Cryon Crypto Signer – Um Sistema de Assinatura Digital baseado no OpenPGP

Yan Uehara de M., Brivaldo Junior[1]

[1]Faculdade de Computação – Universidade Federal do Mato Grosso do Sul (UFMS)
Campo Grande – MS – Brasil

{yan.uehara, brivaldo}@facom.ufms.br

Abstract. *Digital signing mechanisms are powerful to validate digital documents. And, to ensure the integrity and confiability of the human communications, cryptographic mecanisms capable of validating identities were built. One of those mechanisms, important in that process, is the digital signing. Nowadays, digital signatures are done using tokens or similar systems and their maintenance is expensive. This work shows a mechanism of digital signing using the OpenPGP specification to create a safe topology of document signing without using additional specific physical devices.*

Resumo. *Mecanismos de assinatura digital são poderosos para validar documentos digitais. Para garantir a integridade e confiabilidade nas comunicações humanas, criaram-se mecanismos criptográficos capazes de validar identidade. Um dos mecanismos, importantes neste processo, é o de assinatura digital de documentos. Hoje, assinaturas são realizadas por meio de tokens ou sistemas similares e sua manutenção é dispendiosa. Este trabalho apresenta um mecanismo de assinatura digital utilizando o padrão OpenPGP para criar uma topologia segura de assinatura de documentos sem a necessidade de dispositivos físicos adicionais.*

Apoio financeiro da Fundect - T.O. nº 052/2014 - SIAFEM nº 023538

1. Introdução

A tecnologia é cada vez mais presente no cotidiano das pessoas e empresas. Por esse motivo, a confidencialidade, integridade e disponibilidade na troca de informações (pessoais ou não) é uma preocupação constante. Além disso, a tecnologia pode e deve ser usada como meio para acelerar processos. O problema é como garantir que os documentos tramitados estão autorizados ou tenham o ciente dos respectivos gestores ou responsáveis.

A criptografia, do grego *"kryptós graphia"* – escrita escondida – veio para garantir a confidencialidade e integridade para essas interações. A criptografia é utilizada nas mais variadas situações, tanto na Internet quanto no mundo real, por exemplo, quando se acessa um site pelo protocolo HTTPS [Rescorla 2000, Dierks 2008] ou quando se realizam transações com cartão de crédito [Degabriele et al. 2012].

A criptografia é, geralmente, dividida em duas grandes áreas: simétrica e assimétrica, essa última também chamada de criptografia de chave pública que segue os princípios regidos pelo PKI (*Public Key Infrastructure*). Na primeira, a chave que cifra a

mensagem é a mesma que decifra (*shared key*). Já na segunda, existem duas chaves matematicamente relacionadas: a pública e a privada. A privada utilizada para criptografia (ou geração de *hash* criptográfico) e a pública na descriptografia (ou validação de um *hash* gerado pela chave privada).

A assinatura digital usa o mecanismo de geração de sumários de mensagens assinadas pela chave privada e validados pela chave pública. No modelo atual, as chaves públicas são armazenadas em repositórios acessíveis publicamente. O modelo tradicional hierárquico utiliza uma estrutura híbrida de autoridades certificadoras (AC), autoridades de requisição (AR) e as chaves propriamente ditas e a confiança está na autoridade que emitiu ou validou o certificado. No modelo distribuído, como o proposto pelo PGP [Garfinkel 1995], servidores armazenam chaves públicas e a confiança é par-a-par.

Este trabalho utiliza a estrutura proposta pelo PGP para criar um ambiente de segurança para assinaturas de documentos digitais de forma a não depender do modelo de confiança par-a-par. Por um lado, a ferramenta desenvolvida assina documentos usando os princípios de chaves pública e privada e, por outro, um centralizador de chaves garante que a estrutura altamente flexível do PGP seja restringida.

O trabalho está organizado da seguinte maneira: na Seção 2, são avaliadas as tecnologias relacionadas a assinaturas digitais. Na Seção 3 é descrito o sistema proposto e seus requisitos operacionais. Na Seção 4 é descrita a ferramenta desenvolvida e o processo de assinatura e validação de documento. Finalmente, na Seção 5 é feita a conclusão e descrição dos trabalhos futuros.

2. Tecnologias Associadas

2.1. Padrão x509

O padrão x509 é uma especificação aprovada pela *Internet Engineering Task Force – IETF* que é responsável por definir *standards* da web, sendo a versão mais recente do padrão a RFC 5280 [Boeyen et al. 2008], que estabelece a versão 3. O padrão estabelece uma Infraestrutura de Chaves Públicas – também chamada de PKI, do inglês *Public Key Infrastructure* – hierárquica. O padrão define a existência de uma Autoridade Certificadora raiz – CA e é dela que a confiança se origina, por isso a CA é também chamada de terceiro confiável.

No padrão x509 a chave pública é emitida pela CA e armazenada em um certificado assinado com a chave privada da CA. Esse certificado contém a chave pública e nome do dono do certificado, a data de expiração, nome da CA que emitiu o certificado, seu número serial, e a assinatura da CA. Esse certificado pode ser utilizado para diversas aplicações, incluindo o protocolo SSL/TLS na Internet, autenticação e assinatura digital [Boeyen et al. 2008, Fatima et al. 2015]. O certificado é também armazenado pela CA em um diretório publicamente acessível, cuja função é tão-somente facilitar a busca pelos certificados e validação das chaves por ela emitidas.

Revogação de um Certificado

É esperado que o certificado seja utilizado por todo período em que ele é válido, mas por diversas circunstâncias o certificado pode se tornar inválido antes do período de expiração,

tais como: falha de segurança da chave privada associada, mudança de nome, mudança do status de associação entre o possuidor do certificado e da CA, etc.

Nesses casos a CA deve revogar o certificado, sendo tal revogação feita por meio da publicação de uma Lista de Certificados Revogados – CRL. A CRL é uma lista com um marcador de tempo que contém a lista dos números seriais dos certificados revogados e é assinada pela CA. Um sistema que utilize esses certificados não só verifica a autenticidade e validade do mesmo mas também se aquele aparece na mais recente CRL publicada.

2.2. Padrão OpenPGP

O OpenPGP é uma outra especificação aprovada pela IETF. A especificação OpenPGP surgiu cerca de seis anos após a criação do PGP, abreviação de *Pretty Good Privacy*, um programa criado em 1991 por Phil Zimmermann [Garfinkel 1995]. Na época o PGP era distribuído como software livre, mas o software infringia algumas patentes, como a da criptografia RSA. Tal situação só foi normalizada quando em 1993, a ViaCrypt, que detinha a licença de uso de algumas das tecnologias utilizadas, negociou com Zimmermann para lançar uma versão comercial do PGP.

Em 1996 Zimmermann fundou a PGP Inc., que eventualmente foi absorvida pela ViaCrypt. Por iniciativa do próprio Zimmermann, enquanto trabalhava na PGP Inc, a empresa propôs para o IETF o padrão OpenPGP, além de permitir o uso da marca para os programas que atendessem ao formato. Atualmente a especificação mais recente do padrão é a RFC 4880 [Callas et al. 2007].

Funcionamento e *Web of Trust*

Segundo a especificação do padrão, o OpenPGP oferece, para mensagens e arquivos, as seguintes funções:

- Encriptação
- Assinatura digital
- Compactação
- Gerência de chaves (criação, edição e certificação de chaves de terceiros)

Para o uso do PGP são criadas duas chaves: uma para encriptação e outra para certificação (assinatura). A recomendação atual é que as duas chaves sejam criadas utilizando o algoritmo RSA, que cria um par de chaves público/privada com o tamanho de chave de pelo menos 2048 bits. Cada chave pública é armazenada em um certificado PGP que contém a chave pública em si, o(os) id(s) do proprietário da chave, a data de expiração e uma lista de assinaturas na chave [Garfinkel 1995, Callas et al. 2007]. Os certificados PGP podem ser opcionalmente, para maior facilidade de busca, serem transferidos para um servidor de chaves, como os servidores da RNP e MIT [CAIS/RNP , MIT IS&T Server Operations].

A habilidade do OpenPGP de certificar/assinar a chave de terceiros é a grande diferença entre o mesmo e o padrão x509. Cada proprietário de um par de chaves pode certificar uma chave pública e atestar sua veracidade e confiança, e cada chave pode ter inúmeras certificações desse tipo. Essas certificações podem ser transitivas, por exemplo, se eu confio na chave de Alice, e ela assinou a chave pública de Bob, eu confio (em algum nível) na chave de Bob. Esse esquema de assinatura cria uma rede de confiança distribuída também chamada de de *Web of Trust* [Yamane et al. 2003].

Revogação de um Certificado/Chave Pública

Devido a quebra de segurança, ou outros motivos, talvez seja necessário revogar uma chave pública. Essa revogação é feita criando-se um certificado de revogação, assinado pela chave privada associada, que posteriormente é distribuído publicamente, principalmente por meio do servidor de chaves.

Uma diferença derivada da arquitetura dos dois padrões, é que, enquanto no x509 a revogação pode ser feita pelo terceiro confiável, no OpenPGP, uma vez que se perca a chave privada ou sua senha seja esquecida, não é possível revogá-la sem o certificado de revogação.

3. *Crayon Crypto Signer*

O *Crayon Crypto Signer*, abreviado como CCS, é uma proposta de sistema para assinatura digital aliado a um sistema de informação pré-existente que necessite e suporte o *upload* de arquivos assinados digitalmente. O CCS foi concebido para ser uma alternativa de baixo custo a assinatura digital "clássica" – aquela baseada em certificados x509 – utilizando o OpenPGP como padrão.

Utilizando um sistema preexistente, o funcionamento segue os seguintes passos:

1. O usuário pede no sistema uma autorização para criar a chave;
2. O sistema gera um arquivo de autorização que será utilizado no programa *desktop* do CCS como entrada para criação do par de chaves;
3. No programa *desktop* o usuário insere o arquivo de autorização, confirma os dados e define a senha;
4. O sistema gera o par de chaves, armazena-os em um pendrive e gera o formulário de validação da chave;
5. O usuário faz o *upload* da chave pública para o servidor de chaves e ele deverá entregar o formulário de validação devidamente assinado no local onde o administrador do sistema designar;
6. Somente após o passo anterior os arquivos assinados serão validados corretamente pelo sistema;
7. Para fazer a assinatura, o usuário, no programa *desktop*, insere o pendrive e seleciona os arquivos a serem assinados;
8. O programa *desktop* pede então a senha ao usuário e assina os arquivos, trocando a extensão para .sig (que indica um arquivo assinado);
9. Para verificar e abrir o arquivo, o usuário, no programa *desktop*, seleciona o arquivo e pede para validá-lo.

A partir desse roteiro são estabelecidos os requisitos apresentados na Tabela 1. Eles também podem ser visualizados por meio de modelos *BPMN*. Estes modelos são utilizados para melhor visualização das funcionalidades e comportamentos do sistema.

Os requisitos de Geração de Chaves e Armazenagem de Chaves, podem ser melhor apresentados pelo modelo *BPMN* da Figura 1 que também inclui os requisitos da Geração do Formulário de Autorização e Validação do Formulário de Autorização.

A Recriação de Chave é um dos poucos requisitos que funcionam de maneira automática, avisando ao usuário da necessidade de se criar um chave nova após a expiração da chave anterior. Esse requisito é apresentado no modelo da Figura 2.

Já os requisitos de Assinatura de arquivos e Revogação de chaves podem ser vistos, respectivamente, nas Figuras 3 e 4.

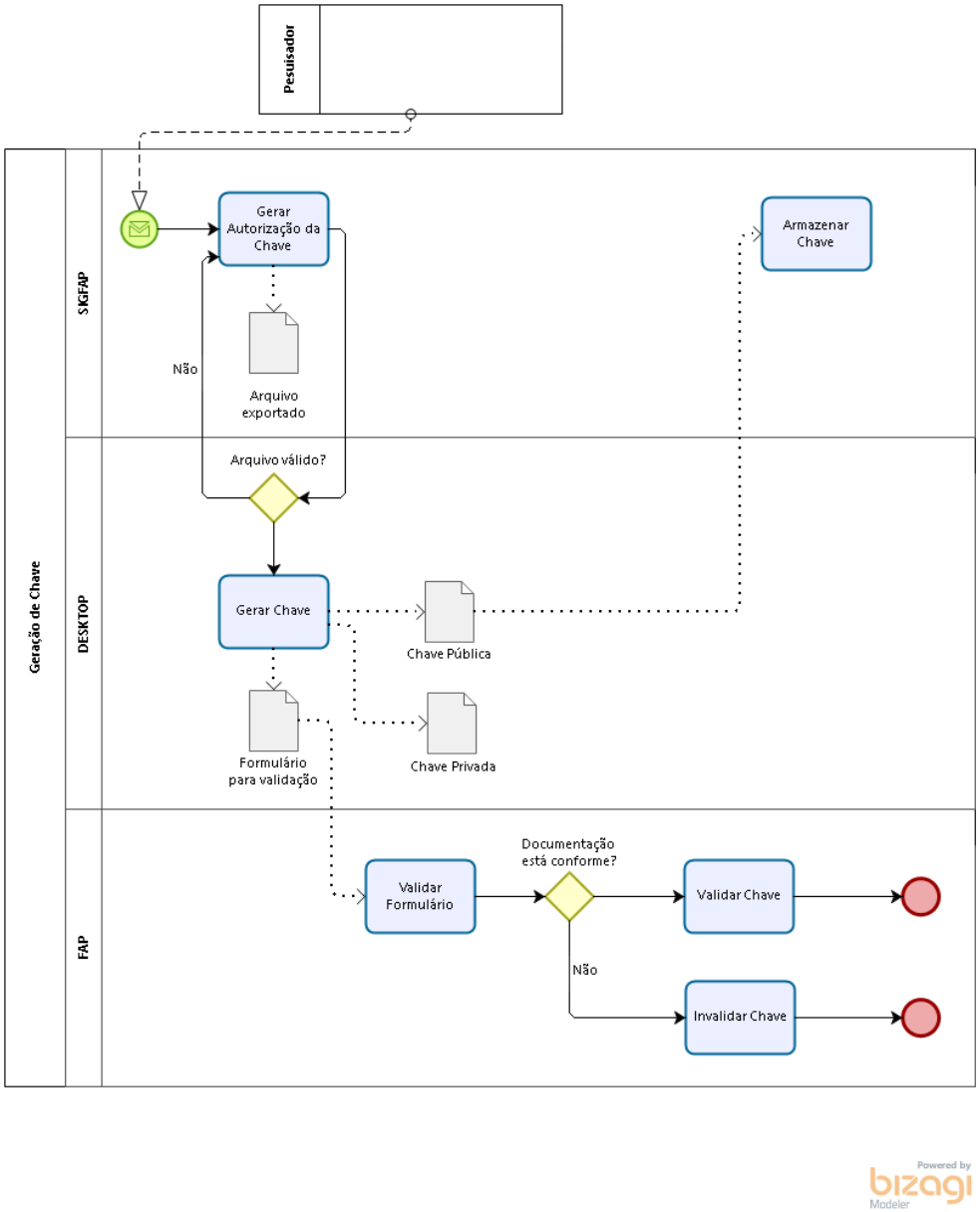

Figura 1. Modelo BPMN da geração de uma chave

Arquitetura

O sistema se divide em três grandes módulos: o programa *desktop*, o repositório de chaves e a componente junto ao sistema em que ele será utilizado.

Cada parte do sistema tem uma responsabilidade diferente: o programa *desktop* é responsável pela criação das chaves, assinatura e verificação dos arquivos; o repositório é o local onde as chaves públicas ficarão armazenadas; e a componente junto ao sistema é responsável por verificar se os arquivos enviados foram assinados com uma chave válida e autorizada.

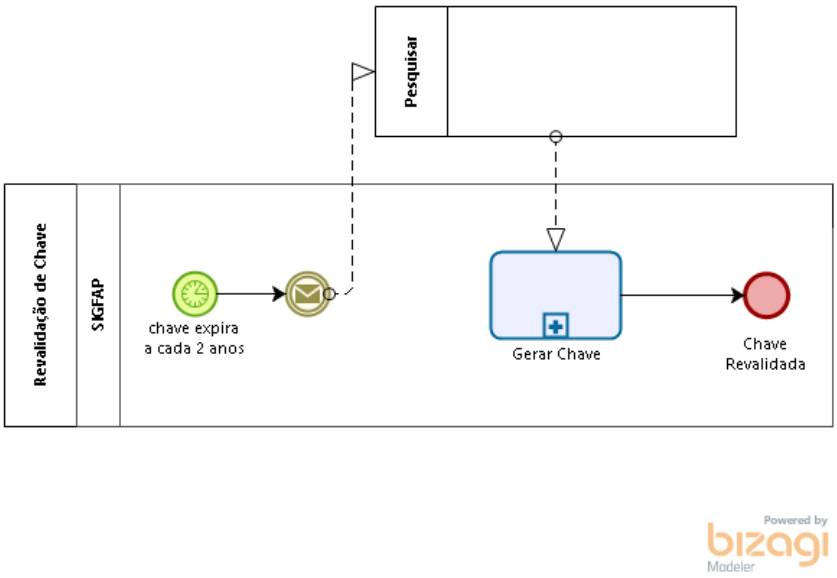

Figura 2. Modelo BPMN da Recriação de uma chave

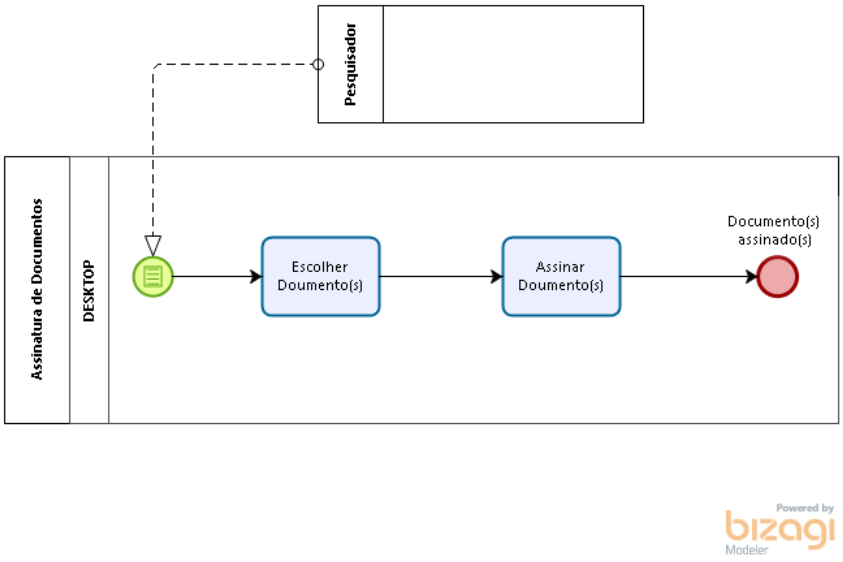

Figura 3. Modelo BPMN da Assinatura de documento

Ciclo de vida da chave e servidor de chaves

O repositório de chaves é a componente que guarda todas as chaves geradas, tanto as expiradas quanto as válidas, de acordo com os requisitos RF006 e RF007 da Tabela 1. Além das chaves, o repositório também guarda seu *status* sendo válida ou inválida.

A componente que associa um usuário a uma chave é o formulário de validação de chave. Ele contém o *fingerprint* – a identificação única – da chave, além dos dados pessoais do usuário e sua assinatura física. Após sua entrega, a chave associada ao usuário passa para o *status* válido, e permanece nesse estado até que ela seja revogada ou expire.

A chave tem um ciclo de vida controlado, sendo que cada uma tem validade de

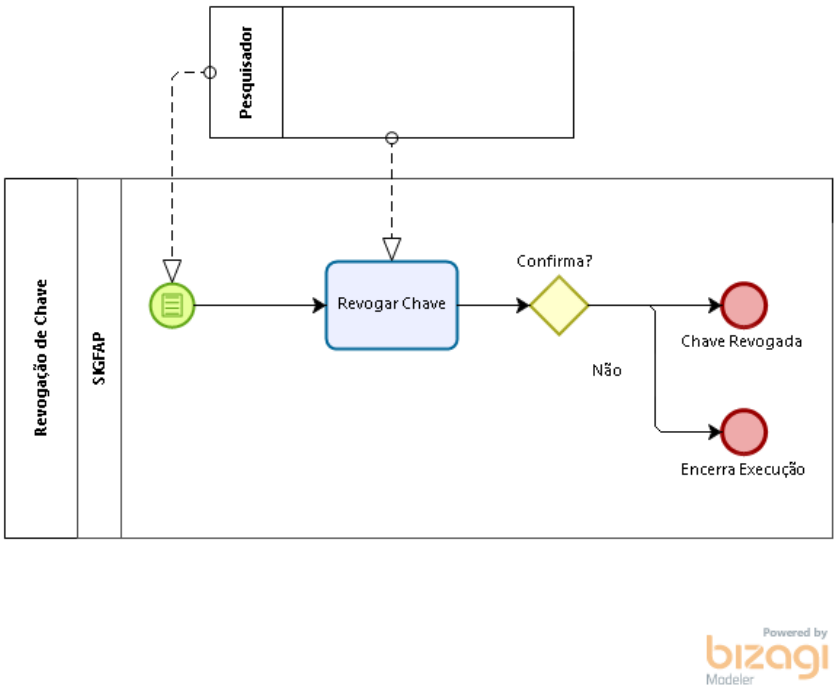

Figura 4. Modelo BPMN da Revogação de uma chave

dois anos. Ela é armazenada em um pendrive do usuário para diminuir a superfície de ataque e cada usuário poderá ter somente uma chave dentro do sistema. O *id* da chave é construído da seguinte maneira: Nome completo (CPF) <e-mail cadastrado no sistema>.

A chave é automaticamente expirada depois do prazo de validade. Quando usuário acessa o sistema depois desse tempo, ele indica ao usuário a necessidade de se criar uma nova chave. O usuário também pode revogar a chave antes do fim do período de validade, podendo fazê-lo pelo programa *desktop* ou pela componente junto ao sistema. Caso o usuário tenha perdido a chave, a revogação se dá somente no servidor de chaves, marcando-a como inválida no repositório.

Adicionalmente aos requisitos RF005 e RF007, presentes na Tabela 1, a chave tem um tamanho mínimo e formato de senha definidos no requisito RF008. Além disso, para aumentar a segurança da chave, ela tem o tamanho de 4096 *bits* e assina os arquivos utilizando o *hash* SHA256 em detrimento ao, inseguro, SHA1.

A componente junto ao sistema de informação

Quando o usuário faz o *upload* de um arquivo assinado, essa componente verifica o emissor, busca no repositório de chaves a chave pública do usuário e valida o arquivo. A componente é responsável também por mostrar o *status* da validação do arquivo para o usuário. Além disso, chaves expiradas são mantidas para garantir que documentos antigos sejam validados. A componente leva em consideração a data de submissão do documento com a chave pública válida no período. Desta forma, documentos antigos continuaram sendo válidos dentro do sistema.

4. Programa *Desktop*

A aplicação *desktop* foi desenvolvida para atingir ao maior número de plataformas operacionais. Por este motivo, optou-se pelo desenvolvido na linguagem Java e uso da biblioteca JavaFX [Oracle Corporation] (como biblioteca para aplicações gráficas em *desktop* adaptadas ao sistema do usuário). Para as rotinas de criação e manipulação das chaves OpenPGP utilizou-se a biblioteca *Bouncycastle* [bou].

O desenvolvimento do programa *desktop* utilizou o paradigma *Model-View-Controller* (MVC). O *Model* do programa é a manipulação das chaves e arquivos, a *View* é a interface pré-desenhada em formato FXML para carregamento pelo JavaFX, e o *Controller* que é o núcleo do programa em Java. A *view* principal e a *view* de criação de chaves podem ser visualizadas respectivamente nas Figuras 5 e 6.

Tabela 1. Requisitos Funcionais

ID	Requisito	Descrição
RF001	Geração de Chaves	O sistema deve gerar um par de chaves (pública e privada), a qual dar-se-á através de uma senha particular do usuário
RF002	Assinatura de Documentos	O sistema desktop deve realizar a assinatura digital de documentos
RF003	Armazenagem de Chaves	O sistema deve armazenar as chaves públicas geradas
RF004	Validação dos documentos assinados	O sistema deve verificar os documentos assinados
RF005	Recriação de chaves	O sistema deve expirar as chaves que possuam mais de 2 anos, fazendo com que o usuário deva gerar uma nova chave
RF006	Assinaturas Antigas	Documentos antigos assinados por chaves expiradas, devem continuar válidos, porém, aquelas não podem assinar novos documentos
RF007	Revogação de Chaves	O sistema deve permitir que o usuário revogue sua chave; documentos antigos assinados com chaves revogadas devem continuar válidos, porém chaves revogadas não podem assinar novos documentos
RF008	Senha da chave privada	O programa deve solicitar que o usuário crie uma senha com o seguinte formato: pelo menos 01 caractere em maiúsculo, 01 caractere em minúsculo e 01 caractere alfanumérico, totalizando no mínimo 08 caracteres para a senha

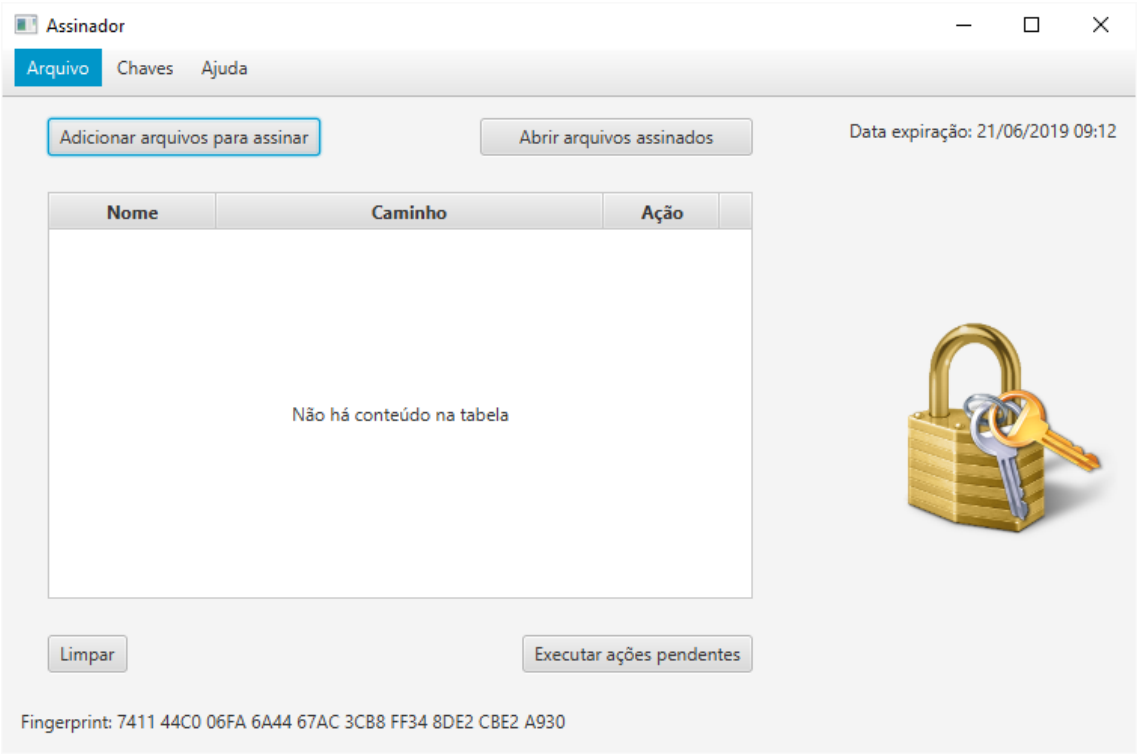

Figura 5. Tela principal.

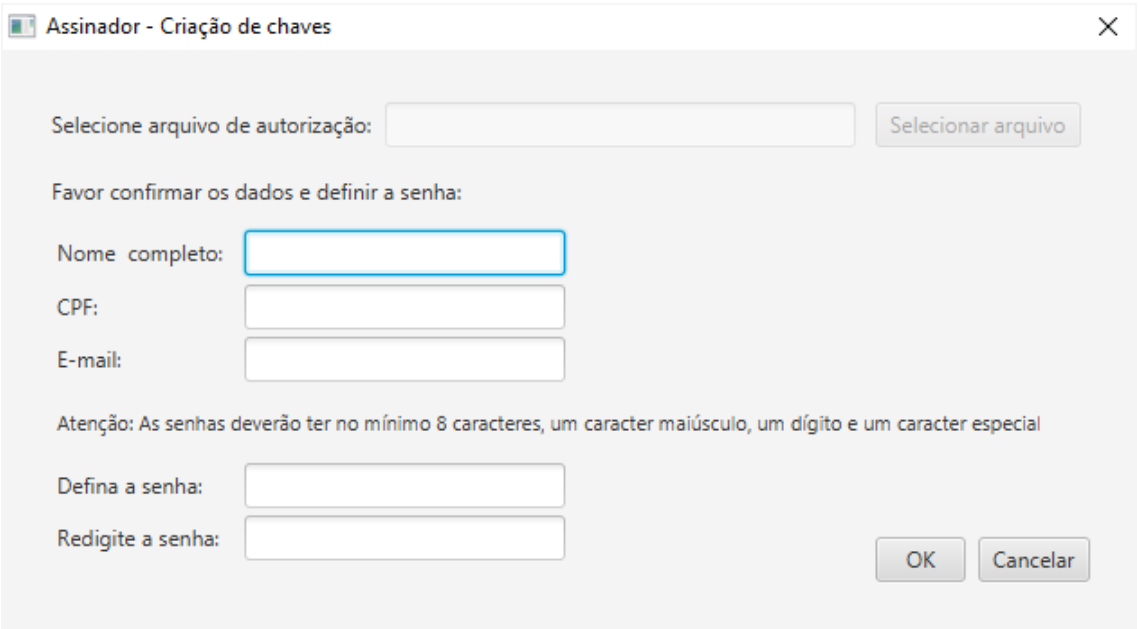

Figura 6. Tela de criação de chaves

Para exercer a responsabilidade do *Model* foi criada uma biblioteca chamada *libsigner*, responsável unicamente pelas funções básicas de manipulação das chaves: criação, revogação, assinatura de documentos e verificação de arquivos assinados.

Além disso, a biblioteca faz a captura dos erros da biblioteca *bouncycastle* e relança-os em uma hierarquia própria de erros. Essa interceptação das exceções traz mais

facilidade em seu tratamento, encapsulando os erros que a biblioteca *bouncycastle* possa lançar para erros mais verbosos e que indiquem com mais clareza o erro e/ou falha que ocorreram. A hierarquia de exceções pode ser vista na Figura 7

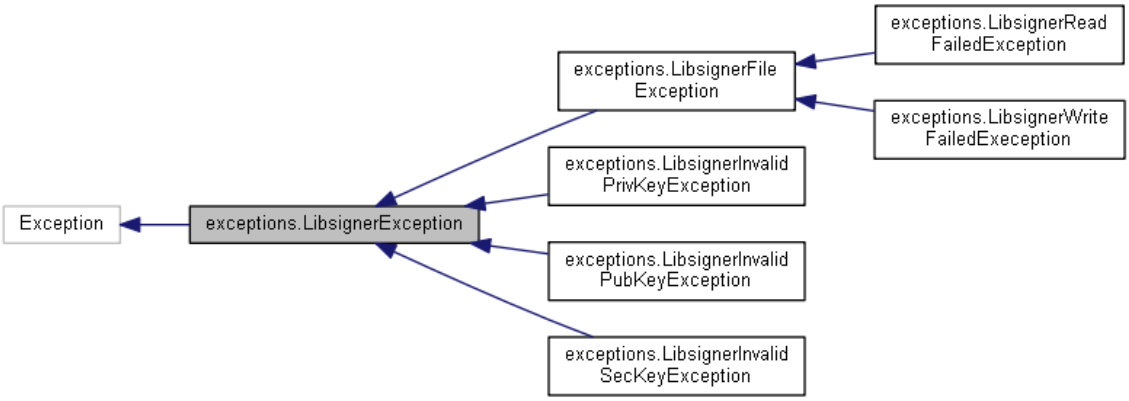

Figura 7. Hierarquia de Exceções

4.1. Corretude da criação das chaves e assinatura de arquivos

Para garantir que a criação de chaves e a assinatura de arquivos fossem realizadas dentro do padrão OpenPGP, durante a etapa de testes realizou-se a validação dos artefatos criptográficos gerados utilizando o GnuPG [gnu].

Foi criada uma chave exemplo com CPF gerado aleatoriamente, cujo *fingerprint* e data de validade podem ser vistos dentro da interface do programa na Figura 5. Os dados da chave obtidos pelo programa GnuPG, podem ser vistos na Figura 8. Nas Figuras é possível perceber que os dados apresentados pelos dois programas são iguais, garantindo que a chave criada pela aplicação *desktop* segue o padrão corretamente.

Além disso, foram realizados testes da assinatura e verificação de um arquivo. O arquivo de entrada com o nome de *teste.pdf* e o arquivo de saída, *teste.sig*. A assinatura foi corretamente validada utilizando novamente o programa GnuPG, cuja saída pode ser vista na Figura 4.1.

```
>gpg2 --list-keys --keyid-format long --fingerprint
pubring.gpg
------------------------------------------------
pub   4096s/FF348DE2CBE2A930 2017-06-21 [expires: 2019-06-21]
      Key fingerprint = 7411 44C0 06FA 6A44 67AC  3CB8 FF34 8DE2 CBE2 A930
uid                  [ unknown] Fulano de Tal (58338668630) <fulano.detal@empresa.com.br>
sub   4096r/A43C8BC373D61813 2017-06-21
```

Figura 8. Dados da chave, pelo programa GnuPG

5. Conclusão

O sistema proposto utiliza o OpenPGP de modo a garantir de modo inequívoco a associação entre a identidade física do usuário e sua assinatura digital. Ele garante ainda os princípios da segurança da informação: confidencialidade, integridade, autenticidade, disponibilidade e não-repúdio.

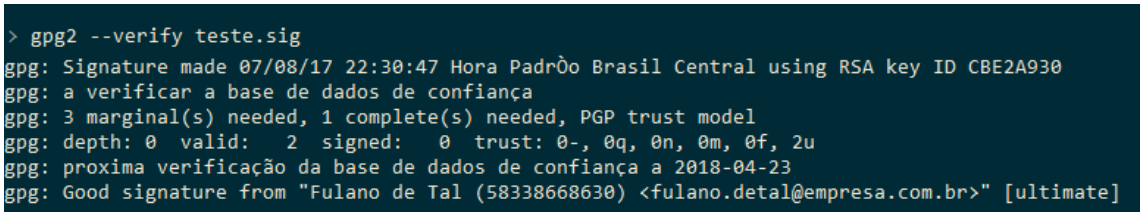

Figura 9. Verificação do arquivo de teste, pelo programa GnuPG

A integridade, a autenticidade e o não-repúdio são garantidos pelo processo de assinatura, dado que qualquer alteração no arquivo assinado invalida sua assinatura que só pode ser efetuada com a chave unicamente associada ao usuário. A disponibilidade é alcançada pela disponibilidade em conjunto do sistema e do servidor de chaves. Já a confidencialidade é garantida por meio da necessidade da criação do arquivo de autorização para a criação da chave e a senha segura criada pelo usuário.

O uso da linguagem Java no programa *desktop* permite que o ele seja multiplataforma e disponha de uma interface amigável ao usuário e uniforme em todos os sistemas operacionais. E o armazenamento da chave no pendrive do usuário permite que a chave seja utilizada em todas as máquinas e sistemas que o usuário vier a utilizar.

Apesar dos requisitos do sistema estarem enumerados e o programa estar em fase de teste e finalização, incluindo seu registro no órgão competente, o servidor de chaves e o componente junto ao sistema de informação, e seus demais aspectos, ainda necessitam de maior pesquisa e posterior implementação. Uma versão de demonstração, que somente cria as chaves localmente e assina arquivos, pode ser obtido em [ccs].

Referências

[bou] Bouncycastle. `https://www.bouncycastle.org/`.

[ccs] Crayon Crypto Signer – CCS (link não oficial dado o processo de revisão cego). `http://condector.dyndns.org:8888/appdesktop.zip`.

[gnu] GnuPG. `https://www.gnupg.org/`.

[Boeyen et al. 2008] Boeyen, S., Santesson, S., Polk, W. T., Housley, R., Farrell, S., and Cooper, D. (2008). Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280.

[CAIS/RNP] CAIS/RNP. Servidor de chaves públicas PGP do CAIS/RNP. `https://www.rnp.br/servicos/servicos-avancados/icpedu/servidor-chaves-pgp`.

[Callas et al. 2007] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and Thayer, R. (2007). OpenPGP Message Format. RFC 4880.

[Degabriele et al. 2012] Degabriele, J. P., Lehmann, A., Paterson, K. G., Smart, N. P., and Strefler, M. (2012). *On the Joint Security of Encryption and Signature in EMV*, pages 116–135. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Dierks 2008] Dierks, T. (2008). The transport layer security (TLS) protocol version 1.2. RFC 5246.

[Fatima et al. 2015] Fatima, S., Ahmad, S., and Siddiqui, S. (2015). X. 509 and PGP Public Key Infrastructure methods: A critical review. *International Journal of Computer Science and Network Security (IJCSNS)*, 15(5):55.

[Garfinkel 1995] Garfinkel, S. (1995). *PGP: Pretty Good Privacy*. Encryption for everyone. O'Reilly Media, Incorporated.

[MIT IS&T Server Operations] MIT IS&T Server Operations. MIT PGP Public Key Server. `https://pgp.mit.edu/`.

[Oracle Corporation] Oracle Corporation. JavaFX. `https://docs.oracle.com/javase/8/javase-clienttechnologies.htm`.

[Rescorla 2000] Rescorla, E. (2000). HTTP over tls. RFC 2818.

[Yamane et al. 2003] Yamane, S., Wang, J., Suzuki, H., Segawa, N., and Murayama, Y. (2003). Rethinking openpgp PKI and openpgp public keyserver. *CoRR*, cs.CY/0308015.