CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA
FUNDAÇÃO DE ENSINO "EURÍPIDES SOARES DA ROCHA"
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Implementação de uma API Java de geração de assinaturas digitais de acordo com as regulamentações da ICP-Brasil

CLEVERSON ABREU TEOTONIO

Marília, 2012

CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA
FUNDAÇÃO DE ENSINO "EURÍPIDES SOARES DA ROCHA"
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Implementação de uma API Java de geração de assinaturas digitais de acordo com as regulamentações da ICP-brasil

Trabalho de conclusão de curso apresentado ao Curso de Bacharelado em Ciência da Computação, do Centro Universitário Eurípides de Marília, como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Me. Rodolfo Barros Chiaramonte.

Marília, 2012

CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

TRABALHO DE CONCLUSÃO DE CURSO – AVALIAÇÃO FINAL

Cleverson Abreu Teotonio

IMPLEMENTAÇÃO DE UMA API JAVA DE GERAÇÃO DE
ASSINATURAS DIGITAIS DE ACORDO COM AS REGULAMENTAÇÕES DA ICP-BRASIL

Banca examinadora da monografia apresentada ao Curso de Bacharelado em Ciência da Computação do UNIVEM/F.E.E.S.R., para obtenção do Título de Bacharel em Ciência da Computação.

Nota: 9 (Nove)

Orientador: Rodolfo Barros Chiaramonte

1°. Examinador: César Giacomini Penteado

2°. Examinador: Fábio Dacêncio Pereira

Marília, 04 de dezembro de 2012.

Agradecimentos

Um trabalho de conclusão de curso representa muitos mais que o trabalho desenvolvido no ultimo da graduação. Ele é resultado de todos os anos de estudos do curso e também de todo conhecimento anteriormente adquirido. Nesse sentido gostaria de agradecer a todos que possibilitaram meu ingresso no curso de Bacharelado em Ciência de Computação, em especial à unidade de Marília do Centro Paula Souza e a todos seus alunos e professores, pois foram fundamentais na minha opção profissional pela computação.

Gostaria de agradecer também a todos que me auxiliaram durante a faculdade, de uma maneira ou de outra. Nesse sentido gostaria de agradecer aos meus avôs que ofereceram ajuda financeira fundamental, aos meus pais por todos os auxílios e mordomias que tornaram esses anos de estudo muito mais fáceis, a minha irmã por todas as conversas e conselhos, aos meus amigos que seja nas horas de diversão ou nas horas de estudo nunca decepcionaram e também a todos os professores e profissionais do UNIVEM.

Por fim, gostaria de agradecer nominalmente a todos que contribuíram diretamente para o desenvolvimento desse trabalho: Adilson Eduado Guelfi, Fábio Dacêncio Pereira, Rodolfo Barros Chiaramonte, Ettore Leandro Tognoli, Leandro Yukio Mano Alves e também a LSI-TEC, idealizadora e patrocinadora do trabalho realizado.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

AC	Autoridade Certificadora
AD	Assinatura Digital
AR	Autoridade Registradora
API	Application Programming Interface
ASN1	Abstract Notation One
BC	Bouncy Castle
BER	Basic Encoding Rules
BES	Basic Electronic Signature
CMS	Cryptographic Message Syntax
CAdES	CMS Advanced Electronic Signatures
DER	Distinguished Encoding Rules
ECC	Elliptic Curve Cryptography
EF	Entidade Final
EPES	Explicit Policy-based Electronic Signature
ES	Electronic Signature
ETSI	European Telecommunications Standards Institute
IAIK	Institute for Applied Information Processing and Communication
ICP	Infraestrutura de chaves públicas
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
ITI	Instituto Nacional de Tecnologia da Informação
JDK	Java Development Kit

JRE	Java Runtime Environment
KPRI	Private Key
KPUB	Public Key
LCR	Lista de Certificados Revogados
OSDT	Oracle Security Developer Tools
PEM	Privacy-Enhanced Mail
PKCS	Public-Key Cryptography Standards
PKI	Public Key Infrastructure
RA	Referências de Arquivamento
RB	Referências Básicas
RC	Referências Completas
RFC	Request For Comments
RT	Referências de Tempo
RV	Referências de Validação
TSA	Timestamp Authority

Resumo

Desde os primórdios o homem registra por meio de escrita, mesmo que rudimentar, a autoria de uma obra ou sua propriedade por meio de uma assinatura. No âmbito eletrônico a correspondente da assinatura manual é a assinatura digital. Esse trabalho trata da geração de assinaturas digitais, de acordo com as normas que regulamentam a mesma. Tem-se por objetivo a implementação de uma API Java com capacidade de gerar assinaturas digitais de acordo com as normas e regulamentações da ICP-Brasil. Para tanto se deve primeiramente estudar as regulamentações da ICP-Brasil e todas as regulamentações por ela referenciada, direta ou indiretamente; e, posteriormente, implementar as funcionalidades pertinentes de acordo com estes requisitos.

Abstract

Since the beginning man through written records, even if rudimentary, the authorship of a work or their property through a subscription. Under the corresponding electronic signature is the digital signature manual. This paper deals with the generation of digital signatures in accordance with the rules governing the same. It has been engaged in the implementation of a Java API capable of generating digital signatures in accordance with the rules and regulations of the ICP-Brazil. To do so one must first study the regulations of ICP-Brazil and all regulations referenced by it, directly or indirectly, and subsequently implement the relevant features according to these requirements.

Introdução

Desde os primórdios o homem registra por meio de escrita, mesmo que rudimentar, a autoria, ou propriedade, de uma obra por meio de uma assinatura. A palavra assinatura tem origem no latim "*assignare*", que significa afirmar, fazer verdadeiro o que está escrito antes. A assinatura está presente nos atos cotidianos como: certidões, contratos, cheques, registros e cartas.

Com o avanço tecnológico a assinatura manuscrita foi concebida no meio digital, e permitiu a identificação da autoria, coautoria, consenso sobre o conteúdo de um documento digital. Apesar da analogia com a assinatura manuscrita, a assinatura digital é elaborada e validada por sistemas computacionais, utilizando técnicas matemáticas e algoritmos criptográficos. A integração com outras soluções como o certificado digital permitiu não só a garantia de autenticidade, mas também outros serviços como a integridade e o não repúdio sobre um documento digital.

A assinatura digital é definida por Stallings (2008, p. 272) como "[...] mecanismo de autenticação que permite ao criador de uma mensagem anexar um código que atue como uma assinatura.", e segundo a ICP-Brasil (2010, p. 4) "[...] esse tipo de assinatura possui o mesmo valor de uma assinatura manuscrita". Tem, portanto, caráter e valor jurídico. Deve, por conseguinte, ser muito bem definida e documentada, afim de que cumpra suas pretensões. Estando no Brasil, existem duas instituições normativas pertinentes: a ETSI e a ICP-Brasil.

O presente trabalho, retratado nessa monografia, é parte formadora de um *software* de assinatura digital. Esse software completo é capaz de realizar assinaturas digitais, criar o CMS (*Cryptographic Message Syntax*, explicado na seção 2.2), validar certificados e assinaturas digitais, e possui um carimbador de tempo próprio (Guelfi, 2012). O desenvolvimento desse software foi dividido em três partes, sendo elas: criação de uma API de assinatura (retratado nesse trabalho), criação de uma API de validação de CMSs e certificados digitais e criação de um servidor de carimbo de tempo. A Figura 1 ilustra, com alto grau de abstração, o funcionamento desse *software*. É importante ressaltar que, na Figura 1, o objetivo desse trabalho é retratado na entidade "API Java de geração de CMSs".

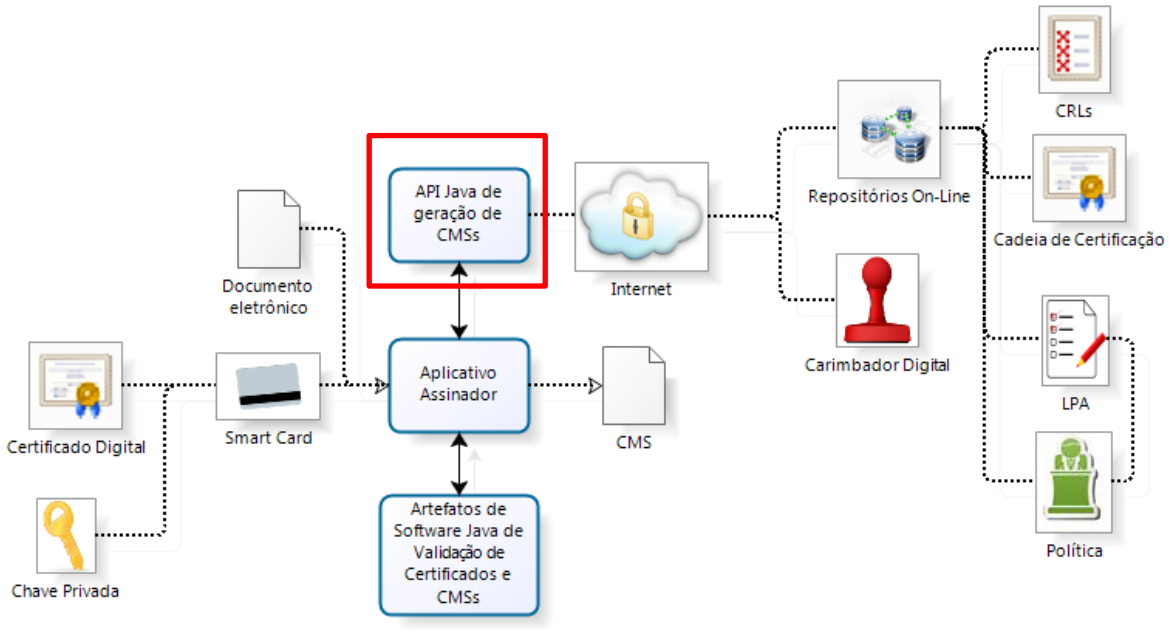


Figura 1 - Fluxo básico do aplicativo assinador

Este software foi desenvolvido em Java utilizando as bibliotecas criptográficas e a API de desenvolvimento Bouncy Castle (BC). A demonstração do funcionamento desse software e de seus componentes não faz parte do escopo desse projeto. A partir de agora discorrer-se-á, exclusivamente, sobre os detalhes de funcionamento e de implementação da API Java de geração de CMSs.

Problemática

A automatização de procedimentos e a crescente necessidade de resolver cada vez mais processos via internet demandou uma maneira de se assinar documentos eletronicamente. Porém, apesar da grande demanda, ainda é difícil encontrar ferramentas que realizem assinaturas digitais de acordo com as normas brasileiras, regulamentadas pela ICP-Brasil. Sendo assim, o presente trabalho propõe o desenvolvimento de uma ferramenta que realize assinaturas digitais de acordo com as normas ICP-Brasil.

Trabalhos Correlatos e Justificativa

Existem diversos trabalhos no âmbito da assinatura digital. Esses estão em grande parte relacionados aos profissionais de direito, principalmente no que tange a trabalhos acadêmicos. Existem diversas ferramentas de software nesse domínio, desde as APIs de desenvolvimento às aplicações que desenvolvem as assinaturas digitais em si, voltadas ao usuário final.

Tanto em relação às APIs quanto em relação às aplicações é possível afirmar que ainda há uma predominância dos padrões internacionais, que são os regulamentados pelo ETSI, sobre os padrões nacionais regulamentados pela ICP-Brasil. Isso pode dever-se tanto ao tempo da existência da regulamentação quanto ao transtorno de se migrar sistemas que já funcionavam satisfatoriamente no padrão internacional para o padrão nacional.

As aplicações disponíveis atualmente estão em sua maioria relacionadas a empresas privadas, que oferecem as mesmas em forma de produtos. Existem também muitas relacionadas a organizações, governamentais e não governamentais, que as oferecem, gratuitamente ou não, para determinados nichos ou publicamente.

Na presente pesquisa não foi encontrada API de desenvolvimento de assinaturas digitais concebidas exclusivamente para o padrão brasileiro de assinaturas digitais, mas existem diversas concebidas de forma genérica de modo a permitir o desenvolvimento de assinaturas em qualquer padrão, contanto que esse tenha se embasado nas regulamentações internacionais do IETF (Internet Engineering Task Force). Ainda assim, é possível notar nessas APIs a ênfase nos padrões internacionais do ETSI (European Telecommunications Standards Institute), no qual boa parte das normas ICP-Brasil foram inspiradas ou mesmo referenciadas.

Dentre as APIs de desenvolvimento de assinaturas digitais existentes, é possível citar:

a) A API Java, que oferece já de forma nativa diversas ferramentas para o desenvolvimento de assinaturas digitais;
b) A API OSDT (Oracle Security Developer Tools), um conjunto de bibliotecas Java para uso empresarial que fornecem rotinas que auxiliam no processo de

desenvolvimento de assinaturas digitais. É fornecido juntamente com determinados "pacotes" do Oracle 10g;

c) A API IAIK (Institute for Applied Information Processing and Communication), fornecida pelo instituto de mesmo nome da TU Graz (Graz University of Technology) como produto;
d) A API BC (Bouncy Castle), fornecida pela organização de mesmo nome gratuitamente e utilizada para o desenvolvimento da API aqui proposta.

Objetivo

O objetivo do trabalho é o desenvolvimento de uma API Java de suporte a geração de assinaturas digitais de acordo com as regulamentações da ICP-Brasil. A API tem por função oferecer um conjunto de rotinas que sejam capazes de gerar todos os cinco padrões de assinaturas previstos pela ICP-Brasil, porém, se necessário, o desenvolvedor poderá utilizar-se da API da maneira que lhe for mais conveniente, isto é, desenvolvendo e complementando as rotinas oferecidas. Além disso, o conhecimento obtido com o desenvolvimento da API deve ser utilizado de forma a contribuir com a evolução da infraestrutura brasileira de chaves públicas através de artigos.

Objetivos Específicos

a) A API de suporte ao desenvolvimento de assinaturas digitais deverá realizar operações criptográficas relacionadas à assinatura digital previstas nas regulamentações da ICP-Brasil. Conforme definido nos documentos: DOC-ICP-15, DOC-ICP-15.01, DOC-ICP-15.02 e DOC-ICP-15.03;
b) Para um mesmo documento eletrônico, possibilitar com que uma ou mais assinaturas digitais possam ser realizadas por um ou mais signatários (co-assinatura);

c) As assinaturas digitais produzidas pela API de suporte ao desenvolvimento de assinaturas digitais devem estar em conformidade com o padrão CMS Advanced Electronic Signature (CAdES), com o DOC-ICP-15.01 e com um dos seguintes formatos:
   - Assinatura Digital com Referência Básica (AD-RB);
   - Assinatura Digital com Referência do Tempo (AD-RT);
   - Assinatura Digital com Referências para Validação (AD-RV);
   - Assinatura Digital com Referências Completas (AD-RC);
   - Assinatura Digital com Referências para Arquivamento (AD-RA);

d) Desenvolver a API de suporte ao desenvolvimento de assinaturas digitais com capacidade de criação de todos os formatos de assinatura padronizados pela ICP-Brasil discriminados acima;

e) Desenvolver a API de suporte ao desenvolvimento de assinaturas digitais para a linguagem Java e totalmente compatível com o sistema operacional Microsoft Windows XP SP 3.

Materiais e Métodos

Primeiramente efetuou-se uma pesquisa das APIs e artefatos de software já existentes, a fim de facilitar o processo de desenvolvimento do assinador digital. Depois de selecionados, estes foram estudados e testados a fim de se obter o conhecimento necessário para sua utilização, e, também, verificar se possuem todas as funcionalidades necessárias para o desenvolvimento do projeto.

O estudo das normas ICP-Brasil, e das normas internacionais, foram feitos paralelamente ao desenvolvimento do software e suas funcionalidades, de modo e agilizar o processo de desenvolvimento. Tinha-se por objetivo o desenvolvimento dos cinco tipos de assinatura ICP-Brasil, além das funcionalidades previstas, que foram desenvolvidos na ordem discriminada abaixo:

a) Assinatura Digital com Referência Básica (AD-RB);

b) Assinatura em lote;

c) Co-assinatura;

d) Assinatura Digital com Referência do Tempo (AD-RT);

e) Assinatura Digital com Referências para Validação (AD-RV);

f) Assinatura Digital com Referências Completas (AD-RC);

g) Assinatura Digital com Referências para Arquivamento (AD-RA).

O desenvolvimento foi efetuado em linguagem de programação Java, JRE e JDK versão 1.7.0.20.13, utilizando o ambiente de desenvolvimento NetBeans IDE 7.1, API Java BC-Provider e BC-PKI versão 1.47.

Organização do Trabalho

O seguinte trabalho está organizado de modo a oferecer um completo entendimento da ferramenta aqui retratada. Portanto, todas as informações teóricas e técnicas necessárias para a perfeita compreensão do funcionamento e modo de utilização da API estão descritas nesse documento.

Os capítulos foram organizados de modo a fornecer o conhecimento prévio necessário, antes da discussão da API em si, de modo simples. A seção 1 discorre sobre os conceitos teóricos da criptografia que resultaram na assinatura digital, além de uma introdução ao próprio conceito de criptografia. Diversos artefatos dessa natureza são apresentados, definidos e explicados, de modo conceitual.

A seção 2 discorre sobre a infraestrutura de chaves públicas. Nessa seção são apresentadas as principais normas internacionais e suas respectivas instituições normalizadoras. É apresentada, também, a ICP-Brasil, suas normas e regulamentos, como essas foram feitas, e em que foram inspiradas. Além de se explicar como os conceitos, introduzidos na seção 1, são utilizados na prática.

A seção 3 retrata o processo de desenvolvimento da API, sua estrutura, organização, funcionamento, as principais dificuldades no desenvolvimento, peculiaridades da implementação, resultados alcançados e testes. Por fim, no capítulo 4, são apresentadas as conclusões, apreciação do trabalho efetuado, análise da API obtida, e trabalhos futuros.

1. Conceitos Básicos de Criptografia

Esta seção discorre sobre os conceitos teóricos da criptografia que resultaram na assinatura digital, além de uma introdução ao próprio conceito de criptografia. Diversos artefatos dessa natureza são apresentados, definidos e explicados de modo conceitual.

1.1. Criptografia

Criptografia, do grego: *kryptós* (escondido), e *gráphein* (escrita); é a ciência que usa um conjunto de conhecimentos, principalmente matemáticos, aplicados de modo a codificar informações de maneira secreta. Tornando possível que só pessoas autorizadas tenham acesso ao conteúdo codificado, pois conhecem a codificação ou possuem artefatos que possibilitam isso.

Segundo Trinta e Macêdo (1998, p. 2) "A criptografia representa a transformação de informação inteligível numa forma aparentemente ilegível, a fim de ocultar informação de pessoas não autorizadas, garantindo privacidade".

Na criptografia o processo de codificação, mencionado acima, tem o nome de cifração e o processo de descodificação é nomeado decifração. É importante usar e respeitar os termos corretos a fim de evitar ambiguidade e erros de interpretação. Também é possível citar que "conhecer a codificação" seria conhecer o funcionamento do algoritmo criptográfico (um que não utilize chaves) e "possuir artefatos que possibilitam isso" seria possuir a chave do algoritmo criptográfico.

Os primeiros algoritmos criptográficos não faziam uso de chave do processo de cifração. Eles utilizavam algoritmos que tornavam a informação de entrada incompreensível, no processo de cifração. Para decifrar, aplicava-se um algoritmo que efetuava o processo inverso ao efetuado pelos algoritmos cifradores, no processo de decifração. A segurança do algoritmo dependia do fato do funcionamento do mesmo ser mantido secreto. Uma vez descoberto seu funcionamento, toda informação cifrada por ele estava insegura.

Porém, não é sábio atrelar a segurança de um algoritmo criptográfico ao fato de seu funcionamento ser secreto, pois como a historia já provou, manter secreto o funcionamento de um algoritmo é praticamente impossível; como diz Burnett (2002, p. 18) não importa como, eles sempre descobrem o funcionamento do algoritmo. Por esse motivo algoritmos que não fazem uso de chaves, são pouco utilizados.

Os algoritmos que utilizam chave não necessitam ser secretos, de fato a grande maioria deles estão publicados e disponíveis para qualquer um que tenha interesse. A segurança de um algoritmo criptográfico desse tipo está na chave, essa sim deve ser protegida com todos os recursos disponíveis. Dentre os algoritmos que utilizam chave existem dois tipos, são eles:

a) Algoritmos de criptografia de chave simétrica;

b) Algoritmos de criptografia de chaves assimétricas.

1.2. Gerenciamento de Chaves

Uma vez que a segurança da informação é dependente da qualidade e confidencialidade da chave, essa deve ser gerenciada com muito cuidado. A qualidade e confidencialidade da chave é um grande problema. Uma das primeiras dificuldades é gerar a chave, que deve ser resistente a ataques de todos os tipos, mas principalmente ao de força bruta. O ataque de força bruta consiste em adivinhar a chave tentando todas as possibilidades possíveis. Para gerar a chave geralmente são utilizados algoritmos específicos, que utilizam-se de informações randômicas para gerar uma boa chave.

Mas, uma vez gerada a chave, ainda tem o problema de armazená-la, uma vez que uma boa chave não é uma informação facilmente memorizável, e é tão importante que não pode ser confiada a memória de ninguém. Nesse caso uma das opções mais seguras é a utilização de um Token. Um dispositivo computacional com proteção contra ataques, que serve apenas para armazenar a chave.

Ele é mais seguro por estar protegido de ataques remotos, pelo menos enquanto não estiver conectado a um computador. Além disso, possui defesas contra ataques físicos e pode ser carregado para qualquer lugar que seu detentor quiser. Ele requer senha para a

disponibilização da chave, essa sim pode ser confiada à memória de alguém. O Token pode ser qualquer dispositivo computacional com as características mencionadas anteriormente. Um Token muito popular é o Smart Card, que é a tecnologia dos cartões de banco.

Outro problema é a distribuição das chaves, isto é, a entrega da mesma a seu detentor; pois muitas vezes o processo de geração e utilização da chave é feito por pessoas distintas e muito distantes geograficamente. Uma solução para esse problema é distribuição da chave por uma rede segura, porém devido à dificuldade de uma rede com tais características, essa é uma medida pouco utilizada.

1.3. Criptografia de Chave Simétrica

A criptografia de chave simétrica consiste em (Figura 2): no processo de cifração, dada uma *stream* de entrada (M), aplicar um algoritmo criptográfico sobre essa (M), utilizando uma chave (K), produzindo uma *stream* de saída cifrada (C). No processo de decifração aplicar sobre a *stream* de saída (C), o mesmo algoritmo criptográfico e utilizando a mesma chave (K), que foram utilizados na cifração, obtendo a *stream* de saída (M), a mesma da entrada (E).

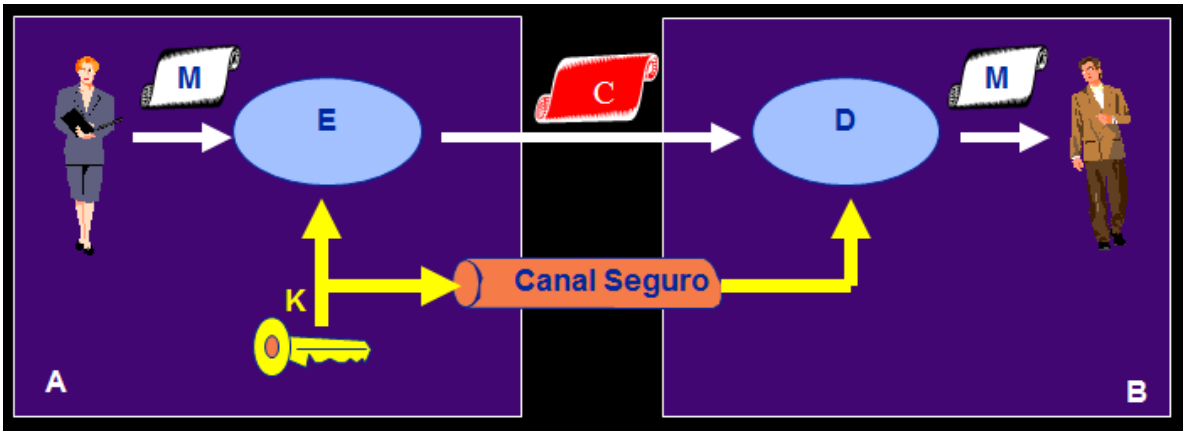


Figura 2 - Ilustração do processo de cifração/decifração com chave simétrica
Fonte: GUELFI, 2012b, p. 4

O algoritmo criptográfico simétrico possui propriedades especiais que permitem aplicar exatamente o mesmo algoritmo e a mesma chave no processo de decifração que são utilizadas no processo de cifração. Uma vez que não é pertinente a definição rigorosa do

processo de cifração/decifração no presente trabalho, esse processo será apenas demostrado intuitivamente (Figura 3).

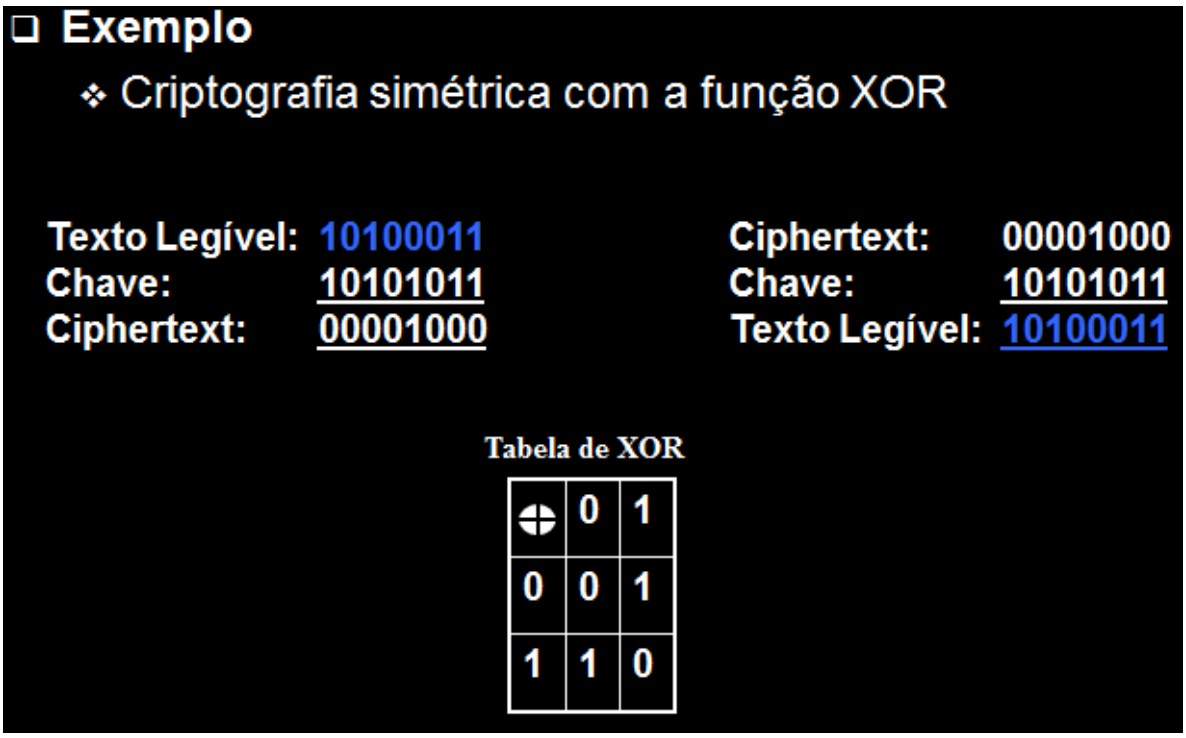


Figura 3 - Demonstração intuitiva do processo de cifração/decifração
Fonte: GUELFI, 2012b, p. 5

Graças a suas propriedades, a criptografia de chave simétrica tem várias vantagens: a complexidade computacional do algoritmo é baixa; o tempo de cifração/decifração é diminuto; porém, possui pelo menos um grande problema que é importante ressaltar, o processo de distribuição das chaves é crítico. Repare que na Figura 2, a chave (K) teve de ser transmitida do emissário (E) para o destinatário (D) por uma rede segura (Canal Seguro).

Northcutt (2002, p. 618) discorre a respeito de algoritmos simétricos, definindo-os, abaixo:

> [...] chave simétrica, é um método de criptografia que utiliza o mesmo valor de chave para codificar e decodificar. Seu uso considera que tiveram tempo antecipado para trocar uma chave secreta com segurança, que ninguém mais conhece. Esse valor de chave é, então, utilizado para a codificação das informações que são trocadas. Esse meio de criptografia pode ser rápido, porque a matemática necessária para criar o texto cifrado a partir de uma chave secreta compartilhada não precisa ser tão complexo quanto o tipo usado com algoritmos assimétricos A maior dificuldade

> do algoritmo simétrico é que é difícil trocar chaves remotamente ou iniciar uma troca simétrica com um parceiro desconhecido e autenticar que essa pessoa é quem diz ser. Como você poderia dar sua chave a um parceiro remoto se ele ainda não a tiver? Você precisaria ter um canal seguro para passar a chave. Como você provavelmente não tem esse canal (de outra forma, não precisaria passar a chave), a situação vira um impasse.

A tabela abaixo mostra os principais algoritmos simétricos e seus respectivos: tipos, tamanhos da chave, e, tamanhos de bloco.

Tabela 1 - Algoritmos de chave simétrica (1998)

Fonte: GUELFI, 2012b, p. 9

Nome	Tipo	Tam. chave	Tam. bloco
DES	bloco	56	64
Triple DES (2 ch.)	bloco	112	64
Triple DES (3 ch.)	bloco	168	64
IDEA	bloco	128	64
BLOWFISH	bloco	32 a 448	64
RC5	bloco	0 a 2040	32,64,128
CAST-128	bloco	40 a 128	64
RC2	bloco	0 a 1024	64
RC4	stream	0 a 256	-
Rijndael (AES)	bloco	128,192,256	128, 192, 256
MARS	bloco	variável	128
RC6	bloco	variável	128
Serpent	bloco	variável	128
Twofish	bloco	128,192,256	128

1.4. Criptografia de Chave Assimetrica

A criptografia de chave assimétrica consiste em (Figura 4): no processo de cifração, dada uma *stream* de entrada (M), aplicar um algoritmo criptográfico sobre essa (M), utilizando uma chave de cifração (K_E), produzindo uma *stream* de saída cifrada (C). No processo de decifração, aplicar sobre a *stream* de entrada (C), o mesmo algoritmo criptográfico, que foi utilizado na cifração, utilizando a chave de decifração (K_D), e obter a *stream* de saída (M).

Figura 4 - Ilustração do processo de cifração/decifração com chave assimétrica

Fonte: GUELFI, 2012c, p. 4

A principal mudança em relação ao processo com chave simétrica é, como sugere o nome, a utilização de chaves distintas no processo com chaves assimétricas. Nesse sistema cada entidade possuirá um par de chaves, uma, a chave privada, deve ser mantida em segredo absoluto, a outra, a chave pública, pode e deve estar publicada, disponível a qualquer um que tenha interesse (Figura 5).

Figura 5 - Par de chaves no sistema de chaves assimétricas

Fonte: GUELFI, 2012c, p. 7

Uma característica muito pertinente, retratada na Figura 5, é que, em alguns algoritmos assimétricos, se pode utilizar qualquer uma das chaves (KPRI ou KPUB) em qualquer um dos processos (cifração ou decifração). Contanto que se respeite a utilização da chave parceira no processo complementar, como mostra a Figura 6.

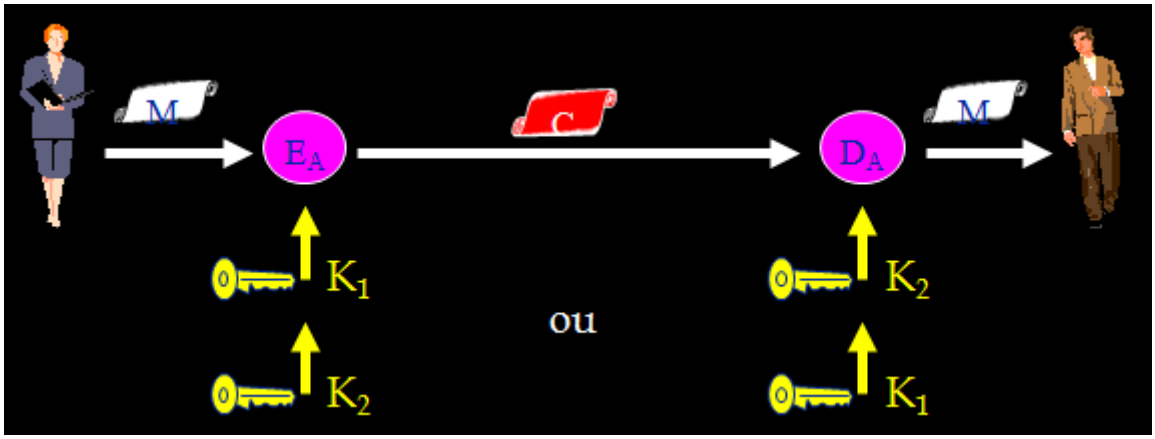


Figura 6 - Utilização do par de chaves assimétricas

Fonte: GUELFI, 2012c, p. 5

O uso do par de chaves possibilita muitas coisas até então impossíveis, porém, os algoritmos de criptografia assimétrica são computacionalmente complexos e consequentemente muito mais lentos, se comparados com os de criptografia simétrica. Segundo Mendes (2007, p. 23) "Na prática, os algoritmos criptográficos de chave pública (assimétrico) são entre 10 e 100 vezes mais lentos do que os equivalentes de chave secreta (simétrico)". Esse é um dos motivos da criptografia simétrica não ter se tronado obsoleta, hoje ela é utilizada em conjunto com a criptografia assimétrica em muitas aplicações.

É pertinente ressaltar dois algoritmos de criptografia de chave assimétrica utilizados na grande maioria das aplicações, são eles:

a) RSA (*Rivest-Shamir-Adelman*) teve seu algoritmo publicado em 1978, cujo patente expirou em 2000. Suas chaves têm tipicamente 512, 1024 e 2048 bits de tamanho;

b) ECC (*Elliptic Curve Cryptography*), baseado em curvas elípticas.

1.5. Funções de Resumo (hash)

Uma função de *hash* consiste em (Figura 7): um algoritmo que a partir de uma *stream* de entrada (M), faça um resumo dessa e como saída produza uma *stream* de saída de tamanho fixo (H(M)). As principais propriedades de um algoritmo de *hash* são:

a) Deve se dificultar ao máximo que a função de *hash* gere o mesmo resumo para duas mensagens distintas. Visto que temos um domínio infinito com uma imagem finita, é, consequentemente, impossível impedir que isso aconteça;

b) A alteração de qualquer bit na *stream* de entrada deve alterar drasticamente o valor da *stream* de saída de resumo.

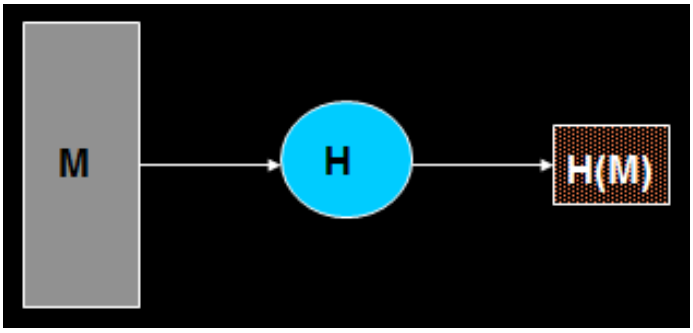


Figura 7 - Função de Hash
Fonte: GUELFI, 2012e, p. 6

Terada (2008, p. 213) define as funções de *hash*, referidas por ele como *funções de espelhamento*, como:

> [...] algoritmos E() que calculam um valor y = E(x) de comprimento relativamente menor que o comprimento de um texto legível x. Por exemplo, $|x|^1$ é da ordem de centenas de bytes como o nome e o endereço de uma pessoa, e |y| é da ordem de 128 bits, ou seja |y| << |x|. E() é chamada de *função de espelhamento* ou *função hashing*. O valor de y é chamado de resumo de x, ou também MAC (*Message Authentication Code*), ou HMAC, ou ainda *Message Digest* de x.

Moreno, Pereira e Chiaramonte (2005, p. 157) discorrem a respeito da utilidade dessas funções, segundo eles:

> Essas funções são utilizadas para garantir a integridade da mensagem, já que o objetivo delas é gerar um valor y diferente para cada mensagem. Com isso, caso a mensagem x seja modificada para uma x', quando o destinatário receber a mensagem x', terá de recalcular y para a mensagem que recebeu (x'). Como cada mensagem gera um y diferente, ele irá detectar que y para a mensagem que ele recebeu é diferente daquilo que esperava e, com isso, detectar que a mensagem foi alterada.

A tabela abaixo mostra os principais algoritmos de *hash* e seus respectivos: comprimentos, e, tamanhos.

Tabela 2 - Principais algoritmos de *hash*
Fonte: GUELFI, 2012e, p. 16

Algoritmo de Hash	Compr. Hash	kbytes/s
GOST Hash	256	11
MD4 - Message Digest 4	128	236
MD5 - Message Digest 5	128	174
N-HASH (12 rounds)	128	29
N-HASH (15 rounds)	128	24
RIPE-MD	128	182
RIPE-MD-160	160	---
SHA - Secure Hash Algorithm	160	75
SHA-2 Family (224, 256, etc)	----	----
SNEFRU (4 passos)	128	48
SNEFRU (8 passos)	128	23
WHIRLPOOL (ISO/IEC 10118-3:2004)	512	----

1.6. Serviços de Segurança

Utilizando os conceitos apresentados até agora, e mais alguns que não serão explicados aqui, por não fazerem parte do escopo desse trabalho, consegue-se a implementação de diversos serviços de segurança. Um serviço de segurança é uma funcionalidade relacionada à segurança computacional (GUELFI, 2012, p. 4 [1]). Os tópicos sequentes irão discorrer a respeito de alguns serviços de segurança.

1.6.1. Serviço de Confidencialidade

Esse serviço visa a proteger a informação contra divulgação não autorizada, isto é, apenas pessoas com privilégios para tal poderão visualizar o conteúdo da mensagem. Segundo Ferreira (2003, p.60):

> Confidencialidade significa proteger informações contra sua revelação a alguém não autorizado – interna ou externamente. Consiste em proteger a informação contra leitura e/ou cópia por alguém que não tenha sido explicitamente autorizado pelo proprietário daquela informação.

A Figura 8 retrata a seguinte situação: Bob envia uma mensagem para Alice via internet, como sabido, um meio de transmissão inseguro, no meio do caminho Carlos toma conhecimento do conteúdo das mensagens.

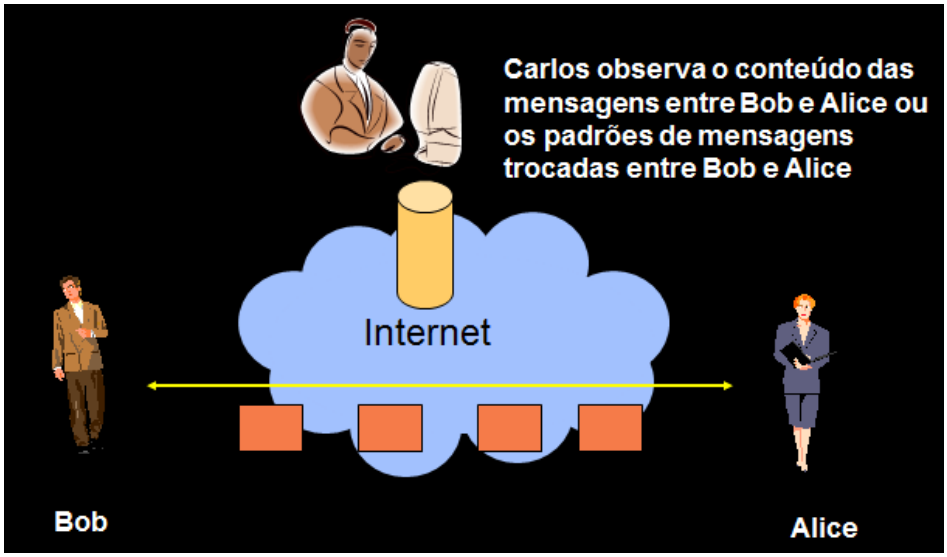


Figura 8 - Quebra de Confidencialidade

Fonte: GUELFI, 2012a, p. 7

A confidencialidade é implementada da seguinte maneira (Figura 9): a remetente A, cifra (E_A) a mensagem (M) com a chave pública ($KPUB_B$) do destinatário B. Para ler sua mensagem, o destinatário B, decifra (D_A) a mensagem cifrada (C) com a sua chave privada ($KPRI_B$).

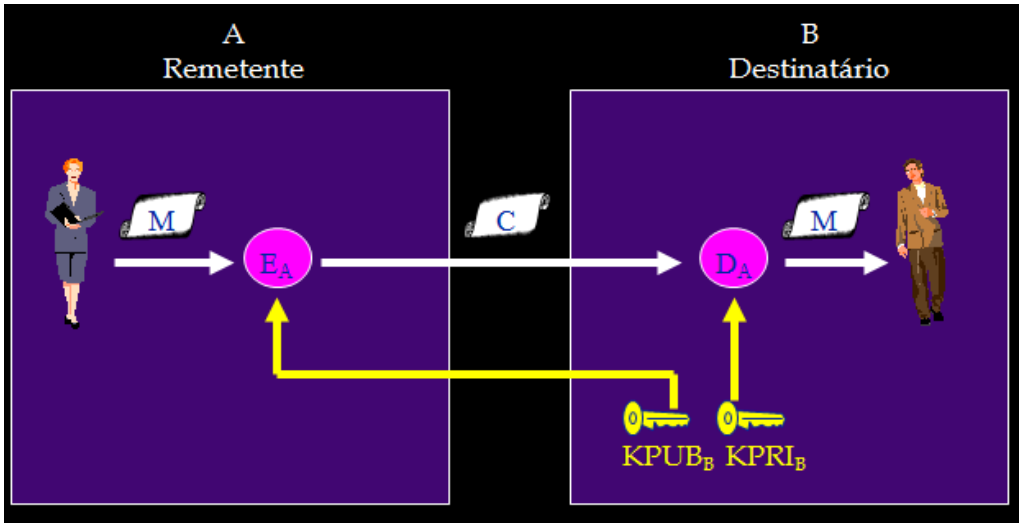


Figura 9 - Implementação da confidencialidade

Fonte: GUELFI, 2012c, p. 11

1.6.2. Serviço de Autenticação

Esse serviço visa garantir a identidade da pessoa ou entidade com que se está a manter comunicação e a respectiva autoria das mensagens da mesma. Isso pode ser dividido em três principais tipos de autenticação:

a) Autenticação de um usuário no sistema, isto é, garantir que o usuário que está a utilizar o sistema é quem diz ser;

b) Autenticação de parceiro de comunicação, isto é, garantir a identidade da pessoa ou entidade com que se está comunicando;

c) Autenticação de mensagem, também chamado de *serviço de autoria*, consiste em comprovar o autor de determinada mensagem ou documento.

Como diz Ferreira (2003, p.59):

> O serviço de autenticação em um sistema deve assegurar ao receptor que a mensagem é realmente procedente da origem informada em seu conteúdo. Normalmente, isto é implementado a partir de um mecanismo de senhas ou de assinatura digital. A verificação da autenticidade é necessária após todo processo de identificação, seja de um usuário para um sistema, de um sistema para o usuário ou de um sistema para outro sistema.

O Serviço de autenticação visa evitar a ocorrência de algumas situações, como as descritas a seguir:

a) Carlos envia mensagens para Alice como se ele fosse Bob (Figura 10);

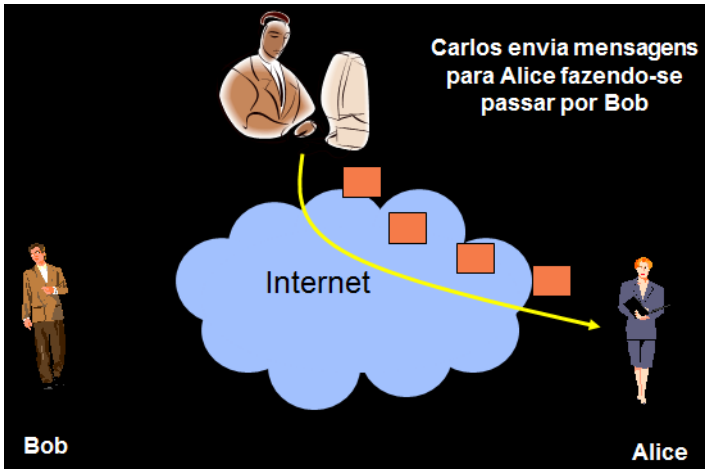


Figura 10 - Carlos passa-se por Bob

Fonte: GUELFI, 2012a, p. 10

b) Carlos intercepta as mensagens de Bob e depois as repassa para Alice, conhecido como ataque ativo de *Replay* (Figura 11).

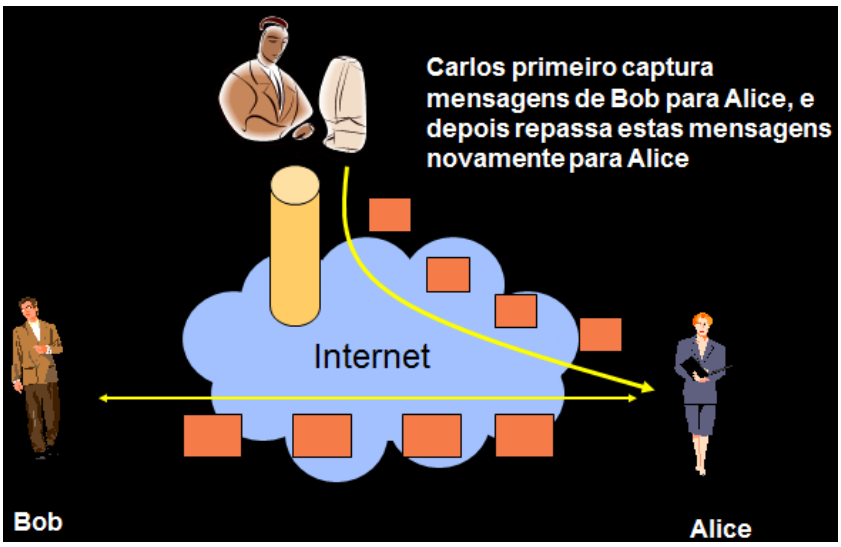


Figura 11 - Carlos captura as mensagens de Bob e as repassa para Alice

Fonte: GUELFI, 2012a, p. 11

A autenticação é implementada da seguinte forma (Figura 12): a remetente A cifra (E_A) a mensagem (M) com sua chave privada ($KPRI_A$). O destinatário B, para verificar a autenticidade da mensagem (C), decifra (D_A) a mensagem (C) com a chave pública da remetente A ($KPUB_A$).

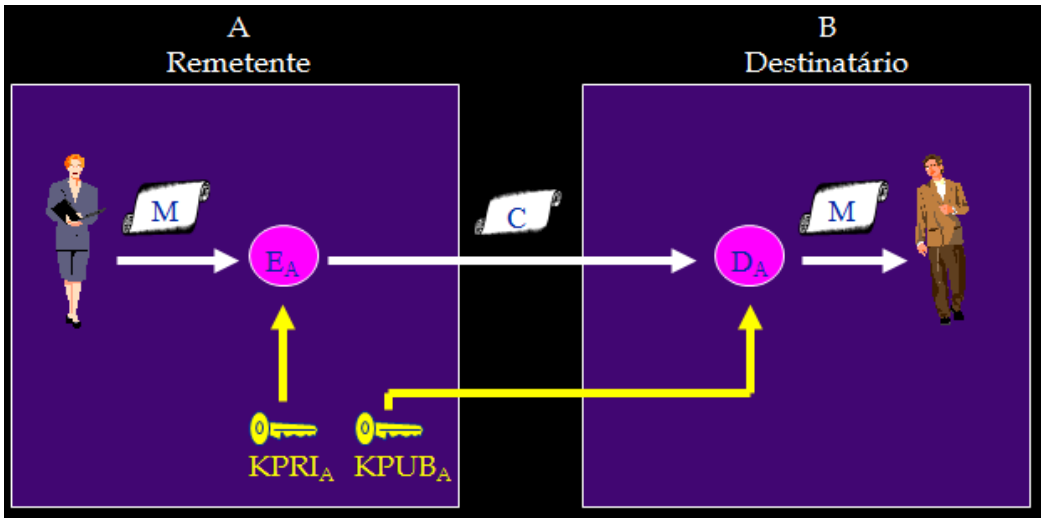


Figura 12 - Implementação da autenticação

Fonte: GUELFI, 2012c, p. 13

1.6.3. Serviço de Integridade

O serviço de integridade visa garantir que uma informação transmitida por um meio inseguro não foi alterada por uma entidade não autorizada. Ferreira (2003, p.59) define esse serviço como: “A integridade consiste em proteger a informação contra modificação sem permissão explicita do proprietário daquela informação. A modificação inclui ações como: escrita, alteração de conteúdo, alteração de status, remoção e criação de informações.”.

Esse serviço visa evitar a ocorrência de situações como a retratada na Figura 13. Nela Carlos intercepta e modifica as mensagens de Bob, e depois as reenvia para Alice, passando-se por Bob. Aqui são necessários dois serviços, o de autenticação e o de integridade. O serviço de integridade pode ser implementado através de uma assinatura digital.

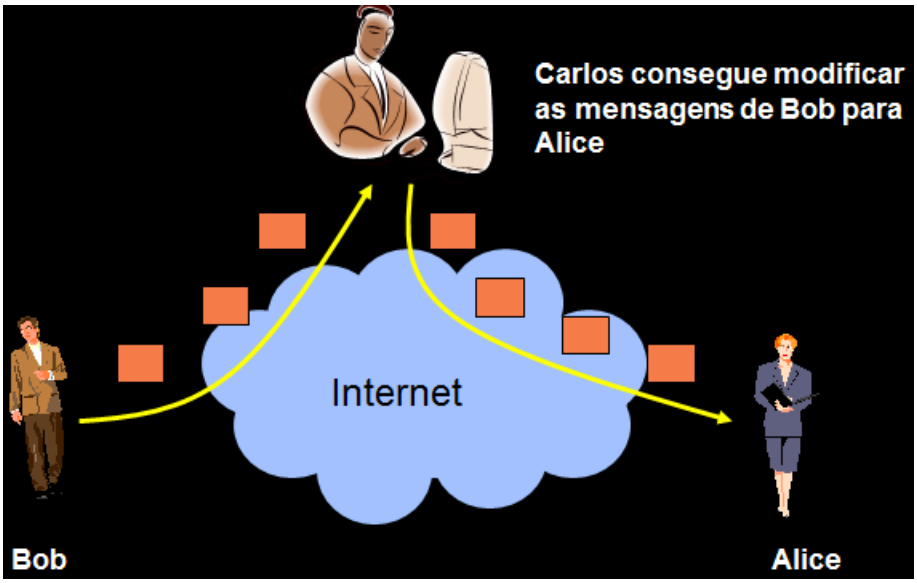


Figura 13 - Carlos modifica as mensagens enviadas por Bob para Alice

Fonte: GUELFI, 2012a, p. 12

1.6.4. Serviço de Irretratabilidade

O serviço de irretratabilidade, também conhecido como serviço de não repudiamento, visa impedir que uma pessoa ou entidade negue a produção, ou transmissão, de determinada informação, ou ainda, a execução de determinada ação. A irretratabilidade pode ser implementada com uma assinatura digital, estando essa obrigatoriamente relacionada a um certificado digital.

1.6.5. Serviço de Comprovação Temporal

O serviço de comprovação temporal visa garantir a existência de determinada informação depois de determinado ponto no tempo. Ele é implementado através de *Time Stamp Tokens*.

1.7. Solução do Problema de Distribuição de Chaves

No seção 1.2 foi apresentado o problema da distribuição de chaves, porém, não foi apresentada nenhuma solução satisfatória. Será então mostrada uma maneira de se distribuir chaves de algoritmos simétricos de forma segura e viável.

Para se enviar uma chave simétrica de maneira segura e viável é necessário o uso de chaves assimétricas. Como retrata a Figura 14, o processo consiste em cifrar (E_A) a chave simétrica (K_S), que foi gerada aleatoriamente, com a chave pública do destinatário (B). Para, só então enviá-la.

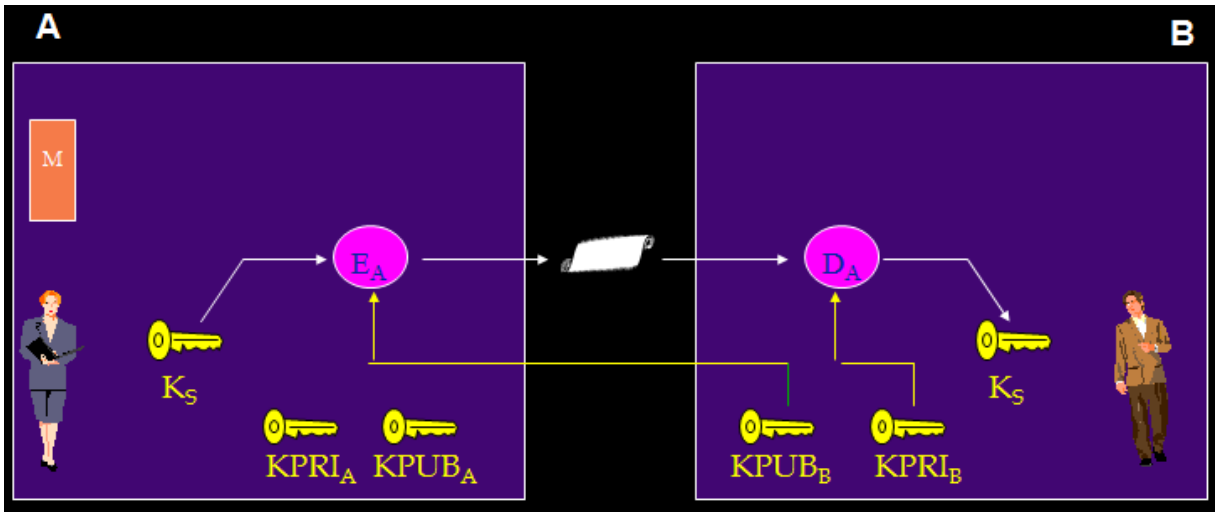


Figura 14 - Alice (A) envia chave simétrica para Bob (B)

Fonte: GUELFI, 2012d, p. 11

Feito essa distribuição, é possível transmitir uma mensagem usando criptografia simétrica como retratado na Figura 15. Uma vez que, poder-se-ia criptografar assimetricamente a mensagem e então enviá-la, parece desinteligência usar esse método.

Porém, uma vez que, o algoritmo simétrico é muito mais rápido em seu processo de cifração, a chave provavelmente será menor que a mensagem, e, uma vez distribuída á chave esse processo não precisa ser repetido, o método descrito na Figura 14 é, na maior parte das vezes, mais rápido.

.

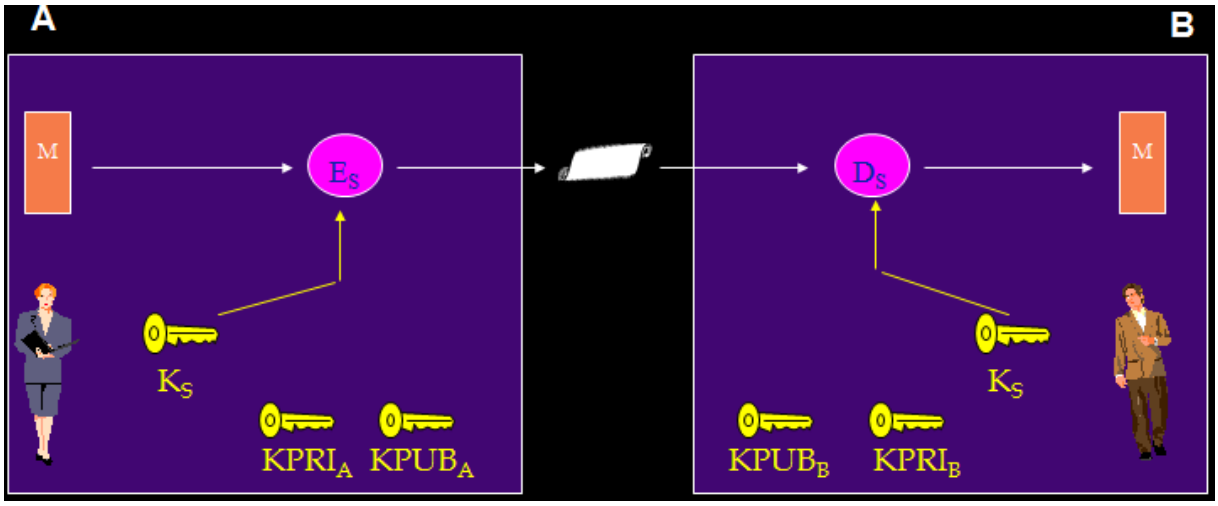


Figura 15 - Alice (A) envia mensagem para Bob (B)

Fonte: GUELFI, 2012d, p. 13

1.8. Assinatura Digital

A assinatura digital consiste em (Figura 16): a partir de uma mensagem (MA), gerar o hash da mesma (H(MA)), criptografar, com algoritmo de chave assimétrica (EA), o hash (H(MA)) com a chave privada (KPRIA) e compactar tudo (ASSA e MA) em um arquivo CMS. Para validar a assinatura o destinatário (entidade B), deve decifrar a assinatura (ASSA) com a chave pública (KPUBA), gerar o hash da mensagem (MA) e comparar ambos, se for igual, a assinatura é valida, caso contrário, é inválida.

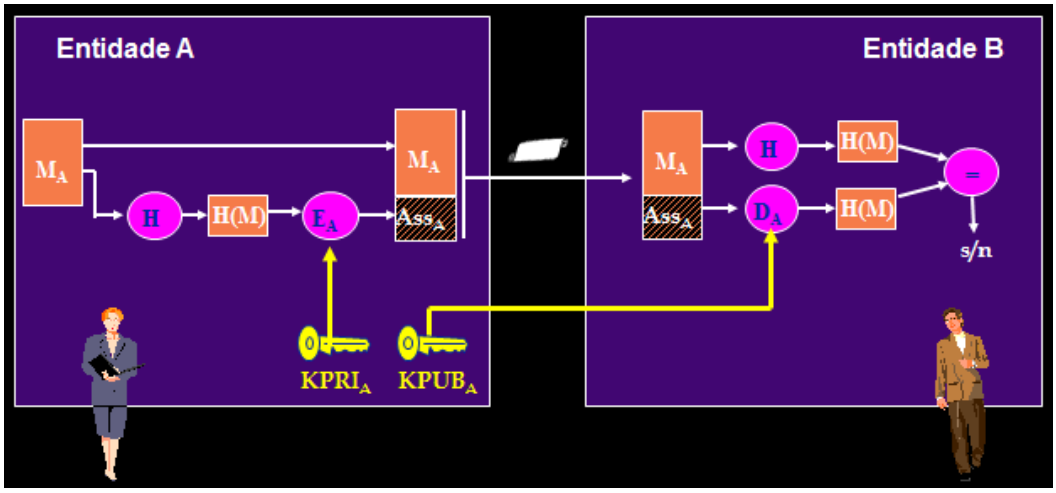


Figura 16 - Assinatura digital

Fonte: GUELFI, 2012f, p. 6

Moreno, Pereira e Chiaramonte (2005, p. 157) definem a assinatura digital da seguinte maneira:

> [...] uma assinatura digital é o criptograma resultante da cifração de um determinado bloco de dados (documento) pela utilização da chave chave-privada de quem assina em um algoritmo assimétrico. A verificação da assinatura é feita decifrando-se o criptograma (assinatura) com a suposta chave-pública correspondente, Se o resultado for válido, a assinatura é considerada válida, ou seja, autêntica, uma vez que apenas o detentor da chave privada, par da chave pública utilizada, poderia ter gerado esse criptograma.

1.9. Certificado Digital

O certificado digital é um documento eletrônico que possui informações de determinada pessoa ou entidade. Sua função é garantir um meio seguro de vinculação do par de chaves (chave pública e chave privada) com uma pessoa ou entidade e entre elas mesmas. Serve também como meio confiável de obtenção da chave pública de um usuário. Os certificados digitais são públicos e estão disponíveis a qualquer pessoa que tenha interesse, publicado nos repositórios padrões de suas respectivas *Autoridades Certificadoras*. A Figura 17 faz uma analogia entre um certificado digital e um documento de identidade.

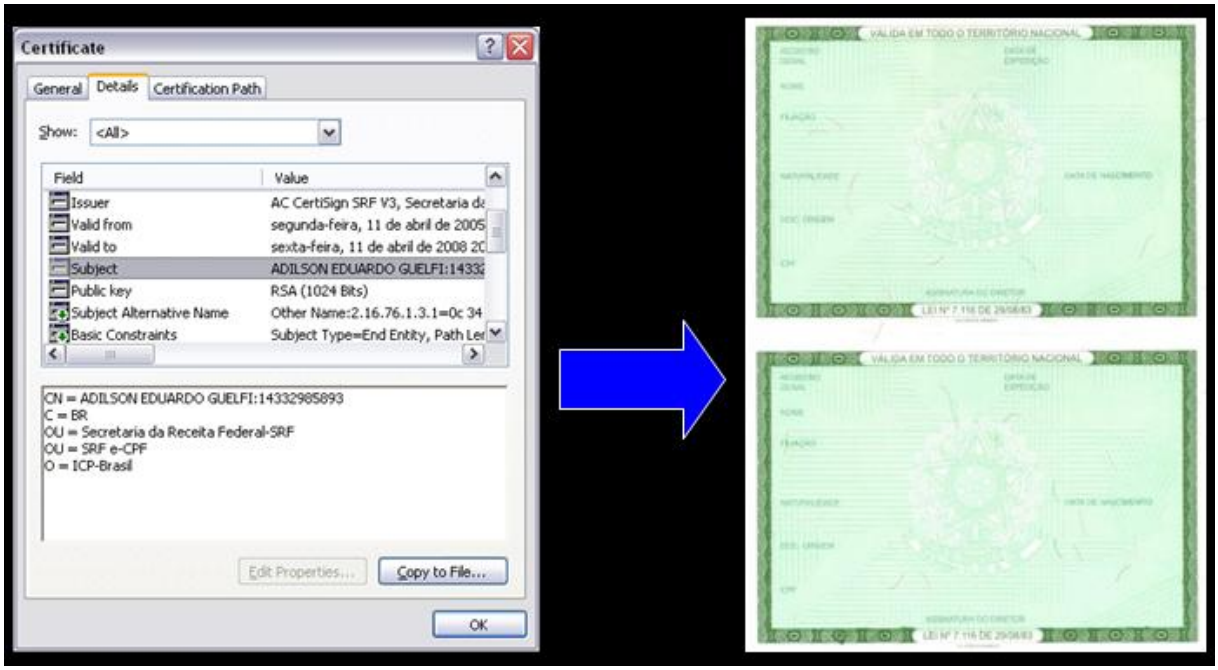


Figura 17 - analogia entre um certificado digital e um documento de identidade

Fonte: GUELFI, 2012h, p. 6

Friedrich e Medina (2007, p. 2) definem um certificado digital da seguinte maneira:

> Um certificado digital é um documento que contém informações relativas ao seu usuário/proprietário (seja pessoa física, jurídica ou computador), entre elas, a sua chave pública e dados necessários para comprovar sua identidade e tambéminformações como versão, número de série e período de validade. Para garantir sua autenticidade e a veracidade dos dados, o certificado é assinado digitalmente pela autoridade que o emitiu, ligando oficialmente um usuário a uma chave pública. (Alecrim, 2005). Porém, a aceitação do certificado dependerá dos níveis de confiança dos usuários em relação às práticas e políticas da autoridade que o emitiu.

2. Infraestrutura de Chaves Públicas

Esse capítulo discorre sobre a infraestrutura de chaves públicas brasileira. Aqui é apresentada a ICP-Brasil, suas normas e regulamentos, como essas foram feitas, e em que foram inspiradas. São apresentadas também as principais normas internacionais e suas respectivas instituições normalizadoras. Além de se explicar como os conceitos, introduzidos na seção 1, são utilizados na prática.

2.1. Orgãos Regulamentadores

A IETF é uma grande comunidade internacional que visa a produção de padrões e regulamentações, pertinentes a arquitetura da internet, que possibilitem e facilitem a evolução, e perfeito funcionamento da mesma. As recomendação IETF são publicadas nos documentos intitulados RFCs (*Request for Comments*), disponíveis no site da mesma.

O ETSI produz padrões, aplicáveis no mundo inteiro, para tecnologias de informação e comunicação, incluindo rádio, *broadcast*, e tecnologias pertinentes à internet. Ele é reconhecido oficialmente pela União Europeia como Organização Europeia de Padrões. O ETSI é uma organização sem fins lucrativos com mais de setecentas organizações membros, de 62 países distintos, dos cinco continentes do globo.

2.2. Formatos e Codificações

A ASN1 é uma linguagem flexível de notação abstrata, que descreve regras para representar, codificar, transmitir e decodificar dados em telecomunicação e redes de computador. Ela é regulamentada pela *International Organization for Standardization* (ISO), *International Electrotechnical Commission* (IEC), e *International Telecommunication Union*. É usada para se especificar as estruturas de dados que representam os diversos artefatos aqui apresentados apenas de forma conceitual, como: certificado digital, assinaturas digitais, etc.

A ASN1 pode ser codificada de várias maneiras, abaixo estão descritas algumas delas:

a) *Basic Encoding Rules* (BER) – Define regras básicas de codificação ASN1, possui como principal vantagem a facilidade computacional de se codificar nesse formato, porém não possui tamanho fixo, além de oferecer mais de uma maneira de se codificar uma mesma informação;

b) *Distinguished Ending Rules* (DER) – Restringe as regras BER, oferecendo um código único para cada estrutura ASN1. A codificação DER, com seu código único e tamanho fixo, é muito útil para conteúdos grandes ou para determinadas técnicas de processamento. Porém, tem uma significativa desvantagem: é difícil codificar em DER (IETF RFC 5652, 2009, p. 6);

c) BASE 64 – Uma vez que tanto a codificação BER quanto a DER são binárias, e que alguns meios não aceitam a transmissão de arquivos binários, a ICP-Brasil utiliza a codificação BASE 64 como alternativa para essas situações. Trata-se de uma codificação em caracteres ASCII, com 64 caracteres, daí o nome;

d) PEM – Outra opção de codificação texto, utilizada principalmente para compatibilidade com outros sistemas.

O CMS é um padrão de mensagens assinadas. Derivou do PKCS #7 1.5, descrito na RFC 2315 (Atualmente RFC 5652). Originalmente desenvolvido pela RSA *Laboratories Technical Note* em novembro de 1993. Desde então a ITEF assumiu a responsabilidade de desenvolver e manter o CMS. O CMS é um *ContentInfo*, esse por sua vez tem a seguinte descrição ASN1:

```
ContentInfo ::= SEQUENCE {
        contentType ContentType,
        content [0] EXPLICIT ANY DEFINED BY contentType
}
```

O *contentType* deve especificar o tipo de conteúdo contido no *content*. Existem seis *contentTypes*, são eles: *data*, *signed-data*, *enveloped-data*, *digested-data*, *encrypted-data*, e *authenticated-data*. Cada um desses *contentTypes* apresentados tem uma função, uma aplicação, porém, devido ao escopo dessa pesquisa, aqui será discutido apenas o *signed-data*, que é usado para CMSs que representam assinaturas digitais.

Um CMS com *contentType signed-data* tem a seguinte descrição ASN1:

```
id-signedData OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs7(7) 2
}

SignedData ::= SEQUENCE {
        version CMSVersion,
        digestAlgorithms DigestAlgorithmIdentifiers,
        encapContentInfo EncapsulatedContentInfo,
        certificates [0] IMPLICIT CertificateSet OPTIONAL,
        crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,
        signerInfos SignerInfos
}
```

Com base no CMS o IETF definiu o CAdES que é um sistema avançado de assinaturas digitais. Ele é documentado na IETF RFC 5126 e na ETSI TS 101 733. Tem por função padronizar os diversos tipos de arquivos CMSs que podem ser construídos para assinatura digital através dos formatos. Os formatos definem, dentre os atributos assinados e não assinados, os que devem ser incluídos no *signerInfo*. Muito do que foi padronizado nesse documento, foi utilizado na padronização das assinaturas digitais da ICP-Brasil.

Os formatos de assinatura digital CAdES definidos pelo ETSI são mostrados nas figuras 18 à 26, com determinado nível de abstração:

a) Formato básico de assinatura (*Basic Electronic Signature* – CAdES-BES);

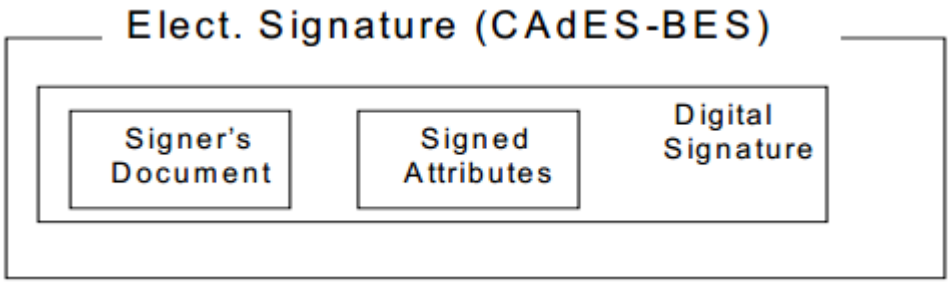


Figura 18 - CAdES-BES

Fonte: ETSI TS 101 733, 2008, p. 18

b) Formato de assinatura com política (*Explicit Policy-based Electronic Signatures* – CAdES-EPES);

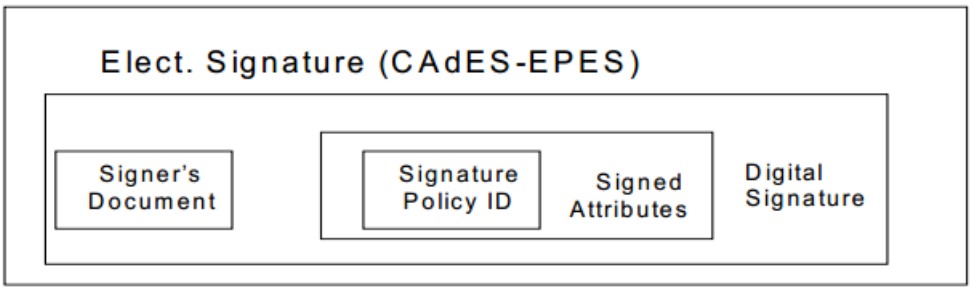


Figura 19 - CAdES-EPES

Fonte: ETSI TS 101 733, 2008, p. 18

c) Formato de assinatura com tempo (*Electronic Signature with Time* – CAdES-T);

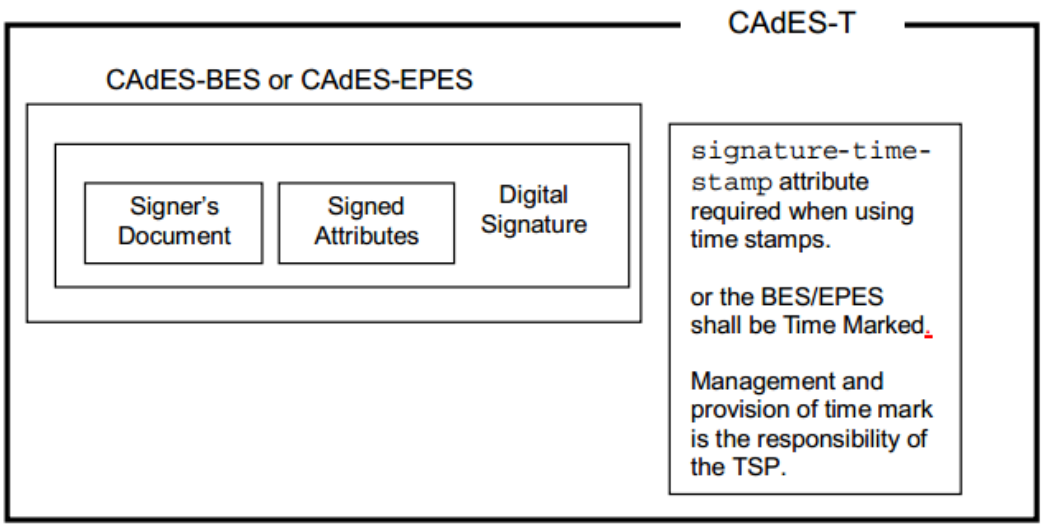


Figura 20 - CAdES-T

Fonte: ETSI TS 101 733, 2008, p. 20

d) Formato de assinatura com dados de referencias para validação (*ES with Complete Validation Data References* – CAdES-C);

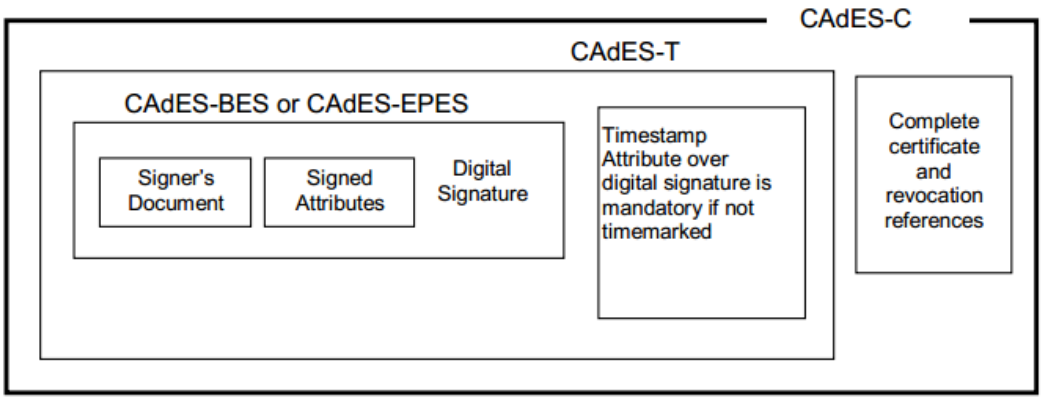


Figura 21 - CAdES-C

Fonte: ETSI TS 101 733, 2008, p. 20

e) Formato de assinatura estendido (*EXtended Long Electronic Signature* – CAdES-X Long), esse possui algumas variações como o tipo 1 (*EXtended Electronic Signature with Time Type 1* – CAdES-X Type 1), o tipo 2 (CAdES-X Type 2) e o com tempo (EXtended Long Electronic Signature with Time – CAdES-X Long Type 1 or 2).

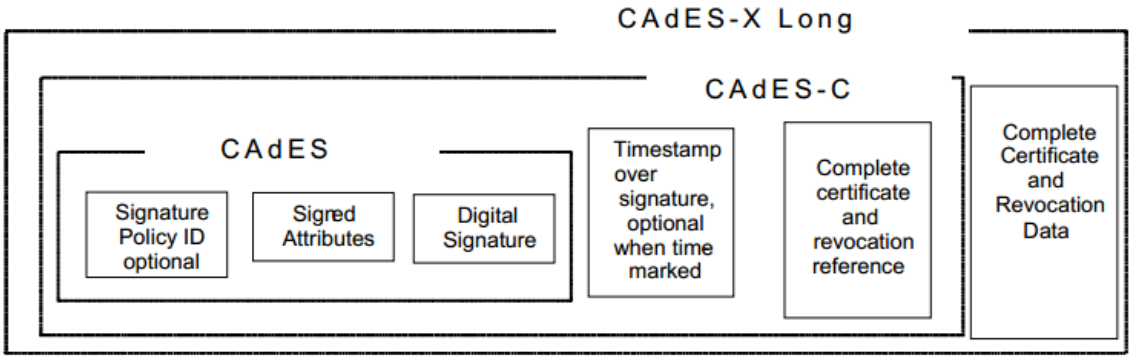


Figura 22 - CAdES-X Long

Fonte: ETSI TS 101 733, 2008, p. 22

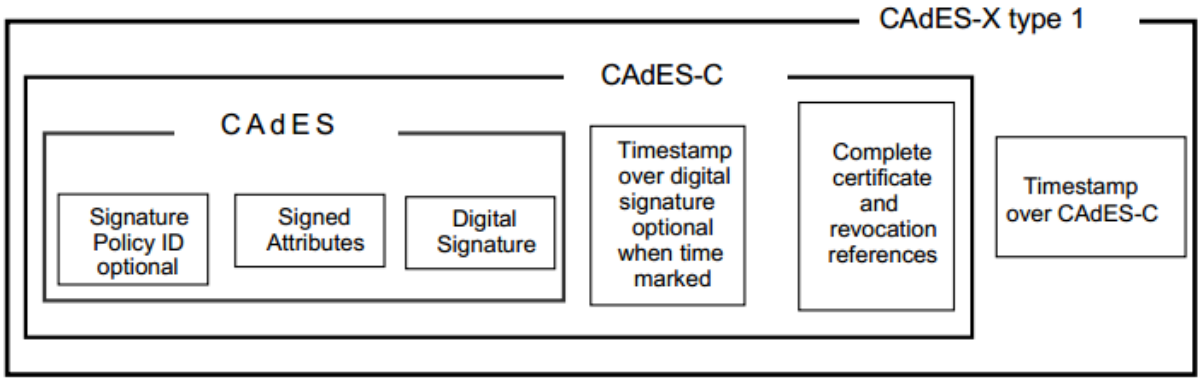


Figura 23 - CAdES-X Type 1

Fonte: ETSI TS 101 733, 2008, p. 22

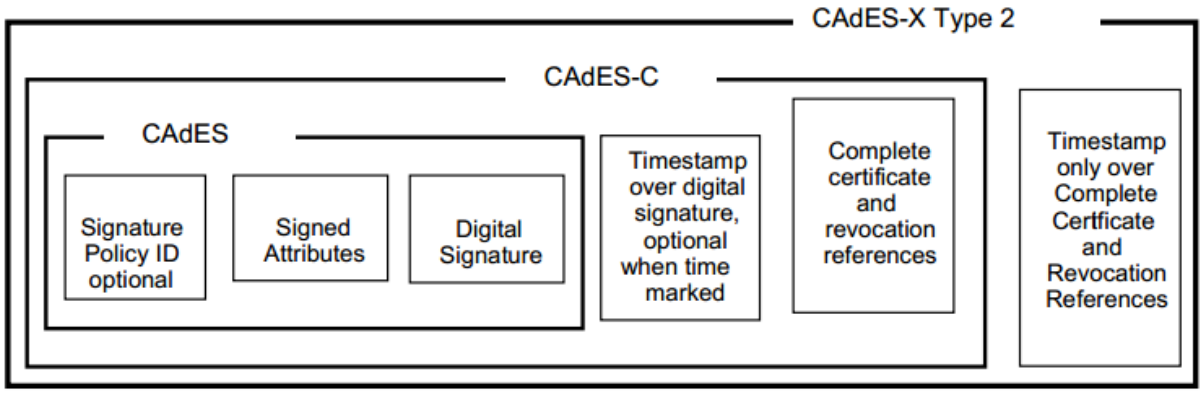


Figura 24 - CAdES-X Type 2

Fonte: ETSI TS 101 733, 2008, p. 23

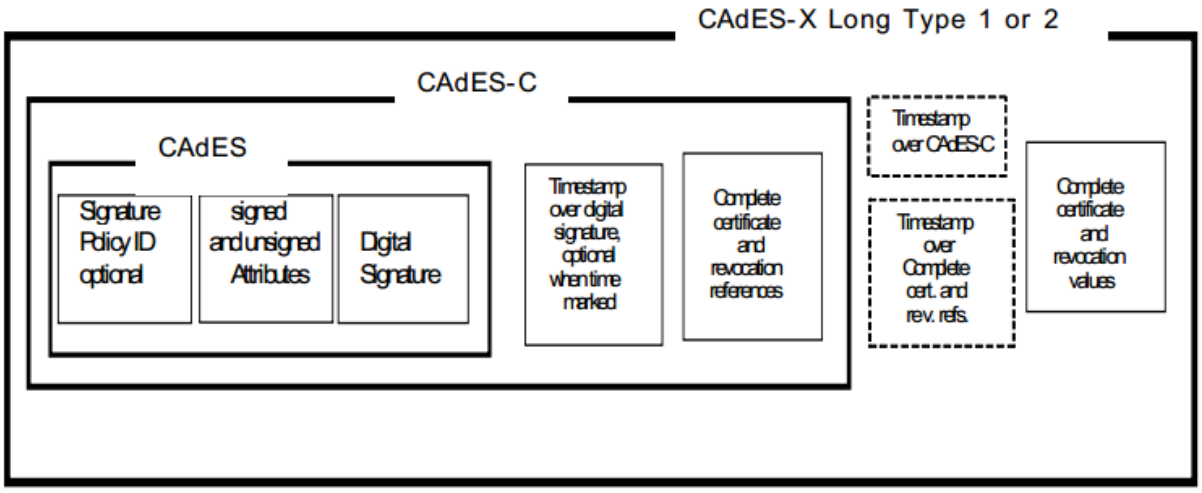


Figura 25 - CAdES-X Long Type 1 or 2

Fonte: ETSI TS 101 733, 2008, p. 23

f) Formato de assinatura para arquivamento (*Archival Electronic Signature* – CAdES-A).

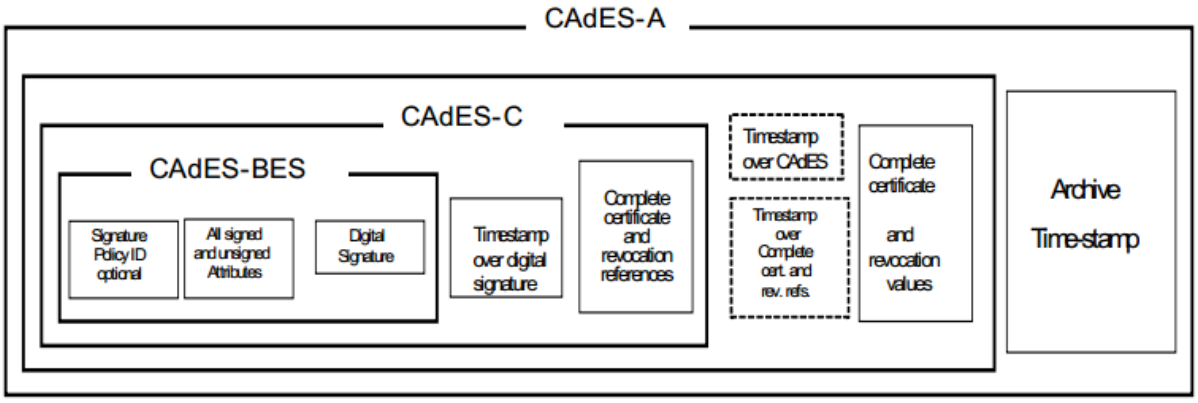


Figura 26 - CAdES-A

Fonte: ETSI TS 101 733, 2008, p. 23

2.3. ICP-Brasil

O Instituto Nacional de Tecnologia da Informação (ITI), em seu site oficial[1], se define como "[...] uma autarquia federal vinculada à Casa Civil da Presidência da República, cujo objetivo é manter a Infraestrutura de Chaves Públicas Brasileira - ICP-Brasil, sendo a primeira autoridade da cadeia de certificação - AC Raiz.".

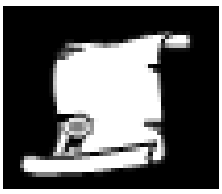
O ITI, em seu site oficial[2], define a ICP-Brasil como "[...] uma cadeia hierárquica e de confiança que viabiliza a emissão de certificados digitais para identificação virtual do cidadão.". Sua função é gerenciar o ciclo de vida dos certificados digitais e das chaves assimétricas, para tanto ela utiliza suas padronizações conhecidas como DOCs ICP.

Muitas das normas pertinentes à assinatura digital estão disponíveis tanto nas RFCs da IETF quanto nas regulamentações do ETSI, por esse último ser reconhecido oficialmente como órgão padrão, muitas das normas da ICP-Brasil referenciam as regulamentações do ETSI, que por sua vez, referenciam muitas RFCs do IETF.

[1] http://www.iti.gov.br/index.php/institucional/quem-somos
[2] http://www.iti.gov.br/index.php/icp-brasil/o-que-e

Antes de explicar o funcionamento de alguns dos processos gerenciados pela ICP-Brasil, é importante definir alguns componentes desses ciclos de vida, e os respectivos ícones com os quais estarão associados nos diagramas ilustrativos aqui utilizados. Abaixo estão relacionados os principais componentes:

Tabela 3 – Componentes do ciclo de vida da ICP-Brasil

Entidades	Entidade Final (EF)		Uma entidade final pode ser: um usuário humano, um servidor, uma aplicação ou uma organização. Ela é quem utiliza os recursos oferecidos pela ICP-Brasil.
	Autoridade Certificadora (AC)		Uma autoridade certificadora é uma entidade confiável para gestão do ciclo de vida dos certificados digitais, isso envolve: emitir certificados, renovar certificados, revogar certificados. Uma AC pode emitir dois tipos principais de certificados: certificado para EF ou certificado para uma AC subordinada.
	Autoridade Registradora (AR)		Uma autoridade registradora é uma entidade, opcional, subordinada à AC. Sua função é apenas manter repositórios de certificados emitidos e de LCRs. Sua justificativa é separar os processos mais críticos dos mais simples em uma AC.
Certificados Digitais			Seu conceito foi explicado na seção 1.9

Infraestrutura	Repositório de Certificados (RC)		Sua função é armazenar e tornar disponível os certificados digitais emitidos por uma AC.
	Lista de Certificados Revogados (LCRs)		Sua função é possibilitar a uma entidade qualquer verificar se um dado certificado digital foi revogado ou não, elas devem ser mantidas disponíveis pela AC e devem ser periodicamente atualizadas e publicadas pela mesma.

A Figura 27 ilustra a ICP-Brasil em determinado grau de abstração, como é possível reparar, nela estão retratados as ACs, ARs e EFs. Observando-a é possível visualizar o fluxo básico de várias operações.

Por exemplo: Para se gerar uma assinatura digital, a EF1 dispõe dos repositórios da AR para validar seu próprio certificado, através do repositórios de certificados e de CRLs. Além disso, é possível validar toda a cadeia de certificação usando os mesmos repositórios.

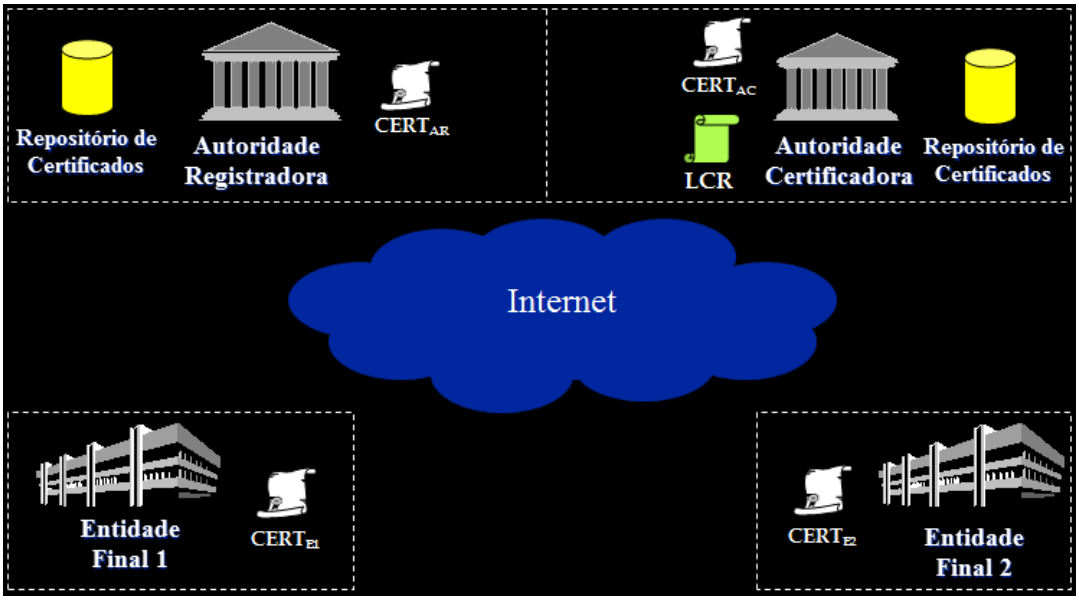


Figura 27 - ICP-Brasil grau de abstração 1

Fonte: GUELFI, 2012i, p. 6

Na Figura 28 é retratado o processo de geração de certificados, e da comunicação entre entidades além da hierarquia desse processo. Aqui, é importante ressaltar que cada membro do processo possui seu próprio certificado e par de chaves, e obedece a uma hierarquia de subordinação para validação.

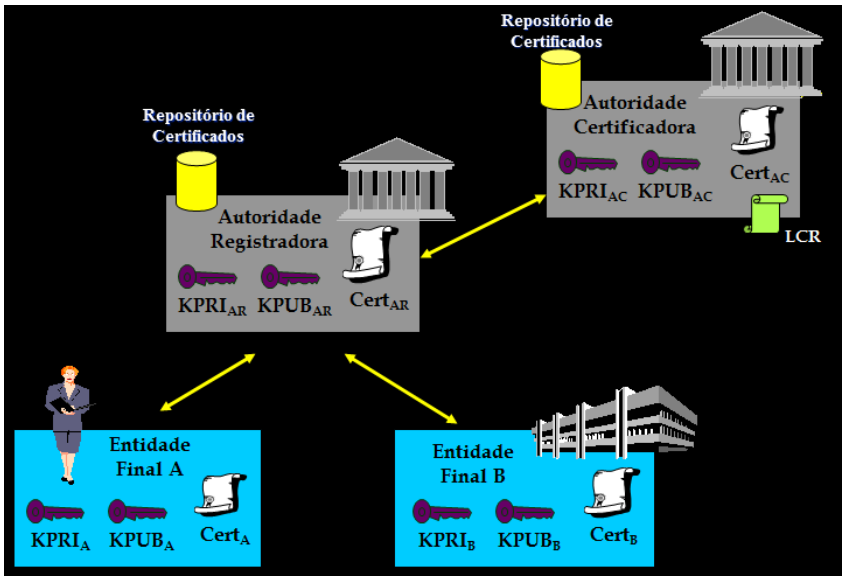


Figura 28 - ICP-Brasil grau de abstração 2

Fonte: GUELFI, 2012i, p. 7

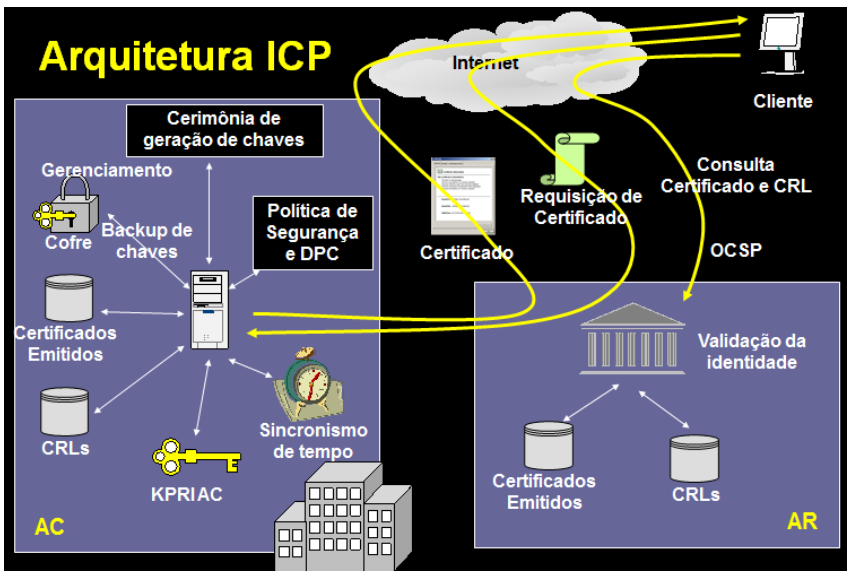


Figura 29 - ICP-Brasil grau de abstração 3

Fonte: GUELFI, 2012i, p. 24

A ultima imagem referente à ICP-Brasil (Figura 29), retrata os processos já descritos aqui, do ponto de vista do cliente, o usuário final. Como ele tem acesso a toda essa infraestrutura.

A ICP-Brasil criou o padrão brasileiro baseado no internacional, e é por isso que existem semelhanças entre eles, porém, visando atender ao âmbito nacional e simplificar o padrão internacional, algumas mudanças foram feitas. As figuras 30 à 34, ilustram os cinco formatos brasileiros de assinatura digital, que foram implementados no presente trabalho.

a) Assinatura Digital com Referencia Básica (AD-RB) – É o formato de assinatura digital básico da ICP-Brasil, para que uma assinatura digital no Brasil seja considerada válida ela deve ter, ao menos, os atributos definidos nesse formato;

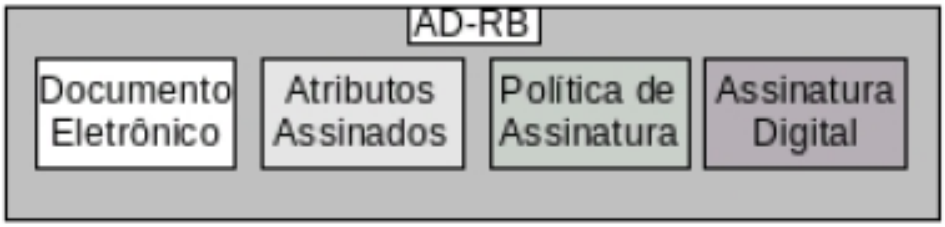


Figura 30 - AD-RB

Fonte: DOC-ICP-15.01, 2008, p. 13

b) Assinatura Digital com Referencia de Tempo (AD-RT) – É o formato de assinatura digital com informações relativas a tempo. Esse formato contém uma informação chamada carimbo de tempo, que garante que a assinatura existia naquela data (a data em que o carimbo de tempo foi gerado pela TSA);

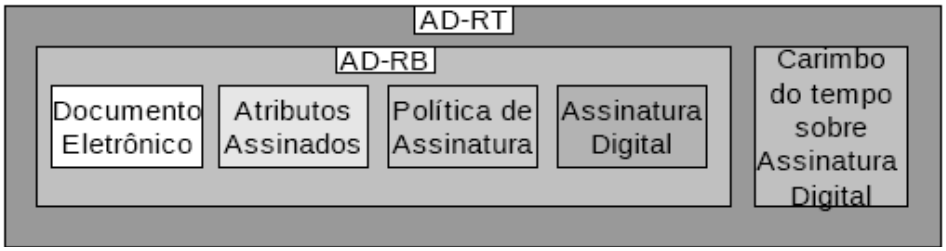


Figura 31 - AD-RT

Fonte: DOC-ICP-15.01, 2008, p. 13

c) Assinatura Digital com Referencias de Validação (AD-RV) – Esse formato contém os resumos criptográficos dos certificados e das CLRs, além de um carimbo de tempo sobre os mesmos. Sua intenção fornecer, de maneira segura, referencias para a validação das informações referentes à assinatura digital;

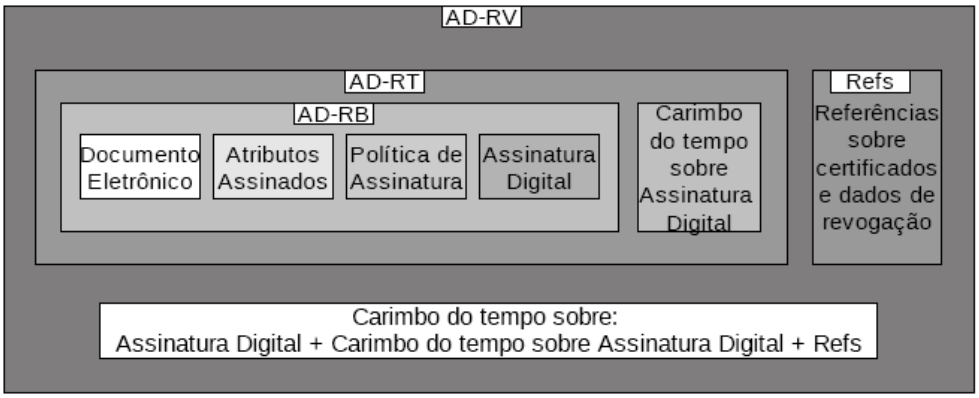


Figura 32 - AD-RV

Fonte: DOC-ICP-15.01, 2008, p. 13

d) Assinatura Digital com Referencias Completas (AD-RC) – Esse formato adiciona os certificados e as CRLs em sua lista de atributos não assinados;

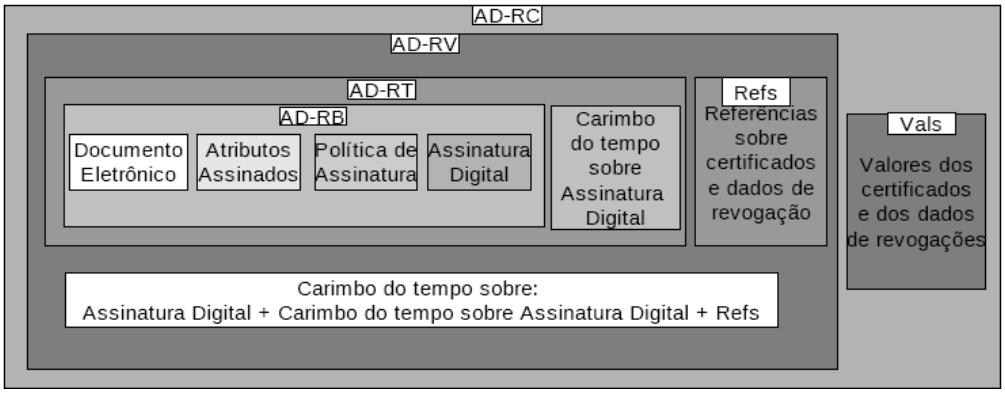


Figura 33 - AD-RC

Fonte: DOC-ICP-15.01, 2008, p. 14

e) Assinatura Digital com Referencia de Arquivamento (AD-RA) – O AD-RA foi criado com o intuito de permitir que assinaturas antigas, feitas com algoritmos criptográficos já considerados inseguros, fossem arquivadas com segurança.

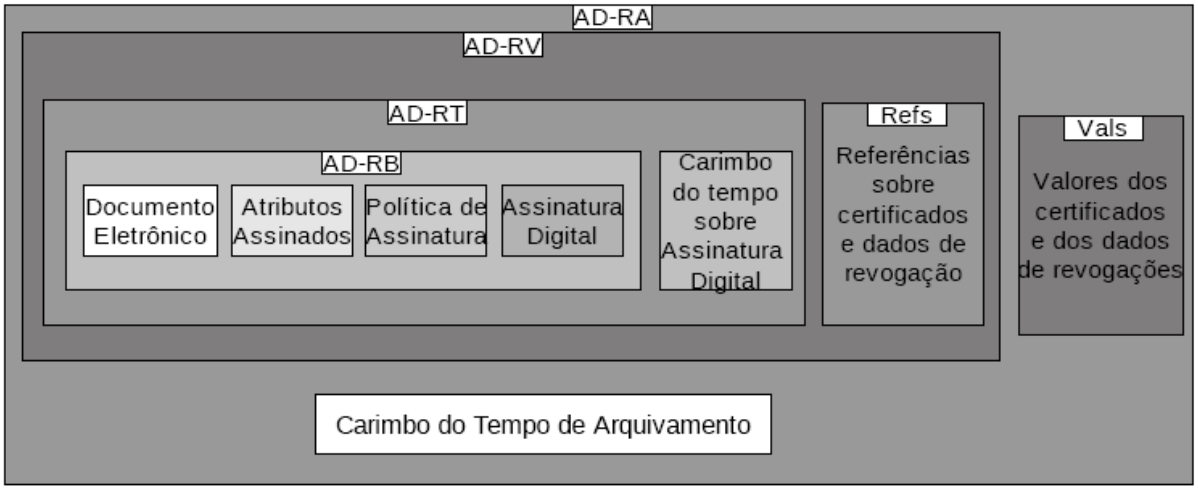


Figura 34 - AD-RA

Fonte: DOC-ICP-15.01, 2008, p. 14

3. API de Assinatura Digital de Acordo com as Normas ICP-Brasil

Esse capítulo retrata o processo de desenvolvimento da API, sua estrutura, organização, funcionamento, as principais dificuldades no desenvolvimento, peculiaridades da implementação, resultados alcançados e testes.

3.1. Definições Pertinentes da ICP-brasil

A ICP-Brasil regulamenta em seus DOCs as operações que devem ser oferecidas por um assinador digital, além de definir alguns termos. A seguir as definições de alguns desses termos:

a) Uma assinatura eletrônica representa um conjunto de dados, no formato eletrônico, que é anexado ou logicamente associado a um outro conjunto de dados, também no formato eletrônico, para conferir-lhe autenticidade ou autoria (DOC-ICP-15, 2010, p. 10);

b) Cadeia de certificação é uma série hierárquica de certificados assinados por sucessivas Autoridades Certificadoras (ACs) (DOC-ICP-15, 2010, p. 6);

c) Carimbo do tempo é documento eletrônico emitido por uma parte confiável, que serve como evidência de que uma informação digital existia numa determinada data e hora (DOC-ICP-15, 2010, p. 6);

d) Com relação ao processo de geração de assinatura digital, podemos ter três contextos diferentes: assinaturas simples, co-assinaturas e contra-assinaturas (DOC-ICP-15, 2010, p. 16):

- A geração de assinatura digital simples ocorre quando uma única assinatura digital é gerada sobre um conteúdo digital disponível;

- A geração de co-assinaturas digitais ocorre quando duas ou mais assinaturas digitais são geradas de forma paralela e independente pelos signatários, utilizando

conteúdos digitais idênticos. Cada co-assinatura gerada pode conter atributos assinados e não assinados próprios;

- A geração de contra-assinaturas digitais ocorre quando uma ou mais assinaturas digitais são realizadas sobre a seqüência de bytes (bloco) que representa uma assinatura digital já existente. Uma contra-assinatura pode conter outros atributos assinados próprios;

e) O termo “assinaturas digitais em lote” representa um caso particular da assinatura digital, no qual é necessário realizar diversas assinaturas digitais em um lote de conteúdos digitais (uma assinatura digital para cada conteúdo do lote), resultando assim em diversas operações criptográficas sequenciais utilizando a mesma chave assimétrica privada do signatário (DOC-ICP-15, 2010, p. 16).

f) O Termo “assinatura simples” utilizado nessa monografia se refere a realização de uma assinatura digital qualquer, utilizando qualquer um dos seguintes formatos: AD-RB, AD-RT, AD-RV e AD-RC.

3.2. API de Assinaturas Digitais ICP-Brasil

A API Boucy Castle Criptografhy Library (BC) consiste de uma coleção bibliotecas usadas em processos criptográficos, sendo periodicamente atualizada e revisada por seus mantenedores. Ela oferece muitas facilidades, como código fonte aberto, e uma considerável parte das estruturas ETSI implementadas. A API BC é implementada em Java e oferece diversos pacotes para desenvolvimento de assinaturas e afins.

Para o desenvolvimento da API aqui documentada, foi utilizada a API BC. A API desenvolvida foi feita para ser totalmente compatível com as normas e regulamentações da ICP-Brasil. Logo, a API tem suporte aos cinco formatos de assinatura previstos na ICP-Brasil, são eles: AD-RB, AD-RT, AD-RV, AD-RC, AD-RA.

3.2.1. Organização da API de assinatura

A API foi estruturada da seguinte forma, existe uma classe CMS que é a responsável por gerar todos os elementos do CMS: o *CMSVersion*, o *DigestAlgorithmIdentifiers*, o *EncapsulatedContentInfo*, o *CertificateSet*, o *CertificateRevocationLists* e o *SignerInfos*. Essa classe recebe as informações de entrada a partir de objetos das classes:

a) *Subscriber*: classe que armazena as informações pertinentes ao assinante;

b) *Signature*: classe que armazena as informações pertinentes à informação a ser assinada.

Essa separação é feita para facilitar a assinatura em lote, que é realizada através da classe *BatchSignature*, dessa maneira basta N objetos *Subscriber* e um *Signature* para efetuar esse tipo de assinatura.

A diferença básica de um formato de assinatura para outro são os atributos não assinados (*unsignedAttrs*) presentes dentro do *SignerInfo*. Assim sendo, existem classes separadas para geração dos atributos não assinados, essas classes são: ADRB, ADRT, ADRV, ADRC, ADRA. Abaixo encontra-se a descrição ASN.1 do *SignerInfo*, na Tabela 4 encontra-se a tabela de atributos não assinados segundo a ICP-Brasil:

```
SignerInfo ::= SEQUENCE {
        version CMSVersion,
        sid SignerIdentifier,
        digestAlgorithm DigestAlgorithmIdentifier,
        signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
        signatureAlgorithm SignatureAlgorithmIdentifier,
        signature SignatureValue,
        unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL
}
```

É importante ressaltar que existem inúmeros *SignerInfos* dentro de um CMS, de fato, existe um para cada assinante, e pode haver N assinantes. O *SignerInfo* é onde está a assinatura digital em si, ou seja, onde está o *hash* criptografado com a chave privada. O *hash* é calculado a partir do documento assinado concatenado com os atributos assinados. A Tabela 5 mostra os atributos assinados segundo a ICP-Brasil.

O fato da alteração, apenas, de atributos não assinados de um formato para outro, permite a fácil migração entre eles. O ETSI funciona dessa maneira, a ICP-Brasil fez algumas

alterações. Umas das mais impactantes delas no processo de desenvolvimento dessa API, foi a criação de uma política para cada formato de assinatura, sendo a política um atributo assinado. Isso impede a migração de um formato para outro, obrigando a recriação de toda a assinatura quando se pretende alterar o formato.

Tabela 4 - Atributos não assinados no SignerInfo do assinante

DOC-ICP-15.03, 2012, p. 21

Nome do atributo / Propriedade	**Identificação do atributo**	**Perfil AD**				
	Propriedade	**RB**	**RT**	**RV**	**RC**	**RA**
Contra assinatura (*countersignature*)	**id-countersignature**	P	P	P	P	P
	CounterSignature					
Carimbo do tempo de assinatura (*signature time stamp*)	**id-aa-signatureTimeStampToken**	ND	O	O	O	O
	SignatureTimeStamp					
Referências completas aos certificados (*complete certificate references*)	**id-aa-ets-certificateRefs**	P	P	O	O	O
	CompleteCertificateRefs					
Referências completas à revogação *(complete revogation references)*	**id-aa-ets-revocationRefs**	P	P	O	O	O
	CompleteRevocationRefs					
Referências aos certificados de atributo (*attribute certificate references*)	**id-aa-ets-attrCertificateRefs**	P	P	P	P	P
	AttributeCertificateRefs					
Referências à revogação de atributo (*attribute revogation references*)	**id-aa-ets-attrRevocationRefs**	P	P	P	P	P
	AttributeRevocationRefs					
Carimbo do tempo das referências (*time-stamped certificate crls references*)	**id-aa-ets-escTimeStamp**	ND	P	O	O	P
	SigAndRefsTimesStamp					
Valores dos certificados (*certificate values*)	**id-aa-ets-certValues**	P	P	P	O	O
	CertificateValues					
Valores de revogação (*revocation values*)	**id-aa-ets-revocationValues**	P	P	P	O	O
	RevocationValues					
Carimbo do tempo de arquivamento (*archive time-stamp)*	**id-aa-ets-archiveTimestampV2**	ND	ND	ND	ND	O
	ArchiveTimeStamp					

Dessa forma, toda vez que uma assinatura é redefinida, passa de um formato para outro, ela é, na verdade, refeita. As classes anteriormente definidas como responsáveis pela geração dos atributos não assinados agregam, também, os métodos para geração dos atributos assinados.

A API oferece suporte as seguintes opções de assinatura: assinatura simples; co-assinatura; assinatura em lote; e redefinição de assinatura. Todas essas opções são geradas de forma idêntica, através da classe CMS. Essa classe por sua vez fará ligação com as classes: ADRB; ADRT; ADTV; ADRC; ADRA; que deverão gerar as tabelas de atributos assinados (*signedAttrs*) e de atributos não assinados (*unsignedAttrs*).

Tabela 5 - Atributos assinados no SignerInfo do assinante

DOC-ICP-15.03, 2012, p. 20

Nome do atributo / Propriedade	**Identificação do atributo**	**Perfil AD**				
	Propriedade	**RB**	**RT**	**RV**	**RC**	**RA**
Tipo de conteúdo (*content type*)	**id-contentType**	O	O	O	O	O
Resumo criptográfico da mensagem (*message digest*)	**id-messageDigest**	O	O	O	O	O
Certificado do signatário (*ESS signing certificate*)	**Id-aa-signingCertificate** [1] **id-aa-signingCertificateV2** [2]	O	O	O	O	O
	SigningCertificate					
Identificador da política de assinatura (*signature policy identifier*)	**id-aa-ets-sigPolicyId**	O	O	O	O	O
	SignaturePolicyIdentifier					
Atributos do signatário (*signer attributes*)	**id-aa-ets-signerAttr**	P	P	P	P	P
	SignerRoles					
Instante da assinatura (*signing time*)	**id-signingTime**	P	P	P	P	P
	SigningTime					
Localização do signatário (*signer location*)	**id-aa-ets-signerLocation**	P	P	P	P	P
	SignerProductionPlace					
Carimbo do tempo de conteúdo (*content time stamp*)	**id-aa-ets-contentTimeStamp**	P	P	P	P	P
	AllDataObjectsTimeStamp, IndividualDataObjectsTimeStamp					

A Figura 35 abaixo mostra um diagrama do pacote *signatures* (Figura 35), em sequência são mostrados os diagramas dos pacotes: *formats* (Figura 36), *policy* (Figura 37) e *util* (Figura 38) e suas respectivas descrições.

a) O pacote *signatures* contém as principais classes da API, elas são utilizadas por praticamente todas as funções disponibilizadas pela ferramenta, de maneira quase idêntica;

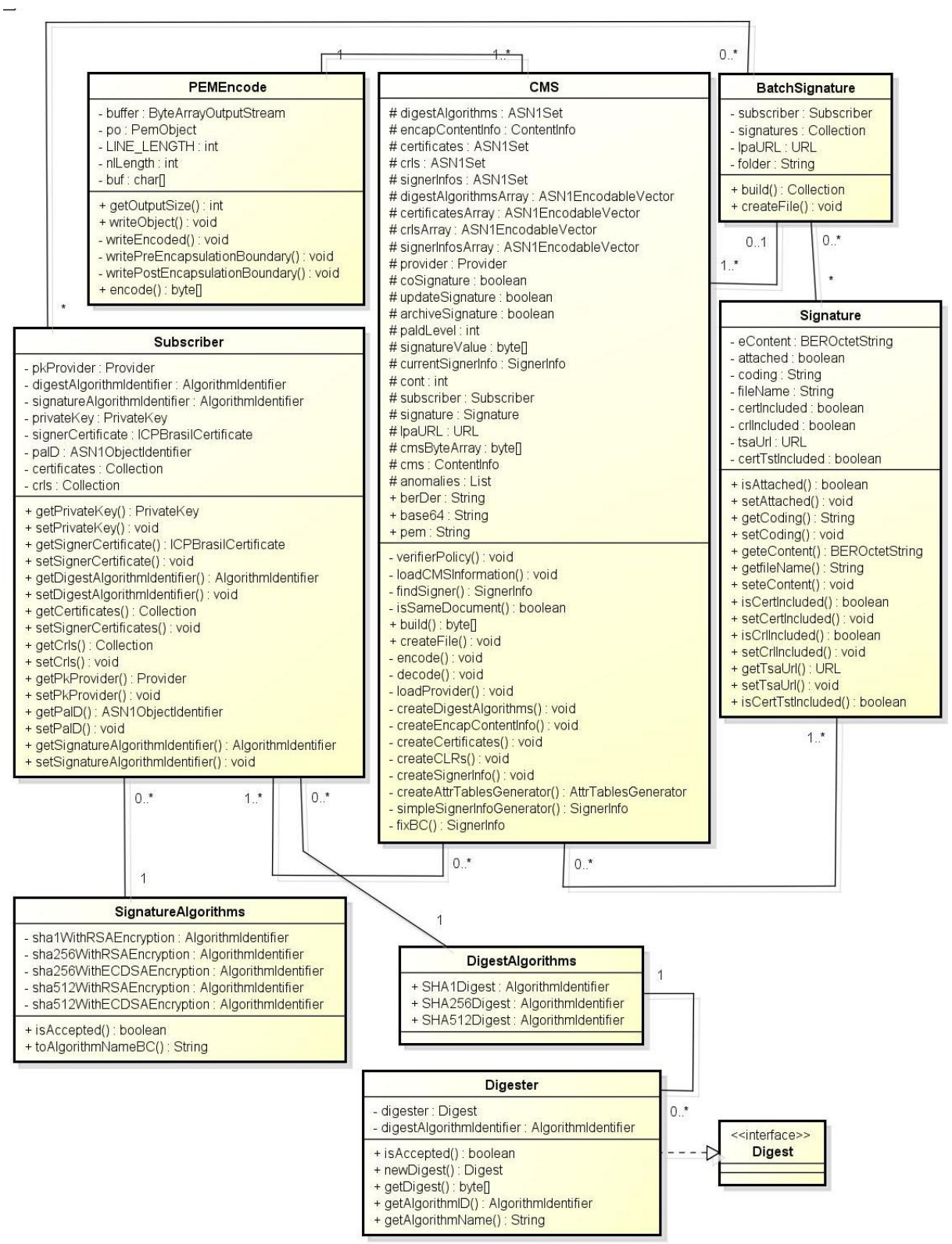


Figura 35 - Diagrama de classe do pacote *signatures*

b) O pacote *formats* possui as classes responsáveis pela criação da tabelas de atributos assinados e não assinados, elas herdam umas as outras em ordem incremental de acordo com os formatos ICP-Brasil;

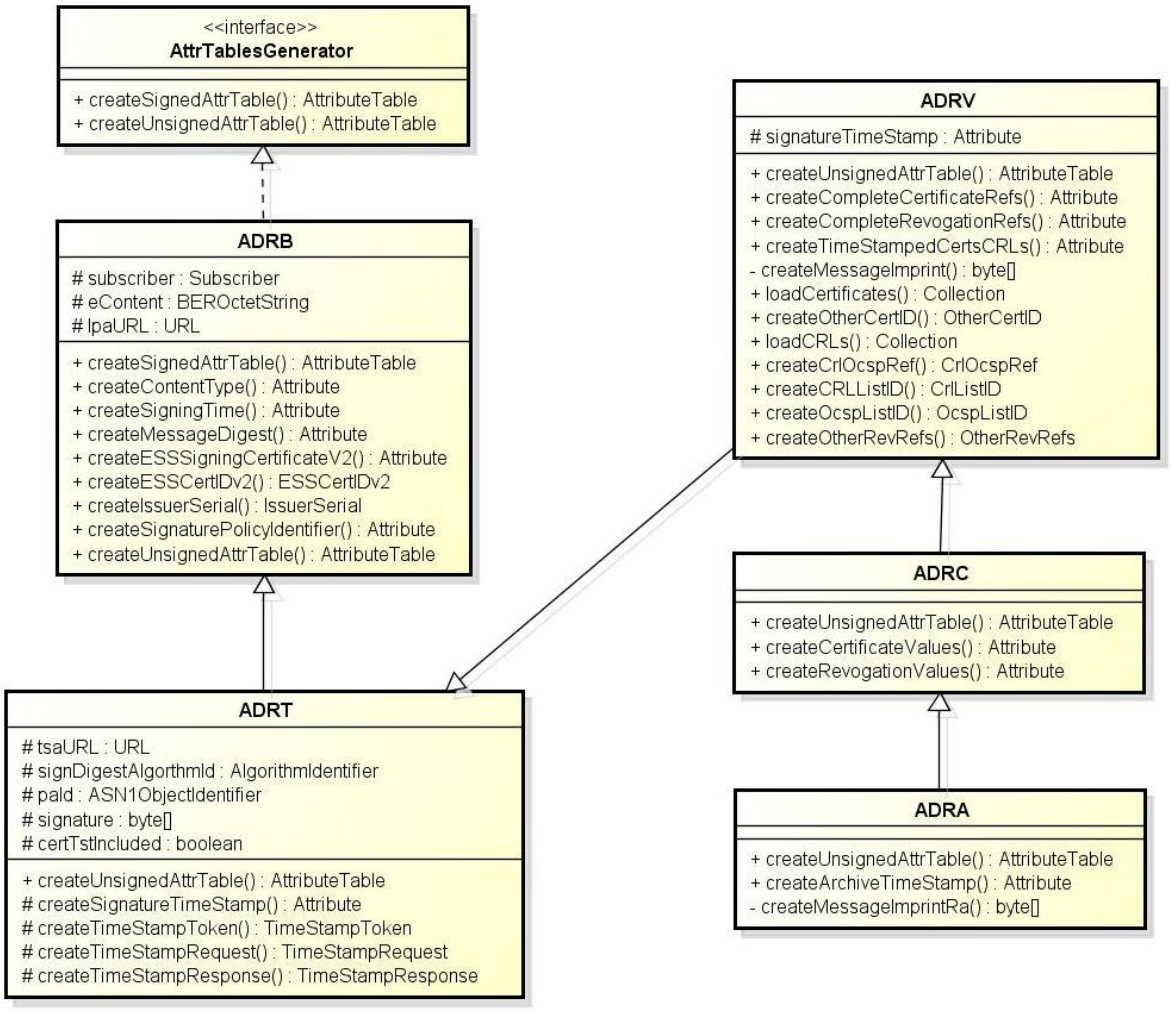


Figura 36 - Diagrama de classe do pacote *formats*

c) O pacote *policy* possui as classes pertinentes as politicas da ICP-Brasil, implementa a estrutura ASN.1 da LPA, que é exclusiva da ICP-Brasil, além de implementar o *SignaturePolicyIdentifier*;

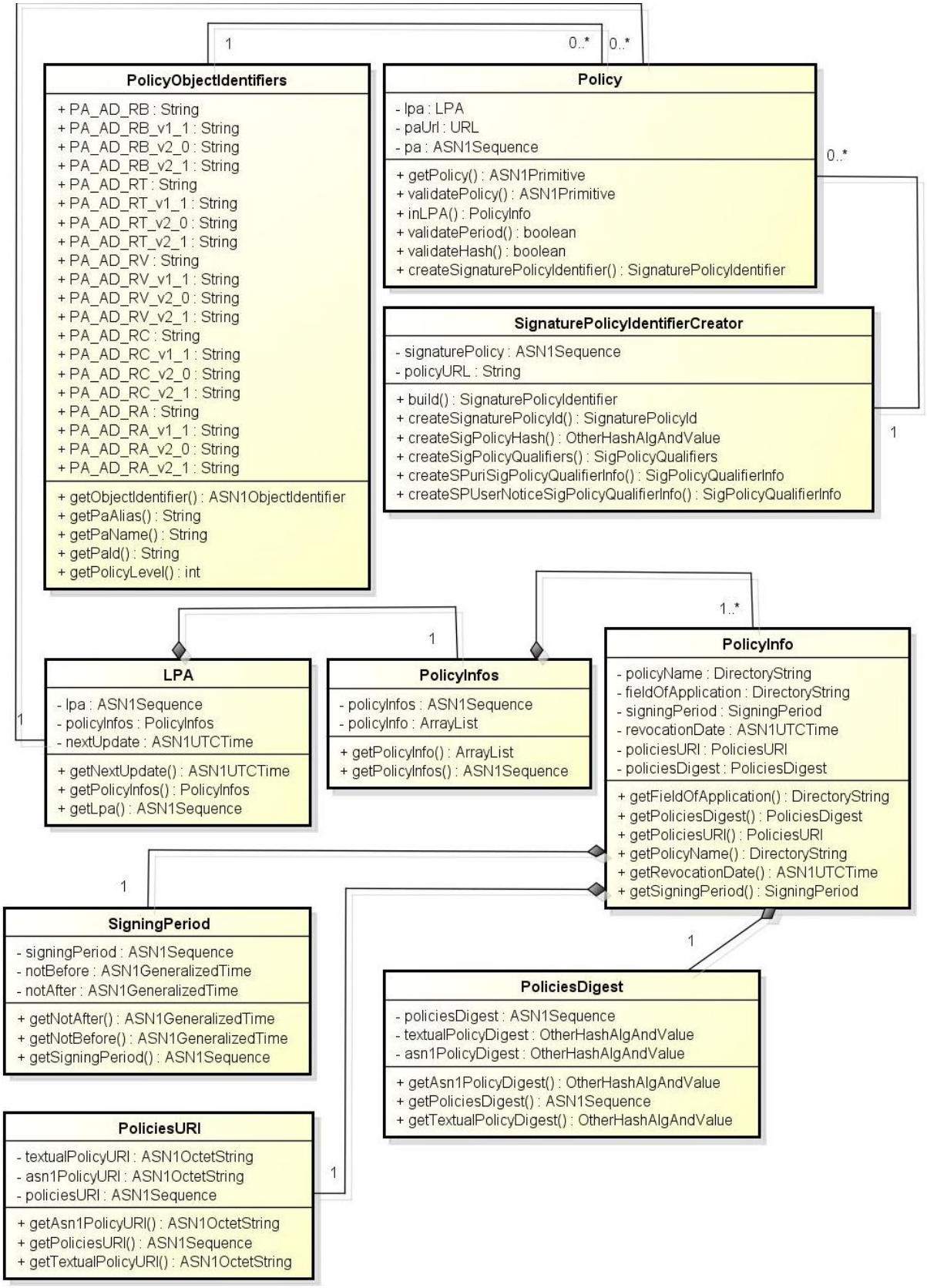


Figura 37 - Diagrama de classe do pacote *policy*

d) O pacote *util* possui apenas uma classe, esse por sua vez possui atributos estáticos que fornecem funções genéricas utéis a API como um todo.

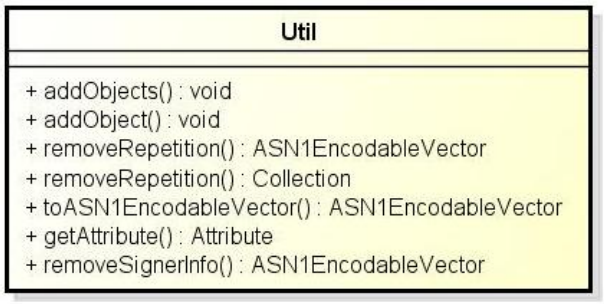


Figura 38 - Diagrama de classe do pacote *util*

3.2.2. Definição dos Atributos Assinados e Não Assinados

Como é possível observar na Tabela 4 e na Tabela 5, não é necessária a implementação de todos os atributos presentes nessa tabela para a criação de um assinador que atenda aos requisitos mínimos da ICP-Brasil. Sendo assim, foram implementados apenas os atributos considerados obrigatórios, os mesmos são descritos abaixo:

a) O *ContentType* especifica o tipo do conteúdo presente no *ContentInfo* do *signed-data*. Precisa corresponder ao *eContentType* do *encapContentInfo*. Devido ao escopo desse projeto, o *ContentType* será sempre o de dados (RFC 5652, p. 40). Definição em ASN.1:

```
id-contentType OBJECTIDENTIFIER ::= {
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) 3
}

ContentType ::= OBJECT IDENTIFIER
```

b) O *SigningTime* especifíca quando a assinatura foi gerada. Não há nenhuma regra para validação desse atributo, assume-se apenas que o assinante é confiável. Para maiores garantias é necessário o carimbo de tempo. Datas entre 1 de janeiro de 1950 e 31 de dezembro de 2049 (inclusive) precisam ser representadas como *UTCTime*, datas fora desse período devem ser representadas com *GeneralizedTime* (RFC 5652, p. 41). Definição em ASN.1:

```
id-signingTime OBJECT IDENTIFIER ::= {
```

```
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) 5
}

SigningTime ::= Time

Time ::= CHOICE {
        utcTime         UTCTime,
        generalizedTime  GeneralizedTime
}
```

c) O *MessageDigest* especifica o *hash* do *eContent* do *encapContentInfo* sendo assinado no *signed-data*. É calculado utilizando-se o algoritmo de *hash* do assinante (RFC 5652, p. 40-41). Definição em ASN.1:

```
id-messageDigest OBJECT IDENTIFIER ::= {
        iso(1) member-body(2)us(840) rsadsi(113549) pkcs(1) pkcs9(9) 4
}

MessageDigest ::= OCTET STRING
```

d) O *ESSSigningCertificate* é uma estrutura que cria um link criptográfico entre o certificado e a assinatura. Devido ataques contra o SHA-1 uma nova estrutura foi definida, o SigningCertificateV2 (RFC 5035, p. 2).

O *certs* contém a lista de certificados que serão usados na validação da mensagem. O primeiro certificado identificado na sequencia de identificadores de certificados precisa ser o certificado usado na verificação da assinatura. A codificação do *ESSCertIDv2* desse certificado deve conter o campo *issuerSerial*. Se mais de um certificado estiver presente, os certificados subsequentes limitam o conjunto de certificados que serão usados durante o processo de validação. Se apenas o certificado do assinante está presente nessa *sequence*, não há restrições no conjunto de certificados usados no processo de validação. Nessa API será codificado apenas o certificado do assinante (RFC 5035, p. 4).

O *policies* contem informações de políticas de certificados. Ele é opcional e não será utilizado nessa API (RFC 5035, p. 5).

Definição em ASN.1:

```
id-aa-signingCertificateV2 OBJECT IDENTIFIER ::= {
```

```
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
                        smime(16) id-aa(2) 47
}

SigningCertificateV2 ::=  SEQUENCE {
        certs       SEQUENCE OF ESSCertIDv2,
        policies    SEQUENCE OF PolicyInformation OPTIONAL
}
```

e) O *Signature time-stamp* é um *TimeStampToken* calculado em cima do valor da assinatura de um assinante específico. Muitas instâncias desse atributo podem ocorrer numa mesma assinatura, de diferentes TSAs (RFC 5126, p. 47-48). Definição em ASN.1:

```
id-aa-signatureTimeStampToken OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
                        smime(16) id-aa(2) 14
}

SignatureTimeStampToken ::= TimeStampToken

TimeStampToken ::= ContentInfo
        -- contentType is id-signedData ([CMS])
        -- content is SignedData ([CMS])
```

f) O *CompleteCertificateReferences* referencia o conjunto completo de certificados da CA que foi usado para validar uma ES com dados completos de validação do certificado do assinante (mas não o certificado). Só pode ocorrer uma instância desse atributo (RFC 5126, p. 48-49). Definição em ASN.1:

```
id-aa-ets-certificateRefs OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
                        smime(16) id-aa(2) 21
}

CompleteCertificateRefs ::= SEQUENCE OF OtherCertID
```

g) O *CompleteRevocationReferences* referencia o conjunto completo de CRL, ACRL ou OCSP respostas que devem ser usadas no processo de validação do assinante, e certificados de CA usados na ES (RFC 5126, p. 49-51). Definição em ASN.1:

```
id-aa-ets-revocationRefs OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
```

```
                    smime(16) id-aa(2) 22
}

CompleteRevocationRefs ::= SEQUENCE OF CrlOcspRef
```

h) O *TimeStampedCertsCRLsReferences* é usado para proteger através de um compromisso da CA (RFC 5126, p. 57). Definição em ASN.1:

```
id-aa-ets-certCRLTimestamp OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
                    smime(16) id-aa(2) 26
}

TimestampedCertsCRLs ::= TimeStampToken
```

i) O *CertificateValues* contém os valores de certificados da cadeia de certificação (RFC 5126, p. 54). Definição em ASN.1:

```
id-aa-ets-certValues OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
                    smime(16) id-aa(2) 23
}

CertificateValues ::= SEQUENCE OF Certificate
```

j) O *RevogationValues* contém os valores de *clrs* dos certificados da cadeia de certificação (RFC 5126, p. 54). Definição em ASN.1:

```
RevocationValues ::= SEQUENCE {
        crlVals [0] SEQUENCE OF CertificateList OPTIONAL,
        ocspVals [1] SEQUENCE OF BasicOCSPResponse OPTIONAL,
        otherRevVals [2] OtherRevVals OPTIONAL
}
OtherRevVals ::= SEQUENCE {
        otherRevValType OtherRevValType,
        otherRevVals ANY DEFINED BY OtherRevValType
}
OtherRevValType ::= OBJECT IDENTIFIER
```

l) *ArchiveTimeStamp* é utilizado quando uma assinatura digital é necessária por muito tempo, ela talvez tenha que ser carimbada muitas vezes, devido a ataques contra os algoritmos. Quando isso acontece, deve-se usar o carimbo de tempo de arquivamento (RFC 5126, p. 58-60). Definição em ASN.1:

```
id-aa-ets-archiveTimestampV2 OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
                smime(16) id-aa(2) 48
}

ArchiveTimeStampToken ::= TimeStampToken
```

3.2.3. As Funcionalidades da API de Assinatura Digital

A API de assinatura oferece funcionalidades já mencionadas nessa monografia, com o intuído de tornar seu uso o mais simples possível, existem poucas diferenças no modo de proceder para realizar as operações. Abaixo será descrito como proceder na realização de cada uma.

a) Assinatura simples – Para gerar uma assinatura simples deve-se preencher os objetos: *Subscriber* e *Signature*; com as informações necessárias (estão descritas em seu construtor), todas as informações necessárias para a geração da assinatura estão neles; deve-se então usar o construtor com a seguinte assinatura: *public CMS(Subscriber subscriber, Signature signature, URL lpaURL)*; nesse ponto há duas opções: usar o método *build*, que irá retornar a assinatura em um vetor de bytes armazenado na memória, ou, utilizar o método *createFile* passando como parâmetro o diretório em que a assinatura será salva e a API salvará a assinatura em um arquivo com extensão “p7s”;

b) Co-assinatura – Para gerar uma co-assinatura o processo é quase idêntico, a diferença é que você deverá utilizar o construtor com a seguinte assinatura: *public CMS(Subscriber subscriber, Signature signature, File inputCMS, URL lpaURL, int option)*; o valor do parâmetro *option* deve ser o número um;

c) Redefinição de assinatura – Redefinir uma assinatura é passar uma assinatura de um formato para outro, para fazer isso basta seguir os passos já descritos com uma alteração, o valor do parâmetro *option* deve ser o número dois;

d) Arquivamento de assinatura – Arquivar uma assinatura nada mais é do que gerar uma assinatura com política AD-RA a partir de uma outra política, para fazer isso basta seguir os passos já descritos com uma alteração, o valor do parâmetro *option* deve ser o número três;

e) Assinatura em lote – Essa operação é um pouco diferente, aqui a classe usada deve ser a *BatchSignature*, ainda se utilizam os objetos das classes *Subscriber* e *Signature* como provedores da informação, porém, aqui utilizar-se-á uma *Collection* de *Signatures*; a partir daí o processo é idêntico, essa classe possui os mesmos métodos (*build* e *createFile*) da classe CMS, e esses funcionam de maneira idêntica ao dessa.

Como já mencionado anteriormente, o objetivo dessa monografia, o desenvolvimento de uma API de assinatura digital de acordo com as normas e regulamentações da ICP-Brasil, é parte formadora de um software de assinatura digital. A API resultante desse trabalho foi utilizada no software ICP-Brasil Signer, como é mostrado na figura 39.

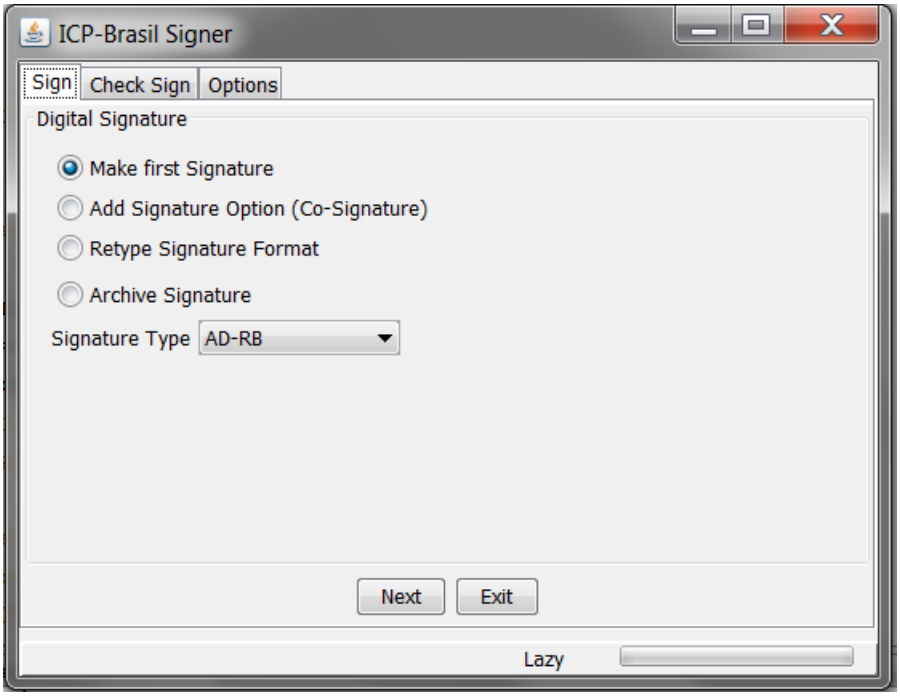


Figura 39 – *Software* de assinatura digital

3.3. Artigos Publicados

Um dos objetivos dessa pesquisa é, utilizando os conhecimentos adquiridos na implementação da API de assinatura digital, contribuir para a evolução da infraestrutura brasileira de chaves públicas através da publicação de artigos. Nesse sentido foi publicado um

artigo no 10º CertForum. O CertForum é o maior fórum sobre certificação digital do Brasil. É promovido pelo ITI em vários estados brasileiros.

O artigo publicado tinha como conteúdo, além de apresentar a ferramenta de assinatura digital completa, isto é, apresentando também os outros módulos, conforme discutido na introdução, discorrer sobre uma das maiores dificuldades no processo de desenvolvimento da API de assinatura. Essa dificuldade foi a alteração do papel da política em relação ao ETSI, e é melhor apresentada na conclusão da presente monografia.

4. Conclusão

A ICP-Brasil, como já mencionado anteriormente, fez modificações e adaptações no padrão internacional (ETSI), quando definiu o padrão brasileiro. Uma das modificações foi referente à política de assinatura.

A ICP-Brasil, apesar de definir a política de assinatura com referencia aos documentos ETSI, atrelou a política ao formato da assinatura, em outras palavras, para cada formato de assinatura previsto na ICP-Brasil existe uma, ou mais, políticas válidas específicas.

A consequência dessa diferença é a incapacidade de se migrar de um formato de assinatura para outro. Isto é, não se pode adaptar uma assinatura de formato de nível inferior para um de nível superior apenas incluindo os atributos não assinados presentes em sua definição. É preciso gerar novamente a assinatura presente no CMS, uma vez que a política é um atributo assinado, sua alteração invalida a assinatura, e ela precisa ser alterada para se mudar de formato.

Essa impossibilidade transforma-se em um grande transtorno, uma vez que muitos formatos, com exceção da política, mudam apenas a composição dos atributos não assinados presentes em seu *SignerInfo*, e poderiam ser facilmente migrados. Isto é, poderia-se apenas inserir os atributos obstantes. Processo que é possível no padrão ETSI, após a criação do formato EPES.

Esse foi sem dúvidas um dos maiores problemas enfrentados no processo de desenvolvimento do assinador digital ICP-Brasil, e suas soluções demandaram muito esforço. Outro problema que contribui para aumentar esse já citado é o fato da existência de um grande número de documentos e normas pertinentes. Apesar das normas ICP-Brasil serem relativamente diminutas, elas fazem referência a muitas normas internacionais, que por sua vez fazem referência à outras normas. Também é possível ressaltar como dificuldade a falta de documentação das classes BC, que por muitas vezes partem do principio que o utilizador das mesmas possui conhecimento avançado das normas internacionais.

Como resultado final foi desenvolvido, com êxito, em linguagem de programação Java, JRE e JDK versão 1.7.0.20.13, utilizando o ambiente de desenvolvimento NetBeans

IDE 7.1, API Java BC-Provider e BC-PKI versão 1.46, uma API que efetua assinaturas digitais de acordo com as normas da ICP-Brasil em todos os formatos previstos pela mesma.

Existem muitas estruturas ETSI e, na pesquisa efetuada, não foi encontrada empresa ou instituição que tenha implementado todas elas, esse trabalho concentrou-se nas pertinentes a ICP-Brasil, que estão devidamente documentadas no *DOC-ICP-15.02*, sendo implementadas as necessárias para obtenção de uma API funcional. Como trabalhos futuros cogita-se a implementação das outras estruturas documentadas no documento mencionado acima.

Referências Bibliográficas

ETSI. **ETSI TS 101 733 (V1.7.4):** Electronic Signatures and Infrastructures (ESI); CMS Advanced Electronic Signatures (CAdES). 2008.

ETSI. **ETSI TR 102 041 (V1.1.1):** Signature Policies Report. 2002.

FERREIRA, Fernando Nicolau Freitas. **Segurança da Informação**. 1. ed. Rio de Janeiro: Editora Ciência Moderna Ltda., 2003.

FRIEDRICH, Diego Mostardeiro; MEDINA, Roseclea Duarte. **Certificação Digital Acadêmica**: Implantação do Sistema de Gerenciamento de Certificados Digitais ICPEDU na UFSM. Universidade Federal de Santa Maria (UFSM). Dezembro, 2007.

GUELFI, Adilson Eduardo. **Serviços de Segurança**. Marília, 2012a. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Criptografia Simétrica**. Marília, 2012b. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Criptografia de Chave Pública**. Marília, 2012c. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Troca de Chaves Criptográficas**. Marília, 2012d. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Funções *Hash***. Marília, 2012e. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Assinatura Digital**. Marília, 2012f. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Envelopes Digitais**. Marília, 2012g. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Certificado Digital**. Marília, 2012h. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Introdução à ICP (PKI)**. Marília, 2012i. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Estrutura dos Certificados Digitais**. Marília, 2012j. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **CMS – Cryptographic Message Syntax**. Marília, 2012l. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo. **Padrão e Modelos de Assinatura Digital ICP-Brasil**. Marília, 2012m. Apresentação de slides utiliza na ministração de treinamento em processos e conceitos pertinentes a assinatura digital no Brasil.

GUELFI, Adilson Eduardo et al. 10º CERTFORUM, 2012n, Florianópolis-SC. **Assinador Digital ICP-Brasil**: Implementação de Referência em Java.

ICP-BRASIL. **DOC-ICP-01.01**: Padroes e algoritimos criptograficos da ICP-Brasil – Versão 2.3, 2012.

ICP-BRASIL. **DOC-ICP-15**: Visão Geral sobre Assinaturas Digitais na ICP-Brasil – Versão 2.0, 2010.

ICP-BRASIL. **DOC-ICP-15.01**: Visão Geral sobre Assinaturas Digitais na ICP-Brasil – Versão 2.0, 2010.

ICP-BRASIL. **DOC-ICP-15.02**: Visão Geral sobre Assinaturas Digitais na ICP-Brasil – Versão 2.0, 2010.

ICP-BRASIL. **DOC-ICP-15.03**: Visão Geral sobre Assinaturas Digitais na ICP-Brasil – Versão 3.0, 2011.

IETF. **IETF RFC 5652**: Cryptographic Message Syntax (CMS). 2009.

IETF. **IETF RFC 5035**: Enhanced Security Services (ESS) Update Adding CertID Algorithm Agility. 2007.

IETF. **IETF RFC 5126**: CMS Advanced Electronic Signatures (CAdES). 2002.

MENDES, Aliane Veloso. **Estudo de Criptografia com Chave Pública Baseada em Curvas Elípticas**. 2007. 50 f. Monografia (Bacharel em Sistemas de Informação) – Universidade Estadual de Montes Claros, Montes Claros, 2007.

MORENO, Edward David; PEREIRA, Fábio Dacêncio; CHIARAMONTE, Rodolfo Barros. **Criptografia em Software e Hardware**. 1.ed. São Paulo: Novatec Editora Ltda., 2005.

NORTHCUTT, Stephen et al. **Desvendando segurança em redes**: O guia definitivo para fortificação de perímetros de rede usando Firewalls, VPNs, roteadores e sistemas de detecção de invasores. 1. ed. Rio de Janeiro: Campus, 2002.

STALLINGS, William. **Criptografia e segurança de redes**: princípios e práticas. 4. ed. São Paulo: Pearson Prentice Hall, 2008.

TERADA, Routo. **Segurança de dados**: criptografia em redes de computador. 2. ed. São Paulo: 2008.

TRINTA, Fernando Antonio Mota; MACÊDO, Rodrigo Cavalcanti de. **Um Estudo sobre Criptografia e Assinatura Digital**. Departamento de Informática Universidade Federal de Pernambuco. Setembro, 1998.