UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Martín Augusto Gagliotti Vigil

Autoridade Certificadora de Correio Eletrônico

Trabalho de Conclusão de Curso submetido à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Bacharel em Ciência da Computação.

Túlio Cícero Salvaro de Souza
Orientador

Prof. Ricardo Felipe Custódio, Dr.
Co-Orientador

Florianópolis, Julho de 2007

Autoridade Certificadora de Correio Eletrônico

Martín Augusto Gagliotti Vigil

Este Trabalho de Conclusão de Curso foi julgado adequado para a obtenção do título de Bacharel em Ciência da Computação e aprovado em sua forma final pelo Departamento de Informática e Estatística da Universidade Federal de Santa Catarina.

Prof. José Mazzuco Júnior, Dr.
Coordenador do Curso

Banca Examinadora

Túlio Cícero Salvaro de Souza
Orientador

Marcelo Carlomagno Carlos

Prof. Júlio da Silva Dias, Dr.

”Mergulhe no meio das coisas, suje as m aos, caia de joelhos e, então, procure alcançar as estrelas”. John. L Curcio

Ofereço este trabalho a meus pais e avós, que me deram tudo o necessário para eu superá-los em conhecimento.

Agradecimentos

Os primeiros agradecimentos vão a minha família, que desde o meu primeiro dia de aula na escola básica, até o final da graduação, me dá todo o apoio e carinho mais do que necessários para meu sucesso.

Em seguida, gostaria de deixar clara minha gratidão àqueles que também participaram da minha formação profissional. Desde de meus supervisores de estágio - João Goulart Júnior, Alexandre Nuernberg, Roberto Soldi e Jean Everson Martina - até o Professor Ricardo Felipe Custódio, quem me confiou o desenvolvimento deste trabalho.

Por último, agradeço aos meus colegas de trabalho, em especial a Túlio Salvaro, Marcelo Carlomagno, Jeandré Monteiro, entre outros, por todo o apoio e conhecimento passado, além da amizade e descontração que alegram minhas tarde de trabalho no LabSEC.

Sumário

Lista de Figuras **x**

Lista de Siglas **xi**

Resumo **xii**

Abstract **xiv**

1 Introdução **1**

1.1 Justificativa 1

1.2 Objetivos 2

1.2.1 Objetivos Específico 2

1.3 Trabalhos Relacionados 3

1.4 Metodologia 3

1.5 Estrutura do Trabalho 3

1.6 Estrutura do Trabalho 4

2 Criptografia **5**

2.1 Criptografia Simétrica 6

2.2 Funções de Resumo Criptográfico 7

2.3 Criptografia Assimétrica 8

2.4 Assinatura Digital 8

3 Infra-estrutura de Chaves Públicas **10**

3.1 Certificados Digitais 10
3.2 Lista de Certificados Revogados 11
3.3 Autoridade Certificadora 11
3.4 Autoridade de Registro 12
3.5 Políticas de Certificação 12
3.6 Módulo Público 14
3.7 Modelos de Transação 14

4 Tecnologias Utilizadas **16**
4.1 OpenSSL 16
4.2 LibCryptoSec 16
4.3 PHP 17
4.4 Zend Framework 17
4.5 C++ 17
4.6 Qt4 18
4.7 JavaScript 18
4.8 HTML 18
4.9 Linux 19
4.10 UML 19
4.11 XML 19
4.12 PostgreSQL 19

5 AC de Correio Eletrônico **21**
5.1 Projeto 21
5.1.1 Requisitos Funcionais 21
5.1.2 Requisitos Não Funcionais 24
5.1.3 Protocolo de Desafio-resposta e de Emissão de Certificado . . 24
5.1.4 Organização Modular 26
5.2 Módulo Criptográfico 27
5.2.1 Organização Modular 29

5.2.2 Par de Chaves da AC Correio 30
5.2.3 Modelo de Certificados Emitidos 31
5.2.4 Emissão de Certificado 31
5.2.5 Emissão de LCR 33
5.2.6 Limpeza de Dados Expirados 33
5.2.7 Implementação 34
5.3 Módulo Público 35
5.3.1 Organização modular 37
5.3.2 Páginas 37
5.3.3 Verificação de Endereço de E-mail 38
5.3.4 Geração da URL Secreta de Desafio 39
5.3.5 Geração do Par de Chaves 39
5.3.6 Implentação 40
5.4 Banco de Dados 41

6 Políticas de Certificação **44**
6.1 RFC3647 44
6.1.1 Introdução 45
6.1.2 Responsabilidade referentes a publicações e repositórios . . . 45
6.1.3 Identificação e Autenticação 45
6.1.4 Requisitos Operacionais do Ciclo de Vida do Certificado . . . 45
6.1.5 Controles Técnicos de Segurança 46
6.1.6 Perfis dos Certificados, LCR e OCSP 46

7 Considerações Finais **47**

Referências **49**

8 Anexos **53**
8.1 Arquivo de configuração do módulo criptográfico 53
8.2 Diagrama de Classes de Análise d Módulo Criptográfico 57

8.3 Classe SystemController 58
8.4 Classe Persistence 61
8.5 Classe Crypto 66
8.6 Classe ConfigData 68
8.7 Classe ConfigurationFileReader 75
8.8 Classe AbstractLogger 78
8.9 Classe ConsoleLogger 80
8.10 Classe FileLogger 81
8.11 Classe Notifier 83
8.12 Classe ReqBuilder 86
8.13 Classe Scheduler 88
8.14 Classe Utils 90
8.15 Classe LibCryptoSecException 93
8.16 Classe FileSystemException 94
8.17 Classe PersistenceException 97
8.18 Classe SyntaxException 101
8.19 Página Inicial do Módulo Público 102
8.20 Página de Submissão de Endereço de E-mail 102
8.21 Página de Geração de Chaves no Mozilla Firefox 102
8.22 Página de Geração de Chaves no Internet Explorer 102
8.23 Classe Persistence 102
8.24 Classe DomainsChecker 112
8.25 Arquivo de Configuração do Módulo Público 113
8.26 Arquivo de configuração do Notificador de Desafio 114
8.27 Classe ChallengeNotifier 115
8.28 Rotinas de Geração de Chaves do Microsoft Crypto API 116
8.29 Formulário de Requisição de Certificado 128

Lista de Figuras

2.1 Assinatura digital. 9

3.1 Diagrama UML da Relação entre PC e DPC. 13
3.2 Modelo de transação da AC Correio. 15
3.3 Modelo de transação de três partes. 15

5.1 Protocolo de desafio-resposta da AC Correio. 25
5.2 Protocolo de Emissão de Certificado pela AC Correio. 26
5.3 Organização modular da AC Correio. 27
5.4 Fluxograma da execução do módulo criptográfico. 28
5.5 Organização modular do módulo criptográfico. 29
5.6 Passos para obtenção de certificado. 36
5.7 Organização modular do módulo público. 37
5.8 Diagrama de navegaçao do módulo público. 38
5.9 Modelagem do Banco de Dados da AC Correio. 42

8.1 Diagrama de classes de análise do módulo criptográfico. 57
8.2 Página inicial do módulo público. 103
8.3 Página de submissão de endereço de e-mail. 104
8.4 Página de geração de chave no Mozilla Firefox. 105
8.5 Página de geração de chave no Internet Explorer. 106

Lista de Siglas

AC	Autoridade Certificadora
AGP	Autoridade Gestora de Políticas
API	Application Programming Interface
AR	Autoridade de Registro
CBC	Cipher Block Chaining
CPF	Cadastro de Pessoa Física
DES	Data Encryption Standard
GOPAC	Grupo de Operação Autoridade Certificadora
ICP	Infra-estrutura de Chave Pública
LabSEC	Laboratório de Segurança em Computação - UFSC
LCR	Lista de Certificados Revogados
NIST	National Institute of Standards and Technology
OCSP	Online Certificate Status Protocol
PC	Políticas de Certificação
DPC	Declaração de Práticas de Certificação
PHP	Hypertext Preprocessor
RSA	Rivest-Shamir-Adleman
SHA	Secure Hash Algorith
SQL	Structured Query Language
SSL	Secure Socket Layer

Resumo

Este trabalho de conclusão de curso é parte integrante do projeto de Infraestrutura de Chaves Públicas Educacional (ICPEDU) e tem como objetivo desenvolver uma aplicação gestora para uma Autoridade Certificadora de Correio Eletrônico que seja simples e de licença livre.

Buscando automatizar a emissão de certificados digitais, o sistema gestor proposto trabalha sem a cooperação de uma Autoridade de Registro. Para isso, projetou-se um protocolo de desafio-resposta cuja finalidade é comprovar a identidade do requerente de certificado perante a Autoridade Certificadora.

Este trabalho utiliza idéias de outras soluções de ICP, como o NewPKI. Assim, o sistema gestor apresenta módulos público e criptográfico - responsáveis pela interface com o usuário final e emissão de certificados, respectivamente - interligados através de uma entidade central: o banco de dados.

Com base na RFC3647 elabora-se o documento de Declaração de Práticas de Certificação da Autoridade Certificadora de Correio Eletrônico. Neste definem-se as políticas de operação da AC, bem os perfis e ciclo de vida dos certificados emitidos, como ainda questões de segurança física e lógica envolvidas, entre outros ítens.

Por fim, alcançam-se todos os objetivos traçados, uma vez que o sistema gestor desenvolvido opera com corretude bem como os certificados emitidos foram testados com sucesso em aplicativos de correio eletrônico. Sugerem-se como trabalhos futuros a criação de funcionalidades de registro de eventos, uma interface de administração e uso de certificados digitais para garantir autenticação entre os módulos do sistema gestor.

Palavras chave: Autoridade Certificadora, correio eletrônico, certificado digital

Abstract

Capítulo 1

Introdução

De papel e tinta para *bytes* - mudança nítida de *mídia* resultado do emprego de computadores no cotidiano da atualidade. Isso transformou a maneira do homem manipular informações bem como protegê-las: enquanto segurança física e autenticação por meio de assinatura a mão protegem documentos em forma de papéis, o mesmo não ocorre com informações digitais. Para tal, é aconselhável se fazer uso da criptografia.

Um exemplo de proteção de informação digital por meio de criptografia que se torna a cada dia mais utilizado é o correio eletrônico seguro. Através deste podem-se enviar informações de maneira sigilosa, bem como autenticá-las por meio de assinatura digital. Todavia, para este serviço se faz necessário que o usuário adquira previamente um certificado digital através de uma Autoridade Certificadora.

Assim, este trabalho propõe uma solução específica de sistema gestor de Autoridade Certificadora focada na emissão de certificados digitais para correio eletrônico.

1.1 Justificativa

O sistemas gestores de Autoridades Certificadoras disponíveis hoje não são, em sua maioria, focados apenas na emissão de certificados para e-mail, mas sim para qualquer propósito, como assinatura de documentos eletrônicos, túneis SSL, entre

outros. Desta maneira, tais softwares gestores apresentam uma gama de funcionalidades que os tornam mais complexos de utilizá-los, exigindo de quem os opera sólidos conhecimentos de ICP. Como exemplo podem citar o NewPKI [1], melhor detalhado na Seção 1.3.

Com exceção dos dois softwares citados acima, grande parte das soluções para gestão de Autoridades Certificadoras são proprietárias. Portanto, de código fechado e sua licença deve ser paga ao autor.

Assim, os fatores citados anteriormente ilustram um cenário na área de certificação digital que justifica e motiva o desenvolvimento deste trabalho.

1.2 Objetivos

Conforme cenário de softwares gestores de Autoridade Certificadora citados anteiormente, tem-se como objetivo principal desenvolver um sistema simples e específico para a emissão de certificados digitais para correio eletrônico. Além disso, também deve ser simples o módulo público de modo que seja possível que um usuário leigo consiga obter um certificado digital.

Por último, todo conhecimento e software concebido neste trabalho deve ser de domínio público, ou seja, será livre e de licença livre.

1.2.1 Objetivos Específico

A Autoridade Certificadora de Correio Eletrônico deve operar com par de chaves armazenada em disco ou em um módulo de hardware de segurança (HSM). Ou seja, o módulo criptográfico deve poder ser integrado com qualquer HSM através de uma *engine* OpenSSL.

Por último, o par de chaves criptográficas do requerentes de certificados devem ser gerados localmente no navegador. Para isto, é necessário que o módulo público disponibilize rotinas de geração de chaves de acordo com as particularidades do navegador do usuário.

1.3 Trabalhos Relacionados

Estudou-se o NewPKI, cujas idéias contribuíram significativamente para o projeto da sistema gestor da Autoridade Certificadora de Correio Eletrônico. Aquela aplicação é uma solução completa para ICP orientada a entidades: Autoridade Certificadora, Autoridades de Registro, Módulo Público, repositórios. Essas comunicam-se através de um banco de dados e autenticam-se uma perante as outras através do uso de certificados digitais emitidos pela Autoridade Certificadora Interna do NewPKI.

Como vantagens do NewPKI podem-se citar a flexibilidade de configurações de ICP e de perfis de certificados emitidos, como ainda a operação de múltiplas Autoridades Certificadoras de maneira centralizada em um único servidor. Já as desvantagens tornam-se visíveis devido à complexidade de uso da aplicação e à geração do par de chaves dos requerentes de certificados: esta é realizada na parte do servidor. Tal fato passível de crítica, uma vez que a chave privada estaria sob "tutela" temporária do NewPKI enquanto o requerente não solicitasse a remoção ao público do NewPKI.

1.4 Metodologia

A elaboração deste documento se deu seguindo metodologia de pesquisa bibliográfica, uma vez que os fundamentos deste trabalho referem-se a materiais já publicados. Destes valem citar livros de criptografia, bem como sites web e documentos técnicos.

1.5 Estrutura do Trabalho

Neste trabalho descreve-se detalhadamente uma proposta de Autoridade Certificadora de Correio da maneira mais rica possível através da seguinte subdivisão de tópicos: contextualização teórica a respeito de criptografia e ICP, solução desenvolvida - projeto, implementação e tecnologias adotadas - políticas de certificação elaboradas para a AC Correio, bem como considerações finais e anexos.

A elaboração deste documento se deu seguindo metodologia de pesquisa bibliográfica, uma vez que os fundamentos deste trabalho referem-se a materiais já publicados. Destes valem citar livros de criptografia, bem como sites web e documentos técnicos.

1.6 Estrutura do Trabalho

Este trabalho está estruturado da seguinte forma:

- Capítulo 2: inicia-se a revisão bibliográfica com conceitos básicos relacionados à certificação digital: desde criptografia simétrica e assimétrica, funções resumo e assinatura digital;
- Capítulo 3: finaliza-se a revisão bibliográfica, abordando conceitos sobre Infraestrutura de Chaves Públicas;
- Capitulo 4: apresenta-se a solução de sistema gestor: desde o projeto do protocolo de desafio-resposta e dos módulos público e criptográfico, até a implementação desses;
- Capítulo 5: apresentam-se as técnologias utilizadas no desenvolvimento da solução;
- Capítulo 6: apresenta-se brevemente a RFC3647 e o documento de Declaração de Práticas de Certificação elaborado para a Autoridade Certificadora de Correio Eletrônico no contexto da ICPEDU.

Capítulo 2

Criptografia

A palavra *criptografia* tem origens na Língua Grega e é um composto de *kryptós* e *graphía*, que significam, respectivamente, oculto, secreto, obscuro, ininteligível; e escrita, de acordo com [2]. Ou seja, refere-se, literalmente, a uma escrita secreta cujo entendimento é restrito apenas aos que podem entendê-la.

Conceitualmente, Criptografia é a ciência que estuda as transformações matemáticas sobre uma informação inteligível, tornando-a incompreensível. Chama-se essa operação de *cifragem*. A operação contrária, que torna a informação compreensível novamente é denominada de *decifragem*. Por último, tais operações se dão utilizando-se uma terceira parte: a *chave* ou segredo [2].

Os relatos mais antigos do início da história da Criptografia datam de 4000 AC e são a respeito da civilização egípcia. Esta possuía uma cultura de elaborar documentos funerários que apresentavam a substituição alguns símbolos comuns de escrita por outros não usuais, conhecidos por poucos indivíduos. Todavia, incialmente o intuito destas modificações não buscavam esconder informações, mas sim transmitir dignidade e autoridade [3].

Contribuíram para a Criptografia outras civilizações, como a chinesa, indiana, mesopotâmica, hebraica, romana. Esta desenvolveu o cifrador de César [3], que cifrava mensagens através de técnicas de substituição de letras. Dezenas de séculos a frente, na Modernidade, a criptografia torna-se elemento crucial pra proteger segredos

e estratégias de nações, sendo peça decisiva no desfecho de duas guerras mundiais [4].

De acordo com [4], as técnicas matemáticas de transformação estudadas pela criptografia relacionam-se a os seguintes aspectos da segurança da informação:

- confidencialidade: capacidade de manter a informação secreta a todos exceto aos autorizados a ela;
- integridade: capacidade de verificar alterações - inserção, remoção ou substituição - no conteúdo de uma informação;
- autenticação: capacidade de identificar entidades uma perante as outras;
- não repúdio: capacidade de prevenir que entidades realizem atos e posteriomente os neguem.

2.1 Criptografia Simétrica

Chama-se de criptografia simétrica a técnica de cifragem em que usa-se a mesma chave para cifrar e decifrar uma mensagem [5]. Tal abordagem também é conhecida por *sistemas de chave compartilhada*.

Assim, se Alice deseja enviar uma mensagem com privacidade para Beto, ambos devem definir uma chave e um algoritmo de cifragem. Alice cifra a mensagem com a chave escolhida e envia a Beto. Este recebe a mensagem cifrada e a decifra utilizando a mesma chave e algorítmo de Alice.

Algoritmos criptografia simétrica são de uso simples por parte dos usuário e, em sua maioria, apresentam alto desempenho computacional, sendo recomendados para cifrar grandes quantidades de dados. No entanto, o gerenciamento das chaves compartilhadas é um ponto crítico dessa abordagem. Por exemplo, se Alice deseja enviar mensages sigilosas a seus amigos, ela deverá definir uma chave diferente com cada um deles.

Outro fator que merece atenção é o compartilhamento de uma chave entre duas entidades. Por exemplo, Alice define uma chave e a manda por e-mail para Beto

para que ambos possam trocar mensagens com privacidade. Se, neste momento, um terceiro individuo interceptar o e-mail de Alice, ele poderá decifrar todas as mensagens trocadas entre Alice e Beto.

Por último, há algoritmos de criptografia simétrica largamente difundidos hoje. Entre eles vale citar o DES - *Data Encryption Standard* - e o AES - *Advanced Encryption Standard* -, ambos patrocinados pelo Governo Norte-americano [5]. Esses utilizam uma unidade mínima de informação - um bloco de bits - que é cifrada a cada rodada do algorítmo. A forma como os blocos são processados denomina-se de *modo de operação*, sendo a mais utilizada a *Cipher Block Chaining* (CBC) [5].

2.2 Funções de Resumo Criptográfico

São funções matemáticas que, para cada valor de entrada, produzem uma saída de tamanho fixo que identifica unicamente, ou resume, uma informação [4]. Tais funções também são conhecidas como funções de espalhamento ou dispersão: tradução do inglês *hash* para o português.

Do modo que, idealmente, duas informações distintas não podem ter o mesmo resumo, tais funções são úteis para a verificação da integridade de mensagens. Isso porque, qualquer modificação efetuada sobre um dado, provocará a mudança do valor do resumo, acusando a violação da integridade.

São características que se esperam de uma função de resumo criptográfico:

- Ser computacionalmente inviável recriar uma mensagem a partir de seu resumo;
- Ser computacionalmente inviável construir duas diferentes mensagens que produzam o mesmo resumo.

O algoritmo de resumo criptográfico mais difundido atualmente é o *Secure Hash Algorith* (SHA-1) [5]. Atualmente o *National Institute of Standards and Technology* (NIST) [6] - orgão do Governo Estadunidense que regulariza padrões - vem trabalhando na homologação de novas versões do SHA-1 - SHA-256, SHA-384 e SHA-512 - buscando aprimorar a segurança do algorítmo.

2.3 Criptografia Assimétrica

A busca por soluções para os problemas da criptografia simétricas citados na Seção 2.1 foram as principais motivações para o desenvolvimento da criptografia assimétrica.

Considera-se como marco inicial trabalho dos americanos Whitfield Diffie e Martin Hellman [7] que propunha o uso de duas chaves: uma privada e outra pública. A primeira devia ser mantida secreta e utilizada para cifrar mensagens. A pública devia ser distribuída a todos os destinatários, uma vez que ela permitia decifrar as mensagens cifradas com a primeira. Tal concepção era baseada em aritmética modular, no entanto, no ano da publicação - 1976 - ainda não havia tecnologia suficiente para implementar e utilizar de fato a criptografia assimétrica.

Mesmo com as limitações tecnológicas o conceito da criptografia assimétrica foi muito bem aceito, pois vinha ao encontro das necessidades de uma forma de gerenciamento mais simples e segura do que a das chaves simétricas. Dois anos depois, Rivest, Shamir e Adleman propões a primeira implementação prática de um cifrador assimétrico, e o batizam de RSA. Este nasceu sobre o problema de inviabilidade computacional de se fatorar grandes números primos.

Nos anos seguintes o RSA foi submetido a melhorias e, atualmente, é o algoritmo de criptografia assimétrica mais difundido [8].

2.4 Assinatura Digital

O propósito da assinatura digital é prover um meio que permita a um indivíduo anexar sua identidade a um pedaço de informação. Assim, o processo de assinar implica em transformar uma mensagem e alguma informação secreta do indivíduo em uma “marca”, chamada de assinatura [4].

A assinatura digital provê autenticação e não repúdio. Portanto, ao assinar o assinante atesta o conhecimento do conteúdo da mensagem, bem como concorda com ele [8].

Faz-se uso da criptografia assimétrica para assinar digitalmente. Utiliza-se a chave privada para gerar as assinaturas: cifra-se o resumo criptográfico da mensagem. Para validar a assinatura, usa-se a chave pública: decifra-se o resumo criptográfico cifrado e compara-se com o resumo do conteúdo da mensagem calculado localmente. Se ambos forem iguais, então a assinatura é atestada. A Figura 2.1 ilustra como se dá a assinatura digital e sua verificação.

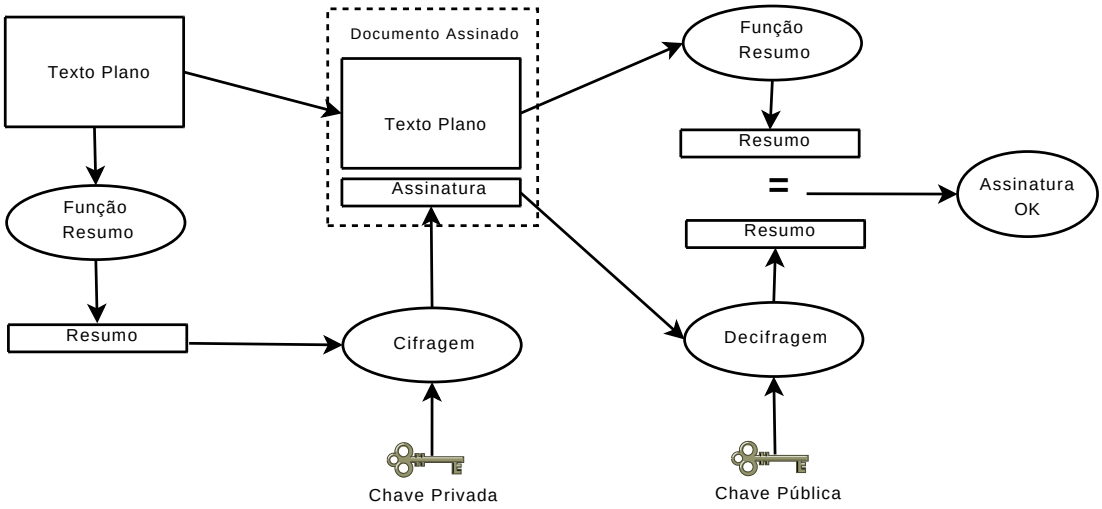

Figura 2.1: Assinatura digital.

Capítulo 3

Infra-estrutura de Chaves Públicas

Uma infra-estrutura de chaves públicas (ICP) é um sistema composto por protocolos, certificados digitais e entidades - certificadora e de registro - que visa prover serviços de suporte a aplicações de criptografia de chave pública. Por exemplo, através de uma ICP pode-se atestar a validade de um certificado digital.

3.1 Certificados Digitais

O certificado digital tem como objetivo principal informar qual indivíduo detém a chave privada correspondente a uma pública. Por esse motivo, também denomina-se certificado de chave pública.

Incluem-se ainda no certicado digital informações como endereços de contato do detentor da chave privada, datas de validade do certificado, entre outras informações. Buscando aferir veracidade a tais informações, uma terceira parte, denominada de autoridade certificadora, atribui seus dados e um serial ao certificado e, por fim, assina-o.

Dos padrões de certificados de chave pública disponíveis hoje, tem-se, por exemplo, o *X.509v3* [9], que é largamente difundido. Além das informações comuns a todo certificado digital mencionadas anteriormente, certificados nesse padrão ainda apresentam outros dados, como extensões que tornam-o mais flexível a aplicações.

Como exemplo de extensão, pode-se citar a *keyUsage*, que informa para que fins pode ser usada a chave privada relacionada ao certificado [9].

3.2 Lista de Certificados Revogados

Como os certificados, a lista de certificados revogados (LCR) também é emitida - assinada - por uma autoridade certificadora. A LCR tem como objetivo publicar quais certificados não-expirados e emitidos por uma AC deixaram de ser válidos [5].

O conteúdo da LCR é definido pela RFC3280 [9] e contém, principalmente, informações da AC emissora, uma lista de seriais de certificados revogados e a assinatura da AC emissora. Além disso, é possível a inserção de extensões à LCR com o objetivo de flexibilidade.

3.3 Autoridade Certificadora

A autoridade certificadora (AC) é a base de toda infra-estrutura de chaves pública. A AC se compõe de equipamentos de hardware, softwares e as pessoas que a operam [5].

A AC utiliza sua chave privada para emitir certificados digitais a outras partes - usuários finais e outras ACs, por exemplo - atestando assim a identidade destes. Por fim, terceiros que confiam na AC podem, consecutivamente, confiar nos certificados emitidos pela AC desde que a assinatura dessa seja válida.

A AC é conhecida por dois atributos: seu nome e chave pública [5]. Além disso, ela desempenha as seguintes função dentro de uma ICP:

- emitir certificados (assiná-los);
- manter informações sobre o estados dos certificados e emitir LCR;
- publicar certificados (não-expirados) e LCRs;

- manter arquivos de informação de status de certificados expirados ou revogados

Dependendo da demanda por certificados, pode se tornar inviável para uma AC atender todos as requisições dentro de uma ICP. Neste momento, é interessante que as responsabilidades da AC sejam delegadas a outras entidades. Por exemplo, pode-se delegar a uma Autoridade de Registro as funções de verificação de identidade dos requerentes de certificados. Por último, sugerem-se também que as operações de emissão de certificados sejam distribuídas entre outras ACs: subjulgadas a AC principal. Conceitualmente, demomina-se a última de Autoridade Certificadora Raiz, e as demais como Autoridades Certficadoras Intermediárias ou Finais.

3.4 Autoridade de Registro

Como a AC, a autoridade de registro (AR) se compõe de equipamentos de hardware, software e pessoas que a operam. Todavia, diferente da AC, uma única pessoa pode operar uma AR. Pode-se entender a AR como uma entidade responsável por verificar a identidade e outras informações pertinentes aos usuários que desejam solicitar um certificado à AC [10].

A AC confia na AR e delega a esta a função de verificar a validade dos dados submetidos pelas entidades que solicitam um certificado. Uma vez validadas as informações submetidas, a chave pública e os dados do requerente são compilados na forma de uma requisição de certificado, que é assinada pela AR e enviada a AC. Esta valida a assinatura da AR e, sem nenhuma verificação adicional, emite o certificado.

3.5 Políticas de Certificação

As políticas de certificação definem um conjunto de regras sobre uma ICP que estabelecem requisitos que devem ser cumpridos pelos indíviduos envolvidos - operadores de ACs e RAs, usuários de certificados. Por exemplo, define-se um conjunto de procedimentos de segurança que toda AC de uma ICP deve efetuar. Assim,

é através da análise dos documentos de políticas que terceiras partes decidem se um certificado é confiável e aplicável.

Como mencionado, as políticas são definidas em documentos - Políticas de Certificação (PC) e Declaração de Práticas de Certificação (DPC) - e a relação entre eles é definida no diagrama UML ilustrado na Figura 3.1.

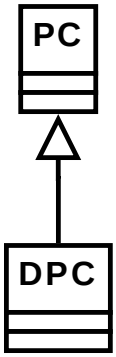

Figura 3.1: Diagrama UML da Relação entre PC e DPC.

A PC é um documento de alto nível em que se definem as práticas de segurança para emissão de certificados bem como o mantenimento das informaçõe de certificado [5]. Descrevem-se, dessa maneira, a operação de uma AC, como ainda as responsabilidades dos usuários ao solicitar, usar e manipular certificados. Por exemplo, a PC pode definir que o requerente deve dirigir-se pessoalmente a AR e identificar-se. Todavia, a PC não especifica como as regras de segurança devem ser implementadas, como por exemplo, quais documentos o requerente deve apresentar à AR ao solicitar um certificado.

Assim, cabe a Declaração de Práticas de Cerficação - documento altamente detalhado - descrever como uma determinada AC implementa as regras definidas na PC. Tomando-se novamente o exemplo do requerente identificando-se perante a AR, a DPC pode estabelecer, num caso hipotético, que o indivíduo deve apresentar uma cópia autenticada em cartório do documento de identidade e de CPF, bem como do título de eleitor.

Portanto, este é o documento que deve ser analisado por auditores para se

validarem as operações de uma AC levando-se em consideração uma PC. Esta deve ser genérica e seu uso estimado por vários anos [5]. Por último, a PC deve ser públicada, sendo que o mesmo não é necessário para a DPC.

3.6 Módulo Público

O módulo público é a interface entre os usuários e a ICP: através dele um indivíduo pode solicitar certificados.

3.7 Modelos de Transação

A comunicação entre usuários de certificados e autoridades certificadoras é estabelecida através de transações, que podem váriar de acordo com o número de entidades envolvidas e o grau de confiabilidade envolvido. O exemplo mais simples é o de transação direta entre AC e usuário certificado: ambos comunicam-se por troca de mensagens assinadas. Esta abordagem é largamente utilizada para implementar requisições básicas de certificados e de revogação, e é conhecida como *modelo de transação de duas partes* [5].

Por requisitos de simplicidade a AC Correio implementa a abordagem citada anteriormente: a comunicação entre usuário de certificado e AC Correio se dá por troca de mensagens, sem a presença da AR. No entando, vale ressaltar que assinam-se apenas as requisições de certificados, bem como os próprios certificados emitidos. Por outro lado, mensagens relativas à protocolos de identificação do usuário não são assinadas por ambas das partes.

A Figura 3.2 ilustra o modelo de transação implementado na AC Correio.

Vale citar, contudo, que a comunicação direta entre AC e usuário de certificado pode apresentar deficiências de segurança. Dessa maneira, sugere-se o *modelo de transação de três partes*. Nesta abordagem insere-se o papel da Autoridade de Registro (AR), responsável por verificar a veracidade das informações fornecidas pelo

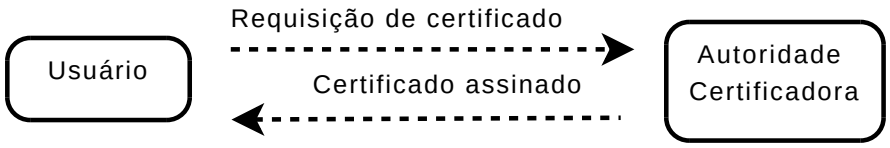

Figura 3.2: Modelo de transação da AC Correio.

requerente de certificado.

O último modelo de transação citado possui variantes de acordo com os tipos de interações entre as entidades envolvidas. O caso mais simples segue os passos discriminados abaixo e é ilustrado na Figura 3.3:

1. O requerente de certificado gera seu par de chaves e então apresenta-se pessoalmente à AR;
2. a AR valida os credenciais do requerente e então gera uma requisição assinada e a envia à AC;
3. a AC verifica o certificado da AR bem como a respectiva assinatura na requisição recebida;
4. a AC processa a requisição e envia uma resposta assinada à AR;
5. a AR verifica o certificado da AC bem como a respectiva assinatura na resposta recebida;
6. a AR fornece o certificado ao requerente.

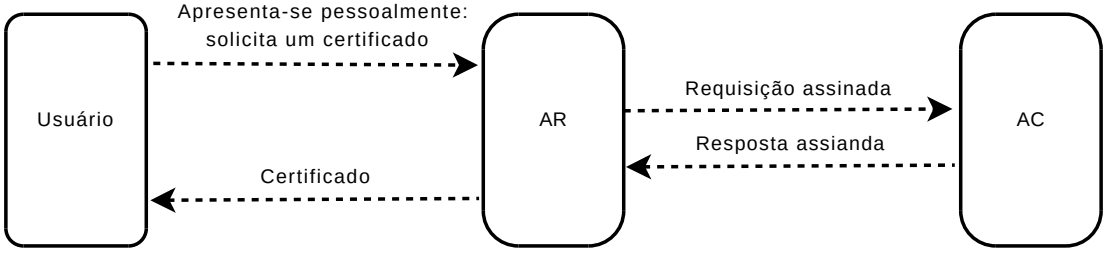

Figura 3.3: Modelo de transação de três partes.

Capítulo 4

Tecnologias Utilizadas

4.1 OpenSSL

O OpenSSL [11] é uma biblioteca de domínio público - gratuita - que pode ser utilizada livremente para desenvolvimento de aplicações comerciais ou não. É mantida por um grupo de voluntários espalhados pelo mundo, cujo obtivo é desenvolver uma biblioteca criptográfica robusta e de vastas funcionalidades.

São implementados nesta biblioteca protocolos para camadas de sockets seguros (SSL v2/v3) e de transporte seguro (TLS v1), além de funcionalidades criptográficas de propósito geral.

Na implementação da AC Correio utilizou-se indiretamente essa biblioteca criptográfica, uma vez que a LibCryptoSec [12] abstrai o OpenSSL fornecendo uma interface de programação orientada a objeto.

4.2 LibCryptoSec

A LibCryptoSec [12] é um biblioteca desenvolvida no paradigma de Orientação a Objeto que abstrai grande parte das funcionalidades providas pelo OpenSSL. Além disso, tal biblioteca foi implementada em C++ para manter a compatilibidade com o OpenSSL e proporcionar uma API mais simples e intuitiva.

Pela complexidade e documentação escassa da API do OpenSSL decidiu-se adotar a LibCryptoSec no desenvolvimentos da operações criptográficas da AC Correio.

4.3 PHP

O PHP é uma linguagem de programação interpretada largamente utilizada no desenvolvimento de páginas para Web, proporcionando dinamismo e integração com a linguagem de marcação HTML [13]. Além disso, apresenta uma API simples e de bem documentada, valendo citar o excelente suporte a banco de dados, fatores que basearam a escolha pelo PHP para desenvolver o módulo público da AC Correio.

4.4 Zend Framework

O Zend Framework [14] é uma biblioteca PHP de alta qualidade e de código aberto para o desenvolvimento de aplicações para Web. Possui funcionalidades extremamente úteis a AC Correio e que não disponíveis nativamente no PHP, como abstração de banco de dados, filtros de comandos SQL, envio de emails, entre outros.

4.5 C++

O C++ é uma linguagem de programação proposta por Bjarne Stroustrup em seu trabalho no *AT&T Bell Laboratories* que tinha como objetivo incorporar características da linguagem *Simula* - abstração de dados, por exemplo - ao *C*, que é uma linguagem estruturada. Sua primeira versão chamou-se de *C with Classes* e foi utilizada pela primeira vez em 1980 [15].

Como características do C++ vale citar citar seu modelo de memória e computação que se aproxima muito à maioria dos computadores [15]. Adicionalmente, a linguagem provê um mecanismo poderoso e flexível de abstração de dados, o que

permite ao programador introduzir e utilizar novos tipos de objetos que se aproximam aos conceitos da aplicação.

Por fim, a escolha pelo uso do C++ para desenvolvimento do módulo criptográfico da AC Correio se deve muito a dependência da biblioteca LibCryptoSec, desenvolvida na mesma linguagem.

4.6 Qt4

O Qt4 [16] é uma biblioteca de classes em C++ que encapsulam toda a infra-estrutura necessária para o desenvolvimento de aplicações finais. Apresenta uma API madura e orientada a objeto que inclui, entre uma rica gama de funcionalidades, facilidades de XML e de banco de dados: características vantajosas aproveitadas no módulo criptográfico.

4.7 JavaScript

JavaScript [17] é uma linguagem interpretada criada pela Netscape em 1995 e é largamente utilizada em páginas web. Entre as características particulares de JavaScript vale citar que um script criado nesta linguagem é executado no navegador de quem visita a páginas, sendo assim útil para promover interação do site com o usuário.

4.8 HTML

O HTML - sigla do ingles *HyperText Markup Language* - uma linguaguem de marcação largamente utilizada na publicação de documentos na Web [18]. Com o os desta desenvolvem-se páginas cujo material é multimída: texto, som, vídeos, entre outros.

4.9 Linux

O Linux [19] é um sistema operacional *POSIX* [20] gratuito originalmente criado por Linus Torvalds com a assistência de desenvolvedores ao redor do Mundo. Entre as vantagens de sua adoção para o desenvolvimento estão a sua licença de uso - gratuita e irrestrita - bem como diversas ferramentas de desenvolvimento extremamente poderosas como o compilador C/C++ GNU GCC [21].

4.10 UML

O UML - sigla do Inglês *Unified Modeling Language*^{TM} - é uma linguaguem de especificação mais utilizada pela comunidade de desenvolvimento de *software* para modelar aplicações representando, gráficamente, suas estruturas e comportamento. As vantagens de se utilizar UML para descrever a AC Correio verificam-se na clareza da especificação, bem como nas facilidades apresentadas pelas ferramentas UML como o JUDE [22]- elaboração de diagramas de classes, de fluxo, de casos de usos, entre outras.

4.11 XML

O XML [23] - sigla do Inglês *Extensible Markup Language* - é uma linguaguem de marcação flexível derivada do SGML (ISO 8879). Entre as vantagens da adoção do XML na AC Correio verificam-se as facilidades de descrição de dados, extremamente útil na criação e leitura do arquivo de configuração dos módulos criptográfico e público.

4.12 PostgreSQL

O PostgreSQL [24] é um banco de dados relacional poderoso e de código aberto cujo desenvolvimento possui mais de 15 anos de maturidade, fato comprova sua

forte reputação em corretude, integridade de dados e confiabilidade. Além disso é implementado sobre o SQL - sigla do Inglês *Structured Query Language* - uma linguagem de pesquisa declarativa.

Das vantagens da adoção do PostgreSQL na AC Correio pode-se citar a licença livre desse, que permite seu uso para aplicações comerciais ou livres. Além disso, apresenta rica documentação e ferramentas práticas de manuseio das bases de dados.

Capítulo 5

AC de Correio Eletrônico

Buscar uma solução em que um usuário final possa obter e utilizar um certificado digital de maneira simples: este é o requisito que norteia a concepção da AC Correio. Assim, buscam-se seguir alternativas que primem pela simplicidade de interação entre usuário de certificado e AC: portanto, escolhe-se o *modelo de transação de duas partes* abordado na Seção 3.7.

Assim, concebe-se o protocolo desafio-resposta - alicerce do sistema de identificação de usuários finais - que implementa o modelo de transação escolhido. Em seguida, definem os módulos que compõem a solução, estabelecendo a interação entre esses bem como suas respectivas funcionalidades.

5.1 Projeto

5.1.1 Requisitos Funcionais

Visando o projeto de uma solução simples para automatizar a emissão de certificados, definiram-os seguintes requisitos funcionais: gerais, de chave e de uso da aplicação

5.1.1.1 Requisitos Funcionais Gerais

- A AC Correio cria sua própria requisição de certificado, que deve ser assinada por uma AC a qual é subordinada;
- a AC Correio importa seu certificado e seu respectivo caminho de certificação;
- a AC Correio trabalha de maneira *online* - todas as requisições de certificado são assinadas automaticamente e publicadas;
- há um modelo de certificado a ser emitido para os requerentes. Este modelo deve ser flexível e editável;
- os certificados emitidos estão de acordo com as políticas de certificação definidas na Declaração de Práticas de Certificação;
- há um banco de dados pelo qual os módulos da AC Correio comunicam-se utilizando uma abordagem produtor-consumidor (uma entidade produz uma informação no banco de dados, que é consumida por outra entidade;
- o banco de dados armazena todas informações necessárias para a AC Correio operar: certificados da AC e os emitidos por ela, LCR, seriais;
- o banco de dados e os módulos da AC Correio são hospedados no mesmo computador;
- disponibiliza-se uma entidade administradora da AC Correio que efetua configuração do aplicativo e revogação de certificado através de uma interface amigável;
- a AC Correio emite, automaticamente, a LCR dentro de um período configurável;
- a LCR emitida é publicada no site web da AC Correio;
- o certificado da AC Correio, e seu caminho de certificacacao, é publicado no site web da AC Correio;

- a entidade administradora define na AC Correio quais domínios de correio podem solicitar certificados;

- são registrados no banco de dados os seguintes eventos: início e fim das atividades, emissão de certificado e de LCR, processo de solicitação de certificados, revogações e acessos da entidade administradora.

5.1.1.2 Requisitos Funcionais de Chaves

- A AC Correio opera com chaves RSA armazenadas em disco ou no HSM da ICPEDU;

- o solicitante de certificado tem opções de tamanhos para a chave a ser gerada localmente em seu browser.

5.1.1.3 Requisitos Funcionais de Uso da Aplicação

- Ao solicitar um certificado, o usuário acessa a interface web e fornece seu endereço de email, para o qual a AC Correio envia uma URL secreta;

- o usuário acessa sua conta de correio e abre a URL secreta, que o direciona a um site em que o browser gera um par de chaves, cria uma requisição de certificado e submete ao site, que armazena a requisição no banco de dados da AC Correio;

- a AC Correio assina todas as requisições submetidas ao banco de dados;

- o certificado emitido pela AC Correio é publicado no site web da AC Correio;

- o usuário tem seu certificado automaticamente instalado no browser ao acessar o site web da AC Correio;

- toda vez que um usuário, que já possui um certificado válido, solicita novamente um certificado, o atual é revogado e outro é emitido.

5.1.2 Requisitos Não Funcionais

Os requisitos não funcionais estabelecidos são:

- O banco de dados utilizado é o PostgreSQL 8 ou superior;
- O sistema operacional que hospeda a AC Correio é o Ubuntu Linux 7.04 ou superior;

5.1.3 Protocolo de Desafio-resposta e de Emissão de Certificado

O protocolo de desafio resposta tem como objetivo comprovar a posse de um usuário sobre uma conta de correio eletrônico, fato suficiente para qualificar um indíviduo candidato a solicitar um certificado. Em outras palavras, este protocolo realiza a função de uma AR, que, no caso da AC Correio, não existe diretamente.

A inexistência da AR é o foco da AC Correio: automatizar a emissão de certificados. Dessa maneira, atendem-se usuários em larga escala, uma vez que esses não tem mais a necessidade de cumprir requisitos burocráticos típicos perante a AR - apresentar-se pessoalmente munido de documentos, por exemplo.

O protocolo segue os passos definidos abaixo, também ilustrados na Figura 5.1:

1. o usuário submete seu endereço de correio à AC Correio no sítio http://ac-correio.icpedu.rnp.br.
2. a AC Correio envia um endereço URL secreto à conta de correio submetida no item anterior.
3. o usuário acessa sua conta de correio e abre a URL secreta, onde prosseguem-se os demais passos para solicitação de certificado.

Uma vez confirmada a posse do endereço de correio, dá-se início ao protocolo de emissão de certificado, descrito a seguir e ilustrado na Figura 5.2:

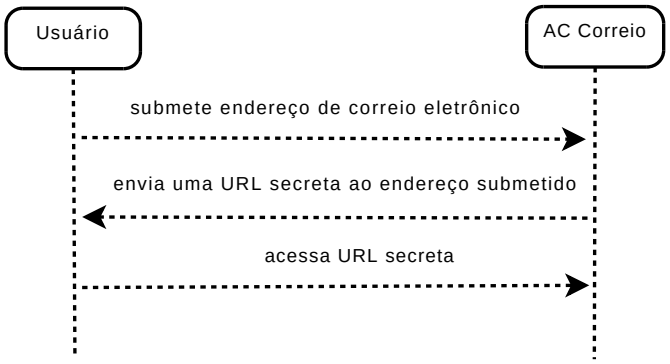

Figura 5.1: Protocolo de desafio-resposta da AC Correio.

1. o usuário gera seu par de chaves RSA;
2. o usuário submete a chave pública à AC Correio;
3. a AC Correio emite um certificado para a conta de e-mail submetida;
4. a AC Correio notifica o usuário informando a URL em que se encontra o certificado emitido;
5. o usuário recebe a notificação e abre a URL em que se encontra seu certificado;
6. o certificado é instalado no navegador web.

É importante ressaltar que a verificação da identidade do usuário se dá quando este tem acesso a URL secreta. Todavia, reforça-se tal verificação ao enviar a URL em que certificado está disponível: outra informação que não é pública. Isto garante que outros indíviduos não terão acesso ao certificado emitido e, consequentemente, não poderão utilizá-lo sem a autorização do titular.

Nota-se a emissão do certificado é somente efetuada após a execução do protocolo de desafio-resposta. Isto protege a AC Correio, sendo que as operações criptográficas de assinatura digital involvidas na emissão de certificados exigem uma porção considerável de recursos computacionais.

Por último, vale citar as restrições temporáis do protocolo de emissão de certificado. É interessante que o tempo entre os passos 2 e 3 seja o mais curto possível.

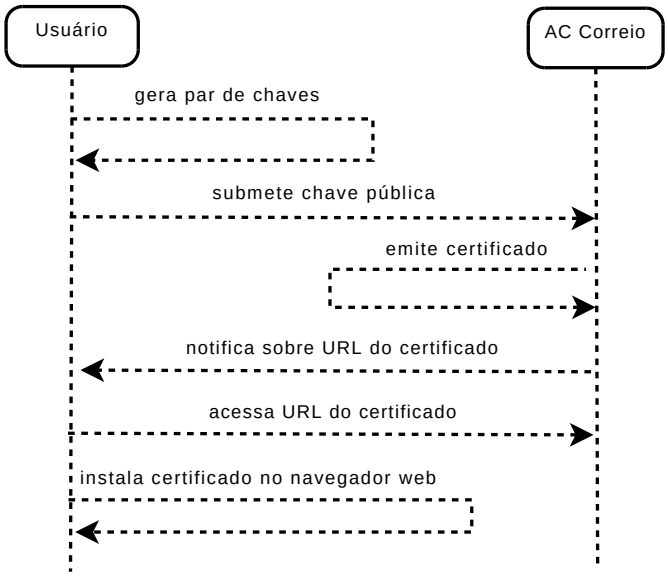

Figura 5.2: Protocolo de Emissão de Certificado pela AC Correio.

Isto porque o requerente só pode exportar seu certificado após instalá-lo no navegador web que gerou as chaves, que pode estar hospedado em um computador público, pelo qual vários indíviduos concorrem para utilizar. Tome-se como exemplo um usuário que requisita um certificado através do computador de um *cybercafe*. É interessante que o certificado seja emitido rapidamente, pois assim o usuário poderia instalar o certificado no navegador e, consequentemente, exportá-lo para uma mídia removível e, assim, deixar o estabelecimento.

5.1.4 Organização Modular

A AC Correio pode ser organizada modularmente, agrupando-se funcionalidades bem definidas em entidades. Portanto, identificam-se três módulos:

- módulo de persistência;
- módulo público;
- módulo criptográfico.

Os módulos público e criptográficos trabalham independemente: sem comunicação direta entre eles. A interação entre ambos ocorre por meio do módulo de persistência - um banco de dados. Cada dos dois primeiros consulta o banco em busca de dados que precisam ser processados e, após realizadas as operações necessárias sobre estes, os armazena no módulo de persistência. Esta configuração ilustra uma abordagem *produtor-consumidor* [25] e tem como vantagens a simplicidade, deixando a cargo do banco de dados resolver os problemas decorrentes da computação paralela, como sincronia entre processos e condições de disputa.

É importante ressaltar que, na abordagem de módulos adotada e ilustrada na Figura 5.3, a segurança do sistema está atrelada, em grande parte, na forma como é implementada a comunicação entre módulos público e criptográfico com o banco de dados. Embora este trabalho não apresente nenhum mecanismo de segurança no canal de comunicação, sugerem-se o uso de túneis SSL para a transmissão dos dados, bem como utilizar assinatura digital na operações de consulta ao banco de dados. Tais exemplos são observados no software NewPKI [1].

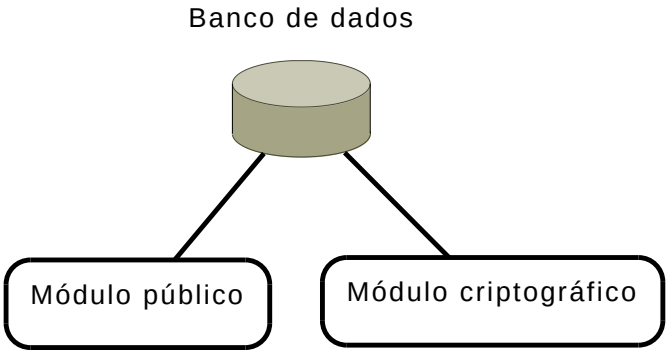

Figura 5.3: Organização modular da AC Correio.

5.2 Módulo Criptográfico

Este módulo é responsável por todas as operações criptográficas realizadas pela AC Correio: emissão de certificados e de LCRs. Portanto, sua interação com o

usuário é mínima - exceto com o administrador da AC Correio - e, por esse motivo, sugere-se que este módulo seja implementado na forma de um *daemon*: processo que roda em segundo plano, permanecendo inativo por maior parte do tempo [25].

A execucação principal do módulo criptográfico segue o fluxograma ilustrado na Figura 5.4 e é brevemente detalhada em seguida.

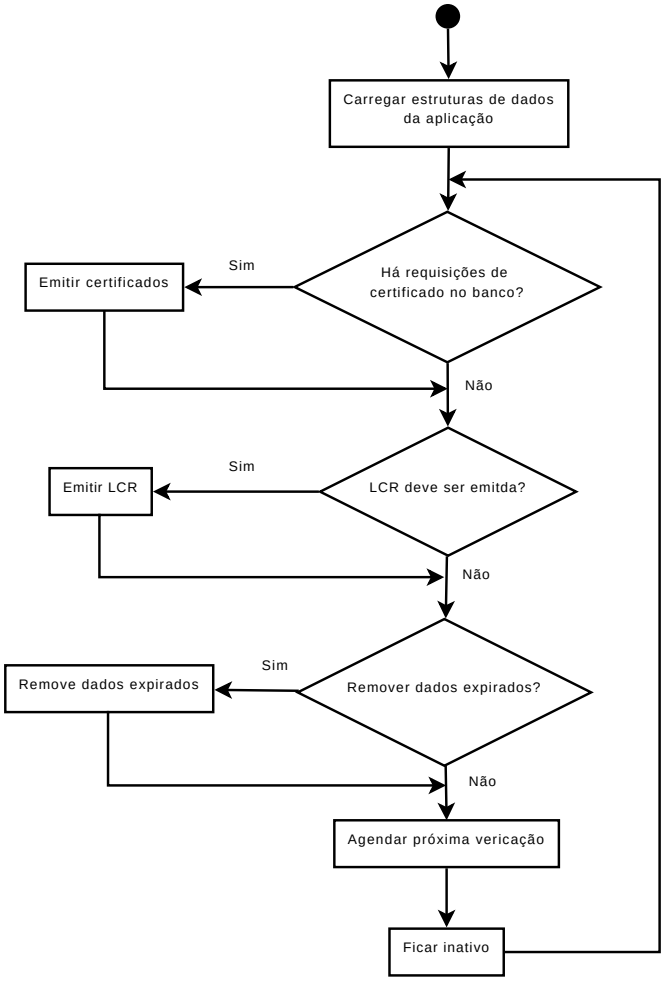

Figura 5.4: Fluxograma da execução do módulo criptográfico.

- inicialização: *daemon* deve carregar todas as estruturas de dados e informações necessárias para as operações criptográficas da AC Correio;
- verificar requisições de certificados: neste momento o *daemon* obtem todas as requisições de certificado no banco e as assina, emitindo os certificados;

- emitir LCR: se for o momento de emitir a LCR, o *daemon* obtem no banco todos os seriais de certificados revogados e emite a LCR;
- remover dados expirados: o *daemon* remove periodicamente dados expirados no banco, como por exemplo, desafios não confirmados;
- agendar próxima verificação: o *daemon* calcula o tempo que permanecerá inativo até que seja momento de realizar alguma das verificações citadas anteriomente;
- ficar inativo: o *daemon* libera os processador do computador hospedeiro.

5.2.1 Organização Modular

O módulo criptográfico apresenta a organização modular ilustrada na Figura 5.5, que é detalhada em seguida:

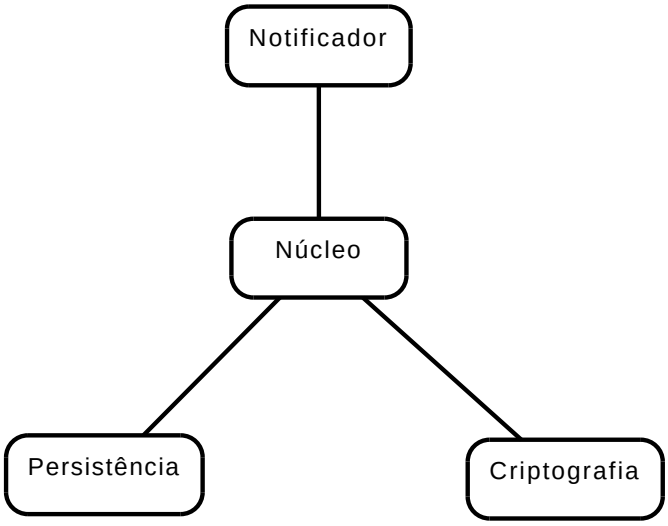

Figura 5.5: Organização modular do módulo criptográfico.

- núcleo: elemento central do módulo criptográfico em que ocorre a execução descrita na Seção 5.4;
- notificador: elemento que envia e-mails aos usuários notificando-os da emissão de seus certificados;

- persistência: elemento que realiza a interface entre módulo criptográfico e banco de dados;

- criptografia: elemento responsável pelas operações criptográficas: emissão de certificaos e de LCR.

5.2.2 Par de Chaves da AC Correio

O módulo criptográfico utiliza um par de chaves RSA de tamanho definido pelo usuário para realizar as emissões de certificados e LCR. A chave pública está contida no certificado da AC Correio e fica armazenada no banco de dados. Já a chave privada pode ser armazenada de duas maneiras: em disco do computador hospedeiro ou em um HSM da ICPEDU [26].

A chave privada armazenada em disco é a forma mais simples, contudo a sua segurança está diretamente relacionada à segurança do computador hospedeiro, podendo ser copiada por qualquer indivíduo que tenha acessa a ele. Por outro lado, quando utiliza-se armazenamento de chave privada em HSM tem-se um elevado grau de segurança, uma vez que se torna praticamente impossível o compromentimento da chave por meio do roubo desta.

Para utilizar-se chave privada armazenada em HSM faz-se uso de uma *engine* do OpenSSL - no caso, uma interface que implementa funções criptográficas disponíveis no HSM. Vale citar que esta é uma *engine* dinâmica: não é nativa do OpenSSL e deve ser carregada antes de ser utilizada.

Da configuração da *engine*, é importante definerem-se:

- identificador atribuído à *engine*;

- objeto binário (*shared object*) da *engine*;

- endereço de rede e porta do HSM;

- idenficador atribuído à chave privada.

Por último, tais informações podem ser conferidas no arquivo de configuração do módulo criptográfico no Anexo 8.1.

5.2.3 Modelo de Certificados Emitidos

O módulo criptográfico emite certificados *X509v3* [9]. Estes seguem um padrão definido para os campos *DN* do *Subject* e para as extensões.

O *Subject* reflete informações do titular do certificado - no caso, quem solicitou um certificado à AC Correio. Os dados desse campo são definidos com o modelo ilustrado na Tabela 5.1, que reflete as definições da DPC da AC Correio 6.1.6.

Tabela 5.1: Campos de Nome dos Usuários Finais

Campo	Conteúdo
C	valor definido no modelo
S	valor definido no modelo
L	valor definido no modelo
O	valor definido no modelo
O	valor definido no modelo
CN	Endereço de e-mail do requerente

Além disso, definem-se as extensões do modelo de certificados. Entre elas, destacam-se *keyUsage* e *extKeyUsage*, que restringem o propósito de uso dos certificados emitidos pela AC Correio - e-mail seguro e autenticação. Todas as extensões estão discriminadas na Tabela 5.2.

Por último, pode-se verificar o modelo de certificado analisando-se o arquivo de configuração do módulo criptográfico no Anexo 8.1.

5.2.4 Emissão de Certificado

A emissão de certificado inicia-se com a recuperação da requisição de certificado presente no banco de dados. Esta pode estar codificada no formato *PKCS#10* [27] ou *SPKAC* [11] e, entre outras informações, contém a chave pública do requerente. Além disso, deve-se obter do banco serial do último certificado emitido e o certificado da AC Correio.

Tabela 5.2: Extensões do Certificado Emitido pela AC Correio

Extensão	**Descrição**	**Crítica**
keyUsage	contém apenas os bits digitalSignature(0), non-Repudiation(1), dataEncipherment(3) ligados.	Sim
extKeyUsage	contém os OIDs 1.3.6.1.5.5.7.3.2 (id-kp-clientAuth) e 1.3.6.1.5.5.7.3.4 (id-kp-emailProtection).	Sim
cRLDistributionPoints	contém o endereço na Web onde se obtém a LCR emitida pela AC Correio.	Não
CertificatePolicies	especifica o Object Identifier (OID) da PC da AC Correio e o atributo id-qt-cps com o endereço na Web da DPC.	Não
AuthorityKeyIdentifier	contém o hash da chave pública da AC Correio.	Não
SubjectKeyIdentifier	contém o hash da chave pública do usuário final.	Não
SubjectAlternativeName	contém o endereço de correio eletrônico do usuário final.	Não

Em seguida, a requisição de certificado é lida. Se houver algum problema de integridade, aquela é descartada e o requerente é notificado. Caso contrário extraem-se da requisição a chave pública e o *DN*. Logo é conferida a assinatura da requisição para garantir que o requerente realmente tem a posse da chave privada.

Neste momento, emite-se o certificado do requerente, inserindo-se a chave pública do requerente. O *DN* do *Subject* é montado utilizando-se o campos *CN* do *DN* da requisição e os demais - *C*, *S*, *L*, *O*, *OU*, por exemplo - são preenchidos de acordo com o modelo de certificado. No passo seguinte, é inserido o campo *Issuer*, que é extraído do certificado da AC Correio e definem-se a versão do certificado (2), a data de validade - um ano a partir da data de emissão - e o serial do certificado: valor numérico incrementado recuperado do banco de dados como mencionado anteriormente.

Adiante, inserem-se todas as extensões definidas no módelo de certificado

discriminado na Tabela 5.2. Neste momento o certificado é assinado pela chave privada da AC Correio. Em seguida, o certificado é armazenado em formato *Base64* [28] no banco de dados, bem como seu serial e a requisição é apagada daquele. Por fim, o requerente é notificado, através de um e-mail, que seu certificado foi assinado e encontra-se disponível no módulo público da AC Correio, descrito na Seção 5.3.

5.2.5 Emissão de LCR

Inicialmente, verifica-se a existência de requisições de revogação no banco. Em caso positivo, insere-se na tabela de certificados revogados o serial do certificado para que se solicita revogação, bem como apaga-se o último da tabela de certificados emitidos. Logo, remove-se a requisição de revogação do banco.

No passo seguinte, obtém do banco os seriais de certificados revogados, bem como a data e o motivo da revogação. Também recupera-se o serial da última LCR emitida e o certificado da AC Correio.

De posse dos dados citados acima, a LCR é montada. Definem-se a versão e serial da LCR, 1 e o valor incrementado do serial da última LCR respectivamente. Insere-se também o período de validade da LCR, que é calculado a partir da data de emissão mais um valor definido na configuração do módulo público, ilustrado no Anexo 8.1.

Em seguida, inserem-se na LCR todos os seriais dos certificados revogados e ainda não expirados obtidos do banco, bem como a data e o motivo de revogação. Uma vez realizado o passo anterior, a LCR é assinada e guardada no banco em formato *Base64*. Por último, atualiza-se no banco o serial da última LCR emitida.

5.2.6 Limpeza de Dados Expirados

Periodicamente removem-se do banco de dados informações cujo prazo de validade expirou. Essas podem ser: desafios do modulo publico não confirmados no prazo de 24 horas e certificados, revogados ou não, que estejam expirados.

5.2.7 Implementação

O módulo criptográfico é implementado na linguagem C++ - Seção 4.5 - fazendo uso da biblioteca Qt4 (Seção 4.6) e segue-se o Paradigma de Orientação a Objetos, o que é ilustrado no diagrama de classes de análise no Anexo 8.2.

Sob uma visão mais detalhada do diagrama do Anexo 8.2 tem-se, ao centro, a classe núcleo: *SystemController*, disponível no Anexo 8.3. A partir desta são instanciadas todas as outras classes do módulo criptográfico. A partir desta que ocorrem as operações principais mencionadas neste capitulo: emissão de certificados e LCR, bem como a limpeza de dados expirados. As demais classes são sucintamente descritas a seguir:

- *Persistence*: faz a interface entre o núcleo do módulo público e o banco de dados (Anexo 8.4);
- *Cripto*: realiza as operações criptográficas: emissão de certificados e LCR (Anexo 8.5);
- *Scheduler*: determina por quanto tempo o *daemon* deve permancer inativo até que seja hora de realizar alguma atividade: emitir certificado ou LCR, ou remover dados expirados (Anexo 8.13);
- *ConfigData*: encapsula as configurações do módulo público contidas no arquivo de configuração (Anexo 8.6);
- *ConfigurationFileReader*: disponível no Anexo 8.7, é responsável por ler o arquivo de configuração do Anexo 8.1 em formato *XML* (Seção 4.11) e por popular o objeto da classe anterior;
- *Utils*: realiza conversões de *strings* para tipos numéricos como *int*, *float* (Anexo 8.14);
- *AbstractLogger*: classe virtual pura que define a operação de registro de eventos (Anexo 8.8);

- *ConsoleLogger*: implementa *AbstractLogger* e registra eventos no terminal em que roda o *daemon* (Anexo 8.9);

- *FileLogger*: implementa *AbstractLogger* e registra eventos em um arquivo em disco (Anexo 8.10);

- *Notifier*: notifica, através do envio de e-mail, os usuários a respeito da emissão de seus certificados ou caso a requisição apresente problemas de integridade (Anexo 8.11);

- *LibCryptoSecException*: classe da biblioteca LibCryptoSec - descrita na Seção 4.2 - que é lançada quando ocorrem erros referentes às operações criptográficas (Anexo 8.15);

- *FileSystemException*: implementa a classe anterior e define a exceção lançada quando há problemas de leitura ou escrita de arquivos (Anexo 8.16);

- *PersistenceException*: implementa a classe *LibCryptoSecException* e define a exceção lançada quando há problemas com o banco de dados: conexão, erro de sintaxe *SQL*, entre outros (Anexo 8.17);

- *SyntaxException*: implementa a classe *LibCryptoSecException* e define a exceção lançada quando há erros de sintaxe na leitura do arquivo de configuração do módulo criptográfico (Anexo 8.18);

- *ReqBuilder*: gera uma requisição de certificado para a AC Correio (Anexo 8.12).

5.3 Módulo Público

O módulo público é a interface do requerente de certicado com a AC Correio: é através dela que solicitam-se e obtém-se certificados, além de ser o ponto de públicação da LCR e do certificado da AC Correio. No caso deste trabalho, o módulo público é caracterizado por uma aplicação Web.

É no módulo público que o protocolo de desafio-resposta - Seção 5.1.3 - é implementando, bem como os passos seguintes após a confirmação do desafio: fato que se consolida no momento que uma URL secreta o módulo público é acessada pelo requerente de certificado. Depois desse ponto, o usuário gera, em seu navegador, um par de chaves e submete a pública ao módulo público.

Uma vez emitido o certificado - como descrito na Seção 5.2.4 - o usuário acessa a URL do módulo público informada através da notificação enviada pelo módulo criptográfico. Neste endereço, o módulo público recupera o certificado do banco de dados e entrega ao navegador web do requerente, que associa o certificado à respectiva chave privada gerada previamente, instalando-o automaticamente. A partir deste momento o usuário pode utilizar o certificado e a respectiva chave privada para envio seguro de e-mails .

Os passos descritos acima, juntamente com o protocolo de desafio-resposta e as operações módulo criptográfico, são ilustrados na Figura 5.6

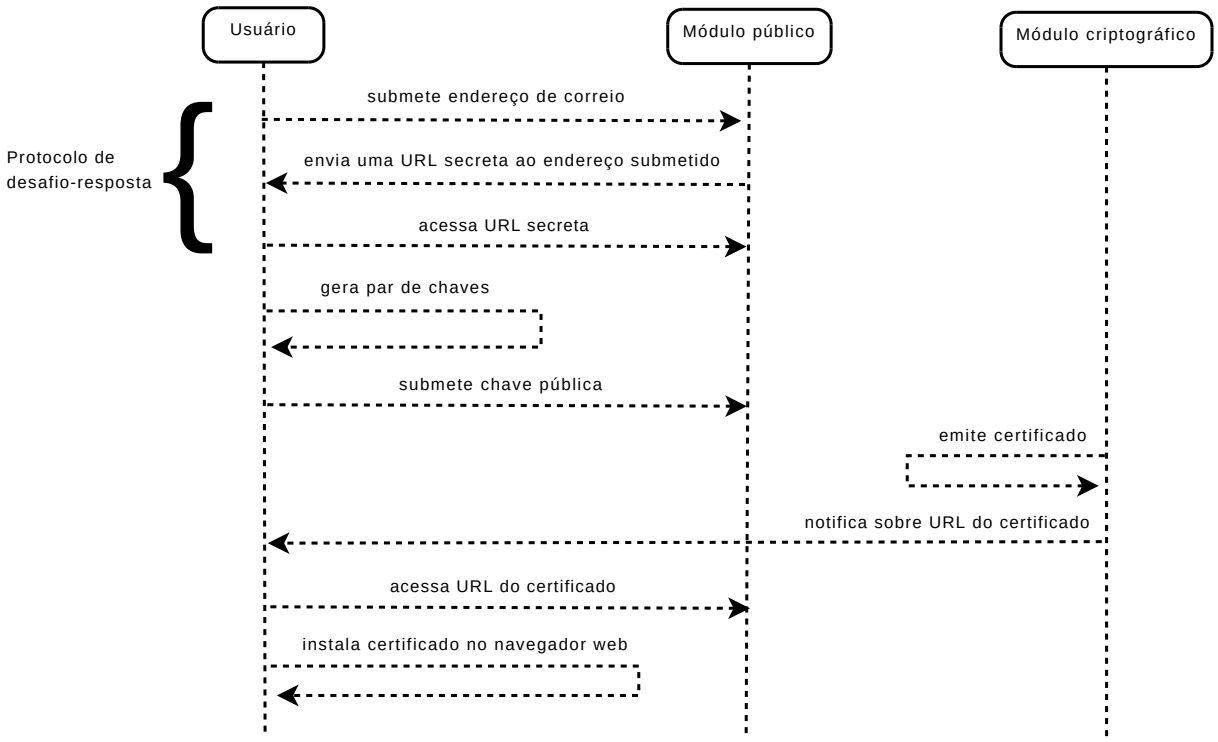

Figura 5.6: Passos para obtenção de certificado.

5.3.1 Organização modular

O módulo público apresenta a organização modular ilustrada na Figura 5.7, que é detalhada em seguida:

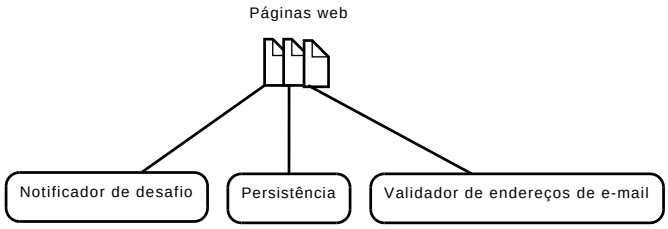

Figura 5.7: Organização modular do módulo público.

- Páginas web: documentos web que reenderizam telas com funcionalidades de solicitação e acesso a certificado e a LCR, além de geração de par de chaves, entre outros;
- Persistência: faz a interface entre as páginas web e o banco de dados da AC Correio;
- Notificador de desafio: envia para o endereço de e-mail do requerente a URL secreta de confirmação;
- Validador de endereço de e-mail: verifica se o e-mail do requerente tem direito a solicitar certificado à AC Correio.

5.3.2 Páginas

As páginas web do módulo público são organizadas como ilustrado na Figura 5.7, que é detalhada em seguida:

- Principal: página inicial do módulo público;
- Submissão de endereço de e-mail: página em que o requerente submete seu e-mail para iniciar a solicitação de certificado à AC Correio;

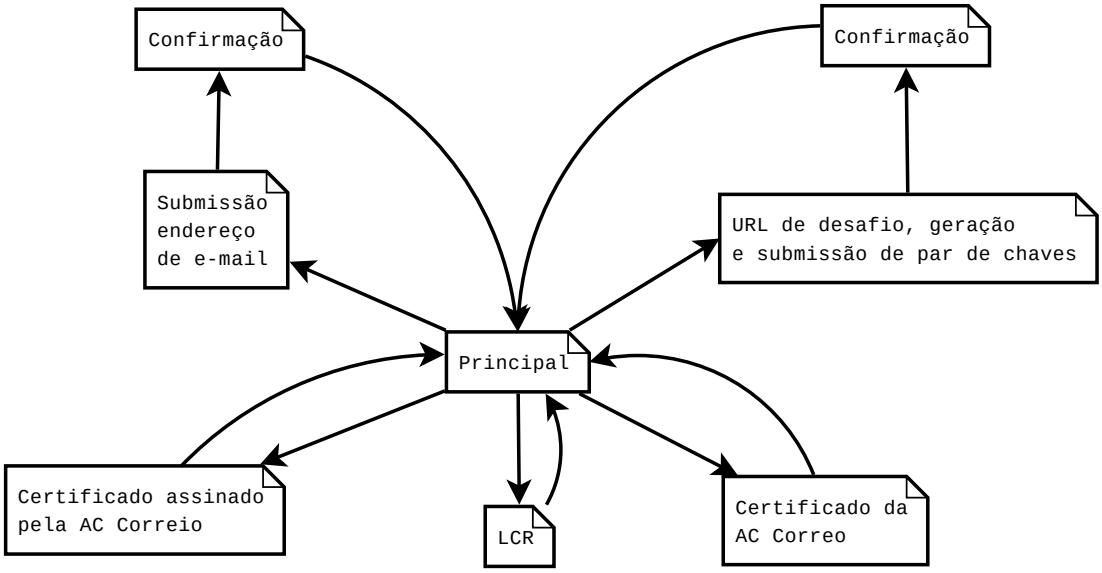

Figura 5.8: Diagrama de navegaçao do módulo público.

- URL de desafio, geração e submissão do par de chaves: página em que se finaliza o protocolo de desafio-resposta e gera-se o par de chaves submetido, caracterizando a última etapa do processo de solicitação de certificado à AC Correio;
- Certificado assinado pela AC Correio: página em que o usuário obtém seu certificado assinado;
- LCR: página em que se obtém a última LCR emitida pela AC Correio;
- Certificado da AC Correio: página em que se obtém o certificado da AC Correio;
- Confirmação: informa se a operação anterior foi realizada com sucesso.

5.3.3 Verificação de Endereço de E-mail

O módulo público define quais usuários que podem solicitar certificados através de domínio de endereços de e-mail - o valor literal após o símbolo “@”. Os domínios autorizados são previamente definidos nas configurações e a verificação desses ocorre durante a submissão do endereço de correio eletrônico pelo usuário, antes do início do protocolo de desafio-resposta.

5.3.4 Geração da URL Secreta de Desafio

Como mencionado anteriormente, o requerente deve finalizar o protocolo de desafio-resposta acessando uma URL secreta enviada ao seu e-mail, fato que caracteriza a confirmação do desafio. Aquela é gerada através do resultado do resumo criptográfico *md5* sobre a palavra formada pela concatenação do endereço de e-mail do requerente com um número aleatório.

5.3.5 Geração do Par de Chaves

A geração do par de chaves ocorre no lado do usuário e efetuada no navegador web ou em servidor de serviços criptográfico, sendo que há particularidades na geração depedendo do modelo do navegador, como descrito a seguir.

5.3.5.1 Mozilla Firefox e Similares

O navegador Mozilla Firefox [29] e similares - Opera [30], por exemplo - geram e gerenciam autonomamente - sem auxilio de outro software ou do sistema operacional - as chaves RSA. Uma vantagem deste fato é que não há a necessidade de se instalar softwares adicionais para a manipulação das chaves, o que torna o ambiente mais seguro uma vez que a API criptográfica é disponibilizada nativamente no browser.

A geração de chaves pode ser feita através de scripts em JavaScript - tecnologia citada na Seção 4.7 - ou, de maneira mais simples, por meio do uso da tag HTML - Seção 4.8 - <*keygen*>. Esta reenderiza na página web um formulário para a escolha do tamanho da chave RSA - 1024 ou 2048 bits - e um botão de confirmação, que quando pressionado inicia geração de um par de chaves, como ilustram nos Anexos 8.21 e 8.22.

As chaves geradas são armazenadas em disco e podem ser cifradas simétricamente, caso deseje-se aumentar o grau de segurança, através do dispositivo de chaveiro do navegador.

5.3.5.2 Internet Explorer

O navegador Internet Explorer [31] utiliza os servidores criptográficos disponíveis no Microsoft Windows [32] para gerar chaves RSA, que são gerenciadas, depois de criadas, por este sistema operacional. Todavia, para fazer uso das funcionalidade de criptografia no Internet Explorer é necessário a instalação do *Active-x Control* - um aplicativo de terceiros que proporciona interação entre a página web e o usuário - *xenroll.dll*, disponibilizado no módulo público. Vale citar que a necessidade de instalação de softwares *Active-x Control* para a geração de chaves abre uma brecha de segurança, uma vez que códigos maliciosos podem roubar a chave privada após a mesma ter sido criada [33].

Em adição ao *xenroll.dll* são necessários scripts em JavaScript que fazem uso das funções do servidor criptográfico para gerar o par de chaves RSA e exportar a pública no formato de requisição de certificado: PKCS#10. Também é função desses scripts a reenderização das opções de servidor criptográfico, como ilustrado no Anexo 8.22.

5.3.6 Implentação

A implementação do módulo público faz uso de marcações HTML para o desenvolvimento do *layout* das páginas ilustradas na Figura 5.8. Já naquelas em que é necessário dinâmismo, como reenderização de objetos particulares a cada modelo de navegador web, utilizou-se código em PHP, ferramenta mencionada na Seção 4.3.

A classe utilitária Persistence (Anexo 8.23) escrita em PHP define uma interface de acesso ao banco de dados da AC Correio, disponibilizando uma gama de funcionalidades como submissão de requisição e recuperação de certificados e LCR. Esta classe utiliza diretamente as facilidades da classe *Zend_Db* da biblioteca Zend para conectar-se ao banco, bem como executar as consultas.

A classe utilitária DomainsChecker - Anexo 8.24 -, também implementada em PHP, faz validações de endereço de e-mail. Para obter a lista de todos os domínios de e-mail permitidos, o classe faz uso classe *SimpleXMLElement* da bibli-

oteca Zend, que lê as entradas em XML no arquivo *publicModule.xml* - Anexo 8.25. Logo o domínio do endereço a ser validado é comparado com os obtidos de *publicModule.xml* através de uma expressão regular.

A última das classes utilitárias, ChallengeNotifier - Anexo 8.27 -, implementada em PHP realiza o envio de e-mails. Ela faz uso de duas classes da biblioteca Zend: Zend_Config_Xml e Zend_Mail. A primeira faz leitura do arquivo de configuração do Notificador (Anexo 8.26) e a segunda faz disponibiliza uma interface para envio de e-mail utilizando protocolos de rede.

Das páginas web elaboradas é relevante citar códigos de challengeForm.php e xenrollUtil.php - Anexos 8.29 e 8.28 respectivamente. O primeiro, além de do layout da página, ilustrada nos Anexos 8.21 e 8.22, contém as instruções para geração de chaves no Mozilla Firefox - tag *keygen*. O segundo refere-se ao código necessário para geração de chaves no Internet Explorer, valendo destacar a função *XEnroll.CreatePKCS10*, que efetivamente cria as chaves.

5.4 Banco de Dados

O banco de dados da AC Correio armazena todos os dados referentes a certificados, à LCR e ao próprio funcionamento da AC, com exceção das configurações desta que são descritas nos arquivos XML citados posteriormente. Essas informações estão organizadas nas tabelas ilustradas na Figura 5.9 e brevemente descritas depois:

- CaSettings: faz controle dos últimos seriais para certificados e LCR emitidos, bem como contém o certificado da AC Correio em formato PEM;
- IssuedCrl: armazena apenas a LCR mais recente;
- IssuedCerts: armazena todos os certificados, exceto os expirados ou revogados, emitidos pela AC Correio;
- RevokedCerts: armazena informações dos certificados revogados pela AC Correio;

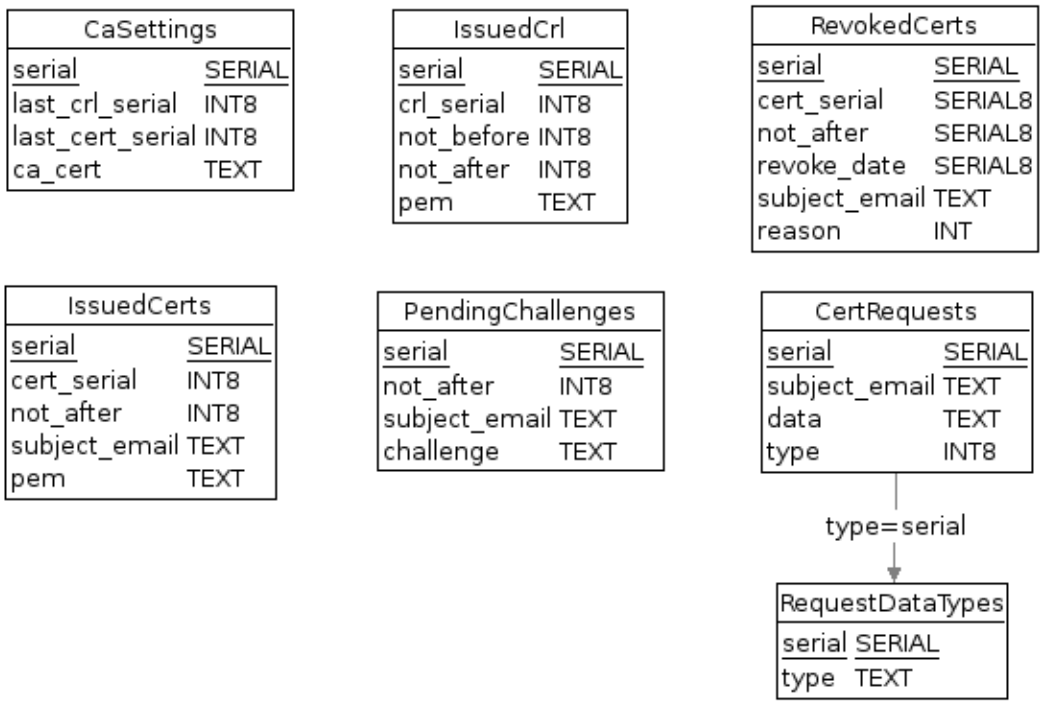

Figura 5.9: Modelagem do Banco de Dados da AC Correio.

- PendingChallenges: armazena as URL secretas geradas referentes desafios;
- CertRequests: armazena as requisições de certificadas submetidas ao final do protocolo de desafio-resposta;
- RequestDataTypes: armazena os tipos de requisição de certificado suportadas: *PKCS#10* e *SPKAC*.

As tabelas que armazenam informações que possuem prazos de validade possuem uma coluna com a data de expiração. Esta é lida e comparada com a data atual na rotina de limpeza de dados expirados (Seção 5.2.6): se a segunda possuir valor superior a da primeira, exclui-se a linha. Portanto, é interessante notar que a AC Correio não guarda histórico dos certificados emitidos, uma vez que são apagados quando expirados. A mesma política é adotada com a revogação daqueles.

Por último, verifica-se a unicidade da coluna que armazena o endereço de e-mail do requerente nas tabelas CertRequests, IssuedCerts, PendingChallenges. Isto garante que um endereço de correio eletrônico não possui mais de um certificado, por exemplo, como ainda proíbe que um usuário dê início a vários desafios ou submeta mais de uma requisição de certificado.

Capítulo 6

Políticas de Certificação

Como citado na Seção 3.5, é necessário que o grupo gestor de uma AC defina documentos de Políticas de Certificação e Declaração de Práticas de Certificação para as operações da mesma. Portanto, neste trabalho, redigiu-se apenas a DPC da AC Correio no escopo da ICPEDU.

A elaboração da DPC da AC Correio ocorreu em reuniões dos membros da Autoridade Gestora de Políticas (AGP) nas quais discutiram-se as seções propostas em [34]. Finalizada a DPC enviou-se ao Comite Gestor (GP) para consecutiva aprovação.

6.1 RFC3647

A RFC3647 [34] é um modelo elaborado para auxiliar a escrever Políticas de Certificação e Declaração de Práticas de Certificação para entidades de uma ICP, como por exemplo, ACs. Em particular, este documento apresenta uma lista compreensiva dos tópicos potencialmente candidatos a serem descritos nas PCs e DPCs, cuja breve descrição voltada a AC Correio da ICPEDU de alguns tópicos relevantes a este trabalho se dão nas seções seguintes.

6.1.1 Introdução

Inicialmente, nesta seção, informa-se a finalidade do documento, bem como a quais entidades de uma ICP ele se refere. Estabelecem-se também quais são os próposito de uso do certificado, como ainda informa-se o contato do grupo responsável pela elaboração do documento.

6.1.2 Responsabilidade referentes a publicações e repositórios

Nesta seção definem locais de livre acesso em que são públicados documentos pertinentes a AC Correio, como DPCs e PCs.

6.1.3 Identificação e Autenticação

Primeiramente, neste seção, definem-se os formatos dos nomes dos usuários de certificados. No caso da AC Correio, é neste ponto que define-se o campo de nomes do modelo de certificado citado na Seção 5.2.3. Em seguida descreve-se como se dá a validação da identidade do requerente de certificado. No que se refere a AC Correio, é neste ponto que informa-se o protocolo de desafio-resposta descrito na Seção 5.1.3.

6.1.4 Requisitos Operacionais do Ciclo de Vida do Certificado

Inicia-se com a descrição dos processos de solicitação, de processamento de requisição, emissão e de aceitação do certificado, bem como as responsabilidades do uso deste. No caso da AC Correio citam-se a validação de endereços de e-mail que podem solicitar certificados - Seção 5.3.3 -, o processamento e emissão de certificado pelo módulo criptográfico - Seção 5.2.4. A aceitação do certificado é definida da seguinte forma: cabe ao usuário, a qualquer momento, solicitar revogação do certicado caso não o aceite. Já no último item informa-se que o usuário deve proteger suas chaves, solicitar revogação de certificado caso haja compromentimento daquelas, como ainda respeitar os propósitos de uso daquele. Relevante a AC Correio, define-se ainda

a revogação de certificados. Esta pode ocorrer por iniciativa do grupo de operação da AC Correio ou por parte do titular do certificado.

6.1.5 Controles Técnicos de Segurança

Começa-se definindo a geração do par de chaves por parte do requerente de certificado, bem como o formato em que a chave pública é exporta, que se referem, respectivamente, ao descrito nas Seções 5.3.5 e 5.2.4. Por último, definem os artifícios de segurança sobre as chaves da AC. No caso da AC Correio, menciona-se o uso de um HSM da ICPEDU, como citado na Seção 5.2.2.

6.1.6 Perfis dos Certificados, LCR e OCSP

Inicialmente define-se a configuração padrão dos certificados emitidos. No caso da AC Correio é neste ponto em que informam-se as extensões presentes nos certificados, como descrito na Seção 5.2.3.

Logo define-se as configurações da LCR, como por exemplo a versão. No caso da AC Correio, utiliza-se a versão corrente: 2.

Por último, define a disponibilidade de OCSP - *Online Certificate Status Protocol*. No caso da AC Correio, informa-se que este serviço não é disponibilizado.

Capítulo 7

Considerações Finais

Inicialmente, reassalta-se que todos os objetivos definidos na Seção 1.2 foram alcançados plenamente. Desenvolveu-se uma AC que apresenta corretude em sua operação, uma vez que os requisitos estabelecidos na Seção 5.1.1 foram atendidos, além de que utilizaram-se com sucesso os certificados emitidos, fato verificado com através do emprego de clientes de e-mail. Em adição, vale mencionar a contextualização da AC Correio na ICPEDU, uma vez que o software desenvolvido apresenta integração com o HSM da ICPEDU.

Percebeu-se que o projeto de uma AC possui complexidade elevada, portanto foi dificil, e em alguns casos impossível, prever todos os requisitos da AC Correio na fase de análise tomando-se apenas como base o NewPKI [1]. Tal fato provocou mudanças de decisões ao longo do projeto o que, em alguns aspectos, vai de encontro com as recomendações de Engenharia de Software.

Embora o desenvolvimento do módulo criptográfico decorreu de maneira simples devido às facilidades proporcionadas pelas bibliotecas LibCryptoSec [12] e Qt4 [16], o mesmo não ocorreu com o módulo público. Neste enfrentou-se problemas no script de geração de chaves 5.3.5 devido a falta de um ambiente de desenvolvimento para JavaScript e a carência de documentação por parte da Microsoft a respeito dos servidores de serviços criptográficos desenvolvidos por ela.

Finalizando os conhecimentos adquiridos é relevante citar o estudo da

RFCs. Isto proporcionou uma familiarização com documentos técnicos bem como permitiu a elaboração da DPC da AC Correio. Essa proporcionou importantes conhecimentos sobre o processo de escrita de DPCs, que se dá em conjunto com outros indivíduos, como ainda de todos os procedimentos de operação e segurança envolvidos em uma ICP.

Como propostas de trabalhos futuros, citam-se as seguintes propostas de melhorias para AC Correio:

- implementar um sistema de registros de eventos da AC Correio em banco de dados;
- desenvolver uma interface de adminstração web em que o administrador possa configurar os módulos da AC Correio, como ainda analisar registros e solicitar revogação;
- incorporar à AC Correio uma AC interna que emita certificados aos módulos da AC Correio, incrementando a segurança do software como faz o NewPKI.

Referências

[1] GIUDICELLI, F. *NewPKI :: Where Open Source PKI Stands*. June 2007. Disponível em: <www.newpki.org>.

[2] ENCICLOPéDIA Mirador Internacional. [S.l.]: Encyclopaedia Britannica do Brasil Publicações Ltda., 1987. 2993-2996 p.

[3] KAHN, D. *The Codebreakers - The Story of Secret Writing*. [S.l.]: The American Library, Inc., 1967,1973.

[4] SCHNEIER, B. Applied cryptography: Protocols, algorithms, and source code in c. In: ____. [S.l.]: John Wiley and Sons, 1996. cap. 2.

[5] HOUSLEY, R.; POLK, T. *Planning for PKI - Best Practices Guide for Deploying Public Key Infrastructure*. [S.l.]: Wiley Computer Publishing, 2001.

[6] NATIONAL Institute of Standards and Technology. June 2007. Disponível em: <www.nist.gov>.

[7] DIFFIE, W.; HELLMANN, M. E. New directions on cryptographic techniques. *Proceedings of the AFIPS National Computer Conference*, 1976.

[8] MARTINA, J. E. *Projeto de um Provedor de Serviços Criptográficos Embarcado para Infra-estrutura de Chaves Públicas e suas Aplicações*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, 2005.

[9] HOUSLEY, R. et al. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. IETF, abr. 2002. RFC 3280 (Proposed Stan-

dard). (Request for Comments, 3280). Updated by RFCs 4325, 4630. Disponível em: <http://www.ietf.org/rfc/rfc3280.txt>.

[10] IGNACZAK, L. *Um novo modelo de Infra-estrutura de Chaves Públicas para uso no Brasil utilizando aplicativos com código fonte aberto*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, 2002.

[11] OPENSSL: The Open Source toolkit for SSL/TLS. June 2007. Disponível em: <www.openssl.org>.

[12] aO, L. de Segurança em I. *LibCryptoSec*. June 2007. Disponível em: <http://projetos.labsec.ufsc.br/libcryptosec>.

[13] GROUP, T. P. *PHP: Hypertext Preprocessor*. June 2007.

[14] ZEND Framework. June 2007. Disponível em: <http://framework.zend.com/>.

[15] STROUSTRUP, B. C++. In: SHAMIR, S. (Ed.). *The Handbook of Object Technology*. [S.l.]: CRC Press, 1998. cap. 15.

[16] QT - Trolltech. June 2007. Disponível em: <http://trolltech.com/products/qt>.

[17] JAVASCRIPT.COM (TM) - The Definitive JavaScript Resource: JavaScript Tutorials, Free Java Scripts, Source Code and Other Scripting Resources. June 2007.

[18] HTML 4.01 Specification. June 2007. Disponível em: <http://www.w3.org/TR/html4/>.

[19] THE Linux Home Page at Linux Online. June 2007. Disponível em: <www.linux.org>.

[20] IEE POSIX®- Certification Authority. June 2007. Disponível em: <http://standards.ieee.org/regauth/posix/index.html>.

[21] GCC, the GNU Compiler Collection. June 2007. Disponível em: <http://gcc.gnu.org/>.

[22] UML Modeling Tool - JUDE. June 2007. Disponível em: <http://jude.change-vision.com/jude-web/index.html>.

[23] EXTENSIBLE Markup Language (XML). June 2007. Disponível em: <http://www.w3.org/XML/>.

[24] POSTGRESQL: The world's most advanced open source database. June 2007. Disponível em: <http://www.postgresql.org/>.

[25] TANENBAUM, A. S. *Sistemas Operacionais Modernos*. 2. ed. [S.l.]: Prentice Hall, 2003.

[26] GT ICP-EDU - Trac. June 2007. Disponível em: <http://projetos.labsec.ufsc.br/icpedu>.

[27] NYSTROM, M.; KALISKI, B. *PKCS #10: Certification Request Syntax Specification Version 1.7*. IETF, nov. 2000. RFC 2986 (Informational). (Request for Comments, 2986). Disponível em: <http://www.ietf.org/rfc/rfc2986.txt>.

[28] JOSEFSSON, S. *The Base16, Base32, and Base64 Data Encodings*. IETF, jul. 2003. RFC 3548 (Informational). (Request for Comments, 3548). Obsoleted by RFC 4648. Disponível em: <http://www.ietf.org/rfc/rfc3548.txt>.

[29] MOZILLA.ORG - Home of the Mozilla Project. June 2007. Disponível em: <http://www.mozilla.org/>.

[30] OPERA Web browser: Homepage. June 2007. Disponível em: <http://www.opera.com/>.

[31] INTERNET Explorer: Home Page. June 2007. Disponível em: <http://www.microsoft.com/windows/products/winfamily/ie/default.mspx>.

[32] WINDOWS Home Page. June 2007. Disponível em: <http://www.microsoft.com/windows/default.mspx>.

[33] MICROSOFT private key recovery. June 2007. Disponível em: <http://insecure.org/sploits/microsoft.private-key.protections.html>.

[34] CHOKHANI, S. et al. *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework.* IETF, nov. 2003. RFC 3647 (Informational). (Request for Comments, 3647). Disponível em: <http://www.ietf.org/rfc/rfc3647.txt>.

Capítulo 8

Anexos

8.1 Arquivo de configuração do módulo criptográfico

```
<?xml version="1.0" encoding="UTF-8"?>
<caserver>

<persistence>
<host>localhost</host>
<port>5432</port>
<user>icpedupiloto</user>
<password>collinsgem</password>
<database>icpedupiloto</database>
</persistence>

<engine using="dynamic">
<none>
<privKeyFile>privkey.pem</privKeyFile>
<passPhrase>123456</passPhrase>
</none>
<static>
```

```
<keyId>key</keyId>
<engineId>whatever</engineId>
<engineCmd>COMANDO:VALOR</engineCmd>
</static>
<dynamic>
<keyId>chave</keyId>
<engineId>openhsmd</engineId>
<engineCmd>ADDRESS_CONN:150.162.66.91</engineCmd>
<engineCmd>PORT_CONN:5001</engineCmd>
<enginePath>/usr/lib/engines/engine_openhsmd.so</enginePath>
</dynamic>latex symbols
</engine>

<notify>
<smtp>blowfish.labsec.ufsc.br</smtp>

<signedCert>
<from>ac@labsec.ufsc.br</from>
<subject>Certificado assinado</subject>
<bodyText>Seu certificado est\’{a} disponivel em</bodyText>
<certUrl>http://ac-correio.labsec.ufsc.br/cert.php?id=</certUrl>
</signedCert>

<wrongCSR>
<from>ac@labsec.ufsc.br</from>
<subject>Requisi\c{c}\˜{a}o de certificado inv\’{a}lida</subject>
<bodyText>A requisi\c{c}\˜{a}o de certificado gerada pelo seu navegador er
</wrongCSR>
</notify>
```

```
<cycles>
<crlCycle unit="seconds">3</crlCycle>
<certCycle unit="seconds">10</certCycle>
<cleanCycle unit="seconds">30</cleanCycle>
</cycles>

<certificateTemplate>
<dn>
<country>BR</country>
<state>DF</state>
<locality>Brasilia</locality>
<organization>ICPEDU</organization>
<organization>RNP</organization>
</dn>
<extensions>
<crlDistributionPoints critical="false">
<distributionPoint>http://150.162.66.72/icpedu.rnp.br/repositorio/ac-corre
</crlDistributionPoints>

<certificatePolicies critical="false">

<!-- SEQUENCE (1..MAX)-->
<policyInformation>

<!--oid dpc-->
<policyIdentifier>1.3.6.1.4.1.15996.1.2.2.1</policyIdentifier>

<!--opcional-->
```

```
<!-- SEQUENCE (1..MAX)-->
<policyQualifierInfo>
<!-- url dpc -->
<cpsUri>http://150.162.66.72/icpedu.rnp.br/repositorio/ac-correio/dpc-ac-c
</policyQualifierInfo>

<policyQualifierInfo>
<!-- texto para ser mostrado qdo certificado for lido por uma aplicacao --
<!-- SEQUENCE (1..MAX)-->
<userNotice>

<!-- opcional -->
<!-- referencia um item dentro de um determinado arquivo de uma org -->
<!-- SEQUENCE (1..MAX)-->
<!--
<noticeReference>  par(org, number)
<organization>ICPEDU</organization>
<noticeNumbers>1,2,3,4,55</noticeNumbers>
</noticeReference>
-->
<explicitText>Os certificados da ICPEDU sao para uso exclusivo por institu
</userNotice>

</policyQualifierInfo>
</policyInformation>

</certificatePolicies>

<extendedKeyUsage critical="false">
```

```
<!-- SEQUENCE (1..MAX) de OIDs que representam um uso de chave-->
<!-- clientAuth -->
<oid>1.3.6.1.5.5.7.3.2</oid>
<oid>1.3.6.1.5.5.7.3.4</oid>
</extendedKeyUsage>

</extensions>
</certificateTemplate>

</caserver>
```

8.2 Diagrama de Classes de Análise d Módulo Criptográfico

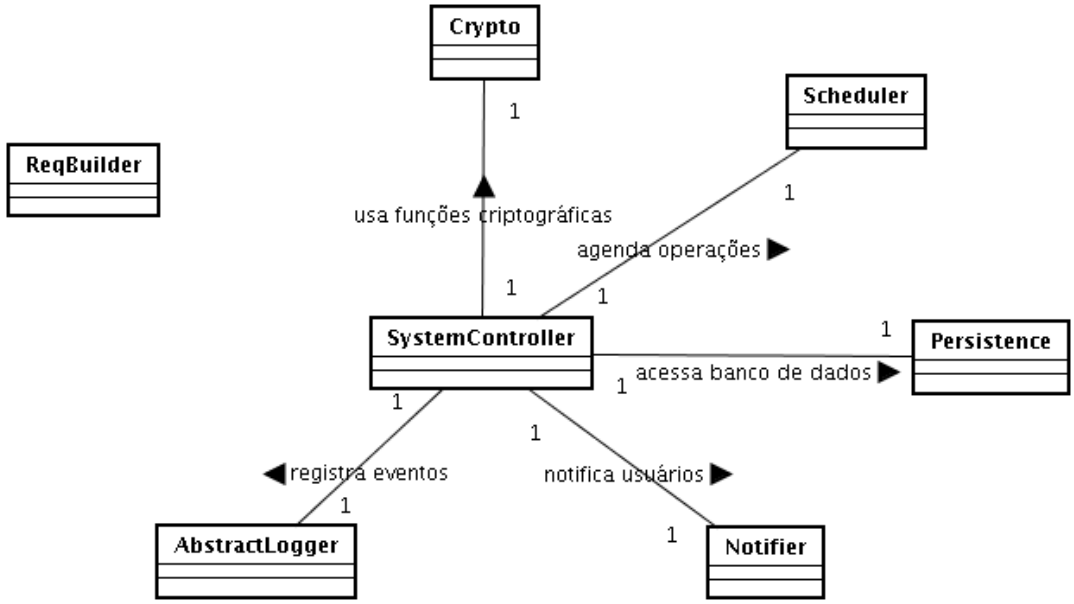

Figura 8.1: Diagrama de classes de análise do módulo criptográfico.

8.3 Classe SystemController

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmai
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef SYSTEMCONTROLLER_H_
```

```
#define SYSTEMCONTROLLER_H_

#include <sstream>
#include <vector>
#include <iostream>
#include <map>
#include <time.h>
#include <string.h>

#include <QtCore/QFile>
#include <certificate/Certificate.h>
#include "Crypto.h"
#include "Notifier.h"
#include "ConfigData.h"
#include "ConfigurationFileReader.h"
#include "Scheduler.h"
#include "ReqBuilder.h"
#include "Persistence.h"
#include "exception/FileSystemException.h"
#include "exception/SyntaxException.h"
#include "FileLogger.h"
#include "ConsoleLogger.h"

void readFile(string path, string& readData) throw(FileSystemException);

class SystemController
{

protected :
```

```
Crypto* crypto;
Persistence* persist;
AbstractLogger* logger;
Notifier* signedCertNotifier;
Notifier* wrongCSRNotifier;
ConfigData configData;
Scheduler* scheduler;

protected :
Certificate* getCACert() throw(EncodeException, PersistenceException);

public:
SystemController() throw();
virtual ~SystemController() throw();
void checkExpiredData() throw (PersistenceException);//nao utilizado ainda
void checkCertRequests() throw (LibCryptoSecException);
void checkRevokeRequests() throw (LibCryptoSecException);
void main()throw (LibCryptoSecException);


template<typename T>
static string numToString(T number)
{
stringstream s;
s << number;
return s.str();
}

template<typename T>
```

```
static T stringToNum(string s) throw (logic_error)
{
T result;
istringstream stream(s);

if (!(stream >> result))
throw ("error while converting a string to a number");

return result;
}
};

#endif /*SYSTEMCONTROLLER_H_*/
```

8.4 Classe Persistence

```
 /*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmai
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
```

```
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef PERSISTENCE_H_
#define PERSISTENCE_H_

#include "ConfigData.h"
#include "exception/PersistenceException.h"
#include <time.h>
#include <string>
#include <vector>
#include <iostream>
#include <QtCore/QString>
#include <QtSql/QSqlDatabase>
#include <QtSql/QSqlError>
#include <QtSql/QSqlQuery>
#include <QtCore/QVariant>
```

```
using namespace std;

class Persistence
{
private:
QSqlDatabase db;

public:

struct certReq
{
long long int serial;
string subjectEmail;
string data;
int type;
};

struct issuedCert
{
time_t notAfter;
long long int certSerial;
string subjectEmail;
string pem;
};

struct revokedCertData
{
long long int certSerial;
time_t notAfter;
```

```
time_t revokeDate;
string subjectEmail;
int reason;
};

struct issuedCrl
{
long long int crlSerial;
time_t notBefore;
time_t notAfter;
string pem;
};

Persistence(ConfigData& settings) throw(PersistenceException);
virtual ~Persistence();
time_t getNextCrl();
long long int getLastCertSerial() throw(PersistenceException);
void setLastCertSerial(long long int l) throw(PersistenceException);
long long int getLastCrlSerial() throw(PersistenceException);
void setLastCrlSerial(long long int l) throw(PersistenceException);
void getCertRequests(vector<struct certReq>& reqs) throw(PersistenceExcept
void storeIssuedCert(struct issuedCert& issuedCert) throw(PersistenceExcep
void deleteCertReq(long long int serial) throw(PersistenceException);
void updateRevokedCerts() throw(PersistenceException);
void deleteRevokedCerts() throw(PersistenceException);
void deleteRevokeReqs() throw(PersistenceException);
void getRevokedCertData(vector<struct revokedCertData>& revokedCertData) t
void deleteCrl() throw(PersistenceException);
void storeIssuedCrl(struct Persistence::issuedCrl& issuedCrls) throw(Persi
```

```
void cleanExpiredIssuedCerts() throw (PersistenceException);
void cleanExpiredRevokedCerts() throw (PersistenceException);
void cleanExpiredChallenges() throw (PersistenceException);
string getCaCertPem() throw(PersistenceException);
void storeCaCert(string pem) throw(PersistenceException);

template<typename T>
static string numToString(T number)
{
stringstream s;
s << number;
return s.str();
}

template<typename T>
static T stringToNum(string s) throw (logic_error)
{
T result;
istringstream stream(s);

if (!(stream >> result))
throw ("error while converting a string to a number");

return result;
}


};
```

```
#endif /*PERSISTENCE_H_*/
```

8.5 Classe Crypto

```
#ifndef CRYPTO_H_
#define CRYPTO_H_

#include <QtCore/QFile>
#include <certificate/Certificate.h>
#include <certificate/CertificateBuilder.h>
#include <certificate/RevokedCertificate.h>
#include <certificate/CertificateRevocationListBuilder.h>
#include <certificate/CertificateRevocationList.h>
#include <certificate/RDNSequence.h>
#include <certificate/SubjectAlternativeNameExtension.h>
#include <certificate/GeneralNames.h>
#include <certificate/GeneralName.h>
#include <certificate/SubjectKeyIdentifierExtension.h>
#include <certificate/AuthorityKeyIdentifierExtension.h>
#include <KeyPair.h>
#include <Engines.h>
#include <DynamicEngine.h>
#include <NetscapeSPKI.h>
#include <SymmetricCipher.h>
#include <exception/EncodeException.h>
#include "ConfigData.h"
#include "Persistence.h"
#include "exception/SyntaxException.h"
```

```
#include "exception/FileSystemException.h"
#include <iostream>
#include <time.h>


class Crypto
{
public:
Crypto(Certificate* cert, ConfigData& settings) throw(LibCryptoSecExceptio
virtual ~Crypto();
void issueCertificate(long long int serial, Persistence::certReq& reqs, Pe
void issueCrl(long long int serial, vector<Persistence::revokedCertData>&

protected:
KeyPair* keyPair;
Certificate* caCert;
time_t crlCycle;
string orgUnit;
string organization;
string locality;
string state;
string country;
RDNSequence subjectsDN;
vector< Extension* > certExtensions;

protected:
void readFile(string path, string& readData) throw(FileSystemException);
void parseLine(const string& line, const string& delim, string& arg, strin
CertificateRequest* buildCertificateRequest(string spkacKey) throw(SyntaxE
```

```
void setupCertificateRequest(CertificateRequest* req, string subjectEmail)
void setupCertificateExtensions(ConfigData& settings);
SubjectAlternativeNameExtension getSubjectAltNameExt(string subjectEmail);
};

#endif /*CRYPTO_H_*/
```

8.6 Classe ConfigData

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmail
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
```

```
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef CONFIGDATA_H_
#define CONFIGDATA_H_

#include <string>
#include <sstream>
#include <vector>
#include <time.h>
#include "exception/SyntaxException.h"
#include "certificate/PolicyInformation.h"
#include <certificate/CRLDistributionPointsExtension.h>
#include <certificate/CertificatePoliciesExtension.h>
#include <certificate/ExtendedKeyUsageExtension.h>
#include <certificate/RDNSequence.h>

using namespace std;

class ConfigData
{
public:
ConfigData();
virtual ~ConfigData();
```

```
string getPersistHost();
void setPersistHost(string b);

int getPersistPort();
void setPersistPort(string b);

string getPersistUser();
void setPersistUser(string b);

string getPersistPasswd();
void setPersistPasswd(string b);

string getPersistDatabase();
void setPersistDatabase(string b);

string getEngEngineType();
void setEngEngineType(string b);

string getEngPrivKeyFile();
void setEngPrivKeyFile(string b);

string getEngPassPhrase();
void setEngPassPhrase(string b);

string getEngKeyId();
void setEngKeyId(string b);

string getEngEngineId();
void setEngEngineId(string b);
```

```
vector<string> getEngEngineCmd();
void setEngEngineCmd(string b);

string getEngEnginePath();
void setEngEnginePath(string b);

string getNotifySignedCertFrom();
void setNotifySignedCertFrom(string b);

string getNotifySignedCertSubject();
void setNotifySignedCertSubject(string b);

string getNotifySignedCertBodyText();
void setNotifySignedCertBodyText(string b);

string getNotifySignedCertCertUrl();
void setNotifySignedCertCertUrl(string b);

string getNotifyWrongCSRFrom();
void setNotifyWrongCSRFrom(string b);

string getNotifyWrongCSRSubject();
void setNotifyWrongCSRSubject(string b);

string getNotifyWrongCSRBodyText();
void setNotifyWrongCSRBodyText(string b);

string getNotifySmtp();
```

```
void setNotifySmtp(string b);

time_t getIssuingCrlCycle();
void setIssuingCrlCycle(string b);

time_t getIssuingCertCycle();
void setIssuingCertCycle(string b);

time_t getCleanCycle();
void setCleanCycle(string b);

string getIssuingUnitCrlCycle();
void setIssuingUnitCrlCycle(string b);

string getIssuingUnitCertCycle();
void setIssuingUnitCertCycle(string b);

string getCleanCycleUnit();
void setCleanCycleUnit(string b);

vector<PolicyInformation> getPolicyInfo();
void setPolicyInfo(PolicyInformation b);

bool getPolicyInfoCritical();
void setPolicyInfoCritical(bool b);

CRLDistributionPointsExtension* getCRLDistributionPointsExt();
void setCRLDistributionPointsExt(CRLDistributionPointsExtension* b);
```

```
CertificatePoliciesExtension* getCertificatePoliciesExtension();
void setCertificatePoliciesExtension(CertificatePoliciesExtension* b);

ExtendedKeyUsageExtension* getExtendedKeyUsageExt();
void setExtendedKeyUsageExt(ExtendedKeyUsageExtension* b);

RDNSequence getDN();
void setDN(RDNSequence b);


private:
string persistHost;
string persistPort;
string persistUser;
string persistPasswd;
string persistDatabase;
//string persistQtPlugin;

string engPrivKeyFile;
string engPassPhrase;
string engEngineType;
string engKeyId;
string engEngineId;
vector<string> engEngineCmd;
string engEnginePath;

string notifySignedCertFrom;
string notifySignedCertSubject;
string notifySignedCertBodyText;
```

```
string notifySignedCertCertUrl;
string notifyWrongCSRFrom;
string notifyWrongCSRSubject;
string notifyWrongCSRBodyText;
string notifySmtp;

string issuingUnitCrlCycle;
string issuingUnitCertCycle;
string cleanCycleUnit;
string issuingCrlCycle;
string issuingCertCycle;
string cleanCycle;

string caLogFile;

vector<PolicyInformation> policyInfo;
bool policyInfoCritical;
CRLDistributionPointsExtension* cRLDistributionPointsExt;
CertificatePoliciesExtension* certificatePoliciesExt;
ExtendedKeyUsageExtension* extendedKeyUsageExt;
RDNSequence rdn;

public:
template<typename T>
static string numToString(T number)
{
stringstream s;
s << number;
return s.str();
```

```
}

template<typename T>
static T stringToNum(string s) throw (logic_error)
{
T result;
istringstream stream(s);

if (!(stream >> result))
throw ("error while converting a string to a number");

return result;
}

};

#endif /*CONFIGDATA_H_*/
```

8.7 Classe ConfigurationFileReader

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmai
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
```

```
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef CONFIGURATIONFILEREADER_H_
#define CONFIGURATIONFILEREADER_H_

#include <QtCore/QString>
#include <QtCore/QFile>
#include <QtXml/QDomDocument>
#include <QtXml/QDomNodeList>
#include <string>
#include "ConfigData.h"
#include "Utils.h"
#include "exception/FileSystemException.h"
```

```
#include <certificate/CRLDistributionPointsExtension.h>
#include <certificate/DistributionPoint.h>
#include <certificate/DistributionPointName.h>
#include <certificate/GeneralNames.h>
#include <certificate/GeneralName.h>
#include <certificate/CertificatePoliciesExtension.h>
#include <certificate/ObjectIdentifierFactory.h>
#include <certificate/ObjectIdentifier.h>
#include <certificate/PolicyInformation.h>
#include <certificate/PolicyQualifierInfo.h>
#include <certificate/UserNotice.h>
#include <certificate/RDNSequence.h>

using namespace std;

class ConfigurationFileReader
{
public:
ConfigurationFileReader(ConfigData* c);
virtual ~ConfigurationFileReader();

ConfigurationFileReader(string path);

virtual void openConfigurationFile(string path) throw (FileSystemExceptio
void readConfiguration() throw(FileSystemException);

private:
QDomDocument caServerSettings;
ConfigData* configData;
```

```
void readPersistenceConfig(QDomNode node) throw(FileSystemException);
void readEngineConfig(QDomNode node) throw(FileSystemException);
void readNotifyConfig(QDomNode node) throw(FileSystemException);
// void readCaCertConfig(QDomNode node) throw(FileSystemException);
// void readCaLogConfig(QDomNode node) throw(FileSystemException);
void readCyclesConfig(QDomNode node) throw(FileSystemException);
void readCertTemplateConfig(QDomNode node) throw(FileSystemException);
void readCrlDistributionPointsConfig(QDomNode node) throw(FileSystemExcept
void readCertificatePoliciesConfig(QDomNode node) throw(FileSystemExceptio
void readPolicyInformationConfig(QDomNode node, PolicyInformation& policyI
void readPolicyQualifierInfoConfig(QDomNode node, PolicyInformation& polic
void readUserNotifyConfig(QDomNode node, PolicyQualifierInfo& qualifier) t
void readExtendedKeyUsageConfig(QDomNode node) throw(FileSystemException,
void readDnConfig(QDomNode node) throw(FileSystemException);
void readExtensionsConfig(QDomNode node) throw(FileSystemException);
};

#endif /*CONFIGURATIONFILEREADER_H_*/
```

8.8 Classe AbstractLogger

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmail
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
```

```
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef ABSTRACTLOGGER_H_
#define ABSTRACTLOGGER_H_

#include <string>
#include <stdexcept>
#include <iostream>

using namespace std;

class AbstractLogger
```

```
{
public:
AbstractLogger();
virtual ~AbstractLogger();
virtual void addEntry(string str) = 0;
};

#endif /*ABSTRACTLOGGER_H_*/
```

8.9 Classe ConsoleLogger

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmai
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
```

```
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef CONSOLELOGGER_H_
#define CONSOLELOGGER_H_

#include "AbstractLogger.h"

class ConsoleLogger : public AbstractLogger
{
public:
ConsoleLogger();
virtual ~ConsoleLogger();
void addEntry(string str);
};

#endif /*CONSOLELOGGER_H_*/
```

8.10 Classe FileLogger

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmail
 *
```

```
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef FILELOGGER_H_
#define FILELOGGER_H_

#include <string>
#include <QtCore/QString>
```

```
#include <QtCore/QFile>
#include <QtCore/QIODevice>
#include "AbstractLogger.h"
#include "exception/FileSystemException.h"


class FileLogger : public AbstractLogger
{
public:
FileLogger(std::string path);
virtual ~FileLogger();
void addEntry(string str) throw (FileSystemException);

private:
QFile* log;

};

#endif /*FILELOGGER_H_*/
```

8.11 Classe Notifier

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmai
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
```

```
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef NOTIFIER_H_
#define NOTIFIER_H_

#include <string>
#include "jwSMTP/jwsmtp.h"

using namespace std;
using namespace jwsmtp;
```

```
class Notifier
{
public:
Notifier(string fromAddr, string subject, string bodyText, string url, str
Notifier(string fromAddr, string subject, string bodyText, string stmp);
virtual ~Notifier();
static void sendEmail(string emailAddr, unsigned long long int id);
void send(string rcpt, string id);
void send();

//getters setters
string getFromAddr();
void setFromAddr(string b);
string getSubject();
void setSubject(string b);
string getBodyText();
void setBodyText(string b);
string getSmtp();
void setSmtp(string b);
string getRcpt();
void setRcpt(string b);

private:
string fromAddr;
string subject;
string bodyText;
string smtp;
string rcpt;
};
```

```
#endif /*NOTIFIER_H_*/
```

8.12 Classe ReqBuilder

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmai
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
```

```
*/

#ifndef REQBUILDER_H_
#define REQBUILDER_H_

#include "ConfigData.h"
#include <string>
#include <stdexcept>
#include <QtCore/QString>
#include <QtCore/QFile>
#include <QtCore/QIODevice>
#include "exception/SyntaxException.h"
#include "exception/FileSystemException.h"

class ReqBuilder
{
public:
ReqBuilder(ConfigData* settings);
virtual ~ReqBuilder();
void buildReq();

private:
ConfigData* settings;
void parseLine(const string& line, const string& delim, string& arg, strin
void writeToFile(string& str) throw (FileSystemException);
void readTemplateFile(string& data, string path) throw (FileSystemExceptio
};

#endif /*REQBUILDER_H_*/
```

8.13 Classe Scheduler

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmai
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef SCHEDULER_H_
```

```
#define SCHEDULER_H_

#include <time.h>
#include <iostream>
#include "ConfigData.h"

class Scheduler
{
public:
Scheduler(ConfigData& settings);
virtual ~Scheduler();

time_t getSleepTime();
bool getIssueCrlStatus();
bool getCertReqStatus();
bool getCleanStatus();
void setNextCrlIssue();
time_t getNextCrlIssue();
void setNextCertReqCheck();
time_t getNextCertReqCheck();
void setNextClean();
time_t getNextClean();

protected:
time_t crlCycle;
time_t certCycle;
time_t cleanCycle;
time_t nextCrlIssue;
time_t nextCertReqCheck;
```

```
time_t nextClean;
};

#endif /*SCHEDULER_H_*/
```

8.14 Classe Utils

```
/*
 * Copyright (C) 2006  Mart\'{i}n Augusto Gagliotti Vigil (gagliotti@gmai
 *
 *  This file is part of AC de Correio Eletr\^{o}nico.
 *
 *  AC de Correio Eletr\^{o}nico is free software; you can redistribute it
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  Foobar is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with Foobar; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
 *
 *  This software has educational purposes and its development has been sp
 *  by Rede Nacional de Pesquisa (RNP) which owns its rights. It is one is
 *  ICPEDU Piloto project whose main goal is to create a educational PKI i
```

```
 *
 *  AC de Correio Eletr\^{o}nico project: http://projetos.labsec.ufsc.br/a
 *  ICPEDU Piloto project: http://www.icpedu.rnp.br/
*/

#ifndef UTILS_H_
#define UTILS_H_

#include <vector>
#include <stdexcept>
#include <sstream>
#include <QtCore/QString>
#include <QtCore/QStringList>



using namespace std;

class Utils
{
public:
Utils();
virtual ~Utils();

template<typename T>
static string numToString(T number)
{
stringstream s;
s << number;
```

```
return s.str();
}

template<typename T>
static T stringToNum(string s) throw (logic_error)
{
T result;
istringstream stream(s);

if (!(stream >> result))
throw ("error while converting a string to a number");

return result;
}

template<typename T>
static vector<T> stringToVector(QString s) throw (logic_error)
{
vector<T> result;

QStringList tokens = s.split(",", QString::SkipEmptyParts);

for(int i = 0 ; i < tokens.size() ; i++)
{
result.push_back(stringToNum<T>(tokens.at(i).toStdString()));
}

return result;
//
```

```
}
};

#endif /*UTILS_H_*/
```

8.15 Classe LibCryptoSecException

```
#ifndef LIBCRYPTOSECEXCEPTION_H_
#define LIBCRYPTOSECEXCEPTION_H_

#include <exception>

#include "OpenSSLErrorHandler.h"

class LibCryptoSecException : public std::exception
{
public:
virtual ~LibCryptoSecException() throw () {}
virtual std::string getMessage() const = 0;
    virtual std::string toString() const = 0;
    virtual const char *what() const throw ()
    {
     return ((this->getMessage()).c_str());
    }
    virtual const std::string getDetails() const throw ()
    {
     return this->details;
    }
protected:
```

```
    std::string where;
    std::string details;
};

#endif /*LIBCRYPTOSECEXCEPTION_H_*/
```

8.16 Classe FileSystemException

```
#ifndef FILESYSTEMEXCEPTION_H_
#define FILESYSTEMEXCEPTION_H_

#include "exception/LibCryptoSecException.h"

using namespace std;

class FileSystemException : public LibCryptoSecException
{

public:

enum ErrorCode
{
UNKNOWN,
FILENOTFOUND,
CORRUPTEDFILE,
OPEN,
READ,
WRITE
};
```

```
    FileSystemException(string where) throw()
    {
     this->error = UNKNOWN;
     this->where = where;
    }

    FileSystemException(ErrorCode error, string where) throw()
    {
     this->error = error;
     this->where = where;
    }

    FileSystemException(string details, ErrorCode error, string where) thr
    {
     this->details = details;
     this->error = error;
     this->where = where;
    }

virtual ~FileSystemException() throw () {}

virtual std::string getMessage() const
{
return (FileSystemException::errorCode2Message(this->error));
}

    virtual string toString() const
    {
```

```
 std::string ret;
 if (this->error == FileSystemException::UNKNOWN)
 {
 ret = "FileSystemException. Called by: " + this->where + ".";
 }
 else
 {
 ret = "FileSystemException: " + FileSystemException::errorCode2Messag
 }
 return ret;
}

virtual FileSystemException::ErrorCode getErrorCode()
{
 return this->error;
}

static std::string errorCode2Message(FileSystemException::ErrorCode er
{
 std::string ret;
 switch (error)
 {
 case FileSystemException::UNKNOWN:
 ret = "Unknown error";
 break;
 case FileSystemException::FILENOTFOUND:
 ret = "File not found";
 break;
 case FileSystemException::CORRUPTEDFILE:
```

```
     ret = "Corrupted file";
     break;
     case FileSystemException::OPEN:
     ret = "Error while opening file";
     break;
     case FileSystemException::READ:
     ret = "Error while reading file";
     break;
     case FileSystemException::WRITE:
     ret = "Error while writing on file";
     break;
     }
     return ret;
    }


protected:
ErrorCode error;
};

#endif /*FILESYSTEMEXCEPTION_H_*/
```

8.17 Classe PersistenceException

```
#ifndef PERSISTENCEEXCEPTION_H_
#define PERSISTENCEEXCEPTION_H_

#include "exception/LibCryptoSecException.h"
```

```
using namespace std;

class PersistenceException : public LibCryptoSecException
{

public:

enum ErrorCode
{
UNKNOWN = 0,
SQLDRIVER = 1,
CONNECTION = 2,
STATEMENT = 3,
TRANSACTION = 4,
};

    PersistenceException(string details, PersistenceException::ErrorCode e
    {
     this->details = details;
     this->error = error;
     this->where = where;
    }


    PersistenceException(PersistenceException::ErrorCode error, string whe
    {
     this->error = error;
     this->where = where;
    }
```

```
    PersistenceException(string where) throw()
    {
     this->error = UNKNOWN;
     this->where = where;
    }

virtual ~PersistenceException() throw () {}

virtual std::string getMessage() const
{
return (PersistenceException::errorCode2Message(this->error));
}

    virtual string toString() const
    {
     std::string ret;
     if (this->error == PersistenceException::UNKNOWN)
     {
     ret = "PersistenceException. Called by: " + this->where + ".";
     }
     else
     {
     ret = "PersistenceException: " + PersistenceException::errorCode2Mess
     }
     return ret;
    }

    virtual PersistenceException::ErrorCode getErrorCode()
```

```
    {
     return this->error;
    }

    static std::string errorCode2Message(PersistenceException::ErrorCode e
    {
     std::string ret;
     switch (error)
     {
     case PersistenceException::UNKNOWN:
     ret = "Unknown error";
     break;
     case PersistenceException::SQLDRIVER:
     ret = "SQL driver";
     break;
     case PersistenceException::CONNECTION:
     ret = "Database connection";
     break;
     case PersistenceException::STATEMENT:
     ret = "Sql statement";
     break;
     case PersistenceException::TRANSACTION:
     ret = "Transaction";
     break;
     }
     return ret;
    }

protected:
```

```
PersistenceException::ErrorCode error;
};


#endif /*PERSISTENCEEXCEPTION_H_*/
```

8.18 Classe SyntaxException

```
#ifndef SYNTAXEXCEPTION_
#define SYNTAXEXCEPTION_

#include "exception/LibCryptoSecException.h"

using namespace std;

class SyntaxException : public LibCryptoSecException
{

public:

    SyntaxException(string error, string where) throw()
    {
     this->error = error;
     this->where = where;
    }

virtual ~SyntaxException() throw () {}

virtual std::string getMessage() const
```

```
{
return string("Syntax exception");
}

    virtual std::string toString() const
    {
     return string("SyntaxException. Called by: " + this->where + ". Unkno
    }

protected:
string error;
};

#endif /*SYNTAXEXCEPTION_*/
```

8.19 Página Inicial do Módulo Público

8.20 Página de Submissão de Endereço de E-mail

8.21 Página de Geração de Chaves no Mozilla Firefox

8.22 Página de Geração de Chaves no Internet Explorer

8.23 Classe Persistence

```
 <?php

require_once('autoLoad.php');
```

Figura 8.2: Página inicial do módulo público.

```
class Persistence
{
 protected $db;
 protected $challenges_table = '"PendingChallenges"';
 protected $requestDataType_table = '"RequestDataTypes"';
 protected $certRequests_table = '"CertRequests"';
 protected $certificates_table = '"IssuedCerts"';
 protected $crl_table = '"IssuedCrl"';
 protected $revoked_table = '"RevokedCerts"';
```

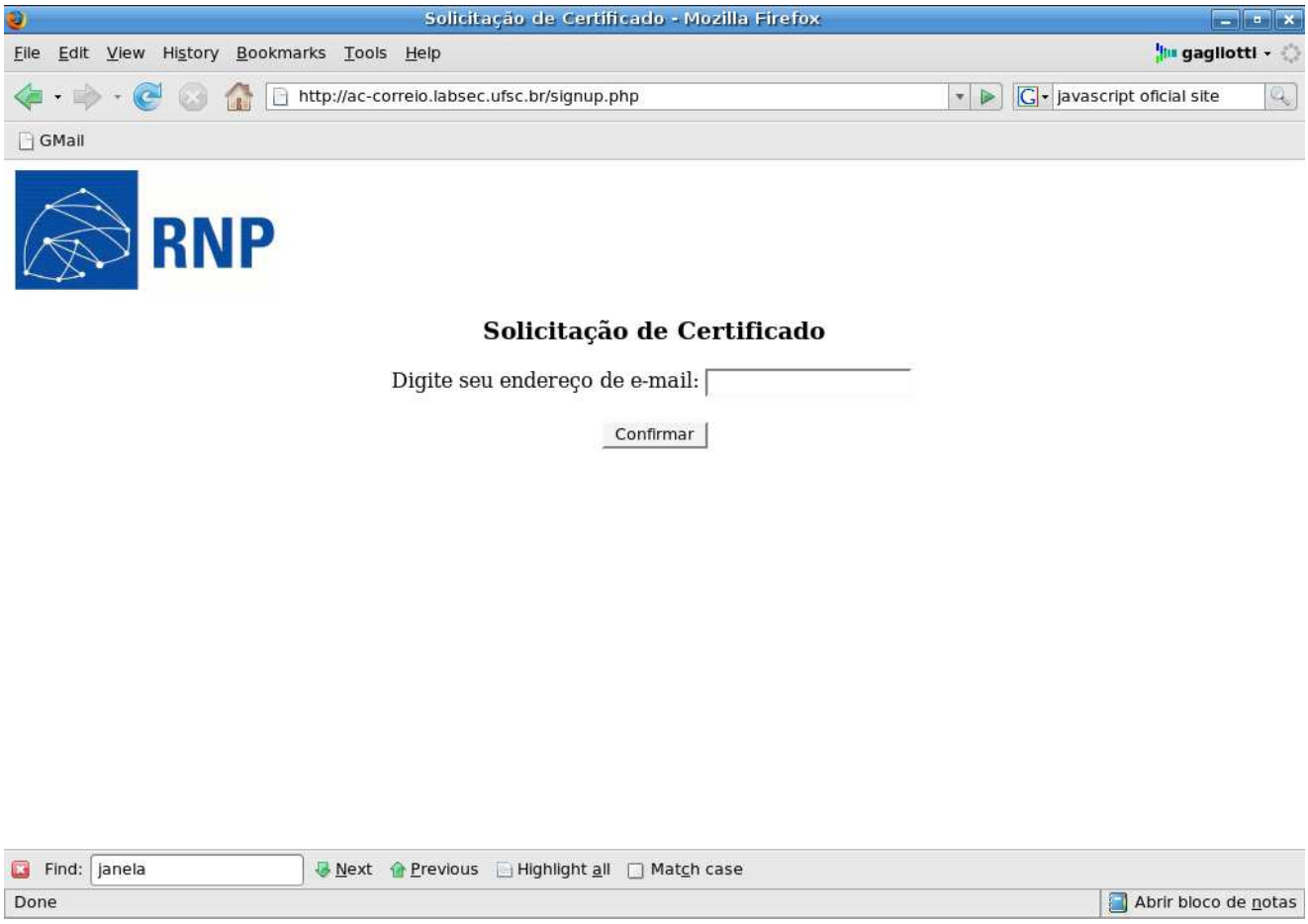

Figura 8.3: Página de submissão de endereço de e-mail.

```
  protected $caSettings_table = '"CaSettings"';
  protected $users_table = '"Users"';

  public function Persistence($backend, $host, $port, $dbname, $login, $pa
  {
  // if (!Zend_Filter::isDigits($backend))
  // throw new Exception("wrong datatype found");

//  if (Zend_Filter::isDigits($host))
  // throw new Exception("wrong datatype found");
```

Figura 8.4: Página de geração de chave no Mozilla Firefox.

```
if (!Zend_Filter::isDigits($port))
throw new Exception("wrong datatype found");

$params = array (’host’ => $host,
                 ’username’ => $login,
                 ’password’ => $passwd,
                 ’dbname’   => $dbname);

      $this->db = Zend_Db::factory($backend, $params);
```

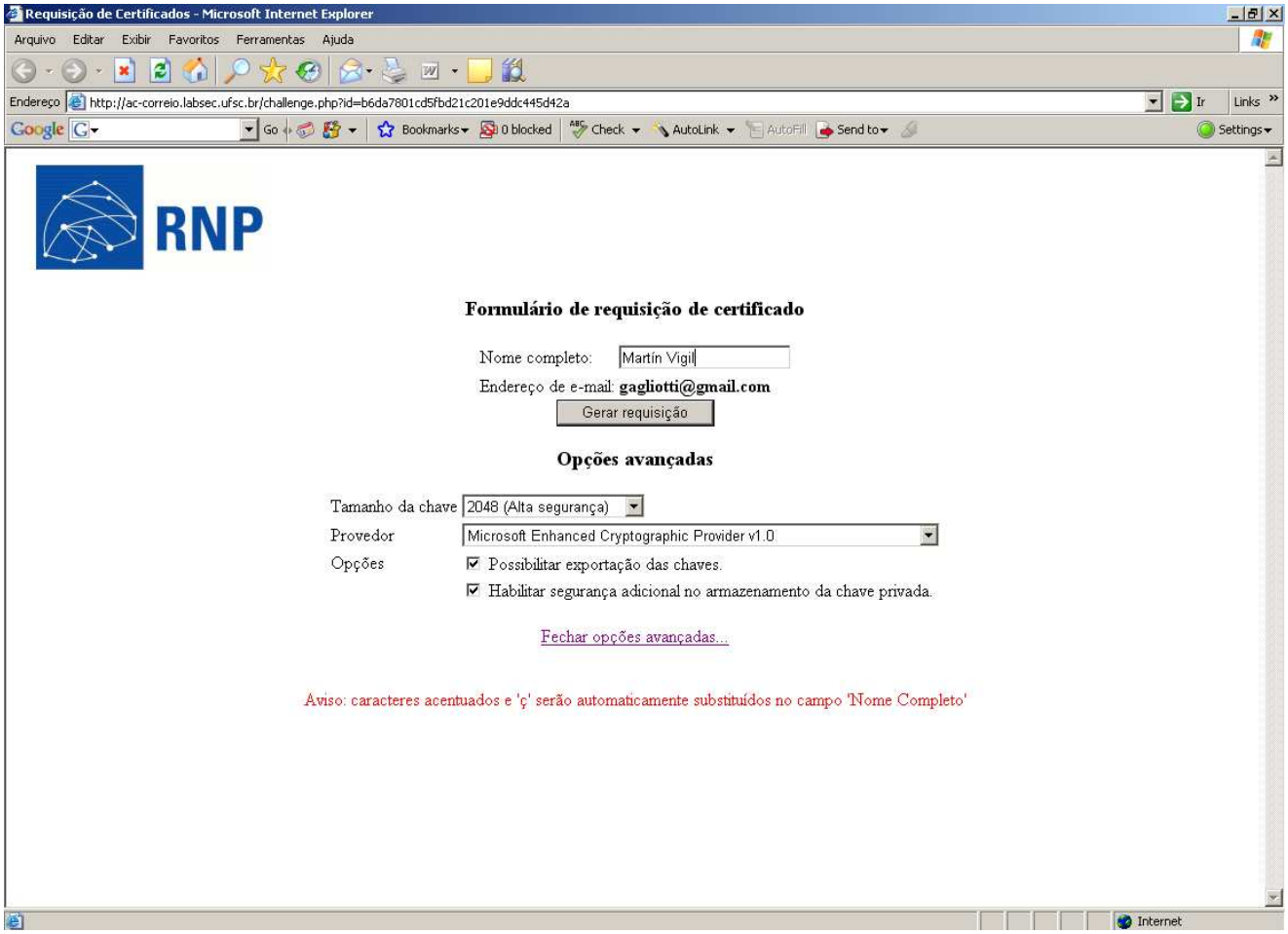

Figura 8.5: Página de geração de chave no Internet Explorer.

```
  }

public function addRequest($email)
{

$query = 'select * from '.$this->challenges_table.' where subject_email =
$result = $this->db->fetchAll($query);

if ( count($result) > 0)
throw new Exception("Requisi\c{c}\~{a}o j\'{a} efetuada.");
```

```
$challenge = md5($email.time());
$not_after = time() + (24 * 60 * 60); //24hs pegar do banco?

$row = array('not_after' => $not_after, 'subject_email' => $email, 'challe
$rowsAffected = $this->db->insert($this->challenges_table, $row);

//echo $rowsAffected;
}

public function isValidChallenge($challenge)
{
$query = 'select * from ' . $this->challenges_table . ' where challenge =
$result = $this->db->fetchAll($query);
$rc = false;

if( count($result) > 0)
$rc = true;

return $rc;
}

public function getRequestDataTypes()
{
$query = 'select * from '.$this->requestDataType_table;
$result = $this->db->fetchAll($query);

if(count($result) < 1)
throw new Exception("no request data types found");
```

```
return $result;
}

public function addPkcs10CSR($subjectEmail, $pcks10){
$query = 'select serial from ' . $this->requestDataType_table . ' where ty
$result = $this->db->fetchAll($query);

// print_r($result);

$query = 'insert into ' . $this->certRequests_table . '(subject_email, dat
values (' . $this->db->quote($subjectEmail) . ',' . $this->db->quote($pcks
',' . $result[0]["serial"] . ')';
$result = $this->db->fetchAll($query);

   //print_r($result);
}

public function addSpkac($subjectEmail, $spkac){
$query = 'select serial from ' . $this->requestDataType_table . ' where ty
$result = $this->db->fetchAll($query);

// print_r($result);

$query = 'insert into ' . $this->certRequests_table . '(subject_email, dat
values (' . $this->db->quote($subjectEmail) . ',' . $this->db->quote($spka
',' . $result[0]["serial"] . ')';
$result = $this->db->fetchAll($query);
```

```
   //print_r($result);
}

public function getEmailOfChallenge($challenge){
$query = 'select subject_email from ' . $this->challenges_table . ' where
$result =  $this->db->fetchAll($query);
return $result[0]['subject_email'];
}

public function getChallengeofEmail($email){
$query = 'select challenge from ' . $this->challenges_table . ' where subj
$result =  $this->db->fetchAll($query);
return $result[0]['challenge'];
}

public function getCertificate($id){
$query = 'select pem from ' . $this->certificates_table . ' where cert_ser
$result =  $this->db->fetchAll($query);

// print_r($result);

if(count($result) < 1)
throw new Exception("Certificate whose id = " . $id . ' does not exist');

return $result[0]['pem'];
}

public function getCRL(){
$query = 'select * from ' . $this->crl_table;
```

```
$result = $this->db->fetchAll($query);

if(count($result) < 1)
throw new Exception('WARNING: No CRL found');

if(count($result) > 1)
throw new Exception('Duplicated CRL found');

//print_r($result);

return $result[0]['pem'];
}

public function deleteChallenge($email){
$query = 'delete from ' . $this->challenges_table . 'where subject_email =
. $this->db->quote($email) . ';';

$this->db->fetchAll($query);
}

public function revokeCert($email){

$reason = 1; //key compromised

$query = 'select * from ' . $this->certificates_table . ' where subject_em
. $this->db->quote($email) . ';';

$result = $this->db->fetchAll($query);
```

```
if(count($result) > 0)
{
$query = ’insert into ’ . $this->revoked_table . ’(cert_serial, not_after,
subject_email, reason) values(’ .
$result[0]["cert_serial"] . ’,’ .
$result[0]["not_after"] . ’,’ .
time() . ’,’ .
$this->db->quote($result[0]["subject_email"]) . ’,’ .
$reason . ’);’;

$result = $this->db->fetchAll($query);

$query = ’delete from ’ . $this->certificates_table . ’ where subject_emai
. $this->db->quote($email) . ’;’;

$result = $this->db->fetchAll($query);
}
}

public function getCACert(){
$query = ’select ca_cert from ’ . $this->caSettings_table . ’;’;
$result = $this->db->fetchAll($query);

return $result[0][’ca_cert’];
}

public function getUserPasswdHash($username){
$query = ’select password from ’ . $this->users_table . ’ where username =
$result = $this->db->fetchAll($query);
```

```
return $result[0][’password’];
}

}

?>
```

8.24 Classe DomainsChecker

```
<?php

require_once(’autoLoad.php’);

 class DomainsChecker
 {
protected $domains;

public function DomainsChecker(){

$file = fopen(’../conf/publicModule.xml’, ’r’);
$xmlStr = fread($file, filesize(’../conf/publicModule.xml’));
$xml = new SimpleXMLElement($xmlStr);
$this->domains = $xml->allowedDomains->domain;
fclose($file);
}

public function checkDomain($email){
$valid = false;
$tokens = explode(’@’, $email);
```

```
$domain = $tokens[1];

if(count($tokens) != 2)
throw new Exception('Valor inv\'{a}lido de endere\c{c}o de e-mail: '. $ema

$i = 0;
while((!$valid) && ($i < count($this->domains))){

$regex = $this->domains[$i] . '$'; //expressao regular: *(dominio)
$valid = ereg($regex, $domain);
//$valid = (strcasecmp($this->domains[$i], $domain) == 0);
$i++;
}
return $valid;
}
 }
?>
```

8.25 Arquivo de Configuração do Módulo Público

```
<?xml version="1.0" encoding="UTF-8"?>
<template>
<persistence>
<host>localhost</host>
<port>5432</port>
<user>icpedupiloto</user>
<password>collinsgem</password>
<database>icpedupiloto</database>
<backend>PDO_PGSQL</backend>
```

```
</persistence>

<allowedDomains>
<domain>cefetsc.edu.br</domain>
<domain>ufsc.br</domain>
<domain>cefetse.edu.br</domain>
<domain>ufs.br</domain>
<domain>epm.br</domain>
<domain>lnls.br</domain>
<domain>cefetsp.br</domain>
<domain>ufscar.br</domain>
<domain>ufabc.edu.br</domain>
<domain>cenpra.gov.br</domain>
<domain>uft.edu.br</domain>
<domain>etfto.gov.br</domain>
<domain>unitins.br</domain>
</allowedDomains>
</template>
```

8.26 Arquivo de configuração do Notificador de Desafio

```
<?xml version="1.0" encoding="UTF-8"?>
<template>
<challengeEmail>
<from> ra@ac.labsec.ufsc.br </from>
<subject> Confirme solicita\c{c}\~{a}o de certificado </subject>
<bodyText> Esta mensagem foi gerada automaticamente pela AC de Correio Ele
<url> http://ac-correio.labsec.ufsc.br/challenge.php?id=</url>
```

```
<smtp> 150.162.66.6 </smtp>
</challengeEmail>
</template>
```

8.27 Classe ChallengeNotifier

```
<?php

require_once('autoLoad.php');

class ChallengeNotifier extends Zend_Mail
{
protected $message;
protected $msgConf;

function ChallengeNotifier($challenge, $recipient)
{
$msgConf = new Zend_Config_Xml('../conf/challengeEmail.xml', 'challengeEma

$preferences = array(
     "input-charset" => "UTF-8",
     "output-charset" => "ISO-8859-1",
     "line-length" => 76,
     "line-break-chars" => "\r\n",
     "scheme" => "Q"
);

$tr = new Zend_Mail_Transport_Smtp(trim($msgConf->smtp));
Zend_Mail::setDefaultTransport($tr);
```

```
$this->addHeader('MIME-Version', '1.0');
$this->addHeader('Content-Type', 'text/plain; charset="ISO-8859-1"', true)

$subjectHeaderString = iconv_mime_encode( '', $msgConf->subject, $preferen
$subjectHeaderStringArray = explode(":", $subjectHeaderString);
//echo($subjectHeaderStringArray[1]);
$this->setSubject($subjectHeaderStringArray[1]);
//$this->setSubject('=?ISO-8859-1?Q?Confirme=20solicita=E7=E3o=20de=20cert


//$this->addHeader('Content-Type', 'format=flowed', true);

$this->setFrom(trim($msgConf->from));
$this->addTo($recipient);


$message = trim($msgConf->bodyText);
$message = $message . $msgConf->url . $challenge;
$this->setBodyText($message, 'ISO-8859-1');

}
}
?>
```

8.28 Rotinas de Geração de Chaves do Microsoft Crypto API

```
<?php
```

```
//    header("Content-type: application/x-javascript; charset=UTF-8");

    $msie_XEnrollVersion = "5,131,3686,0";

    // Dinamic configuration
    $userAgent = $_SERVER['HTTP_USER_AGENT'];
    $isMSIE = (strpos($userAgent, 'MSIE') !== false);

?>
<script type="text/javascript">

var mp_msie_listProviders_tries;
var mp_msie_listProviders_delay;
var mp_msie_listProviders_timer;
var mp_msie_listProviders_select;
var mp_msie_listProviders_xenroll;

function mp_onLoad(form) {
<?php if ($isMSIE) { ?>
    document.getElementById('advancedOptionsKeygen').style.display='none';
document.getElementById('msie_xenroll').innerHTML =
"<OBJECT id=XEnroll classid=clsid:127698e4-e730-4e5c-a2b1-21490a70c8a1 cod
return mp_msie_listProviders(form.msie_provider, document.getElementById('
<?php } else { ?>
    document.getElementById('advancedOptionsMSIE').style.display='none';
    return true;
<?php } ?>

}
```

```
function mp_msie_listProviders(select, xenroll) {
    mp_msie_listProviders_tries = 10;
    mp_msie_listProviders_delay = 2000; // milliseconds
    mp_msie_listProviders_timer = null;
    mp_msie_listProviders_select = select;
    mp_msie_listProviders_xenroll = xenroll;
    mp_msie_listProviders_try();
}

function mp_msie_listProviders_try() {
    var result = false;
    try {
        // clear selection list
        for (i = 0; i < mp_msie_listProviders_select.length; i++)
            mp_msie_listProviders_select.options[i] = null;

        mp_msie_listProviders_xenroll.reset();

        var defaultProvider = "Microsoft Enhanced Cryptographic Provider";
        if (mp_msie_addProviders(mp_msie_listProviders_select, mp_msie_lis
            // select the default provider
            i = mp_msie_listProviders_select.options.length - 1;
            while ((i > 0) && (mp_msie_listProviders_select.options[i].tex
            mp_msie_listProviders_select.selectedIndex = i;
            //mp_msie_changeProvider(mp_msie_listProviders_select);

            result = true;
        } else
```

```
            alert("No providers found.");

    } catch (e) {
        mp_msie_listProviders_tries--;
        if (mp_msie_listProviders_tries > 0)
            mp_msie_listProviders_timer = setTimeout("mp_msie_listProvider
        else {
            if (mp_msie_listProviders_timer != null)
                clearTimeout(mp_msie_listProviders_timer);

            mp_msie_handleException(e);
        }
    }
    return result;
}

//
// Add providers from Enroll in selection list
// Return the number of providers added
//
function mp_msie_addProviders(select, xenroll) {
    var MAX_PROVIDER_TYPES = 25; // Maximum number of providers defined in

    var providerCount = 0;

    // Conditional providers (enabled based on browser and OS version)
    var providerDSS = "Microsoft Base DSS Cryptographic Provider";
    var providerDSSenable = true;
    var providerDSSDH = "Microsoft Base DSS and Diffie-Hellman Cryptograph
```

```
var providerDSSDHenable = true;

var userAgent = navigator.userAgent;
if (userAgent.charAt(userAgent.indexOf("MSIE") + 5) >= ’5’) {
    // MSIE version 5.0 or superior
    if ((userAgent.indexOf("95") >= 0) || (userAgent.indexOf("NT") >=
        // Windows 95 or NT
        providerDSSenable = false;
} else {
    // MSIE version 4.X or inferior
    providerDSSDHenable = false;
    providerDSSenable = false;
}

// Save Enroll.ProviderType
var providerTypeSave = xenroll.ProviderType;

// Check provider types
for (var providerType = 0; providerType <= MAX_PROVIDER_TYPES; provide
    xenroll.ProviderType = providerType;

    // Check providers in type
    try {
        var providerIndex = 0;
        // repeat until ’no more providers’ exception
        while (true) {
            var provider = xenroll.enumProviders(providerIndex, 0);

            // Skip conditional providers
```

```
                if ((providerDSSenable || (provider != providerDSS))
                    && (providerDSSDHenable || (provider != providerDSSDH)

                    // Add provider to option list
                    var option = document.createElement("Option");
                    option.text = provider;
                    option.value = providerType;
                    select.add(option);

                    providerCount++;
                }
                providerIndex++;
            }
        } catch (e) {
            // Verify 'no more providers' exception (0x80070103)
            if (e.number != -2147024637) throw e; // other exception
        }
    }

    // Restore XEnroll.ProviderType
    xenroll.ProviderType = providerTypeSave;

    return providerCount;
}

function mp_createRequest(form) {
<?php if ($isMSIE) { ?>
     return mp_msie_submitForm(form);
<?php } else { ?>
```

```
     return true;
<?php } ?>
}

//---------------------------------------------------------------
// Strings to be localized
//var L_CspLoadErrNoneFound_ErrorMessage = "An unexpected error occurred w
//var L_CspLoadErrUnexpected_ErrorMessage = "\"An unexpected error (\"+sEr
//var L_StillLoading_ErrorMessage = "This page has not finished loading ye
//var L_Generating_Message = "Generating request...";
//var L_Waiting_Message = "Waiting for server response...";
//var L_ErrNameUnknown_ErrorMessage = "(unknown)";
//var L_SugCauseNone_ErrorMessage = "No suggestion.";
var L_SugCauseBadCSP_ErrorMessage = "The CSP you chose was unable to proce
var L_SugCauseBadSetting_ErrorMessage = "The CSP you chose does not suppor
var L_SugCauseBadChar_ErrorMessage="You entered an invalid character. Repo
//var L_BadChars_ErrorMessage="No field may contain the characters (, ; \'
//var L_CertGenFailed_ErrorMessage="\"An error occurred while creating the

function mp_msie_handleException(e) {
    var errorName = e.name;
    var errorMessage = e.message;
    var errorNumber = "0x" + (-e.number).toString(16).toUpperCase();

    if (e.number == -2146893816) { // 0x80090008
        errorName = "NTE_BAD_ALGID";
        errorMessage = L_SugCauseBadCSP_ErrorMessage;

    } else if (e.number == -2146893802) { // 0x80090016
```

```
    errorName = "NTE_BAD_KEYSET";
    errorMessage = L_SugCauseBadCSP_ErrorMessage;

} else if (e.number == -2146893799) { // 0x80090019
    errorName = "NTE_KEYSET_NOT_DEF";
    errorMessage = L_SugCauseBadCSP_ErrorMessage;

} else if (e.number == -2146893792) { // 0x80090020
    errorName = "NTE_FAIL";
    errorMessage = L_SugCauseBadCSP_ErrorMessage;

} else if (e.number == -2146893815) { // 0x80090009
    errorName = "NTE_BAD_FLAGS";
    errorMessage = L_SugCauseBadSetting_ErrorMessage;

} else if (e.number == -2146885630) { // 0x80092002
    errorName = "CRYPT_E_BAD_ENCODE";

} else if (e.number == -2146885598) { // 0x80092022
    errorName = "CRYPT_E_INVALID_IA5_STRING";
    errorMessage = L_SugCauseBadChar_ErrorMessage;

} else if (e.number == -2146885597) { // 0x80092023
    errorName = "CRYPT_E_INVALID_X500_STRING";
    errorMessage = L_SugCauseBadChar_ErrorMessage;

} else if (e.number == -2147418113) { // 0x8000FFFF
    errorName = "E_UNEXPECTED";
}
```

```
    alert("Erro " + errorNumber
        + (typeof(errorName) != 'undefined' ? " (" + errorName + ")" : "")
        + (typeof(errorMessage) != 'undefined' ? "\n\n" + errorMessage : '

    // reset XEnroll so the user can select a different CSP, etc.
    XEnroll.reset();
}

function mp_msie_submitForm(form) {

    // the distinguished name is not used for enterprise CAs
    var sDistinguishedName = "";
    sDistinguishedName += "C=\"" + "x" + "\";";
    sDistinguishedName += "S=\"" + "x" + "\";";
    sDistinguishedName += "L=\"" + "x" + "\";";
    sDistinguishedName += "O=\"" + "x" + "\";";
    sDistinguishedName += "OU=\"" + "x" + "\";";
    sDistinguishedName += "E=\"" + "x" + "\";";
    sDistinguishedName += "CN=\"" + form.commonName.value + "\";";

//    // set defaults for values we need on install
/*
    // set defaults for values we need on install
    //document.SubmittedData.CertAttrib.value = "UserAgent:<%=Request.Serv
    document.SubmittedData.CertAttrib.value="UserAgent:Mozilla/4.0 (compat
    document.SubmittedData.TargetStoreFlags.value = 0; // 0=Use default (=
    document.SubmittedData.SaveCert.value = "no";
    document.SubmittedData.Mode.value = "newreq";
```

```
    //document.SubmittedData.FriendlyType.value = "<%=rgAvailReqTypes(CInt
    document.SubmittedData.FriendlyType.value="Web Browser Certificate";
    // append the local date to the type
    document.SubmittedData.FriendlyType.value += " (" + (new Date()).toLoc
*/

    // set the cert type information
    var sCertUsage="1.3.6.1.5.5.7.3.2";
    /*
    If "1.3.6.1.5.5.7.3.4"=sCertUsage Then 'e-mail Protection
     XEnroll.EnableSMIMECapabilities=True
    End If
    */

     // some constants defined in wincrypt.h: (line ~234)

    try {
    // set the CSP
    var nCSPIndex = form.msie_provider.selectedIndex;
    XEnroll.ProviderName = form.msie_provider.options[nCSPIndex].text;
    var nProvType = form.msie_provider.options[nCSPIndex].value
    XEnroll.ProviderType = nProvType;

    var AT_KEYEXCHANGE = 1;
    var AT_SIGNATURE = 2;
    var PROV_DSS = 3;
    var PROV_DSS_DH = 13;
    // default to exchange keys, unless we're doing DSS which only does si
    if ((nProvType == PROV_DSS) || (nProvType == PROV_DSS_DH))
```

```
      XEnroll.KeySpec = AT_SIGNATURE;
    else
      XEnroll.KeySpec = AT_KEYEXCHANGE;

    var CRYPT_EXPORTABLE = 1;
    var CRYPT_USER_PROTECTED = 2;

    // set 'exportable key'
    if (form.msie_crypt_exportable.checked)
      XEnroll.GenKeyFlags |= CRYPT_EXPORTABLE;

    // set 'Strong private key protection'
    if (form.msie_crypt_user_protected.checked)
      XEnroll.GenKeyFlags |= CRYPT_USER_PROTECTED;

//set key size

var keySizeIndex = form.msie_keysize.selectedIndex;
//alert(form.msie_keysize.options[keySizeIndex].value);
    XEnroll.GenKeyFlags |= (form.msie_keysize.value << 16);

    // build the certificate request
    form.msie_pkcs10.value = XEnroll.CreatePKCS10(sDistinguishedName, sCer

    if(form.msie_pkcs10.value.length < 1)
    {
      alert("Erro na gera\c{c}\~{a}o das chaves");
      form.focus();
      return false;
```

```
 }
 // Submit the cert request and move forward in the wizard
 //document.SubmittedData.submit();


// XEnroll.GenKeyFlags |= (form.msie_keysize.value << 16);


             //XEnroll.KeySpec = AT_KEYEXCHANGE;
 /*
             XEnroll.RequestStoreFlags = 0x20000;

             var genKeyFlags = CRYPT_EXPORTABLE;
             if (form.protectkey.checked)
                 genKeyFlags |= CRYPT_USER_PROTECTED;

             XEnroll.GenKeyFlags = (form.msie_keysize.value << 16) | ge

             XEnroll.GenKeyFlags = (form.msie_keysize.value << 16)
                 | CRYPT_EXPORTABLE | CRYPT_USER_PROTECTED;

             // Create certificate request
             form.msie_data.value = XEnroll.createPKCS10(
                     "CN="    + form.commonName.value
                 + ", Email=" + form.emailAddress.value
                 + ", O="     + form.organizationName.value
                 + ", OU="    + form.organizationalUnitName.value
                 + ", L="     + form.localityName.value
                 + ", ST="    + form.stateOrProvinceName.value
```

```
                    + ", C="     + form.countryName.value, "");
    */

            } catch (e) {
                mp_msie_handleException(e);
                return false;
            }

            return true;
}
</script>
```

8.29 Formulário de Requisição de Certificado

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www
<html>

<script language="javascript" type="text/javascript" src="requestCreator.p

<OBJECT CLASSID="clsid:127698e4-e730-4e5c-a2b1-21490a70c8a1"  CODEBASE=xen

<body onload="mp_onLoad(document.requisicao);" language="javascript">

<script language="javascript" type="text/javascript" src="util.js"></scrip

<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Requisi\c{c}\~{a}o de Certificados</title>
</head>
```

```
<body>
<img src="Images/rnp_logo.gif">
<center><h3>Formul\’{a}rio de requisi\c{c}\˜{a}o de certificado</h3></cent
<center>
<span id="msie_xenroll" style="display: none;"></span>
<form name="requisicao" method="post" action="challenge.php" onsubmit="ret
<input type="hidden" name="id" value="<?php echo($_REQUEST[’id’])?>"/>
<table>
<tr>
<td> Nome completo: </td>
<td><input type="text" name="commonName"> </td>
</tr>
<tr>
<td> Endere\c{c}o de e-mail: </td>
<td><b> <?php echo $subjectEmail;?> </b></td>
</tr>
</table>
<input name="submit" value="Gerar requisi\c{c}\˜{a}o" type="submit" />
  <!-- Apresentar opcoes avancadas -->
  <div id="showAdvancedOptions">
    <p>
      <a href="" onclick="showHide(’advancedOptions’, ’showAdvancedOptions
    </p>
  </div>

  <!-- Opcoes avan\c{c}adas (inicialmente n\˜{a}o vis\’{i}vel) -->
  <div id="advancedOptions" style="display: none;">
```

```
<h3>Op\c{c}\~{o}es avan\c{c}adas</h3>

<!--
    Opcoes avan\c{c}adas (navegadores que usam KEYGEN)
    A visibilidade deste campo \'{e} controlada pela fun\c{c}\~{a}o de
-->
<div id="advancedOptionsKeygen">
  <table>
    <tr>
      <td>Tamanho da chave</td>
      <td><keygen name="keygen" challenge="1234567890" /></td>
    </tr>
  </table>
</div>

<!--
    Opcoes avan\c{c}adas (Microsoft Internet Explorer)
    A visibilidade deste campo \'{e} controlada pela fun\c{c}\~{a}o de
-->
<div id="advancedOptionsMSIE">
  <table>
    <tr>
      <td>Tamanho da chave</td>
      <td>
        <input name="msie_pkcs10" value="" type="hidden" />
        <select name="msie_keysize">
          <option value="2048">2048 (Alta seguran\c{c}a)</option>
          <option value="1024">1024 (M\'{e}dia seguran\c{c}a)</option>
        </select>
```

```
        </td>
      </tr>
      <tr>
        <td>Provedor</td>
        <td><select name="msie_provider"></select></td>
      </tr>
      <tr>
        <td>Op\c{c}\~{o}es</td>
        <td>
          <input type="checkbox" name="msie_crypt_exportable" value="tru
          Possibilitar exporta\c{c}\~{a}o das chaves.
        </td>
      </tr>
      <tr>
        <td></td>
        <td>
          <input type="checkbox" name="msie_crypt_user_protected" value=
          Habilitar seguran\c{c}a adicional no armazenamento da chave pr
        </td>
      </tr>
    </table>
  </div>

  <!-- Fechar opcoes avancadas -->
  <p>
    <a href="" onclick="showHide('showAdvancedOptions', 'advancedOptions
  </p>
</div>
```

```
<input name="subjectEmail" type="hidden" value="<?php echo($subjectEmail)?
    <input name="id" type="hidden" value="<?php echo ($_REQUEST['id'])?>">
</form>
<br>
<font color=red> <?php echo $message?></font>
</center>
</body>
</html>
```