Fabio Antonio Rodrigues

Sistema de Denúncia Anônima Segura

Florianópolis

Junho de 2005

Fabio Antonio Rodrigues

Sistema de Denúncia Anônima Segura

Projeto de Conclusão de Curso

Orientador:
Ricardo Felipe Custódio

Departamento de Informática e Estatística
Universidade Federal de Santa Catarina

Florianópolis

Junho de 2005

Proposta de trabalho de conclusão do curso de Bacharelado em Ciências da Computação da Universidade Federal de Santa Catarina.

Ricardo Felipe Custódio
Orientador

Fernando César Oliveira Lopes
Banca examinadora

Júlio da Silva Dias
Banca examinadora

Resumo

Este trabalho propõe a implementação de um Sistema de Denúncia Anônima Segura. O princípio deste sistema baseia-se na aplicação de protocolos criptográficos para envio de mensagens anônimas, havendo a possibilidade de revelação da identidade do emissor num determinado período de tempo. São consideradas as questões legais sobre anonimato no Brasil e uma lista de requisitos de segurança que o sistema deve atender.

Abstract

This research presents the implementation of an Safe Anonymous Accusation System for the web. The beginning of this System be based on apply of cryptographic protocols for sending of anonymous messages, with possibility of revelation of the identify of the message sender in a determined period of the time. The lawful question about anonymity in Brazil are maintain and a list with safety requirements wich the system should attend.

Agradecimentos

Gostaria de agradecer a meus pais, a Deus e a Marília que me deu força em boa parte do curso.

Sumário

1 Introdução

1.1 Introdução

O Sistema de Denúncia Anônima Segura , baseado na Dissertação de Mestrado de Fernando Cézar de Oliveira Lopes(1), deve garantir plenos direitos constitucionais e penais, respeitando as leis brasileiras, a quem denuncia.

É proposto um protocolo criptográfico que garante a ocultação da identidade de quem produz a mensagem, sendo que, neste sistema, o emissor poderá ser identificado unicamente sob a resolução de algum subgrupo de um grupo de entidades autorizadas. É possível identificar o denunciante somente por um determinado período de tempo, sendo sua identidade revelada apenas se necessário e destruída após um período estabelecido.

Na figura 1(1) pode-se observar como funciona a linha do tempo no Sistema de Denúncia Anônima Segura.

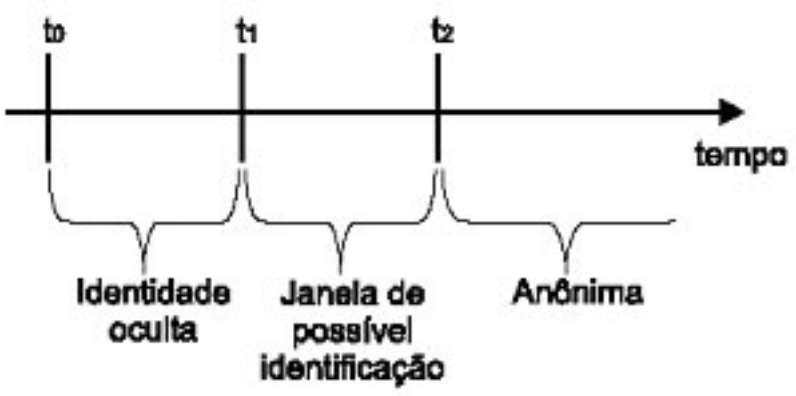

Figura 1: : Linha de tempo: A marcação de ponto t0 indica o momento de envio da mensagem com identidade oculta. Entre t0 e t1, prevalece a ocultação da identidade. O intervalo entre t1 e t2, o período onde pode-se ou não revelar a identidade o emissor. Após t2 a mensagem deve permanecer anônima, e a identidade do emissor deve ser destruída.

O Sistema deve atender alguns requisitos básicos para a garantia de funcionamento correto e seguro.

Segundo Fernando C. De Oliveira Lopes(1), na lista apresentada abaixo constam requisitos de segurança que os protocolos criptográficos propostos devem atender no decorrer de seu funcionamento:

- Anonimato no envio: Deve ser possível ao emissor enviar uma mensagem de forma anônima, ou seja, o emissor deve ter meios para enviar uma mensagem sem identificar-se;
- Confiança distribuída: A identidade do emissor da mensagem pode ser conhecida por qualquer dos subgrupos formados de um grupo de entidades previamente autorizadas, desde que respeite a quantidade mínima de entidades autorizadas necessárias para conhecer a identidade do emissor;
- Anonimato temporal (imparcialidade): A mensagem deve permanecer anônima desde o seu envio pelo emissor até um determinado tempo t1 no futuro;
- Cifra temporal: A identidade do emissor da mensagem poder ser conhecida no período de tempo entre t1 e t2 >t1, caso necessário;
- Destruição da identidade: A mensagem dever permanecer anônima para qualquer t >t2 e após este tempo a identidade deve ser destruída para sempre;
- Aviso ao emissor: O emissor deve saber que a sua identidade foi revelada. No período entre t1 e t2, se a identidade do emissor for revelada ele deve receber uma mensagem padrão de notificação do ocorrido;
- Autenticação: À toda mensagem anônima deve ser passível, identificar seu emissor no período de tempo específico, sendo que a identidade do emissor deve estar contida corretamente;
- Prova (não-coação): O emissor de uma mensagem anônima, não pode provar que foi ele que a emitiu; ou seja, deve-se garantir a não-ligação entre o emissor e a mensagem emitida de forma anônima;
- Autonomia: O emissor da mensagem não deve precisar confiar em qualquer entidade, a menos que ele queira.

1.2 Objetivos

1.2.1 Objetivo Geral

Propor um Sistema de Denúncia Anônima onde se utiliza protocolos criptográficos para produzir uma mensagem com ocultação da identidade do emissor, sendo qualquer subgrupo de um grupo entidades autorizadas, poder revelar a identidade deste emissor em um período de tempo específico no futuro, $t1 \leq t \leq t2$, onde t1 o início e t2 o final do período. Caso a identidade não seja revelada neste período, esta dever ser destruída após t2.

1.2.2 Objetivos Específicos

- Estudar a Dissertação de Mestrado de Fernando Cézar de Oliveira Lopes, cujo tema é Denúncia Anônima Segura;
- Estudar os protocolos criptográficos apresentados na dissertação que garantem comunicação anônima;
- Estudar os requisitos de segurança necessários para que haja uma comunicação anônima com revelação no futuro da identidade;
- Implementar uma aplicação que baseia-se em um dos protocolos propostos;

1.3 Motivação

Cada vez mais, as pessoas têm sua privacidade invadida devido ao grande número de situações nas quais a Internet é utilizada. Por esse meio são compartilhados os mais variados tipos de informações. Sendo assim, existe a necessidade do uso da Internet de uma forma confiável e segura.

A aplicação do Sistema de Denúncia Anônima propõe um método para denúncia difícil de se fraudar onde o usuário teoricamente não precisa confiar em nenhuma entidade para efetuar a operação. Este meio facilita os métodos legais existentes para que uma denúncia seja feita, respeitando a Constituição e os direitos do cidadão.

1.4 Definição Do Problema

O trabalho consiste em transpor o método de denúncia do mundo real para o mundo virtual, ou seja, transpor a realidade de uma denúncia feita em documento de papel para um documento eletrônico.

Para resolver este problema foram desenvolvidos e apresentador por Fernando Lopes(1) protocolos criptográficos para atender os requisitos do sistema, tais como: Anonimato de envio, Confiança distribuída, Anonimato temporal, Cifra temporal, Destruição da identidade, Aviso ao emissor, Autenticação, Prova e Autonomia.

2 Princípios da Criptografia

2.1 Introdução

A criptografia atualmente é a base de muitas aplicações no mundo eletrônico.

Neste capítulo são descritos brevemente conceitos básicos da criptografia, dos quais torna-se essencial uma compreensão , pois o sistema utiliza-se destes para alcançar seus objetivos.

2.2 Criptografia

Criptografia (kriptós = escondido, oculto; grifo = grafia), palavra de origem grega, é a arte ou ciência de escrever em cifra ou em códigos. Pode ser definida como a arte e ciência de garantir a segurança de mensagens(2), de forma que somente o destinatário, após o processo de decifragem, consiga decodificar e ler a mensagem com clareza.

Através do processo de cifragem da mensagem, a criptografia transforma um texto aberto, ou seja, texto na forma legível e compreensível, em um texto cifrado (texto não legível, codificado). Para retornar ao texto original, ou texto aberto, é feito o processo inverso ao da cifragem, o processo de decifragem. A decifragem transforma o texto cifrado em texto aberto, legível.

Para garantir segurança aos processos acima citados, há uma chave. Apenas quem conhecer a chave poderá transformar um texto codificado em texto aberto.

As chaves criptográficas utilizadas no processo de crifragem e decifragem são divididas em dois tipos: chaves simétricas e chaves assimétricas, as quais os conceitos serão apresentados a seguir.

Os principais serviços da criptografia para garantir que um sistema funcione de forma segura são(3)(4):

1. Confidencialidade: garante que somente pessoas ou entidades autorizadas tenham acesso as informações;

2. Autenticação: garantia de identificação de pessoas ou entidades envolvidas na comunicação;

3. Integridade: garantia de que a informação não sofreu alterações ou foi destruída;

4. Não - Repúdio: Assegura que as entidades não possam negar as ações praticadas por elas;

2.3 Criptografia Simétrica

Segundo o princípio de Kerckhoff(5) a segurança não deve estar no algoritmo usado e sim na chave utilizada.

Na criptografia simétrica é utilizada a mesma chave para cifragem e decifragem das informações. Esta chave deve ser secreta, pois a segurança do algoritmo está na mesma.

A chave deve ser secreta e compartilhada entre os envolvidos, sendo assim, há a necessidade de um canal seguro de comunicação para transmissão da chave.

Os algoritmos de cifragem e decifragem são praticamente os mesmos, o que difere é apenas a forma de como a chave é utilizada. Um exemplo de algoritmo simétrico é o DES.

2.4 Criptografia Assimétrica

Algoritmos assimétricos, também chamados de algoritmos de chave pública, apresentam como principal característica o uso de duas chaves diferentes para cifrar e decifrar, uma chave pública e outra privada. As chaves possuem uma relação, pois o que uma cifra, só a outra decifra.

Os algoritmos assimétricos possuem aplicações como: autenticação e confidencialidade. Também é possível conseguir autenticação e confidencialidade ao mesmo tempo, utilizando-se um processo híbrido, ou seja, pode-se cifrar usando a chave privada do emissor e a chave pública do destinatário. Assim é garantida autenticação, pois como o emissor utilizou a sua chave privada, somente a sua chave pública poderá decifrar a mensagem, assegurando que quem enviou a mensagem foi o emissor. Utilizando a chave pública do destinatário, é alcançada confidencialidade, pois somente o dono da chave privada

conseguirá decifrar a mensagem, ou seja, somente o destinatário conseguirá recuperar a mensagem original, cifrada com a sua chave pública.

Não deve haver possibilidade de determinar a chave privada tendo conhecimento da chave pública e do algoritmo utilizado.Um exemplo de algoritmo assimétrico é o RSA.

2.5 Assinatura Digital

Para garantir a autoria de um documento usa-se a assinatura. Tanto para assinatura em um documento em papel, quanto para assinatura em um documento eletrônico deve existir meios que possibilitem identificar de maneira única o seu autor.

Suas aplicações no campo de segurança são muitas, principalmente nos serviços de autenticação, integridade e não-repúdio.(4)

Uma assinatura digital deve fornecer características de uma assinatura do mundo real. Algumas características desejáveis são(3):

- A assinatura digital deve ser simples de produzir por quem assina;
- Deve ser fácil de ser verificada por qualquer pessoa;
- Deve ser muito difícil de ser falsificada;
- Quem assina não pode negar que assinou (não-repúdio).

O algoritmo padrão americano para geração e verificação de assinaturas digitais é o DSA, que utiliza o algoritmo SHA-1 para geração de resumos, que será explicado posteriormente. Além do DSA, existe o algoritmo RSA, ElGamal, entre outros.

2.6 Função Resumo

Função Resumo, ou *hash*, é uma função que aplicada sobre um documento eletrônico, independente do tamanho, gera um resumo de tamanho fixo(2). É uma função, que independente do tamanho da sua entrada, gera como saída um resumo identificador de tamanho único.

Esse resumo identifica um documento de forma singular; é a impressão digital dos documentos eletrônicos. Dois textos diferentes nunca geram dois resumos iguais. Por ser um resumo identificador exclusivo, pode ser usado para autenticação e integridade.

Uma função resumo h deve atender a determinadas propriedades(4):

1. Compressão: aplicando a função h sobre um bloco de dados x de qualquer tamanho, o resultado é uma saída y de tamanho fixo;
2. Fácil computação: tendo a função h e a entrada x, é relativamente fácil computar $h(x)$;
3. Caminho único: é impraticável computacionalmente deduzir o valor de entrada x a partir do valor de saída y;
4. Fraca resistência a colisão: dado x, é impraticável encontrar um valor x' tal que $h(x) = h(x')$;
5. Forte resistência a colisão: é impraticável computacionalmente encontrar duas entradas distintas, x e x', que produzam o mesmo resumo.

Alguns algoritmos para geração da função resumo são SHA-1, MD4, MD5, HMAC.

2.7 Certificado Digital

Para um melhor entendimento de certificados digitais, podemos fazer uma analogia com a carteira de identidade. Se alguém deseja se identificar pode usar a sua carteira de identidade, emitida por um órgão confiável para tal. No mundo virtual usa-se o Certificado Digital.

Deve ser de fácil percepção qualquer tipo de tentativa de fraude no certificado, ou seja, caso o impostor tenha pegado um certificado existente e substituído a chave pública ou o nome ali contido, qualquer pessoa que examinar esse certificado fraudado saberá que se trata de uma fraude.

Um certificado associa uma chave Pública e um nome único para um usuário. O padrão mais amplamente utilizado é o X.509 v3.

Um certificado pode ser solicitado a Autoridades Certificadoras (AC). Uma AC é responsável por gerenciar as chaves públicas e os certificados digitais. É um terceiro confiável para associar uma identidade de uma pessoa a sua chave pública. São as entidades que emitem e assinam os certificados digitais.

2.8 Conclusão

Este capítulo apresentou conceitos sobre criptografia os quais serão necessários para entendimento do trabalho. Esses conceitos aliados aos conceitos explicados nos capítulos seguintes formarão a base e o fundamento teórico do Sistema apresentado neste trabalho.

3 Referência Teórica

3.1 Introdução

O Sistema de Denúncia Anônima Segura deve atender a requisitos de segurança para um funcionamento correto e seguro. Esses requisitos são importantes na definição de um protocolo criptográfico.

O protocolo proposto por Fernando C. O. Lopes(1), para poder atender às condições, deve usar Rede de Misturadores, Quebra-Cabeça Temporal, Compartilhamento de Segredo e Bloco Inverso.

Este capítulo apresentará conceitos importantes para entendimento dos requisitos de segurança e do protocolo proposto.

3.2 Autenticação

> "Se não for possível identificar uma pessoa que esteja tentando entrar em um sistema, então como se pode garantir a segurança(6)."

A autenticação é um aspecto muito importante a ser considerado em sistemas eletrônicos. Garantir que somente pessoas ou entidades autorizadas tenham acesso às informações pode ser uma tarefa difícil. Ter certeza que essas entidades são realmente quem elas dizem ser é uma questão delicada. Sendo assim, não controlar o acesso a determinadas informações é inviável.

Existem vários tipos de autenticação. Pessoas podem se identificar através de algo que elas possuem (como chaves, cartões), algo que elas sabem (senhas), ou, ambos (cartão e senha, pegando como exemplo cartão bancário) entre outros.

Na escolha do sistema de autenticação e de seu grau de segurança, deve ser considerado o valor da informação que será guardada.

A autenticação de um usuário é o principal aspecto para a segurança da informação(1).

No SDAS, para os usuários terem acesso ao sistema, será necessário possuírem um certificado digital. Após passar pela tela de autenticação, o usuário, já autenticado, terá acesso ao cadastro e consultas das denúncias.

3.3 Anonimato

O anonimato é um tópico muito importante deste trabalho. Apesar de ser um sistema de denúncia anônima, deve respeitar as questões legais de anonimato no Brasil, onde ao se fazer uma denúncia, o denunciante deve assumir as conseqüências de seus atos. Há a liberdade de expressão, porém, deve haver um responsável por essa expressão.

Ao se fazer uma denúncia é aberto um Inquérito policial. Este inquérito entra na fase de investigação e tem 30 dias para ser concluído.

Com base nas questões legais, o SDAS segue uma linha de tempo(1), onde o cadastro da denúncia é o ponto *t0*. A partir de *t0* até *t1* a identidade fica ocultada. Entre *t1* e *t2* há o período onde se pode revelar a identidade, caso necessário. A partir de *t2* a identidade deve ser destruída.

Para conseguir as características de anonimato conforme a linha do tempo acima citada, foi necessário utilizar criptografia temporal. O protocolo proposto irá utilizar um quebra cabeça temporal. Este quebra cabeça temporal irá permitir a revelação da identidade somente no espaço de tempo entre *t1* e *t2*. Este tema será abordado no capítulo seguinte.

Também será utilizada uma rede de misturadores. O modelo de rede de misturadores(1) faz com que uma mensagem cifrada passando por um conjunto de servidores, sofra, a decifragem na entrada, e permutação e cifragem na saída. Com este processo, se obtém a garantia de que não existe uma ligação entre a mensagem de entrada e a de saída.

3.4 Criptografia Temporal

Segundo Fernando Carlos Pereira(7), a criptografia temporal permite a uma pessoa determinar em qual momento do futuro a informação poderá ser acessada. E através da cifragem dos dados a confidencialidade desta informação é garantida. Ou seja, a informação só será revelada num futuro pré-determinado.

O seu funcionamento consiste em cifrar a informação a ser protegida. A chave crip-

tográfica necessária para a decifragem dessa informação fica oculta durante um determinado período, no qual, a informação deve ficar escondida. Para garantir o funcionamento desse processo existem métodos(8) como o quebra-cabeça temporal e o de entidades confiáveis.

O funcionamento do quebra-cabeça temporal consiste na resolução de um problema que só será resolvido se um computador o ficar processando continuamente por um período pré-determinado. Após a resolução, a informação poderá ser acessada.

Entidades confiáveis, consistem em agentes ou estruturas que guardam a informação até o momento da sua revelação.

O SDAS deverá utilizar um quebra cabeça temporal para garantir os requisitos de segurança necessários ao seu funcionamento. Porém, a resolução desse quebra cabeça pode ser um problema, pois novas máquinas mais potentes surgem a cada momento. E o quebra cabeça deve garantir, que independente do equipamento ou tecnologia usada, a informação deve permanecer ocultada até o possível momento de sua revelação.

3.5 Compartilhamento de Segredos

O compartilhamento de segredos é interessante, pois não existe confiança mútua entre os envolvidos. Há uma distribuição de responsabilidade(1). Este esquema divide um segredo entre um grupo e para recuperar essa informação, todos ou parte desse grupo devem estar de acordo.

Este conceito utiliza-se de confiança distribuída, ou seja, não basta um querer, mas sim todos ou parte dos integrantes devem concordar. Assim a decisão não fica somente a controle de uma única pessoa ou entidade.

A divisão do segredo pode obrigar todos os membros a estarem de acordo ou apenas algum subconjunto dentro do total de participantes. Um subconjunto autorizado é aquele que possui o número mínimo de participantes necessários para reconstruir o segredo(7).

3.6 Divisão de Segredo do Tipo (n,n)

Este tipo de divisão divide o segredo em partes iguais. Para reconstrução, necessita-se de todas as partes envolvidas. É do tipo (n,n), pois dos n envolvidos, os n tem que estar de acordo.

Este esquema é considerado seguro, já que uma pessoa ,ou, entidade isolada não tem condições de obter o segredo. Porém, possui a desvantagem de que todos os envolvidos devem estar de acordo para ser possível a reconstrução do segredo.

Baseia-se na operação *XOR*, e é utilizada uma função one-time-pad. Tais funções garantem uma segurança incondicional ao esquema e ainda possuem a vantagem de serem fáceis de utilizar(5).

3.7 Divisão de Segredo do Tipo (m,n)

A divisão de segredo do tipo (m,n) não necessita de todas as entidades envolvidas, mas sim m partes do total de n. Quaisquer das m partes conseguem reconstruir o segredo.

O esquema limiar de Shamir(9), é do tipo (m,n) e baseia-se na interpolação de polinômios, onde as partes do segredo são representadas por pontos em um plano bi-dimensional *(xi, yi), i = 1,...,n.* A segurança deste esquema está na propriedade de existir um, e somente um, polinômio *f(x)* de grau *t-1* tal que *f (xi) = yi* para todo *i*.

As propriedades necessárias para um esquema limiar segundo Shamir são(9)(4):

- Perfeito: o esquema é considerado perfeito quando se conhecendo t-1 ou menos partes do segredo, não é possível determinar o segredo correto;
- Ideal: quando o tamanho das partes é igual ao tamanho do segredo;
- Extensível para novos usuários: deve ser possível calcular novas partes do segredo para serem entregues a novos participantes, sem que para isso as partes já distribuídas sejam afetadas;

3.8 Comunicação em grupo

Este conceito baseia-se na assinatura em grupo(10). Ou seja, permite que membros ou entidades manifestem-se em nome do grupo. Sabe-se que a assinatura pertence ao respectivo grupo, porém não se sabe qual membro assinou. Apresenta algumas propriedades:

- Apenas membros do grupo podem assinar mensagens;

- O receptor da mensagem pode verificar que esta é uma assinatura válida para o grupo, porém não pode descobrir a qual dos membros do grupo ela pertence;
- Em caso de disputa futura, ou seja, se por algum motivo houver necessidade futura de identificar o membro que emitiu uma determinada mensagem, a assinatura pode ser aberta, com ou sem a ajuda dos membros individuais do grupo, revelando a identidade do emissor.

Esta visão sobre comunicação em grupo é dada pelo fato que o SDAS deve garantir a correta participação de indivíduos em um grupo(1).

3.9 Conclusão

Neste capítulo foram apresentados conceitos, que aliados aos apresentados no capítulo anterior, serão utilizados para entendimento deste trabalho.

Sem uma base teórica, seria difícil o entendimento do protocolo proposto por Fernando Lopes(1) em sua dissertação de mestrado.

Os conceitos apresentados são fundamentais para a resolução do problema da denúncia anônima segura e de seus requisitos de segurança. Sem isso não seria possível o desenvolvimento do protocolo proposto.

4 Protocolo Proposto

4.1 Introdução

Um protocolo é uma série de passos, envolvendo um ou mais participantes, objetivando executar uma tarefa específica(2). No caso deste trabalho, a tarefa é um sistema de denúncia anônima segura, e o protocolo apresentando por Fernando Lopes(1) irá especificar uma série de etapas que serão executas, de forma ordenada, para o funcionamento correto do sistema.

Um protocolo apresenta requisitos de segurança. Os requisitos dão uma visão do que ele pode fazer(1). No caso do SDAS estes requisitos são: anonimato no envio, confiança distribuída, anonimato temporal, cifra temporal, destruição da identidade, aviso ao emissor, autenticação, prova e autonomia. No decorrer do capítulo voltarei a falar sobre eles.

4.2 Protocolos Criptográficos

Criptosistemas quando utilizados somente para cifragem e decifragem têm uma utilidade restrita. Para maior utilidade, a criptografia deve prover meios para a troca de mensagens com segurança. Os protocolos criptográficos geralmente têm o objetivo de criar aplicações mais complexas do que apenas cifrar e decifrar.

A utilização de um protocolo adequado e eficiente é de extrema importância para garantir a segurança das mensagens trocadas durante as transações e o correto funcionamento do sistema.

Para que uma série de passos possa ser considerada um protocolo, é necessário que as seguintes características sejam satisfeitas(2)(4):

1. Deve existir uma seqüência do protocolo do início ao fim;

2. Deve conter duas ou mais entidades envolvidas no funcionamento do protocolo;
3. Todos os participantes de um protocolo devem conhecê-lo e saber todas as suas etapas para segui-lo;
4. Todos os participantes de um protocolo devem concordar com o seu funcionamento;
5. O protocolo não deve apresentar ambigüidade, ou seja, cada etapa deve ser bem definida e não permitir uma má compreensão;
6. O protocolo deve ser completo, ou seja, deverá ser especificada uma ação para cada situação possível;
7. Não deve ser possível fazer mais ou aprender mais do que está especificado no protocolo.

4.3 Requisitos de Segurança

Todo sistema deve atender a alguns requisitos básicos para garantia de funcionamento correto e seguro(1).

A lista de requisitos para atender o SDAS segundo Fernando Lopes é a seguinte:

1. Anonimato no envio: deve ser possível ao emissor enviar uma mensagem de forma anônima;
2. Confiança distribuída: A identidade do emissor pode ser conhecida por qualquer subgrupo formado de um grupo de entidades previamente autorizadas;
3. Anonimato temporal: A mensagem deve permanecer anônima desde o seu envio até um determinado tempo t1 no futuro;
4. Cifra temporal: A identidade do emissor da mensagem poderá ser conhecida no período de tempo entre t1 e t2;
5. Destruição da identidade: A mensagem deverá permanecer anônima após t2;
6. Aviso ao emissor: O emissor deve saber que sua identidade foi revelada;
7. Autenticação: A toda mensagem anônima deve ser possível identificar o emissor, caso necessário e no tempo específico;

8. Prova (não-coação): O emissor de uma mensagem anônima não pode provar que foi ele que a emitiu;

9. Autonomia: O emissor não precisa confiar em qualquer entidade.

4.4 Protocolo Proposto

O protocolo proposto baseia-se em um Protocolo com Terceiro Intermediário. Esse terceiro é uma entidade confiável responsável por receber a denúncia, enviar e liberar a identidade no futuro, caso seja possível.

O seu funcionamento consiste em:

1. Um usuário preencher o formulário de cadastro de denúncia;

2. Os dados são enviados para o sistema;

3. É gerada uma senha;

4. A identidade é cifrada com a senha;

5. É calculada a função resumo da denúncia;

6. Solicita-se a datação;

7. A divisão de segredo é gerada;

8. É enviado aos administradores do sistema um e-mail com a denúncia e uma parte da chave;

9. Os dados são armazenados no banco.

A quebra do anonimato funciona da seguinte forma:

1. Os administradores do sistema solicitam a quebra do anonimato;

2. A chave é reconstruída, a partir das chaves enviadas aos administradores;

3. É verificada a data, para a quebra de anonimato;

4. Se possível a quebra do anonimato, a identidade é decifrada;

5. A identidade é mostrada.

Esse protocolo desenvolvido atende a quase todos os requisitos de segurança. Sua análise será feita a seguir:

- Anonimato no envio: é garantido, pois se utiliza um terceiro intermediário, que é responsável por receber a denúncia e enviar e-mail aos administradores. Não existe ligação entre os administradores e quem fez a denúncia;
- Confiança distribuída: não atende, pois existe uma entidade confiável que é responsável por receber a denúncia, enviar e-mails e quebrar anonimato. Os usuários do sistema têm que confiar nessa entidade;
- Anonimato temporal: Atende, pois a entidade controla o tempo, e libera a identidade somente se respeitar a linha do tempo da denúncia;
- Cifra temporal: não atende, pois essa função é feita pelo terceiro intermediário;
- Destruição da identidade: Atende, pois a entidade após verificar a data e concluir que a identidade não pode ser mais revelada, destrói a mesma;
- Aviso ao emissor: Atende, pois o usuário receba uma mensagem caso sua identidade for revelada;
- Autenticação: Atende, pois a entidade confiável conhece a identidade, e fornece caso necessário e possível;
- Prova: Atende, pois somente o terceiro intermediário pode ligar a denúncia a pessoa que a fez;
- Autonomia: Não atende, pois tudo depende da entidade confiável.

4.5 Conclusão

Este protocolo atende grande parte dos requisitos de segurança propostos. Porém o seu funcionamento baseia-se em uma entidade confiável no sistema. Essa entidade é responsável por enviar e-mails e administrar a identidade do denunciante, liberando a mesma somente se possível ou seja, é obedecida a linha do tempo da denúncia.

Sem as chaves enviadas aos administradores, por e-mail juntamente com a denúncia, não é possível recompor a chave original, a qual foi usada para cifrar a identidade. Então, a entidade confiável não tem condições de decifrar sozinha a identidade, pois necessita das chaves enviadas aos administradores para recompor a chave original (divisão de segredo).

Requisitos do Sistema	*Análise*
Anonimato no envio	O.k.
Confiança distribuída	Não atende
Anonimato Temporal	O.k.
Cifra Temporal	Não atende
Destruição da identidade	O.k.
Aviso ao emissor	O.k.
Autenticação	O.k.
Prova	O.k.
Autonomia	Não atende

Tabela 1: Resultado da análise do protocolo

5 *Implementação - Sistema de Denúncia Anônima Segura*

5.1 Introdução

Com a finalidade de demonstrar o funcionamento do protocolo com terceiro intermediário foi implementado um sistema, o SDAS.

Esse sistema baseia-se em uma entidade confiável no sistema e utiliza-se de criptografia para proteger a identidade e libera-la somente se respeitar a linha do tempo da denúncia.

Para o desenvolvimento do trabalho optou-se pela linguagem JAVA(11) e Java Server Pages (JSP). O banco de dados escolhido foi o Firebird(12). Também são utilizados Javascript e HTML. Para os certificados digitais foi utilizada a biblioteca CAPICOM da Microsoft(13) e o servidor web utilizado foi o TOMCAT(14).

Foi escolhida a linguagem java por apresentar algumas características como ser multiplataforma, suportar orientação a objetos, possuir pacote java security, o qual tem classes bastante usadas e testadas, tornando-se confiável e por possuir grande número de desenvolvedores, facilitando a pesquisa e discussão nas várias listas e fóruns existentes.

5.2 Detalhes do Projeto

O sistema desenvolvido é usado para o cadastro de denúncias. O usuário (denunciante) preenche um formulário web e os dados sobre a denúncia são armazenados no banco.

Caso seja necessário quebrar o anonimato, existe a parte do sistema reservada aos administradores do sistema. Os administradores têm acesso às denúncias cadastradas e tem as opções de publicar, consultar ou quebrar o anonimato de uma denúncia.

Todo o funcionamento do sistema baseia-se nas questões legais sobre anonimato no Brasil. Como foi citado anteriormente, o sistema é considerado uma entidade confiável,

sendo o emissário dos e-mails aos administradores, controlando a data e caso seja possível, liberando a identidade do denunciante.

5.3 Detalhes da Implementação

O SDAS é dividido em duas partes. A primeira é a parte onde os usuários cadastram ou consultam as denúncias. E a outra parte é reservada aos administradores do sistema.

Na primeira parte, os usuários, ao acessarem o sistema selecionaram o seu certificado digital - Figura 2, após visualizaram uma tela com a opção de cadastrar a denúncia ou consultar denúncias, conforme figura 3.

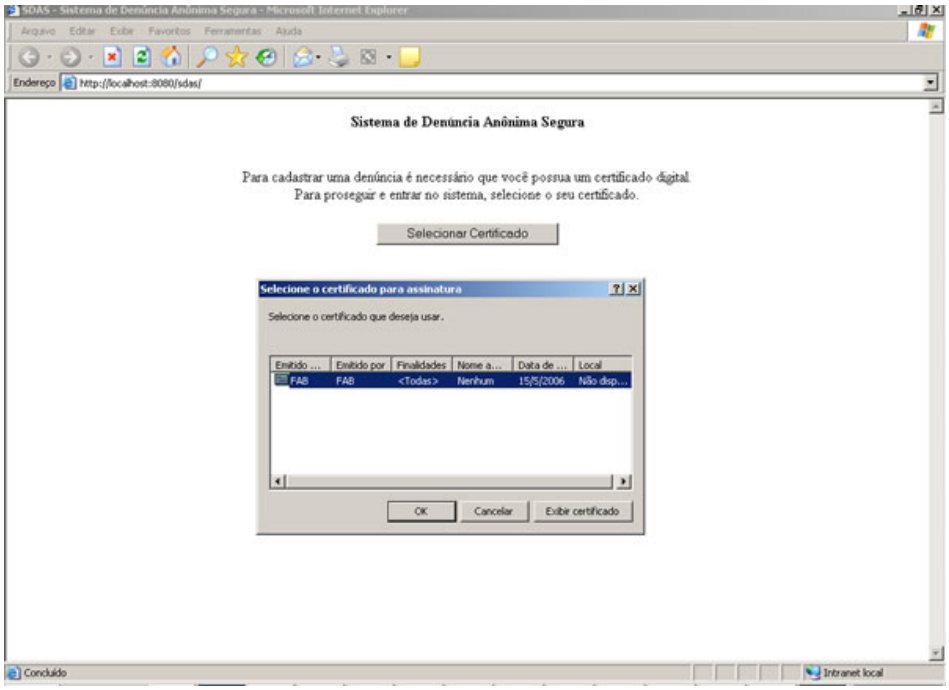

Figura 2: seleção do certificado.

Caso a escolha tenha sido "consulta de denúncias" o usuário irá para a página onde é exibida uma lista com as denúncias publicadas, conforme Figura 4. Apenas os administradores têm a opção de publicar uma denúncia. Os usuários tem apenas a opção de visualizar uma denúncia cadastrada - Figura 5.

Se o usuário optou por cadastrar uma denúncia ele irá para uma página, Figura 6, com um formulário para cadastro. Os campos contidos no formulário são:

- Título;

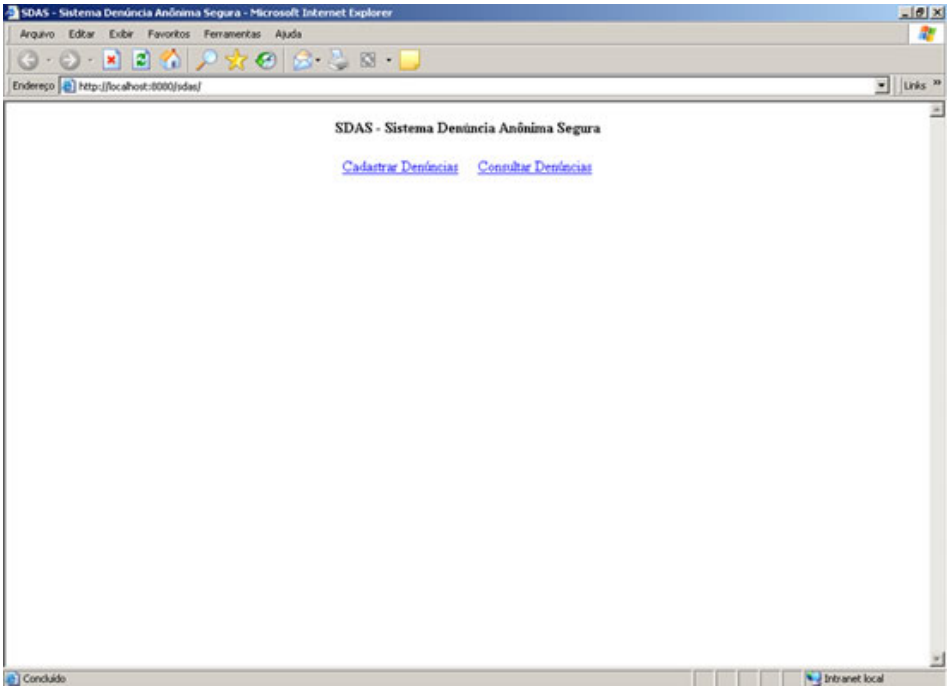

Figura 3: tela com a opção de cadastro ou consulta de denúncias.

- Descrição;
- Local;
- Data do Ocorrido;
- Hora Ocorrido;
- Número Certificado (já vem preenchido, conforme certificado o selecionado).

Os dados, depois de preenchidos e enviados para o sistema, são concatenados e a função resumo é gerada. Uma chave é gerada aleatoriamente. Esta chave será usada para cifrar a identidade e após a cifragem é feita a divisão de segredo da mesma, onde as chaves geradas são enviadas aos administradores . O processo segue na seguinte ordem:

1. Dados da denúncia são enviados ao sistema;
2. As informações são concatenadas;
3. A chave é gerada;
4. A identidade é cifrada utilizando a chave;
5. A função resumo (MD5) dos dados é gerada;

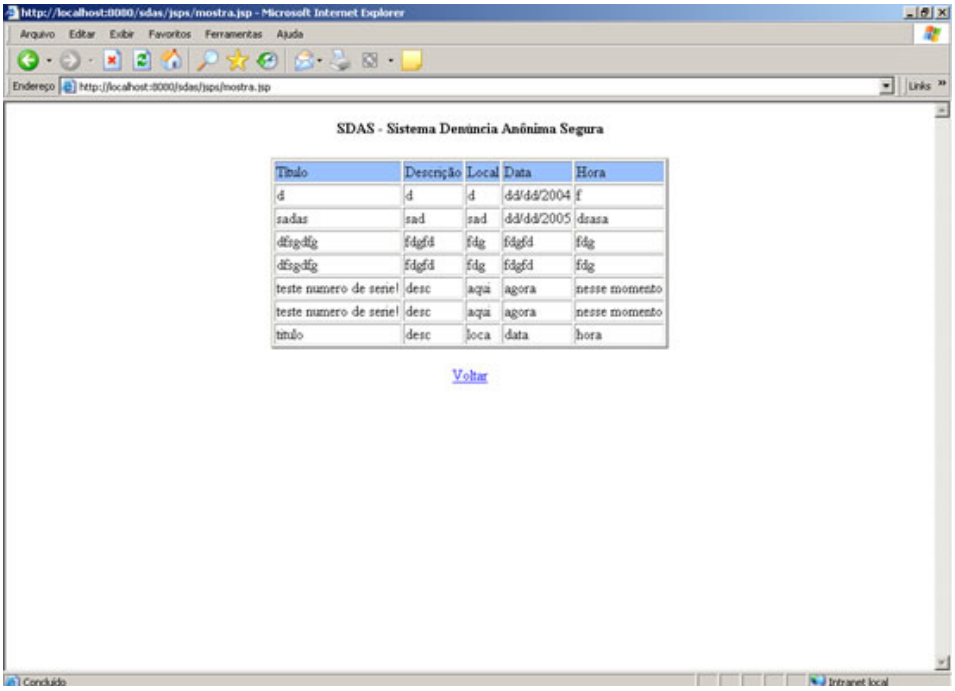

Figura 4: tela com a lista de denúncias publicadas.

6. Solicita-se a datação;
7. A divisão de segredo é gerada a partir da chave;
8. Um e-mail é enviado aos administradores com um identificador da denúncia, o conteúdo da denúncia e a parte da chave gerada com a divisão de segredo;
9. Os dados são armazenados no banco.

No banco de dados são armazenados os seguintes dados:

- Título;
- Descrição;
- Local;
- Hora;
- Data;
- Número do certificado;
- Hash;

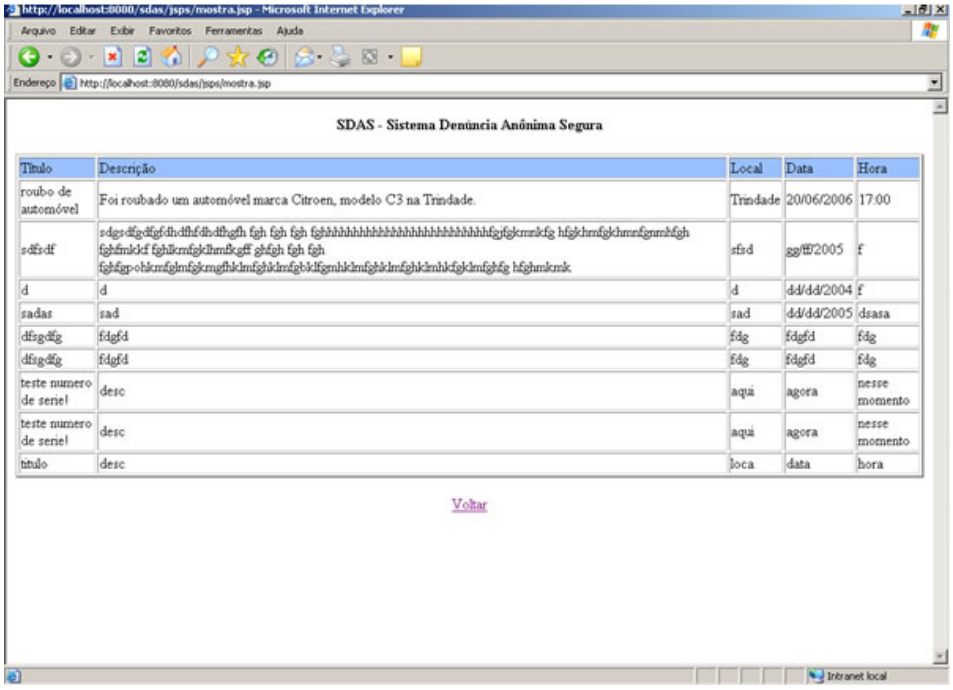

Figura 5: visualização de uma denúncia.

- Identidade (cifrada);
- Data Cadastro;
- Hora Cadastro.

A outra parte, seria a parte de controle do sistema. Essa parte é reservada aos administradores. Para acessar essa parte do SDAS é necessário um login e senha, conforme a figura 7.

Depois de identificado, o administrador vai para a tela onde pode acessar as denúncias publicadas e não publicadas - Figura 8. Ambas as opções levam a tela com a listagem das denúncias, porém cada uma lista as denúncias publicadas e as não publicadas respectivamente.

Na listagem de denúncias - Figura 9, aparece o identificador da denúncia, que o administrador também recebe por e-mail e serve para facilitar a identificação da denúncia que deseja consultar, e os dados da denúncia: titulo, descrição, local, data, hora e o link com a opção de quebrar o anonimato.

A partir da tela citada acima, o administrador tem as opções de quebrar o anonimato e de publicar (ou tirar de publicação) uma denúncia. No caso de publicar, o parecer do juiz terá que ser preenchido.

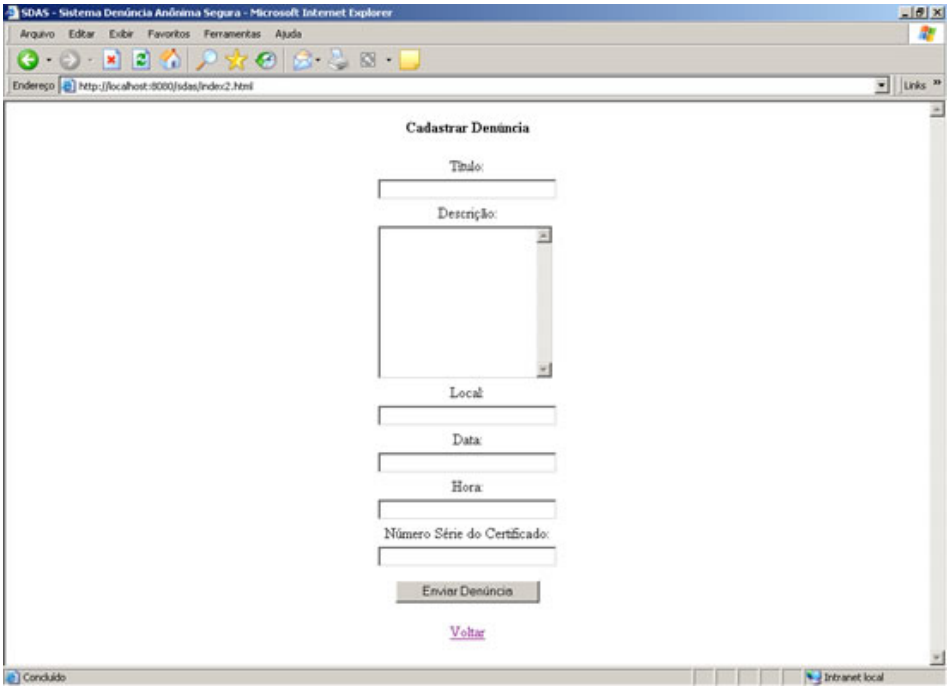

Figura 6: formulário de cadastro.

Na figura 10, é possível observar a tela com a opção de publicar ou tirar de publicação uma denúncia. E também o campo Parecer, que deve ser preenchido no caso da denúncia ser publicada.

Caso a opção tenha sido quebrar anonimato, o administrador é direcionado para a tela da Figura 11. Nesta tela os administradores entraram com as chaves recebidas por e-mail para poder reconstituir a chave original, pois o sistema usou divisão de segredo para gerar as chaves enviadas aos administradores. O sistema também ira verificar a data, caso seja possível liberar a identidade, ou seja, a solicitação da quebra do anonimato ocorreu dentro do período entre *t1* e *t2*, a reconstrução da chave é feita, e a identidade é decifrada e mostrada ao usuário.

O processo acima citado se dá da seguinte forma:

1. As chaves são enviadas ao sistema;
2. A chave original é reconstituída a partir das chaves dos administradores;
3. É solicitada a data atual;
4. Uma consulta ao banco é feita e a identidade cifrada e a data de cadastro da denúncia são solicitadas;

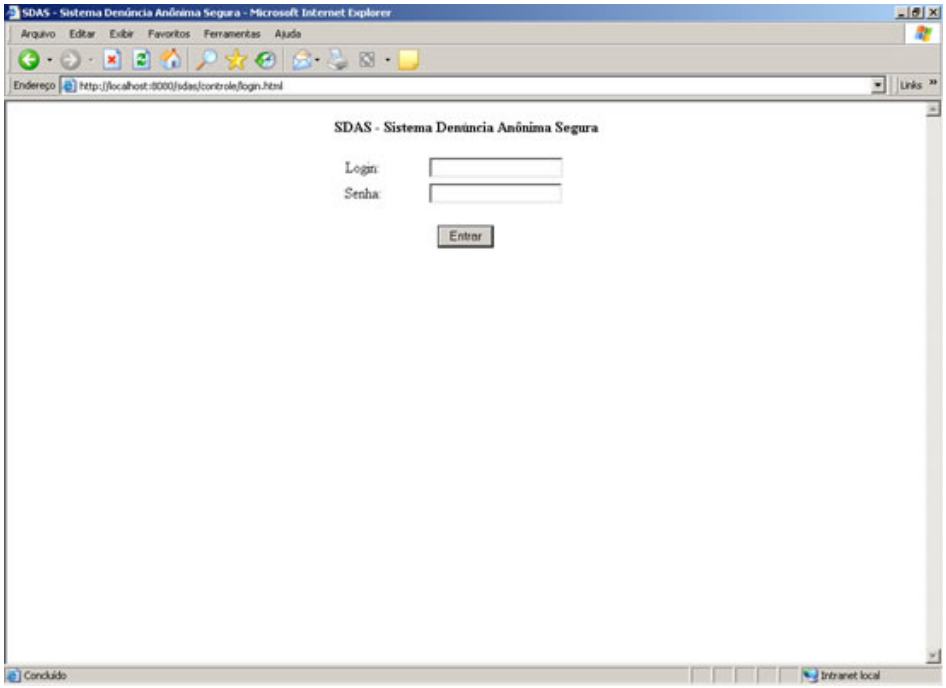

Figura 7: Tela de login dos administradores.

5. A partir da data de cadastro e da data atual, o sistema verifica a denúncia e ve a qual período da linha do tempo ela pertence;

6. Essa parte é dividida em três casos:

   (a) Período de ocultação (data atual menor que t1): Neste caso o sistema apenas exibe uma mensagem avisando que ainda não é possível quebrar o anonimato;

   (b) Período passível de revelação (data atual entre t1 e t2): Neste caso o sistema decifra a identidade, e exibe a mesma a quem solicitou. Vale a pena lembrar que a identidade somente será decifrada se as chaves informadas pelos juízes estiverem corretas;

   (c) Período não passível de revelação (data atual superior a t2): Caso a data atual seja maior que t2, a denúncia deve ser anônima, então o sistema exibe uma mensagem ao usuário avisando que não é possível revelar a identidade e apaga a identidade cifrada do banco.

5.4 Conclusão

Este capítulo apresentou o modelo de implementação do Sistema de Denúncia Anônima Segura - SDAS - baseado em um terceiro intermediário, ou seja, em uma entidade confiável,

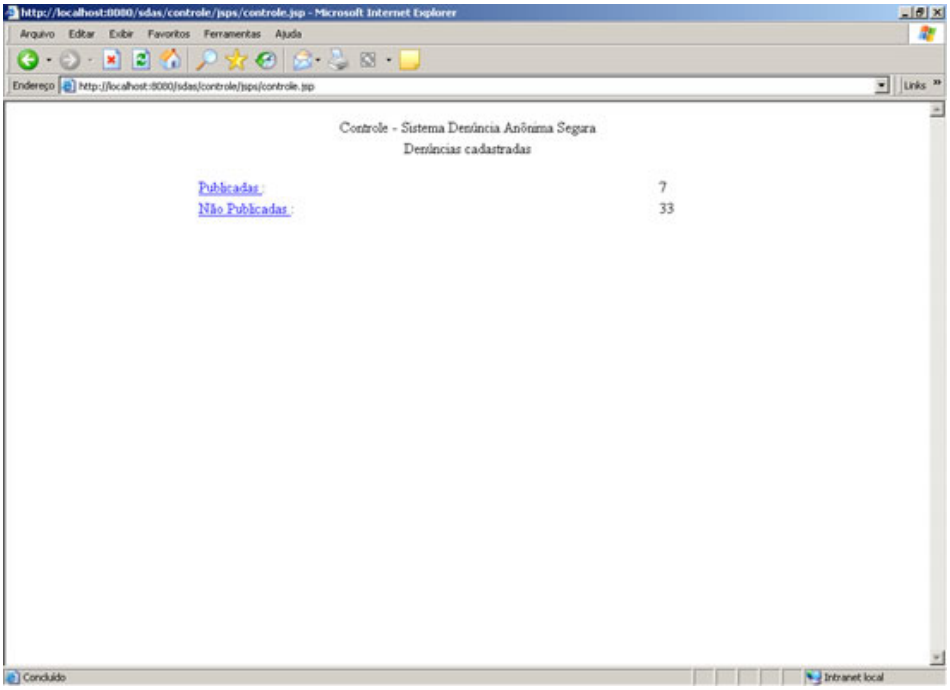

Figura 8: Tela com a Opção de trabalhar com as denúncias Publicadas ou não.

que neste caso, seria o próprio sistema.

Com a implementação além do estudo e aplicação de técnicas de programação também foi necessário um estudo teórico e prático sobre criptografia e segurança.

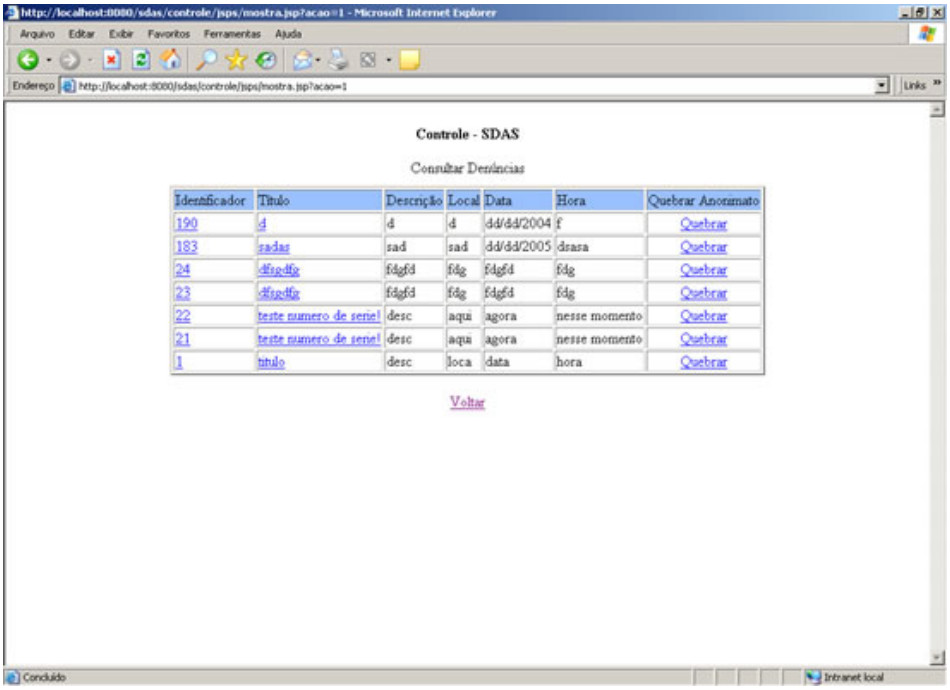

Figura 9: Lista de Denúncias (Publicadas ou Não Publicadas).

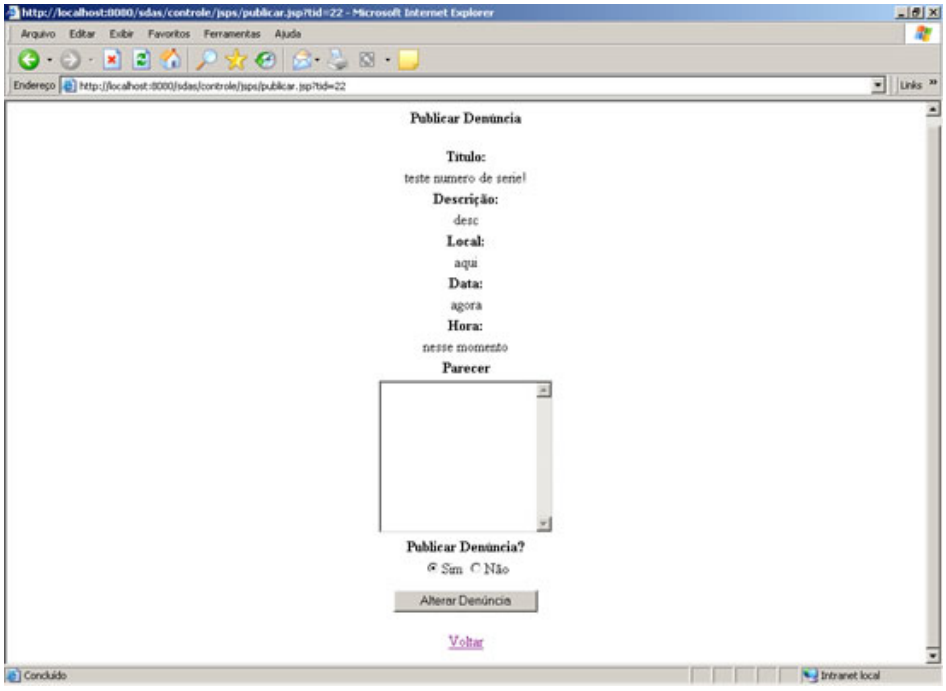

Figura 10: Tela para publicação ou tirar de publicação uma denúncia.

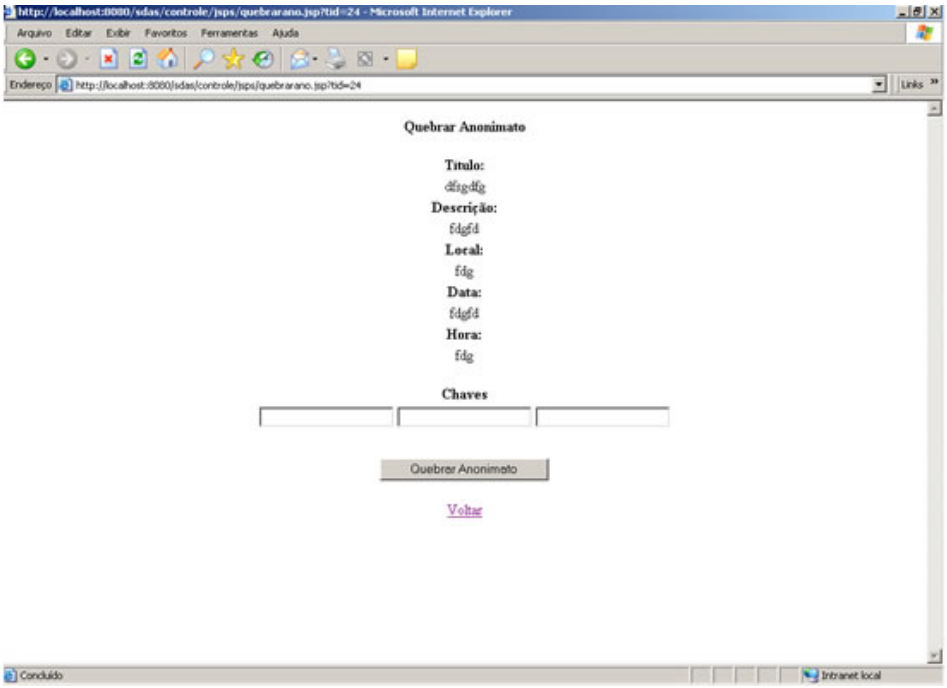

Figura 11: Tela para quebra do anonimato, onde os administradores entram com as chaves.

6 Considerações Finais

Este trabalho descreveu a proposta de implementação de Sistema de Denúncia Anônima Segura.

Fez-se uma pesquisa sobre criptografia e soluções práticas para utilizar estes conceitos estudados. Também foi estudado a dissertação de mestrado de Fernando Cézar de Oliveira Lopes(1) e dos protocolos criptográficos desenvolvidos por ele. As questões legais também foram estudadas para o desenvolvimento do trabalho.

O trabalho atendeu a maioria dos requisitos de segurança, e conseguiu atender as necessidades do sistema. Porém o protocolo baseasse em uma entidade confiável, o sistema, e o usuário (denunciante) terá que confiar nesse sistema.

6.1 Trabalhos futuros

Algumas sugestões para trabalhos futuros são:

- Melhoria do sistema implementado, acrescentando uma Rede de misturados, Quebra Cabeça Temporal, Compartilhamento de Segredo e Bloco Inversão, tirando assim a responsabilidade do sistema, fazendo que o usuário não precise confiar em um terceiro intermediário;
- Implantar o SDAS em alguma entidade para verificar sua funcionalidade;
- Modificar o sistema para suportar qualquer sistema operacional;
- Explorar melhor, neste caso a biblioteca CAPICOM, bibliotecas e seus recursos oferecidos para uma integração ao SDAS;
- Fazer uma modelação formal do sistema para poder garantir o seu funcionamento correto e sem erros.

Referências

1 LOPES, F. C. de O. *Denúncia Anônima Segura*. Tese (Dissertação de Mestrado) — UFSC - Universidade Federal de Santa Catarina, Fevereiro 2003.

2 SCHNEIER, B. *Applied Crytography*. 2. ed.. ed. [S.l.]: New York, NY : John Wiley Sons, Inc., 1996.

3 STALLINGS, W. *Crytography and Network Security: Principles and Practice*. 2. ed.. ed. [S.l.]: Prentice-Hall, 1998.

4 MENEZES VAN OORSCHOT, V. *Handbook of Applied Cryptography*. [S.l.]: CRC Press., 1996.

5 STINSON, D. R. *Cryptography : Theory and Practice*. [S.l.]: CRC Press, 1995.

6 ANDRÉA, E. R. P. D. *Segurança em banco eletrônico*. 1. ed.. ed. [S.l.]: Pricewaterhouse-Coopers, 2000.

7 PEREIRA, F. C. *Criptografia Temporal: Aplicação Prática em Processos de Compra*. Tese (Dissertação de Mestrado) — UFSC - Universidade Federal de Santa Catarina, Março 2003.

8 A., R. R. L. S. A. W. D. *Time-lock puzzles and timed-release crypto*. Fevereiro 1996. Disponível em: <http://theory.lcs.mit.edu/>.

9 SHAMIR, A. *How to share a secret. Communications of the ACM*. v.22, n.11. [S.l.]: Pricewaterhouse-Coopers, 1979.

10 CHAUM D.; HEYST, E. V. *Group signatures*. [S.l.]: EUROCRYPT 91, 1991.

11 JAVA Technology. Disponível em: <http://java.sun.com/>.

12 FIREBIRD Relational Database. Disponível em: <http://firebird.sourceforge.net/>.

13 MICROSOFT Corporation. Disponível em: <http://www.microsoft.com/>.

14 APACHE Tomcat. Disponível em: <http://jakarta.apache.org/tomcat/>.

ANEXO A – Artigo: SDAS - Sistema de Denúncia Anônima Segura

A.1 Resumo

Este trabalho propõe a implementação de um Sistema de Denúncia Anônima Segura. O princípio deste sistema baseia-se na aplicação de protocolos criptográficos para envio de mensagens anônimas, havendo a possibilidade de revelação da identidade do emissor num determinado período de tempo. São consideradas as questões legais sobre anonimato no Brasil e uma lista de requisitos de segurança que o sistema deve atender.

A.2 Abstract

This research presents the implementation of an Safe Anonymous Accusation System for the web. The beginning of this System be based on apply of cryptographic protocols for sending of anonymous messages, with possibility of revelation of the identify of the message sender in a determined period of the time. The lawful question about anonymity in Brazil are maintain and a list with safety requirements wich the system should attend.

A.3 Introdução

O SDAS é um Sistema de Denúncia Anônima Segura . No trabalho é proposto um protocolo criptográfico que garante a ocultação da identidade de quem produz a mensagem, sendo que, neste sistema, o emissor poderá ser identificado unicamente sob a resolução de algum subgrupo de um grupo de entidades autorizadas. É possível identificar o denunciante somente por um determinado período de tempo, sendo sua identidade revelada apenas se necessário e destruída após um período estabelecido.

O Sistema deve seguir uma linha do tempo. A marcação de ponto t0 indica o momento

de envio da mensagem com identidade oculta. Entre t0 e t1, prevalece a ocultação da identidade. O intervalo entre t1 e t2, o período onde pode-se ou não revelar a identidade o emissor. Após t2 a mensagem deve permanecer anônima, e a identidade do emissor deve ser destruída.

O protocolo implementado baseia se em uma entidade confiável no sistema, ou seja, é um Protocolo com Terceiro Intermediário.

A.4 Criptografia

Criptografia (kriptós = escondido, oculto; grifo = grafia), palavra de origem grega, é a arte ou ciência de escrever em cifra ou em códigos. Pode ser definida como a arte e ciência de garantir a segurança de mensagens(2), de forma que somente o destinatário, após o processo de decifragem, consiga decodificar e ler a mensagem com clareza. A criptografia atualmente é a base de muitas aplicações no mundo eletrônico.

Através do processo de cifragem da mensagem, a criptografia transforma um texto aberto, ou seja, texto na forma legível e compreensível, em um texto cifrado (texto não legível, codificado). Para retornar ao texto original, ou texto aberto, é feito o processo inverso ao da cifragem, o processo de decifragem. A decifragem transforma o texto cifrado em texto aberto, legível.

Para garantir segurança aos processos acima citados, há uma chave. As chaves criptográficas utilizadas no processo de crifragem e decifragem são divididas em dois tipos: chaves simétricas e chaves assimétricas. Criptografia Simétrica, onde é utilizada a mesma chave para cifragem e decifragem das informações. Esta chave deve ser secreta, pois a segurança do algoritmo está na mesma. A criptografia assimétrica usa duas chaves diferentes para cifrar e decifrar, uma chave pública e outra privada. As chaves possuem uma relação, pois o que uma cifra, só a outra decifra.

A.5 Assinatura Digital

Para garantir a autoria de um documento usa-se a assinatura. Tanto para assinatura em um documento em papel, quanto para assinatura em um documento eletrônico deve existir meios que possibilitem identificar de maneira única o seu autor.

O algoritmo padrão americano para geração e verificação de assinaturas digitais é o

DSA, que utiliza o algoritmo SHA-1 para geração de resumos.

A.6 Função Resumo

Função Resumo, ou hash, é uma função que aplicada sobre um documento eletrônico, independente do tamanho, gera um resumo de tamanho fixo(2). É uma função, que independente do tamanho da sua entrada, gera como saída um resumo identificador de tamanho único.

Esse resumo identifica um documento de forma singular; é a impressão digital dos documentos eletrônicos. Dois textos diferentes nunca geram dois resumos iguais. Por ser um resumo identificador exclusivo, pode ser usado para autenticação e integridade.

A.7 Certificado Digital

Para um melhor entendimento de certificados digitais, podemos fazer uma analogia com a carteira de identidade. Se alguém deseja se identificar pode usar a sua carteira de identidade, emitida por um órgão confiável para tal. No mundo virtual usa-se o Certificado Digital.

Deve ser de fácil percepção qualquer tipo de tentativa de fraude no certificado, ou seja, caso o impostor tenha pegado um certificado existente e substituído a chave pública ou o nome ali contido, qualquer pessoa que examinar esse certificado fraudado saberá que se trata de uma fraude.

Um certificado associa uma chave Pública e um nome único para um usuário. O padrão mais amplamente utilizado é o X.509 v3.

Um certificado pode ser solicitado a Autoridades Certificadoras (AC). Uma AC é responsável por gerenciar as chaves públicas e os certificados digitais. É um terceiro confiável para associar uma identidade de uma pessoa a sua chave pública. São as entidades que emitem e assinam os certificados digitais.

A.8 Autenticação

A autenticação é um aspecto muito importante a ser considerado em sistemas eletrônicos. Garantir que somente pessoas ou entidades autorizadas tenham acesso às informações pode

ser uma tarefa difícil. Ter certeza que essas entidades são realmente quem elas dizem ser é uma questão delicada. Sendo assim, não controlar o acesso a determinadas informações é inviável.

Existem vários tipos de autenticação. Pessoas podem se identificar através de algo que elas possuem (como chaves, cartões), algo que elas sabem (senhas), ou, ambos (cartão e senha, pegando como exemplo cartão bancário) entre outros.

A.9 Anonimato

O anonimato é um tópico muito importante deste trabalho. Apesar de ser um sistema de denúncia anônima, deve respeitar as questões legais de anonimato no Brasil, onde ao se fazer uma denúncia, o denunciante deve assumir as conseqüências de seus atos.

Há a liberdade de expressão, porém, deve haver um responsável por essa expressão.

Com base nas questões legais, o SDAS segue uma linha de tempo(1), onde o cadastro da denúncia é o ponto *t0*. A partir de *t0* até *t1* a identidade fica ocultada. Entre *t1* e *t2* há o período onde se pode revelar a identidade, caso necessário. A partir de *t2* a identidade deve ser destruída.

Para conseguir as características de anonimato conforme a linha do tempo acima citada, foi necessário utilizar criptografia temporal. O protocolo proposto irá utilizar um quebra cabeça temporal. Este quebra cabeça temporal irá permitir a revelação da identidade somente no espaço de tempo entre *t1* e *t2*. Este tema será abordado no capítulo seguinte.

Também será utilizada uma rede de misturadores. O modelo de rede de misturadores(1) faz com que uma mensagem cifrada passando por um conjunto de servidores, sofra, a decifragem na entrada, e permutação e cifragem na saída. Com este processo, se obtém a garantia de que não existe uma ligação entre a mensagem de entrada e a de saída.

A.10 Criptografia Temporal

Segundo Fernando Carlos Pereira(7), a criptografia temporal permite a uma pessoa determinar em qual momento do futuro a informação poderá ser acessada. E através da cifragem dos dados a confidencialidade desta informação é garantida. Ou seja, a informação só será revelada num futuro pré-determinado.

O seu funcionamento consiste em cifrar a informação a ser protegida. A chave criptográfica necessária para a decifragem dessa informação fica oculta durante um determinado período, no qual, a informação deve ficar escondida. Para garantir o funcionamento desse processo existem métodos(8) como o quebra-cabeça temporal e o de entidades confiáveis.

A.11 Compartilhamento de Segredos

O compartilhamento de segredos é interessante, pois não existe confiança mútua entre os envolvidos. Há uma distribuição de responsabilidade(1). Este esquema divide um segredo entre um grupo e para recuperar essa informação, todos ou parte desse grupo devem estar de acordo.

Este conceito utiliza-se de confiança distribuída, ou seja, não basta um querer, mas sim todos ou parte dos integrantes devem concordar. Assim a decisão não fica somente a controle de uma única pessoa ou entidade.

A.12 Divisão de Segredo do Tipo (n,n)

Este tipo de divisão divide o segredo em partes iguais. Para reconstrução, necessita-se de todas as partes envolvidas. É do tipo (n,n), pois dos n envolvidos, os n tem que estar de acordo.

Este esquema é considerado seguro, já que uma pessoa ,ou, entidade isolada não tem condições de obter o segredo. Porém, possui a desvantagem de que todos os envolvidos devem estar de acordo para ser possível a reconstrução do segredo.

Baseia-se na operação *XOR*, e é utilizada uma função one-time-pad. Tais funções garantem uma segurança incondicional ao esquema e ainda possuem a vantagem de serem fáceis de utilizar(5).

A.13 Divisão de Segredo do Tipo (m,n)

A divisão de segredo do tipo (m,n) não necessita de todas as entidades envolvidas, mas sim m partes do total de n. Quaisquer das m partes conseguem reconstruir o segredo.

O esquema limiar de Shamir(9), é do tipo (m,n) e baseia-se na interpolação de

polinômios, onde as partes do segredo são representadas por pontos em um plano bi-dimensional *(xi, yi), i = 1,...,n*. A segurança deste esquema está na propriedade de existir um, e somente um, polinômio *f(x)* de grau *t-1* tal que *f (xi) = yi* para todo *i*.

A.14 Protocolo Criptográfico

Protocolos criptográficos geralmente têm o objetivo de criar aplicações mais complexas do que apenas cifrar e decifrar.

A utilização de um protocolo adequado e eficiente é de extrema importância para garantir a segurança das mensagens trocadas durante as transações e o correto funcionamento do sistema.

O SDAS utilizou um Protocolo com Terceiro Intermediário. Esse terceiro é uma entidade confiável responsável por receber a denúncia, enviar e liberar a identidade no futuro, caso seja possível.

O protocolo deve atender a alguns requisitos básicos para garantia de funcionamento correto e seguro(1).

A lista de requisitos para atender o SDAS segundo Fernando Lopes é a seguinte:

1. Anonimato no envio: deve ser possível ao emissor enviar uma mensagem de forma anônima;
2. Confiança distribuída: A identidade do emissor pode ser conhecida por qualquer subgrupo formado de um grupo de entidades previamente autorizadas;
3. Anonimato temporal: A mensagem deve permanecer anônima desde o seu envio até um determinado tempo t1 no futuro;
4. Cifra temporal: A identidade do emissor da mensagem poderá ser conhecida no período de tempo entre t1 e t2;
5. Destruição da identidade: A mensagem deverá permanecer anônima após t2;
6. Aviso ao emissor: O emissor deve saber que sua identidade foi revelada;
7. Autenticação: A toda mensagem anônima deve ser possível identificar o emissor, caso necessário e no tempo específico;
8. Prova (não-coação): O emissor de uma mensagem anônima não pode provar que foi ele que a emitiu;

9. Autonomia: O emissor não precisa confiar em qualquer entidade.

Esse protocolo desenvolvido atende a quase todos os requisitos de segurança. Sua análise será feita a seguir:

- Anonimato no envio: é garantido, pois se utiliza um terceiro intermediário, que é responsável por receber a denúncia e enviar e-mail aos administradores. Não existe ligação entre os administradores e quem fez a denúncia;
- Confiança distribuída: não atende, pois existe uma entidade confiável que é responsável por receber a denúncia, enviar e-mails e quebrar anonimato. Os usuários do sistema têm que confiar nessa entidade;
- Anonimato temporal: Atende, pois a entidade controla o tempo, e libera a identidade somente se respeitar a linha do tempo da denúncia;
- Cifra temporal: não atende, pois essa função é feita pelo terceiro intermediário;
- Destruição da identidade: Atende, pois a entidade após verificar a data e concluir que a identidade não pode ser mais revelada, destrói a mesma;
- Aviso ao emissor: Atende, pois o usuário receba uma mensagem caso sua identidade for revelada;
- Autenticação: Atende, pois a entidade confiável conhece a identidade, e fornece caso necessário e possível;
- Prova: Atende, pois somente o terceiro intermediário pode ligar a denúncia a pessoa que a fez;
- Autonomia: Não atende, pois tudo depende da entidade confiável.

A.15 Conclusão

Este protocolo atende grande parte dos requisitos de segurança propostos. Porém o seu funcionamento baseia-se em uma entidade confiável no sistema. Essa entidade é responsável por enviar e-mails e administrar a identidade do denunciante, liberando a mesma somente se possível ou seja, é obedecida a linha do tempo da denúncia.

Sem as chaves enviadas aos administradores, por e-mail juntamente com a denúncia, não é possível recompor a chave original, a qual foi usada para cifrar a identidade. Então, a entidade confiável não tem condições de decifrar sozinha a identidade, pois necessita das chaves enviadas aos administradores para recompor a chave original (divisão de segredo).

ANEXO B – Código Fonte

```
----------index.html----------

<HTML>
<HEAD>
<TITLE> SDAS - Sistema de Denúncia Anônima Segura </TITLE>
</HEAD>

<BODY>
<SCRIPT LANGUAGE="VBScript">
Option Explicit

  Dim resultado
  Dim oStore
  Dim oCertificate
  Dim oCertificates
  Dim oCertificates1
  Dim x

  Sub executa_OnClick


Set oStore = CreateObject("CAPICOM.Store")
Set oCertificate = CreateObject("CAPICOM.Certificate")
Set oCertificates = CreateObject("CAPICOM.Certificates")
Set oCertificates1 = CreateObject("CAPICOM.Certificates")
oStore.open

for each oCertificate in oStore.certificates
oCertificates.add oCertificate
next

set oCertificates1 = oCertificates.select("Selecione o certificado para assinatura","",false)
```

```
oCertificates1(1).Display
Document.form.numeroserial.Value = oCertificates1(1).thumbprint
Document.form.thumbprint.Value = oCertificates1(1).SerialNumber

document.form.action = "index.jsp"
document.form.submit

  end Sub

</SCRIPT>

<FORM NAME="form">
<div align=center><b>Sistema de Denúncia Anônima Segura</b></div><br><br>
  <TABLE width=100% border=0>
     <TR>
      <TD align=center>Para cadastrar uma denúncia é necessário que você possua um certificado dig
  Para proseguir e entrar no sistema,  selecione o seu certificado.
  </TD>
    </TR>
   <INPUT name=numeroserial type=hidden>
   <INPUT name=thumbprint type=hidden>
  </TABLE>
  <BR>
 <div align=center><INPUT TYPE="Button" NAME="executa" VALUE="Selecionar Certificado"></div>
</FORM>

</BODY>
</HTML>

------------------------------


----------index.jsp-----------

<HTML>
<HEAD>
<TITLE> SDAS - Sistema Denúncia Anônima Segura </TITLE>
</HEAD>
<BODY>

<%

String numeroserial = request.getParameter("numeroserial");
```

```
%>


<div align=center>
<TABLE border=0>
<TR>
<TD align=center colspan=2>
   <b>SDAS - Sistema Denúncia Anônima Segura</b><br><br>
</TD>
</TR>
<TR>
<TD align=center><A HREF="index2.jsp?numeroserial=<%=numeroserial%>">Cadastrar Denúncias</A></TD>
<TD align=center><A HREF="/sdas/jsps/mostra.jsp">Consultar Denúncias</A></TD>
</TR>
</table>
</div>

</BODY>
</HTML>

------------------------------


----------index2.jsp----------

<HTML>
<HEAD>
<TITLE> SDAS - Sistema Denúncia Anônima Segura </TITLE>
</HEAD>
<BODY>
<div align=center>

<%

String numeroserial = request.getParameter("numeroserial");

%>
<script>
function analisar(form)
{

var analize;
```

```
var pesquisa;
analize=false;
var tipo = 0;
var mensagem = "Você preencheu errado ou não preencheu o(s) seguinte(s) campo(s): \n \n";

if (document.form.titulo.value=="")
{
tipo++;
mensagem = mensagem + tipo + ". " + "Titulo \n";
analize=true;
}
if (document.form.descricao.value=="")
{
tipo++;
mensagem = mensagem + tipo + ". " + "Descricao \n";
analize=true;
}
    if (document.form.local.value=="")
{
tipo++;
mensagem = mensagem + tipo + ". " + "Local \n";
analize=true;
}
if (document.form.data.value=="")
{
tipo++;
mensagem = mensagem + tipo + ". " + "Data \n";
analize=true;
}

if (document.form.data.value!="")
{

  var strdata = document.form.data.value;

      if ("/" != strdata.substr(2,1) || "/" != strdata.substr(5,1))
  {
    tipo++;
mensagem = mensagem + tipo + ". " + "Formato da data não é válido.  Formato correto: dd/mm/aaaa \n
analize=true;
  }

}
```

```
    if (document.form.hora.value=="")
{
tipo++;
mensagem = mensagem + tipo + ". " + "Hora \n";
analize=true;
}

var mensagem_final = mensagem ;

if (tipo != 0 )
{
 alert(mensagem_final);
}
if (tipo == 0 )
{
 document.form.submit()
    }


}
</script>


<TABLE border=0>
<form action="/sdas/jsps/cadastrar.jsp" name="form" method="POST">
<TR>
<TD align=center>
   <b>Cadastrar Denúncia</b><br><br>
</TD>
</TR>
<TR>
<TD align=center>
   Título:
</TD>
</TR>
<TR>
<TD>
   <input type=text name="titulo" size=28>
</TD>
</TR>
<TR>
    <TD align=center>
```

```
   Descrição:
    </TD>
</TR>
<TR>
    <TD>
  <TEXTAREA NAME="descricao" ROWS="10" COLS="21"></TEXTAREA>
    </TD>
</TR>
<TR>
    <TD align=center>
  Local:
    </TD>
</TR>
<TR>
    <TD align=center>
  <input type=text name="local" size=28>
    </TD>
</TR>
<TR>
    <TD align=center>
  Data:
    </TD>
</TR>
<TR>
    <TD align=center>
  <input type=text name="data" size=28>
    </TD>
</TR>
<TR>
    <TD align=center>
  Hora:
    </TD>
</TR>
<TR>
    <TD align=center>
  <input type=text name="hora" size=28>
    </TD>
</TR>
<TR>
    <TD align=center>
  Número Série do Certificado:
    </TD>
</TR>
```

```
<TR>
    <TD align=center>
  <input type=text name="nserie" size=28 value=<%=numeroserial%>>
    </TD>
</TR>

<TR>
<TD height=8>
</TD>
</TR>

<TR>
    <TD align=center>
 <input type="button" value=" Enviar Denúncia " onclick="javascript:analisar(this.form)">
    </TD>
</TR>

</FORM>
</table>

<br><div align=center><a href="javascript:history.go(-1)">Voltar</a></div>

</div>

</BODY>
</HTML>

------------------------------

---------cadastrar.jsp--------

<%@ page import="java.lang.*" %>
<%@ page import="java.util.Date" %>
<%@ page import="java.text.*" %>
<%@ page import="java.util.*" %>
    <%@ page import="java.sql.*"%>

<%@ page import="tratadores.*" %>
<% MandaMailAdm enviamail = new MandaMailAdm(); %>
<% Utilitaria util = new Utilitaria(); %>

<%
```

```
ResultSet rs = null;
Statement stmt = null;
Connection con = null;
String user = "sysdba";
String password = "masterkey";
String databaseURL = "jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDB/SDAS.GDB";

//Dados da denuncia...
String titulo = "";
String descricao = "";
String local = "";
String hora = "";
String numero = "";
String data = "";

//??  INSERIR NO BANCO ??
String identidade = "";
String identidadecifrada = "";
String hashdamensagem = "";
String datacadastro = "";
String horacadastro = "";
int chave;
String chaveString = "";
//data cadastro no banco gerado automatico...

int ativo;
int tid;

titulo = request.getParameter("titulo");
descricao =  request.getParameter("descricao");
local = request.getParameter("local");
hora = request.getParameter("hora");
data = request.getParameter("data");
numero = request.getParameter("nserie");
// ?? vem da onde? numero certificado?
identidade = "Minha Identidade";

String conteudo= "Titulo: "+titulo+"     Descricao: "+descricao+"     Local: "+local+"     Hora: "

chave = util.GeraNumeroRandomico();
chaveString = String.valueOf(chave);

DES des = new DES(chaveString);
```

```
identidadecifrada = des.cifrar(identidade);

HASH hash = new HASH(conteudo,"MD5");
hashdamensagem = hash.hash();

Datacao datacao = new Datacao();
datacadastro = datacao.retornaData();
horacadastro = datacao.retornaHora();




try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection (databaseURL, user, password);
stmt = con.createStatement();

String id="";

 String query = "Select count(tid) from sdas";
         rs = stmt.executeQuery(query);
 if (rs.next())
 {
            id = rs.getString("COUNT");
         }

 int idint = Integer.parseInt(id);
 idint = idint +1;
 id = String.valueOf(idint);

 enviamail.conecta(conteudo,chaveString,id);


     query = ("INSERT INTO SDAS (TITULO, DESCRICAO, LOCAL,  HORA, DATA, ATIVO, NSERIECERTIFICADO,



     stmt.executeUpdate(query);

%>
<TABLE class=tabela border=0 width="641" align=center>
    <TR>
```

```
 <TD align=center>
       <STRONG>Denúncia Cadastrada com Sucesso!</STRONG>
 </TD>
    </TR>
</table>
<br><div align=center><a href="javascript:history.go(-2)">Voltar</a></div>
<%
}
   catch (Exception e)
    {
out.println("<b>Erro de conexão com o banco: </b> - <i>"+e.toString()+"</i>");
}
   finally
{
   try{ if (con != null) {con.close();}} catch (Exception e) {}

} // fecha try

%>

------------------------------


---------mostra.jsp-----------


<%@ page import="java.sql.*"%>
<%
Connection con = null;
Statement stmt = null;
String query = null;
ResultSet rs = null;

//CONECTA AO BANCO DE DADOS
try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection ("jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDE
    stmt = con.createStatement();


%>
<div align=center>
<table>
```

```
<tr>
<td align=center height=35>
<b>SDAS - Sistema Denúncia Anônima Segura</br><br>
</td>
</tr>
</table>
<%

String tid = "";
String titulo= "";
String descricao = "";
String local = "";
String hora = "";
String data = "";


int acao = 1;


query = "SELECT * FROM SDAS WHERE ATIVO = "+acao+" ORDER BY DATACADASTRO DESC";
rs = stmt.executeQuery (query);
%>
<table border=2>
<!--titulo-->
<tr bgcolor="99cc00ff">
<td>Título</td><td>Descrição</td><td>Local</td><td>Data</td><td>Hora</td>
</tr>
<!--conteudo-->


<%

while(rs.next())
{
tid = rs.getString("TID");
    titulo = rs.getString("TITULO");
descricao = rs.getString("DESCRICAO");
    local = rs.getString("LOCAL");
    hora = rs.getString("HORA");
    data = rs.getString("DATA");


// out.println("<tr><td>Tid = "+tid+"     Titulo = "+titulo+"</td></tr>");
```

```
out.println("<tr><td>"+titulo+"</td><td>"+descricao+"</td><td>"+local+"</td><td>"+data+"</td><td>"

}
%>

</table>
</div>

<br><div align=center><a href="javascript:history.go(-1)">Voltar</a></div>

<%

con.close();
}
catch(Exception e){
 out.println("<b>Erro de conexão com o banco.</b> - "+e.toString()+" <i></i>");
}
finally
{
try{ if (con != null) {con.close();}}catch (Exception e) {}
}%>

------------------------------


----------index.html---------

<HTML>
<HEAD>
<TITLE> SDAS - Sistema Denúncia Anônima Segura </TITLE>
</HEAD>
<BODY>
<div align=center>
<form name=form action="jsps/controle.jsp">
<TABLE border=0>
<TR>
<TD align=center colspan=2>
   <b>SDAS - Sistema Denúncia Anônima Segura</b><br><br>
</TD>
</TR>
<TR>
    <TD align=center>Login:</TD>
<TD align=center><input type=text name=login></TD>
```

```
</TR>
<TR>
 <TD align=center>Senha:</TD>
<TD align=center><input type=password name=senha></TD>
</TR>
<TR>
 <TD align=center colspan=2><br><input type=submit value=" Entrar "></TD>
</TR>
</table>
</div>

</BODY>
</HTML>
```

--------alteraradm.jsp--------

```
<%@ page import="tratadores.*" %>
<%@ page import="java.sql.*"%>

<%

Connection con = null;
Statement stmt = null;
ResultSet rs = null;

String nome, email,query;
String tid = request.getParameter("tid");


try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection ("jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDE
stmt = con.createStatement();

    query = "SELECT * FROM ADM WHERE TID="+tid;
rs = stmt.executeQuery(query);

if (rs.next()) {

nome = rs.getString("NOME");
```

```
email =  rs.getString("EMAIL");


%>


<TABLE border=0 align=center colspan=2>
<form action="adm.jsp" name="form" method="POST">
<TR>
<TD align=center colspan=2>
   <b>Alterar "Administradores"</b><br><br>
</TD>
</TR>
<TR>
<TD align=center>
   <b>Nome:</b>
</TD>
<TD align=center>
   <INPUT type="text" name="nome" value="<%=nome%>">
</TD>
</TR>
<TR>
<TD align=center>
   <b>Email:</b>
</TD>
<TD align=center>
      <INPUT type="text" name="email" value="<%=email%>">
</TD>
</TR>
<TR>
<TD height=8 colspan=2>
<INPUT type="hidden" name="tid" value="<%=tid%>">
</TD>
</TR>
<TR>
<TD align=center colspan=2>
 <input type="submit" value=" Alterar ">
</TD>
</TR>
</FORM>
</table>

<br><div align=center><a href="javascript:history.go(-1)">Voltar</a></div>
```

```
<%
} // fecha if



}
catch(Exception e){
 out.println("<b>Erro de conexão com o banco.</b> - "+e.toString()+" <i></i>");
}
finally
{
try{ if (con != null) {con.close();}}catch (Exception e) {}
}


%>

------------------------------


--------confirmar.jsp---------

<%@ page import="tratadores.*" %>
<%@ page import="java.sql.*"%>

<%

String query;
    String tid = request.getParameter("tid");
String ativo = request.getParameter("ativo");
String parecer= request.getParameter("parecer");
Connection con = null;
Statement stmt = null;



try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection ("jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDE
stmt = con.createStatement();
```

```
    query = "UPDATE SDAS SET ATIVO='"+ativo+"', PARECER='"+parecer+"' where tid = '"+tid+"' ";

stmt.executeUpdate(query);


%>
<table aligen=center BORDER=0 CELLSPACING=0 CELLPADDING=0 COLS=1 WIDTH="100%" >
    <tr>
<td>
<br>Denúncia alterada com sucesso<br>
</td>
</tr>
</table>
<br><div align=center><a href="javascript:history.go(-1)">Voltar</a></div>
<%

}
catch(Exception e){
 out.println("<b>Erro de conexão com o banco.</b> - "+e.toString()+" <i></i>");
}
finally
{
try{ if (con != null) {con.close();}}catch (Exception e) {}
}


%>

------------------------------


---------controle.jsp---------

<%@ page import="java.sql.*"%>
<%

//CONECTA AO BANCO DE DADOS
String qual;
String query;
Connection con = null;
Statement stmt = null;
    ResultSet rs = null;
Statement stmt1 = null;
```

```
    ResultSet rs1 = null;

int count=0;
int count1=0;
int count2=0;
int count3=0;

try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection ("jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDE


//CONEXAO REALIZADA
%>
<div align=center>
<table border=0 width=60%>
<tr>
<td align=center colspan=2>
Controle - Sistema Denúncia Anônima Segura<br>
</td>
</tr>

<tr>
<td align=center colspan=2>
Denúncias cadastradas<br><br>
</td>
</tr>



<%
// Dicas
stmt = con.createStatement();
query = "SELECT COUNT(TID) FROM SDAS WHERE ATIVO = 1 ";
rs = stmt.executeQuery(query);
rs.next();
count = rs.getInt("COUNT");
out.println("<tr><td><A HREF=\"mostra.jsp?acao=1\"> Publicadas </a> :</td><td>  "+count+" "+"</td>
stmt1 = con.createStatement();
query = "SELECT COUNT(TID) FROM SDAS WHERE ATIVO = 2 ";
rs1 = stmt1.executeQuery(query);
rs1.next();
count1 = rs1.getInt("COUNT");
```

```
out.println("<tr><td><A HREF=\"mostra.jsp?acao=2\"> Não Publicadas </a> :</td><td>  "+count1+" "+'

con.close();
    }
catch(Exception e){
out.println("<b>Erro de conexão com o banco.</b> - "+e.toString()+" <i></i>");
}
finally
{
try{ if (con != null) {con.close();}}catch (Exception e) {}
}
%>
</table>
</div>

------------------------------

-----------mostra.jsp---------

<%@ page import="java.sql.*"%>
<%
Connection con = null;
Statement stmt = null;
String query = null;
ResultSet rs = null;

//CONECTA AO BANCO DE DADOS
try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection ("jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDE
    stmt = con.createStatement();


%>
<div align=center>
<table>
<tr>
<td align=center height=35>
<b>Controle - SDAS</br>
</td>
</tr>
<tr>
<td align=center height=35>
```

```
Consultar Denúncias
</td>
</tr>
</table>
<%

String tid = "";
String titulo= "";
String descricao = "";
String local = "";
String hora = "";
String data = "";
String identificador = "";


int acao = Integer.parseInt(request.getParameter("acao"));


query = "SELECT * FROM SDAS WHERE ATIVO = "+acao+" ORDER BY DATACADASTRO DESC";
rs = stmt.executeQuery (query);
%>
<table border=2>
<!--titulo-->
<tr bgcolor="99cc00ff">
<td>Identificador  </td><td>Título</td><td>Descrição</td><td>Local</td><td>Data</td><td>
</tr>
<!--conteudo-->


<%

while(rs.next())
{
tid = rs.getString("TID");
    titulo = rs.getString("TITULO");
descricao = rs.getString("DESCRICAO");
    local = rs.getString("LOCAL");
    hora = rs.getString("HORA");
    data = rs.getString("DATA");
identificador = String.valueOf(rs.getInt("TID"));

// out.println("<tr><td>Tid = "+tid+"     Titulo = "+titulo+"</td></tr>");
out.println("<tr><td><A HREF=\"publicar.jsp?tid="+tid+"\">"+identificador+"</a></td><td><A HREF=\"
```

```
}
%>

</table>
</div>

<br><div align=center><a href="javascript:history.go(-1)">Voltar</a></div>

<%

con.close();
}
catch(Exception e){
 out.println("<b>Erro de conexão com o banco.</b> - "+e.toString()+" <i></i>");
}
finally
{
try{ if (con != null) {con.close();}}catch (Exception e) {}
}%>

------------------------------

----------mostraadm.jsp-------

<%@ page import="java.sql.*"%>
<%
Connection con = null;
Statement stmt = null;
String query = null;
ResultSet rs = null;

//CONECTA AO BANCO DE DADOS
try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection ("jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDE
    stmt = con.createStatement();


%>
<div align=center>
<table>
<tr>
```

```
<td align=center height=35>
<b>SDAS - Sistema Denúncia Anônima Segura</br><br>ADM
</td>
</tr>
</table>
<%

String tid = "";
String nome= "";
String email = "";


query = "SELECT * FROM ADM WHERE ATIVO = 1 ORDER BY DATACADASTRO DESC";
rs = stmt.executeQuery (query);
%>
<table border=2>
<!--titulo-->
<tr bgcolor="99cc00ff">
<td>Nome</td><td>Email</td><td></td>
</tr>
<!--conteudo-->


<%

while(rs.next())
{
tid = rs.getString("TID");
    nome = rs.getString("NOME");
email = rs.getString("EMAIL");
out.println("<tr><td>"+nome+"</td><td>"+email+"</td><td><A HREF=alteraradm.jsp?tid="+tid+">Alterar

}
%>

</table>
</div>


<%

con.close();
}
```

```
catch(Exception e){
 out.println("<b>Erro de conexão com o banco.</b> - "+e.toString()+" <i></i>");
}
finally
{
try{ if (con != null) {con.close();}}catch (Exception e) {}
}%>

------------------------------

----------publicar.jsp--------

<%@ page import="tratadores.*" %>
<%@ page import="java.sql.*"%>

<%

String query;
String tid;
String titulo = "";
String descricao = "";
String local = "";
String hora = "";
String numero = "";
String data = "";
String ativo = "";
Connection con = null;
Statement stmt = null;
ResultSet rs = null;

tid = request.getParameter("tid");


try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection ("jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDE
stmt = con.createStatement();

    query = "SELECT * FROM SDAS WHERE TID = "+tid+"";
rs = stmt.executeQuery(query);

if (rs.next()) {
```

```
titulo = rs.getString("TITULO");
descricao =  rs.getString("descricao");
local = rs.getString("local");
hora = rs.getString("hora");
data = rs.getString("data");
ativo = rs.getString("ativo");
%>

<script>

 function analisar(form)
     {
 var analize;
 var pesquisa;
 analize=false;
 var tipo = 0;
 var mensagem = "Você deve preencher o campo";
     /*
 if (document.form.parecer.value=="")
 {
tipo++;
mensagem = mensagem + " " + "Parecer \n";
analize=true;
 }
     */
 var mensagem_final = mensagem ;

 if (tipo != 0 )
 {
    alert(mensagem_final);
 }
 if (tipo == 0 )
 {
       document.form.submit()
 }

 }
 </script>

<TABLE border=0 align=center>
<form action="confirmar.jsp" name="form" method="POST">
<TR>
<TD align=center>
```

```
   <b>Publicar Denúncia</b><br><br>
</TD>
</TR>
<TR>
<TD align=center>
   <b>Título:</b>
</TD>
</TR>
<TR>
<TD align=center>
   <%=titulo%>
</TD>
</TR>
<TR>
<TD align=center>
   <b>Descrição:</b>
</TD>
</TR>
<TR>
<TD align=center>
  <%=descricao%>
</TD>
</TR>
<TR>
<TD align=center>
  <b>Local:</b>
</TD>
</TR>
<TR>
<TD align=center>
   <%=local%>
</TD>
</TR>
<TR>
<TD align=center>
  <b>Data:</b>
</TD>
</TR>
<TR>
<TD align=center>
          <%=data%>
</TD>
</TR>
```

```
<TR>
<TD align=center>
  <b>Hora:</b>
</TD>
</TR>
<TR>
<TD align=center>
  <%=hora%>
</TD>
</TR>
<TR>
<TD align=center>
  <b>Parecer</b>
</TD>
</TR>
<TR>
<TD align=center>
 <TEXTAREA NAME="parecer" ROWS="10" COLS="21"></TEXTAREA>
</TD>
</TR>

<TR>
<TD align=center>
  <b>Publicar Denúncia?</b>
</TD>
</TR>
<TR>
<TD align=center>
  <INPUT type="radio" name="ativo" value="1" <% if (ativo.compareTo("1")==0) {out.println("checked
      <INPUT type="radio" name="ativo" value="2" <% if (ativo.compareTo("2")==0) {out.println("che

</TD>
</TR>

<TR>
<TD height=8>
<INPUT type="hidden" name="tid" value="<%=tid%>">
</TD>
</TR>

<TR>
<TD align=center>
 <input type="button" value=" Alterar Denúncia " onclick="javascript:analisar(this.form)">
```

```
</TD>
</TR>
</FORM>
</table>

<br><div align=center><a href="javascript:history.go(-1)">Voltar</a></div>

<%
} // fecha if



}
catch(Exception e){
 out.println("<b>Erro de conexão com o banco.</b> - "+e.toString()+" <i></i>");
}
finally
{
try{ if (con != null) {con.close();}}catch (Exception e) {}
}


%>

-------------------------------

--------quebrar.jsp-----------

<%@ page import="java.lang.*" %>
<%@ page import="java.util.Date" %>
<%@ page import="java.text.*" %>
<%@ page import="java.util.*" %>
<%@ page import="java.sql.*"%>
<%@ page import="tratadores.*" %>

<%
ResultSet rs = null;
Statement stmt = null;
Connection con = null;
String user = "sysdba";
String password = "masterkey";
String databaseURL = "jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDB/SDAS.GDB";
```

```
    String tid = request.getParameter("tid");
int segredo1 = Integer.parseInt(request.getParameter("segredo1"));
int segredo2 = Integer.parseInt(request.getParameter("segredo2"));
int segredo3 = Integer.parseInt(request.getParameter("segredo3"));

DivisaoSegredo divisao = new DivisaoSegredo();
int senha = divisao.recupera(segredo1,segredo2,segredo3);

Datacao datacao = new Datacao();
Date dataatual = datacao.getData();

//out.println("Data Atual: "+dataatual+"<br>");

//t0
Date t0 = new Date();

String identidadecifrada = "";
String identidade = "";


try
{


Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection (databaseURL, user, password);
stmt = con.createStatement();

        String query = "Select IDENTIDADE, DATACADASTRO from sdas where tid ="+tid;

 rs = stmt.executeQuery(query);
 if (rs.next())
 {
 identidadecifrada = rs.getString("IDENTIDADE");
 t0 = rs.getDate("DATACADASTRO");
         }
 //out.println("T0: "+t0+"<br><br>");

 int ano = t0.getYear();
 int mes = t0.getMonth();
 int dia = t0.getDate()+15;

 Date t1  =  new Date(ano,mes,dia);
```

```
 //out.println("T1: "+t1+"<br><br>");

 dia = t0.getDate()+30;
 Date t2 = new Date(ano,mes,dia);
 //out.println("T2: "+t2+"<br><br>");

 //if (dataatual.before(t0))
 //{

 //}

 if ((dataatual.after(t0))&&(dataatual.before(t1)))
 {
          //  entre t0 e t1  Identidade Ocultada
   %>
   <TABLE border=0 align=center>
 <TR>
<TD align=center>Identidade não pode ser revelada ainda. Aguarde período de revelação.</td>
 </tr>
   </table>
   <%
        }

 if ((dataatual.after(t1))&&(dataatual.before(t2)))
 {
          //  entre t1 e t2  Identidade Disponivel
   String chaveString = String.valueOf(senha);
   DES des = new DES(chaveString);
   identidade = des.decifra(identidadecifrada);
   %>
   <TABLE border=0 align=center>
 <TR>
<TD align=center>Identidade Passível de Relevação<br><br><b> <% if (identidade==null) { out.printl
 </tr>
   </table>
   <%
        }


        if (dataatual.after(t2))
 {
   //  depois de t2   Sem identidade
   %>
```

```
   <TABLE border=0 align=center>
 <TR>
<TD align=center>Identidade não disponível. Período Não Passível de Revelação.</td>
 </tr>
   </table>
   <%
         }

    %>
    <br><div align=center><a href="javascript:history.go(-1)">Voltar</a></div>
    <%

}
   catch (Exception e)
    {
out.println("<b>Erro: </b> - <i>"+e.toString()+"</i>");
}
   finally
{
   try{ if (con != null) {con.close();}} catch (Exception e) {}

} // fecha try

%>
-------------------------------

---------quebrarano.jsp-------

<%@ page import="tratadores.*" %>
<%@ page import="java.sql.*"%>

<%

String query;
String tid;
String titulo = "";
String descricao = "";
String local = "";
String hora = "";
String numero = "";
String data = "";
String ativo = "";
Connection con = null;
```

```
Statement stmt = null;
ResultSet rs = null;

tid = request.getParameter("tid");


try {
Class.forName ("org.firebirdsql.jdbc.FBDriver");
con = java.sql.DriverManager.getConnection ("jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDE
stmt = con.createStatement();

    query = "SELECT * FROM SDAS WHERE TID = "+tid+"";
rs = stmt.executeQuery(query);

if (rs.next()) {

titulo = rs.getString("TITULO");
descricao =  rs.getString("descricao");
local = rs.getString("local");
hora = rs.getString("hora");
data = rs.getString("data");
ativo = rs.getString("ativo");
%>

<script>
function analisar(form)
{

var analize;
var pesquisa;
analize=false;
var tipo = 0;
var mensagem = "Você não preencheu todos os Campos  - Chaves - ";

if (document.form.segredo1.value=="")
{
tipo++;
analize=true;
}
if (document.form.segredo2.value=="")
{
tipo++;
analize=true;
```

```
}
    if (document.form.segredo3.value=="")
{
tipo++;
analize=true;
}
var mensagem_final = mensagem ;

if (tipo != 0 )
{
 alert(mensagem_final);
}
if (tipo == 0 )
{
 document.form.submit()
    }


}
</script>


<TABLE border=0 align=center>
<form action="quebrar.jsp" name="form" method="POST">
<TR>
<TD align=center colspan=3>
   <b>Quebrar Anonimato</b><br><br>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
   <b>Título:</b>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
   <%=titulo%>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
   <b>Descrição:</b>
</TD>
```

```
</TR>
<TR>
<TD align=center colspan=3>
  <%=descricao%>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
  <b>Local:</b>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
   <%=local%>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
  <b>Data:</b>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
          <%=data%>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
  <b>Hora:</b>
</TD>
</TR>
<TR>
<TD align=center colspan=3>
  <%=hora%>
</TD>
</TR>

<TR>
<TD align=center colspan=3>
  <br><b>Chaves</b>
</TD>
</TR>
<TR>
```

```
<TD align=center><input type=text name="segredo1"></TD>
<TD align=center><input type=text name="segredo2"></TD>
<TD align=center><input type=text name="segredo3"></TD>
</TR>

<TR>
 <TD height=8 colspan=3>
<INPUT type="hidden" name="tid" value="<%=tid%>">
 </TD>
</TR>

<TR>
<TD align=center colspan=3>
 <br><input type="button" value=" Quebrar Anonimato " onclick="javascript:analisar(this.form)">
</TD>
</TR>
</FORM>
</table>

<br><div align=center><a href="javascript:history.go(-1)">Voltar</a></div>

<%
} // fecha if



}
catch(Exception e){
 out.println("<b>Erro de conexão com o banco.</b> - "+e.toString()+" <i></i>");
}
finally
{
try{ if (con != null) {con.close();}}catch (Exception e) {}
}


%>

------------------------------

-----DivisaoSegredo.java------

package tratadores;
```

```
import tratadores.*;
import java.util.Random;
import java.lang.Math;
import java.lang.String;
import java.lang.Integer;

/**
 * <p>Title: TCC</p>
 * <p>Description: SDAS</p>
 * <p>Copyright: Copyright (c) 2005</p>
 * <p>Company: </p>
 * @author Fabio
 * @version 1.0
*/

public class DivisaoSegredo
{

  private int s;  // mensagem
  private int r;  // sequencia aleatoria
  private int u;  // sequencia aleatoria
  private int t;
  private Random rand;


  public DivisaoSegredo()
  {
   rand = new Random();
   this.r =  Math.abs(rand.nextInt());
   this.u =  Math.abs(rand.nextInt());
  }

  public int geraSegredos(int s)
  {
  this.s = s;
  this.t = s ^ this.r ^ this.u;
  return this.t;
  }

  public int retornaS()
  {
  return this.s;
  }
```

```
  public int retornaT()
  {
  return this.t;
  }

  public int retornaR()
  {
  return this.r;
  }

  public int retornaU()
  {
  return this.u;
  }

  public int recupera(int t, int r, int u)
  {
   this.s = s = t ^ r ^ u;
   return this.s;
  }

}
```

```
-----------HASH.java----------

package tratadores;

import java.lang.Object;
import java.security.*;

/**
 * <p>Title: TCC</p>
 * <p>Description: SDAS</p>
 * <p>Copyright: Copyright (c) 2005</p>
 * <p>Company: </p>
 * @author Fabio
 * @version 1.0
 */
```

```
public class HASH {

  private String str;
  private MessageDigest hash;

  public HASH(String str, String algoritmo)
  {
    try
    {
      this.str = str;
      algoritmo = algoritmo.toUpperCase();
      this.hash = MessageDigest.getInstance(algoritmo);
    }
    catch (Exception e)
    {
      System.out.println("Erro");
    }
  }

  public static String Converte(byte[] b)
  {
    StringBuffer buf = new StringBuffer();
    String hexDigits = "0123456789abcdef";

    for (int i = 0; i < b.length; i++)
    {
        int j = ((int) b[i]) & 0xFF;
        buf.append(hexDigits.charAt(j / 16));
        buf.append(hexDigits.charAt(j % 16));
    }
    return buf.toString();
  }

  public String hash()
  {
    hash.reset();
    return  Converte(hash.digest(this.str.getBytes()));
  }
}

-----------------------------
```

```
------MandaMailAdm.java-------

package tratadores;

import java.io.*;
import java.util.*;
import java.sql.*;
import java.net.*;
import java.util.Properties;
import java.util.Date;
import javax.mail.*;
import javax.mail.internet.*;
import javax.activation.*;


public class MandaMailAdm {

public MandaMailAdm()
{
}

public void conecta(String texto, String chave, String id)

{
String nome = "";
String email = "";
Connection con = null;
Statement stmt = null;
ResultSet rs = null;
String user = "SYSDBA";
String password = "MASTERKEY";
String databaseURL = "jdbc:firebirdsql:localhost/3050:C:/paginajsp/SDAS/GDB/SDAS.GDB";
String SQL = "";

 try {

org.firebirdsql.jdbc.FBDriver driver = new org.firebirdsql.jdbc.FBDriver();
con = DriverManager.getConnection(databaseURL,"SYSDBA","masterkey");
stmt = con.createStatement();
SQL = "SELECT NOME, EMAIL FROM ADM WHERE ATIVO=1";
rs = stmt.executeQuery(SQL);

DivisaoSegredo divisao = new DivisaoSegredo();
```

```
divisao.geraSegredos(Integer.parseInt(chave));
int[] segredo = new int[3];
segredo[0] = divisao.retornaT();
segredo[1] = divisao.retornaR();
segredo[2] = divisao.retornaU();
int i = 0;

while (rs.next())
{
nome = rs.getString(1);
email = rs.getString(2);
    enviarmail(nome,email,texto,String.valueOf(segredo[i]),id);
i++;
    } // fecha while rs.next

     con.close();

    } catch(Throwable e) {
  System.out.println("Erro de conexão com o banco. - "+e.toString());
    }
    finally
    {
  try {
  if (con != null) {con.close();
  }
      }
  catch(Throwable e)
  {}
    }
} //fim conecta

public void enviarmail(String nome, String email, String texto, String chave, String id)
{

 try
  {

  String assunto="SDAS";
  String smtpAddr = "grucon.ufsc.br";
  String mailType = "text/plain";
  String strFemail = "admim@cimm.com.br"; //tem que ser assim por que o servidor de SMTP só manda
  String emaildest = email;
  String strMensagem = "\n Prezado(a) Senhor(a) "+nome+" \n  \n Você recebeu um e-mail do SDAS cor
```

```
  String subject = assunto;
  String smtpverif = "T";
  Properties props = System.getProperties();
  props.put("mail.smtp.host", smtpAddr);
  MimeMessage msg = new MimeMessage(Session.getDefaultInstance(props, null));
  msg.setFrom(new InternetAddress(strFemail));
  msg.setSubject(subject);
  msg.setContent(strMensagem,mailType);
  msg.addRecipient(Message.RecipientType.TO, new InternetAddress(emaildest));
  Transport.send(msg);
  }
 catch (Exception ex)
  {
  System.out.println("Erro ao enviar email aos administradores: "+ex.toString());
  }

}



} // fim classe

------------------------------

--------Utilitaria.java-------

package tratadores;

import java.util.Random;
import java.lang.Math;
import java.lang.String;
import java.lang.Integer;

/**
 * <p>Title: TCC</p>
 * <p>Description: SDAS</p>
 * <p>Copyright: Copyright (c) 2005</p>
 * <p>Company: </p>
 * @author Fabio
 * @version 1.0
 */

public class Utilitaria {
```

```
  public Utilitaria()
  {
  }

  public int GeraNumeroRandomico()
  {
    Random rand = new Random();
    return Math.abs(rand.nextInt());
  }

  public String converteInt(int x)
  {
    return String.valueOf(x);
  }

  public int converteString(String x)
  {
    return Integer.parseInt(x);
  }

  /**
  * Converte um array de byte em uma representação, em String, de seus hexadecimais.
  */
  public static String ConverteparaString(byte[] bytes)
  {
        if( bytes == null ) return null;
        String hexDigits = "0123456789abcdef";
        StringBuffer sbuffer = new StringBuffer();
        for (int i = 0; i < bytes.length; i++)
        {
            int j = ((int) bytes[i]) & 0xFF;
            sbuffer.append(hexDigits.charAt(j / 16));
            sbuffer.append(hexDigits.charAt(j % 16));
        }
        return sbuffer.toString();
  }


}
```

-------Certificado.java-------

```
package tratadores;

import java.lang.Object;
import java.security.*;
import java.security.cert.Certificate;
import java.io.*;

/**
 * <p>Title: TCC</p>
 * <p>Description: SDAS</p>
 * <p>Copyright: Copyright (c) 2005</p>
 * <p>Company: </p>
 * @author Fabio
 * @version 1.0
 */
public class Certificado
{

  public Certificado()
  {
  }

  public PrivateKey obterChavePrivadaArquivo( File cert, String alias, String senha ) throws Excep
  {
          KeyStore ks = KeyStore.getInstance ( "JKS" );
          char[] pwd = senha.toCharArray();
          InputStream is = new FileInputStream( cert );
          ks.load( is, pwd );
          is.close();
          Key key = ks.getKey( alias, pwd );
          if( key instanceof PrivateKey ) {
              return (PrivateKey) key;
          }
          return null;
  }

  public PublicKey obterChavePublicaArquivo( File cert, String alias, String senha ) throws Except
  {
        KeyStore ks = KeyStore.getInstance ( "JKS" );
        char[] pwd = senha.toCharArray();
        InputStream is = new FileInputStream( cert );
        ks.load( is, pwd );
```

```
        Key key = ks.getKey( alias, pwd );
        Certificate c = ks.getCertificate( alias );
        PublicKey p = c.getPublicKey();
        return p;
    }

    /**
     * Retorna a assinatura para o buffer de bytes, usando a chave privada.
     */
    public byte[] criarAssinaturaChavePrivada( PrivateKey chave, byte[] buffer ) throws Exception
    {
        Signature sig = Signature.getInstance("MD5withRSA");
        sig.initSign(chave);
        sig.update(buffer, 0, buffer.length);
        return sig.sign();
    }

    /**
     * Retorna a assinatura para o buffer de bytes, usando a chave publica.
     */
/*   public byte[] criarAssinaturaChavePublica( PublicKey chave, byte[] buffer ) throws Exceptior
    {
        Signature sig = Signature.getInstance("MD5withRSA");
        sig.initVerify(chave);
        sig.update(buffer, 0, buffer.length);
        return sig.sign();
    }*/


    /**
    * Verifica a assinatura para o buffer de bytes, usando a chave pública.
    * @param key PublicKey
    * @param buffer Array de bytes a ser verficado.
    * @param sgined Array de bytes assinado (encriptado) a ser verficado.
    */
   public boolean verificaAssinatura(PublicKey chave, byte[] buffer, byte[] signed ) throws Except
       Signature sig = Signature.getInstance("MD5withRSA");
       sig.initVerify(chave);
       sig.update(buffer, 0, buffer.length);
       return sig.verify( signed );
   }
```

```
}

-----------------------------

--------Datacao.java----------

package tratadores;

import java.util.Date;
import java.util.GregorianCalendar;
import java.text.DateFormat;
import java.text.SimpleDateFormat;

import br.bry.pdde.PDDE;
import br.bry.util.*;
import java.io.*;

/**
 * <p>Title: TCC</p>
 * <p>Description: SDAS</p>
 * <p>Copyright: Copyright (c) 2005</p>
 * <p>Company: </p>
 * @author Fabio
 * @version 1.0
 */

public class Datacao {

 PDDE pdde;
 Date data;
 SimpleDateFormat formatador;

 public Datacao()
 {
    pdde = new PDDE();
    data = new Date();
 }

 public void SetaPDDE()
 {
    pdde.setServidor("200.247.159.135");
pdde.setPorta(318);
pdde.setTipoArqSaida(0);
```

```
pdde.setPoliticaRequisicao("1.3.6.1.4.1.14975.2.1.0");
pdde.setCaminhoArqSaida("C:/htdocs/sdas/arquivos");
pdde.setVerificaOIDTimestamp(1);
 }

 public void SetaArquivo(String arq)
 {
pdde.protocolaArquivo("C:/htdocs/sdas/arquivos/"+arq);
 }

 public String DataPDDE()
 {
    return pdde.getDataRecibo();
 }

 public String retornaData()
 {
formatador = new SimpleDateFormat("dd/MM/yyyy");
    return formatador.format(data);
 }


 public String retornaHora()
 {
    formatador = new SimpleDateFormat("hh:mm");
    return formatador.format(data);
 }

 public Date getData()
 {
    return this.data;
 }


}

------------------------------


----------DES.java------------

package tratadores;
```

```
import java.lang.Object;
import java.security.*;
import java.security.cert.Certificate;
import java.io.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import javax.crypto.spec.PBEKeySpec;
import javax.crypto.spec.PBEParameterSpec;
import java.security.spec.KeySpec;
import java.security.spec.AlgorithmParameterSpec;

/**
 * <p>Title: TCC</p>
 * <p>Description: SDAS</p>
 * <p>Copyright: Copyright (c) 2005</p>
 * <p>Company: </p>
 * @author Fabio
 * @version 1.0
 */

public class DES {

  private Cipher cifrador;
  private Cipher decifrador;

  private byte[] salt =
      {
           (byte)0xA9, (byte)0x9B, (byte)0xC8, (byte)0x32,
           (byte)0x56, (byte)0x35, (byte)0xE3, (byte)0x03
      };

  private int contadordeinteracao = 19;


  public DES(String senha)
  {
          try
          {
              // Cria as chaves
              KeySpec keySpec = new PBEKeySpec(senha.toCharArray(), salt, contadordeinteracao);
              SecretKey chave = SecretKeyFactory.getInstance("PBEWithMD5AndDES").generateSecret(ke
              cifrador = Cipher.getInstance(chave.getAlgorithm());
              decifrador = Cipher.getInstance(chave.getAlgorithm());
```

```
            AlgorithmParameterSpec paramSpec = new PBEParameterSpec(salt, contadordeinteracao);

            // Criando os cifradores...
            cifrador.init(Cipher.ENCRYPT_MODE, chave, paramSpec);
            decifrador.init(Cipher.DECRYPT_MODE, chave, paramSpec);
        }
        catch (Exception  e)
        {
            System.out.println("Erro DES: "+e);
        }
}

public String cifrar(String str)
{
        try
        {
            byte[] utf8 = str.getBytes("UTF8");
            //Cifrando...
            byte[] enc = cifrador.doFinal(utf8);
            return new sun.misc.BASE64Encoder().encode(enc);
        }
        catch (Exception e)
        {
            System.out.println("Erro DES: "+e);
        }
        return null;
}

public String decifra(String str)
{
        try
        {
            byte[] dec = new sun.misc.BASE64Decoder().decodeBuffer(str);
            // Decifrando...
            byte[] utf8 = decifrador.doFinal(dec);
            return new String(utf8, "UTF8");
        }
        catch (Exception e)
        {
            System.out.println("Erro DES: "+e);
        }
        return null;
```

```
  }

  }  //fim classe

------------------------------
```