Segurança na Internet no dia-a-dia

Introdução para leigos ao eschemas X.509, para a troca de chaves públicas entre usuários e sites seguros, e ao eschema OpenPGP, para a troca de chaves públicas entre usuários.

Enno Nagel

UFAL, Maceió — 11 de Dezembro de 2018

Privacidade

Esta palestra **não** será sobre a proteção da privacidade na internet, mas sobre a a questão como saber que posso confiar no (dono do) site?

Para proteger a sua privacidade na internet, sejam recomendados:

- o navegador `Firefox` com as extensões
  - `uBlock Origin`,
  - `Privacy Badger`, e
  - `Decentraleyes`

  e contra o traçamento por `Google`

  - `Searchonymous`
  - `Don't track me Google`
- e eventualmente evitar sites como `Gmail` e `facebook.com`.

Cifração é uma transformação de

dados inteligíveis

$$\Downarrow$$

*dados in*inteligíveis

tal que só uma informação adicional secreta, a *chave* pode desfazê-la.

Outrora (antes da época digital):

- dados = textos, e
- chave = senha (= cadeia de caracteres memorizada)

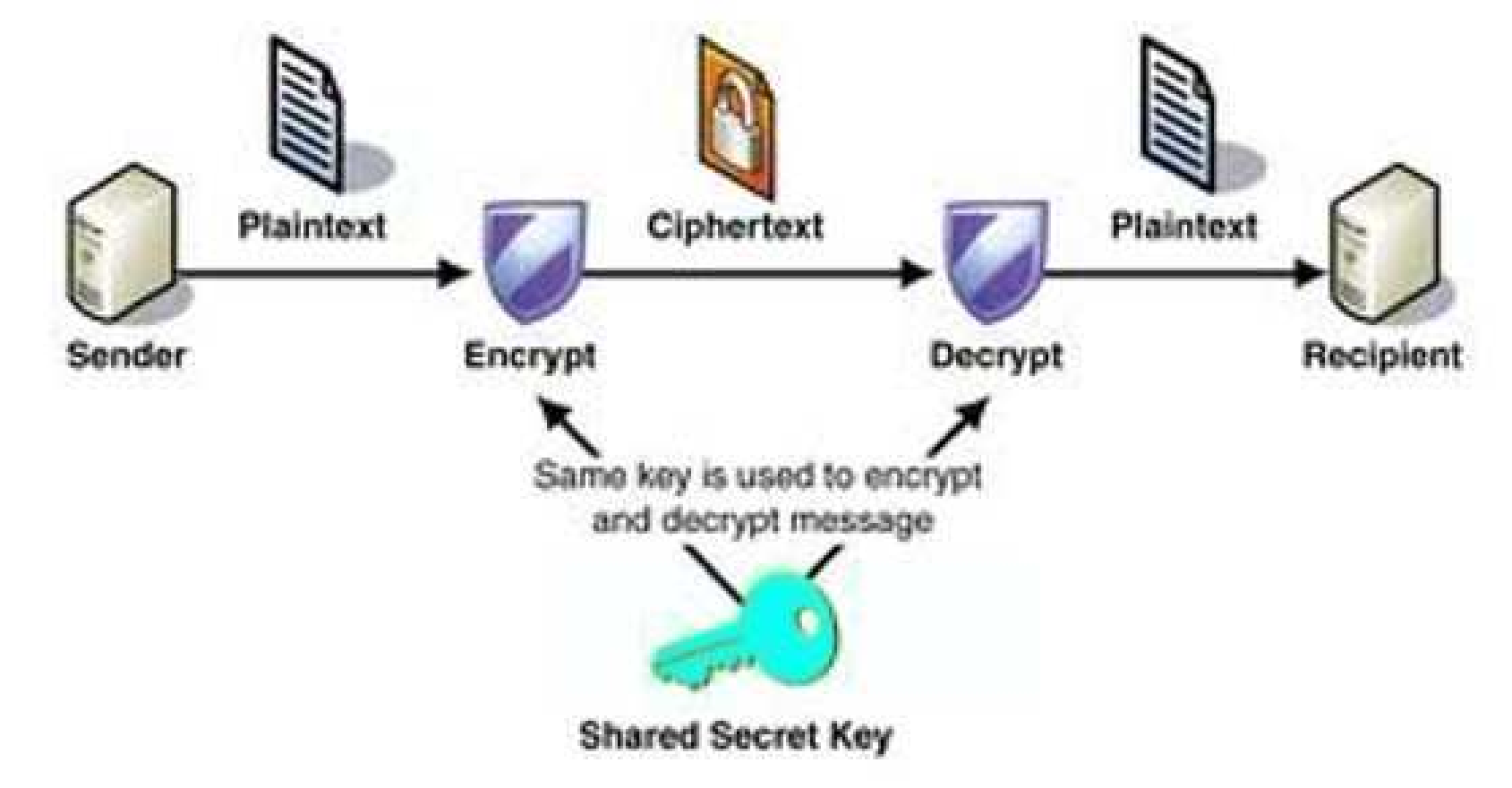
Hoje (na época digital):

- dados = arquivo digital (de texto, imagem, som, vídeo, ...), e
- chave = ou senha (na criptografia *simétrica*), ou arquivo digital (na criptografia *assimétrica*)

Criptografia Simétrica

Na *criptografia simétrica* (já usada pelos egípcios quase 2000 anos antes de Cristo)

chave para cifrar = chave para decifrar.



Substituição (do Alfabeto por traslado das letras)

Este método foi usado por César (63 - 14 a.C.).

- Escolhemos uma *chave*

  δ = uma distância entre letras em ordem alfabética

  = um número entre 0 e 25

- trasladamos por δ cada letra do alfabeto (latino e cujas letras são arranjadas em um anel).

Por exemplo, se $\delta = 3$, então

$$A \mapsto D, B \mapsto E, C \mapsto F, ..., W \mapsto Z, X \mapsto A, ..., Z \mapsto C.$$

e, por exemplo, para $\delta = 3$,

$$CARA \mapsto FDUD$$

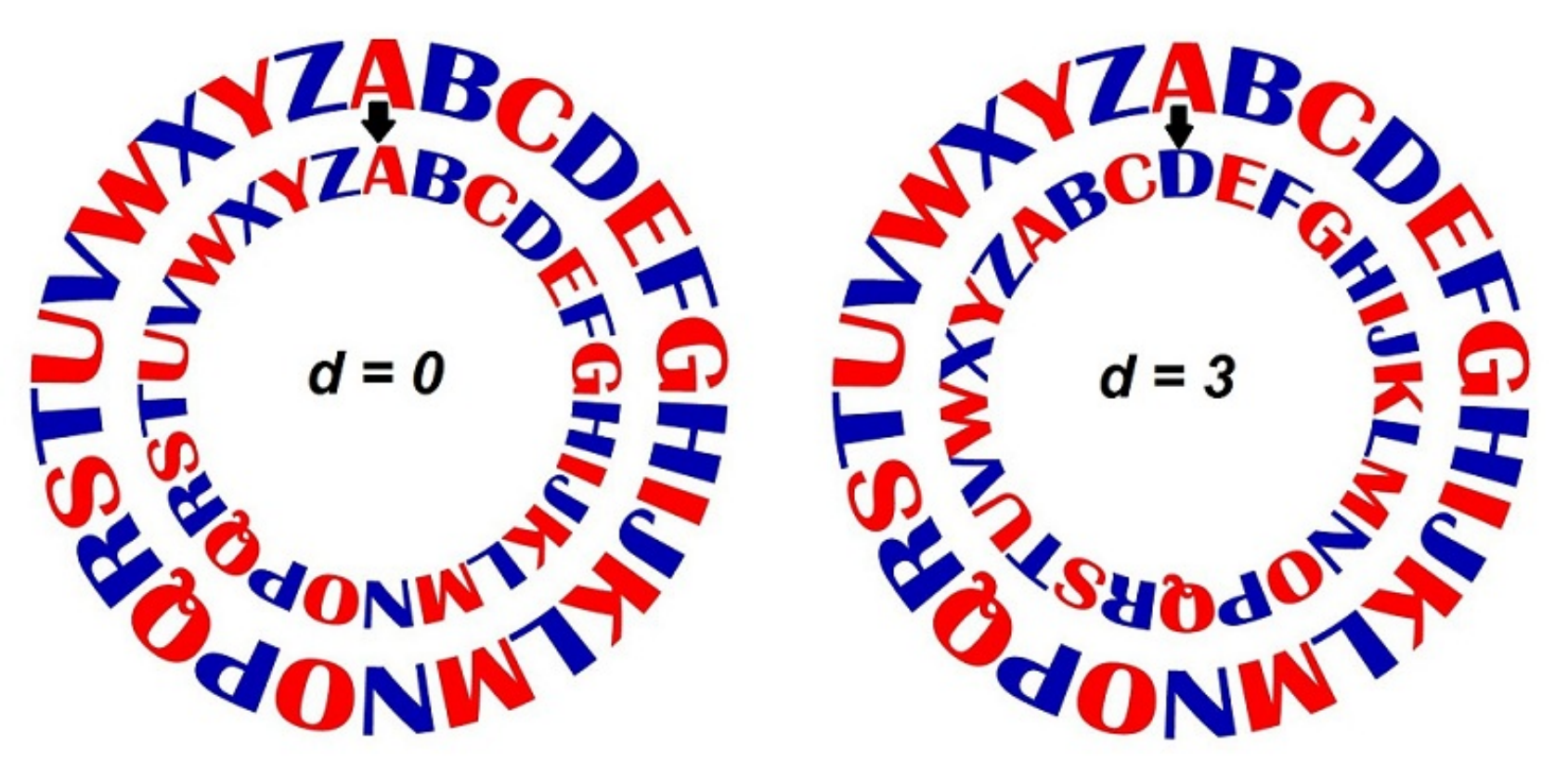


Figura 1: César como duas rodas deslocadas

Transposição (das letras do Texto Claro)

A *cítala* ou o *bastão de Licurgo* (= legislador de Esparta) é um bastão com o qual os espartanos cifravam como segue:

- enrolar o bastão com um pano estreito,
- escrever neste pano no sentido da borda maior, e
- desenrolar o pano do bastão.

As letras assim transpostas no pano unicamente podiam ser decifradas por um bastão com

- a mesma circunferência (e o mesmo comprimento),

pela mesma maneira como o texto foi cifrado:

- enrolar o bastão com um pano,
- ler este pano no sentido da borda maior.

A chave é o número dado pelo **número das letras que cabem nesta circunferência**.

Figura 2: A cítala enrolada por um cinto de couro

Algoritmos Modernos

Os algoritmos criptográficos simétricos modernos, tais que `DES` e `AES`, além

- da adição da chave (bit por bit por `ou exclusivo`),

aplicam ambas as operações,nam

- a **substituição** (isto é, permutação do *alfabeto*) e
- a **permutação** ou transposição (isto é, permutação do *texto*)

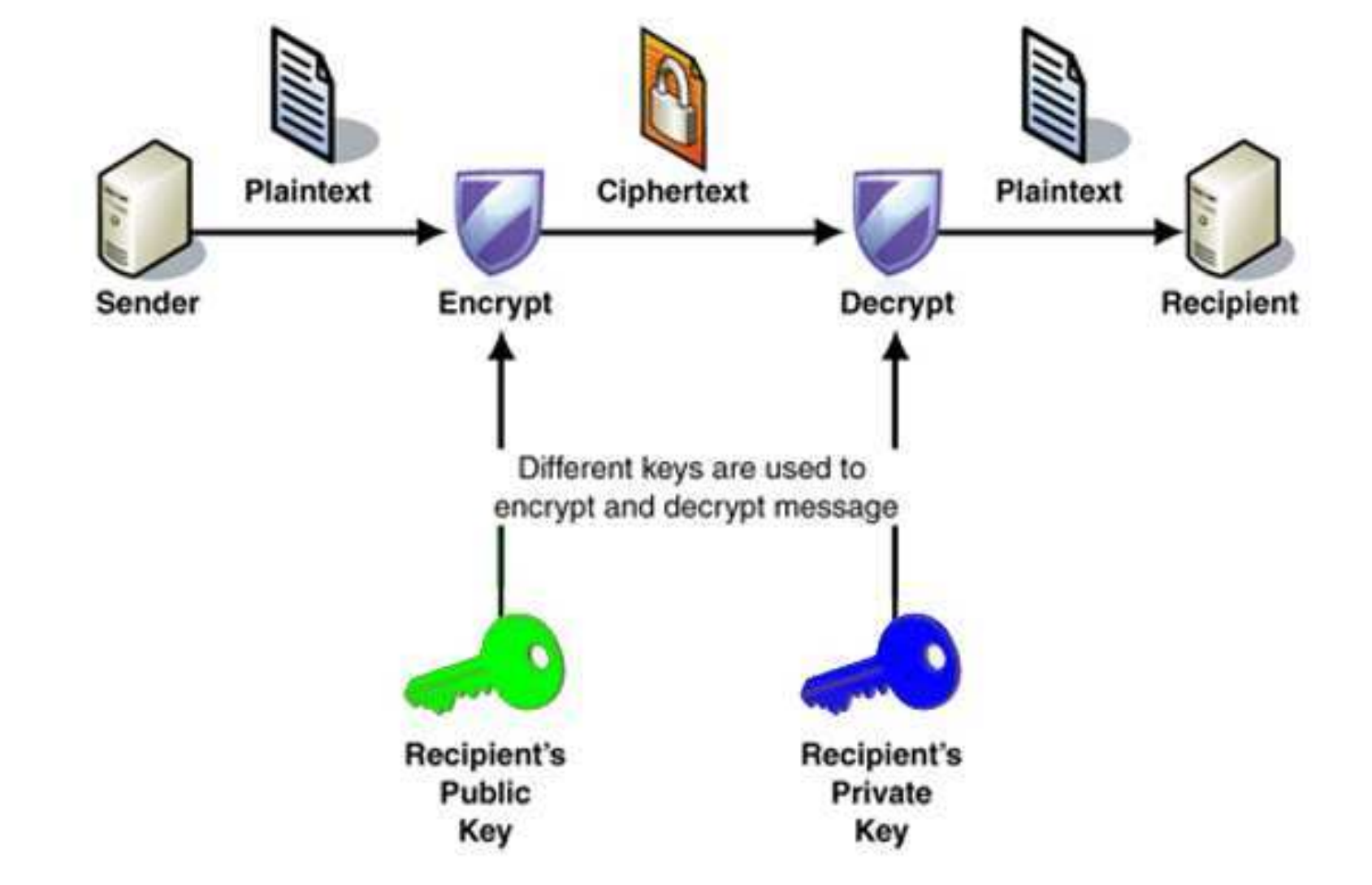
e iteram-nas (por exemplo o `AES` ≥ 10 vezes).

A chave é uma senha (transformada em uma cadeia de bytes de comprimento determinado por uma função [`hash`] criptográfica).

Figura 3: Segurando a rede sem fio pelo AES

Nos anos 70, surgiu a *criptografia assimétrica*, na qual

chave para cifrar ≠ chave para decifrar .



Chave pública e privada

Há duas chaves, uma chave **pública** e outra **privada**; e dois usos:

- **Cifrar e Decifrar**: A chave *pública* usa-se para cifrar, enquanto a chave *privada* para decifrar.

Isto serve a transmitir dados do cifrador exclusivamente ao decifrador.

Porém, os papéis das chaves públicas e privada podem ser invertidos:

- **Assinar e Verificar**: A chave *privada* usa-se para cifrar, enquanto a chave *pública* para decifrar.

Isto serve ao cifrador provar a todo decifrador a sua posse da chave privada; é a *assinatura digital*.

Inventores da Ideia da Criptografia Assimétrica

A criptografia assimétrica foi sugerida pela primeira vez, publicamente, em Diffie e Hellman (1976).

Figura 4: Diffie e Hellman

RSA = Rivest, Shamir e Adleman

Com efeito, Diffie e Hellman (1976) introduziu apenas um esquema para a criptografia assimétrica, mas não o pôs em prática. Isto foi feito pela primeira vez em Rivest, Shamir, e Adleman (1978), em que o algoritmo criptográfico RSA foi criado.

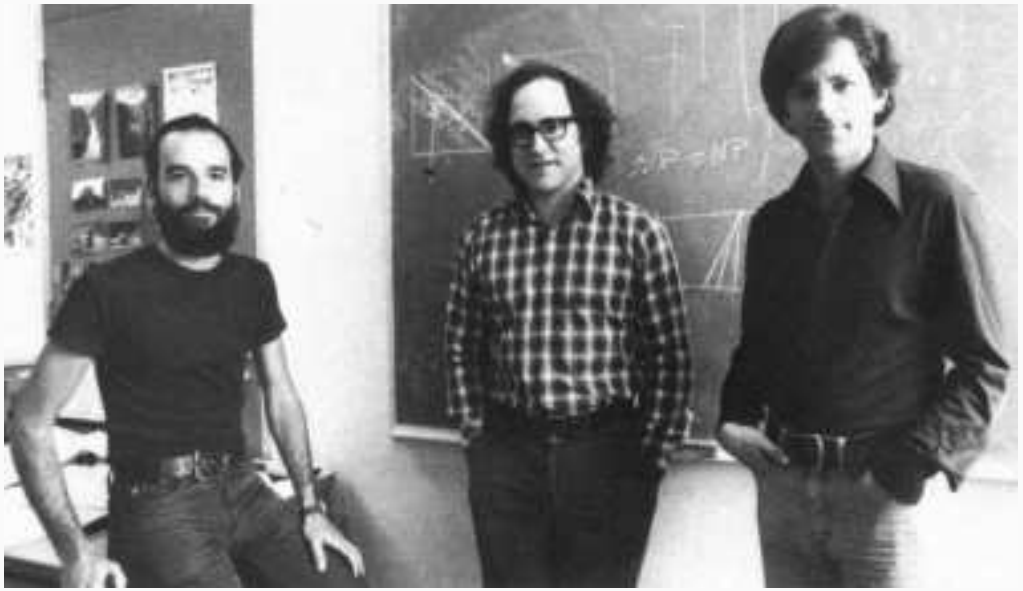


Figura 5: Os inventores do algoritmo RSA, Shamir, Rivest e Adleman

Tamanhos de Chaves Simétricas e Assimétricas

- Para os algoritmos simétricos como AES com uma chave só, uma chave de 16 bytes é suficiente.
- Para os algoritmos assimétricos como RSA, ambas as chaves tem comumente ≥ 256 bytes.

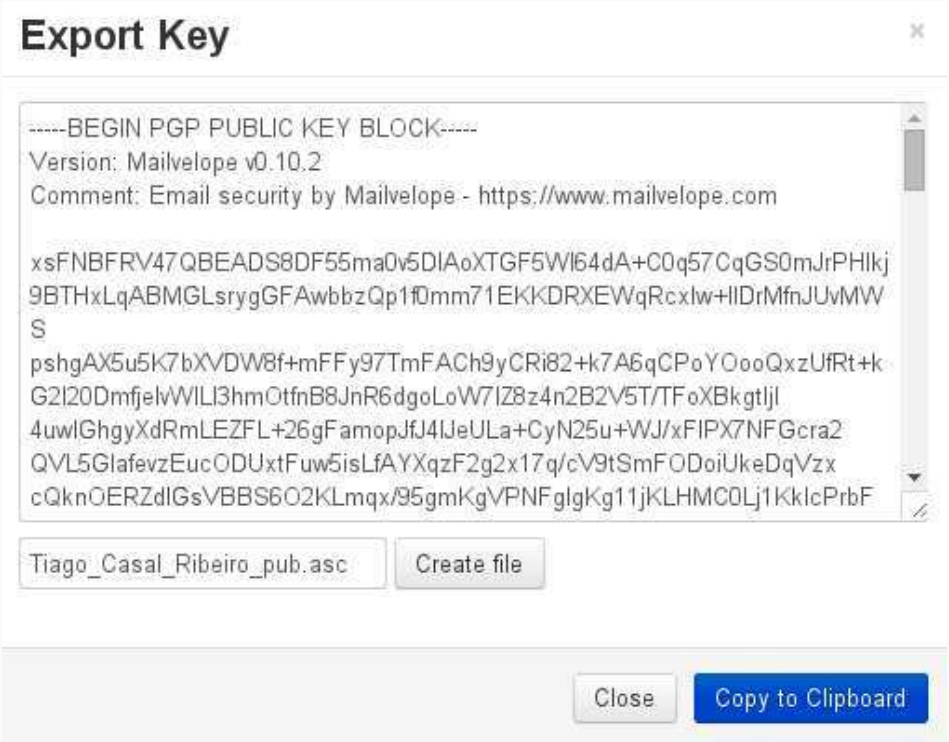


Figura 6: Chave Assimétrica

Aritmética Módulo 12 = Aritmética do Relógio

A criptografia assimétrica baseia-se na aritmética modular.

Figura 7: A adição no relógio é a da aritmética modular com módulo 12.

O dono desconhecido da chave privada

A criptografia assimétrica permite a Alice, que conhece uma chave pública,

- enviar um texto cifrado ao dono da chave privada, ou
- confirmar que um texto provenha do dono da chave privada.

Com o conforte que *Alice* pode obter a chave pública *impessoalmente*, sem encontrar o *Bob*, surge o problema da *identidade do dono da chave pública*: Considerando que Alice

- quer enviar um texto cifrado (ou confirmar a origem de um texto supostamente pertencendo) ao Bob, e
- não obteve a chave pública pessoalmente do Bob,

Questão

Como garantir a **identidade da chave privada**, isto é, que o dono da chave privada seja verdadeiramente o Bob?

O dono desconhecido!

Justamente pela falta da verificação do dono da chave privada que corresponde à chave pública, é vulnerável ao ataque “man-in-the middle” (MITM) em que o atacante se interpõe entre os correspondentes, assumindo a identidade de cada um, assim observando e interceptando todas as mensagens.

Figura 8: MITM

Supúnhamos que Maria Bonita intercepte todas as mensagens entre os correspondentes Alice e Bob:

1. Bob envia a sua chave pública à Alice. Maria intercepta-a, e envia à Alice a sua chave pública **própria** que alega Bob ser o seu dono!
2. Se Alice enviar uma mensagem ao Bob, ela usa, sem ter noção, a chave pública da *Maria*!
3. Alice cifra então uma mensagem com a **chave pública da Maria** e envia-a ao Bob.
4. Maria intercepta a mensagem, **decifra-a** com a sua chave privada; ela pode ler a mensagem e, se quiser, modificá-la.
5. Em seguida **cifra** a mensagem com a chave pública do Bob.
6. Bob decifra a sua mensagem com a sua chave privada.

Assim, Alice como Bob são persuadidos que usem a chave pública do outro, mas, na verdade, é a da Maria!

Protocolo ARP = Protocolo de Resolução de Endereços (do inglês Address Resolution Protocol) de 1982 padroniza a resolução - de endereços da camada de internet (IPv4) em - endereços da camada de enlace (Ethernet ou endereço MAC).

Ataque ARP poisoning (= poluição de cache ARP)

Maria Bonita quer interceptar as mensagens da Alice ao Bob:

1. Maria envia um pacote `arp who-has` à Alice que contem como endereço de IP fonte o do Bob cuja identidade queremos usurpar (ARP spoofing) e o endereço físico MAC da placa de rede de Maria.
2. A Alice criará uma entrada que associa o endereço MAC de Maria ao endereço IP do Bob.
3. Assim quando Alice comunicar com o Bob no nível IP, será Maria que receberá os pacotes da Alice!

Estabelecimento de Confiança

Esta garantia é estabelecida por *terceiros* = donos de chaves privadas que confirmam por suas assinaturas digitais que o Bob é o dono da chave privada.

O problema da identidade das chaves públicas surge outra vez:

Questão

Como garantir as identidades dos donos das chaves privadas?

Reposta

Há duas soluções:

- as *autoridades hierárquicas*, e
- a *teia de confiança*.

Autoridades Hierárquicas

- os donos de chaves privadas distinguem-se por níveis, e
- no topo jazem as *autoridades radicais* nas quais se confia incondicionalmente (como `VeriSign`, `GeoTrust`, `Comodo`).

Teia de Confiança

Os donos de chaves privadas não se distinguem, e e a ausência de entidades incondicionalmente confiáveis é compensada pela

- confiança estabelecida por ter obtido a chave pública pessoalmente; por exemplo,
  - em *key-sign parties*, encontros em que os participantes trocam e assinam as suas chaves públicas mutuamente ou
  - por ter comunicado a chave pública no telefone
- a qual se transfere de um ao outro, isto é, se Alice confia em Bob, e Bob confia em Carlos, então Alice confia em Carlos.

- as *autoridades hierárquicas*, e

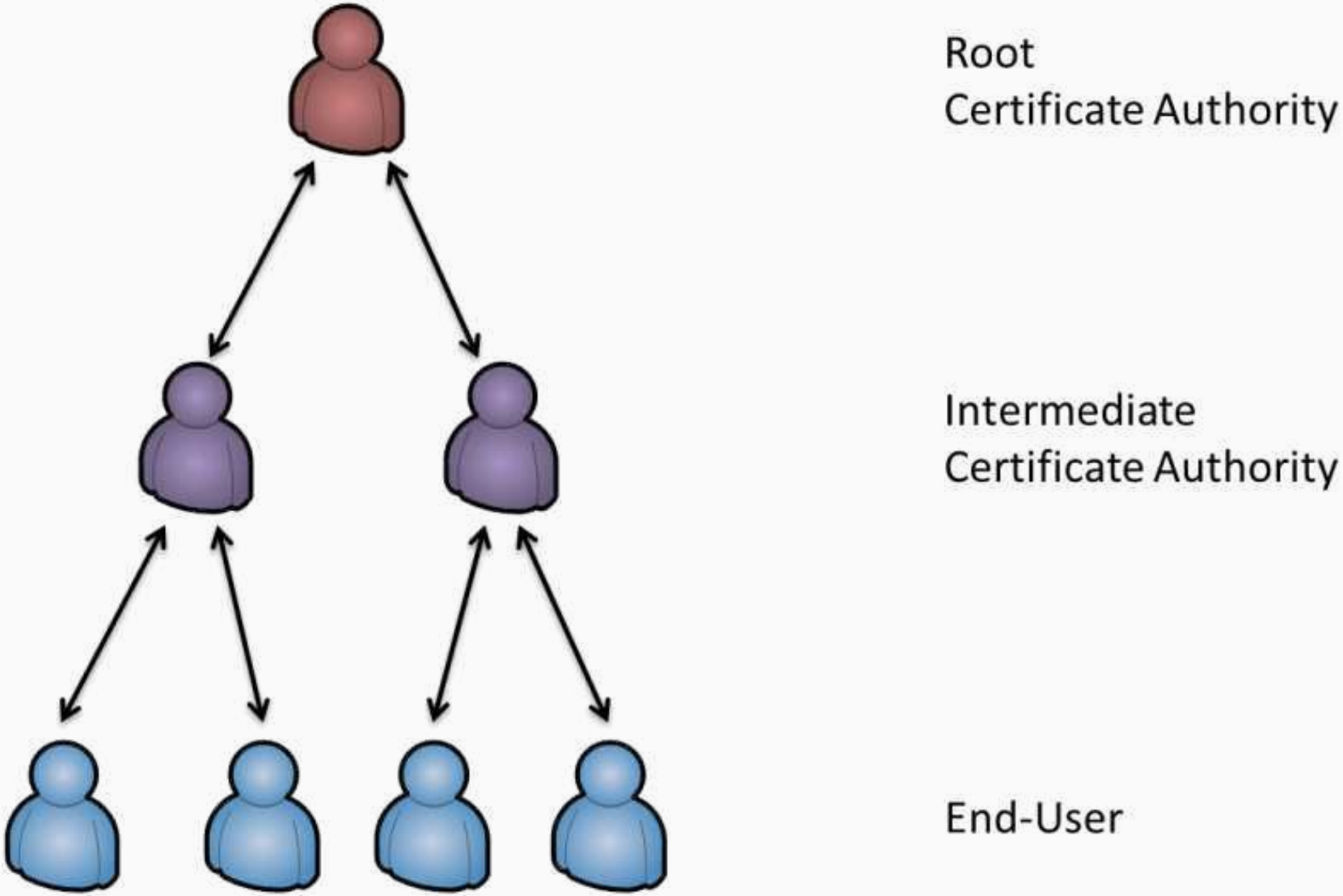


Figura 9: autoridades hierárquicas

- a *teia de confiança*.

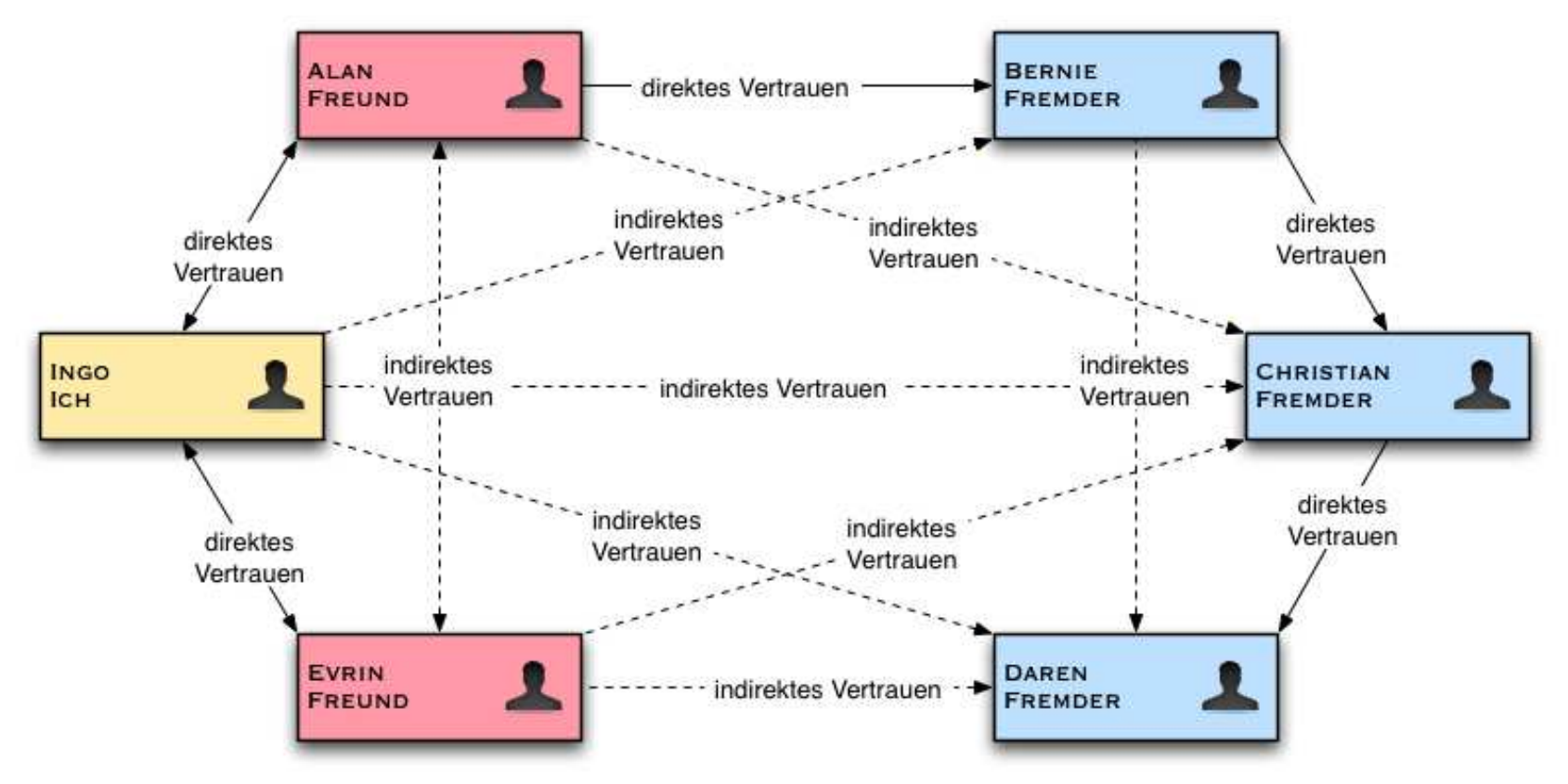


Figura 10: teia de confiança

Padronização dos Sistemas de Confiança

Na internet,

- o sistema de confiança por autoridades hierárquicas foi padronizado pelo esquema **`X.509`**, com o seu maior uso de **cifrar a comunicação entre um usuário e um site** (frequentemente comercial), e
- o pela teia de confiança pelo esquema **OpenPGP** (e principalmente implementado pelo programa GnuPG), com o seu maior uso de **cifrar os e-mails entre dois usuários**. Este esquema radicalmente rejeita qualquer hierarquia: o usuário pode publicar uma chave pública com um endereço de e-mail em um servidor de chaves públicas (por exemplo, em pgp.mit.edu) sem sequer confirmar (por um e-mail de ativação) que tem acesso à conta deste endereço de e-mail.

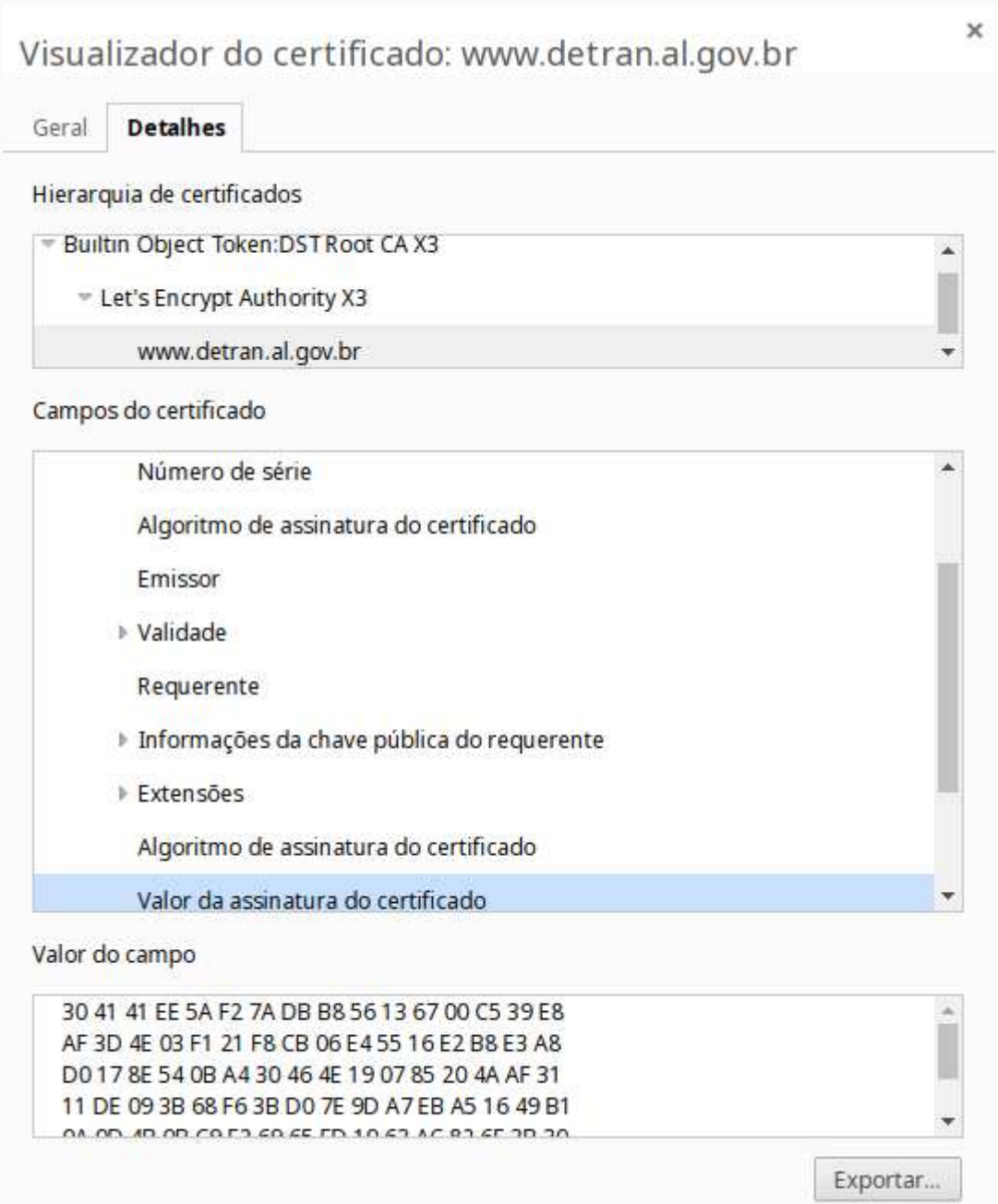


Figura 11: Certificado X.509 do DETRAN em Alagoas

Vantagens e Inconveniências do `X.509`

O sistema de *autoridades hierárquicas* tem como grande vantagem

- o conforto que a troca de chave poder ser automatizada;

como calcanhar-de-aquiles a confiança (absoluta) na autoridade (radical):

- a confiança absoluta que a chave pública pertence à autoridade,
- a confiança absoluta que a chave privada da autoridade não é comprometida.
- a autoridade pode abusar do seu poder, por exemplo:
  - cobrar indevidamente caro (o que levou a recente criação da autoridade livre Let's Encrypt que fornece certificados gratuitos (e tem um orçamento de 3 Milhões $; enquanto `VeriSign` cobra por cada certificado 399$ por ano),

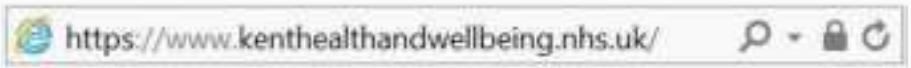
- não trabalhar seriamente, por exemplo, na verificação da identidade do terceiro pela autoridade. Com efeito, o nível de segurança reflete-se pela forma do cadeado na barra-de-endereço do navegador:
  - Na emissão de um certificado comum, `domain certificate`, a verificação é completamente automatizada; nenhuma verificação offline é feita. Basta ter acesso ao domínio para obter o certificado.
  - Na emissão de um `extended certificate` a verificação do proprietário do site é feita pessoalmente.

EV SSL in Chrome

EV SSL in IE

Figura 12: Certificado Avançado

DV SSL in Chrome



DV SSL in IE

Figura 13: Certificado Comum

Por isso, para ter certeza que o site pertence a quem pretende (por exemplo, evitar a confusão entre `bancobrasil.com.br` e `bancobrazil.com.br`), é importante

- verificar na barra-de-endereço que o cadeado indica um `extended certificate`; por exemplo, os navegadores `Firefox` e `Chrome` indicam-no pela cor verde do nome da entidade.
- Pelo cadeado comum que corresponde a um `domain certificate`, o usuário somente tem certeza que comunica com o dono do domínio; mas não que ele pertence à empresa ou organização que o site aparenta representar.

Vantagens e Inconveniências do OpenPGP

O sistema da *teia de confiança*

... tem como grande vantagem que

- é um sistema par-a-par; é independente de qualquer autoridade ou terceiro particular;

e como grande inconveniência que

- precisa de manutenção pessoal.

Fraca Repercussão

Como historicamente repetidamente provado, o conforto ganha:
Enquanto

- o esquema `X.509` é ubíquo no comércio eletrônico,
- pouquíssimos usuários usam o esquema OpenPGP.

O protocolo X.509 na prática

Relatamos os primeiros passos entre um cliente e o servidor, por exemplo, de um site de comércio eletrônico, para estabelecer uma conexão cifrada (por exemplo, para receber os detalhes do cartão de crédito do cliente):

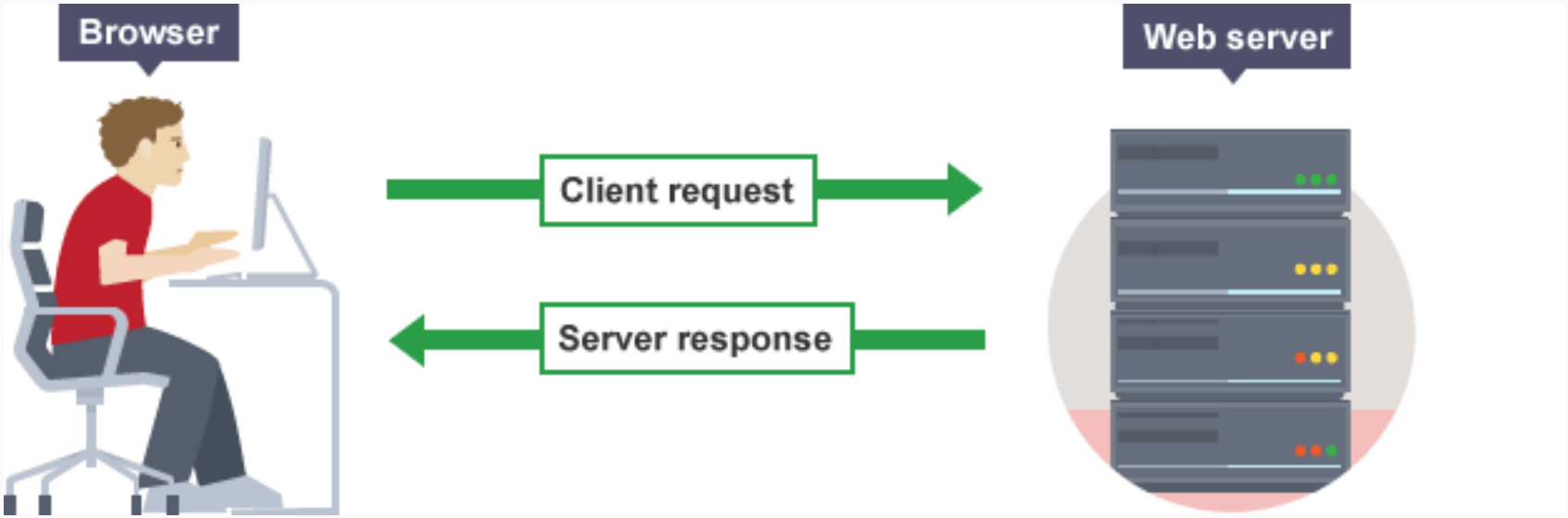
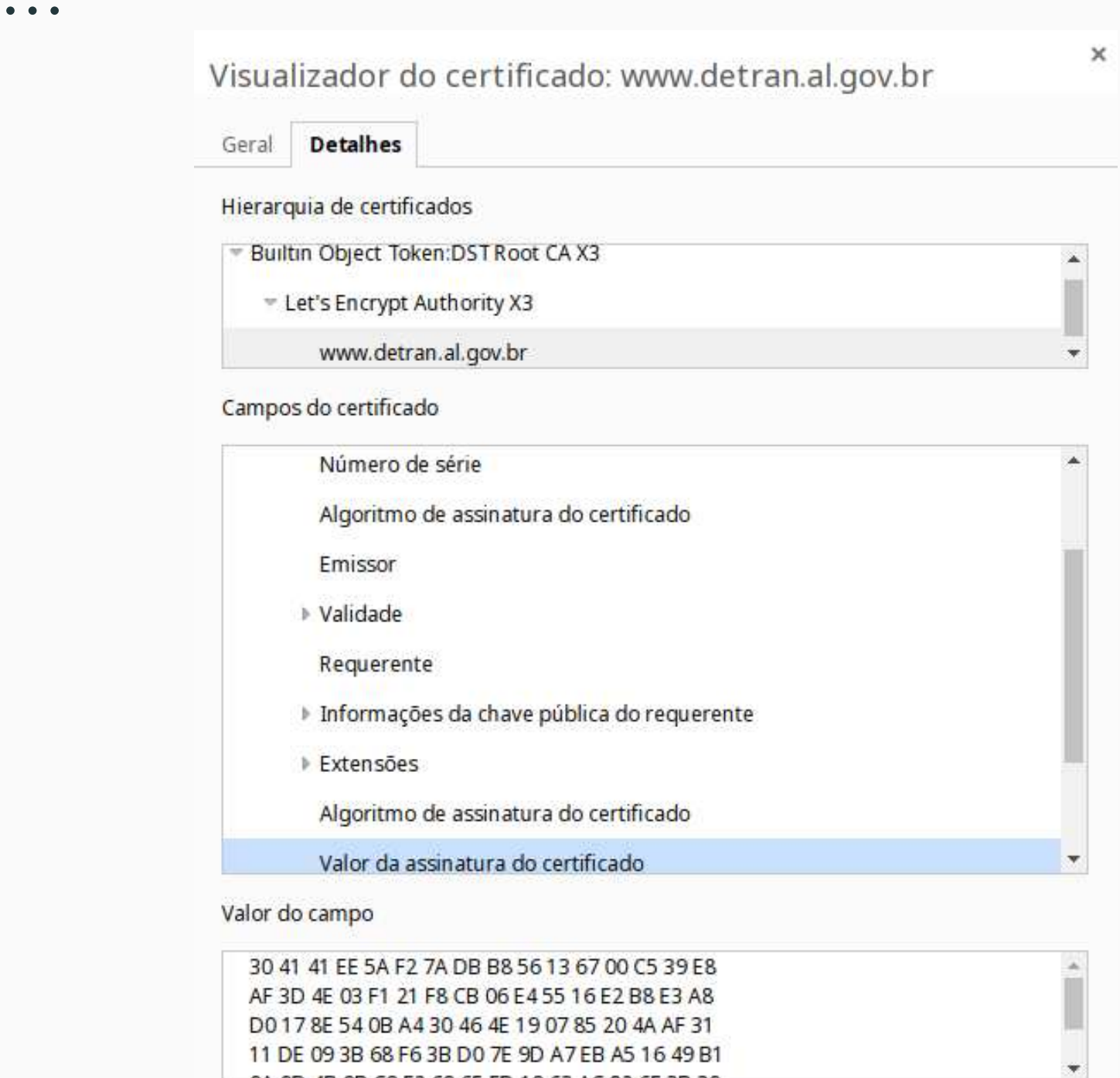


Figura 14: Cliente e Servidor

...

1. O cliente pede
   - usar uma conexão segura com o servidor, e para isto
   - propõe vários pares de algoritmos criptográficos: o primeiro **assimétrico** (`ECDSA`, `RSA`, ...) e o segundo **simétrico** (`AES`, `Camellia`, `RS4`, ...).
2. O servidor
   - escolhe um par entre os propostos pelo cliente, por exemplo, o par `RSA` com `RS4`. (Frequentemente, a escolha não é a mais segura, mas computacionalmente a mais económica.)
   - manda o seu **certificado**:


Visualizador do certificado: www.detran.al.gov.br
Geral
Detalhes
Hierarquia de certificados
Builtin Object Token:DST Root CA X3
Let's Encrypt Authority X3
www.detran.al.gov.br
Campos do certificado
Número de série
Algoritmo de assinatura do certificado
Emissor
Validade
Requerente
Informações da chave pública do requerente
Extensões
Algoritmo de assinatura do certificado
Valor da assinatura do certificado
Valor do campo
30 41 41 EE 5A F2 7A DB B8 56 13 67 00 C5 39 E8
AF 3D 4E 03 F1 21 F8 CB 06 E4 55 16 E2 B8 E3 A8
D0 17 8E 54 0B A4 30 46 4E 19 07 85 20 4A AF 31
11 DE 09 3B 68 F6 3B D0 7E 9D A7 EB A5 16 49 B1

...

O certificado contém (entre outros, mas sobretudo):

- o **endereço do servidor** e **a sua chave pública**
- um nome de **uma autoridade** (radical, por exemplo, `VeriSign`), e **a sua assinatura digital** (= a cifração pela chave privada da autoridade da soma de verificação
  - do endereço do servidor, e
  - da chave pública do servidor.)

...

3. O cliente procura a **chave pública da autoridade** (radical) indicada no certificado (que é usualmente inclusa no navegador), e usa-a para decifrar esta assinatura digital. Se o resultado é a soma de verificação esperada (isto é, do endereço do servidor e da sua chave pública), então
   - a assinatura digital provém verdadeiramente da autoridade indicada, e
   - a autoridade confia neste servidor.

   Como o cliente (ou, mais exatamente, o seu navegador) confia incondicionalmente nas autoridades radicais, a este ponto ele tem certeza que a chave pública pertence verdadeiramente ao servidor visado.

...

4. O cliente
    - cria aleatoriamente uma chave secreta para o algoritmo **simétrico** combinado antes com o servidor, por exemplo, RC4, e
    - primeiro recheia-a (*padding*) e depois cifra-a pelo algoritmo **assimétrico** combinado, para finalmente mandá-la ao servidor.
5. O servidor
    - modifica por seu lado a chave secreta do cliente por acrescimento dos valores aleatórios que foram trocadas no início da conexão (no “hello” do cliente e servidor), para garantir que a chave que será usada entre eles se refere unicamente a esta conexão, e
    - cifra-a com a chave secreta do cliente para mandá-la de volta ao cliente, e poderem começar a comunicação secreta.

O que é GnuPG?

O programa GnuPG é um programa de *linha-de-comando*.

Objetivo

Oferecer ao grande público *aberta e gratuitamente* métodos criptográficos para cifrar dados eletrônicos confidenciais.

Funções Principais

Um programa de linha-de-comando para

- cifrar e decifrar de dados (por exemplo, e-mails), e
- criar e verificar assinaturas digitais (para garantir autenticidade e integridade dos dados).

Aceitação

- Pre-instalado na maioria das distribuições de Linux, e
- disponível sob Mac OS e Microsoft Windows

História

1997. O desenvolvimento do programa `Gnu Privacy Guard` (= `GPG`) pelo alemão *Werner Koch* começou (e até hoje não cessou) para ter uma alternativa livre ao programa de criptografia de e-mail comercial `Pretty Good Privacy` (= `PGP`) por Phil Zimmermann.

1999. A versão 1.0 foi completada.

2000. O Ministério Federal Alemão de Economia e Tecnologia patrocinou a portabilidade para o Microsoft Windows.

2006. A versão 2.0 foi anunciada e trouxe mudanças significativas na arquitetura do programa.

Werner Koch

Figura 16: Werner Koch

```
--- ~ » gpg2 --full-generate-key
Por favor selecione o tipo de chave desejado:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (apenas assinatura)
   (4) RSA (apenas assinatura)
Opção? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
O tamanho de chave pedido é 2048 bits
Por favor especifique por quanto tempo a chave deve ser válida.
         0 = chave não expira
      <n>  = chave expira em n dias
      <n>w = chave expira em n semanas
      <n>m = chave expira em n meses
      <n>y = chave expira em n anos
A chave é valida por? (0) 1y
Key expires at Ter 02 Abr 2019 13:11:59 -03
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Nome completo: Enno
O nome deve ter pelo menos 5 caracteres
Nome completo: Enno Nagel
Endereço de correio eletrónico: biz.nagel@t-online.de
Comentário:
Você selecionou este identificador de utilizador:
    "Enno Nagel <biz.nagel@t-online.de>"

Mudar (N)ome, (C)omentário, (E)ndereço ou (O)k/(S)air? O
Precisamos gerar muitos bytes aleatórios. É uma boa ideia realizar outra
actividade (escrever no teclado, mover o rato, usar os discos) durante a
geração dos números primos; isso dá ao gerador de números aleatórios
uma hipótese maior de ganhar entropia suficiente.
_
```

Figura 17: Criação de um par de chaves no GPG na linha-de-comando

Cria um par de chaves, uma pública e a outra privada, escolhendo

- o algoritmo (por exemplo, RSA),
- o tamanho (por exemplo, 2048 bits),
- a validade (por exemplo, um ano),
- uma senha para a chave privada, e
- a identidade: o nome e endereço de e-mail do dono.

A chave *pública* é destinada à divulgação. Ela serve

- a cifrar e
- a verificar assinaturas.

A chave *privada* é guardada e protegida por uma senha. Ela serve

- a decifrar e
- a assinar.

Enigmail

- O programa Enigmail é uma extensão para o programa de e-mail gráfico Thunderbird que adiciona a este a função de
  - criptografar,
  - descriptografar,
  - assinar e
  - verificar assinaturas de e-mails,

  pelo GnuPG, assim que o usuário pode aceder a estas funções por botões no próprio Thunderbird.

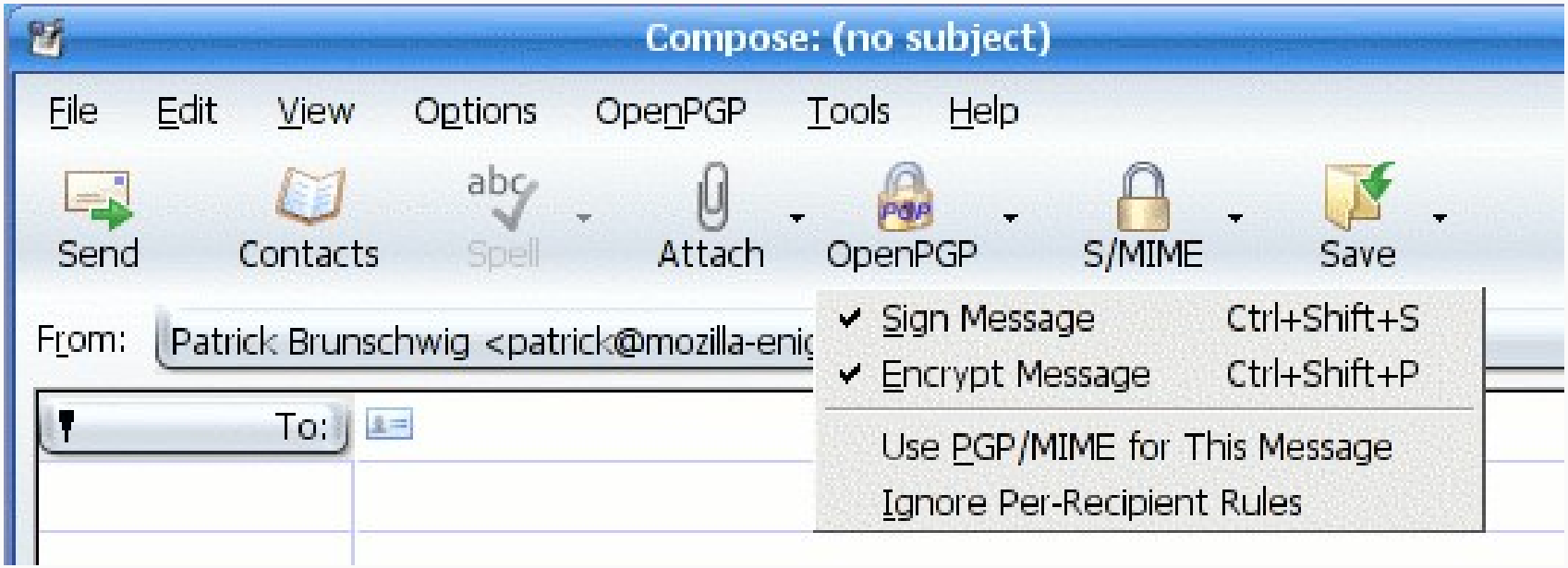


Figura 18: Enigmail

Mailvelope

O Mailvelope é uma extensão para os navegadores Firefox e Chrome, desenvolvido por uma empresa alemã, que adiciona recursos de criptografia e descriptografia à interface de provedores comuns de webmail como:

- Gmail
- Hotmail.com
- Yahoo!

Por exemplo, cifra e decifra mensagens (usando o padrão OpenPGP), arquivos em seu disco rígido e envie anexos de e-mail criptografados.

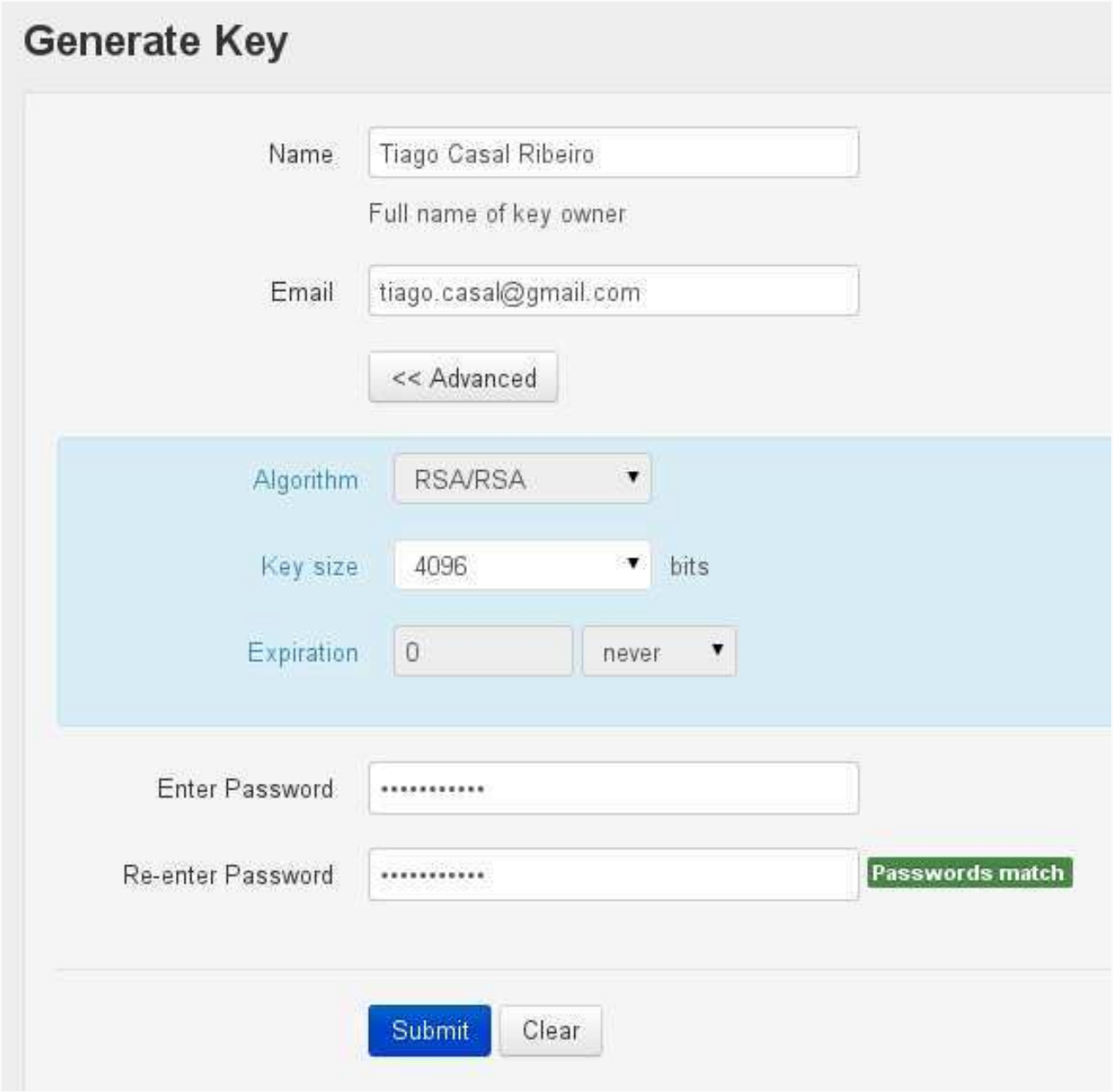


Figura 19: Criar uma chave pública pelo `Mailvelope`

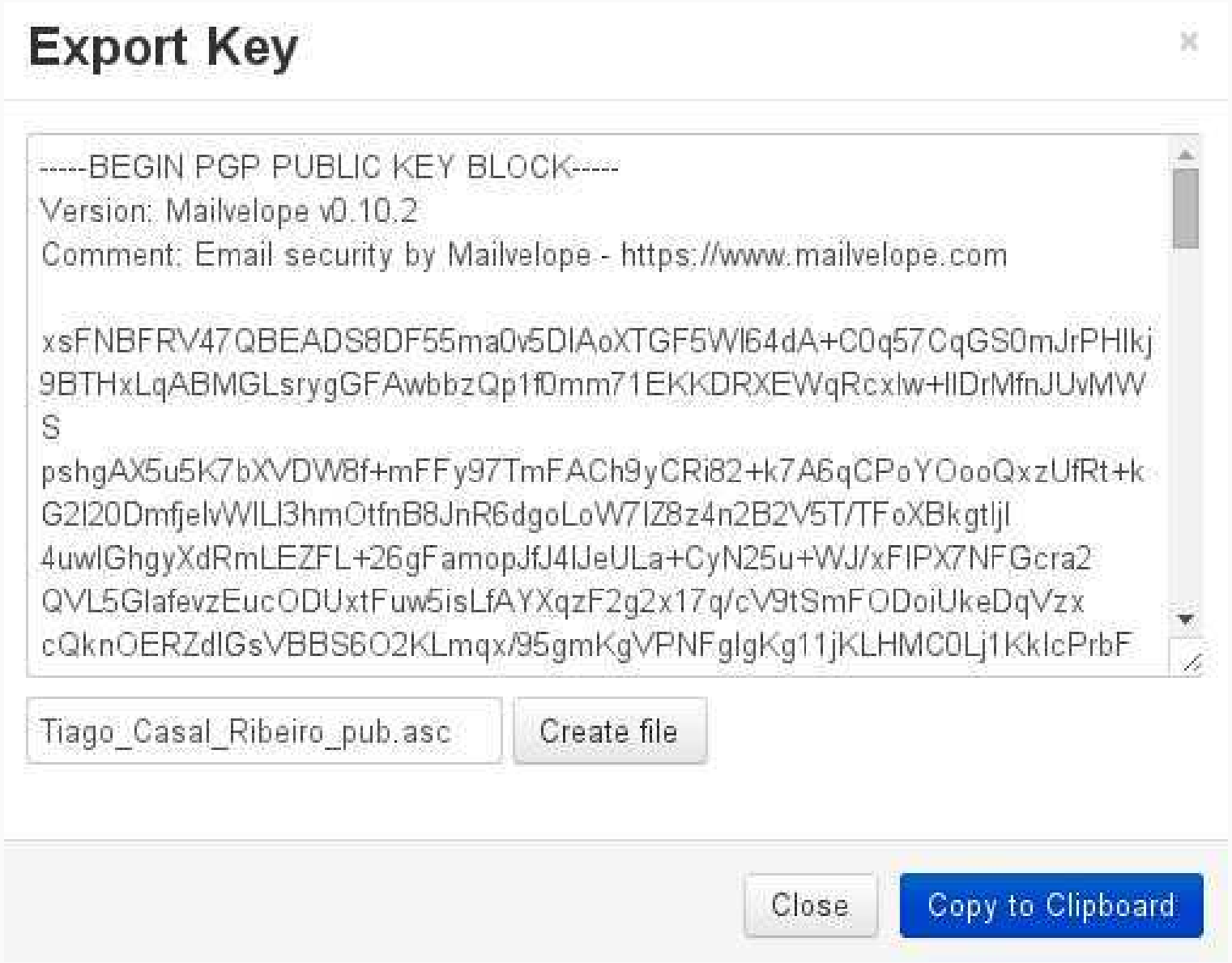


Figura 20: Uma chave pública criada pelo `Mailvelope`

Automatizar a Troca de Chaves

Todo programa que será apresentado (`Autocrypt` ou `prettyeasyprivacy` ou `Delta-Chat`) não é uma solução perfeita, mas conforme a `RFC 7435` (Request for Comments, padrão livremente criado pelos usuários na internet) oferece unicamente :

“Segurança Oportunista: Alguma proteção na maioria do tempo”.

Com efeito, não oferece nenhuma proteção contra o man-in-the-middle! Para evitá-lo, tem de verificar pessoalmente com o dono a sua chave pública pelo seu “**fingerprint**” (= impressão digital = uma soma de verificação criptográfica)!

Extensões de Troca de Chaves automatizada

Autocrypt

Na versão 2.0 a extensão Enigmail tem suporte para o programa Autocrypt que automatiza a troca de chaves públicas: insere uma linha adicional no cabeçalho do e-mail (que é normalmente invisível para o usuário) que contém o certificado (nome, endereço de e-mail e referencia à chave pública do usuário).

prettyeasyprivacy

Outro projeto para mandar e-mails cifrados automaticamente, fundado por uma iniciativa privada, semelhante a Autocrypt, que dá suporte a programas comerciais como Microsoft Outlook é prettyeasyprivacy.

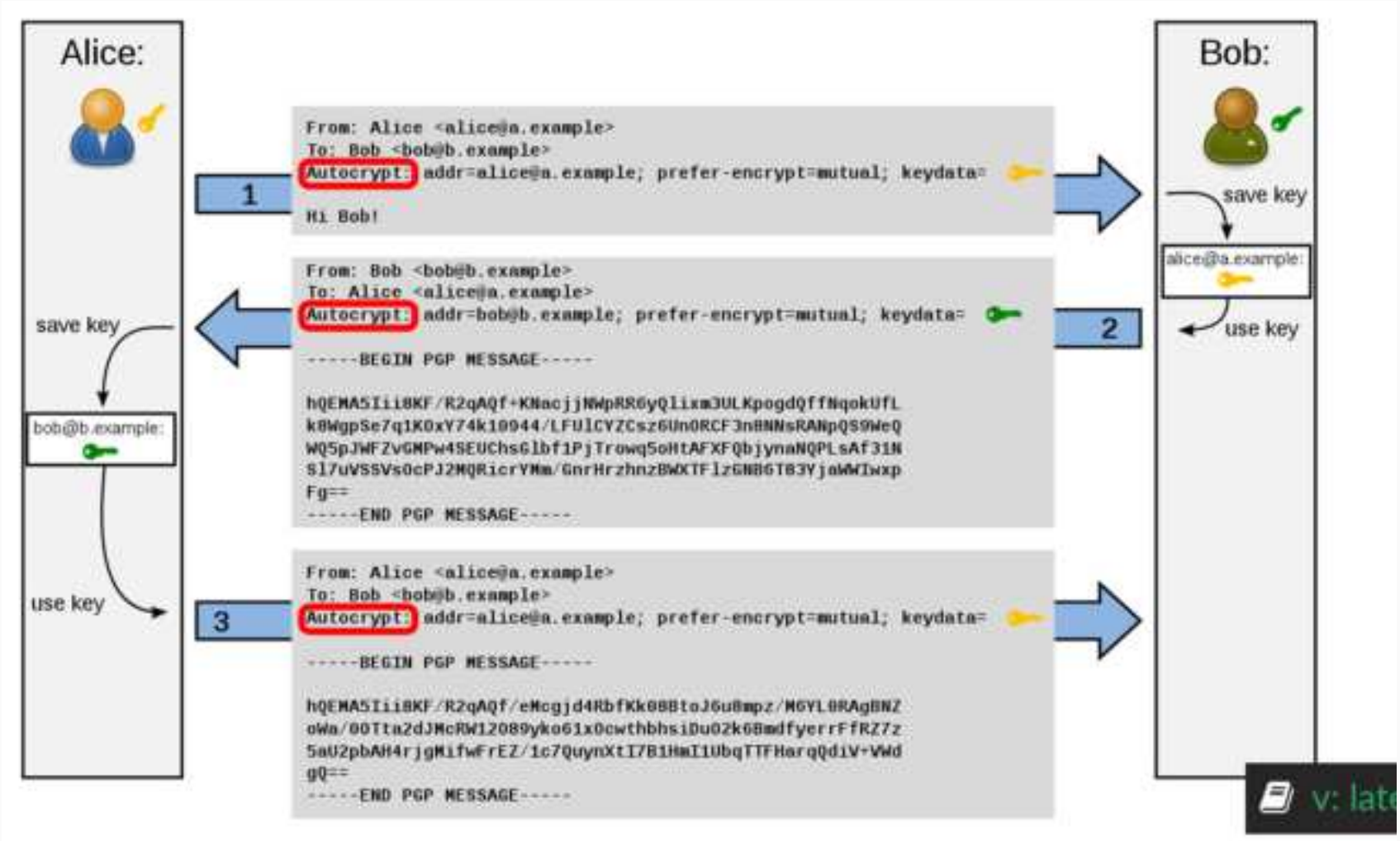


Figura 21: Funcionamente de Autocrypt

Delta-Chat

Para mandar mensagens instantâneas automaticamente cifradas o aplicativo Delta-Chat usa a conta de e-mail do usuário.

Figura 22: Delta-Chat

Problemas Gerais

Como usa o mesmo protocolo aberto como o e-mail, em comparação a muitos outros desses aplicativos (como `WhatsApp`), não depende dos servidores de uma empresa especifica e evita que:

- o lucro vem da compilação dos meta-dados dos usuários armazenados nos seus servidores (como `facebook.com`),
- o governo poderia suspender ou bloquear estes servidores (como no caso de `WhatsApp` aconteceu no Brasil e na China),
- os servidores da empresa podem ser comprometidos,
- a empresa muda de ideia sobre o modelo empresarial, e
- o usuário tem que ter confiança total nesta empresa que para quais todo mundo é uma grande entidade anônima.

Vantagens `Delta-Chat`

- Independente de companhias e serviços. Você é o dono dos seus dados.
- Seus dados não são guardados num servidor central; desta forma, em contraste com a maioria de outros aplicativos do gênero, o Delta Chat protege até mesmo seus metadados (quem escreve para quem?)
- Você não entrega sua agenda de contatos para ninguém.
- Compatível — destinatários que não usam o `Delta Chat` podem ser contatados por e-mail também

Cifração Meta-Dados

Isto dito, os meta-dados entre os servidores de e-mail pelo antigo protocolo `IMAP` *não* são cifrados.

O aplicativo Conversations propõe um protocolo moderno `XMPP` que cifra também os meta-dados.

É uma solução muito completa quanto à segurança; infelizmente pouco estabelecida: por exemplo, em comparação ao `Delta-Chat`,

- há muitos servidores de e-mail (isto é, que comunicam pelo protocolo `IMAP`),
- porém pouquíssimos que entendem `XMPP`.

Isto dito, o único meio para controlar os próprios meta-dados é rodar o seu próprio servidor.

Subchaves Efêmeras

Para mais segurança, o dono frequentemente usa uma chave (pública e privada) principal unicamente para emitir outras chaves assinadas, **subchaves**, com datas de validade:

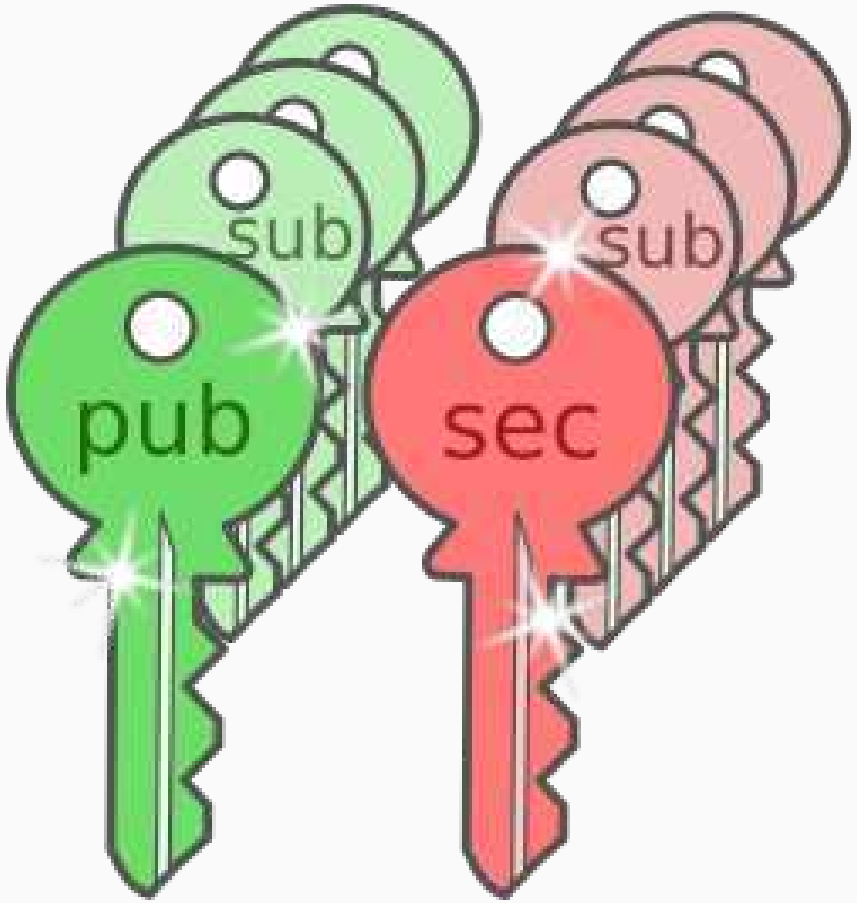


Figura 23: subchaves

Perfect Forward Secrecy.

No passo 2. da troca entre o cliente e servidor, o servidor manda além do seu certificado uma chave pública **efêmera** = uma chave (pública e privada) aleatoriamente criada que

- serve para **cifrar a comunicação**,

enquanto a sua chave principal (**permanente**) que é assinada pela autoridade (radical)

- serve para **assinar a sua chave efêmera**,

para o cliente confiar nela. Após a conexão com o cliente, a chave efêmera é apagada.

$\Longrightarrow$ o comprometimento da chave privada do servidor (= sua identidade digital) não afeta a cifração da conexão com o cliente.

Cadeia das Assinaturas das Chaves

Chave Privada da Autoridade Radical

$\Downarrow$

(Chave da Autoridade Intermediária)

$(\Downarrow)$

Chave do Servidor

$\Downarrow$

Chave Pública Efêmera

$\Longrightarrow$

O usuário confia na chave pública efêmera.

Funcionamento GPG

O programa GnuPG é um programa de **linha-de-comando** que cria um par de chaves, uma *pública* e a outra *privada*, escolhendo

- o *algoritmo* (por exemplo, RSA),
- o *tamanho* (por exemplo, 2048 bits),
- a *validade* (por exemplo, um ano),
- uma *senha* para a chave privada, e
- a *identidade: o* nome e *endereço de e-mail* do dono.

A chave *pública* é destinada à divulgação. Ela serve

- a cifrar e
- a verificar assinaturas.

A chave *privada* é guardada e protegida por uma senha. Ela serve

- a decifrar e
- a assinar.

Subchaves GPG

Por exemplo, (é o que o GnuPG faz) o dono de uma chave privada cria

- uma subchave para **decifrar** no dia-a-dia (por exemplo, os e-mails cifrados recebidos), e
- uma subchave para **assinar** no dia-a-dia (por exemplo, os seus e-mails enviados);

com data de **validade**. Antes das suas invalidações, prolonga ou revoca-as e cria outras.

$\Longrightarrow$ Se uma subchave for comprometida, o dono **revoca**-a, ele publica um informe, digitalmente assinado pela sua chave principal privada, sobre a invalidação da subchave.

Unicamente a chave **principal** é **imutavelmente ligada** a identidade do dono, e todas as **outras** são **substituíveis**.

Melhores Práticas para a Gerência das Chaves

- A **chave principal** é guardada em um **cofre** em casa e sobretudo só vê a luz quando precisar de assinar chaves;
- As **subchaves** são guardadas em um **cartão inteligente** (smartcard) que se acede por um leitor de USB com o seu próprio teclado. Em comparação ao uso de um arquivo digital, ele tem a vantagem que
  - a leitura das chaves de um cartão inteligente é muito mais difícil do que de um arquivo guardado (num `pendrive` ou `HD`)
  - pode ser portado facilmente, e
  - deixa menos traços:
    - Nunca revela a chave mas somente prova que a possua, e
    - é imune contra `keyloggers` que gravam as letras tecladas.

Figura 24: Leitor de Cartão Inteligente

Anotações

Estão **disponíveis**

- os **eslaides** desta palestra e
- um **manuscrito** sobre criptografia que aprofunda o que aprendemos

online em `konfekt.bitbucket.io/talks/criptografia`

Referências

Diffie, Whitfield, e Martin Hellman. 1976. «New directions in cryptography». *IEEE transactions on Information Theory* 22 (6). IEEE: 644–54.

Rivest, Ronald L, Adi Shamir, e Leonard Adleman. 1978. «A method for obtaining digital signatures and public-key cryptosystems». *Communications of the ACM* 21 (2). ACM: 120–26.

1. Criptografia Assimétrica
2. Confiança
3. Protocolos
4. Aplicativos OpenPGP
5. Subchaves