PUBLIC KEY CRYPTO WITH APPLICATIONS

Marc SCHAEFER

19th june 2002

Programme

– The general problem

– The available tools

– The concepts

– The problem of Trust

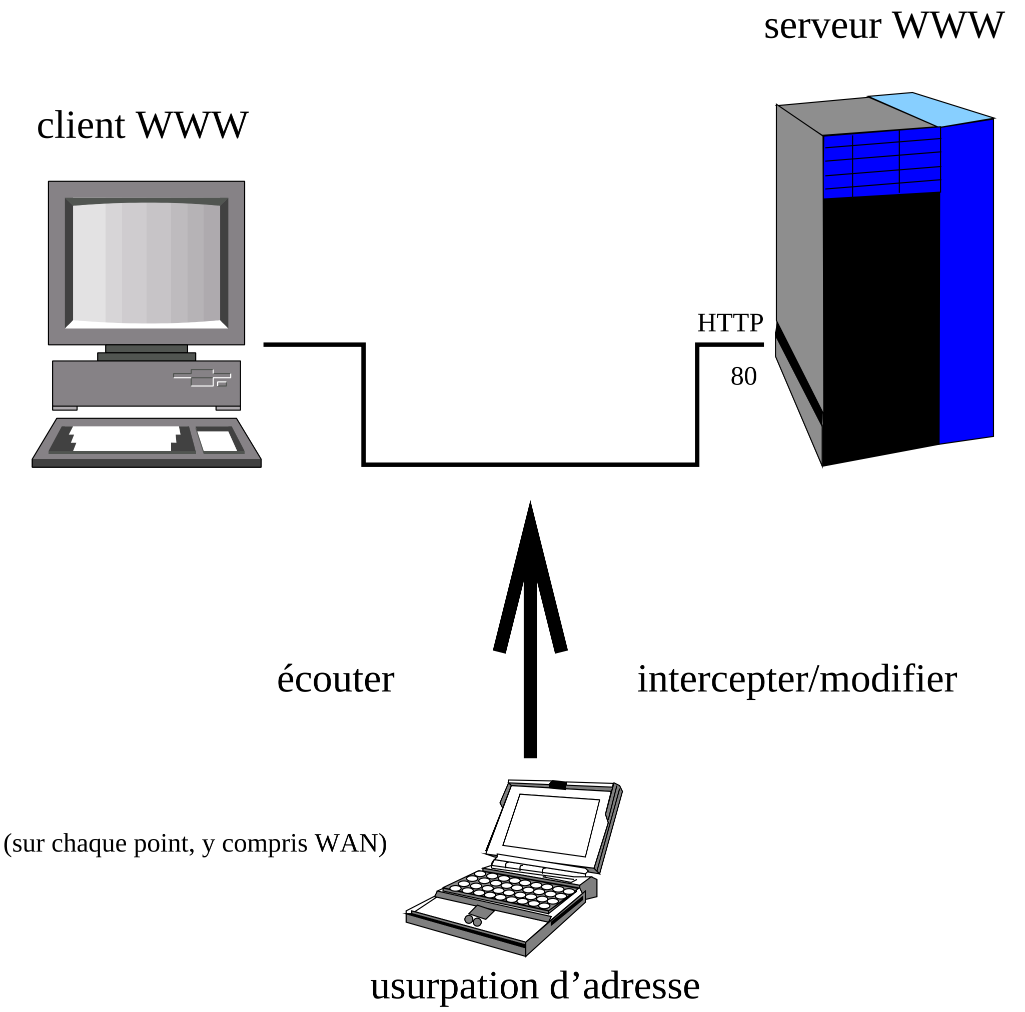
– Other attacks

– PKI

– Exchange your keys !

– Questions

The general problem



The problem

- The communication channel isn't safe (in this example : Internet and the HTTP protocol are not safe).
- Attackers can prevent data from arriving, modify them, generate new data and read them.

Targets

- "New" ways of using Internet
- electronic payment
- personal/medical data
- cracking
- social engineering

The required functionality

We need ways to ensure some or all of :

- data comes from the advertised sender
- data is not modified while in transit
- only the wanted recipient can read the data
- no additional data can be generated
- data will not be prevented from arriving.

What users want

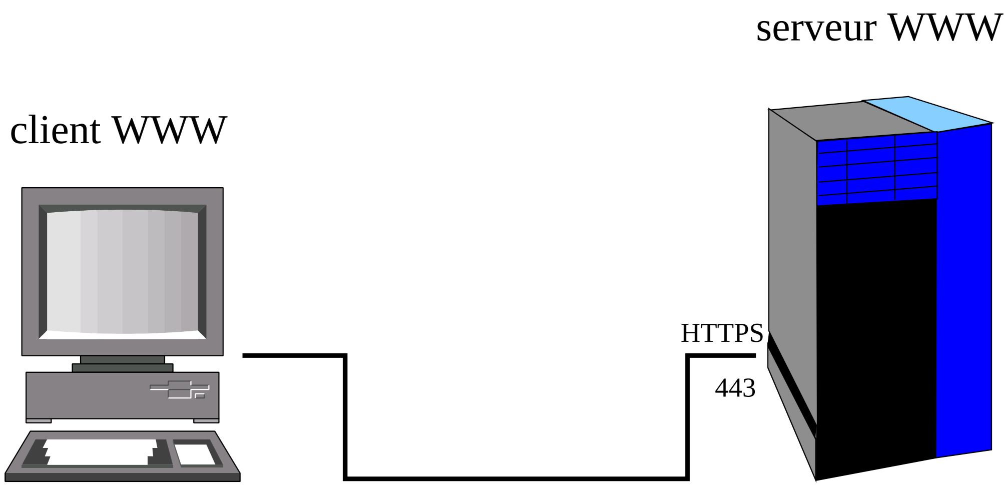
- ensure received data are reliable/true
- ensure personal/medical data stay confidential
- pay securely over the Internet (**Yellownet**, **Internet banking**)
- other future – and sometimes irrealist – applications : **e-voting**, **e-government**.

The available tools

WWW SSL/TLS : Secure Socket Layer (HTTP + SSL == HTTPS)

mail, data GPG/PGP : GNU privacy Guard / Pretty Good Privacy

Example : SSL/TLS



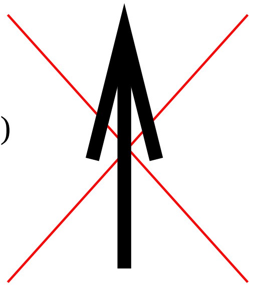
serveur WWW

client WWW

HTTPS

443

HTTPS = HTTP + SSL



authentification (identification)

encryption (chiffrement)

The technical solution When there are no implementation errors, no incorrect security procedures (permissions, passwords), it is today possible to build a safe information transfer infrastructure over the Internet, for example for WWW applications (HTTP).

Unfortunately, it is still not possible to ensure the absence of implementation errors or incorrect security procedures, and this will not change in the foreseeable future. Not only the client workstations are subject to many and regular attacks (using security holes or incorrect default configuration for example in Microsoft software), but servers too are vulnerable : for example credit card numbers are usually stolen in mass through electronic merchants.

Example : GPG/PGP

- encrypt/decrypt files (also symetric)
- sign files/verify signatures
- can be used together with e-mail software packages
- key servers, trust network

Main commands of GNU Privacy Guard (GPG)

generate a key pair	`gpg --gen-key`
generate a revocation certificate	`gpg --gen-revoke`
get key ID 7F76BFC9 from key server	`gpg --recv-keys 7F76BFC9`
send key ID 7F76BFC9 to key server	`gpg --send-keys 7F76BFC9`
get finger print from key	`gpg --fingerprint 7F76BFC9`
sign a key with yours	`gpg --sign-key 7F76BFC9`
verify a signature	`gpg --verify SIG-FILE FILE`
encrypt	`gpg -e < file > file.gpg`
decrypt	`gpg < file.gpg > file`

Configuration of GNU Privacy Guard (GPG) The configuration file is

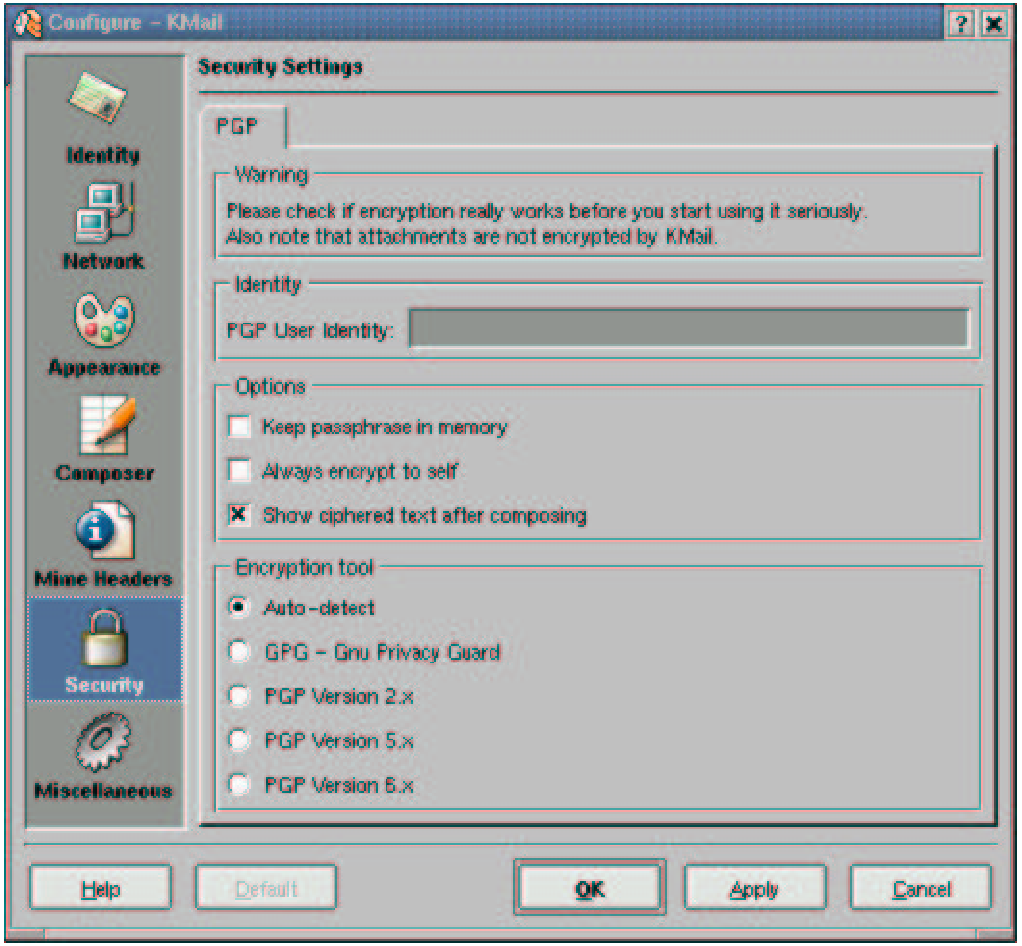
```
~/.gnupg/options
```

Some interesting options to set are :

`load-extension idea` Especially when handling older PGP keys. Warning : IDEA is patented ! Check the license of the package before using.

`keyserver www.ch.pgp.net` A key server to send/receive keys to/from.

`honor-http-proxy` Set this if you have an HTTP proxy.



Security Settings

PGP

Warning

Please check if encryption really works before you start using it seriously.
Also note that attachments are not encrypted by KMail.

Identity

PGP User Identity:

Options

□ Keep passphrase in memory

□ Always encrypt to self

☒ Show ciphered text after composing

Encryption tool

● Auto-detect

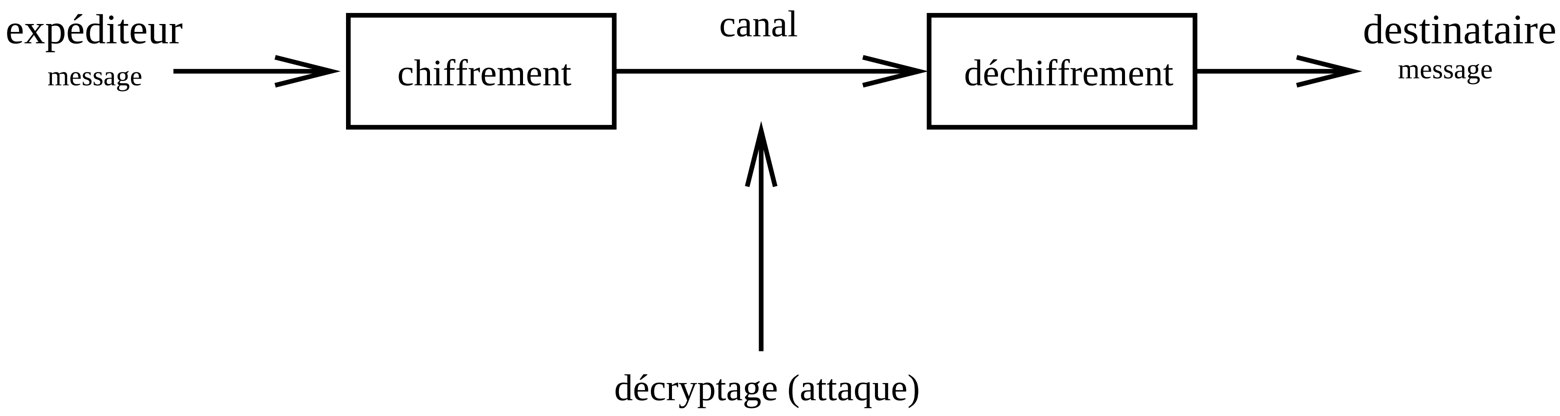
○ GPG - Gnu Privacy Guard

○ PGP Version 2.x

○ PGP Version 5.x

○ PGP Version 6.x

The concepts



expéditeur
message

chiffrement

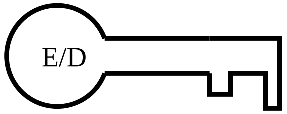
canal

déchiffrement

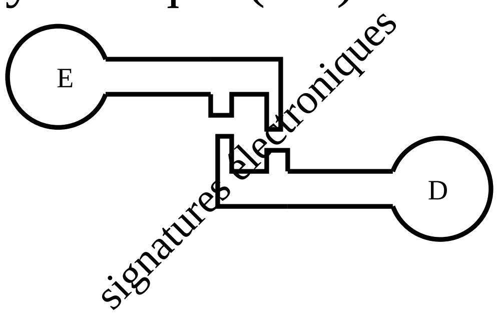
destinataire
message

décryptage (attaque)

chiffrement symétrique



asymétrique (PK)



Symetric cryptography (secret key) We use a key (secret, known to the emitter and the recipient of the message). This is infact a bijective transforming function which is to be applied to a message. The same function – or it's invert bijective function – is applied to decipher the message.

Properties :

- safe, well-known
- small key sizes are enough
- needs a pre-transmission (**safe !**) of the key
- key handling complex (one key per P2P link)

Asymetric cryptography (public key) Two keys are used (they are generated in a pair and are symetric/complement of each other). There is no *easy* way to go from one key to the other. One key can be used to cipher, and the other to decipher, and the other way. Only one of the keys is published, the other remains secret.

Properties :

- slow
- no safe channel necessary to transmit keys
- makes electronic signatures possible.

– makes key handling simpler (but without guaranteeing anything without a *trust network*).

In general, Asymetric cryptography is based on functions which are easy to compute in one direction and difficult in another. For example, take two small prime numbers, let's say 23 and 37. Multiply them together (that'a quite easy). You get 851. Now try to find them just from the product. A public key will be the 851. The private key will be the (23, 37). Properties on numbers will be used which require to know the factors to be able to decrypt.

In practice In practice, most of the used protocols combine symetric and asymetric technology. A *session key* is generated using the asymetric technology, and then this session key, transmitted safely through asymetric technology, is used to cipher the message, with a performance and safety advantage (difficult to break the asymetric keys since the data exchange is minimal, and the session key are not reused).

Other concepts

- Diffie-Hellman
- hashing
- signature
- random number generator
- key length (sym/asym)
- certificates

Diffie-Hellman This scheme, using properties of the modulo arithmetic, allows to exchange a key without previously establishing a safe channel, but without authentification.

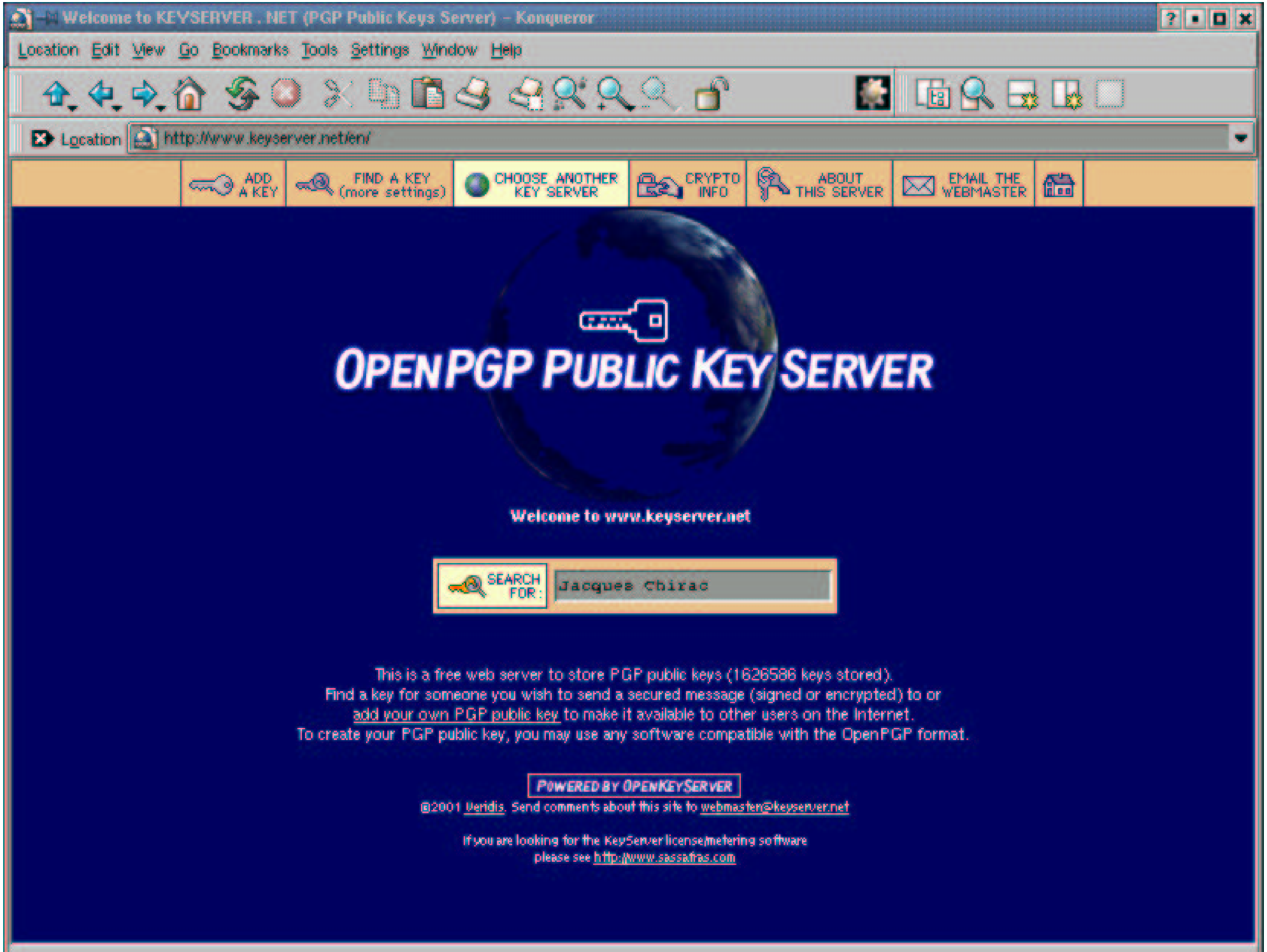
Hashing and signatures A hash is a projection of a message to a shorter value. The value has differing properties depending on the hashing function. For example, changes in bits which are in sequence in the message will be detected until a certain number (this is usually used in the *Cyclical Redundancy Checks* in telecommunication). Or, slightly dissimilar messages generate a very different value and it's difficult to maintain a similar value artificially on a changed message (chaotical response). In a way, hashing functions are *trapdoor* functions : they cannot be inverted. The only way is to go through the space of all possible solutions. Because this space is usually big and hashing functions, when used for cryptographic purposes are designed chaotic, this is time consuming – and that's the goal.
Examples :
– UNIX passwords
– `md5sum` UNIX command

Side effect : electronic signatures If a hash value is encrypted by the sender, using his *private key*, any recipient can decrypt it using the public key, and recompute the hash on the actual message. If the hashes are the same, the message was not modified. This works only, of course, if you know that the real sender's public key is the one you got. This question opens a can of worms : the *trust network* and certificates.

The problem of Trust

Problem : how do you know this is **my** public key ?

Solution : *trust network* or Certificate Authorities


Welcome to KEYSERVER . NET (PGP Public Keys Server) - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location
http://www.keyserver.net/en/
ADD A KEY
FIND A KEY (more settings)
CHOOSE ANOTHER KEY SERVER
CRYPTO INFO
ABOUT THIS SERVER
EMAIL THE WEBMASTER
OPENPGP PUBLIC KEY SERVER
SEARCH FOR :
POWERED BY OPENKEYSERVER

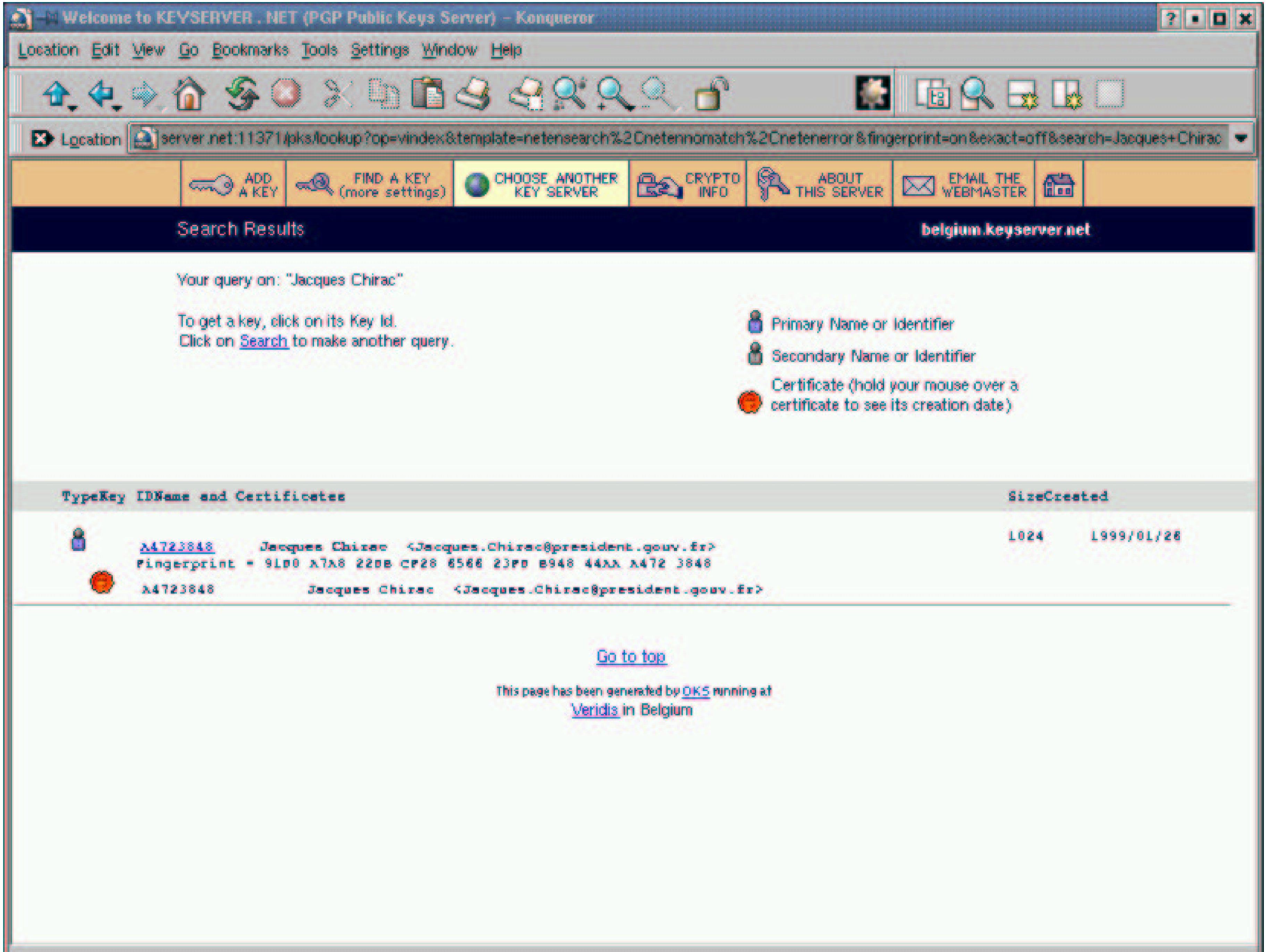
Welcome to www.keyserver.net

Jacques Chirac

This is a free web server to store PGP public keys (1626586 keys stored).
Find a key for someone you wish to send a secured message (signed or encrypted) to or
add your own PGP public key to make it available to other users on the Internet.
To create your PGP public key, you may use any software compatible with the OpenPGP format.

©2001 Veridis. Send comments about this site to webmaster@keyserver.net

If you are looking for the KeyServer license/metering software
please see http://www.sassafras.com


Welcome to KEYSERVER . NET (PGP Public Keys Server) – Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location
server.net:11371/pks/lookup?op=vindex&template=netensearch%2Cnetennomatch%2Cneteneror&fingerprint=on&exact=off&search=Jacques+Chirac
ADD A KEY
FIND A KEY (more settings)
CHOOSE ANOTHER KEY SERVER
CRYPTO INFO
ABOUT THIS SERVER
EMAIL THE WEBMASTER

Search Results belgium.keyserver.net

Your query on: "Jacques Chirac"

To get a key, click on its Key Id.
Click on Search to make another query.

Primary Name or Identifier

Secondary Name or Identifier

Certificate (hold your mouse over a certificate to see its creation date)

```
TypeKey IDName and Certificates                                                   SizeCreated

        A4723848      Jacques Chirac  <Jacques.Chirac@president.gouv.fr>          1024    1999/01/28
        Fingerprint = 91D0 A7A8 220B CF28 6566 23FD B948 44AA A472 3848
        A4723848            Jacques Chirac   <Jacques.Chirac@president.gouv.fr>
```

Go to top

This page has been generated by OKS running at
Veridis in Belgium

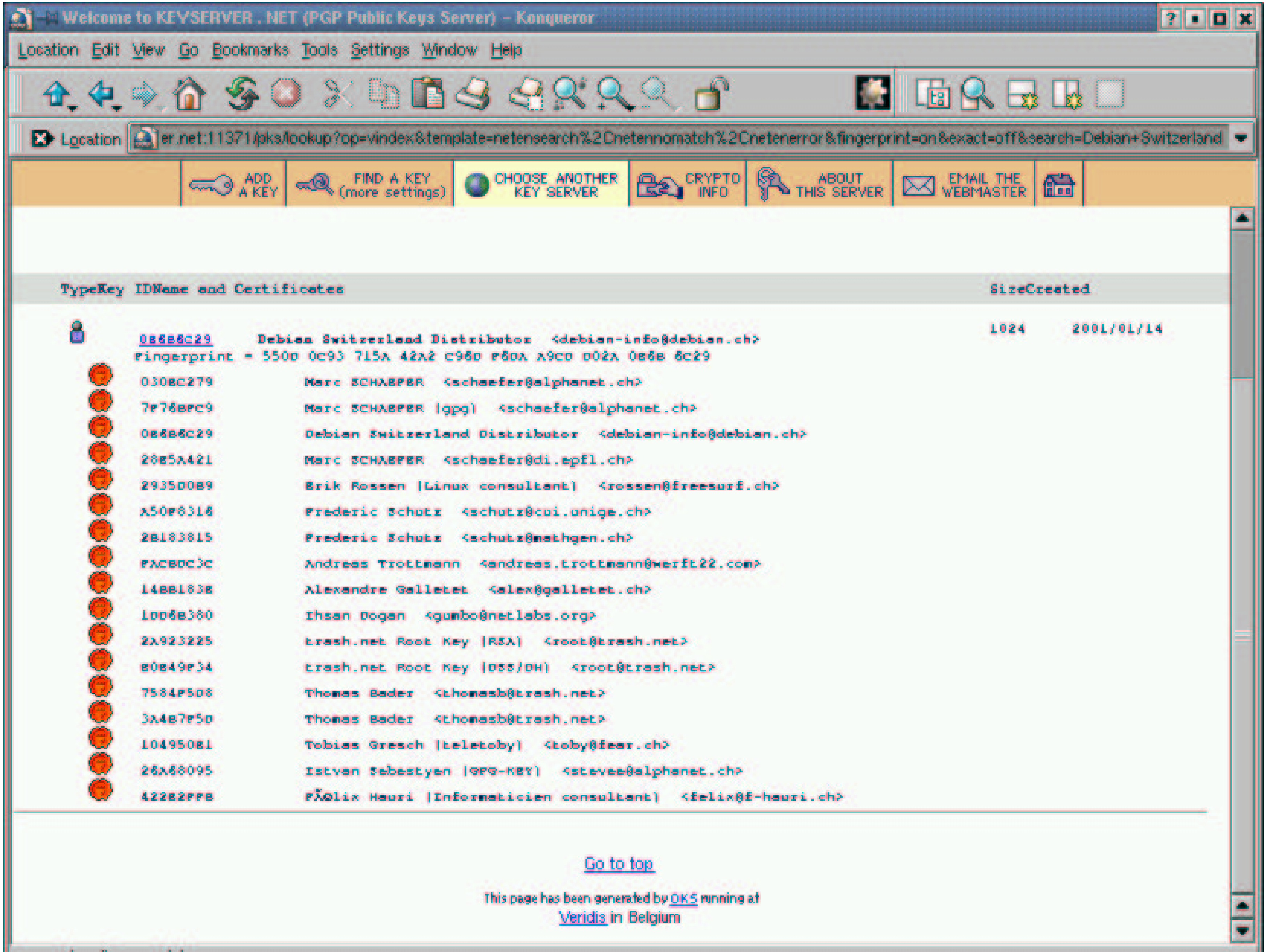
from, as long as each of the signee trusts the signed key.

For example, with GPG or PGP, you can deposit your key(s) to a *key server*, for example `http://www.keyserver.net` along with the signatures of the person who directly know you. When you meet people and give them your key, they can sign it and deposit the signature on the key server directly.

It then suffices for any of the signee to be known directly or indirectly by a recipient and he will be able to assert that your key really comes from you.

Giving a finger-print of the key (a hash, really) is in general sufficient : by phone, or in person. In some cases you may want to check the ID card of the person.

All this is called a trust ring.



Welcome to KEYSERVER . NET (PGP Public Keys Server) - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location
ADD A KEY
FIND A KEY (more settings)
CHOOSE ANOTHER KEY SERVER
CRYPTO INFO
ABOUT THIS SERVER
EMAIL THE WEBMASTER

Location: er.net:11371/pks/lookup?op=vindex&template=netensearch%2Cnetennomatch%2Cnetenerror&fingerprint=on&exact=off&search=Debian+Switzerland

```
TypeKey IDName and Certificates                                                    SizeCreated

        0E6B6C29     Debian Switzerland Distributor  <debian-info@debian.ch>        1024     2001/01/14
       Fingerprint = 550D 0C93 715A 42A2 C96D F6DA A9CD D02A 0E6B 6C29
        030BC279          Marc SCHAEFER  <schaefer@alphanet.ch>
        7F76BFC9          Marc SCHAEFER (gpg)  <schaefer@alphanet.ch>
        0E6B6C29          Debian Switzerland Distributor  <debian-info@debian.ch>
        2BE5A421          Marc SCHAEFER  <schaefer@di.epfl.ch>
        2935D0B9          Erik Rossen (Linux consultant)  <rossen@freesurf.ch>
        A50F8316          Frederic Schutz  <schutz@cui.unige.ch>
        2B183815          Frederic Schutz  <schutz@mathgen.ch>
        FACB0C3C          Andreas Trottmann  <andreas.trottmann@werft22.com>
        14BB183B          Alexandre Galletet  <alex@galletet.ch>
        1DD6B360          Ihsan Dogan  <gumbo@netlabs.org>
        2A923225          trash.net Root Key (RSA)  <root@trash.net>
        E0B49F34          trash.net Root Key (DSS/DH)  <root@trash.net>
        7584F5D8          Thomas Bader  <thomasb@trash.net>
        3A4B7F5D          Thomas Bader  <thomasb@trash.net>
        10495081          Tobias Gresch (teletoby)  <toby@fear.ch>
        26A68095          Istvan Sebestyen (GPG-KEY)  <stevee@alphanet.ch>
        42282FFB          FÃ©lix Hauri (Informaticien consultant)  <felix@f-hauri.ch>
```

Go to top

This page has been generated by OKS running at
Veridis in Belgium

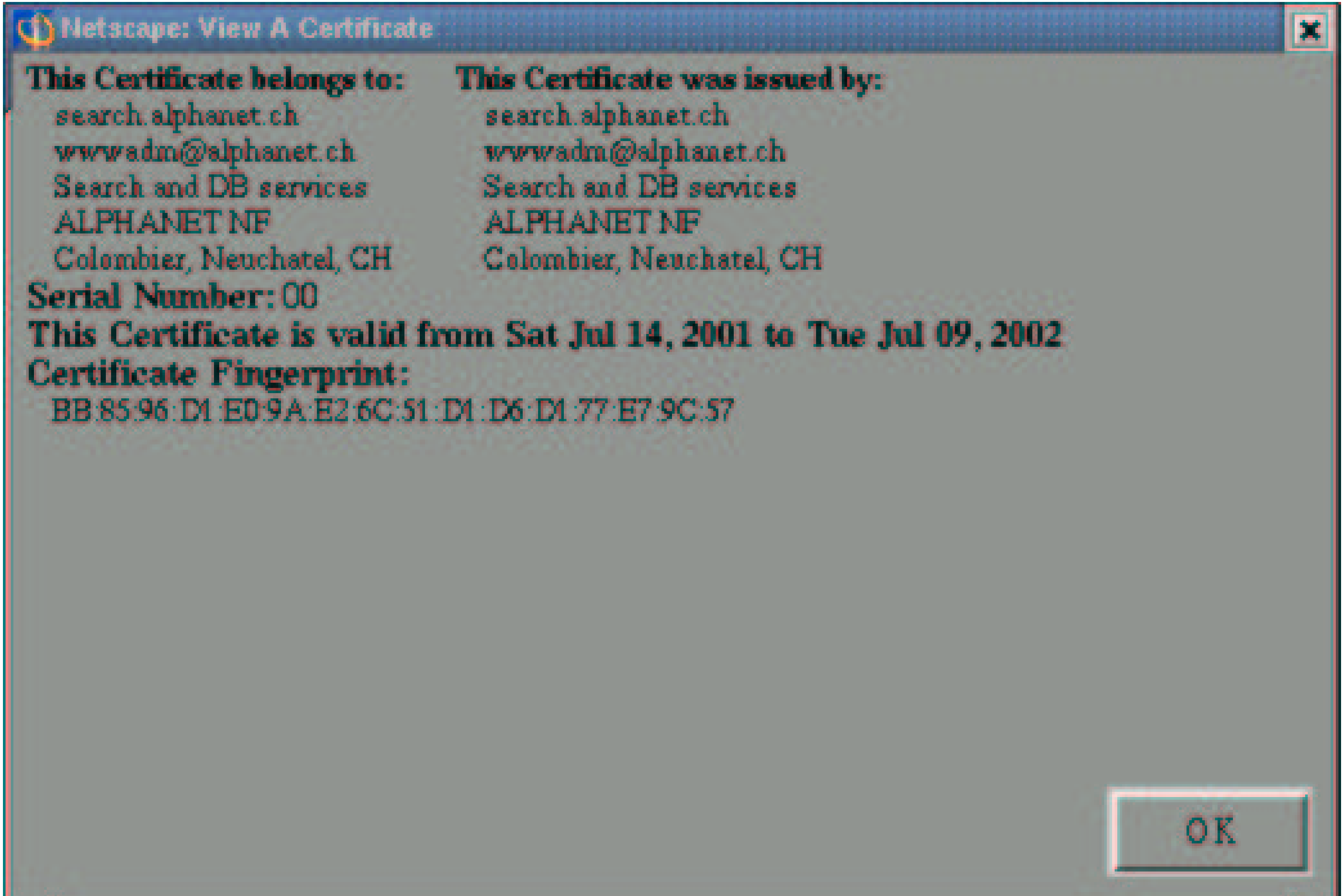
Certificate Authorities A certificate is a signed key. This signature means that the signee testifies that the key belongs to his owner. If you trust the signee, you trust the key !

All WWW browsers have a list of known-safe keys built-in. Unfortunately, providing a safe service is not a prerequisite from being listed : in general, you just need to give money to the WWW browser software editor.

This privilege doesn't mean that this authority doesn't deliver certificates to people who are not who they pretend.

For example, **Verisign** has delivered recently a certificate for the `microsoft.com` domain to a third-party (if you don't believe me, maybe you will believe Microsoft : `http://support.microsoft.com/default.aspx?scid=kb;EN-US;q293818`).

The real solution (until some reliable and trustworthy infrastructure exists, see above for Verisign) is to ask the finger-print of the key to the service provider directly, through a safer mean (for example telephone). Unfortunately, until recently at least, most of those didn’t even know what a finger-print was.



This Certificate belongs to:

search.alphanet.ch
wwwadm@alphanet.ch
Search and DB services
ALPHANET NF
Colombier, Neuchatel, CH

This Certificate was issued by:

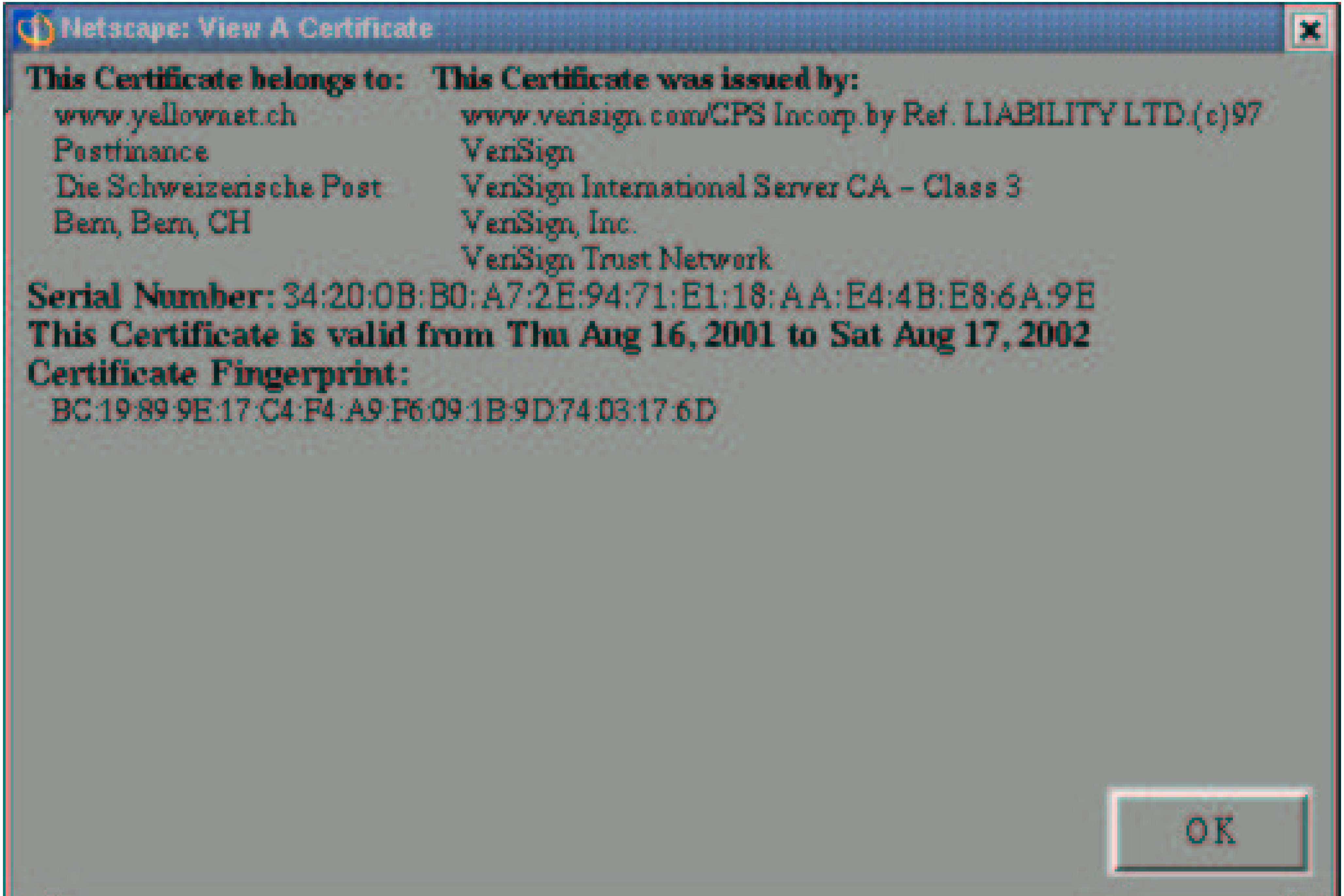
search.alphanet.ch
wwwadm@alphanet.ch
Search and DB services
ALPHANET NF
Colombier, Neuchatel, CH

Serial Number: 00

This Certificate is valid from Sat Jul 14, 2001 to Tue Jul 09, 2002

Certificate Fingerprint:

BB:85:96:D1:E0:9A:E2:6C:51:D1:D6:D1:77:E7:9C:57



This Certificate belongs to:

www.yellownet.ch
Postfinance
Die Schweizerische Post
Bern, Bern, CH

This Certificate was issued by:

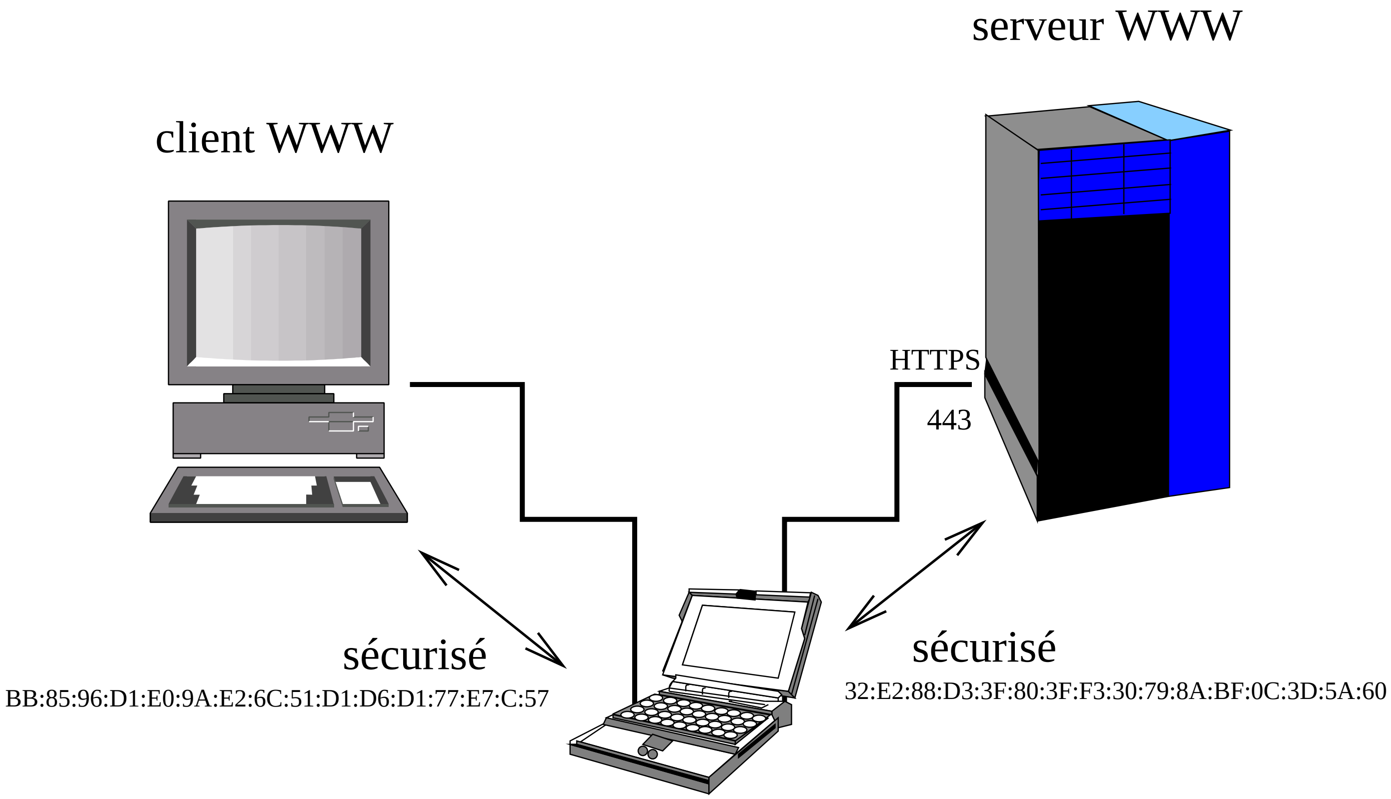
www.verisign.com/CPS Incorp.by Ref. LIABILITY LTD.(c)97
VeriSign
VeriSign International Server CA - Class 3
VeriSign, Inc.
VeriSign Trust Network

Serial Number: 34:20:0B:B0:A7:2E:94:71:E1:18:AA:E4:4B:E8:6A:9E

This Certificate is valid from Thu Aug 16, 2001 to Sat Aug 17, 2002

Certificate Fingerprint:
BC:19:89:9E:17:C4:F4:A9:F6:09:1B:9D:74:03:17:6D

Application : Man in the middle attack


serveur WWW
client WWW
HTTPS
443
sécurisé
BB:85:96:D1:E0:9A:E2:6C:51:D1:D6:D1:77:E7:C:57
sécurisé
32:E2:88:D3:3F:80:3F:F3:30:79:8A:BF:0C:3D:5A:60

man−in−the−middle attack

Man in the middle attack This attack allows a cracker to read and modify all the data he wants in a connection to a secure WWW server. This of course would also work for e-mail in some cases. The attacker will simply reside on the path between the two computers and replace the key exchange by his own key, creating two safely encrypted tunnels instead of one. **DNS spoofing** attacks may make those attacks quite easy, even without a physical access to the paths between the two machines.

Solutions :

- trust rings
- certification authorities.
- ask the key fingerprint directly to the service provider, through a safe channel.

In the case of WWW, note that in general, the WWW browsers will alarm if a server SLL key changes : note that because keys usually have an expiration date (for security reasons), keys WILL change from time to time, making everything fuzzier.

Other attacks

by **decreasing importance** :

- *social engineering*
- software failures and bugs
- *tempest* radiations (wireless !)
- *brute force* (computers getting better)
- différential (**DPA**) (on-chip)
- theory attacks

weak link

Weak link As always, your problem is the weakest link : if a server offers SSL but it is not mandatory, or it ships passwords with e-mail in cleartext, it won’t be safe.

The complexity of systems makes often human beings the weak link.

PKI

- ITU-T standard X.509
- specifies interoperable way to store and exchange certificates
- **OpenSSL** supports most operations

Exchange your keys !

help build a stronger/wider trust network

Thank you !

More information

- Cours postgrade sécurité ESNIG, October 2002-June 2003, Neuchâtel, `http://www.esnig.ch`
- GPG Home Page : `http://www.gnupg.org`
- PKI Home Page : `http://www.pki-page.org`
- Crypto Home Page ETHZ : `http://www.crypto.ethz.ch`
- Thomas BADER's GPG article (German) :

  `http://www.linux-magazin.de/ausgabe/1999/12/GnuPG/gnupg.html`

```
$Id: content.tex,v 1.10 2002/06/17 11:40:10 schaefer Exp $
```