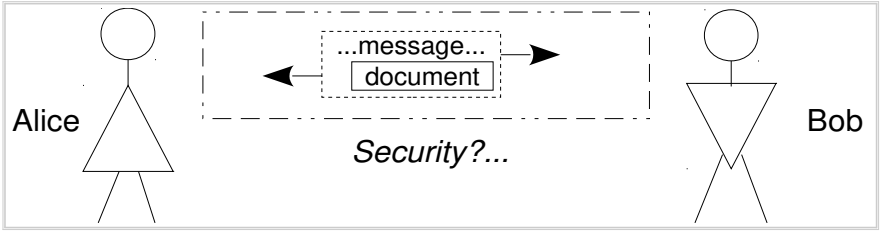
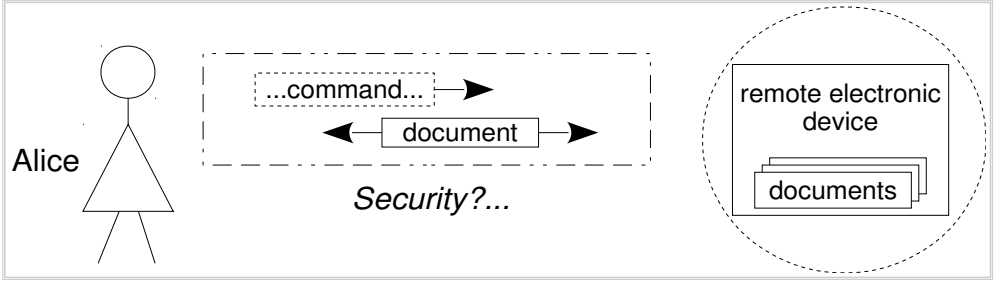
General Protection Techniques (cont.): two case studies (2)
- Technology case study one: (3)
  - OpenPGP – Open *Pretty Good Privacy* (3)
- Technology case study two: (8)
  - SSH – Secure Shell (8)

General Protection Techniques (cont.): two case studies

OpenPGP – Open *Pretty Good Privacy*





SSH – Secure SHell

Technology case study one:

OpenPGP – Open *Pretty Good Privacy*

History

- 1991: original author (PGP): Philip Zimmermann
  - private electronic mail for everyone!
- «*If privacy is outlawed, only outlaws will have privacy*!»
  - 1993-96: conflict with the government of the United States
- 2007: OpenPGP, IETF standards track (RFC 4880)

Features

- confidentiality, authentication and message integrity
  - does not protect headers! (`Subject:`, `To:`, `From:`,...)
- usually, two asymmetric keys
  - primary key: Signing, Certification, Authentication [SCA]
  - sub key: Enciphering [E]
- hybrid cryptography
  - symmetrical cipher, with session key ciphered asymmetrically (sub-key!)

```
$ gpg --list-keys
pub   dsa1024 2006-05-15 [SCA]
      C88068D16B39AB6FC14C45E5FC1E58BECFB027E7
uid        ultimate] J.M.Cruz <jmcruz@fe.up.pt>
sub   elg1024 2006-05-15 [E]
```

- validation of public keys: interesting decentralized technique (*web of trust*)
  - worldwide[1] key server: `keys.openpgp.org`[2]
- compaction of messages (automatic, usually)

1 FEUP used to have a local PGP key server, keysrv.fe.up.pt, but it has been put down for some misterious reason...
2 lookup "jmcruz@fe.up.pt"!

Public key management – the *web* (or *ring)* of trust [FIG]

- each user assigns a certain degree of trust to another user[1]
  - trust: `unknown`, `none`, `marginal`, `total`
- system calculates validity of a public key based on assigned trust of signers
  - validity: `unknown`, `doubtful`, `valid`

Key validity

- classically, public key is valid if signed by:
  - one user with total trust
  - two users with marginal trust
- with GnuPG, public key is valid if signed by:
  - a number of users with total trust (default, 1)
  - a number of users with marginal trust (default, 3)
    - but only if the signature path[2] is limited (default, less than 5)

1 in the sense that he/she finds that user to be a reliable key signer!
2 *X* signed K_Y, --> *Y* signed K_Z, --> *Z* signed...

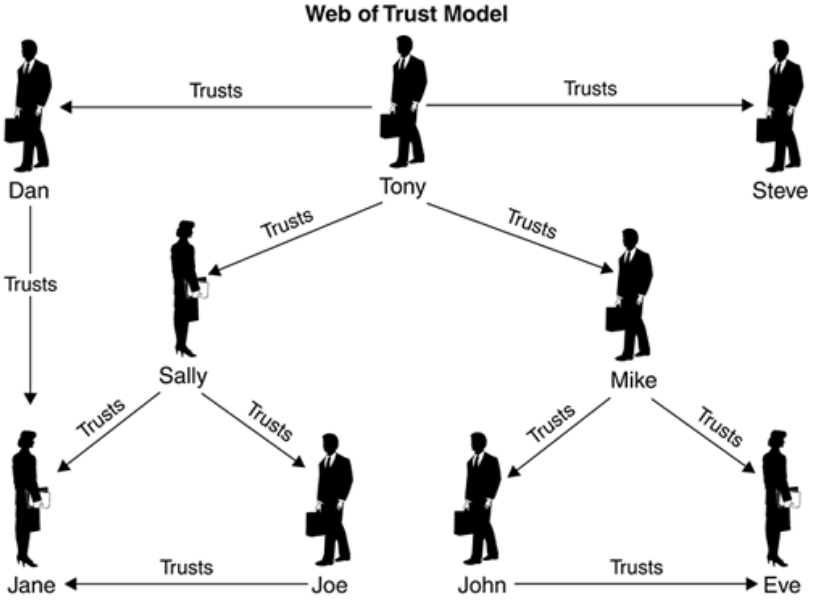


Fig. OpenPGP's web of trust. (in: litux.nl/mirror/securitytools/ddu/ch09lev1sec1.html)

Short comparison between OpenPGP, S/MIME and PEM[1]

	OpenPGP	*S/MIME*	*PEM*
certification of public keys	directly or through digital certificates	only through digital certificates	only through digital certificates
validation of certificates	up to the user	multiple parallel hierarchies of Certification Authorities	single hierarchy[2] of Certification Authorities
certification's procedure	hard, because relies only on the user (*web of trust*)	easy, based on PKIX's model, with X.509 certificates	easy, once the hierarchy is established
user trust level on system	up to the user	user might choose the hierarchy	complete (single hierarchy)
security's potential	great	great	low
character encoding scheme	Radix-64[3] ~ Base 64 + CRC	~ Base 64	Base 64 (RFC 1421)

1 Privacy-Enhanced Mail
2 top entity: IPRA - Internet Policy Registration Authority
3 also known as *ASCII Armor*

Technology case study two:

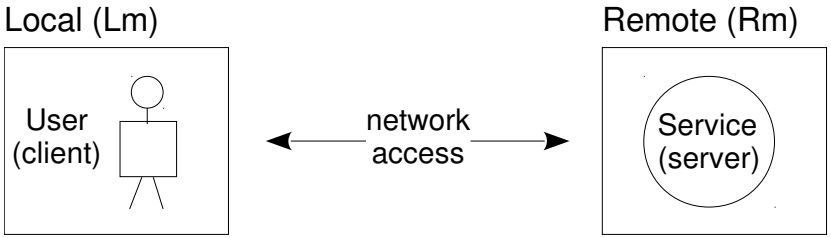
SSH – Secure Shell

Services

- authentication, confidentiality and integrity of sessions of
  - remote terminal
  - file transfer
  - port rerouting

History

- 1995: Tatu Ylönen, TKK - Helsinki University of Technology
- 1996: v.2, modularization, protocol negotiation, channel multiplexing, DH...
- 2006: proposed IETF standard, RFC 4250-4
- OpenSSH, free version! (`www.openssh.org`)



SSH: operation phases

- 1st: basic security services are setup (Transport Protocol)
  - server authentication, keys negotiation, ciphering, ...
- 2nd: client authenticates to server (Authentication Protocol)
  - public key, password, ...
- 3rd: user services are setup and operate (Connection Protocol)
  - remote login, file transfer, ...

SSH's phase 1: transport protocol

- basic security services:
  - server authentication (beware of 1st connection!) [Fig]
  - confidentiality (negotiable algorithm)
  - data integrity (negotiable algorithm)
  - session identification (useful to upper layers)
  - perfect forward secrecy (“random” temporary session keys!)
  - compression (optional)

...SSH: transport protocol (cont.)

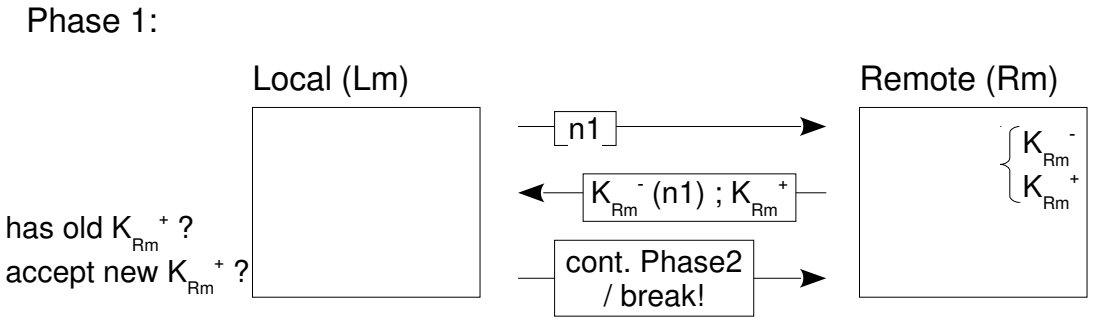


Fig. SSH authentication protocol for server (in remote machine).

Important problem

- does Client know that Server is the real one?
  - Yes, if he has access to genuine K_S^+ !
  - But, does he normally has?...

```
$ sftp gnomo.fe.up.pt

The authenticity of host 'gnomo.fe.up.pt (10.227.240.69)'
can't be established.
ED25519 key fingerprint is
SHA256:/OZ1yHqYHuZyKRLZ1WS9UpTN5Q8qS3zOOM/QUzvX3Aw.

Are you sure you want to continue connecting
(yes/no/[fingerprint])?
```

SSH's phase 2: (client) authentication protocol

- of client by server:
  - via password (most used!) [Fig-phase2]
  - via public-key (preferred!) [Fig-phase2(alt)]
  - via machine (dangerous!)
  - other...

Phase 2:

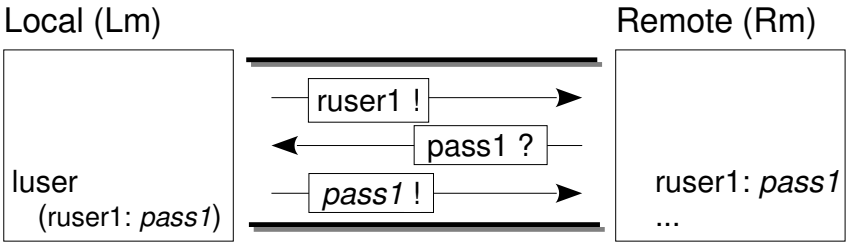


Fig. Authentication protocol for client – via password.

Phase 2 (alt):

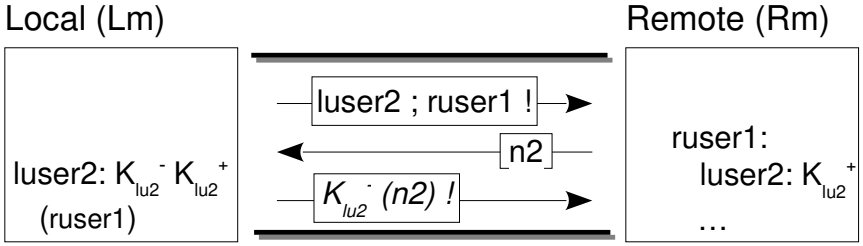


Fig. Authentication protocol for client – via public-key.

SSH's phase 3: connection protocol

- user level services:
  - point-to-point security
    - remote terminal
    - file transfer
  - tunneling
    - port forwarding

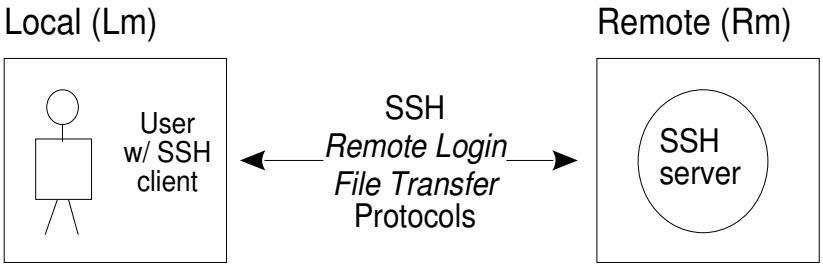


Fig. SSH at work!