, data de criação e descrição) e tem a opção de fazer *download* do recurso na forma de um arquivo. Isso acontece porque o **Servidor PDS** suporta o compartilhamento e armazenamento de quaisquer tipos de recursos (por exemplo, vídeos, arquivos executáveis, XMLs e fotos), mas não é possível suportar a visualização no navegador *web* do **Usuário** de tipos de dados genéricos. O módulo de extensão de **Renderizadores** visa suprir essa

necessidade e permitir que o **Usuário** visualize tipos de recursos pré-definidos na interface de gerenciamento.

Cada **Renderizador** contém a representação interna de um tipo de recurso específico e da forma como esse recurso será apresentado ao **Usuário**. Caso seja importante mostrar para o **Usuário** a visualização gráfica de um tipo de recurso específico e não existir um **Renderizador** para este tipo de recurso, deve-se implementar um novo **Renderizador**.

5.4.1.3 Indexador

É um módulo que adiciona flexibilidade à arquitetura do **Sistema PDS**, pois permite que os recursos do **Usuário** sejam armazenados de forma distribuída em outros servidores. Ao invés de salvar um recurso no **Servidor PDS**, como normalmente é feito, o **Usuário** concede um *token* de acesso permitindo que o **Servidor PDS** acesse seus recursos em outro serviço. Dessa forma, o **Servidor PDS** passa a atuar como **Cliente** na arquitetura OAuth (conforme explicado na seção 4.2). A vantagem dessa abordagem é que o **Usuário** pode manter o armazenamento distribuído, mas o gerenciamento centralizado. É importante destacar que para permitir o funcionamento desse módulo, o *site* que armazena efetivamente o recurso precisa implementar os componentes **Servidor de Autorização** e **Servidor de Recursos** do protocolo OAuth.

Por exemplo, imagine que um **Usuário** tenha fotos armazenadas em um *Site A*, mas não deseja transferir fisicamente esses arquivos para o **Servidor PDS**. Entretanto, esse **Usuário** quer que o gerenciamento e compartilhamento de recursos aconteçam no **Servidor PDS**. Usando o módulo Indexador, o **Usuário** pode entregar um *token* de acesso para permitir que o **Servidor PDS** acesse o seu conjunto de recursos no *Site A*. Assim, o **Servidor PDS** será responsável pelo gerenciamento e compartilhamento tanto dos recursos que estão armazenados em seu próprio serviço, quanto dos recursos acessíveis por esse *token* de acesso que estão armazenados no *Site A*. Caso uma aplicação *web* solicite ao **Servidor PDS** um recurso do usuário que está armazenado no *Site A* e é gerenciado pelo **Servidor PDS**, este atuará como intermediário na transferência desse recurso.

5.5 Estruturas de dados

O compartilhamento e gerenciamento de recursos do **Usuário** envolvem a manipulação de diversas estruturas de dados, referentes a recursos, escopo e *tokens*. Esta seção descreve como essas estruturas de dados são representadas pelo **Sistema PDS**.

5.5.1 Representação de Tipos de Recursos

Para atender os requisitos previamente especificados, o sistema deve suportar o **compartilhamento de tipos de recursos genéricos**, ou seja, qualquer tipo de dado (por exemplo, imagens, dados cifrados, lista de contatos e formatos binários). Internamente, o **Servidor PDS** trata os recursos do **Usuário** como arquivos sem se preocupar com o conteúdo. Para facilitar o compartilhamento, os tipos dos recursos são identificados univocamente por meio de URLs (*Uniform Resource Locator*) de forma que a requisição de acesso a recursos do **Usuário** inclui como parâmetro o tipo de recurso a ser compartilhado.

Por exemplo, um **Servidor PDS** poderia definir tipos de recursos próprios para os tipos mais comuns, tais como “http://pds.larc.usp.br/image” e “http://pds.larc.usp.br/video”, e, também, aceitar que um **Provedor de Serviço** crie seu próprio tipo de dados que também poderá ser utilizado por outras aplicações *web*, tais como “http://www.games.com/playerProfile” e “http://www.gourmet.com/recipe”.

A identificação por URLs facilita o compartilhamento de recursos, pois permite que uma aplicação *web*, ou um consórcio de aplicações, defina os seus próprios tipos de recursos com acordos sobre a sintaxe e a semântica associada aos valores que cada tipo de recurso pode receber.

5.5.2 Representação do Escopo da Requisição de Acesso

O escopo determina o modo de acesso (leitura ou escrita) e os tipos de recursos requisitados pelo **Provedor de Serviço**. O valor do parâmetro escopo é representado por uma lista de *Strings* delimitadas por espaços, em que cada *String* é um URL que contém o tipo de recurso seguido pelo modo de acesso.

O modo de acesso pode receber dois valores: “*pull*” para um acesso de leitura ou “*push*” para um acesso de escrita. As palavras *pull* e *push* são palavras reservadas quando usadas no final de um URL. O sistema suporta apenas os modos de acesso de leitura e escrita, mas, posteriormente, outros modos de acesso (e.g., remoção e sobrescrita) poderão ser adicionados.

Por exemplo, o escopo “http://pds.larc.usp.br/image/pull http://pds.larc.usp.br/video/pull http://pds.larc.usp.br/posts/pull” representa um escopo de leitura para os tipos de recursos “http://pds.larc.usp.br/image”, “http://pds.larc.usp.br/video” e “http://pds.larc.usp.br/posts”.

5.5.3 Lista de Recursos

Quando o **Servidor PDS** recebe um *token* para acesso de leitura, ele entrega de volta para o **Provedor de Serviço** uma lista dos recursos do **Usuário** acessíveis por meio daquele *token*. Para representação dessa lista, o **Sistema PDS** usa o formato JSON (*JavaScript Object Notation*) (<http://www.json.org>), pois é um formato de texto para troca de dados que é fácil tanto para um ser humano ler, quanto para máquinas analisarem e gerarem.

A Figura 22 mostra um exemplo de uma lista de recursos em que cada recurso possui um URL onde o **Provedor de Serviço** pode fazer o *download* do arquivo relativo ao recurso. Nesse exemplo, o **Servidor PDS** respondeu a requisição do **Provedor de Serviço** entregando uma lista que contém três recursos do tipo “http://pds.larc.usp.br/image”, um recurso do tipo “http://pds.larc.usp.br/video” e nenhum recurso do tipo

“http://pds.larc.usp.br/posts”. Posteriormente, o **Provedor de Serviço** pode, após a verificação do token, acessar os URLs individualmente e fazer *download* dos recursos propriamente ditos.

```
{
  "resources": [
    {
      "value": "http://pds.larc.usp.br/images",
      "attributes": [
        {
          "url": "http://pds.larc.usp.br/server/deliver_resource.do?rcode=u003d0febefa8367ec86401367ed274bc000c"
        },
        {
          "url": "http://pds.larc.usp.br/server/deliver_resource.do?rcode=u003d402881e635a799470135a7998f420002"
        },
        {
          "url": "http://pds.larc.usp.br/server/deliver_resource.do?rcode=u003d402881e635a56ed10135a572d5940009"
        }
      ]
    },
    {
      "value": "http://pds.larc.usp.br/video",
      "attributes": [
        {
          "url": "http://pds.larc.usp.br/server/deliver_resource.do?rcode=u003d0febefa8367ec86401367ed275a7000d"
        }
      ]
    },
    {
      "value": "http://pds.larc.usp.br/posts",
      "attributes": [ ]
    }
  ]
}
```

Figura 22. Lista de recursos em formato JSON (*JavaScript Object Notation*)

5.5.4 Token

O protocolo OAuth faz uso extensivo de *tokens* (*tokens* de acesso, *tokens* de atualização e códigos de autorização). O conteúdo de informação de um *token* pode ser representado de duas maneiras (LODDERSTEDT; MCGLOIN; HUNT, 2012):

- **Handle (ou artefato):** é apenas uma referência a alguma estrutura de dados interna dentro do **Servidor de Autorização**; a estrutura de dados interna contém os atributos do *token*, como ID do **Usuário** e escopo. Esse tipo de *token* permite revogação simples e não necessita de mecanismos de criptografia para proteger o conteúdo contra modificação, pois o **Provedor de Serviço** recebe apenas um identificador que referencia as informações de autorização armazenadas no **Servidor de Autorização**. Por outro lado, o uso de *handles* exige comunicação entre as entidades emissora e consumidora a fim de validar o *token*. Dessa forma, *handles* são normalmente usados se as entidades emissora e consumidora do *token* são a mesma entidade. Além disso, a transmissão do *token*

do **Servidor de Autorização** para o **Provedor de Serviço** deve ser protegida contra acesso não autorizado.

- **Asserção (ou *token* auto-suficiente)**: é um *token* verificável, que normalmente tem uma duração, um escopo, um destinatário, e é assinado digitalmente contendo informações sobre o **Usuário** e o cliente (por exemplo, asserções SAML e tíquetes Kerberos). Asserções normalmente podem ser diretamente validadas e usadas por um **Servidor de Recursos**, sem interações com o **Servidor de Autorização**. Isso resulta em melhor desempenho e escalabilidade quando as entidades emissora e consumidora residem em sistemas diferentes. Por outro lado, a implementação de revogação é mais difícil.

Como discutido na seção 4.2.2, o protocolo OAuth especifica dois tipos de *tokens*: *Token Bearer* (JONES; HARDT; RECORDON, 2012) e *Token MAC* (HAMMER-LAHAV, 2012). O *Token Bearer* é um exemplo de *Handle*, em que o **Provedor de Serviço** que possui o *token* pode usá-lo sem a necessidade de provar que possui uma chave criptográfica. Já o *Token MAC* é um exemplo de Asserção, em que o **Provedor de Serviço** tem que assinar digitalmente a requisição de acesso com um segredo correspondente ao *token* enviado junto com a requisição.

O **Sistema PDS** usa o *Token Bearer*, pois, como as entidades **Servidores de Autorização** e **de Recursos** estão contidos no componente **Servidor PDS**, o tipo *Handle* é mais adequado. Além disso, esse tipo é o que oferece a menor complexidade no lado do **Provedor de Serviço**, pois ele não precisa realizar nenhuma operação criptográfica para provar que é o proprietário do *token*. O formato do *token* utilizado é definido em uma especificação auxiliar do protocolo OAuth (JONES; HARDT; RECORDON, 2012). Adicionalmente, o sistema aproveita-se do fato de que o **Provedor de Serviço** está em uma sessão autenticada no momento da entrega do *token* para comprovar que ele é, de fato, o proprietário do *token*.

A Figura 23.a mostra a representação do *token*, que é armazenado no **Servidor PDS** para cada autorização de acesso concedida por um **Usuário** a um **Provedor de Serviço**, e a Figura 23.b mostra um exemplo de *token*. Nesse exemplo, o **Provedor de Serviço** “http://example.com” obteve autorização para escrever recursos do tipo “http://pds.larc.usp.br/image” na conta do **Usuário** “http://idprovider.com/alice” com

expiração em “2012-03-31 12:00:00”. A Figura 24 mostra o *handle* correspondente recebido pelo **Provedor de Serviço** para esse mesmo exemplo. Essas figuras mostram que o **Provedor de Serviço** recebe apenas uma referência (representada pela *String access_token*) para apresentar no momento em que requisita um recurso, enquanto o restante das informações fica guardada em uma estrutura de dados interna do **Servidor PDS**.

Figura 23. Token do Sistema PDS

Figura 24. Exemplo de Handle

5.6 Protocolo de Comunicação do Sistema PDS

Esta seção apresenta o **protocolo** do **Sistema PDS**. Inicialmente, descrevem-se as condições que devem ser atendidas para que o protocolo possa ser executado e a autenticação OpenID. Depois, para um melhor entendimento do protocolo, apresenta-se uma visão geral e simplificada da seqüência de mensagens trocadas em um fluxo completo de compartilhamento de recursos (Figura 26). Em seguida, de maneira mais detalhada, apresentam-se as mensagens do protocolo (Tabela 13 e Tabela 14) para os dois modos de acesso (leitura e escrita).

5.6.1 Condições para Execução do Protocolo

Existem duas condições que devem ser atendidas previamente para execução do fluxo do protocolo do **Sistema PDS**:

- 1) O **Usuário** deve possuir uma conta em um **Provedor de Identidade** com suporte a OpenID. Esta condição é decorrente do uso do protocolo OpenID.
- 2) O **Provedor de Serviço** deve se registrar previamente no **Servidor PDS** para obter as credenciais de autenticação, que contêm um identificador da aplicação e uma chave secreta. Esta condição é decorrente do uso do protocolo OAuth.

5.6.2 Autenticação OpenID e Single Sign-On

Na arquitetura descrita na seção 5.3, tanto o **Provedor de Serviço** como o **Servidor PDS**, não implementam seus próprios mecanismos de autenticação, ao invés disso, eles implementam a parte cliente do protocolo OpenID, chamada **Relying Party** (RP). Como discutido na seção 2.3, a vantagem dessa abordagem é que o **Usuário** não precisa criar novas contas em cada *website* e, uma vez feito o *login* no **Provedor de Identidade**, o **Usuário** terá seu *login* nos outros *websites* realizado automaticamente. Assim como o protocolo OpenID, o **Sistema PDS** também não determina os métodos de autenticação a serem utilizados entre o **Usuário** e o seu **Provedor de Identidade**. Portanto, cabe ao

Provedor de Identidade a escolha de um método de autenticação apropriado às suas políticas de segurança. Um conjunto de recomendações e melhores práticas sobre o assunto pode ser encontrado em (BURR; DODSON; POLK, 2006).

A Figura 25 mostra o fluxo do protocolo OpenID, que foi anteriormente mostrado na seção 4.1.2, com o objetivo de facilitar o entendimento do funcionamento do protocolo no **Sistema PDS**. Nesta figura, o papel do **RP** (*Relying Party*) do protocolo OpenID pode ser desempenhado tanto pelo **Servidor PDS** quanto pelo **Provedor de Serviço** no momento em que for necessário comprovar a identidade do **Usuário**. No primeiro passo, o **Usuário** submete o seu identificador OpenID para o **RP**, que, então, começa a descoberta do **Provedor de Identidade** do **Usuário** usando resolução XRI, protocolo Yadis ou descoberta com base em HTML. Depois de ter descoberto o **Provedor de Identidade**, o **RP** pode, opcionalmente, compartilhar um segredo com ele para validação de mensagens. Em seguida, o **RP** envia um pedido de autenticação, redirecionando o navegador do **Usuário** para o URL do **Provedor de Identidade**. Este, então, autentica o **Usuário** e, uma vez que a autenticação é concluída, redireciona o navegador web do **Usuário** para o URL de retorno do *website* do **RP**, juntamente com alguns parâmetros descrevendo a asserção feita sobre a identidade do **Usuário** e uma assinatura digital. Por fim, o **RP** verifica a validade da assinatura usando o segredo compartilhado anteriormente definido com o **Provedor de Identidade** (passo 3), ou pela interação direta com o **Provedor de Identidade** (passo 7).

Figura 25. Fluxo de Autenticação OpenID

Na primeira vez em que o **Usuário** participar do fluxo de autenticação do OpenID, toda a sequência de passos descrita anteriormente será executada. Uma vez que o **Usuário** já tenha feito *login* no seu **Provedor de Identidade** (Passo 5), não será mais necessário executar este passo enquanto a sessão do **Usuário** for válida. Assim, quando o **Usuário** acessar outros *websites*, as demais etapas ficam transparentes para ele, que não precisará efetuar o *login* separadamente para cada *website* de acordo com a funcionalidade de *Single Sign-On* do OpenID.

5.6.3 Fluxo Simplificado do Protocolo de Compartilhamento

A Figura 26, a seguir, apresenta uma visão geral do protocolo de compartilhamento de recursos por meio de um fluxo simplificado. Este fluxo será detalhado na próxima seção.

Figura 26. Fluxo de mensagens para compartilhamento de dados

O fluxo simplificado é explicado a seguir, de acordo com os números das mensagens na Figura 26:

1. Se o **Usuário** não tiver feito *login* no **Provedor de Serviço**, segue-se o fluxo de autenticação do OpenID descrito na Seção 5.6.2, envolvendo o **Usuário**, o **Provedor de Serviço** e o **Provedor de Identidade**.
2. No segundo passo, o **Usuário** envia o endereço do seu **Servidor PDS** para o **Provedor de Serviço**.

3. O **Provedor de Serviço** envia uma requisição de autorização de acesso aos recursos do **Usuário**, redirecionando o navegador *web* do **Usuário** para o URL do **Servidor PDS**, juntamente com o seu identificador de cliente, o escopo solicitado, e um URL de redirecionamento para o qual o **Servidor PDS** irá enviar o **Usuário** de volta depois que o acesso for permitido (ou negado).
4. Neste ponto, se o **Usuário** não tiver feito *login* no **Servidor PDS**, segue-se o fluxo de autenticação do OpenID, mas, de forma transparente para o **Usuário**, pois ele já fez o *login* previamente no **Provedor de Identidade**.
5. O **Servidor PDS** informa , então, ao **Usuário** sobre a lista de recursos e o escopo solicitados pelo **Provedor de Serviço**.
6. O **Usuário** responde se permite ou nega o acesso. Caso o **Usuário** decida permitir o acesso, ele define, também, as políticas de compartilhamento.
7. Se o **Usuário** conceder o acesso, o **Servidor PDS** redireciona o navegador *web* do **Usuário** de volta para o **Provedor de Serviço** usando o URL de redirecionamento fornecido anteriormente, incluindo um código de autorização (conforme descrito na seção 4.2.3).
8. O **Provedor de Serviço**, por sua vez, requisita um *token* de acesso ao **Servidor PDS**, e autentica-se usando suas credenciais de autenticação, o código de autorização recebido e o URL de redirecionamento usado para obter o Código de Autorização.
9. O **Servidor PDS** valida as informações recebidas e, se válidas, responde de volta com um *token* de acesso (conforme descrito na seção 4.2.3).
10. O **Provedor de Serviço** apresenta o *token* de acesso e solicita o recurso protegido para o **Servidor PDS**.
11. Por fim, o **Servidor PDS** valida o *token* de acesso e, se válido, entrega o recurso protegido do **Usuário** ao **Provedor de Serviço**.

5.6.4 Mensagens do Protocolo de Compartilhamento

A seção anterior descreveu, de maneira simplificada, o protocolo de compartilhamento de recursos do **Sistema PDS**. Como base nessa descrição, esta seção detalha as mensagens do protocolo e especifica os seus parâmetros. Inicialmente, são apresentadas

as notações utilizadas para descrição do protocolo. Em seguida, é apresentado o fluxo de mensagens do protocolo para um acesso de leitura. Por fim, é apresentado o fluxo de mensagens do protocolo para um acesso de escrita.

5.6.4.1 Notações

A Tabela 11, a seguir, apresenta as notações utilizadas na descrição das entidades do **Sistema PDS**.

Tabela 11. Notações para descrição das entidades do sistema

Notação	Significado
U	Entidade Agente do Usuário (navegador <i>web</i>)
S	Entidade Provedor de Serviço
PDS	Entidade Servidor PDS
IdP	Entidade Provedor de Identidade

A Tabela 12, a seguir, apresenta as notações utilizadas na descrição dos parâmetros das mensagens do protocolo. Esses parâmetros serão usados posteriormente nas seções 5.6.4.2 e 5.6.4.3 para descrição das mensagens do protocolo.

Tabela 12. Notações para descrição dos parâmetros das mensagens do protocolo

Parâmetro	Significado
<i>PDS_url</i>	Endereço do Servidor PDS . Exemplo: “https://pds.larc.usp.br”
<i>response_type</i>	Modo de autorização do protocolo OAuth. Conforme explicado na seção 4.2.2, o Sistema PDS utiliza o modo “Código de autorização”. Logo, o valor do parâmetro “response_type” deve ser “code” em concordância com a especificação do protocolo OAuth (HAMMER-LAHAV; RECORDON; HARDT, 2012).
<i>client_id</i>	Identificador único gerado pelo Servidor PDS para um Provedor de Serviço em um processo de registro prévio. Exemplo: “s6BhdRkqt3”
<i>scope</i>	O escopo da requisição de acesso conforme explicado na seção 5.5.2. Exemplo: “http://pds.larc.usp.br/image/pull http://pds.larc.usp.br/audio/pull http://pds.larc.usp.br/video/pull”
<i>redirect_url</i>	Endereço do Provedor de Serviço para o qual o Servidor PDS deve redirecionar o Usuário em resposta a uma requisição de autorização. Exemplo: “https://client.example.com/redirect”

<i>resource_list</i>	Lista dos recursos que o Usuário possui referentes aos tipos requisitados pelo Provedor de Serviço . Este parâmetro é usado apenas em acessos de leitura. Exemplo: Para o escopo " http://pds.larc.usp.br/image/pull " " http://pds.larc.usp.br/audio/pull ", essa lista poderia ser {foto1.jpg, foto2.jpg, foto3.jpg, foto4.jpg, song1.mp3, song2.mp3, song3.mp3}
<i>approved_resource_list</i>	Lista de recursos que o Usuário autorizou compartilhar com o Provedor de Serviço . É um subconjunto do parâmetro <i>resource_list</i> e é usado apenas em acessos de leitura. Exemplo: {foto1.jpg, foto3.jpg, song3.mp3}
<i>approved_scope</i>	Escopo aprovado pelo Usuário para uma requisição do Provedor de Serviço . É um subconjunto do parâmetro <i>scope</i> . Este parâmetro é usado apenas em acessos de escrita. Exemplo: " http://pds.larc.usp.br/image/push " " http://pds.larc.usp.br/video/push "
<i>expiration_date</i>	Data de expiração para a autorização de acesso concedida pelo Usuário ao Provedor de Serviço . Exemplo: "2012-03-07 12:26:26 GMT"
<i>code</i>	Código de autorização gerado pelo Servidor PDS quando o Usuário autoriza uma requisição do Provedor de Serviço . Este código tem prazo de expiração curto e só pode ser usado uma única vez. Exemplo: "Splxl0BeZQQYbYS6WxSbIA"
<i>grant_type</i>	Modo de autorização do protocolo OAuth. Conforme explicado na seção 4.2.2, o Sistema PDS utiliza o modo "Código de autorização". Logo, o valor do parâmetro " <i>grant_type</i> " deve ser "authorization_code" em concordância com a especificação do protocolo OAuth (HAMMER-LAHAV; RECORDON; HARDT, 2012).
<i>client_secret</i>	Chave gerada pelo Servidor PDS e entregue ao Provedor de Serviço em um processo de registro prévio. Exemplo: "7Fjfp0ZBr1KtDRbnfVdmIw"
<i>token</i>	<i>Token</i> emitido pelo Servidor PDS para o Provedor de Serviço a partir de um código de autorização. Exemplo: { "access_token": "2YotnFZFEjr1zCsicMWpAA", "token_type": "bearer", "expires_in": 3600, "refresh_token": "tGzv3JOkF0XG5Qx2TlKWIA" }
<i>access_token</i>	<i>Token</i> de acesso emitido pelo Servidor PDS que permite que o Provedor de Serviço acesse os recursos do Usuário . Exemplo: "2YotnFZFEjr1zCsicMWpAA"
<i>json_list</i>	Lista com URLs para <i>download</i> dos recursos compartilhados pelo Usuário com o Provedor de Serviço , conforme explicado na seção 5.5.3. Este parâmetro é usado apenas em acessos de leitura.
<i>download_url</i>	Um URL para <i>download</i> de um recurso, retirado do parâmetro <i>json_list</i> . Este parâmetro é usado apenas em acessos de leitura. Exemplo: http://pds.larc.usp.br/server/deliver_resource.do?rcode=febefa8367ec86401367ed275a7000d
<i>resource</i>	Arquivo referente ao recurso que é recebido ou enviado pelo Provedor de Serviço , dependendo se o acesso é de leitura ou escrita.

5.6.4.2 Acesso de leitura

A Tabela 13 apresenta as mensagens do protocolo do **Sistema PDS** para um acesso de leitura, conforme notações apresentadas na seção 5.6.4.1.

Para efeito de simplificação, todo o fluxo de autenticação OpenID, usando um **Provedor de Identidade**, entre o **Usuário** e o **Provedor de Serviço** e entre o **Usuário** e o **Servidor PDS** é representado por meio das mensagens M1 e M4, respectivamente. A representação [Mi] indica que a mensagem é repetida individualmente para cada recurso lido.

Tabela 13. Mensagens do protocolo do Sistema PDS para acesso de leitura

#	Direção	Conteúdo das mensagens trocadas
M1	U ↔ IdP ↔ S	<i>Autenticação OpenID</i>
M2	U → S	<i>PDS_url</i>
M3.1	S → U	<i>PDS_url, response_type, client_id, scope, redirect_url</i>
M3.2	U → PDS	<i>PDS_url, response_type, client_id, scope, redirect_url</i>
M4	U ↔ IdP ↔ PDS	<i>Autenticação OpenID</i>
M5	PDS → U	<i>scope, redirect_url, resource_list</i>
M6	U → PDS	<i>approved_resource_list, redirect_url, expiration_date</i>
M7.1	PDS → U	<i>code</i>
M7.2	U → S	<i>code</i>
M8	S → PDS	<i>client_id, client_secret, grant_type, code, redirect_url</i>
M9	PDS → S	<i>token</i>
M10	S → PDS	<i>access_token, scope</i>
M11	PDS → S	<i>json_list</i>
[M12]	S → PDS	<i>download_url</i>
[M13]	PDS → S	<i>resource</i>

5.6.4.3 Acesso de escrita

A Tabela 14 apresenta as mensagens do protocolo do **Sistema PDS** para um acesso de escrita, conforme notações apresentadas na seção 5.6.4.1.

Para efeito de simplificação, todo o fluxo de autenticação OpenID, usando um **Provedor de Identidade**, entre o **Usuário** e o **Provedor de Serviço** e entre o **Usuário** e o **Servidor PDS** é representado por meio das mensagens M1 e M4 respectivamente. A representação [Mi] indica que a mensagem é repetida individualmente para cada recurso escrito.

Tabela 14. Mensagens do protocolo do Sistema PDS para acesso de escrita

#	Direção	Conteúdo das mensagens trocadas
M1	U ↔ IdP ↔ S	<i>Autenticação OpenID</i>
M2	U → S	<i>PDS_url</i>
M3.1	S → U	<i>PDS_url, response_type, client_id, scope, redirect_url</i>
M3.1	U → PDS	<i>PDS_url, response_type, client_id, scope, redirect_url</i>
M4	U ↔ IdP ↔ PDS	<i>Autenticação OpenID</i>
M5	PDS → U	<i>scope, redirect_url</i>
M6	U → PDS	<i>approved_scope, redirect_url, expiration_date</i>
M7.1	PDS → U	<i>code</i>
M7.2	U → S	<i>code</i>
M8	S → PDS	<i>client_id, client_secret, grant_type, code, redirect_url</i>
M9	PDS → S	<i>Token</i>
[M10]	S → PDS	<i>access_token, scope, resource</i>
[M11]	PDS → S	<i>OK</i>

5.7 Considerações sobre o Sistema PDS

As seções anteriores descreveram a arquitetura e o protocolo do **Sistema PDS**, que usa as tecnologias OpenID e OAuth para habilitar suas funcionalidades de **gerenciamento**

de identidade e autorização, respectivamente. Inicialmente, esta seção discute como é feito o **gerenciamento de recursos e autorizações** nessa **arquitetura**. Depois, discutem-se as soluções adotadas pelo **Sistema PDS** para utilização do protocolo **OAuth**, pois a especificação deste protocolo oferece diferentes modos de utilização e deixa alguns componentes indefinidos (HAMMER-LAHAV; RECORDON; HARDT, 2012). É importante destacar que não é necessário realizar uma análise semelhante para o protocolo **OpenID**, pois a sua especificação, conforme descrito em (OPENID FOUNDATION, 2007), determina uma única opção de utilização e um fluxo bem definido de mensagens.

5.7.1 *Análise da Arquitetura*

Hoje, a maioria das aplicações *web* funde o serviço oferecido pela aplicação com o armazenamento de dados e o gerenciamento de autorizações, forçando o **Usuário** a acessar individualmente cada aplicação para obter seus dados e suas políticas. Essa abordagem apresenta algumas desvantagens: primeiro, existe um custo alto para os *websites*, pois é necessário que os **Provedores de Serviço** mantenham mecanismos de gerenciamento de autorizações, gerenciamento de recursos, segurança e armazenamento; segundo, como já mencionado anteriormente, dificulta o compartilhamento de dados entre diferentes aplicações *web*, pois cada aplicação oferece sua própria API; terceiro, o **Usuário** não possui controle de seus dados, cuja confidencialidade e integridade ficam à mercê da segurança e da honestidade dos *websites* (HSU; CHEN, 2009).

Uma solução para este problema é **centralizar** as políticas de compartilhamento em um serviço especializado responsável pelo **gerenciamento de autorizações**, com quem os **Provedores de Serviço** se comunicam para requisitar acesso a um recurso do **Usuário**. Nessa abordagem, os recursos continuam armazenados de forma distribuída nos próprios *sites* e apenas o **gerenciamento de autorizações** é centralizado. Entretanto, o custo para os **Provedores de Serviço** continua alto, pois eles têm que implementar mecanismos de **validação dos tokens** emitidos pelo **Serviço de Autorização** (por meio

de comunicação direta ou análise de assinatura), **gerenciamento de recursos, segurança e armazenamento**.

Outra solução é usar um **serviço centralizado** que é responsável tanto pelo **gerenciamento de autorização**, quanto pelo **gerenciamento e armazenamento de recursos**. As vantagens dessa abordagem são: o **Usuário** pode escolher livremente o provedor em quem confia para armazenar e gerenciar seus dados; o compartilhamento de dados entre aplicações *web* pode ser feito por meio de uma API padronizada; e o desenvolvimento de aplicações *web* fica mais simples e mais barato, já que elas não precisam gerenciar e armazenar dados de seus **Usuários**. A principal desvantagem em relação à primeira solução é que os recursos do **Usuário** ficam centralizados em um único serviço, que atua como intermediário para salvar e recuperar recursos do **Usuário**.

Conforme descrito nas seções 5.3 e 5.4, a arquitetura do **Sistema PDS** adota uma abordagem híbrida que combina as duas soluções anteriormente descritas. Para o caso geral, os **gerenciamentos de autorização e de recursos** são **centralizados** em uma única entidade: o **Servidor PDS**. Além das vantagens já citadas, essa decisão oferece baixa complexidade para as aplicações *web* para que elas se concentrem em sua lógica de negócios ao invés de ter que implementar mecanismos custosos para o compartilhamento de recursos e permite que os **Usuários** tenham controle centralizado sobre os seus recursos. Adicionalmente, o **Sistema PDS** possui um módulo de extensão chamado **Indexador**, descrito na seção 5.4.1.3, que permite que os **recursos** do **Usuário** sejam **armazenados de forma distribuída** em outros servidores ao mesmo tempo em que o **gerenciamento de recursos** continua centralizado. Neste caso, o **Provedor de Serviço** possui a possibilidade de continuar armazenando o recurso localmente desde que implemente os mecanismos adequados para compartilhamento segundo o protocolo OAuth. A principal vantagem dessa solução híbrida adotada pelo **Sistema PDS** é que ela atende a diferentes necessidades dos **Usuários** e **Provedores de Serviço**. Por exemplo, um desenvolvedor independente pode escolher usar a solução geral, em que as operações mais complexas são executadas pelo **Servidor PDS**, enquanto uma aplicação que dispõe de mais recursos pode escolher usar o módulo **Indexador**.

5.7.2 Adequação do Protocolo OAuth para o Sistema PDS

Conforme explicado na seção 4.2.2, o protocolo OAuth consegue atender aos requisitos de diversas aplicações, pois possui diferentes escopos de utilização. Por outro lado, a especificação do protocolo deixa alguns componentes parcialmente ou totalmente indefinidos (por exemplo, registro do cliente e descoberta do endereço do **Servidor de Autorização**). De fato, a especificação (HAMMER-LAHAV; RECORDON; HARDT, 2012) diz que o protocolo foi projetado com o objetivo de, no futuro, definir os perfis prescritivos e extensões necessários para alcançar a interoperabilidade integral na *Web*.

Dessa forma, para utilizar o protocolo OAuth neste trabalho, foi necessário inicialmente entender quais modos de utilização eram mais adequados para o **Sistema PDS**, conforme descrito na seção 4.2.2. De acordo com a especificação (HAMMER-LAHAV; RECORDON; HARDT, 2012), o **Sistema PDS** utiliza os seguintes modos do protocolo OAuth:

- **Tipo de Cliente:** Este trabalho usa o perfil “**Aplicação Web**”, classificado também como tipo privado, pois é o único dos três em que é possível garantir a confidencialidade das credenciais do **Provedor de Serviço** em relação ao **Usuário** e a outras aplicações instaladas no dispositivo do **Usuário**.
- **Modo de Autorização:** O modo de autorização escolhido para este trabalho foi o “**Código de Autorização**”, pois neste esquema o **Servidor de Autorização** atua como um intermediário entre o **Usuário** e o **Provedor de Serviço** de forma que o **Usuário** só se autentica com o **Servidor de Autorização** e suas credenciais nunca são compartilhadas com o **Provedor de Serviço**.
- **Tipo de Token:** O tipo de *token* escolhido para este trabalho foi o “**Bearer**” (JONES; HARDT; RECORDON, 2012), conforme apresentado na seção 5.5.4, onde foi apresentada a justificativa para escolha desse tipo de *token* e descrito o seu formato.

Depois, foi necessário propor soluções para preencher algumas lacunas que a especificação do OAuth não define. A lista a seguir explica as soluções adotadas pelo **Sistema PDS**.

- **Autenticação do Usuário:** O protocolo especifica que o **Servidor de Autorização** tem que autenticar o **Usuário**, mas não determina como essa autenticação deve ocorrer. A solução adotada foi o uso do protocolo OpenID para autenticação do **Usuário** e *Single Sign-On*, conforme foi explicado na seção 5.6.2.
- **Escopo da requisição de autorização:** O protocolo não especifica qual é o formato do parâmetro de escopo em que o **Provedor de Serviço** requisita acesso de leitura ou escrita para um conjunto de recursos específico e como os recursos são representados. A solução adotada foi o uso de URLs (e.g., <http://axschema.org/media/image/read> <http://pds.larc.usp.br/video/write>) para descrição do escopo da requisição conforme foi descrito nas seções 5.5.1 e 5.5.2.
- **Descoberta do Servidor de Autorização:** O protocolo não determina os meios pelos quais o **Provedor de Serviço** obtém o endereço do **Servidor de Autorização**. A solução adotada foi pedir para o **Usuário**, no primeiro uso do serviço, informar o endereço do seu **Servidor PDS** em um formulário no primeiro passo do protocolo. Em trabalhos futuros, pretende-se implementar um processo de descoberta de forma semelhante ao implementado pelo protocolo OpenID (OPENID FOUNDATION, 2007).
- **Comunicação entre Servidor de Recursos e Servidor de Autorização:** O protocolo não determina como o **Servidor de Recursos** deve se comunicar com o **Servidor de Autorização** para determinar se o *token* de acesso recebido é válido e, conseqüentemente, se o **Provedor de Serviço** pode acessar um recurso. No caso do **Sistema PDS**, escolheu-se reunir essas duas entidades em uma mesma entidade, o **Servidor PDS**, conforme foi discutido na seção 5.4. Uma vantagem dessa abordagem é que, uma vez que os **Servidores de Recursos e de Autorização** estão concatenados na mesma entidade, remove-se a necessidade de verificação externa do *token* de acesso.
- **Comunicação entre Provedor de Serviço e Servidor de Recursos:** O protocolo não determina como o **Provedor de Serviço** irá receber ou enviar um recurso ou um conjunto de recursos para **Servidor de Recursos**. Para um acesso de leitura, a solução adotada foi entregar uma lista contendo um URL para *download* individual de cada recurso (como descrito na seção 5.5.3) e, para um acesso de

escrita, o **Provedor de Serviço** deve entregar um arquivo por vez para o **Servidor de Recursos** como mostrado na descrição do protocolo na seção 5.6.4.

- **Registro do Provedor de Serviço:** O protocolo OAuth exige que o **Provedor de Serviço** se registre previamente no **Servidor de Autorização** para obter as credenciais de autenticação, mas não determina a forma como esse registro deve acontecer. A solução adotada foi criar um *site* com um formulário de registro, conforme discutido na seção 5.4.

5.8 *Resumo do Capítulo*

Este capítulo apresentou o **Sistema PDS** como uma proposta de **Sistema para Gerenciamento e Compartilhamento de Dados (SGCD)** para *Web*. Foram descritos a arquitetura e o protocolo do sistema de forma a permitir que o **Usuário** gerencie seus recursos na *Web* por meio de um serviço centralizado e compartilhe seus recursos com aplicações *web* por meio de um gerenciador centralizado de autorização. Além disso, descreveu-se como o protocolo OpenID é utilizado nessa arquitetura para habilitação das funcionalidades de autenticação, gerenciamento de identidades e *Single Sign-On*. Este capítulo apresentou também uma discussão a respeito de como utilizar o protocolo OAuth para autorização no cenário escolhido e quais as limitações do protocolo.

A seguir, o Capítulo 6 explorará como o **Sistema PDS**, descrito de forma genérica neste capítulo, pode ser utilizado em diferentes cenários de uso e atender às necessidades de aplicações com diferentes lógicas de negócios.

6 EXEMPLOS DE CENÁRIOS DE USO DO SISTEMA PDS

Como mencionado nos capítulos anteriores, o **Sistema PDS** foi projetado para habilitar o compartilhamento de tipos de dados genéricos entre aplicações *web*. Assim, espera-se que aplicações com diferentes lógicas de negócios possam se beneficiar desse sistema para compartilhar dados de forma centrada no **Usuário**.

Este capítulo exemplifica como dois cenários diferentes, em termos de lógica de negócios e requisitos de segurança do **Usuário** final, podem usar o **Sistema PDS**. O primeiro trata do ambiente de redes sociais em que aplicações *web* precisam ter acesso aos dados dos **Usuários** para prover serviços que realizam alguma operação sobre esses dados. Já o segundo cenário mostra um ambiente de Saúde Móvel (uso de dispositivos móveis para aplicações na área de saúde) em que o compartilhamento de dados médicos pode proporcionar uma melhor qualidade de assistência médica aos pacientes. Esses cenários são exemplos de uso do **Sistema PDS**. Não faz parte do escopo deste trabalho realizar uma análise aprofundada dos requisitos específicos de cada cenário.

6.1 Redes Sociais

Redes sociais proliferaram e evoluíram rapidamente nos últimos anos, atraindo a atenção de um número crescente de **Usuários** (BOYD; ELLISON, 2007). Esse crescimento da importância das redes sociais criou, também, a necessidade de serviços centrados no **Usuário**, que permitam que os **Usuários** controlem seus recursos ao compartilhar dados entre *sites* de redes sociais. Dessa forma, este é um cenário muito interessante para aplicação do **Sistema PDS**.

No cenário de redes sociais, um grande número de aplicações necessita manipular dados dos **Usuários** que foram criados por outras aplicações para oferecer os seus serviços. Grande parte dessas aplicações é criada por grupos com poucos desenvolvedores que possuem recursos bastante limitados para o desenvolvimento dos seus respectivos serviços. Neste contexto, a adoção do **Sistema PDS** faz com que esse tipo de aplicação não precise investir muitos recursos para implementar as funcionalidades relacionadas

ao compartilhamento de dados entre aplicações e possa focar nas funcionalidades específicas do seu serviço.

Outra vantagem do **Sistema PDS** é que, ao contrário da maioria das soluções de compartilhamento existentes para redes sociais, o sistema permite o compartilhamento de tipos de dados genéricos. Assim, o **Sistema PDS** pode habilitar uma nova gama de serviços que manipulam dados diferentes (por exemplo, XMLs genéricos, jogos e filmes) dos dados tradicionais de redes sociais (por exemplo, fotos, listas de contatos e *posts*).

A seguir é descrito um exemplo de uso do **Sistema PDS** para o cenário de redes sociais.

6.1.1 Exemplo de Uso

João acessa uma aplicação executada em um servidor *web* em “https://photoss.com” e edita uma foto da sua festa de formatura. Ao final, a aplicação pergunta se João quer salvar a foto editada em uma conta local ou na sua **conta PDS**. Então, ele instrui a aplicação a salvar a foto no seu **Servidor PDS** em “https://pds.larc.usp.br”. Com isso, o navegador *web* de João é redirecionado para o *site* do seu Servidor PDS onde ele irá se autenticar e autorizar que a aplicação “https://photoss.com” salve recursos do **tipo foto**. A aplicação de fotos recebe, então, a autorização e, posteriormente, salva a **foto** na conta de João no **Servidor PDS**.

Depois, João resolve escrever uma mensagem sobre a sua festa de formatura no *site* “https://blueblog.com” e deseja anexar à mensagem a foto que ele editou anteriormente no *site* “https://photoss.com”. Assim, ele instrui o *site* do *blog* a recuperar a foto no seu **Servidor PDS** em “https://pds.larc.usp.br”. Com isso, o navegador *web* de João é redirecionado para o *site* do seu **Servidor PDS** onde ele irá se autenticar e selecionar qual dado do **tipo fotos** a aplicação “https://blueblog.com” pode ler. A aplicação de *blog* recebe, então, a autorização e posteriormente recupera a foto da festa de formatura que está armazenada na conta de João no **Servidor PDS**.

6.2 Saúde Móvel

A tecnologia de Saúde Móvel, pode ser definida como a integração de computação móvel, sensores e dispositivos de comunicação para garantir ou melhorar serviços na área de saúde (ISTEPANIAN; JOVANOVA; ZHANG, 2004). Esta tecnologia está relacionada ao conceito de *e-Health* (processos eletrônicos na área de saúde), mas esta última é mais focada em computadores *desktop*, enquanto a primeira tenta explorar mais intensamente os avanços em comunicação sem fio, computação ubíqua e sensores na área da saúde (TACHAKRA *et al.*, 2003).

O uso do **Sistema PDS** neste cenário está ligado à facilitação da troca de informações médicas do paciente, tais como, resultados de exames, prescrição de medicamentos e sinais capturados por sensores que monitoram algum aspecto da saúde do paciente. A possibilidade de compartilhar dados médicos, com o consentimento do paciente, entre unidades de atendimento de saúde pode proporcionar uma melhor qualidade de atendimento aos pacientes e uma assistência mais eficaz. Por exemplo, pode-se obter mais facilmente o histórico médico de um paciente ou evitar realizar um exame que já foi feito recentemente.

Como o **Sistema PDS** não exige a instalação de nenhum aplicativo no dispositivo do **Usuário**, a idéia é que o celular do paciente possa ser usado como agente do **Usuário** desde que suporte um navegador *web* padrão. Assim, o paciente pode autorizar o acesso a seus dados médicos no **Sistema PDS**, de forma segura, usando o seu celular. Além disso, pode-se tirar proveito do uso de celulares para autenticar o **Usuário** usando um mecanismo de autenticação de dois fatores baseado em uma senha que o **Usuário** sabe e no cartão *SIM* do celular que o **Usuário** possui. Um exemplo desse mecanismo é a tecnologia GBA (*Generic Bootstrapping Architecture*) (HOLTMANN *et al.*, 2009).

Um desafio para o compartilhamento de dados médicos refere-se ao fato de não existir um modelo padronizado para representação dos diferentes tipos de dados médicos que seja amplamente adotado. Logo, o **Sistema PDS** pode ser usado neste cenário, pois ele permite o compartilhamento de tipos de dados genéricos sem precisar entender o modelo de dados de cada tipo. Assim, se uma aplicação quiser ler um recurso que foi previamente escrito por outra aplicação, tais aplicações precisam utilizar o mesmo tipo

de dados (conforme descrito na seção 5.5.1); por exemplo, se um hospital escreveu um tipo de dado chamado “http://hospitalx.com/examex” na **conta PDS** de um paciente e outro hospital deseja ler este dado, será necessário um acordo prévio entre os dois hospitais sobre qual é o URL que representa o tipo de dados envolvido. Para os tipos de dados mais comuns já padronizados, o próprio **Servidor PDS** ou uma entidade terceira conhecida por todos poderia definir os URLs dos tipos de dados necessários para aplicações médicas, como, por exemplo, “http://pds.larc.usp.br/radiografia”.

Outro desafio importante diz respeito à escolha do **Usuário** sobre qual entidade ele considera suficientemente confiável para ser o seu **Servidor PDS**, ou seja, a entidade que controlará os seus dados médicos. No caso médico, a relação de confiança entre o **Usuário** e o seu **Servidor PDS** é muito rigorosa, pois o vazamento indevido dessas informações pode ter um impacto altamente negativo na vida pessoal ou profissional do paciente. Assim, uma entidade em que o **Usuário** confia para armazenar informações de redes sociais, como fotos e mensagens, pode não ser considerada confiável ou adequada para armazenar seu histórico médico¹⁴. Uma sugestão para este problema é adotar o **Sistema PDS** como solução de compartilhamento de dados médicos do governo federal, estadual e/ou municipal de forma que o Ministério da Saúde ou a Secretaria de Saúde pudesse atuar como essa entidade confiável. Como exemplo, pode-se mencionar o caso de utilização de uma instituição governamental, como entidade confiável para o gerenciamento de dados médicos, que foi adotada pelo governo do Canadá para um sistema nacional de registros de saúde eletrônicos (GIOKAS, 2005).

A seção a seguir descreve um exemplo de uso do **Sistema PDS** para o cenário de Saúde Móvel.

6.2.1 Exemplo de Uso

Alice acessa uma aplicação executada em um servidor *web* em “https://hospitalsaintpeter.com” e instrui esta aplicação a acessar seus exames de radiografia e ultrasonografia que estão armazenados no seu **Servidor PDS**:

¹⁴ Como exemplo, pode-se citar a iniciativa do Google com o serviço Google Health para armazenamento e compartilhamento de dados médicos do **Usuário** que foi descontinuado, pois não obteve boa adesão por parte dos **Usuários** (BROWN; WEIHL, 2011).

“https://ministeriodasaude.gov.br/pds”. O navegador *web* de Alice é então redirecionado para o *site* do seu **Servidor PDS**, onde ela irá se autenticar e autorizar que a aplicação em “https://hospitalsaintpeter.com” acesse seus exames. A aplicação do hospital *Saint Peter* recebe, então, a autorização de Alice e, posteriormente, troca essa autorização pelos arquivos solicitados que estão armazenados no **Servidor PDS**.

O Capítulo 8, que descreve o protótipo implementado para o sistema, utilizará um exemplo semelhante e mostrará as telas correspondentes da interface com o **Usuário**.

6.3 *Resumo do Capítulo*

Este capítulo mostrou por meio de dois exemplos bastante distintos (redes sociais e Saúde Móvel) que o **Sistema PDS** é flexível e atende às necessidades de compartilhamento de aplicações *web* com requisitos e lógicas de negócio bem diferentes. Além disso, os exemplos ajudam a entender melhor o funcionamento do sistema em cenários reais e os benefícios para os **Usuários** e as aplicações envolvidas. Nos dois cenários, o uso do **Sistema PDS** proporciona aos **Usuários** maior controle sobre seus dados na *Web*. No cenário de Redes Sociais, em especial, as aplicações têm acesso mais fácil aos dados do **Usuário** e podem beneficiá-lo ao oferecer serviços que manipulam esses dados para gerar novas funcionalidades. No cenário de Saúde Móvel, o gerenciamento e compartilhamento de dados médicos podem trazer benefícios ligados a melhoria do atendimento médico aos pacientes e até redução de custos de operação dos hospitais e do próprio sistema de saúde, seja ele público ou privado, pela redução de exames médicos repetidos.

No capítulo a seguir, será apresentada uma análise das vulnerabilidades e ameaças de segurança do **Sistema PDS**, que permite a verificação do cumprimento dos requisitos de segurança relacionados à confidencialidade, integridade e disponibilidade. Posteriormente, o Capítulo 8 descreverá um protótipo que implementa a arquitetura do **Sistema PDS** e apresentará exemplos de utilização do sistema por meio de telas de interface com o **Usuário** do protótipo.

7 ANÁLISE DE VULNERABILIDADES E AMEAÇAS DO SISTEMA PDS

A análise de ameaças é um processo formal de identificação, documentação e mitigação das ameaças de segurança de um sistema (STANGO; PRASAD; KYRIAZANOS, 2009). Este capítulo realiza uma ampla análise de ameaças do **Sistema PDS**, cuja arquitetura foi proposta no Capítulo 5, com o objetivo de detectar potenciais vulnerabilidades e propor um plano com medidas de mitigação ou eliminação das vulnerabilidades encontradas. Esse plano será utilizado como base para um protótipo, apresentado posteriormente no Capítulo 8, que aplica as medidas selecionadas para proteger o sistema contra a violação dos seus serviços de segurança e, conseqüentemente, atende aos requisitos relacionados à proteção da confidencialidade, integridade e disponibilidade definidos no Capítulo 3 (RNF-01e RNF-06).

7.1 Conceitos

A RFC 2828 (SHIREY, 2000) define os seguintes conceitos:

- **Vulnerabilidade:** uma falha ou fraqueza no projeto, implementação, configuração, operação ou gerenciamento de um sistema que pode ser explorada para violar a política de segurança do sistema.
- **Ameaça:** uma possibilidade de violação de segurança, que existe quando há uma circunstância, capacidade, ação ou evento que possa violar a segurança e causar danos. Ou seja, uma ameaça é um possível perigo que pode explorar uma vulnerabilidade.
- **Ameaça Inteligente:** uma circunstância em que um adversário tem a capacidade técnica e operacional para detectar e explorar uma vulnerabilidade, e também tem a intenção demonstrada, presumida, ou inferida de fazê-lo.
- **Ataque:** uma violação de segurança do sistema que deriva de uma ameaça inteligente, i.e., uma ação que é uma tentativa deliberada (especialmente no

sentido de um método ou uma técnica) para invadir serviços de segurança e violar a política de segurança de um sistema.

Uma análise de ameaças pode fornecer as seguintes habilidades (BISHOP, 2002): especificar, projetar e implementar um sistema de computação sem vulnerabilidades; analisar um sistema de computação para detectar vulnerabilidades; tratar qualquer vulnerabilidade que possa estar presente durante a operação de um sistema de computação; e detectar tentativas de explorações de vulnerabilidades de um sistema de computação.

7.2 Trabalhos Relacionados sobre Análise de Ameaças

Há relativamente poucos estudos na literatura focados em métodos formais e gerais para análise de ameaças em sistemas complexos e em rede. Entre as obras mais relevantes, podem-se citar os trabalhos desenvolvidos em (STANGO; PRASAD; KYRIAZANOS, 2009) e (MYAGMAR; LEE; YURCIK, 2005): no primeiro, os autores propõem um método genérico para a análise de ameaças em redes pessoais; no segundo, os autores investigam como a análise de ameaças pode ser usada como base para a especificação de requisitos de segurança e definem uma abordagem para identificar ameaças em sistemas em rede. Em comparação, é mais comum encontrar trabalhos que fornecem um modelo de ameaça para algum sistema específico. Exemplos incluem a análise de ameaças de *Web Services* e *Grids* (NI *et al.*, 2007) (DEMCHENKO *et al.*, 2005), de redes de sensores (CARDENAS; ROOSTA; SASTRY, 2009), e de sistemas móveis (LIU *et al.*, 2009).

Em relação ao sistema proposto, existem alguns trabalhos que analisam as vulnerabilidades do protocolo OpenID (TSYRKLEVICH; TSYRKLEVICH, 2007) (OH; JIN, 2008) (CHADWICK; SHAW, 2008) (LEE *et al.*, 2008) e do protocolo OAuth (LODDERSTEDT; MCGLOIN; HUNT, 2012) (HSU; LEE, 2010). No entanto, estes trabalhos normalmente concentram-se em uma ou mais vulnerabilidades específicas e não realizam uma análise de ameaças abrangente. Além disso, não foi encontrada nenhuma iniciativa que considera a modelagem de ameaças das características de um sistema parecido com o sistema sob estudo, que integra as tecnologias OpenID e OAuth.

7.3 Método e Classificação

Neste trabalho, será usado um método para análise de ameaças que combina os processos de dois estudos relacionados: (STANGO; PRASAD; KYRIAZANOS, 2009) e (MYAGMAR; LEE; YURCIK, 2005). O processo resultante consiste em três fases principais: modelagem de ameaças, gestão de riscos e plano de mitigação, como mostrado na Figura 27.

Figura 27. Processo de análise de ameaças

Inicialmente, é feita a modelagem de ameaças, que envolve quatro etapas: caracterização do sistema, identificação de ativos, determinação de ameaças e determinação de vulnerabilidades. Após a modelagem de ameaças, a próxima fase é a gestão de risco, onde as ameaças e vulnerabilidades identificadas anteriormente são classificadas com base no seu impacto sobre os ativos, e a probabilidade de que a ameaça vai acontecer. Finalmente, a última fase é o plano de mitigação, em que as medidas preventivas para cada ameaça são identificadas e toma-se a decisão de reduzir, remover, transferir ou aceitar os riscos relacionados.

As vulnerabilidades de um sistema podem ser classificadas de acordo com várias perspectivas, tais como: as técnicas utilizadas para explorá-las, a sua natureza, ou os

componentes de *software* e *hardware* que compõem a vulnerabilidade (BISHOP, 2002). Neste trabalho, será usada a taxonomia de atores para agrupar as ameaças e vulnerabilidades identificadas, de forma semelhante ao modelo descrito em (NI *et al.*, 2007). Em primeiro lugar, é necessário identificar os atores (isto é, os principais componentes do sistema) e suas responsabilidades na operação do sistema. Então, as ameaças são identificadas, agrupadas e classificadas de acordo com o componente do sistema (ou ator) que elas afetam.

7.4 Caracterização do Atacante

Para a Análise de Ameaças, foram feitas as seguintes considerações a respeito do atacante:

- O atacante possui recursos computacionais limitados.
- O atacante pode escutar o meio de transmissão de mensagens.
- Em alguns casos, o atacante pode obter acesso à máquina do **Usuário** (e.g., por meio da instalação de aplicativos maliciosos). Quando isso acontecer, esses casos serão indicados na descrição da ameaça/vulnerabilidade.
- Em alguns casos, o atacante pode atuar como um dos componentes do sistema de forma maliciosa (e.g., um **Servidor PDS** ou um **Provedor de Identidade** forjado). Quando isso acontecer, esses casos serão indicados na descrição da ameaça/vulnerabilidade.

7.5 Modelagem de Ameaças

Esta seção realiza a modelagem de ameaças que é a primeira fase do processo de análise de ameaças e envolve as etapas de caracterização do sistema, identificação de ativos, determinação de ameaças e determinação de vulnerabilidades.

7.5.1 Caracterização do sistema

Essa fase engloba a identificação dos atores e suas responsabilidades no sistema. De acordo com a arquitetura do sistema descrita no Capítulo 5 (Figura 20), é possível determinar os seguintes atores: **Usuário**, **Provedor de Serviço**, **Servidor PDS** e **Provedor de Identidade**. A funcionalidade de cada um desses atores foi discutida na seção 5.3.

7.5.2 Identificação de ativos

Identificar ativos que podem ser danificados ou violados no sistema é importante a fim de determinar o que efetivamente necessita ser protegido (STANGO; PRASAD; KYRIAZANOS, 2009) (MYAGMAR; LEE; YURCIK, 2005). Os principais ativos do sistema sob estudo são apresentados na Tabela 15, e classificados de acordo com a taxonomia de atores.

Tabela 15. Ativos do Sistema PDS

Categoria	Código	Ativo	Descrição
Usuário	AU01	Credenciais do Usuário	Credenciais usadas pelo Usuário para autenticação no Provedor de Identidade .
	AU02	Atividades do Usuário	Lista dos <i>sites</i> visitados pelo Usuário .
	AU03	Sessão do Usuário	Sessão do Usuário no navegador <i>web</i> devidamente autenticada pelo Provedor de Identidade .
Provedor de Serviço	AS01	Credencial de acesso ao Servidor PDS	Credencial (chave+segredo) utilizada para autenticação do Provedor de Serviço junto ao Servidor PDS .
	AS02	<i>Tokens</i> de acesso	Conjunto de <i>tokens</i> de acesso de cada Usuário registrado no Provedor de Serviço .
	AS03	<i>Tokens</i> de atualização	Conjunto de <i>tokens</i> de atualização de cada Usuário registrado no Provedor de Serviço .
	AS04	Códigos de autorização	Conjunto de códigos de autorização de cada Usuário registrado no Provedor de Serviço .
	AS05	Disponibilidade do Serviço	Disponibilidade dos serviços oferecidos pelo Provedor de Serviço deve ser garantida aos Usuários autorizados.
Servidor PDS	AP01	Recursos dos Usuários	O conjunto de recursos protegidos de cada Usuário registrado no Servidor PDS .
	AP02	Credenciais dos Provedores de Serviço	Credenciais (chave/segredo) usadas para autenticação de cada Provedor de Serviço no Servidor PDS .
	AP03	<i>Tokens</i> de acesso	Conjunto de <i>tokens</i> de acesso e as respectivas informações sobre eles (e.g., data de expiração, URL de redirecionamento e identificador do Provedor de Serviço) de cada Usuário registrado no Servidor PDS .
	AP04	<i>Tokens</i> de atualização	Conjunto de <i>tokens</i> de atualização relativos a cada <i>token</i> de acesso (se existir).
	AP05	Códigos de autorização	Conjunto de códigos de autorização e as respectivas informações sobre eles (e.g., data de expiração, URL de redirecionamento e identificador do Provedor de Serviço) de cada Usuário registrado no Servidor PDS .
	AP06	Disponibilidade do Serviço	Disponibilidade dos serviços oferecidos pelo Servidor PDS deve ser garantida aos Usuários autorizados.
Provedor de Identidade	AI01	Credenciais dos Usuários	Credenciais usadas pelos Usuários para autenticação no Provedor de Identidade .
	AI02	Disponibilidade do Serviço	Disponibilidade dos serviços oferecidos pelo Provedor de Identidade deve ser garantida aos Usuários autorizados.

Legenda: AU: Ativo do **Usuário**, AS: Ativo do **Provedor de Serviço**, AP: Ativo do **Servidor PDS**, AI: Ativo do **Provedor de Identidade**

7.5.3 Determinação de Ameaças e Vulnerabilidades

Usando as informações geradas nos passos anteriores e analisando o fluxo de mensagens delineado na descrição da arquitetura do sistema (Capítulo 5), é possível identificar um conjunto de ameaças para o sistema.

Para enumeração de ameaças foram utilizados três métodos: percorrer cada um dos ativos do sistema e criar hipóteses de ameaças que violam os serviços de segurança (confidencialidade, integridade e disponibilidade) desses ativos (BISHOP, 2002) (MYAGMAR; LEE; YURCIK, 2005); estudar trabalhos que descrevem vulnerabilidades específicas dos protocolos OpenID e OAuth (TSYRKLEVICH; TSYRKLEVICH, 2007) (OH; JIN, 2008) (CHADWICK; SHAW, 2008) (LEE *et al.*, 2008) (LODDERSTEDT; MCGLOIN; HUNT, 2012) (HAMMER-LAHAV; RECORDON; HARDT, 2012) (HSU; LEE, 2010); e analisar vulnerabilidades relacionadas na lista do CWE¹⁵ (*Common Weakness Enumeration*) (MARTIN, 2011).

As próximas seções mostrarão as ameaças e vulnerabilidades enumeradas nas categorias **Usuário**, **Provedor de Serviço**, **Servidor PDS** e **Provedor de Identidade**, segundo a classificação de atores.

7.5.3.1 Ameaças e Vulnerabilidades da Categoria Usuário

A Tabela 16 mostra as ameaças identificadas na categoria **Usuário**, segundo um modelo que identifica cada ameaça por um código, um nome e uma descrição. Em seguida, a Tabela 17 apresenta as vulnerabilidades e ativos associados a cada uma dessas ameaças.

¹⁵ O CWE (*Common Weakness Enumeration*) é uma lista que enumera as deficiências de segurança bem conhecidas em sistemas de *software* em geral para servir como uma base de conhecimento comum para identificação de vulnerabilidades e medidas de prevenção e mitigação para essas vulnerabilidades. O documento pode ser encontrado em <http://cwe.mitre.org/data/>.

Tabela 16. Categoria Usuário: ameaças do Sistema PDS

Código	Ameaça	Descrição
TU01	Captura das credenciais do Usuário usando um navegador comprometido	Um aplicativo malicioso pode realizar um ataque de <i>phishing</i> ¹⁶ contra as credenciais do Usuário por meio da utilização de um navegador comprometido. Neste caso, o navegador faria a renderização de uma página falsa no processo de autorização.
TU02	Captura de códigos de autorização ou <i>tokens</i> por meio de vazamento no navegador <i>web</i>	Se os códigos de autorização são passados como parâmetros de consulta no URL, um navegador <i>web</i> pode, de forma não intencional, vaziar essas informações para atacantes de diferentes maneiras: <ul style="list-style-type: none"> – Navegadores <i>web</i> frequentemente passam um cabeçalho de referência (<i>Referre Header</i>) para o <i>site</i> de destino, que contém o endereço do <i>site</i> de origem, quando um Usuário navega de um <i>site</i> para outro. – <i>Websites</i> podem enviar os Usuários para outro destino através de um redirecionador e vaziar códigos de verificação para <i>sites</i> de destino não confiáveis. – Navegadores <i>web</i> comumente gravam os URLs visitados no histórico do navegador. Outro Usuário do mesmo navegador pode ser capaz de visualizar URLs que foram visitadas por Usuários anteriores.
TU03	Captura das credenciais do Usuário por meio de Provedor de Identidade forjado	O uso dos protocolos OpenID e OAuth pode fazer com que os Usuários se acostumem a serem redirecionados para <i>sites</i> onde eles são convidados a inserir suas senhas. Um atacante pode tirar proveito disso para interceptar requisições a um Provedor de Identidade , retornando respostas enganosas ou incorretas usando <i>spoofing</i> ¹⁷ de DNS (<i>Domain Name System</i>) ou ARP (<i>Address Resolution Protocol</i>). Se o Usuário não tiver o cuidado de verificar a autenticidade do seu Provedor de Identidade antes de entregar suas credenciais, o atacante pode roubá-las. Esse ataque também pode ser realizado por Servidores PDS ou Provedores de Serviço maliciosos que redirecionam o Usuário para uma página falsa visualmente idêntica ao Provedor de Identidade do Usuário com o objetivo de roubar credenciais.
TU04	Captura das credenciais do Usuário por escuta do meio de transmissão	Um atacante pode tentar capturar as credenciais do Usuário por meio de um ataque de escuta do meio de transmissão (<i>eavesdropping</i>) no momento em que o Usuário autentica-se junto ao seu Provedor de Identidade .

Legenda: TU: Ameaça do **Usuário**

¹⁶ O ataque de *Phishing* é uma técnica de engenharia social, em que um atacante se disfarça como uma entidade legítima em quem a vítima confia para solicitar que esta revele alguma informação confidencial (THE MITRE CORPORATION, 2012).

¹⁷ No ataque de *spoofing*, um atacante interage com a vítima, de tal modo a convencê-la que está interagindo com outra entidade, com o objetivo de realizar ações baseadas no nível de confiança que existe entre a vítima e essa entidade (THE MITRE CORPORATION, 2012).

Tabela 17. Categoria Usuário: relação das ameaças com as vulnerabilidades e ativos

Ameaça		Vulnerabilidade Associada		Ativos Afetados	
Cód.	Descrição	Cód.	Descrição	Cód.	Descrição
TU01	Captura das credenciais do Usuário usando um navegador comprometido	VU01	Instalação de aplicações maliciosas na máquina do Usuário	AU01	Credenciais do Usuário
TU02	Captura de códigos de autorização ou <i>tokens</i> por vazamento no navegador <i>web</i>	VU02	Navegador <i>web</i> do Usuário vazava informações sobre parâmetros de requisições	AS02	<i>Tokens</i> de acesso de um Provedor de Serviço
				AS03	<i>Tokens</i> de atualização de um Provedor de Serviço
				AS04	Códigos de Autorização de um Provedor de Serviço
TU03	Captura das credenciais do Usuário por meio de Provedor de Identidade forjado	VU03	O Usuário não verifica a autenticidade do Provedor de Identidade	AU01	Credenciais do Usuário
TU04	Captura das credenciais do Usuário por escuta do meio de transmissão	VU04	Meio de transmissão insuficientemente protegido	AU01	Credenciais do Usuário

Legenda: *TU:* Ameaça do **Usuário**, *VU:* Vulnerabilidade do **Usuário**, *AU:* Ativo do **Usuário**

7.5.3.2 Ameaças e Vulnerabilidades da Categoria Provedor de Serviço

A Tabela 18 mostra as ameaças identificadas na categoria **Provedor de Serviço**, segundo um modelo que identifica cada ameaça por um código, um nome e uma descrição. Em seguida, a Tabela 19 apresenta as vulnerabilidades e ativos associados a essas ameaças.

Tabela 18. Categoria Provedor de Serviço: ameaças do Sistema PDS

Código	Ameaça	Descrição
TS01	Captura de credenciais do Provedor de Serviço	Um invasor obtém as credenciais do Provedor de Serviço por meio da superação dos mecanismos de segurança do servidor <i>web</i> .
TS02	Captura de <i>tokens</i> de acesso e de atualização no Provedor de Serviço	Um invasor obtém os <i>tokens</i> de um Provedor de Serviço por meio da superação dos mecanismos de segurança do servidor <i>web</i> .
TS03	Captura de <i>tokens</i> usando redirecionadores abertos no Provedor de Serviço	Um redirecionador aberto é um URL usado para redirecionar automaticamente um agente de Usuário para o endereço especificado sem a validação adequada desse URL. Se o Servidor PDS permitir que o Provedor de Serviço registre apenas uma parte do URL de redirecionamento, um atacante pode usar um redirecionador aberto existente no Provedor de Serviço para construir um URL que irá enviar o código de autorização ou <i>token</i> a um endereço controlado pelo atacante. Exemplo: <code>redirect_url = http://site.com/redirect.html?q=http://attackersite.com</code>
TS04	Captura de <i>tokens</i> de acesso e de atualização por meio de Servidor PDS forjado	Um atacante pode capturar <i>tokens</i> válidos ao interceptar requisições para o Servidor PDS usando <i>spoofing</i> de DNS ou ARP. Se um Provedor de Serviço enviar um <i>token</i> válido para este servidor falsificado, o atacante pode usar esse <i>token</i> para acessar recursos do Usuário em nome do Provedor de Serviço legítimo.
TS05	Ataque de <i>Cross-Site Request Forgery (CSRF)</i>	Nesse ataque, o Provedor de Serviço não verifica se uma requisição consistente foi fornecida intencionalmente pelo Usuário autenticado que submeteu a requisição. Com isso, o Provedor de Serviço associa a conta do Usuário no seu serviço com a conta PDS do atacante. O ataque segue o seguinte fluxo: 1. O atacante autoriza um código de autorização para os seus próprios recursos no Servidor PDS e aborta o redirecionamento de volta para o Provedor de Serviço . 2. Em seguida, ele induz outro Usuário (vítima) a executar o redirecionamento para esse Provedor de Serviço (por exemplo, induz o Usuário a clicar em um <i>link</i>). 3. O Provedor de Serviço recebe o código de autorização, busca o <i>token</i> correspondente no Servidor PDS e associa a sessão da vítima no seu serviço com os recursos acessíveis usando o <i>token</i> do atacante. Assim, o Provedor de Serviço pode, por exemplo, salvar recursos da vítima em uma conta PDS controlada pelo atacante ou ler recursos falsos ao recuperar recursos da conta do atacante.
TS06	URL malicioso	Quando um Usuário faz <i>login</i> em um Provedor de Serviço , ele fornece um URL com a sua identidade OpenID. Com isso, o Provedor de Serviço faz o <i>download</i> do URL e extrai o endereço do Provedor de Identidade . O mesmo acontece quando o Usuário fornece o endereço do seu Servidor PDS no processo de autorização. Neste contexto, um atacante pode inserir um URL malicioso e levar a ataques nocivos, como exploração de scripts internos e ataques de negação de serviço.

TS07	Troca de sessão.	Um atacante pode forçar um Usuário a fazer <i>login</i> em um Provedor de Serviço usando uma conta OpenID controlada pelo atacante. Isso acontece porque o protocolo OpenID não vincula as asserções positivas OpenID ao navegador do Usuário . Exemplo: Alice autentica-se em seu Provedor de Identidade , e em vez de entregar a asserção positiva para um determinado <i>site</i> , ela engana Bob e passa a asserção para ele (por exemplo, induz Bob a clicar em um <i>link</i>), que então autentica nesse <i>site</i> usando a asserção de Alice. O Usuário pode não perceber que fez <i>login</i> como o atacante, permitindo que o atacante monitore suas atividades naquele <i>site</i> .
-------------	------------------	---

Legenda: **TS:** Ameaça do Provedor de Serviço

Tabela 19. Categoria Provedor de Serviço: relação das ameaças com as vulnerabilidades e ativos

Ameaça		Vulnerabilidade Associada		Ativos Afetados	
Cód.	Descrição	Cód.	Descrição	Cód.	Descrição
TS01	Captura de credenciais do Provedor de Serviço	VS01	Credenciais do Provedor de Serviço insuficientemente protegidas	AS01	Credencial do Provedor de Serviço para acesso no Servidor PDS
TS02	Captura de <i>tokens</i> de acesso e de atualização no Provedor de Serviço	VS02	<i>Tokens</i> do Provedor de Serviço insuficientemente protegidos	AS02	<i>Tokens</i> de acesso de um Provedor de Serviço
				AS03	<i>Tokens</i> de atualização de um Provedor de Serviço
TS03	Captura de <i>tokens</i> usando redirecionadores abertos no Provedor de Serviço	VS03	O Provedor de Serviço aceita redirecionadores abertos	AS02	<i>Tokens</i> de acesso de um Provedor de Serviço
				AS03	<i>Tokens</i> de atualização de um Provedor de Serviço
				AS04	Código de Autorização de um Provedor de Serviço
TS04	Captura de <i>tokens</i> de acesso e de atualização por meio de Servidor PDS forjado	VS04	O Provedor de Serviço não autentica o Servidor PDS	AS02	<i>Tokens</i> de acesso de um Provedor de Serviço
				AS03	<i>Tokens</i> de atualização de um Provedor de Serviço
TS05	Ataque de <i>Cross-Site Request Forgery</i> (CSRF)	VS05	Provedor de Serviço suscetível a ataque de CSRF	AP01	Recursos de um Usuário
TS06	URL malicioso	VS06	Validação imprópria de dados de entrada	AS05	Disponibilidade do serviço do Provedor de Serviço
TS07	Troca de sessão	VS07	Verificação insuficiente da autenticidade dos dados na autenticação	AU02	Atividades do Usuário

Legenda: **TS:** Ameaça do Provedor de Serviço, **VS:** Vulnerabilidade do Provedor de Serviço, **AS:** Ativo do Provedor de Serviço

7.5.3.3 Ameaças e Vulnerabilidades da Categoria Servidor PDS

A Tabela 20 mostra as ameaças identificadas na categoria **Servidor PDS**, segundo um modelo que identifica cada ameaça por um código, um nome e uma descrição. Em seguida, a Tabela 21 apresenta as vulnerabilidades e ativos associados a essas ameaças.

Tabela 20. Categoria Servidor PDS: ameaças do Sistema PDS

Código	Ameaça	Descrição
TP01	Usuário involuntariamente concede escopo de acesso incorreto a um Provedor de Serviço	Ao aprovar uma requisição de autorização no Servidor PDS , o Usuário pode não compreender o escopo de acesso a ser concedido e para quem está concedendo o acesso e, por isso, conceder um escopo de acesso incorreto a um Provedor de Serviço .
TP02	Captura de credenciais do Usuário por meio de redirecionadores abertos no Servidor PDS	Um redirecionador aberto é um URL usado para redirecionar automaticamente um agente de Usuário para o endereço especificado sem a validação adequada desse URL. Um atacante poderia usar o Servidor PDS como um redirecionador aberto usando o parâmetro do URL de redirecionamento do protocolo. Com isso, um atacante pode utilizar a confiança do Usuário em seu Servidor PDS para fazer um ataque <i>phishing</i> .
TP03	Captura de informações confidenciais por escuta do meio de transmissão	Um atacante pode tentar capturar informações confidenciais por meio de um ataque de escuta do meio de transmissão (<i>eavesdropping</i>) nas seguintes etapas do protocolo: <ul style="list-style-type: none"> – No registro do Provedor de Serviço, quando o Servidor PDS transmite as credenciais para o Provedor de Serviço. – Nos processos de autenticação ou requisições de <i>token</i> do Provedor de Serviço, quando o Provedor de Serviço transmite as credenciais para o Servidor PDS. – Na transmissão de códigos de autorização e <i>tokens</i> entre o Servidor PDS e o Provedor de Serviço.
TP04	Captura de <i>tokens</i> , códigos de autorização e credenciais de Provedores de Serviço por meio de ataque ao banco de dados do Servidor PDS	Um atacante pode obter os <i>tokens</i> e as credenciais do banco de dados do Servidor PDS por meio de um ataque de injeção SQL (<i>SQL Injection</i>) ao banco de dados.
TP05	Captura de credenciais do Provedor de Serviço por tentativa <i>online</i>	Um atacante tenta adivinhar pares de identificador/segreto válidos.
TP06	Captura de códigos de autorização e <i>tokens</i> por tentativa <i>online</i>	Um atacante tenta adivinhar códigos de autorização ou <i>tokens</i> válidos.

TP07	Ataque de negação de serviço na criação dinâmica de segredos do Provedor de Serviço	Se o Servidor PDS possui uma quantidade baixa de entropia para segredos do Provedor de Serviço (restringindo assim o número de segredos possíveis) e esses segredos são entregues automaticamente no processo de registro, o invasor pode esgotar as possibilidades fazendo repetidas requisições de registro.
TP08	Ataque de negação de serviço na criação de códigos de autorização e <i>tokens</i>	Se o Servidor PDS possui uma quantidade baixa de entropia para códigos de autorização ou <i>tokens</i> (restringindo assim o número de possíveis códigos/ <i>tokens</i>), concede ambos automaticamente e não tem limite no número de <i>tokens</i> por Usuário , um atacante pode esgotar as possibilidades disponíveis ao redirecionar o navegador do Usuário repetidas vezes para solicitar <i>tokens</i> ou códigos.
TP09	Captura de código de autorização por Provedor de Serviço falsificado	Um atacante personifica o <i>site</i> do Provedor de Serviço para capturar um código de autorização usando <i>spoofing</i> de DNS ou ARP.
TP10	Captura e reprodução de requisições válidas no acesso a recursos protegidos	Quando um Provedor de Serviço possui um <i>token</i> de acesso, ele requisita ao Servidor PDS acesso a recursos protegidos do Usuário . Neste momento, um atacante pode tentar capturar e reproduzir (<i>capture & replay</i>) essa requisição válida de forma a conseguir acesso aos recursos do Usuário .
TP11	Ataque de <i>clickjacking</i> contra a autorização do Usuário	No ataque de <i>clickjacking</i> , um <i>site</i> malicioso carrega o <i>site</i> alvo em um <i>iframe</i> ¹⁸ transparente sobreposto em cima de um conjunto de botões fictícios que são cuidadosamente construídos para serem posicionados sobre botões importantes no <i>site</i> alvo. Quando um Usuário clica em um botão visível, ele está, na verdade, clicando em um botão na página escondida (por exemplo, o botão de Autorizar na página de autorização do Servidor PDS).
TP12	Personificação da sessão do Usuário	Este ataque visa associar uma autorização de acesso legítima com uma conta do atacante no Provedor de Serviço para obter acesso aos recursos protegidos de um Usuário . O atacante realiza o seguinte fluxo: 1. O atacante captura o código de autorização quando ele é enviado pelo navegador do Usuário (vítima) para o endereço de redirecionamento do Provedor de Serviço usando um ataque de <i>spoofing</i> ou de escuta do meio de transmissão. 2. O atacante acessa o mesmo Provedor de Serviço com uma conta de Usuário própria e submete o código capturado a fim de associar uma autorização de acesso aos recursos da vítima com a conta do atacante no Provedor de Serviço legítimo. 3. O Provedor de Serviço irá trocar o código de autorização por um <i>token</i> de acesso e usar o <i>token</i> para acessar os recursos da vítima, beneficiando o atacante que terá acesso a esses recursos. Nota: Autenticar o Provedor de Serviço não previne este ataque, pois é o Provedor de Serviço legítimo que obtém os <i>tokens</i> .
TP13	URL malicioso	Quando um Usuário faz <i>login</i> no Servidor PDS , ele fornece um URL com a sua identidade OpenID. Com isso, o Servidor PDS faz o <i>download</i> do URL e extrai o endereço do Provedor de Identidade do Usuário . Neste contexto, um atacante pode inserir um URL malicioso e levar a ataques nocivos, como exploração de scripts internos e ataques de negação de serviço.

¹⁸ Um *iframe* é um recurso usado para embutir outro documento HTML em uma subdivisão dentro do documento HTML atual.

TP14	Troca de sessão	Um atacante pode forçar um Usuário a fazer <i>login</i> em um Servidor PDS usando uma conta OpenID controlada pelo atacante. Isso acontece porque o protocolo OpenID não vincula as asserções positivas OpenID ao navegador do Usuário . Exemplo: Alice autentica-se em seu Provedor de Identidade , e em vez de submeter a asserção positiva para um <i>site</i> , ela engana Bob e passa a asserção para ele, que então autentica em um <i>site</i> usando a asserção de Alice. O Usuário pode não perceber que ele fez <i>login</i> como o atacante, permitindo que o atacante monitore suas atividades naquele <i>site</i> .
-------------	-----------------	--

Legenda: TP: Ameaça do Servidor PDS

Tabela 21. Categoria Servidor PDS: relação das ameaças com as vulnerabilidades e ativos

Ameaça		Vulnerabilidade Associada		Ativos Afetados	
Cód.	Descrição	Cód.	Descrição	Cód.	Descrição
TP01	Usuário involuntariamente concede escopo de acesso incorreto a um Provedor de Serviço	VP01	Interface de autorização do Servidor PDS é difícil de entender	AP01	Recursos dos Usuários
TP02	Captura de credenciais do Usuário por meio de redirecionadores abertos no Servidor PDS	VP02	O Servidor PDS faz redirecionamentos sem validar o parâmetro do URL de redirecionamento	AU01	Credenciais do Usuário
				AP01	Recursos dos Usuários
TP03	Captura de informações confidenciais por escuta do meio de transmissão	VP03	Meio de transmissão insuficientemente protegido	AS01	Credencial do Provedor de Serviço para acesso ao Servidor PDS
				AS02	<i>Tokens</i> de acesso de um Provedor de Serviço
				AS03	<i>Tokens</i> de atualização de um Provedor de Serviço
				AS04	Código de Autorização de um Provedor de Serviço
TP04	Captura de <i>tokens</i> , códigos de autorização e credenciais de Provedores de Serviço por meio de ataque ao banco de dados do Servidor PDS	VP04	Servidor PDS suscetível a ataques de injeção SQL (<i>SQL Injection</i>)	AP02	Credenciais dos Provedores de Serviço
				AP03	<i>Tokens</i> de acesso dos Provedores de Serviço
				AP04	<i>Tokens</i> de atualização dos Provedores de Serviço
				AP05	Código de Autorização dos Provedores de Serviço

TP05	Captura de credenciais do Provedor de Serviço por tentativa <i>online</i>	VP05	O Servidor PDS possui baixa entropia para geração de credenciais do Provedor de Serviço	AS01	Credencial do Provedor de Serviço para acesso no Servidor PDS
TP06	Captura de códigos de autorização e <i>tokens</i> por tentativa <i>online</i>	VP06	O Servidor PDS possui baixa entropia para geração de <i>tokens</i> e códigos de autorização	AS02	<i>Tokens</i> de acesso de um Provedor de Serviço
				AS03	<i>Tokens</i> de atualização de um Provedor de Serviço
				AS04	Código de Autorização de um Provedor de Serviço
TP07	Ataque de negação de serviço na criação dinâmica de segredos do Provedor de Serviço	VP05	O Servidor PDS possui baixa entropia para geração de credenciais do Provedor de Serviço	AP06	Disponibilidade do Serviço
TP08	Ataque de negação de serviço na criação de códigos de autorização e <i>tokens</i>	VP06	O Servidor PDS possui baixa entropia para geração de <i>tokens</i> e códigos de autorização	AP06	Disponibilidade do Serviço
TP09	Captura de código de autorização por Provedor de Serviço falsificado	VP07	Agente do Usuário não autentica o Provedor de Serviço na entrega do código de autorização	AS04	Código de Autorização de um Provedor de Serviço
TP10	Captura e reprodução de requisições válidas no acesso a recursos protegidos	VP08	Acesso aos recursos insuficientemente protegido no Servidor PDS	AP01	Recursos dos Usuários
TP11	Ataque de <i>clickjacking</i> contra a autorização do Usuário	VP09	Site do Servidor PDS permite que suas páginas sejam carregadas em um <i>iframe</i> por outros <i>sites</i>	AP05	Código de Autorização dos Provedores de Serviço
TP12	Personificação da sessão do Usuário	VP10	O Servidor PDS aceita que o Provedor de Serviço use um URL de redirecionamento que não é HTTPS	AP01	Recursos dos Usuários
				AU03	Sessão do Usuário
TP13	URL malicioso	VP11	Validação imprópria de dados de entrada	AP06	Disponibilidade do serviço do Servidor PDS
TP14	Troca de sessão	VP12	Verificação insuficiente da autenticidade dos dados na autenticação	AU02	Atividades do Usuário

Legenda: **TP:** Ameaça do Servidor PDS, **VP:** Vulnerabilidade do Servidor PDS, **AP:** Ativo do Servidor PDS

7.5.3.4 Ameaças e Vulnerabilidades da Categoria Provedor de Identidade

A Tabela 22 mostra as ameaças identificadas na categoria **Provedor de Identidade**, segundo um modelo que identifica cada ameaça por um código, um nome e uma descrição. Em seguida, a Tabela 23 apresenta as vulnerabilidades e ativos associados a essas ameaças.

Tabela 22. Categoria Provedor de Identidade: ameaças do Sistema PDS

Código	Ameaça	Descrição
TI01	Ataque de negação de serviço	Um <i>site</i> malicioso pode realizar ataques de negação de serviço contra um Provedor de Identidade solicitando repetidamente associações, autenticação ou verificação de uma assinatura. O ataque potencialmente mais grave é durante a fase de associação do protocolo OpenID, pois cada mensagem requer que o Provedor de Identidade execute uma exponenciação discreta.
TI02	Captura e reprodução de asserção de identidade	Uma vez que o Usuário é autenticado no Provedor de Identidade , o agente do Usuário é redirecionado de volta para o Provedor de Serviço com uma asserção de autenticação bem-sucedida. Um atacante pode interceptar essa asserção e reproduzi-la para autenticar-se como esse Usuário em outros <i>sites</i> .
TI03	Ataque de <i>Cross-Site Request Forgery</i> (CSRF)	O Provedor de Identidade não verifica suficientemente se uma requisição consistente foi intencionalmente fornecida por um Usuário autenticado. Exemplo: Depois que o Usuário fez <i>login</i> em sua conta OpenID (e portanto habilitou o <i>Single Sign-On</i>), uma página <i>web</i> maliciosa poderia usar <i>iframes</i> escondidos para contatar silenciosamente um <i>site</i> em que o Usuário tenha se registrado com sua conta OpenID e realizar uma solicitação em nome do Usuário .

Legenda: **TI:** Ameaça do Provedor de Identidade

Tabela 23. Categoria Provedor de Identidade: relação das ameaças com as vulnerabilidades e ativos

Ameaça		Vulnerabilidade Associada		Ativos Afetados	
Cód.	Descrição	Cód.	Descrição	Cód.	Descrição
TI01	Ataque de negação de serviço	VI01	Consumo não controlado de recursos	AI02	Disponibilidade do serviço do Provedor de Identidade
TI02	Captura e reprodução de asserção de identidade	VI02	Meio de transmissão insuficientemente protegido	AU03	Sessão do Usuário
TI03	Ataque de <i>Cross-Site Request Forgery</i> (CSRF)	VI03	Provedor de Identidade suscetível a ataque de CSRF	AU03	Sessão do Usuário

Legenda: **TI:** Ameaça do Provedor de Identidade, **VI:** Vulnerabilidade do Provedor de Identidade, **AI:** Ativo do Provedor de Identidade

7.6 Análise de Riscos

O gerenciamento de riscos é o processo que envolve identificar o risco, analisá-lo e tomar medidas para reduzi-lo a um nível aceitável. Esta seção faz a análise de riscos e a próxima seção desenvolve um Plano de Mitigação a partir dessa análise. Para a análise de riscos será adotado o método descrito pelo NIST (*National Institute of Standards and Technology*) em (STONEBURNER; GOGUEN; FERINGA, 2002).

Para decidir qual o nível de proteção adequado, é necessário analisar a probabilidade de que as ameaças se materializem e o impacto que o ataque terá se for bem sucedido (BISHOP, 2002). Para determinar a probabilidade de um ataque acontecer e ser bem sucedido, as ameaças ao sistema devem ser analisadas em conjunto com as vulnerabilidades potenciais e os controles já existentes para o sistema. O impacto refere-se à magnitude do dano que pode ser causado quando uma vulnerabilidade potencial é exercitada e depende de como os ativos serão afetados (por exemplo, quão crítico e sensível o componente ou dado afetado é). As tabelas *Tabela 24* e *Tabela 25* descrevem as definições dos níveis de probabilidade e de impacto, respectivamente.

Tabela 24. Definições do Nível de Probabilidade - P() (STONEBURNER; GOGUEN; FERINGA, 2002)

P() (Nível de Probabilidade)	Definição
Alto	O atacante é altamente motivado, suficientemente capaz, e os controles existentes para impedir que a vulnerabilidade seja explorada são ineficazes.
Médio	O atacante é motivado e capaz, mas existem controles que podem dificultar a exploração bem sucedida da vulnerabilidade.
Baixo	O atacante não tem motivação ou capacidade suficiente, ou existem controles para evitar, ou pelo menos dificultar significativamente que a vulnerabilidade seja explorada.

Tabela 25. Definições do Nível de Impacto – I() (STONEBURNER; GOGUEN; FERINGA, 2002)

I() (Nível de Impacto)	Definição
Alto	A exploração da vulnerabilidade pode resultar na perda altamente custosa dos principais ativos do sistema.
Médio	A exploração da vulnerabilidade pode resultar na perda custosa dos principais ativos do sistema.
Baixo	A exploração da vulnerabilidade pode resultar na perda de alguns ativos do sistema.

A determinação do nível de risco é derivada da multiplicação dos valores atribuídos à probabilidade e ao impacto de uma ameaça. A Tabela 26 mostra como a matriz de nível de risco pode ser determinada a partir da combinação dos valores possíveis para probabilidade e impacto. Posteriormente, a Tabela 27 define a escala de risco e quais as ações necessárias para cada nível de risco.

Tabela 26. Matriz de Nível de Risco – R() (STONEBURNER; GOGUEN; FERINGA, 2002)

		I() (Nível de Impacto)		
		Baixo (10)	Médio (50)	Alto (100)
PO (Nível de Probabilidade)	Alto (1,0)	Baixo $10 \times 1,0 = 10$	Médio $50 \times 1,0 = 50$	Alto $100 \times 0,1 = 100$
	Médio (0,5)	Baixo $10 \times 0,5 = 5$	Médio $50 \times 0,5 = 25$	Médio $100 \times 0,1 = 50$
	Baixo (0,1)	Baixo $10 \times 0,1 = 1$	Baixo $50 \times 0,1 = 5$	Baixo $100 \times 0,1 = 10$

Escala de Nível de Risco - Alto: $50 < R() \leq 100$; Médio: $10 < R() \leq 50$; Baixo: $1 \leq R() \leq 10$

Tabela 27. Escala de risco e ações necessárias (STONEBURNER; GOGUEN; FERINGA, 2002)

R() <i>(Nível de Risco)</i>	Definição
Alto	Existe forte necessidade de ações corretivas. Em alguns casos, um sistema existente pode continuar a operar, mas um plano de ação corretiva deve ser colocado em prática o mais rapidamente possível.
Médio	Ações corretivas são necessárias e um plano deve ser desenvolvido para incorporar essas ações dentro de um período razoável de tempo.
Baixo	O administrador do sistema deverá determinar se as ações corretivas são necessárias ou decidir aceitar o risco.

Com base nas definições anteriores, a Tabela 28 classifica os níveis de impacto e de probabilidade de cada uma das ameaças no sistema (Tabela 16, Tabela 18, Tabela 20 e Tabela 22) e deriva o nível de risco dessas ameaças. Essa análise de risco do sistema será utilizada na próxima seção para definição do Plano de Mitigação. Os Níveis de Impacto e de Probabilidade foram determinados com base nas definições da Tabela 24 e da Tabela 25, de acordo com a experiência da autora e com o auxílio de especialistas da área de segurança de informação.

Tabela 28. Análise de risco

Ameaça	Impacto	I()	P()	R()
TU01	Exposição das credenciais de um único Usuário . O atacante pode acessar todos os recursos do Usuário .	Alto	Baixo	Baixo
TU02	Exposição de um único código de autorização ou <i>token</i> relativo a um Usuário de um Provedor de Serviço .	Médio	Médio	Médio
TU03	Exposição das credenciais de um único Usuário . O atacante pode acessar todos os recursos do Usuário .	Alto	Baixo	Baixo
TU04	Exposição das credenciais de um único Usuário . O atacante pode acessar todos os recursos do Usuário .	Alto	Médio	Médio
TS01	Exposição das credenciais de um único Provedor de Serviço . O atacante pode autenticar-se no lugar do Provedor de Serviço para obter <i>tokens</i> ou fazer um ataque de <i>replay</i> usando <i>tokens</i> de atualização ou códigos de autorização roubados.	Alto	Baixo	Baixo
TS02	Exposição de todos os <i>tokens</i> de atualização e de acesso daquele Provedor de Serviço . O atacante pode acessar todos os recursos associados com o escopo dos <i>tokens</i> .	Alto	Baixo	Baixo
TS03	Exposição de um único <i>token</i> relativo a um Usuário de um Provedor de Serviço . O atacante pode acessar todos os recursos associados com o escopo do <i>token</i> .	Médio	Baixo	Baixo
TS04	Exposição de um único <i>token</i> relativo a um Usuário de um Provedor de Serviço . O atacante pode acessar todos os recursos associados com o escopo do <i>token</i> .	Médio	Baixo	Baixo
TS05	O Usuário acessa recursos em nome do atacante. O impacto efetivo depende do tipo de recurso acessado. Por exemplo, o Usuário pode salvar itens particulares na conta PDS de um atacante.	Alto	Baixo	Baixo
TS06	O atacante pode realizar ataques de URLs maliciosos. Exemplos: exploração de <i>scripts</i> internos: varredura de <i>site</i> de terceiros: http://target.com:1 , http://target.com:2 ; https://localhost/internal/auth?ip=1.1.1.1 ; e ataques de negação de serviço: http://www.site.com/largemovie.flv .	Alto	Médio	Médio
TS07	O Usuário pode não perceber que ele fez o <i>login</i> no lugar do atacante, permitindo que o atacante monitore suas atividades naquele <i>site</i> .	Médio	Baixo	Baixo
TP01	O Usuário pode conceder um escopo de acesso mais amplo do que gostaria para um Provedor de Serviço .	Alto	Médio	Médio
TP02	Um atacante pode utilizar a confiança do Usuário em seu Servidor PDS para fazer um ataque <i>phishing</i> e roubar as credenciais do Usuário .	Alto	Baixo	Baixo
TP03	Exposição das credenciais de um único Provedor de Serviço . O atacante pode autenticar-se no lugar do Provedor de Serviço para obter <i>tokens</i> ou fazer um ataque de captura e reprodução usando <i>tokens</i> de atualização ou códigos de autorização roubados. Exposição de um único <i>token</i> relativo a um Usuário de um Provedor de Serviço . O atacante pode acessar todos os recursos associados ao escopo do <i>token</i> . Exposição de um único código de autorização ou <i>token</i> relativo a um Usuário de um Provedor de Serviço .	Alto	Médio	Médio

TP04	Exposição de todos os códigos de autorização, <i>tokens</i> de acesso, <i>tokens</i> de atualização e credenciais de todos os Provedores de Serviço registrados naquele Servidor PDS . Isso permite que o atacante possa acessar recursos dos Usuários e atuar em nome de Provedores de Serviço legítimos para aquisição de <i>tokens</i> e recursos.	Alto	Alto	Alto
TP05	Exposição de um único par de credenciais (<i>identificador/secreto</i>) de um Provedor de Serviço . Isso permite que o atacante possa atuar em nome de um Provedor de Serviço para aquisição de <i>tokens</i> e recursos.	Alto	Baixo	Baixo
TP06	Exposição de um único código de autorização, <i>token</i> de acesso, ou <i>token</i> de atualização. Isso permite que o atacante possa atuar em nome de um Provedor de Serviço para aquisição de <i>tokens</i> e recursos.	Médio	Baixo	Baixo
TP07	Esgotamento de segredos disponíveis e consequente impossibilidade de registrar novos Provedores de Serviço .	Alto	Médio	Médio
TP08	Esgotamento de <i>tokens</i> ou códigos de autorização disponíveis e consequente impossibilidade de gerar novos <i>tokens</i> ou códigos de autorização.	Alto	Médio	Médio
TP09	Exposição de um único código de autorização e, potencialmente, aos <i>tokens</i> de acesso e de atualização correspondentes.	Médio	Baixo	Baixo
TP10	O atacante pode obter ou modificar os recursos do Usuário .	Alto	Baixo	Baixo
TP11	Um atacante pode conseguir que um Usuário autorize o acesso aos seus recursos sem que o Usuário perceba.	Alto	Baixo	Baixo
TP12	O atacante consegue acesso aos recursos do Usuário , pois eles estão associados à sua conta no <i>site</i> do Provedor de Serviço .	Alto	Baixo	Baixo
TP13	O atacante pode realizar ataques de URLs maliciosos. Exemplos: exploração de scripts internos: varredura de <i>site</i> de terceiros: http://target.com:1 , http://target.com:2 ; https://localhost/internal/auth?ip=1.1.1.1 ; e ataques de negação de serviço: http://www.site.com/largemovie.flv .	Alto	Médio	Médio
TP14	O Usuário pode não perceber que fez <i>login</i> como o atacante, permitindo que o atacante monitore suas atividades naquele <i>site</i> . Além disso, ele pode salvar recursos na conta PDS do atacante.	Médio	Baixo	Baixo
TI01	Indisponibilidade de acesso ao <i>site</i> do Provedor de Identidade . Esta ameaça é particularmente prejudicial, pois o Usuário depende do Provedor de Identidade para ter acesso a todos os outros componentes do sistema, como os <i>sites</i> do Servidor PDS e dos Provedores de Serviço , causando uma interrupção de disponibilidade também nesses outros serviços.	Alto	Médio	Médio
TI02	O atacante pode autenticar-se no lugar de um Usuário legítimo em outros <i>sites</i> .	Alto	Médio	Médio
TI03	O atacante pode autenticar-se no lugar de um Usuário legítimo em outros <i>sites</i> .	Alto	Baixo	Baixo

Legenda: **TU:** Ameaça do **Usuário**, **TS:** Ameaça do Provedor de Serviço, **TP:** Ameaça do Servidor PDS, **TI:** Ameaça do Provedor de Identidade, **I():** Nível de Impacto, **P():** Nível de Probabilidade, **R():** Nível de Risco

7.7 Plano de Mitigação

O Plano de Mitigação envolve priorizar, avaliar e implementar controles com base na análise de riscos. A Tabela 29 estabelece um conjunto de medidas para mitigação das ameaças do sistema. Uma mesma medida pode atuar na mitigação de uma ou mais ameaças, conforme mostra essa tabela. Essas medidas serão posteriormente analisadas para criação do Plano de Mitigação do sistema.

Tabela 29. Medidas para mitigação das ameaças

Código	Medidas Recomendadas	Ameaças Relacionadas
MU01	Os Usuários são responsáveis pela confiabilidade dos aplicativos instalados em suas máquinas. Dessa forma, deve-se tentar educá-los sobre os riscos de se instalar aplicativos não verificados ou originados de fontes não confiáveis.	TU01
MP01	O Servidor PDS deve usar cabeçalhos de autorização (<i>Authorization Headers</i>) ou parâmetros passados por método POST em vez de parâmetros de consulta no URL para envio de <i>tokens</i> e códigos de autorização.	TU02
MP02	O Servidor PDS deve associar cada código de autorização com o identificador do respectivo Provedor de Serviço e exigir que este se autentique no fluxo de troca do código de autorização. Dessa forma, é possível validar se o Provedor de Serviço requisitante é realmente o proprietário do código de autorização.	TU02, TP06 e TP09
MP03	O Servidor PDS deve emitir códigos de autorização de duração limitada e que só podem ser utilizados uma única vez.	TU02, TP03 e TP06
MP04	O Servidor PDS deve associar cada <i>token</i> com o identificador do respectivo Provedor de Serviço e exigir que este esteja autenticado nas requisições de apresentação de <i>tokens</i> . Dessa forma, é possível validar se o Provedor de Serviço requisitante é realmente o proprietário do <i>token</i> .	TU02, TS02, TS04, TP03, TP04, TP06 e TP10
MP05	O Servidor PDS deve reduzir o escopo e o tempo de expiração para <i>tokens</i> de acesso para reduzir o dano no caso de vazamentos.	TU02, TS02 e TP03
MP06	O Servidor PDS deve ter mecanismos para revogação de credenciais, <i>tokens</i> e códigos de autorização no caso de detecção de vazamentos.	TS01 e TS02
MP07	O Servidor PDS deve exigir que os Provedores de Serviço registrem apenas um URL de redirecionamento completo e sempre validar este URL registrado em relação ao URL de redirecionamento recebido nas requisições de acesso daquele Provedor de Serviço . Caso a verificação não seja bem sucedida, o Servidor PDS não deve redirecionar o agente do Usuário para esse URL.	TS03 e TP02
MP08	O Servidor PDS deve ter uma interface que explique de forma compreensível ao Usuário qual é o escopo (os recursos e as permissões) e o tempo de expiração da requisição de acesso do Provedor de Serviço .	TP01
MP09	Ao obter a autorização do Usuário , o Servidor PDS pode decidir se deve conceder o escopo total autorizado ou um subconjunto deste escopo com base no Provedor de Serviço que fez a requisição e em algum critério de segurança.	TP01

MP10	O Servidor PDS deve garantir que todas as transmissões de informações confidenciais são protegidas usando mecanismos da camada de transporte, como TLS e SSL.	TP03 e TP10
MP11	O Servidor PDS deve usar meios alternativos para autenticação dos Provedores de Serviço que não necessitam de transmissão das credenciais de texto em claro (por exemplo, autenticação <i>Digest</i>).	TP03
MP12	O Servidor PDS deve aplicar as melhores práticas de segurança para proteção de servidores <i>web</i> (por exemplo, para proteção de arquivos de configuração e bancos de dados).	TP04
MP13	O Servidor PDS deve armazenar apenas os <i>hashes</i> dos códigos de autorização, <i>tokens</i> de acesso, <i>tokens</i> de atualização e credenciais.	TP04
MP14	O Servidor PDS deve aplicar medidas padrão contra injeção de SQL, tais como: <ul style="list-style-type: none"> – Certificar que o código da aplicação do servidor está usando o mínimo privilégio possível para acesso ao banco de dados. – Evitar SQL dinâmico, utilizando SQL estático quando possível. Ao usar SQL dinâmico, parametrizar consultas usando variáveis (<i>bind variables</i>). – Filtrar e higienizar os parâmetros de entrada. Por exemplo, garantir que um parâmetro fornecido segue às regras de sintaxe da linguagem SQL. 	TP04
MP15	O Servidor PDS deve incluir uma alta entropia na criação de <i>tokens</i> , códigos de autorização e segredos. O código gerado deve ser construído a partir de uma sequência de números aleatórios ou pseudo-aleatórios criptograficamente forte (EASTLAKE; CROCKER; SCHILLER, 1994).	TP05, TP06, TP07 e TP08
MP16	O Servidor PDS deve bloquear as respectivas contas de Provedores de Serviço após certo número de tentativas falhas.	TP05
MP17	O Servidor PDS deve usar autenticação de dois fatores para autenticação do Provedor de Serviço .	TP05
MP18	O Servidor PDS deve incluir alguma etapa de verificação de novos Provedores de Serviço antes de registrá-los.	TP07
MP19	O Servidor PDS deve limitar o número de <i>tokens</i> de acesso concedidos por Usuário ou por Provedor de Serviço .	TP08
MP20	O Servidor PDS deve exigir que o URL de redirecionamento do Provedor de Serviço seja um <i>endpoint</i> HTTPS de forma que seja possível fazer uma ligação confiável entre o nome DNS de um Provedor de Serviço e a chave pública apresentada durante o estabelecimento da conexão.	TP09 e TP12
MP21	O Servidor PDS deve empregar requisições assinadas, juntamente com números aleatórios novos e <i>timestamps</i> , a fim de identificar requisições. Dessa forma, é possível detectar e recusar todas as requisições repetidas.	TP10
MP22	Para navegadores mais recentes, a prevenção de <i>iframes</i> pode ser executada do lado do Servidor PDS usando o cabeçalho X-FRAME-OPTION. Este cabeçalho possui dois valores possíveis: <i>deny</i> (a página não pode ser mostrada em um <i>iframe</i> por nenhum <i>site</i>) e <i>sameorigin</i> (a página só pode ser mostrada em um <i>iframe</i> pelo próprio <i>site</i>).	TP11
MP23	Para navegadores mais antigos, o Servidor PDS pode usar técnicas <i>JavaScript framebusting</i> , que são códigos em <i>JavaScript</i> que impedem o uso de <i>iframes</i> .	TP11
MP24	O Servidor PDS deve assumir que todas as entradas podem ser maliciosas e executar a validação adequada. As entradas aceitáveis devem ser filtradas usando uma <i>whitelist</i> . Além disso, uma <i>blacklist</i> pode ser útil para a detecção de potenciais ataques desse tipo.	TP13

MP25	O Servidor PDS deve rejeitar asserções positivas OpenID que não tiverem sido solicitadas previamente.	TP14
MP26	O Servidor PDS deve gerar um número aleatório novo (<i>fresh nonce</i>) no início do protocolo, guardar o <i>nonce</i> em um <i>cookie</i> do navegador <i>web</i> do Usuário e incluir o <i>nonce</i> no parâmetro <i>return_to</i> do protocolo OpenID. Ao receber uma asserção positiva do Provedor de Identidade do Usuário , o Servidor PDS deve validar se o <i>nonce</i> contido no parâmetro <i>return_to</i> corresponde ao <i>nonce</i> armazenado no <i>cookie</i> .	TP14
MS01	O Provedor de Serviço deve recarregar, sem parâmetros, a página de destino do URL de redirecionamento para limpar automaticamente o cache do navegador <i>web</i> do Usuário .	TU02
MS02	O Provedor de Serviço deve aplicar as melhores práticas de segurança para proteção de servidores <i>web</i> (por exemplo, para proteção de arquivos de configuração e bancos de dados).	TS01 e TS02
MS03	O Provedor de Serviço não deve permitir redirecionadores abertos.	TS03
MS04	O Provedor de Serviço deve autenticar o Servidor PDS usando autenticação HTTPS de forma a fazer uma ligação confiável entre o nome DNS de um Servidor PDS e a chave pública apresentada no estabelecimento da conexão.	TS04
MS05	Os Usuários podem ser educados a não seguir URLs não confiáveis.	TS05
MS06	O Provedor de Serviço deve usar o parâmetro <i>state</i> do protocolo OAuth (LODDERSTEDT; MCGLOIN; HUNT, 2012) para enviar um valor que identifica a sessão de um Usuário autenticado (por exemplo, um <i>hash</i> do <i>cookie</i> de sessão) ao fazer uma requisição de autorização ao Servidor PDS . Quando o Usuário concede a autorização, o Servidor PDS redireciona o agente do Usuário de volta para o Provedor de Serviço com uma resposta que inclui o valor original do parâmetro <i>state</i> . Assim, o Provedor de Serviço pode verificar se a sessão do Usuário que trouxe a autorização é igual ao identificador de sessão representado pelo parâmetro <i>state</i> que originou o pedido de autorização.	TS05
MS07	O Provedor de Serviço deve assumir que todas as entradas podem ser maliciosas e executar a validação adequada. As entradas aceitáveis devem ser filtradas usando uma <i>whitelist</i> . Além disso, uma <i>blacklist</i> pode ser útil para a detecção de potenciais ataques desse tipo.	TS06
MS08	O Provedor de Serviço deve rejeitar asserções positivas OpenID que não tiverem sido solicitadas previamente.	TS07
MS09	O Provedor de Serviço deve gerar um número aleatório novo (<i>fresh nonce</i>) no início do protocolo, guardar o <i>nonce</i> em um <i>cookie</i> do navegador <i>web</i> do Usuário e incluir o <i>nonce</i> no parâmetro <i>return_to</i> do protocolo OpenID. Ao receber uma asserção positiva do Provedor de Identidade do Usuário , o Provedor de Serviço deve validar se o <i>nonce</i> contido no parâmetro <i>return_to</i> corresponde ao <i>nonce</i> armazenado no <i>cookie</i> .	TS07
MI01	O Provedor de Identidade deve exigir o uso mecanismos de segurança da camada de transporte, como TLS e SSL, para todas as requisições nas quais a autenticidade do Provedor de Identidade é um problema.	TU03
MI02	Deve-se tentar educar os Usuários sobre os riscos que os ataques de <i>phishing</i> representam e fornecer mecanismos que tornam mais fácil para os Usuários confirmar a autenticidade de seus Provedores de Identidade .	TU03
MI03	O Provedor de Identidade deve usar autenticação de dois fatores para autenticar o Usuário (por exemplo, usar o SIM card do celular do Usuário).	TU04

MI04	O Provedor de Identidade deve garantir que todas as transmissões de credenciais do Usuário e asserções de identidade são protegidas usando mecanismos da camada de transporte, como TLS e SSL.	TU04 e TI02
MI05	O Provedor de Identidade deve limitar os recursos alocados para cada <i>site</i> e estabelecer quotas com base em alguns parâmetros, como o endereço IP (<i>Internet Protocol</i>) ou URL do <i>site</i> .	TI01
MI06	O Provedor de Identidade deve se certificar que a requisição foi entregue intencionalmente por um Usuário específico. Uma forma é colocar um parâmetro de formulário oculto contendo algum valor baseado em um segredo ou em dados de sessão. Desta forma, apenas uma requisição submetida por um determinado Usuário irá gerar uma submissão válida para aquele Usuário .	TI03

Legenda: **MU:** Medida do **Usuário**, **MS:** Medida do **Provedor de Serviço**, **MP:** Medida do **Servidor PDS**, **MI:** Medida do **Provedor de Identidade**, **TU:** Ameaça do **Usuário**, **TS:** Ameaça do **Provedor de Serviço**, **TP:** Ameaça do **Servidor PDS**, **TI:** Ameaça do **Provedor de Identidade**

Dado que geralmente não é possível eliminar todos os riscos, é necessário implementar os controles mais adequados para diminuir os riscos a um nível aceitável, com o mínimo impacto negativo possível sobre os ativos do sistema (STONEBURNER; GOGUEN; FERINGA, 2002). O gerenciamento do risco pode ser alcançado usando uma das opções abaixo (MYAGMAR; LEE; YURCIK, 2005):

- **Aceitar o risco:** o risco é tão baixo ou a mitigação desse risco é tão cara que compensa aceitá-lo.
- **Transferir o risco:** transferir o risco usando outras opções para compensar a perda, tais como seguros.
- **Eliminar o risco:** remover o componente do sistema ou recurso associado com o risco, se o componente ou recurso não é essencial ou não compensa o risco.
- **Mitigar o risco:** gerenciar o risco desenvolvendo um plano de mitigação que contém medidas para controlar os riscos do sistema.

No caso do **Sistema PDS**, escolheu-se mitigar cada uma das ameaças do sistema, pois o custo para mitigar as ameaças era aceitável e compensava aplicar as medidas de mitigação. Para seleção das medidas foram considerados o nível de risco da ameaça (conforme Tabela 28), o custo para implementar a medida, a eficiência da medida na mitigação do risco e se uma mesma medida poderia mitigar duas ou mais ameaças simultaneamente (por exemplo, a medida MP04 é capaz de mitigar várias ameaças). A Tabela 30 mostra o Plano de Mitigação desenvolvido para o **Sistema PDS** que define as medidas selecionadas para mitigação dos riscos para cada uma das ameaças do sistema.

Tabela 30. Plano de mitigação do sistema

Ameaça	Medidas Seleccionadas
TU01	MU01
TU02	MP01, MP02, MP03, MP04
TU03	MI01 e MI02
TU04	MI03 e MI04
TS01	MS02 e MP06
TS02	MS02, MP06 e MP04
TS03	MP07 e MS03
TS04	MS04 e MP04
TS05	MS05 e MS06
TS06	MS07
TS07	MS08
TP01	MP08
TP02	MP07
TP03	MP10, MP03 e MP04
TP04	MP12, MP13, MP14 e MP04
TP05	MP15 e MP16
TP06	MP15, MP02, MP03 e MP04
TP07	MP18 e MP15
TP08	MP19 e MP15
TP09	MP20 e MP02
TP10	MP10 e MP04
TP11	MP22 e MP23
TP12	MP20
TP13	MP24
TP14	MP25
TI01	MI05
TI02	MI04
TI03	MI06

Legenda: **MU:** Medida do **Usuário**, **MS:** Medida do Provedor de Serviço, **MP:** Medida do Servidor PDS, **MI:** Medida do Provedor de Identidade, **TU:** Ameaça do **Usuário**, **TS:** Ameaça do Provedor de Serviço, **TP:** Ameaça do Servidor PDS, **TI:** Ameaça do Provedor de Identidade

7.8 *Resumo do Capítulo*

Este capítulo realizou uma análise das vulnerabilidades e ameaças do **Sistema PDS**, proposto no Capítulo 5. Inicialmente, apresentaram-se os conceitos e o método utilizados na análise. Então, foram realizadas as quatro etapas da fase de modelagem de ameaças: caracterização do sistema, identificação de ativos, determinação de ameaças e determinação de vulnerabilidades. Com base nas ameaças e vulnerabilidades descobertas para o sistema, realizou-se a análise de riscos e, como resultado final, foi gerado um Plano de Mitigação para o **Sistema PDS**.

O próximo capítulo apresentará um protótipo do **Sistema PDS**, que implementa todas as medidas do plano de mitigação gerado neste capítulo de forma a atender aos requisitos de segurança do sistema (RNF-01 e RNF-06) definidos no Capítulo 3. O objetivo desse protótipo é mostrar a viabilidade técnica de implementação da arquitetura e facilitar o entendimento do sistema ao mostrar a interface com o **Usuário** e o fluxo de mensagens do protocolo de compartilhamento.

8 PROTÓTIPO DO SISTEMA PDS

Este capítulo descreve a implementação de um protótipo do **Sistema PDS** como prova de conceito da arquitetura detalhada no Capítulo 5. O objetivo do protótipo é validar se a arquitetura do **Sistema PDS** atende aos requisitos definidos no Capítulo 3 e se existem restrições técnicas para implementação dessa arquitetura. Além disso, o protótipo oferece uma visão global da interface com o **Usuário** para gerenciamento e compartilhamento de recursos e do fluxo de mensagens do protocolo do **Sistema PDS**.

8.1 Estrutura Geral do Protótipo

Esta seção descreve os componentes do protótipo, o ambiente de testes e algumas considerações sobre a implementação do protótipo.

8.1.1 Componentes

O protótipo cobre todos os componentes da arquitetura do **Sistema PDS** mostrados na Figura 20 do Capítulo 5, conforme descrição a seguir:

- **Usuário:** o sistema foi projetado para funcionar utilizando um navegador *web* padrão, logo não é necessário instalar nenhum aplicativo no dispositivo do **Usuário**.
- **Servidor PDS:** desenvolvido em linguagem de programação Java, o protótipo do **Servidor PDS** implementa a arquitetura básica conforme diagrama mostrado na Figura 21 do Capítulo 5. Este é o componente de maior relevância e complexidade de implementação, pois atua como **Servidor de Autorização**, gerenciando *tokens* e políticas, e **Servidor de Recursos**, gerenciando e protegendo os recursos do **Usuário**. Os módulos de extensão não foram implementados, pois, apesar de oferecerem funcionalidades interessantes aos **Usuários** e **Provedores de Serviço**, não estão diretamente ligados aos requisitos especificados.

- **Provedor de Serviço:** desenvolvido em linguagem de programação Java, representa a parte cliente do **Sistema PDS**.
- **Provedor de Identidade:** será utilizado o **Provedor de Identidade** do Ericsson Labs(<https://idm.labs.ericsson.net>). É importante destacar que a arquitetura do **Sistema PDS** suporta o uso de qualquer **Provedor de Identidade** que implemente o protocolo OpenID.

8.1.2 Exemplo de Testes

Como exemplo para apresentação do protótipo, foi escolhido o cenário na área de saúde, já discutido na seção 6.2. Neste exemplo, os componentes do **Sistema PDS**, apresentados na seção anterior, são representados por:

- **Usuário:** um paciente que possui uma conta no **Servidor PDS**, que gerencia e armazena os seus dados médicos, e uma conta no **Provedor de Identidade**, que comprova sua identidade.
- **Servidor PDS:** um serviço web em quem o **Usuário** confia para gerenciar e armazenar os seus dados médicos (<https://pds.larc.usp.br>).
- **Provedor de Serviço:** uma aplicação de um hospital hipotético: *Hospital Saint Peter* (<https://idp.larc.usp.br/saintpeterhospital>), que deseja obter acesso para salvar ou recuperar dados médicos da conta PDS do paciente.
- **Provedor de Identidade:** o **Provedor de Identidade** do Ericsson Labs (<https://idm.labs.ericsson.net>), em que o paciente já possui uma conta registrada.

Para fins de teste do protótipo, os componentes **Servidor PDS** e **Provedor de Serviço** foram implantados e receberam um IP público e um domínio DNS, conforme mostrado na Figura 28, que detalha o ambiente de testes e como os componentes do protótipo se comunicam entre si por meio da Internet.

Figura 28. Ambiente de testes

8.1.3 Implementação do Servidor PDS

Para lidar com a complexidade do protótipo, a aplicação do **Servidor PDS** foi conceitualmente dividida em camadas com base na arquitetura MVC (*Model-View-Controller*, Modelo-Visualização-Controlador), que possui as seguintes camadas (SILBERSCHATZ; KORTH; SUDARSHAN, 2010):

- **Camada de Visualização ou Apresentação:** define a apresentação de dados e trata da interação com o **Usuário**.
- **Camada de Modelo:** corresponde à camada de lógica de negócios, que fornece uma visão de alto nível de dados e ações sobre os dados.
- **Camada de Controle ou Controlador:** recebe eventos (ações do **Usuário**), executa ações no modelo, e retorna uma visão para o **Usuário**.

- **Camada de Acesso a Dados:** fornece a interface entre a camada de lógica de negócios e a base de dados subjacente. O protótipo usa uma linguagem orientada a objetos (Java) para codificar a camada de lógica de negócios e de modelo, enquanto o banco de dados é relacional (MySQL). Nesse caso, a camada de acesso a dados fornece, também, o mapeamento do modelo de dados orientado a objeto para o modelo relacional suportado pelo banco de dados.

A Figura 29 mostra as camadas da arquitetura MVC, juntamente com a sequência de passos executada para processar uma requisição de um **Usuário** por meio de um navegador *web* (SILBERSCHATZ; KORTH; SUDARSHAN, 2010). As setas dessa figura representam a sequência de execução dos passos. Quando a requisição é recebida pelo **Servidor PDS**, o **Controlador** envia uma solicitação para a **Camada de Modelo**. Essa camada processa o pedido, usando a lógica de negócios, que pode envolver a atualização dos objetos do modelo seguida pela criação de um objeto resultante. A **Camada de Modelo**, então, utiliza a **Camada de Acesso a Dados** para atualizar ou recuperar informações do banco de dados. Por fim, o objeto resultante criado pela **Camada de Modelo** é enviado, por meio do **Controlador**, para a **Camada de Visualização**, que cria uma página HTML a ser exibida no navegador *web*.

Figura 29. Arquitetura MVC (*Model-View-Controller*) (Adaptado de (SILBERSCHATZ; KORTH; SUDARSHAN, 2010))

No protótipo, foi utilizado o arcabouço Struts (<http://struts.apache.org>) para as **Camadas de Controle, Visualização e Modelo**, o arcabouço Hibernate (<http://www.hibernate.org>) para a **Camada de Acesso a Dados** e o banco de dados relacional MySQL (<http://www.mysql.com>) para o banco de dados.

8.1.4 Bibliotecas

Além das bibliotecas dos arcabouços Struts e Hibernate, foram utilizadas as bibliotecas Openid4java (<http://code.google.com/p/openid4java>), para implementação do protocolo OpenID, e Amber (<https://cwiki.apache.org/confluence/display/AMBER/Index>), para implementação do protocolo OAuth.

8.2 Interface com o Usuário

O **Usuário** pode interagir com o **Sistema PDS** de duas formas: quando ele, por conta própria, acessa o *site* do portal de gerenciamento do seu **Servidor PDS** ou quando ele está em um *site* de um **Provedor de Serviço** que o redireciona para a interface de compartilhamento do **Servidor PDS** para requisitar acesso (de leitura ou escrita) aos seus recursos protegidos. Esta seção descreve essas duas formas e apresenta a interface do protótipo do **Servidor PDS** com o **Usuário**.

8.2.1 Interface do Portal de Gerenciamento

No acesso direto ao portal de gerenciamento por meio do URL do **Servidor PDS** (<https://pds.larc.usp.br>), o **Usuário** realiza o *login* no sistema, conforme mostrado na Figura 30.

Figura 30. Página inicial do portal de gerenciamento

O **Servidor PDS**, então, pede ao **Usuário** que entre com o URL do seu **Provedor de Identidade** OpenID. Neste exemplo, o **Provedor de Identidade** utilizado pelo **Usuário** é o Provedor do Ericsson Labs <https://idm.labs.ericsson.net> (Figura 31).

Figura 31. Página de *login* do portal de gerenciamento

Neste momento, o navegador *web* do **Usuário** é redirecionado para a página de *login* do seu **Provedor de Identidade** para que ele se autentique, conforme mostra a Figura 32. O método de autenticação é determinado pelo **Provedor de Identidade** de acordo com as suas políticas de segurança. No exemplo da Figura 32, o **Provedor de Identidade**

autentica o **Usuário** usando nome de **Usuário** e senha, mas também poderia exigir outro modo de autenticação.

Figura 32. Login usando Provedor de Identidade OpenID

Se a autenticação do **Usuário** foi bem sucedida, o navegador *web* do **Usuário** é redirecionado de volta para o *site* do portal de gerenciamento do **Servidor PDS** juntamente com uma asserção OpenID que comprova a identidade do **Usuário**. Depois de validar essa asserção, o **Servidor PDS** efetua o *login* do **Usuário**, conforme mostrado na Figura 33.

Figura 33. Página de login bem sucedido

Uma vez feito o seu *login*, o **Usuário** pode acessar as opções de gerenciamento de sua conta no **Servidor PDS** (Figura 34). Nesta página, o **Usuário** pode escolher gerenciar seus recursos, suas autorizações ou seu histórico de atividades.

Figura 34. Interface de gerenciamento de conta do Usuário

Na página de gerenciamento de recursos (Figura 35), o **Usuário** pode visualizar informações sobre seus recursos (tais como, nome do arquivo, tipo, descrição, *tag*, criador e data de criação), fazer *download* de recursos, criar novos recursos ou remover recursos. Essa página permite ainda pesquisar os recursos usando dois filtros: por *tag* ou por tipo. As *tags* são palavras-chave que o **Usuário** escolhe no momento da inserção de recursos para facilitar futuras buscas. No exemplo da Figura 35, é possível ver dois tipos de recursos: <http://pds.larc.usp.br/ultrasound> (ultrasound_abdominal.jpg e ultrasound_breast.jpg) e <http://saintpeterhospital.com/operativenotes> (operative_note_cataract.txt e operative_note_artery_bypass.txt). Os arquivos do primeiro tipo foram criados pelo próprio **Usuário**, enquanto os arquivos do segundo tipo foram criados pelo site <http://idp.larc.usp.br/saintpeterhospital> na conta do **Usuário**.

The screenshot shows a web browser window with the URL `https://pds.larc.usp.br/server/manage_attributes_del.do?delete=true`. The page title is "Personal Data Sharing" with the subtitle "Secure Data Share on the Web". The navigation bar includes "Home", "About", and "Logout". The main content area is titled "My resources" and includes a link to "Create new Resource". Below this are filters for "type" and "tag", each with a "Filter" button. A table lists four resources, each with a checkbox, filename, type, description, tags, creator, creation date, and a download icon. A "Delete Selection" button is below the table, and a "Back" button is at the bottom left. A dashed blue box highlights the table, which is shown in detail below.

	Filename	Type	Description	Tags	Creator	Creation Date	Download
☐	operative_note_cataract.txt	http://saintpeterhospital.com/operativenotes			http://idp.larc.usp.br/saintpeterhospital/Redirect.do	03/Abr/2012	
☐	ultrasound_abdominal.jpg	http://pds.larc.usp.br/ultrasound	Abdominal Ultrasound Exam - 24 fev 2011	Hospital ABC ultrasound doctor john	Created by User	02/Abr/2012	
☐	operative_note_artery_bypass.txt	http://saintpeterhospital.com/operativenotes			http://idp.larc.usp.br/saintpeterhospital/Redirect.do	03/Abr/2012	
☐	ultrasound_breast.jpg	http://pds.larc.usp.br/ultrasound	Breast Ultrasound Exam - 13 july 2011	Hospital ABC ultrasound doctor john	Created by User	30/Mar/2012	

Filename	Type	Description	Tags	Creator
operative_note_cataract.txt	http://saintpeterhospital.com/operativenotes			http://idp.larc.usp.br/saintpeterhospital/Redirect.do
ultrasound_abdominal.jpg	http://pds.larc.usp.br/ultrasound	Abdominal Ultrasound Exam - 24 fev 2011	Hospital ABC ultrasound doctor john	Created by User
operative_note_artery_bypass.txt	http://saintpeterhospital.com/operativenotes			http://idp.larc.usp.br/saintpeterhospital/Redirect.do
ultrasound_breast.jpg	http://pds.larc.usp.br/ultrasound	Breast Ultrasound Exam - 13 july 2011	Hospital ABC ultrasound doctor john	Created by User

Figura 35. Interface de gerenciamento de recursos

Quando o **Usuário** escolhe criar novos recursos, ele acessa a página mostrada na Figura 36 e insere informações sobre tipo, descrição, *tags* e arquivo (o conteúdo efetivo) do novo recurso a ser criado na sua **conta PDS**.

Figura 36. Criação de recursos

Na página de gerenciamento de autorizações (Figura 37), o **Usuário** pode visualizar ou remover as autorizações existentes na sua conta. Essa página permite realizar pesquisa sobre as autorizações concedidas usando um filtro por **Provedor de Serviço** e habilitar ou não a visualização de autorizações expiradas. Cada autorização representa uma permissão de acesso a um ou mais recursos que o **Usuário** concedeu para um determinado **Provedor de Serviço** com uma determinada data de expiração. Quando o **Usuário** remove uma autorização, ele revoga a permissão de acesso que aquela autorização representava. O exemplo da Figura 37 mostra três autorizações: na primeira, o **Provedor de Serviço** <https://idp.larc.usp.br/client> possui uma autorização para acesso de leitura aos recursos `ultrasound_abdominal.jpg` e `ultrasound_breast.jpg` que foi criada em 2 de abril de 2012 e expira em 59 minutos; na segunda, o **Provedor de Serviço** <http://idp.larc.usp.br/saintpeterhospital> pode escrever recursos do tipo <http://idp.larc.usp.br/saintpeterhospital/operativenotes> e <http://pds.larc.usp.br/progressnotes> e a data de expiração para essa autorização é de 48 minutos; na terceira, o **Provedor de Serviço** <https://idp.larc.usp.br/saintpeterhospital> pode ler os recursos `radiograph_chest.jpg` e `ultrasound_breast.jpg` por dois dias.

Personal Data Sharing
Secure Data Share on the Web

Home About Logout

Hi, <https://idm.labs.ericsson.net/portal/id/cristina/>

Site Authorizations

[Show expired](#)

Filter by site:

	Site URL	Creation Date	Read	Write	Expires in	Shared Data
☐	https://idp.larc.usp.br/client/Redirect.do	02/Abr/2012	true	false	59 min(s)	[ultrasound_abdominal.jpg, ultrasound_breast.jpg]
☐	https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	false	true	48 min(s)	[http://pds.larc.usp.br/progressnotes , http://idp.larc.usp.br/hospitalsaintpeter/operativenotes]
☐	https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	true	false	2 day(s)	[radiograph_chest.jpg, ultrasound_breast.jpg]

Site URL	Creation Date	Read	Write	Expires in	Shared Data
https://idp.larc.usp.br/client/Redirect.do	02/Abr/2012	true	false	59 min(s)	[ultrasound_abdominal.jpg, ultrasound_breast.jpg]
https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	false	true	48 min(s)	[http://pds.larc.usp.br/progressnotes , http://idp.larc.usp.br/hospitalsaintpeter/operativenotes]
https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	true	false	2 day(s)	[radiograph_chest.jpg, ultrasound_breast.jpg]

Figura 37. Interface de gerenciamento de autorizações

Na página de gerenciamento de histórico (Figura 38), o **Usuário** pode visualizar as atividades realizadas na sua conta PDS. A partir dessa página é possível ainda pesquisar as atividades ou *logs* relacionadas à conta PDS usando um filtro por data de início e fim. No exemplo da Figura 38, pode-se observar atividades de criação e remoção de recursos, criação de autorizações e *login*.

Personal Data Sharing
Secure Data Share on the Web

Home About Logout

Hi, <https://dm.labs.ericsson.net/portal/id/cristina/>

Activity History

From / / to / /

Date	Time*	Action	Description
02/Abr/2012	12:40:26	Authorization created	Site: https://idp.larc.usp.br/client/Redirect.do / Scope: read
02/Abr/2012	12:41:37	User logged in	Method: OpenID
02/Abr/2012	12:41:56	Resource removed	Resource Filename: operative_note_appendicitis.txt
02/Abr/2012	12:41:56	Resource removed	Resource Filename: operative_note_cataract.txt
02/Abr/2012	12:42:30	User logged in	Method: OpenID
02/Abr/2012	12:44:42	Resource removed	Resource Filename: ultrasound_abdominal.jpg
02/Abr/2012	12:44:51	Resource created	Resource Description: Abdominal Ultrasound Exam - 24 fev 2011
02/Abr/2012	12:45:23	User logged in	Method: OpenID
02/Abr/2012	12:46:21	Authorization created	Site: https://idp.larc.usp.br/saintpeterhospital/Redirect.do / Scope: write
02/Abr/2012	12:46:38	User logged in	Method: OpenID

1 of 2 >>>

*Time zone: GMT

Date	Time*	Action	Description
02/Abr/2012	12:40:26	Authorization created	Site: https://idp.larc.usp.br/client/Redirect.do / Scope: read
02/Abr/2012	12:41:37	User logged in	Method: OpenID
02/Abr/2012	12:41:56	Resource removed	Resource Filename: operative_note_appendicitis.txt
02/Abr/2012	12:41:56	Resource removed	Resource Filename: operative_note_cataract.txt
02/Abr/2012	12:42:30	User logged in	Method: OpenID
02/Abr/2012	12:44:42	Resource removed	Resource Filename: ultrasound_abdominal.jpg
02/Abr/2012	12:44:51	Resource created	Resource Description: Abdominal Ultrasound Exam - 24 fev 2011
02/Abr/2012	12:45:23	User logged in	Method: OpenID
02/Abr/2012	12:46:21	Authorization created	Site: https://idp.larc.usp.br/saintpeterhospital/Redirect.do / Scope: write
02/Abr/2012	12:46:38	User logged in	Method: OpenID

Figura 38. Interface de gerenciamento de histórico

8.2.2 Interface de Compartilhamento

Esta seção mostra a interface com o **Usuário** no fluxo de compartilhamento de recursos para acessos de leitura e escrita, respectivamente. As figuras *Figura 39* e *Figura 40* mostram o estado inicial da conta do **Usuário** no **Servidor PDS**, de forma que seja possível identificar as mudanças que aconteceram após cada interação de leitura ou escrita. Neste exemplo o **Usuário** possui inicialmente três recursos (ultrasound_abdominal.jpg, ultrasound_breast.jpg e radiograph_chest.jpg) e nenhuma autorização.

The screenshot shows the 'Personal Data Sharing' web application. The main content area is titled 'My resources' and includes a 'Create new Resource' link and filter options. Below the filters is a table listing three resources. A dashed blue box highlights the table, and a larger table below the screenshot provides the full data for these resources.

Filename	Type	Description	Tags	Creator	Creation Date	Download
☐ radiograph_chest.jpg	http://pds.larc.usp.br/radiograph	Radiograph Exam - 16 jan 2012	radiograph doctor ann	Created by User	01/Abr/2012	
☐ ultrasound_abdominal.jpg	http://pds.larc.usp.br/ultrasound	Abdominal Ultrasound Exam - 24 fev 2011	Hospital ABC ultrasound doctor john	Created by User	30/Mar/2012	
☐ ultrasound_breast.jpg	http://pds.larc.usp.br/ultrasound	Breast Ultrasound Exam - 13 july 2011	Hospital ABC ultrasound doctor john	Created by User	30/Mar/2012	

Filename	Type	Description	Tags	Creator
radiograph_chest.jpg	http://pds.larc.usp.br/radiograph	Radiograph Exam - 16 jan 2012	radiograph doctor ann	Created by User
ultrasound_abdominal.jpg	http://pds.larc.usp.br/ultrasound	Abdominal Ultrasound Exam - 24 fev 2011	Hospital ABC ultrasound doctor john	Created by User
ultrasound_breast.jpg	http://pds.larc.usp.br/ultrasound	Breast Ultrasound Exam - 13 july 2011	Hospital ABC ultrasound doctor john	Created by User

Figura 39. Estado inicial dos recursos

Figura 40. Estado inicial das autorizações

8.2.2.1 Acesso de Leitura

Assim como o **Servidor PDS**, o *site* do **Provedor de Serviço** implementa também a autenticação do **Usuário** usando OpenID, conforme mostrado na Figura 41.

Figura 41. Página de login do *site* do Hospital Saint Peter

Depois que o **Usuário** insere o seu endereço OpenID, o navegador *web* do **Usuário** é redirecionado para a página de *login* do **Provedor de Identidade** para que ele se autentique, conforme mostra a Figura 42.

Figura 42. Login usando Provedor de Identidade OpenID

Se a autenticação do **Usuário** foi bem sucedida, o navegador *web* do **Usuário** é redirecionado de volta para o *site* do **Provedor de Serviço** juntamente com uma asserção OpenID que comprova a identidade do **Usuário**. Depois de validar essa asserção, o **Provedor de Serviço** efetua o *login* do **Usuário** no sistema (Figura 43).

Figura 43. Login bem sucedido

No exemplo da Figura 44, após fazer *login* o **Usuário**, o **Provedor de Serviço** pede que o **Usuário** entre com o endereço do seu **Servidor PDS** e inicia um pedido de acesso de leitura aos seguintes exames médicos do **Usuário**: ultra-sonografia (<http://pds.larc.usp.br/ultrasound>), radiografia (<http://pds.larc.usp.br/radiograph>) e tomografia computadorizada (<http://pds.larc.usp.br/ct>).

Figura 44. Requisição de acesso de leitura

O navegador *web* do **Usuário** é, então, redirecionado para o *site* do seu **Servidor PDS** onde o **Usuário** deverá informar qual é o seu endereço OpenID (Figura 45). Como o **Usuário** já efetuou o seu *login* anteriormente para acessar o *site* do **Provedor de Serviço**, os passos do protocolo OpenID são transparentes para esse **Usuário**.

Figura 45. Página de login do Servidor PDS

O site do **Servidor PDS** mostra então uma tela (Figura 46) para que o **Usuário** visualize a requisição de acesso feita pelo **Provedor de Serviço** e escolha as configurações de autorização. Neste momento, o **Usuário** verifica o *site* que fez a requisição (<https://idp.larc.usp.br/saintpeterhospital>) o escopo solicitado (escopo de leitura para os tipos <http://pds.larc.usp.br/ultrasound> e <http://pds.larc.usp.br/radiograph>) e quais os arquivos que ele possui para os tipos solicitados (ultrasound_abdominal.jpg, ultrasound_breast.jpg e radiograph_chest.jpg). Apesar do **Provedor de Serviço** ter requisitado arquivos do tipo <http://pds.larc.usp.br/ct>, a interface não mostra a requisição para esse tipo, pois o **Usuário** não possui nenhum arquivo deste tipo na sua conta PDS.

Figura 46. Requisição de autorização para acesso de leitura

A Figura 47 mostra as opções escolhidas para o **Usuário** neste exemplo: acesso de leitura para os arquivos `ultrasound_breast.jpg` e `radiograph_chest.jpg` com um tempo de expiração de 3 dias. A interface permite, também, que o **Usuário** visualize as informações sobre cada um dos recursos e faça *download* do arquivo, se necessário.

Figura 47. Autorização do Usuário para acesso de leitura

A partir da autorização do **Usuário**, o **Servidor PDS** gera um código de autorização e redireciona o **Usuário** de volta para o *site* do **Provedor de Serviço**. As próximas etapas

do protocolo (troca de *token* e entrega do recurso) são transparentes ao **Usuário**, pois ocorrem por meio de troca de mensagens diretamente entre o **Provedor de Serviço** e o **Servidor PDS**. Ao final, o **Usuário** irá visualizar uma tela de acesso de leitura bem sucedido como mostra a Figura 48. Neste exemplo, o **Provedor de Serviço** conseguiu ler os arquivos `ultrasound_breast.jpg` e `radiograph_chest.jpg` de acordo com a autorização concedida pelo **Usuário**, mas não recebeu nenhum arquivo do tipo `http://pds.larc.usp.br/ct`.

Figura 48. Acesso de leitura bem sucedido

Neste momento, se o **Usuário** fizer *login* no portal de gerenciamento do **Servidor PDS**, poderá verificar que a autorização que ele concedeu foi criada na sua conta de acordo com as configurações que ele escolheu, como mostra a Figura 49.

Figura 49. Autorização para acesso de leitura

8.2.2.2 Acesso de Escrita

Nesta seção, considera-se que o mesmo **Provedor de Serviço** da seção anterior faz uma requisição de acesso de escrita na conta do **Servidor PDS** do **Usuário**. Para fins de simplificação, considera-se que a sessão do **Usuário** no navegador *web* é a mesma da seção 8.2.2.1 (acesso de leitura) e, portanto, o **Usuário** já fez *login* no **Provedor de Identidade**, no **Provedor de Serviço** e no **Servidor PDS**.

No exemplo da Figura 50, o **Provedor de Serviço** pede que o **Usuário** entre com o endereço do seu **Servidor PDS** e inicia um pedido de acesso de escrita para os seguintes tipos de recursos: <http://idp.larc.usp.br/hospitalsaintpeter/operativenotes> e <http://pds.larc.usp.br/progressnotes>.

Figura 50. Requisição de acesso de escrita

O navegador *web* do **Usuário** é, então, redirecionado para o *site* do seu **Servidor PDS**, que mostra uma tela (Figura 51) para que o **Usuário** visualize a requisição feita pelo **Provedor de Serviço** e escolha as configurações de autorização. Neste momento, o **Usuário** verifica que o *site* <https://idp.larc.usp.br/saintpeterhospital> está requisitando acesso de escrita para os tipos <http://idp.larc.usp.br/hospitalsaintpeter/operativenotes> e <http://pds.larc.usp.br/progressnotes>. No exemplo da Figura 51, o **Usuário** concedeu acesso de escrita para os dois tipos requeridos com um tempo de expiração de 1 hora.

Firefox

Personal Data Sharing

usp.br https://pds.larc.usp.br/server//authorize.do?scope=http%3A%2F%2Fidp.larc.usp.br%2Fhospitalsaintpeter%2F

Home

Personal Data Sharing
Secure Data Share on the Web

Hi, https://idm.labs.e

Authorization Request

The site **https://idp.larc.usp.br/saintpeterhospital/Redirect.do** wants to **write** the following resources into your account.

1. Choose the type(s) of files the site can save into your account:

- ☒ http://idp.larc.usp.br/hospitalsaintpeter/operativenotes
- ☒ http://pds.larc.usp.br/progressnotes

2. Define the expiration time for this authorization:

☒ Expire in:

☐ Never expire

Figura 51. Requisição de autorização para acesso de escrita

A partir da autorização do **Usuário**, o **Servidor PDS** gera um código de autorização e redireciona o **Usuário** de volta para o *site* do **Provedor de Serviço**. As próximas etapas do protocolo (troca de *token* e salvamento do recurso) são transparentes ao **Usuário**, pois ocorrem por meio de troca de mensagens diretamente entre o **Provedor de Serviço** e o **Servidor PDS**. Ao final, o **Usuário** irá visualizar uma tela de acesso de escrita bem sucedido como mostra a Figura 52. Neste exemplo, o **Provedor de Serviço** conseguiu escrever os arquivos dos tipos autorizados pelo **Usuário**.

Figura 52. Acesso de escrita bem sucedido

Neste momento, se o **Usuário** fizer *login* no portal de gerenciamento do **Servidor PDS** poderá verificar que a autorização que ele concedeu foi criada na sua conta de acordo com as configurações que ele escolheu, como mostra a Figura 53.

Personal Data Sharing
Secure Data Share on the Web

Home About Logout

Hi, <https://idm.labs.ericsson.net/portal/id/cristina/>

Site Authorizations

[Show expired](#)

Filter by site:

	Site URL	Creation Date	Read	Write	Expires in	Shared Data
☐	https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	true	false	2 day(s)	[ultrasound_breast.jpg, radiograph_chest.jpg]
☐	https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	false	true	58 min(s)	[http://pds.larc.usp.br/progressnotes , http://idp.larc.usp.br/hospitalsaintpeter/operativenotes]

Site URL	Creation Date	Read	Write	Expires in	Shared Data
https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	true	false	2 day(s)	[ultrasound_breast.jpg, radiograph_chest.jpg]
https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	false	true	58 min(s)	[http://pds.larc.usp.br/progressnotes , http://idp.larc.usp.br/hospitalsaintpeter/operativenotes]

Figura 53. Autorizações concedidas pelo Usuário

Além disso na opção de gerenciamento de recursos, o **Usuário** pode verificar que o **Provedor de Serviço** criou dois recursos do tipo <http://pds.larc.usp.br/progressnotes> e três recursos do tipo <http://saintpeterhospital.com/operativenotes> conforme permissão concedida, como mostra a Figura 54.

Firefox

Personal Data Sharing

usp.br https://pds.larc.usp.br/server/manage_attributes_del.do?delete=true

Home About Logout

Personal Data Sharing
Secure Data Share on the Web

Hi, https://dm.labs.ericsson.net/portal/id/cristina!

My resources

[Create new Resource](#)

Filter by type:

Filter by tag:

	Filename	Type	Description	Tags	Creator	Creation Date	Download
☐	operative_note_appendicitis.txt	http://saintpeterhospital.com/operativenotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	
☐	operative_note_cataract.txt	http://saintpeterhospital.com/operativenotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	
☐	ultrasound_abdominal.jpg	http://pds.larc.usp.br/ultrasound	Abdominal Ultrasound Exam - 24 fev 2011	Hospital ABC ultrasound doctor john	Created by User	30/Mar/2012	
☐	ultrasound_breast.jpg	http://pds.larc.usp.br/ultrasound	Breast Ultrasound Exam - 13 july 2011	Hospital ABC ultrasound doctor john	Created by User	30/Mar/2012	
☐	progress_note_oncology_1.txt	http://pds.larc.usp.br/progressnotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	
☐	progress_note_oncology_2.txt	http://pds.larc.usp.br/progressnotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	
☐	operative_note_artery_bypass.txt	http://saintpeterhospital.com/operativenotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do	02/Abr/2012	
☐	radiograph_chest.jpg	http://pds.larc.usp.br/radiograph	Radiograph Exam - 16 jan 2012	radiograph doctor ann	Created by User	01/Abr/2012	

Filename	Type	Description	Tags	Creator
operative_note_appendicitis.txt	http://saintpeterhospital.com/operativenotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do
operative_note_cataract.txt	http://saintpeterhospital.com/operativenotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do
ultrasound_abdominal.jpg	http://pds.larc.usp.br/ultrasound	Abdominal Ultrasound Exam - 24 fev 2011	Hospital ABC ultrasound doctor john	Created by User
ultrasound_breast.jpg	http://pds.larc.usp.br/ultrasound	Breast Ultrasound Exam - 13 july 2011	Hospital ABC ultrasound doctor john	Created by User
progress_note_oncology_1.txt	http://pds.larc.usp.br/progressnotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do
progress_note_oncology_2.txt	http://pds.larc.usp.br/progressnotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do
operative_note_artery_bypass.txt	http://saintpeterhospital.com/operativenotes			https://idp.larc.usp.br/saintpeterhospital/Redirect.do
radiograph_chest.jpg	http://pds.larc.usp.br/radiograph	Radiograph Exam - 16 jan 2012	radiograph doctor ann	Created by User

Figura 54. Criação de recursos pelo Provedor de Serviço

8.3 Registro do Provedor de Serviço

Para poder requisitar acesso a recursos protegidos do **Usuário**, o **Provedor de Serviço** deve se registrar previamente no **Servidor PDS** para obter as credenciais de autenticação, que contêm um identificador da aplicação e uma chave secreta. Esta seção mostra a interface em que o desenvolvedor da aplicação do **Provedor de Serviço** deve se registrar para obter essas credenciais, conforme previsto na especificação do sistema na seção 5.6.1.

A Figura 55 mostra a página principal de registro onde o desenvolvedor entra o nome, o URL, a descrição e o URL de redirecionamento (usado no protocolo OAuth) da sua aplicação.

The screenshot shows a web browser window with the address bar displaying `https://pds.larc.usp.br/server/register.do`. The page title is "Personal Data Sharing" with the tagline "Secure Data Share on the Web". The navigation menu includes "Home", "About", and "Login". The main form is titled "Registration" and contains the following fields:

Application Name:	<input type="text" value="Saint Peter Hospital"/>
Application URL:	<input type="text" value="https://idp.larc.usp.br/saintpeterhospital"/>
Application Description:	<input type="text" value="Saint Peter Hospital Application"/>
Application Redirect URI:	<input type="text" value="https://idp.larc.usp.br/saintpeterhospital/Redirect.do"/>

Below the fields is a button labeled "Register Client".

Figura 55. Página de registro

Após entrar com as informações de registro, o desenvolvedor precisa confirmar que realmente é o proprietário do domínio indicado criando um arquivo com nome aleatório na raiz do seu domínio, conforme indicado na Figura 56.

Figura 56. Confirmação de domínio

Quando o desenvolvedor confirma a criação do arquivo, o **Servidor PDS** tenta fazer *download* do arquivo de confirmação e, caso seja bem sucedido, cria a conta do **Provedor de Serviço** e entrega as credenciais de autenticação (Figura 57). A credencial é composta por um identificador e um segredo conforme especificado no protocolo OAuth (seção 4.2.3).

Figura 57. Registro bem-sucedido

8.4 Fluxo de mensagens

Esta seção mostra que o fluxo de mensagens do protótipo segue o protocolo para compartilhamento de recursos do **Sistema PDS** para acessos de leitura e escrita descrito na seção 5.6 (Tabela 13 e Tabela 14). Para captura do tráfego de rede, foi utilizado o analisador de protocolos de rede Wireshark (<http://www.wireshark.org>).

A primeira parte do fluxo de mensagens (mensagens M1 a M7), tanto para um acesso de leitura quanto para um acesso de escrita, acontece no navegador *web* do **Usuário**, enquanto as mensagens seguintes são trocadas diretamente entre o **Provedor de Serviço** e o **Servidor PDS**. Dessa forma, a captura das mensagens que ocorrem no navegador foi feita na máquina do **Usuário** e a captura das mensagens restantes foi feita no **Servidor PDS**. Para permitir a análise, capturou-se o tráfego em HTTP ao invés de HTTPS e suprimiu-se as mensagens de autenticação OpenID (mensagens M1 e M4).

8.4.1 Acesso de Leitura

A Figura 58 mostra o fluxo correspondente às mensagens **M2 a M7** (Tabela 13 da seção 5.6) para um acesso de leitura capturado com a ferramenta Wireshark na máquina do **Usuário**.

No.	Time	Source	Destination	Protocol	Info
27	4.583458	172.20.5.64	143.107.111.26	HTTP	POST /saintpeterhospital/OAuthGetAuthorization.do
29	4.586119	143.107.111.26	172.20.5.64	HTTP	HTTP/1.1 302 Moved Temporarily
33	4.590381	172.20.5.64	143.107.111.40	HTTP	GET /server/authorize.do?scope=http%3A%2F%2Fpds.1
50	5.179897	143.107.111.40	172.20.5.64	HTTP	HTTP/1.1 200 OK (text/html)
87	14.430670	172.20.5.64	143.107.111.40	HTTP	POST /server/user_authorize_client.do HTTP/1.1 (a
97	15.351965	143.107.111.40	172.20.5.64	HTTP	HTTP/1.1 302 Moved Temporarily
98	15.354740	172.20.5.64	143.107.111.26	HTTP	GET /saintpeterhospital/Redirect.do?code=baf44d926
114	16.833416	143.107.111.26	172.20.5.64	HTTP	HTTP/1.1 200 OK (text/html)

Figura 58. Mensagens HTTP capturadas na máquina do Usuário para um acesso de leitura

A Tabela 31 detalha as mensagens mostradas na Figura 58 por meio da filtragem do arquivo de captura do Wireshark para ressaltar as informações mais importantes

relacionadas ao **protocolo PDS** referente ao compartilhamento de recursos para um acesso de leitura.

Tabela 31. Mensagens HTTP capturadas na máquina do Usuário para um acesso de leitura

#	Nº	Origem	Destino	Mensagem	Parâmetros
M1					
M2	27	172.20.5.64 (U)	143.107.111.26 (S)	POST /saintpeterhospital/OAuthGetAuthorization.do	pds_url = http://pds.larc.usp.br/server/
M3.1	29	143.107.111.26 (S)	172.20.5.64 (U)	HTTP/1.1 302 Moved Temporarily	scope =http://pds.larc.usp.br/ultrasound/pull http://pds.larc.usp.br/radiograph/pull http://pds.larc.usp.br/ct/pull response_type =code redirect_url =http://idp.larc.usp.br/saintpeterhospital/Redirect.do client_id =226f414074cab5f52b44bed71b4eaac
M3.2	33	172.20.5.64 (U)	143.107.111.40 (PDS)	GET /server/authorize.do	scope =http://pds.larc.usp.br/ultrasound/pull http://pds.larc.usp.br/radiograph/pull http://pds.larc.usp.br/ct/pull response_type =code redirect_url =http://idp.larc.usp.br/saintpeterhospital/Redirect.do client_id =226f414074cab5f52b44bed71b4eaac
M4					
M5	50	143.107.111.40 (PDS)	172.20.5.64 (U)	HTTP/1.1 200 OK	(text/html) "The site http://idp.larc.usp.br/saintpeterhospital/Redirect.do wants to read the following resources from your account. ..."
M6	87	172.20.5.64 (U)	143.107.111.40 (PDS)	POST /server/user_authorize_client.do	attributeIds =0febefa8367eb7aa01367ec4d2030007 attributeIds =0febefa8367e6f2c01367eb04209000b expirationValue =03 expirationUnit =days redirectUri =http://idp.larc.usp.br/saintpeterhospital/Redirect.do scope =pull
M7.1	97	143.107.111.40 (PDS)	172.20.5.64 (U)	HTTP/1.1 302 Moved Temporarily	code =baf44d926ca2808c17b87e1922345785
M7.2	98	172.20.5.64 (U)	143.107.111.26 (S)	GET /saintpeterhospital/Redirect.do	code =baf44d926ca2808c17b87e1922345785

A Figura 59 mostra o fluxo para um acesso de leitura correspondente às mensagens **M8** a **M13** (Tabela 13 da seção 5.6) para um acesso de leitura capturado com a ferramenta Wireshark na máquina do **Servidor PDS**.

read_pds.pcap - Wireshark

Filter: (ip.addr==143.107.111.40 || ip.addr==143.107.111.26) && http

No.	Time	Source	Destination	Protocol	Info
1184	30.274741	143.107.111.26	143.107.111.40	HTTP	POST /server//token.do?client_secret=13f3fe3a1365e4
1242	31.036268	143.107.111.40	143.107.111.26	HTTP	HTTP/1.1 200 OK (text/html)
1245	31.038483	143.107.111.26	143.107.111.40	HTTP	GET /server//resource.do?scope=http://pds.larc.usp
1260	31.293205	143.107.111.40	143.107.111.26	HTTP	HTTP/1.1 200 OK (text/html)
1262	31.296191	143.107.111.26	143.107.111.40	HTTP	GET /server/deliver_resource.do?oauth_resource_cod
1348	31.589535	143.107.111.40	143.107.111.26	HTTP	HTTP/1.1 200 OK (application/octet-stream)
1349	31.590902	143.107.111.26	143.107.111.40	HTTP	GET /server/deliver_resource.do?oauth_resource_cod
1403	31.746515	143.107.111.40	143.107.111.26	HTTP	HTTP/1.1 200 OK (application/octet-stream)

File: "C:\Users\cdominic\Dropbox\Dissertac..." Packets: 1608 Displayed: 8 Marked: 0 Ignored: 6

Figura 59. Mensagens HTTP capturadas no Servidor PDS para um acesso de leitura

A Tabela 32 detalha as mensagens mostradas na Figura 59 por meio da filtragem do arquivo de captura do Wireshark para ressaltar as informações mais importantes relacionadas ao **protocolo PDS** referente ao compartilhamento de recursos para um acesso de leitura.

Tabela 32. Mensagens HTTP capturadas no Servidor PDS para um acesso de leitura

#	Nº	Origem	Destino	Mensagem	Parâmetros
M8	1184	143.107.111.26 (S)	143.107.111.40 (PDS)	POST /server/token.do	client_secret =13f3fe3a1365ecb996d7546e1e3c69c grant_type =authorization_code redirect_url =http://idp.larc.usp.br/saintpeterhospital/Redirect.do code =baf44d926ca2808c17b87e1922345785 client_id =226f414074cab5f52b44bed71b4eaac
M9	1242	143.107.111.40 (PDS)	143.107.111.26 (S)	HTTP/1.1 200 OK	(text/html) { "expires_in" :"3600", "token_type" :"bearer", "refresh_token" :"a94797acd076d99ac6709023d597c75e", "access_token" :"417094e66199b65e173b37f4ee15ad6d"}
M10	1245	143.107.111.26 (S)	143.107.111.40 (PDS)	GET /server/resource.do	scope =http://pds.larc.usp.br/ultrasound/pull http://pds.larc.usp.br/radiograph/pull http://pds.larc.usp.br/ct/pull access_token =417094e66199b65e173b37f4ee15ad6d
M11	1260	143.107.111.40 (PDS)	143.107.111.26 (S)	HTTP/1.1 200 OK	(text/html) { "resources" : [{ "value":"http://pds.larc.usp.br/ultrasound", "attributes": [{ "url":"http://pds.larc.usp.br/server/deliver_resource.do?oauth_resource_code\u003d0febefa8367ec86401367ed274bc000c" }] }, { "value":"http://pds.larc.usp.br/radiograph", "attributes": [{ "url":"http://pds.larc.usp.br/server/deliver_resource.do?oauth_resource_code\u003d0febefa8367ec86401367ed275a7000d" }] }, { "value":"http://pds.larc.usp.br/ct", "attributes": [] }] }
M12	1262	143.107.111.26 (S)	143.107.111.40 (PDS)	GET /server/deliver_resource.do	oauth_resource_code =0febefa8367ec86401367ed274bc000c
M13	1348	143.107.111.40 (PDS)	143.107.111.26 (S)	HTTP/1.1 200 OK	(application/octet-stream) Content-Disposition: attachment;filename= ultrasound_breast.jpg
M12	1349	143.107.111.26 (S)	143.107.111.40 (PDS)	GET /server/deliver_resource.do	oauth_resource_code =0febefa8367ec86401367ed275a7000d
M13	1403	143.107.111.40 (PDS)	143.107.111.26 (S)	HTTP/1.1 200 OK	(application/octet-stream) Content-Disposition: attachment;filename= radiograph_chest.jpg

8.4.2 Acesso de Escrita

A Figura 60 mostra o fluxo correspondente às mensagens **M2 a M7** (Tabela 14 da seção 5.6) para um acesso de escrita capturado com a ferramenta Wireshark na máquina do **Usuário**.

No.	Time	Source	Destination	Protocol	Info
701	17.143214	172.20.5.64	143.107.111.26	HTTP	POST /saintpeterhospital/OAuthGetAuthorization.do
703	17.146445	143.107.111.26	172.20.5.64	HTTP	HTTP/1.1 302 Moved Temporarily
707	17.169519	172.20.5.64	143.107.111.40	HTTP	GET /server//authorize.do?scope=http%3A%2F%2Fidp.1
722	18.206516	143.107.111.40	172.20.5.64	HTTP	HTTP/1.1 200 OK (text/html)
746	22.481616	172.20.5.64	143.107.111.40	HTTP	POST /server/user_authorize_client.do HTTP/1.1 (a
749	23.043525	143.107.111.40	172.20.5.64	HTTP	HTTP/1.1 302 Moved Temporarily
750	23.046319	172.20.5.64	143.107.111.26	HTTP	GET /saintpeterhospital/Redirect.do?code=34eb4de1d
777	25.921033	143.107.111.26	172.20.5.64	HTTP	HTTP/1.1 200 OK (text/html)

Figura 60. Mensagens HTTP capturadas na máquina do Usuário para um acesso de escrita

A Tabela 33 detalha as mensagens mostradas na Figura 60 por meio da filtragem do arquivo de captura do Wireshark para ressaltar as informações mais importantes relacionadas ao **protocolo PDS** para compartilhamento de recursos em um acesso de escrita.

Tabela 33. Mensagens HTTP capturadas na máquina do Usuário para um acesso de escrita

#	Nº	Origem	Destino	Mensagem	Parâmetros
M1					
M2	701	172.20.5.64 (U)	143.107.111.26 (S)	POST /saintpeterhospital/OAuthGetAuthorization.do	pds_url = http://pds.larc.usp.br/server/
M3.1	703	143.107.111.26 (S)	172.20.5.64 (U)	HTTP/1.1 302 Moved Temporarily	scope =http://idp.larc.usp.br/hospitalsaintpeter/operativenotes/push http://pds.larc.usp.br/progressnotes/push response_type =code redirect_url =http://idp.larc.usp.br/saintpeterhospital/Redirect.do client_id =226f414074cab5f52b44bed71b4eaac
M3.2	707	172.20.5.64 (U)	143.107.111.40 (PDS)	GET /server/authorize.do	scope =http://idp.larc.usp.br/hospitalsaintpeter/operativenotes/push http://pds.larc.usp.br/progressnotes/push response_type =code redirect_url =http://idp.larc.usp.br/saintpeterhospital/Redirect.do client_id =226f414074cab5f52b44bed71b4eaac
M4					
M5	722	143.107.111.40 (PDS)	172.20.5.64 (U)	HTTP/1.1 200 OK	(text/html) "The site http://idp.larc.usp.br/saintpeterhospital/Redirect.do wants to write the following resources into your account."
M6	746	172.20.5.64 (U)	143.107.111.40 (PDS)	POST /server/user_authorize_client.do	attributeSetIds =0febefa8367ec86401367f0ce d8c0010 attributeSetIds =0febefa8367ec86401367f0ce df90011 expirationValue =01 expirationUnit =hours redirectUri =http://idp.larc.usp.br/saintpeterhospital/Redirect.do scope =push clientId =226f414074cab5f52b44bed71b4eaac
M7.1	749	143.107.111.40 (PDS)	172.20.5.64 (U)	HTTP/1.1 302 Moved Temporarily	code =34eb4de1d8f1b4874f653b7c17714248
M7.2	750	172.20.5.64 (U)	143.107.111.26 (S)	GET /saintpeterhospital/Redirect.do	code =34eb4de1d8f1b4874f653b7c17714248

A Figura 61 mostra o fluxo para um acesso de escrita correspondente às mensagens **M8** a **M13** (Tabela 14 da seção 5.6) para um acesso de escrita capturado com a ferramenta Wireshark na máquina do **Servidor PDS**.

No.	Time	Source	Destination	Protocol	Info
1793	52.999922	143.107.111.26	143.107.111.40	HTTP	POST /server//token.do?client_secret=13f3fe3a1365ecb99
1845	53.851910	143.107.111.40	143.107.111.26	HTTP	HTTP/1.1 200 OK (text/html)
1869	53.959984	143.107.111.26	143.107.111.40	HTTP	POST /server//resource.do HTTP/1.1 (application/octet-stream)
1874	54.284860	143.107.111.40	143.107.111.26	HTTP	HTTP/1.1 200 OK (text/html)
1961	54.973903	143.107.111.26	143.107.111.40	HTTP	POST /server//resource.do HTTP/1.1 (application/octet-stream)
1968	55.518527	143.107.111.40	143.107.111.26	HTTP	HTTP/1.1 200 OK (text/html)

Figura 61. Mensagens HTTP capturadas no Servidor PDS para um acesso de escrita

A Tabela 34 detalha as mensagens mostradas na Figura 61 por meio da filtragem do arquivo de captura do Wireshark para ressaltar as informações mais importantes relacionadas ao **protocolo PDS** para compartilhamento de recursos em um acesso de escrita.

Tabela 34. Mensagens HTTP capturadas no Servidor PDS para um acesso de escrita

#	Nº	Origem	Destino	Mensagem	Parâmetros
M8	1793	143.107.111.26 (S)	143.107.111.40 (PDS)	POST /server/token.do	client_secret =13f3fe3a1365ecb996d7546e1e3c69c grant_type =authorization_code redirect_url =http://idp.larc.usp.br/saintpeterhospital/Redirect.do code =34eb4de1d8f1b4874f653b7c17714248 client_id =226f414074cab5f52b44bed71b4eaac
M9	1845	143.107.111.40 (PDS)	143.107.111.26 (S)	HTTP/1.1 200 OK	(text/html) { "expires_in" :"3600", "token_type" :"bearer", "refresh_token" :"b831c8885a8fd8d6a8522c86b925853", "access_token" :"6fc29a07f653ddba9d376534f2b76f8"}
M10	1869	143.107.111.26 (S)	143.107.111.40 (PDS)	GET /server/resource.do	(application/octet-stream) MIME Multipart Media Encapsulation, Type: multipart/form-data Content-Disposition =form-data name ="resource" filename ="operative_note_appendicitis.txt" Content-Disposition : form-data name ="scope" data ="http://saintpeterhospital.com/operative notes/push" Content-Disposition : form-data name =" access_token" data ="6fc29a07f653ddba9d376534f2b76f8"
M11	1874	143.107.111.40 (PDS)	143.107.111.26 (S)	HTTP/1.1 200 OK	-
M12	1961	143.107.111.26 (S)	143.107.111.40 (PDS)	GET /server/resource.do	(application/octet-stream) MIME Multipart Media Encapsulation, Type: multipart/form-data Content-Disposition =form-data name ="resource" filename ="progress_note_oncology_1.txt" Content-Disposition : form-data name ="scope" data =http://saintpeterhospital.com/progress notes/push Content-Disposition : form-data name =" access_token" data ="6fc29a07f653ddba9d376534f2b76f8"
M13	1968	143.107.111.40 (PDS)	143.107.111.26 (S)	HTTP/1.1 200 OK	-

8.5 Considerações sobre Segurança

O protótipo implementa todas as medidas do Plano de Mitigação desenvolvido no Capítulo 7 (Tabela 30) e atende, dessa forma, aos requisitos relacionados à proteção dos serviços de segurança (confidencialidade, integridade e disponibilidade) definidos no Capítulo 3 (RNF-01 e RNF-06).

8.6 Considerações sobre o Cumprimento dos Requisitos

Esta seção tem como objetivo mostrar que o protótipo representa uma implementação completa da arquitetura do **Sistema PDS**, definida no Capítulo 5, que atende aos requisitos especificados no Capítulo 3.

8.6.1 Requisitos de Armazenamento

A verificação do atendimento aos requisitos de armazenamento (definidos na seção 3.3.2) pode ser feita por meio de um mapeamento entre os requisitos e as entidades do banco de dados do protótipo, mostradas na Figura 62 a seguir.

Figura 62. Estrutura do banco de dados do Servidor PDS

A Tabela 35 faz o mapeamento entre os requisitos de armazenamento e as entidades do banco de dados da Figura 62. Como os requisitos foram definidos de forma independente de tecnologia, a representação mais adequada de um requisito de armazenamento pode envolver uma ou mais entidades do banco de dados dependendo da solução adotada para o sistema. No caso do requisito RA-03, por exemplo, foi mais adequado representar uma autorização como um *token* (entidade *token*), que contém um conjunto de autorizações a recursos individuais (entidade *authorization*) e um conjunto de autorizações a grupos de recursos (entidade *group_authorization*). Este foi, também, o caso do requisito RA-02, que foi mapeado por meio de duas entidades do banco de dados: *attribute_set*, que agrupa recursos para facilitar a autorização de grupos de recursos do mesmo tipo, e *attribute*, que representa as informações de um único recurso. Já no caso dos requisitos RA-01, RA-04 e RA-05, a relação foi direta com as entidades *user*, *activity_log* e *site*, respectivamente.

Tabela 35. Mapeamento entre requisitos de armazenamento e entidades do banco de dados

RA	Campo de RA	Campo equivalente no banco de dados
RA-01 <i>Conta do Usuário</i>	Identificador do Usuário	user.user_id
	Identificador do Provedor de Identidade	user.openid_identifier
	Data de criação da conta	user.creation_date
RA-02 <i>Recursos do Usuário</i>	Identificador do recurso	attribute.attribute_id
	Identificador do Usuário	attribute_set.user_id
	Tipo	attribute_set.type
	Descrição	attribute.description
	Criador	attribute.creator
	Caminho do diretório de armazenamento + Nome do arquivo	attribute.path
	Tamanho do Arquivo	attribute.size
	Arquivo	Armazenado no sistema de arquivos do servidor e apontado pelo campo attribute.path
	<i>Tags</i>	attribute.tags
	Data de criação	attribute.creation_date
RA-03 <i>Autorizações</i>	Identificador da autorização	token.token_id
	<i>Site</i> autorizado	token.site_id
	Recursos compartilhados	authorization.attribute_id e group_authorization.attribute_id
	Escopo de leitura	token.scope_pull
	Escopo de escrita	token.scope_push
	<i>Token</i>	token.access_token
	Tipo de <i>token</i>	token.type
	Data de expiração	token.expiration_date
	Data de criação	token.creation_date
RA-04 <i>Histórico</i>	Identificador do Usuário	activity_log.user_id
	Atividade	activity_log.activity
	Descrição	activity_log.description
	Data de criação	activity_log.date
RA-05 <i>Sites registrados</i>	URL	site.realm
	Descrição	site.description
	Chave	site.key
	Segredo	site.secret
	Data de expiração	site.expiration_date
	Data de criação	site.creation_date

8.6.2 Requisitos Funcionais e Interativos

A verificação do atendimento aos requisitos funcionais e interativos (definidos na seção 3.3.4 e 3.3.5) pode ser feita por meio da análise do fluxo de mensagens (seção 8.4) e da interface com o **Usuário** do protótipo (seções 8.2 e 8.3), como mostram a Tabela 36 e a Tabela 37, respectivamente.

Tabela 36. Cumprimento de requisitos funcionais

Requisito	Descrição
RF-01: Autenticar Usuário	A seção 8.2.1 mostrou por meio do fluxo de telas da interface com o Usuário (Figura 30 até Figura 33) que o protótipo implementa a funcionalidade de autenticação do Usuário .
RF-02: Consultar Recursos	A seção 8.2.1 mostrou por meio do fluxo de telas da interface com o Usuário (Figura 34 e Figura 35) que o protótipo implementa a funcionalidade de consulta de recursos.
RF-03: Criar Recursos	A seção 8.2.1 mostrou por meio de telas da interface com o Usuário (Figura 35 e Figura 36) que o protótipo implementa a funcionalidade de criação de recursos.
RF-04: Remover Recursos	A seção 8.2.1 mostrou por meio de uma tela da interface com o Usuário (Figura 35) que o protótipo implementa a funcionalidade de remoção de recursos.
RF-05: Consultar Histórico	A seção 8.2.1 mostrou por meio de telas da interface com o Usuário (Figura 34 e Figura 38) que o protótipo implementa a funcionalidade de consulta de histórico.
RF-06: Remover Atividades	A seção 8.2.1 mostrou por meio de uma tela da interface com o Usuário (Figura 38) que o protótipo implementa a funcionalidade de remoção de atividades.
RF-07: Consultar Autorizações	A seção 8.2.1 mostrou por meio de telas da interface com o Usuário (Figura 34 e Figura 37) que o protótipo implementa a funcionalidade de consulta de autorizações.
RF-08: Remover Autorizações	A seção 8.2.1 mostrou por meio de uma tela da interface com o Usuário (Figura 37) que o protótipo implementa a funcionalidade de remoção de autorizações.
RF-09: Requisitar Autorização	A seção 8.2.2 mostrou por meio do fluxo de telas da interface com o Usuário (Figura 44 até Figura 48 e Figura 50 até Figura 52) que o protótipo implementa a funcionalidade de requisição de autorização ao Usuário para acessos de leitura e escrita.
RF-10: Ler Recurso	A seção 8.4.1 mostrou por meio de uma captura de tráfego de rede no Wireshark (Figura 59 e Tabela 32) que o protótipo implementa a funcionalidade de ler recurso seguindo as mensagens do protocolo PDS especificado na seção 5.6.4.
RF-11: Escrever Recurso	A seção 8.4.2 mostrou por meio de uma captura de tráfego de rede no Wireshark (Figura 61 e Tabela 34) que o protótipo implementa a funcionalidade de escrever recurso seguindo as mensagens do protocolo PDS especificado na seção 5.6.4.
RF-12: Autenticar Provedor de Serviço	As seções 8.4.1 e 8.4.2 mostraram por meio de uma captura de tráfego de rede no Wireshark (Figura 59, Tabela 32, Figura 61 e Tabela 34) que implementa a funcionalidade de autenticação do Provedor de Serviço.
RF-13: Registrar Provedor de Serviço	A seção 8.3 mostrou por meio do fluxo de telas da interface com o Usuário (Figura 55, Figura 56 e Figura 57) que o protótipo implementa a funcionalidade de registro do Provedor de Serviço.

Tabela 37. Cumprimento de requisitos interativos

Requisito	Descrição
RI-01: Informação sobre os Recursos do Usuário	A seção 8.2.1 mostrou por meio da tela de gerenciamento de recursos da interface com o Usuário (Figura 35) que este requisito é cumprido.
RI-02: Informação sobre as Atividades	A seção 8.2.1 mostrou por meio da tela de gerenciamento de atividades da interface com o Usuário (Figura 38) que este requisito é cumprido.
RI-03: Informação sobre as Autorizações do Usuário	A seção 8.2.1 mostrou por meio da tela de gerenciamento de autorizações da interface com o Usuário (Figura 37) que este requisito é cumprido.
RI-04: Informação sobre uma Requisição de Autorização	A seção 8.2.2 mostrou por meio das telas de requisição de autorização da interface com o Usuário (Figura 46 e Figura 51) que este requisito é cumprido.

8.6.3 Requisitos Não Funcionais

A verificação do atendimento aos requisitos não funcionais (definidos na seção 3.3.6) é mostrada na Tabela 38.

Tabela 38. Cumprimento de requisitos não funcionais

Requisito	Descrição
RNF-01: Confidencialidade e integridade	Este requisito é cumprido pela aplicação do plano de mitigação que descreve as melhores práticas para proteção da confidencialidade e da integridade dos recursos do Usuário , conforme mostrado seção 8.5.
RNF-02: Compartilhamento de tipos de dados genéricos	Este requisito é cumprido, pois os recursos são tratados internamente pelo sistema como tipos de dados genéricos na forma de arquivos e os tipos de dados são representados univocamente na forma de URLs. Além disso, o Provedor de Serviço pode definir os seus próprios tipos de dados ou utilizar tipos de dados do sistema.
RNF-03: Autorização explícita do Usuário	Este requisito é cumprido, pois nenhuma autorização de acesso a recursos protegidos no sistema é concedida sem que o Usuário tenha autorizado explicitamente o acesso.
RNF-04: Uso de navegador <i>web</i> como agente do Usuário	Este requisito é cumprido, pois o Usuário não precisa instalar nenhuma aplicação em seu dispositivo, além do navegador <i>web</i> , para utilizar o sistema.
RNF-05: Uso de padrões abertos e amplamente adotados	Este requisito é cumprido pela adoção dos protocolos OpenID e OAuth e pela utilização do navegador <i>web</i> como agente do Usuário .
RNF-06: Disponibilidade	Este requisito é cumprido pela aplicação do plano de mitigação que descreve as melhores práticas para garantia da disponibilidade dos componentes do sistema, conforme mostrado seção 8.5.
RNF-07: Complexidade baixa para as aplicações <i>web</i>	Este requisito é cumprido, pois o Sistema PDS foi projetado de forma que as operações mais complexas relacionadas ao gerenciamento e compartilhamento de recursos são executadas pelo Servidor PDS , enquanto as aplicações <i>web</i> (Provedores de Serviço) realizam apenas operações de menor complexidade.

8.7 Resumo do Capítulo

Este capítulo apresentou o protótipo do **Sistema PDS** que foi implementado como prova de conceito da arquitetura proposta no Capítulo 5. Este protótipo permitiu demonstrar a viabilidade técnica de implementação da arquitetura do **Sistema PDS** e verificar o cumprimento dos requisitos especificados no Capítulo 3.

Em especial, o protótipo facilitou o entendimento do sistema, pois permitiu uma visualização geral da interface com **Usuário** e do fluxo de mensagens do protocolo do **Sistema PDS**. Além disso, o protótipo implementou o plano de mitigação desenvolvido

no Capítulo 7 de forma a garantir que o sistema é seguro para proteger os recursos do **Usuário**.

A seguir, o Capítulo 9 apresentará as conclusões e contribuições deste trabalho, além de discutir as possibilidades de trabalhos futuros.

9 CONSIDERAÇÕES FINAIS

Este capítulo apresenta as considerações finais deste trabalho, que propõe uma solução para o gerenciamento e compartilhamento de dados entre aplicações *web* por meio de uma abordagem centrada no **Usuário**. Inicialmente, é apresentada uma análise geral do sistema proposto, ressaltando as contribuições desta dissertação. Depois, são discutidas as possibilidades de continuidade deste trabalho como sugestões de trabalhos futuros. Por fim, apresentam-se as publicações realizadas durante o desenvolvimento desta dissertação.

9.1 Contribuições

Como apresentado no capítulo de introdução, este trabalho se propôs a investigar e responder às seguintes questões de pesquisa:

*“Como permitir que o **Usuário** gerencie seus dados na Web de forma centralizada?”*

*“Como permitir que o **Usuário** compartilhe seus dados com aplicações web de forma fácil e segura?”*

*“Como equilibrar a proteção dos dados do **Usuário** com a necessidade de permitir que uma **aplicação web** manipule os dados criados por outra aplicação?”*

De modo a responder essas perguntas, este trabalho especificou os requisitos necessários para a solução de um SGCD (Sistema de Gerenciamento e Compartilhamento de Dados) para *Web*, no Capítulo 3. A especificação utilizou um método específico para o ambiente *Web*, chamado NDT (*Navigational Development Techniques*), e definiu requisitos de armazenamento, funcionais, interativos e não funcionais segundo esse método. A especificação de requisitos de um SGCD para o ambiente *Web* é a primeira contribuição deste trabalho.

Visto que o estudo do estado da arte realizado não encontrou soluções que atendessem integralmente aos requisitos especificados, este trabalho propôs o protocolo e a arquitetura PDS (*Personal Data Sharing*), conforme mostrado no Capítulo 5. No Capítulo 8, foi descrito o protótipo e foram discutidas as características do **Sistema PDS** que permitem o cumprimento de todos os requisitos especificados. Dessa forma, este trabalho contribui e inova ao propor a solução de um SGCD que é adequada às necessidades da *Web* atual e responde às questões anteriormente apresentadas. É interessante ressaltar ainda que a utilização dos protocolos OpenID e OAuth torna o **Sistema PDS** interoperável com serviços já estabelecidos na Web que suportam tais protocolos.

Uma das principais funções do **Sistema PDS** é proteger os recursos do **Usuário**. Assim, para assegurar que o sistema toma as medidas de segurança necessárias, foi realizada uma análise que identificou as ameaças do sistema e gerou um plano de mitigação para essas ameaças, conforme mostrado no Capítulo 7. Essa análise de ameaças pode ser expandida e contribui para a análise de sistemas semelhantes que usam os protocolos OpenID e OAuth.

De forma a demonstrar a viabilidade técnica de implementação da arquitetura proposta sistema, foi implementado um protótipo como prova de conceito do sistema. O protótipo representa outra contribuição deste trabalho, pois pode ser utilizado tanto para demonstração do funcionamento do sistema como para verificar que a arquitetura atende aos requisitos especificados. O protótipo com suas funcionalidades foi apresentado no Capítulo 8.

Por fim, pode-se afirmar que este trabalho contribui para os projetos futuros na área de gerenciamento e compartilhamento de dados entre aplicações *web* com uma abordagem centrada no **Usuário**.

9.2 Considerações sobre o Sistema PDS

O **Sistema PDS** apresenta vários benefícios em relação às abordagens tradicionais, tanto para os **Usuários**, quanto para os **Provedores de Serviço**. Esta seção analisa esses benefícios e discute também algumas ressalvas sobre o sistema.

Para os **Usuários**, o **Sistema PDS** permite que eles gerenciem e o compartilhem seus dados na Web de forma segura. Assim, o **Usuário** tem uma visão centralizada de seus dados e de suas preferências de compartilhamento, mantendo o controle sobre os seus dados na Web. Além disso, a arquitetura PDS simplifica o processo de autenticação dos **Usuários**, pois deixa a autenticação a cargo de uma única entidade: o **Provedor de Identidade**. Em contrapartida, para usar o sistema, o **Usuário** precisa ter uma conta registrada em um **Provedor de Identidade**, que implemente o protocolo OpenID. A escolha do **Provedor de Identidade** é feita livremente pelo **Usuário** e depende da relação de confiança existente entre eles.

Do lado dos **Provedores de Serviço**, o **Sistema PDS** facilita o acesso aos dados do **Usuário** pelas aplicações *web*. Além disso, o sistema é flexível e atende aos requisitos de aplicações *web* com diferentes lógicas de negócio; por exemplo, para muitas aplicações pode ser melhor deixar que o **Servidor PDS** implemente os mecanismos de gerenciamento e armazenamento de recursos, enquanto para outras aplicações pode ser melhor usar o módulo **Indexador** do **Servidor PDS** para manter os dados do usuário armazenados localmente em seu serviço. Para ter acesso ao **Sistema PDS**, os **Provedores de Serviço** precisam se registrar previamente no **Servidor PDS** para receber as credenciais de autenticação estabelecidas pelo protocolo OAuth.

Além das funcionalidades básicas de gerenciamento e compartilhamento de dados, o **Servidor PDS** pode ser estendido por meio de módulos que oferecem funcionalidades adicionais para os **Usuários** e **Provedores de Serviço**. Os módulos de extensão são **Proxies**, **Renderizadores** e **Indexador**. Os **Proxies** facilitam a importação de dados do **Usuário** para o **Servidor PDS**. Os **Renderizadores** permitem que o **Usuário** visualize tipos de dados pré-definidos na interface de gerenciamento do **Servidor PDS**. O **Indexador** permite que o **Usuário** armazene seus dados de forma distribuída. Cada

Servidor PDS deve escolher quais módulos de extensão implementar de acordo com os recursos disponíveis para implementação do servidor.

Outra funcionalidade importante do **Sistema PDS** é o compartilhamento de tipos de dados genéricos e não apenas de tipos de dados pré-fixados. Para conseguir isso, o **Sistema PDS** representa os tipos de dados usando URLs. Essa representação adiciona flexibilidade ao **Sistema PDS**, pois permite que uma aplicação *web*, ou um consórcio de aplicações, defina os seus próprios tipos de dados. Contudo, é importante ressaltar que essa flexibilidade pode tornar o entendimento da interface com o **Usuário** mais difícil e exige um esforço extra para que as informações sejam apresentadas de forma clara para o **Usuário**.

Na arquitetura do **Sistema PDS**, podem existir diversos **Provedores de Identidade** e **Servidores PDS** e o **Usuário** pode escolher os provedores em que tiver maior confiança para gerenciar sua identidade e seus recursos na Internet. Essa arquitetura pode ser implantada de modo incremental, ou seja, uma aplicação *web* pode decidir suportá-la individualmente sem a coordenação com outras aplicações. Entretanto, o benefício real da arquitetura só é alcançado quando um grande número de aplicações na *Web* suportar o **Sistema PDS**, pois os **Usuários** poderão gerenciar todos os seus recursos de forma uniforme e centralizada e compartilhar seus dados entre aplicações *web* de maneira simples e segura.

9.3 *Trabalhos Futuros*

A área de compartilhamento e gerenciamento de recursos do **Usuário** na *Web* possui um grande potencial para a geração de pesquisas. Assim, este trabalho pode derivar diversos trabalhos futuros relacionados a adição, refinamento ou extensão de funcionalidades do **Sistema PDS**. Além disso, os conceitos utilizados pelo sistema podem ser aplicados em outras áreas direta ou indiretamente ligadas ao **Sistema PDS**, como, por exemplo, computação em nuvem. Uma lista de possíveis trabalhos futuros e direções para onde esta pesquisa pode evoluir é descrita a seguir:

- **Implementação dos módulos de extensão do Servidor PDS:** Conforme mostrado no Capítulo 8, esta versão do protótipo do **Servidor PDS** não implementa os módulos de extensão da arquitetura (*Proxies*, *Renderizadores* e *Indexador*) especificados na seção 5.4.1. Esses módulos oferecem funcionalidades interessantes aos **Usuários** e **Provedores de Serviço** (em especial, o módulo *Indexador* que permite o armazenamento de recursos distribuído) e, por isso, serão implementados na próxima versão do protótipo.
- **Integração de dados de dispositivos do Usuário com o Servidor PDS:** Atualmente, o **Usuário** pode utilizar o **Servidor PDS** para gerenciar e compartilhar os seus recursos na *Web*. Se os recursos que estão armazenados localmente em dispositivos do **Usuário** também pudessem ser acessados por meio do **Sistema PDS**, ele teria controle centralizado de todos os seus dados independente de estarem armazenados localmente ou na *Web*. Isso pode ser conseguido através da instalação de um aplicativo no dispositivo do **Usuário** que se comunica com o **Servidor PDS** de forma semelhante a realizada em (DAENEN *et al.*, 2010).
- **Web Semântica:** A *Web Semântica*, também chamada de *Web 3.0*, pode proporcionar um ambiente mais estruturado para representar e explorar a informação gerada pela *Web*, pois combina tecnologias *web* e teorias de representação do conhecimento (LASSILA; HENDLER, 2007). Dessa forma, o potencial da *Web Semântica* pode ser explorada pelo **Sistema PDS**, em particular, nas áreas de representação de metadados e busca de dados.
- **Compartilhamento de recursos entre Usuários:** O **Sistema PDS** não leva em consideração o compartilhamento de recursos entre **Usuários** diretamente, mas pode ser utilizado como infra-estrutura para implementação de tal funcionalidade. Além disso, o uso de identificadores OpenID facilita a tarefa de identificar **Usuários** entre domínios diferentes. Assim, o **Sistema PDS** pode ser estendido para suportar o compartilhamento de recursos entre **Usuários**.
- **Geração offline de políticas:** Atualmente, um **Provedor de Serviço** inicia uma requisição de acesso que é autorizada pelo **Usuário** no **Servidor PDS**. O sistema poderia ser estendido para aceitar que o **Usuário** crie políticas de autorização de

forma *offline*. A idéia é que o **Usuário** possa criar políticas no portal de gerenciamento usando uma interface pré-definida que depois converte as políticas geradas para alguma linguagem específica, como o XACML (*eXtensible Access Control Markup Language*) (OASIS, 2005), por exemplo. Dessa forma, ao receber uma requisição, o **Servidor PDS** pode, primeiro, checar se já existe uma política adequada ou requisitar autorização do **Usuário**. Além disso, novos métodos de controle de acesso podem ser incluídos no sistema para oferecer novas funcionalidades, como, por exemplo, criação de políticas para um determinado grupo de **Provedores de Serviço**.

- **Mecanismos de descoberta de recursos:** A forma como o **Provedor de Serviço** fica sabendo o endereço do **Servidor PDS** do **Usuário** pode ser melhorada com um processo de descoberta de maneira semelhante ao implementado pelo protocolo OpenID (OPENID FOUNDATION, 2007), utilizando-se XRDS (*eXtensible Resource Descriptor Sequence*) ou outro protocolo. Neste processo, o **Servidor PDS** também poderia incluir o anúncio dos tipos de recursos suportados ou previamente determinados.
- **Anonimato e privacidade:** Mecanismos semelhantes aos apresentados no protocolo NibbleID (SAKURAGUI, 2011) podem ser incluídos no **Sistema PDS** para proteger informações sobre a identidade do **Usuário** e, ao mesmo tempo, possibilitar a entrega de recursos do **Usuário** para um **Provedor de Serviço** na *Web*. Dentro de certas condições, as propriedades desse protocolo podem permitir que um **Usuário**, ao acessar um serviço, permaneça anônimo e não rastreável pelas entidades do sistema.
- **Computação em nuvem:** No paradigma de computação em nuvem, um conjunto de recursos computacionais escaláveis, compartilhados, e distribuídos é capaz de fornecer infra-estrutura, *software*, plataforma, rede e armazenamento de dados na forma de serviços na Internet (POKHAREL; PARK, 2009). Uma das principais preocupações deste ambiente é como autenticar e autorizar as diferentes partes que compõem o sistema e prover uma comunicação segura entre elas. Vários conceitos e mecanismos utilizados pelo **Sistema PDS** poderiam ser adaptados

para habilitar autorização centrada no **Usuário** para o ambiente de computação em nuvem.

- **Sistemas Federados:** Com o surgimento de vários **Servidores PDS**, que possuem diferentes relações de confiança entre si, é importante estudar como a arquitetura do **Sistema PDS** pode ser utilizada em Federações. Essa abordagem permite a interação e a cooperação entre **Servidores PDS** e pode ter impacto positivo na escalabilidade do **Sistema PDS**.

9.4 Publicações

Esta seção lista as publicações existentes ou artigos previstos da autora que estão relacionados diretamente ou indiretamente com o desenvolvimento desta dissertação.

- Artigos diretamente ligados à dissertação:

DOMINICINI, C.; SIMPLÍCIO, M.; SAKURAGUI, R.; CARVALHO, T. *et al.* **Threat Modeling an Identity Management System for Mobile Internet**. I2TS 2010, dez 2010.

DOMINICINI, C. *et al.* **A User Centric Approach to Data Management and Sharing in the Web**. (a ser submetido)

DOMINICINI, C. *et al.* **A Framework for Sharing Medical Data in Mobile Web**. (em desenvolvimento)

- Patente:

Method for privacy preserving authorization in Pervasive Environments (Patent number: P31750).

- Capítulo em livro:

BRITO CARVALHO, T. DE; MIERS, C.; DOMINICINI, C.; REDÍGOLO, F. **Key Issues on Future Internet**. In: TRONCO, T. (Ed.). *New Network Architectures. Studies in Computational Intelligence*. [S.l.]: Springer Berlin / Heidelberg, 2010. v. 297p. 221–236.

REFERÊNCIAS

- 3G AMERICAS. **Identity Management: Overview of Standards and Technologies for Mobile and Fixed Internet**. . [S.l: s.n.]. , jan 2009.
- ALEXA. **Alexa Top 500 Global Sites**. Disponível em: <<http://www.alexa.com/topsites>>. Acesso em: 18 abr. 2012.
- BALFANZ, D. **Usable access control for the World Wide Web**. 19th Annual Computer Security Applications Conference. **Proceedings...** Las Vegas, NV, USA: IEEE Computer Society. , 2003.
- BAUER, L.; SCHNEIDER, M. A.; FELTEN, E. W. **A general and flexible access-control system for the Web**. 11th USENIX Security Symposium. **Proceedings...** Berkeley, CA, USA: USENIX Association. , 2002.
- BERTINO, E.; TAKAHASHI, K. **Identity Management: Concepts, Technologies, and Systems**. Norwood, MA, USA: Artech House Publishers, 2010.
- BERTOCCI, V.; SERACK, G.; BAKER, C. **Understanding windows cardspace: an introduction to the concepts and challenges of digital identities**. First ed. [S.l.]: Addison-Wesley Professional, 2007.
- BHARGAV-SPANTZEL, A.; CAMENISCH, J.; GROSS, T.; SOMMER, D. User centricity: a taxonomy and open issues. **Journal of Computer Security**, v. 15, n. 5, p. 493–527, 2007.
- BHATTI, R.; BERTINO, E.; GHAFOR, A. **A trust-based context-aware access control model for Web-services**. IEEE International Conference on Web Services. **Proceedings...** Hingham, MA, USA: Kluwer Academic Publishers. , jul 2004.
- BISHOP, M. **Computer Security: Art and Science**. 1. ed. [S.l.]: Addison-Wesley Professional, 2002.
- BLAZE, M.; FEIGENBAUM, J.; IOANNIDIS, J.; KEROMYTIS, A. D. RFC 2704: The KeyNote trust-management system version 2. **IETF Network Working Group**, 1999.
- BOYD, D. M.; ELLISON, N. B. Social network sites: Definition, history, and scholarship. **Journal of Computer-Mediated Communication**, v. 13, n. 1, p. 210–230, 2007.
- BROWN, A.; WEIHL, B. **An update on Google Health and Google PowerMeter | Official Google Blog**. Disponível em: <<http://googleblog.blogspot.com.br/2011/06/update-on-google-health-and-google.html>>. Acesso em: 24 abr. 2012.
- BULL, J.; GONG, L.; SOLLINS, K. Towards security in an open systems federation.

Computer Security—ESORICS 92, p. 1–20, 1992.

BURR, W. E.; DODSON, D. F.; POLK, W. T. Electronic authentication guideline. **NIST special publication**, v. 800, p. 63, 2006.

CANTOR, S.; KEMP, I. J.; PHILPOTT, N. R.; MALER, E. Assertions and protocols for the oasis security assertion markup language. **OASIS Standard (March 2005)**, 2005.

CARDENAS, A. A.; ROOSTA, T.; SASTRY, S. Rethinking security properties, threat models, and the design space in sensor networks: A case study in SCADA systems. **Ad Hoc Networks**, v. 7, n. 8, p. 1434 - 1447, 2009.

CHADWICK, D.; SHAW, S. **OpenID study**. , JISC Final report. Edinburgo: EDINA, University of Edinburgh. , 2008.

CRITES, S.; HSU, F.; CHEN, H. **OMash: enabling secure web mashups via object abstractions**. 15th ACM conference on Computer and communications security. **Proceedings...**, CCS '08. New York, NY, USA: ACM. Disponível em: <<http://doi.acm.org/10.1145/1455770.1455784>>. , 2008.

DAENEN, K.; THEETEN, B.; VANDERFEESTEN, D.; VRANCKEN, B. *et al.* The personal internet. **Bell Lab. Tech. J.**, v. 15, p. 3–21, jun 2010.

DEMCHENKO, Y.; GOMMANS, L.; LAAT, C. DE; OUDENAARDE, B. **Web Services and Grid Security Vulnerabilities and Threats Analysis and Model**. 6th IEEE/ACM International Workshop on Grid Computing. **Proceedings...**, GRID '05. Washington, DC, USA: IEEE Computer Society. Disponível em: <<http://dx.doi.org/10.1109/GRID.2005.1542751>>. , 2005.

DHAMIJA, R.; DUSSEAU, L. The Seven Flaws of Identity Management: Usability and Security Challenges. **Security Privacy, IEEE**, v. 6, n. 2, p. 24 -29, abr 2008.

DOMINICINI, C.; SIMPLÍCIO, M.; SAKURAGUI, R.; CARVALHO, T. *et al.* **Threat Modeling an Identity Management System for Mobile Internet**. I2TS 2010. **Proceedings...** Rio de Janeiro, Brasil: [s.n.], dez 2010.

EASTLAKE, D. E.; CROCKER, S. D.; SCHILLER, J. I. RFC 1750: Randomness Recommendations for Security. **IETF Network Working Group**, 1994.

EL MALIKI, T.; SEIGNEUR, J.-M. **A Survey of User-centric Identity Management Technologies**. The International Conference on Emerging Security Information, Systems, and Technologies, 2007 (SecureWare 2007). **Proceedings...** Valencia, Spain: IEEE Computer Society. , out 2007.

ELLISON, C.; FRANTZ, B.; LAMPSON, B.; RIVEST, R. *et al.* **RFC 2693: SPKI certificate theory**. . [S.l.]: IETF. , 1999.

EMIG, C.; BRANDT, F.; ABECK, S.; BIERMANN, J.; KLARL, H. **An Access Control Metamodel for Web Service-Oriented Architecture**. International Conference on Software Engineering Advances, 2007 (ICSEA 2007). **Proceedings...** French Riviera, France: IEEE Computer Society. , ago 2007.

ESCALONA, M. J.; ARAGON, G. NDT. A Model-Driven Approach for Web Requirements. **Software Engineering, IEEE Transactions on**, v. 34, n. 3, p. 377 -390, jun 2008.

ESCALONA, M. J.; KOCH, N. Requirements engineering for web applications-a comparative study. **Journal of Web Engineering**, v. 2, p. 193–212, 2004.

ESCALONA, M. J.; TORRES, J.; MEJÍAS, M. Requirements capture workflow in global information systems. **Object-Oriented Information Systems**, p. 95–101, 2002.

FENG, X.; GUOYAN, L.; HAO, H.; LI, X. **Role-based access control system for Web services**. The Fourth International Conference on Computer and Information Technology, 2004 (CIT '04). **Proceedings...** Wuhan, China: IEEE Computer Society. , set 2004.

FIELDING, R.; GETTYS, J.; MOGUL, J.; FRYSTYK, H. *et al.* RFC 2616: Hypertext Transfer Protocol–HTTP/1.1. **IETF Network Working Group**, 1999.

FLORENCIO, D.; HERLEY, C. **A large-scale study of web password habits**. 16th International Conference on World Wide Web. **Proceedings...**, WWW '07. New York, NY, USA: ACM. Disponível em: <<http://doi.acm.org/10.1145/1242572.1242661>>. , 2007.

GEAMBASU, R.; CHEUNG, C.; MOSHCHUK, A.; GRIBBLE, S. D.; LEVY, H. M. **Organizing and sharing distributed personal web-service data**. 17th International Conference on World Wide Web. **Proceedings...**, WWW '08. New York, NY, USA: ACM. Disponível em: <<http://doi.acm.org/10.1145/1367497.1367599>>. , 2008.

GIOKAS, D. Canada Health Infoway-Towards a National Interoperable Electronic Health Record (EHR) Solution. **Studies in Health Technology and Informatics**, v. 115, p. 108, 2005.

GOMI, H. **Access Control Model and Design for Delegation Using Authorization Tokens**. IEEE Third International Conference on Privacy, Security, Risk and Trust (passat) and IEEE Third International Conference on Social Computing (socialcom). **Proceedings...** Boston, MA, USA: IEEE Computer Society. , out 2011.

GONG, L. **A secure identity-based capability system**. IEEE Symposium on Security and Privacy, 1989. **Proceedings...** Oakland, California, USA: IEEE Computer Society Press. , 1989.

HAMMER-LAHAV, E. HTTP Authentication: MAC Access Authentication, versão 01. **Network Working Group, Internet Draft, IETF**, 2012.

HAMMER-LAHAV, E. **Introducing OAuth 2.0**. Disponível em: <<http://hueniverse.com/2010/05/introducing-oauth-2-0/>>. Acesso em: 16 abr. 2011.

HAMMER-LAHAV, E. (ED.). The OAuth 1.0 Protocol. **Internet Engineering Task Force (IETF) RFC5849**, abr 2010.

HAMMER-LAHAV, E.; RECORDON, D.; HARDT, D. The OAuth 2.0 Authorization Protocol - Draft 25. **Network Working Group, Internet Draft, IETF**, 8 mar 2012.

HAYTON, R. **OASIS: An open architecture for secure interworking services**. University of Cambridge Computer Laboratory, New Museums Site. **Proceedings...** [S.l.: s.n.], 1996.

HOLTMANN, S.; NIEMI, V.; GINZBOURG, P.; LAITINEN, P.; ASOKAN, N. **Cellular Authentication for Mobile and Internet Services**. [S.l.]: Wiley, 2009.

HSU, F.; CHEN, H. **Secure file system services for web 2.0 applications**. 2009 ACM Workshop on Cloud computing security. **Proceedings...**, CCSW '09. New York, NY, USA: ACM. Disponível em: <<http://doi.acm.org/10.1145/1655008.1655011>>. , 2009.

HSU, Y.; LEE, D. **Authentication and authorization protocol security property analysis with trace inclusion transformation and online minimization**. 18th IEEE International Conference on Network Protocols (ICNP), 2010. **Proceedings...** Kyoto, Japan: IEEE Computer Society. , out 2010.

ISTEPANIAN, R. S. H.; JOVANOVIĆ, E.; ZHANG, Y. T. Guest Editorial Introduction to the Special Section on M-Health: Beyond Seamless Mobility and Global Wireless Health-Care Connectivity. **Information Technology in Biomedicine, IEEE Transactions on**, v. 8, n. 4, p. 405 -414, dez 2004.

IVES, B.; WALSH, K. R.; SCHNEIDER, H. The domino effect of password reuse. **Commun. ACM**, v. 47, p. 75-78, abr 2004.

JONES, M.; HARDT, D.; RECORDON, D. The OAuth 2.0 Authorization Protocol: Bearer Tokens, versão 18. **Network Working Group, Internet Draft, IETF**, mar 2012.

KISSEL, B. **OpenID 2009 Year in Review**. Disponível em: <<http://openid.net/2009/12/16/openid-2009-year-in-review/>>. Acesso em: 10 abr. 2012.

KLYNE, G.; CARROLL, J. J.; MCBRIDE, B. Resource description framework (RDF): Concepts and abstract syntax. **W3C recommendation**, v. 10, 2004.

KO, M. N.; CHEEK, G. P.; SHEHAB, M.; SANDHU, R. Social-Networks Connect Services. **Computer**, v. 43, n. 8, p. 37 -43, ago 2010.

KOSHUTANSKI, H. A survey on distributed access control systems for web business processes. **International Journal of Network Security (IJNS)**, 2009.

LASSILA, O.; HENDLER, J. Embracing“ Web 3.0”. **IEEE Internet Computing**, p. 90–93, 2007.

LEE, H.; JEUN, I.; CHUN, K.; SONG, J. **A New Anti-phishing Method in OpenID**. Second International Conference on Emerging Security Information, Systems and Technologies, 2008 (SECURWARE '08). **Proceedings...** Cap Esterel, France: IEEE Computer Society. , ago 2008.

LEVY, H. **Capability-based computer systems**. [S.l.]: Digital Press, 1984. v. 12

LI, M.; LUO, N. **Data sharing between web applications based on the request of user**. International Colloquium on Computing, Communication, Control, and Management, 2009 (CCCM 2009). **Proceedings...** Sanya, China: [s.n.]. , ago 2009.

LI, N.; MITCHELL, J. C.; WINSBOROUGH, W. H. **Design of a role-based trust-management framework**. IEEE Symposium on Security and Privacy, 2002. **Proceedings...** Berkeley, California, USA: IEEE Computer Society. , 2002.

LIBERTY ALLIANCE. **Liberty Alliance Project**. Disponível em: <<http://www.projectliberty.org>>. Acesso em: 9 abr. 2012.

LIU, L.; ZHANG, X.; YAN, G.; CHEN, S. **Exploitation and threat analysis of open mobile devices**. 5th ACM/IEEE Symposium on Architectures for Networking and Communications Systems. **Proceedings...**, ANCS '09. New York, NY, USA: ACM. Disponível em: <<http://doi.acm.org/10.1145/1882486.1882493>>. , 2009.

LODDERSTEDT, T.; MCGLOIN, M.; HUNT, P. OAuth 2.0 Threat Model and Security Considerations. **Web Authorization Protocol (oauth), Internet Draft, IETF**, 19 fev 2012.

MACHULAK, M. P.; MALER, E. L.; CATALANO, D.; MOORSEL, A. VAN. **User-managed access to web resources**. 6th ACM Workshop on Digital Identity Management. **Proceedings...**, DIM '10. New York, NY, USA: ACM. Disponível em: <<http://doi.acm.org/10.1145/1866855.1866865>>. , 2010.

MARTIN, R. A. Common Weakness Enumeration (CWE v2. 1). 2011. **The MITRE Corporation, National Cyber Security Division of the US Department of Homeland Security**, 2011.

MILES, R.; HAMILTON, K. **Learning UML 2.0**. [S.l.]: O'Reilly Media, Inc., 2006.

MORGAN, R.; CANTOR, S.; CARMODY, S.; HOEHN, W.; KLINGENSTEIN, K. Federated Security: The Shibboleth Approach. **Educause Quarterly**, v. 27, n. 4, p. 6, 2004.

MOSTARDA, M.; PALMISANO, D.; ZANI, F.; TRIPODI, S. **Towards an OpenID-based solution to the Social Network Interoperability problem**. W3C Workshop on the Future of Social Networking. **Proceedings...** [S.l.: s.n.], 2009.

MURUGESAN, S. Understanding Web 2.0. **IT Professional**, v. 9, n. 4, p. 34 -41, ago 2007.

MYAGMAR, S.; LEE, A.; YURCIK, W. **Threat Modeling as a Basis for Security Requirements**. ACM workshop on Storage Security and Survivability, 2005 (StorageSS '05). **Proceedings...** New York, NY, USA: ACM Press. Disponível em: <<http://citeseer.ist.psu.edu/myagmar05threat.html>>. , 2005.

NI, J.; LI, Z.; GAO, Z.; SUN, J. Threats Analysis and Prevention for Grid and Web Service Security. **Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, ACIS International Conference on**, v. 3, p. 526-531, 2007.

NICKULL, D.; HINCHCLIFFE, D. **Web 2.0 Architectures**. [S.l.]: Adobe Developer Library, 2009.

O REILLY, T. What is Web 2.0: Design patterns and business models for the next generation of software. **Communications and Strategies**, v. 65, p. 17, 2007.

OASIS. SAML 2.0 profile of XACML v2.0. **OASIS Standard**, fev 2005.

OH, H.-K.; JIN, S.-H. **The Security Limitations of SSO in OpenID**. 10th International Conference on Advanced Communication Technology, 2008 (ICACT 2008). **Proceedings...** Gangwon-Do, South Korea: IEEE Computer Society. , fev 2008.

OPENID FOUNDATION. **OpenID Authentication 2.0 - Final**. Disponível em: <http://openid.net/specs/openid-authentication-2_0.html>. Acesso em: 16 mar. 2011.

OWASP. **The Open Web Application Security Project (OWASP) - Category:Attack**. Disponível em: <<https://www.owasp.org/index.php/Category:Attack>>. Acesso em: 15 jun. 2012.

PASHALIDIS, A.; MITCHELL, C. A Taxonomy of Single Sign-On Systems. In: SAFAVINAINI, R.; SEBERRY, J. (Eds.). **Information Security and Privacy**. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer-Verlag, 2003. v. 2727p. 219-219.

POKHAREL, M.; PARK, J. S. **Cloud computing: future solution for e-governance**. 3rd International Conference on Theory and Practice of Electronic Governance. **Proceedings...**, ACM International Conference Proceeding Series. Bogota, Columbia: ACM. , 2009.

PRESSMAN, R. S. **Engenharia de software**. [S.l.]: McGraw-Hill, 2006.

RADATZ, J. IEEE standard glossary of software engineering terminology. **IEEE Std 610.12-1990**, v. 121990, 1990.

RECORDON, D.; REED, D. **OpenID 2.0: a platform for user-centric identity management**. Second ACM Workshop on Digital Identity Management. **Proceedings...** Alexandria, VA, USA: ACM. , 2006.

SAKURAGUI, R. **Gerenciamento de identidades com privacidade do usuário em ambiente Web**. São Paulo: Escola Politécnica, Universidade de São Paulo, 2011.

SANDHU, R. S.; COYNE, E. J.; FEINSTEIN, H. L.; YOUMAN, C. E. Role-based access control models. **Computer**, v. 29, n. 2, p. 38 -47, fev 1996.

SHIREY, R. RFC 2828: Internet security glossary. **IETF Network Working Group**, 2000.

SILBERSCHATZ, A.; KORTH, H.; SUDARSHAN, S. **Database System Concepts**. 6. ed. [S.l.]: McGraw-Hill Science/Engineering/Math, 2010.

SMETTERS, D. **Building secure mashups**. Workshop on Web 2.0 Security and Privacy, 2008 (W2SP'08). **Proceedings...** [S.l.: s.n.], 2008.

STANGO, A.; PRASAD, N. R.; KYRIAZANOS, D. M. **A Threat Analysis Methodology for Security Evaluation and Enhancement Planning**. 2009 Third International Conference on Emerging Security Information, Systems and Technologies. **Proceedings...**, SECURWARE '09. Washington, DC, USA: IEEE Computer Society. Disponível em: <<http://dx.doi.org/10.1109/SECURWARE.2009.47>>. , 2009.

STONEBURNER, G.; GOGUEN, A.; FERINGA, A. Risk management guide for information technology systems. **Nist special publication**, v. 800, n. 30, p. 800-30, 2002.

SUN, S.-T.; BEZNOSOV, KONSTANTIN. Open Problems in Web 2.0 User Content Sharing. In: CAMENISCH, JAN; KESDOGAN, D. (Eds.). **iNetSec 2009 – Open Research Problems in Network Security**. IFIP Advances in Information and Communication Technology. [S.l.]: Springer Boston, 2009. v. 309p. 37-51.

SUN, S.-T.; HAWKEY, K.; BEZNOSOV, K. **Secure Web 2.0 Content Sharing Beyond Walled Gardens**. Computer Security Applications Conference, 2009 (ACSAC '09). **Proceedings...** Honolulu, Hawaii: IEEE Computer Society. , dez 2009.

TACHAKRA, S.; WANG, X.; ISTEPANIAN, R. S. H.; SONG, Y. Mobile e-health: the unwired evolution of telemedicine. **Telemedicine Journal and E-health**, v. 9, n. 3, p. 247-257, 2003.

THE MITRE CORPORATION. **CAPEC-2000: Comprehensive CAPEC (Common Attack Pattern Enumeration and Classification) Dictionary (Release 1.7.1)**. Disponível em: <<http://capec.mitre.org/data/slices/2000.html>>. Acesso em: 15 jun. 2012.

TOOTOONCHIAN, A.; GOLLU, K. K.; SAROIU, S.; GANJALI, Y.; WOLMAN, A. **Lockr: social access control for web 2.0**. First Workshop on Online Social Networks. **Proceedings...**, WOSP '08. New York, NY, USA: ACM. Disponível em: <<http://doi.acm.org/10.1145/1397735.1397746>>. , 2008.

TSYRKLEVICH, E.; TSYRKLEVICH, V. Single Sign-On for the Internet: A Security Story. **BlackHat USA**, 2007.

VALDERAS, P.; PELECHANO, V. A Survey of Requirements Specification in Model-Driven Development of Web Applications. **ACM Trans. Web**, v. 5, n. 2, p. 10:1–10:51, maio 2011.

VERMA, D. C. **Principles of computer systems and network management**. New York: Springer-Verlag New York Inc, 2009.

WANG, H. J.; FAN, X.; HOWELL, J.; JACKSON, C. **Protection and communication abstractions for web browsers in MashupOS**. Twenty-first ACM SIGOPS symposium on Operating Systems Principles. **Proceedings...**, SOSP '07. New York, NY, USA: ACM. Disponível em: <<http://doi.acm.org/10.1145/1294261.1294263>>. , 2007.

WEEKS, S. **Understanding trust management systems**. IEEE Symposium on Security and Privacy, 2001 (S&P 2001). **Proceedings...** Oakland, California, USA: IEEE Computer Society. , 2001.

YAO, W. **Trust management for widely distributed systems**. Cambridge, UK: University of Cambridge, 2003.

APÊNDICE A – MODELOS COMPLETOS DO MÉTODO NTD PARA A ESPECIFICAÇÃO DE REQUISITOS DO SISTEMA

Este apêndice apresenta as tabelas detalhadas segundo os modelos definidos pelo método NTD para cada uma das seis atividades que fazem parte do processo de especificação de requisitos do sistema.

(Atividade 1) Definição dos Objetivos

Tabela 39. Objetivo 01 - Gerenciamento de contas dos Usuários

OBJ-01	Gerenciamento de Contas dos Usuários
Descrição	Gerenciar contas de Usuários (visualizar/criar/editar recursos, visualizar/criar/editar autorizações, fornecer histórico de atividades).

Tabela 40. Objetivo 02 - Autenticação de Usuários usando um Provedor de Identidade

OBJ-02	Autenticação de Usuários usando um Provedor de Identidade
Descrição	Usar um Sistema de Gerenciamento de Identidade para autenticar os Usuários .

Tabela 41. Objetivo 03 - Compartilhamento de recursos do Usuário com aplicações web

OBJ-03	Compartilhamento de recursos do Usuário com aplicações <i>web</i>
Descrição	Compartilhar recursos do Usuário com aplicações <i>web</i> (requisição de autorização ao Usuário , entrega de <i>tokens</i> , entrega de recursos).

Tabela 42. Objetivo 04 - Proteção dos recursos do Usuário

OBJ-04	Proteção dos recursos do Usuário
Descrição	O sistema deve ser seguro de forma a proteger os recursos do Usuário contra acesso não autorizado.

Tabela 43. Objetivo 05 - Implantação do sistema em larga escala na Internet

OBJ-05	Implantação do sistema em larga escala na Internet
Descrição	O sistema deve ser possível de ser implantado em larga escala na Internet.

(Atividade 2) Definição dos Requisitos de Armazenamento

Tabela 44. Requisito de Armazenamento 01 - Conta do Usuário

RA-01	Conta do Usuário	
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-02: Autenticação de Usuários usando um Provedor de Identidade	
Descrição	O sistema deve armazenar as informações sobre os Usuários .	
Dados específicos	Nome e descrição	Natureza
	Identificador do Usuário	<i>String</i>
	Identificador do Provedor de Identidade do Usuário	<i>String</i>
	Data de criação da conta	Data

Tabela 45. Requisito de Armazenamento 02 - Recursos do Usuário

RA-02	Recursos do Usuário	
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário	
Descrição	O sistema deve armazenar as informações sobre os recursos dos Usuários .	
Dados específicos	Nome e descrição	Natureza
	Identificador do recurso	<i>String</i>
	Identificador do Usuário	<i>String</i>
	Tipo	<i>String</i>
	Descrição	<i>String</i>
	Criador	<i>String</i>
	Tags	<i>String</i>
	Caminho do diretório de armazenamento	<i>String</i>
	Nome do arquivo	<i>String</i>
	Tamanho do Arquivo	Inteiro
	Arquivo	Arquivo
	Data de criação	Data

Tabela 46. Requisito de Armazenamento 03 - Autorizações

RA-03	Autorizações	
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário	
Descrição	O sistema deve armazenar as informações sobre as autorizações de compartilhamento dos Usuários .	
Dados específicos	Nome e descrição	Natureza
	Identificador da autorização	<i>String</i>
	<i>Site</i> autorizado	<i>String</i>
	Recursos compartilhados	<i>String[]</i>
	Escopo de leitura	<i>Boolean</i>
	Escopo de escrita	<i>Boolean</i>
	<i>Token</i>	<i>String</i>
	Tipo de <i>token</i>	<i>String</i>
	Data de expiração	Data
	Data de criação	Data

Tabela 47. Requisito de Armazenamento 04 - Histórico

RA-04	Histórico	
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários	
Descrição	O sistema deve armazenar informações sobre o histórico de atividades da conta do Usuário (por exemplo, acessos a recursos, criação de recursos e criação de autorizações).	
Dados específicos	Nome e descrição	Natureza
	Identificador do Usuário	<i>String</i>
	Atividade	<i>String</i>
	Descrição	<i>String</i>
	Data de criação	Data

Tabela 48. Requisito de Armazenamento 05 - Sites registrados

RA-05	Sites registrados	
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário	
Descrição	O sistema deve armazenar informações de registro dos <i>sites</i> incluindo os segredos de autenticação.	
Dados específicos	Nome e descrição	Natureza
	URL	<i>String</i>
	Descrição	<i>String</i>
	Chave	<i>String</i>
	Segredo	<i>String</i>
	Data de expiração	Data
	Data de criação	Data

(Atividade 3) Definição dos Atores

Tabela 49. Ator 01 - Usuário

AT-01	Usuário
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-02: Autenticação de Usuários usando um Provedor de Identidade OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário
Descrição	O Usuário que possui uma conta no SGCD.

Tabela 50. Ator 02 - Provedor de Serviço

AT-02	Provedor de Serviço
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário
Descrição	O Provedor de Serviço que quer acessar os recursos de um Usuário do SGCD.

Tabela 51. Ator 03 - Provedor de Identidade

AT-03	Provedor de Identidade
Objetivos Associados	OBJ-02: Autenticação de Usuários usando um Provedor de Identidade
Descrição	O Provedor de Identidade que comprova a identidade de um Usuário do SGCD.

(Atividade 4) Definição dos Requisitos Funcionais

Tabela 52. Requisito Funcional 01 - Autenticar Usuário

RF-01	Autenticar Usuário		
Objetivos Associados	OBJ-02: Autenticação de Usuários usando um Provedor de Identidade OBJ-04: Proteção dos recursos do Usuário		
Descrição	O Usuário autentica-se junto ao seu Provedor de Identidade que, por sua vez, entrega uma asserção para o sistema comprovando a identidade do Usuário .		
Disparo	O Usuário tenta acessar uma parte protegida do sistema.		
Pré-condições	1. O Usuário possui uma conta em um Provedor de Identidade suportado. 2. O Usuário não fez <i>login</i> no sistema.		
Atores	AT-01: Usuário , AT-02: Provedor de Identidade		
Sequência Normal	Passo	Ação	
	1	O sistema mostra a tela de <i>login</i> .	
	2	O Usuário informa o seu Provedor de Identidade.	
	3	O sistema executa o protocolo suportado pelo Provedor de Identidade para autenticação do Usuário .	
	4	O Provedor de Identidade informa para o sistema se a autenticação do Usuário foi bem sucedida.	
	5	O sistema realiza o <i>login</i> o Usuário .	
Pós-condições	O Usuário fez <i>login</i> no sistema.		
Exceções	Código	Passo	Ação
	(E1)	4	<i>A autenticação do Usuário no Provedor de Identidade falha</i> E1.1. O Provedor de Identidade informa ao Sistema que a autenticação do Usuário falhou. Fim do caso de uso. Falha.

Tabela 53. Requisito Funcional 02 - Consultar recursos

RF-02	Consultar recursos		
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários		
Descrição	O Usuário visualiza os seus recursos armazenados no sistema.		
Disparo	O Usuário acessa a interface “Gerenciamento de Recursos” do sistema.		
Pré-condições	O Usuário está autenticado no sistema.		
Atores	AT-01: Usuário		
Sequência Normal	Passo	Ação	
	1	O sistema mostra uma lista dos recursos do Usuário .	
Pós-condições	O Usuário visualiza informações sobre os seus recursos no sistema.		
Exceções	Código	Passo	Ação
	(E1)	1	<i>Não existem recursos</i> E1.1. O Sistema mostra uma mensagem dizendo que o Usuário não possui recursos no sistema. Fim do caso de uso. Falha.

Tabela 54. Requisito Funcional 03 - Criar recursos

RF-03	Criar recursos		
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários		
Descrição	O Usuário cria recursos na sua conta no sistema.		
Disparo	O Usuário acessa a interface “Criar Novo Recurso” do sistema.		
Pré-condições	O Usuário está autenticado no sistema.		
Atores	AT-01: Usuário		
Sequência Normal	Passo	Ação	
	1	O Usuário entra o tipo, a descrição e o arquivo do recurso.	
	2	O sistema cria o novo recurso no banco de dados.	
	3	O Usuário confirma a operação de criação.	
	4	O sistema mostra uma mensagem de confirmação e cria o recurso no banco de dados.	
Pós-condições	As modificações feitas pelo Usuário nos seus recursos são persistidas no banco de dados do sistema.		
Exceções	Código	Passo	Ação
	(E1)	3	<i>Já existe um arquivo com o mesmo nome na conta do Usuário</i> E1.1. O Sistema retorna uma mensagem de erro para o Usuário dizendo que já existe um arquivo com o mesmo nome e que não foi possível criar o recurso. Fim do caso de uso. Falha.

Tabela 55. Requisito Funcional 04 - Remover recursos

RF-04	Remover recursos		
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários		
Descrição	O Usuário visualiza os seus recursos armazenados no sistema e pode remover alguns (ou todos) eles.		
Disparo	O Usuário acessa a interface “Remover Recursos” do sistema.		
Pré-condições	O Usuário está autenticado no sistema.		
Atores	AT-01: Usuário		
Sequência Normal	Passo	Ação	
	1	O sistema mostra uma lista dos recursos do Usuário .	
	2	O Usuário seleciona as entradas que ele quer remover.	
	3	O Usuário confirma a operação de remoção.	
	4	O sistema remove as entradas selecionadas da interface de visualização e do banco de dados.	
Pós-condições	As modificações feitas pelo Usuário nos seus recursos são persistidas no banco de dados do sistema.		
Exceções	Código	Passo	Ação
	(E1)	3	<i>Não existem entradas selecionadas</i> E1.1. O Sistema pergunta ao Usuário para selecionar uma entrada para remoção. Fim do caso de uso. Falha.

Tabela 56. Requisito Funcional 05 - Consultar histórico

RF-05	Consultar histórico		
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários		
Descrição	O Usuário visualiza as atividades ocorridas na sua conta do sistema (por exemplo, acessos a recursos, criação de recursos e criação de autorizações).		
Disparo	O Usuário acessa a interface “Gerenciamento de Histórico” do sistema.		
Pré-condições	O Usuário está autenticado no sistema.		
Atores	AT-01: Usuário		
Sequência Normal	Passo	Ação	
	1	O sistema mostra a lista das atividades ocorridas na conta do Usuário .	
Pós-condições	O Usuário visualiza informações sobre as atividades da sua conta.		
Exceções	Código	Passo	Ação
	(E1)	1	<i>Não existem atividades</i> E1.1. O Sistema mostra uma mensagem dizendo que o Usuário não possui atividades no sistema. Fim do caso de uso. Falha.

Tabela 57. Requisito Funcional 06 - Remover atividades

RF-06	Remover atividades		
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários		
Descrição	O Usuário remove atividades do seu histórico no sistema.		
Disparo	O Usuário acessa a interface “Remover Atividades” do sistema.		
Pré-condições	O Usuário está autenticado no sistema.		
Atores	AT-01: Usuário		
Sequência Normal	Passo	Ação	
	1	O sistema mostra uma lista das atividades ocorridas na conta do Usuário .	
	2	O Usuário seleciona as entradas que ele quer remover.	
	3	O Usuário confirma a operação de remoção.	
	4	O sistema remove as entradas selecionadas da interface de visualização e do banco de dados.	
Pós-condições	As modificações feitas pelo Usuário no seu histórico são persistidas no banco de dados do sistema.		
Exceções	Código	Passo	Ação
	(E1)	3	<i>Não existem entradas selecionadas</i> E1.1. O Sistema pergunta ao Usuário para selecionar uma entrada para remoção. Fim do caso de uso. Falha.

Tabela 58. Requisito Funcional 07 - Consultar autorizações

RF-07	Consultar autorizações		
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário		
Descrição	O Usuário visualiza as suas autorizações de compartilhamento aprovadas no sistema, válidas ou expiradas.		
Disparo	O Usuário acessa a interface “Gerenciamento de Autorizações” do sistema.		
Pré-condições	O Usuário está autenticado no sistema.		
Atores	AT-01: Usuário		
Sequência Normal	Passo	Ação	
	1	O sistema mostra uma lista das autorizações de compartilhamento aprovadas pelo Usuário .	
Pós-condições	O Usuário visualiza as autorizações de compartilhamento aprovadas para a sua conta.		
Exceções	Código	Passo	Ação
	(E1)	1	<i>Não existem atividades</i> E1.1. O Sistema mostra uma mensagem dizendo que o Usuário não possui autorizações no sistema. Fim do caso de uso. Falha.

Tabela 59. Requisito Funcional 08 - Remover autorizações

RF-08	Remover autorizações		
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário		
Descrição	O Usuário visualiza as suas autorizações de compartilhamento aprovadas no sistema, válidas ou expiradas, e pode remover algumas (ou todas) elas. Ao remover uma autorização, o Usuário removerá o acesso concedido a um Provedor de Serviço.		
Disparo	O Usuário acessa a interface “Remover Autorizações” do sistema.		
Pré-condições	O Usuário está autenticado no sistema.		
Atores	AT-01: Usuário		
Sequência Normal	Passo	Ação	
	1	O sistema mostra uma lista das autorizações de compartilhamento aprovadas pelo Usuário .	
	2	O Usuário seleciona as entradas que ele quer remover.	
	3	O Usuário confirma a operação de remoção.	
	4	O sistema remove as entradas selecionadas da interface de visualização e do banco de dados.	
Pós-condições	As modificações feitas pelo Usuário nas suas autorizações são persistidas no banco de dados do sistema.		
Exceções	Código	Passo	Ação
	(E1)	3	<i>Não existem entradas selecionadas</i> E1.1. O Sistema pergunta ao Usuário para selecionar uma entrada para remoção. Fim do caso de uso. Falha.

Tabela 60. Requisito Funcional 09 - Requisitar autorização

RF-09	Requisitar autorização		
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário		
Descrição	Um Provedor de Serviço faz uma requisição ao sistema para acesso a recursos do Usuário . O sistema, por sua vez, exibe a requisição ao Usuário e pergunta se ele autoriza o compartilhamento dos dados.		
Disparo	O Provedor de Serviço requisita acesso a recursos de um determinado Usuário .		
Pré-condições	1. O Usuário possui uma conta válida no sistema. 2. O Provedor de Serviço possui registro no sistema.		
Atores	AT-01: Usuário , AT-02: Provedor de Serviço		
Sequência Normal	Passo	Ação	
	1	O Provedor de Serviço requisita ao sistema recursos de um Usuário .	
	2	O sistema autentica o Provedor de Serviço.	
	3	O sistema autentica o Usuário .	
	4	O sistema exibe as seguintes informações para o Usuário : o escopo da requisição de acesso (leitura ou escrita), quem é o Provedor de Serviço requisitante e os recursos referentes à requisição que o Usuário possui.	
	5	O Usuário seleciona os recursos que ele deseja compartilhar com o Provedor de Serviço.	
	6	O Usuário seleciona o tempo de expiração da autorização de acesso.	
	7	O Usuário confirma a seleção dos passos 5 e 6 permitindo que o Provedor de Serviço tenha acesso aos recursos.	
	8	O sistema entrega uma autorização de acesso ao Provedor de Serviço.	
Fluxos Alternativos	Código	Passo	Ação
	(A1)	1	<i>O Provedor de Serviço já fez login</i> A1.1. O sistema continua no passo 3 do fluxo normal.
	(A2)	3	<i>O Usuário já fez login</i> A2.1. O sistema continua no passo 4 do fluxo normal.
Pós-condições	O Provedor de Serviço recebe uma autorização de acesso do sistema.		
Exceções	Código	Passo	Ação
	(E1)	2	<i>A autenticação do Provedor de Serviço falha</i> E1.1 O sistema envia uma mensagem de erro para o Provedor de Serviço. Fim do caso de uso. Falha.
	(E2)	3	<i>A autenticação do Usuário falha</i> E2.1. O sistema envia uma mensagem de erro para o Provedor de Serviço. Fim do caso de uso. Falha.
	(E3)	4	<i>O Usuário não possui os recursos requisitados</i> E3.1. O sistema envia uma mensagem de erro para o Provedor de Serviço. Fim do caso de uso. Falha.
	(E4)	7	<i>O Usuário cancela o processo de autorização</i> E4.1. O sistema envia uma mensagem de erro para o Provedor de Serviço. Fim do caso de uso. Falha.

Tabela 61. Requisito Funcional 10 - Ler recurso

RF-10	Ler recurso		
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário		
Descrição	Um Provedor de Serviço requisita ao sistema um acesso de leitura a recursos do Usuário .		
Disparo	Um Provedor de Serviço faz uma requisição ao sistema de recursos do Usuário .		
Pré-condições	1. O Provedor de Serviço possui registro no sistema. 2. O Provedor de Serviço possui uma autorização de acesso para ler os recursos requisitados.		
Atores	AT-02: Provedor de Serviço		
Sequência Normal	Passo	Ação	
	1	O Provedor de Serviço envia uma requisição para ler um conjunto de recursos do Usuário juntamente com uma autorização de acesso previamente concedida pelo sistema.	
	2	O sistema autentica o Provedor de Serviço.	
	3	O sistema checa se a autorização de acesso é válida.	
	4	O sistema envia um conjunto de recursos referente à autorização de acesso para o Provedor de Serviço.	
Pós-condições	O Provedor de Serviço recebe um conjunto de recursos do Usuário .		
Exceções	Código	Passo	Ação
	(E1)	2	<i>A autenticação do Provedor de Serviço falha</i> E1.1. O sistema responde ao Provedor de Serviço com uma mensagem de erro. Fim do caso de uso. Falha.
	(E2)	3	<i>Autorização de acesso inválida</i> E2.1. O sistema responde ao Provedor de Serviço com uma mensagem de erro. Fim do caso de uso. Falha.

Tabela 62. Requisito Funcional 11 - Escrever recurso

RF-11	Escrever recurso		
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário		
Descrição	Um Provedor de Serviço requisita ao sistema um acesso de escrita a recursos do Usuário .		
Disparo	Um Provedor de Serviço faz uma requisição ao sistema de recursos do Usuário .		
Pré-condições	1. O Provedor de Serviço possui registro no sistema. 2. O Provedor de Serviço possui uma autorização de acesso para escrever os recursos requisitados.		
Atores	AT-02: Provedor de Serviço		

Sequência Normal	Passo	Ação	
	1	O Provedor de Serviço envia uma requisição para escrever um conjunto de recursos do Usuário juntamente com uma autorização de acesso previamente concedida pelo sistema.	
	2	O sistema autentica o Provedor de Serviço.	
	3	O sistema checa se a autorização de acesso é válida.	
	4	O Provedor de Serviço envia um conjunto de recursos referente à autorização de acesso para o sistema.	
	5	O sistema salva os recursos na conta do Usuário .	
Pós-condições	O sistema salva um conjunto de recursos na conta do Usuário .		
Exceções	Código	Passo	Ação
	(E1)	2	<i>A autenticação do Provedor de Serviço falha</i> E1.1. O sistema responde ao Provedor de Serviço com uma mensagem de erro. Fim do caso de uso. Falha.
	(E2)	3	<i>Autorização de acesso inválida</i> E2.1. O sistema responde ao Provedor de Serviço com uma mensagem de erro. Fim do caso de uso. Falha.

Tabela 63. Requisito Funcional 12 - Autenticar Provedor de Serviço

RF-12	Autenticar Provedor de Serviço		
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário		
Descrição	O Provedor de Serviço autentica-se junto ao sistema usando uma credencial (chave+segredo) previamente registrada.		
Disparo	O Provedor de Serviço requisita um token de acesso.		
Pré-condições	1. O Provedor de Serviço possui registro no sistema. 2. O Provedor de Serviço não fez <i>login</i> no sistema.		
Atores	AT-02: Provedor de Serviço		
Sequência Normal	Passo	Ação	
	1	O sistema requisita autenticação do Provedor de Serviço.	
	2	O Provedor de Serviço informa sua chave e seu segredo.	
	3	O sistema realiza o <i>login</i> do Provedor de Serviço.	
Pós-condições	O Provedor de Serviço fez <i>login</i> no sistema.		
Exceções	Código	Passo	Ação
	(E1)	2	<i>Credenciais do Provedor de Serviço Inválidas</i> E1.1. O sistema responde ao Provedor de Serviço com uma mensagem de erro. Fim do caso de uso. Falha.

Tabela 64. Requisito Funcional 13 - Registrar Provedor de Serviço

RF-13	Registrar Provedor de Serviço		
Objetivos Associados	OBJ-04: Proteção dos recursos do Usuário		
Descrição	Registro de uma nova conta de um Provedor de Serviço e entrega de credenciais de autenticação.		
Disparo	O Provedor de Serviço requisita o registro no sistema.		
Pré-condições	1. O Provedor de Serviço não fez <i>login</i> no sistema. 2. O Provedor de Serviço controla um domínio válido.		
Atores	AT-02: Provedor de Serviço		
Sequência Normal	Passo	Ação	
	1	O Provedor de Serviço fornece as informações requeridas, tais como URL e descrição da aplicação.	
	2	O sistema checa as informações.	
	3	O sistema envia as credenciais geradas para o Provedor de Serviço.	
Pós-condições	O Provedor de Serviço é registrado e recebe uma credencial (chave+segredo).		
Exceções	Código	Passo	Ação
	(E1)	2	<i>URL do Provedor de Serviço já foi registrada anteriormente</i> E1.1. O Sistema informa ao Provedor de Serviço que o URL já foi registrado anteriormente. Fim do caso de uso. Falha.
	(E2)	2	<i>Informações de registro inválidas</i> E2.1. O Sistema informa ao Provedor de Serviço que o URL já foi registrado anteriormente. Fim do caso de uso. Falha.

(Atividade 5) Definição dos Requisitos Iterativos**Tabela 65. Frase 01 - Recuperação de recursos por tipo**

FR-01	Recuperação de recursos por tipo	
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i>	
Descrição	Enunciado	Atores
	'RA-02.Tipo' exatamente igual a 'Tipo pesquisado'	AT-01: Usuário

Tabela 66. Frase 02 - Recuperação de atividade por período

FR-02	Recuperação de atividade por período	
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários	
Descrição	Enunciado	Atores
	'RA-04.Data de criação' menor que 'Data Final pesquisada' e maior que 'Data Inicial pesquisada'	AT-01: Usuário

Tabela 67. Frase 03 - Recuperação de autorização por Provedor de Serviço

FR-03	Recuperação de autorização por Provedor de Serviço	
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários	
Descrição	Enunciado	Atores
	'RA-05.URL' contém o 'URL pesquisado'	AT-01: Usuário

Tabela 68. Frase 04 - Recuperação de Provedor de Serviço por URL

FR-04	Recuperação de Provedor de Serviço por URL	
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário	
Descrição	Enunciado	Atores
	'RA-05.URL' exatamente igual a 'URL pesquisado'	AT-01: Usuário

Tabela 69. Frase 05 - Recuperação de recursos por tag

FR-05	Recuperação de recursos por tag	
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários	
Descrição	Enunciado	Atores
	'RA-02.Tag' contém o 'Tag pesquisado'	AT-01: Usuário

Tabela 70. Requisito Interativo 01 - Informação sobre os recursos do Usuário

RI-01	Informação sobre os recursos do Usuário
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários
Atores	AT-01: Usuário
Descrição	O sistema deve mostrar informações sobre os recursos do Usuário .
Frases	FR-01 e FR-05
Funcionalidades Associadas	RF-02, RF-03, RF-04
Informação Mostrada	'RA-02.Tipo', 'RA-02.Descrição', 'RA-02.Criador', 'RA-02.Nome do arquivo', 'RA-02.Arquivo', 'RA-02.Data de criação'
Saída	RI-02, RI-03
Entrada	RI-02, RI-03

Tabela 71. Requisito Interativo 02 - Informação sobre as atividades

RI-02	Informação sobre as atividades
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários
Atores	AT-01: Usuário
Descrição	O sistema deve mostrar informações sobre as atividades realizadas na conta do Usuário .
Frases	FR-02
Funcionalidades Associadas	RF-05, RF-06
Informação Mostrada	'RA-04.Atividade', 'RA-04.Descrição', 'RA-04.Data de criação'
Saída	RI-01, RI-03
Entrada	RI-01, RI-03

Tabela 72. Requisito Interativo 03 - Informação sobre as autorizações do Usuário

RI-03	Informação sobre as autorizações do Usuário
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i>
Atores	AT-01: Usuário
Descrição	O sistema deve mostrar informações sobre as autorizações de compartilhamento concedidas pelo Usuário .
Frases	FR-03
Funcionalidades Associadas	RF-07, RF-08
Informação Mostrada	'RA-03.Site autorizado', 'RA-03.Recursos compartilhados', 'RA-03.Escopo de leitura', 'RA-03.Escopo de escrita', 'RA-03.Data de expiração', 'RA-03.Data de criação',
Saída	RI-01, RI-02
Entrada	RI-01, RI-02

Tabela 73. Requisito Interativo 04 - Informação sobre uma requisição de Autorização

RI-04	Informação sobre uma requisição de autorização
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-04: Proteção dos recursos do Usuário
Atores	AT-01: Usuário
Descrição	O sistema deve mostrar informações sobre o Provedor de Serviço que está fazendo uma requisição de acesso e quais são os recursos que estão sendo requisitados.
Frases	FR-01, FR-04
Funcionalidades Associadas	RF-09
Informação Mostrada	'RA-02.Tipo', 'RA-02.Descrição', 'RA-02.Criador', 'RA-02.Nome do arquivo', 'RA-02.Arquivo', 'RA-02.Data de criação' 'RA-05.URL', 'RA-05.Descrição'
Saída	-
Entrada	-

(Atividade 6) Definição dos Requisitos Não Funcionais

Tabela 74. Requisito Não Funcional 01 - Confidencialidade e integridade

RNF-01	Confidencialidade e integridade
Objetivos Associados	OBJ-04: Proteção dos recursos do Usuário
Descrição	O sistema deve implementar as melhores práticas de segurança para assegurar que somente as entidades autorizadas pelo proprietário dos recursos poderão ler ou modificar os recursos protegidos.

Tabela 75. Requisito Não Funcional 02 - Compartilhamento de tipos de dados genéricos

RNF-02	Compartilhamento de tipos de dados genéricos
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i>
Descrição	O sistema deve suportar o compartilhamento de qualquer tipo de dado. O sistema não precisa entender as representações dos dados, mas deve permitir que Provedores de Serviço com diferentes lógicas de negócio possam usar o sistema para obter acesso aos dados dos Usuários .

Tabela 76. Requisito Não Funcional 03 - Autorização explícita do Usuário

RNF-03	Autorização explícita do Usuário
Objetivos Associados	OBJ-04: Proteção dos recursos do Usuário
Descrição	Toda requisição de autorização de acesso a recursos protegidos deve ser explicitamente autorizada pelo Usuário .

Tabela 77. Requisito Não Funcional 04 - Uso de navegador *web* como agente do Usuário

RNF-04	Uso de navegador <i>web</i> como agente do Usuário
Objetivos Associados	OBJ-05: Implantação do sistema em larga escala na Internet
Descrição	O sistema deve ser compatível com as tecnologias existentes nos navegadores <i>web</i> e não deve exigir que o Usuário instale nenhuma aplicação em seu dispositivo, além do navegador, para utilizar o sistema.

Tabela 78. Requisito Não Funcional 05 - Uso de padrões abertos e amplamente adotados

RNF-05	Uso de padrões abertos e amplamente adotados
Objetivos Associados	OBJ-05: Implantação do sistema em larga escala na Internet
Descrição	O sistema deve ser construído sobre a infra-estrutura de Internet já existente e padrões abertos, de forma que seja mais fácil atrair provedores e Usuários .

Tabela 79. Requisito Não Funcional 06 - Disponibilidade

RNF-06	Disponibilidade
Objetivos Associados	OBJ-01: Gerenciamento de Contas dos Usuários OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-05: Implantação do sistema em larga escala na Internet
Descrição	O sistema deve implementar as melhores práticas de segurança para assegurar que os serviços de gerenciamento e compartilhamento de dados estejam disponíveis para acesso dos Usuários do sistema

Tabela 80. Requisito Não Funcional 07 - Complexidade baixa para as aplicações web

RNF-07	Complexidade baixa para as aplicações <i>web</i>
Objetivos Associados	OBJ-03: Compartilhamento de recursos do Usuário com Aplicações <i>Web</i> OBJ-05: Implantação do sistema em larga escala na Internet
Descrição	A arquitetura do sistema deve prever que as operações mais complexas relacionadas ao compartilhamento de recursos sejam executadas pelo próprio sistema de forma a oferecer a menor complexidade possível para as aplicações <i>web</i> .