Topic 9: Entity Authentication

User Identification and Authentication

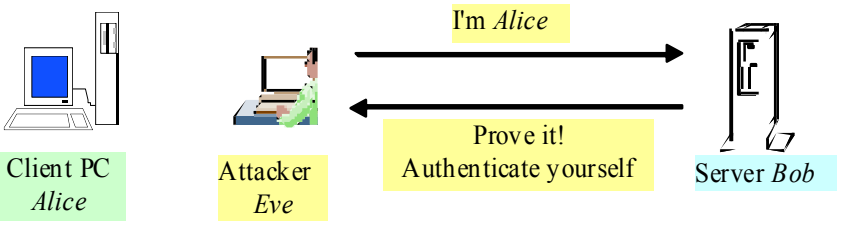
Apply authentication techniques to counter impersonation or masquerading

Overview

- Authentication overview
- Client-server authentication (I teach!)
  - Password-based authentication (Unix authN)
  - Password-based authentication (OTP)
  - Smart-card-based authentication
  - X.509 certificate-based authentication
- Enterprise-wide authentication (You read!)
- Conclusion

source: some stuff in chapter 15 of Cryptography and Network Security by W. Stallings;

Authentication Overview - Why do we need it



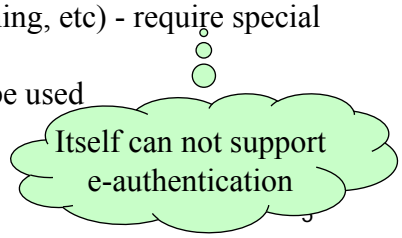
- If the communication takes place over a network, how could *Bob* be assured that the person claiming to be *Alice* really is *Alice*? If *Bob* is a server, the impostor may be able to log in as *Alice* to access data and services, or to use her account to launch further attacks.

Authentication Overview - What it is for

- **Authentication**
  - User identification/authentication or entity authentication
    - The process of verifying a claimed identity
      - Who the user is?
      - Which system? – you could talk to anybody (mutual identification and authentication)
    - The user identity is a parameter in access control decisions - authorisation
    - The user identity is recorded when logging security-relevant events in an audit trail – accounting
    - This is the so called AAA services
  - Communication/message authentication - *we did this already*!
    - The message is from the source it claims to be.
    - The message has not been altered or replayed.

Authentication Overview - Methods

- Methods for user identification/authentication:
  - Where you are (location authentication - physical location/specific terminal, e.g. based on IP addresses).
  - Something you know (passwords, PIN).
  - Something you have (keys – soft tokens, and hard tokens (smart cards)) - may require special hardware.
  - Something you are (biometrics - fingerprint matching, voice recognition, face recognition, iris scanning, etc) - require special hardware.
  - Combined (or multiple) methods may be used for a higher level of assurance.

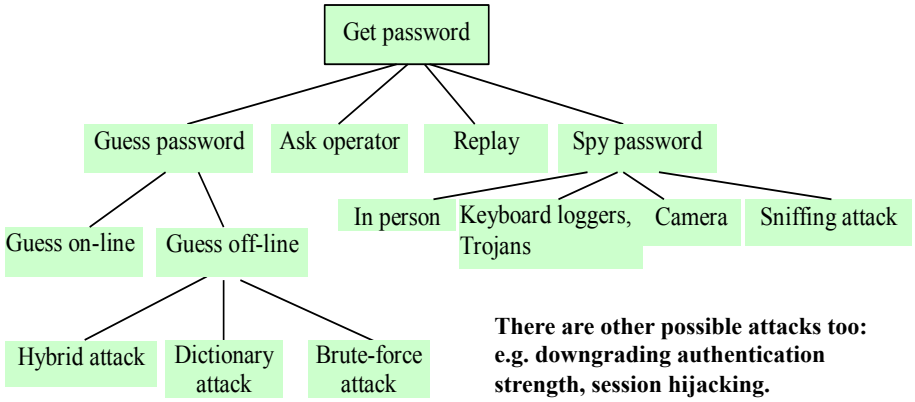


Authentication Overview - Prominent schemes

- Client-server authentication solutions
  - Password-based authentication.
  - Smart-card-based (token-based) authentication.
    - Symmetric key based
    - PKI based - Digital signatures and X.509 certificates.
- Enterprise-wide authentication solutions (touches the issue of single-sign-on)
  - Kerberos (a password centric solution).
  - RADIUS (a centralised AAA service).
- Shibboleth (authenticating access to multiple enterprises/organisations) – outside the scope of this module (not enough time).
- Different authentication schemes provide different levels of assurance.
- There is a trade-off between the level of security vs complexity vs cost.

Password-based authentication - Problems

- Threat tree for obtaining another user's password



There are other possible attacks too: e.g. downgrading authentication strength, session hijacking.

Password-based authentication - Unix Solution (1)

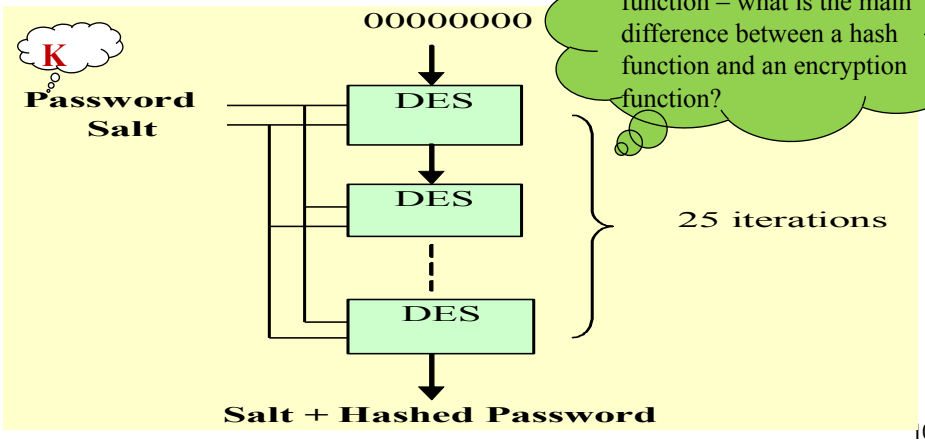
- Unix system chooses not to store plaintext passwords, rather it stores encrypted/hashed passwords in the password file.
  - Storing passwords for all the system users plainly visible in a password file is vulnerable to theft and accidental disclosure (e.g. due to programming errors).
- The hashing algorithm is an one-way function, called *crypt( )* that is modified based upon the DES algorithm.
- It uses salt to make the DES-based one-way function different from DES and to make dictionary attacks harder to succeed.

Password-based authentication - Unix Solution (2)

- UNIX Crypt() algorithm
  - Using DES with first 7 bits of the first 8 characters of password as the key.
  - Iterated 25 times on constant string 0s; making the process slower.
  - Using salt to perturb the DES algorithm, so that
    - DES chip can not be used to (dictionary) attack the algorithm.
    - It makes precompiled dictionary attacks harder (by a factor of 4,096).
    - It prevents an identical password from producing the same encrypted password.
  - The final 64 bits are unpacked into a string of 11 printable characters, called *the encrypted password*.

Password-based authentication - Unix Solution (3)

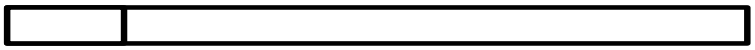
The Crypt() function



This is effectively a hash function – what is the main difference between a hash function and an encryption function?

Password-based authentication - Unix Solution (4)

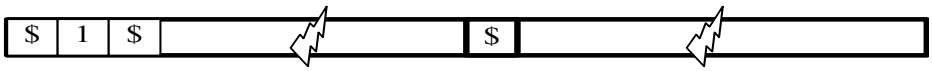
- Most of today's implementations, e.g. Fedora, support MD5 in addition to legacy DES-based crypt() function.



Salt
2 chars

Encrypted password
11 chars

(a) UNIX legacy crypt() password format



Indicate
MD5 hash

MD5 salt used to
initialise MD5
8 chars

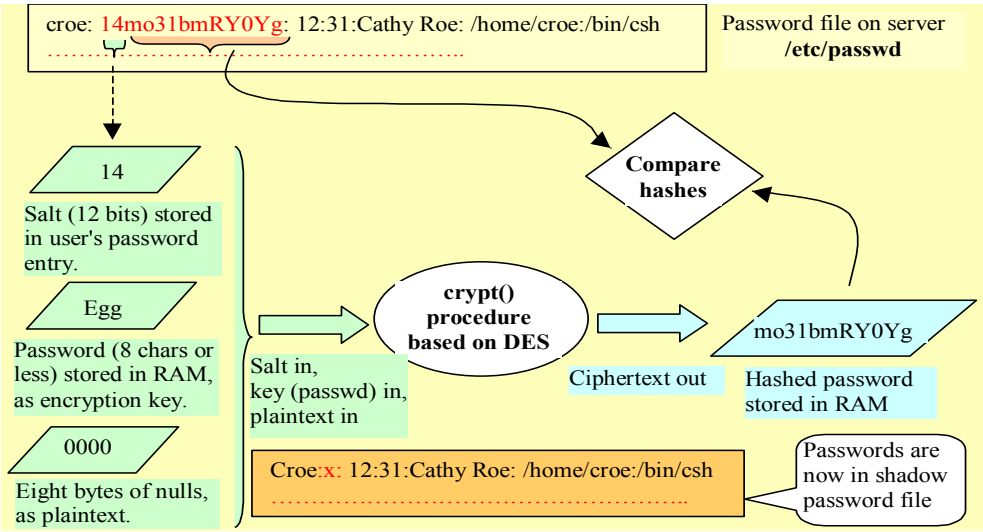
MD5 password hash
22 characters

(b) UNIX MD5 encrypted password format

Password-based authentication - Unix Solution (5)

- When a user have an account created or changes the password, the */bin/passwd* program
  - selects a salt based on the time of day; the DES salt is a 12-bit number, between 0 and 4095, that is converted into a two-character string and is stored in the */etc/passwd* file along with the encrypted password.
  - the password is used as the encryption key to encrypt a block of zero bits using *crypt( )* to generate the encrypted password.
- When a user tries to log in, the program */bin/login* takes the password the user typed, and the salt from the password file, to generate a fresh encrypted password, and compares the newly generated one with the one stored in the */etc/passwd* file. If the two encrypted results match, the system lets you in.

Password-based authentication - Unix Solution (6)



Password-based authentication - Unix Solution (7)

- The Unix password file
  - is */etc/passwd*. Each entry in the file is for one account and has several fields separated by colons:
    - User Name (croe): the account name.
    - Password (14mo31bmRY0Yg): the hashed password (mo31bmRY0Yg) preceded by a salt value (14) to be used with the password. Salts make the password
      - more difficulty to guess, and
      - the hashing algorithm slower!
    - User ID (UID=12): a number assigned to this user name for system use in identifying the account.
    - Group ID (GID = 31): a number for the user's group.
    - Home Directory (/home/croe).
    - Shell (/bin/csh): the user's default shell program.

Password-based authentication - Unix Solution (8)

- More problems
  - **Problem 1**: /etc/passwd file need to be accessed by processes, so solution-1 in the diagram would allow anyone to copy the file and to crack the passwords at his/her leisure!
  - **Countermeasure:** a shadow password file, /etc/shadow, is used, which stores the real passwords and is put in an area accessible only to the root account; put an x (or other placeholder) in the original /etc/passwd file. That is,

/etc/passwd file contains**:**
An example: User1:x:9111:9201:user1:/home/user1:/bin/bash
Meaning: UserName:x *(indicate that the password is stored in the /etc/shadow file)* :UserID:GroupID:FullName:HomeDirectory:UserShell.
/etc/shadow file contains**:**
An example: User1:1/uTQhcV4$2E…../:13030:0:99999:7:::
Meaning:
UserName:*hashedPassword*:passwdLastChanged:PasswdMayBeChanged:PaswdMustBeChanged:PasswdChangeWarning:DisableAccount:DisabledSince:Reserved.

Password-based authentication - Unix Solution (9)

- But attacks still possible - if you run some software processes with root privileges …, and if the attacker can take over such a program ...

- Problem 2: An attacker can eavesdrop on a network to get your login ID and encrypted password and later replays (re-send) it to gain access to the network - the replay attack.
  - In order to perform this attack, the *attacker needs* to
    - modify the client/logon software so that it does not encrypt the encrypted - password, but rather replay it directly;
    - eavesdrop on the network (or access to the password file).
  - *Usually we assume* that
    - the LAN is secure, i.e. eavesdropping can be noticed!
    - You do not bring your own client software in!
  - So we tend to overlook problem 2 in LAN environment.

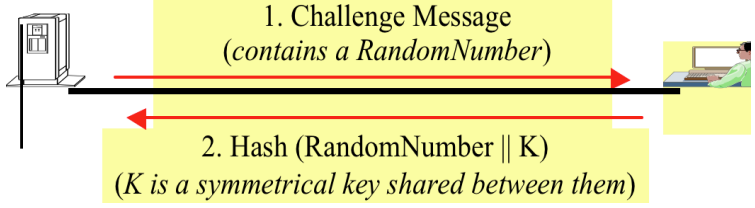
Password-based authentication - OTP

- One-Time Passwords (OTP)
  - Passwords that can be used only once
  - Thwart sniffing and replay attacks
- The approaches
  - Challenge-response
    - RandomNumber-based OTP
    - Clock-based OTP (need token)
      - the clock has to be reliable, and secure;
      - the clocks between the entities must be synchronised.
    - Counter-based OTP (need token).
    - If a hard token is used, then the token should be locked with a PIN or password
  - S/Key

Password-based authentication - Remote logon

- Three authentication protocols commonly used between a client and an infrastructure edge device, e.g. remote access server, a VPN server, a wireless access point:

CHAP (Challenge-Handshake Authentication Protocol)



MS-CHAP: identical to CHAP except for using password to replace the key K.

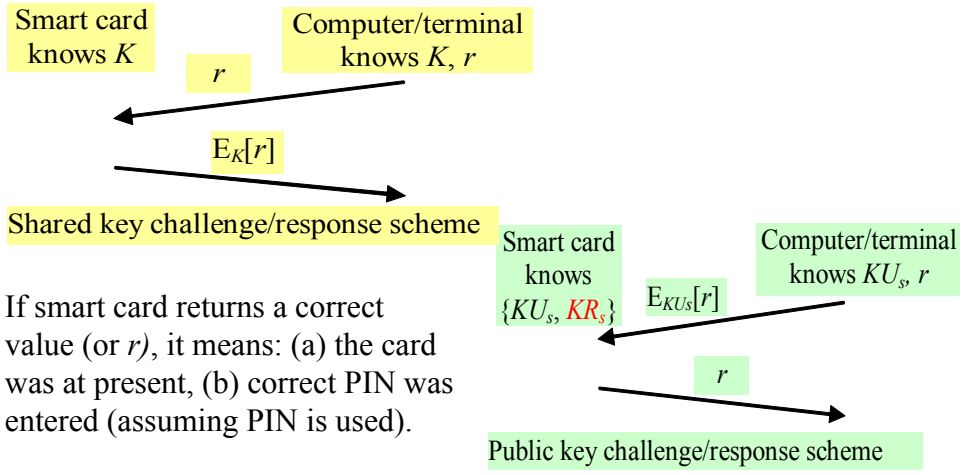
Password-based authentication - PIN protected OTP

- *PIN* protected *Token* authentication scheme
  - Using a PIN to unlock a token, i.e. when a PIN is entered, the token compares the PIN typed in against an internal copy.
    - If positive, the token will compute the OTP using the reading from the clock or counter embedded in the token + the base secret.
    - If repeated PIN entries are wrong, the token takes steps to resist a PIN guessing attack.

Smart-card-based authentication (1)

- A smart-card is an authentication token that a person carries around and uses in authenticating.
- Advantages
  - Unlike memory cards, they can do more than just containing some secret information; they can perform simple crypto operations.
  - Support mobility, can 'memorise' your secret, and can provide two factor authentication.
- Disadvantages
  - Smart-cards require a special hardware reader on every access device, which may be expensive and requires standardisation.
  - They are subject to theft, so used in conjunction with some other authentication mechanisms such as PIN/password.

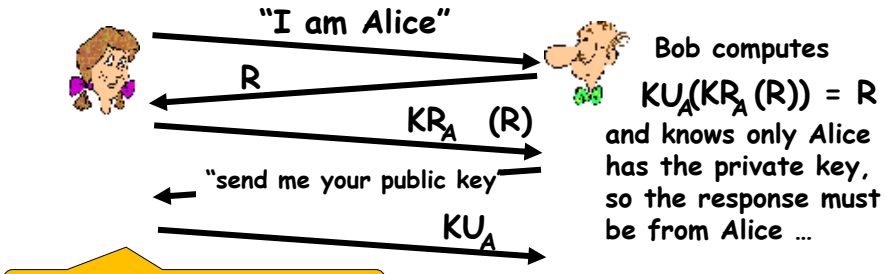
Smart-card-based authentication (2)



If smart card returns a correct value (or *r)*, it means: (a) the card was at present, (b) correct PIN was entered (assuming PIN is used).

The challenge-response using PKC

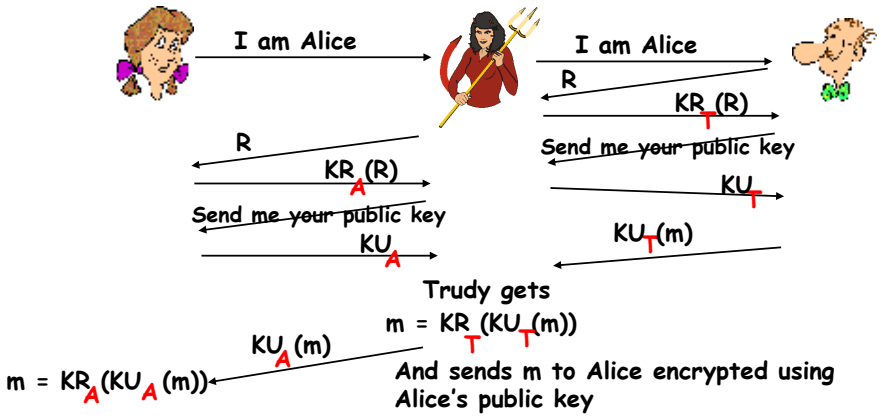
- The previous protocol requires shared symmetric key; what if you have not already established that 'trust relationship'?
- We authenticate using public key technique - Use nonce, public key cryptography:



Any security hole here?

The Security Hole

Man-in-the-middle-attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)

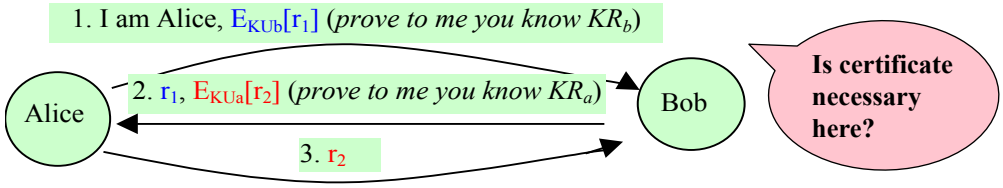


X.509 Certificate-based Authentication Service (1)

- **X.509**
  - defines a framework - a system to enable the validation of, and to give legal meaning to, digital signatures (which require the use of hash functions).
  - allows mutual authentication using public-key technology - digital signatures and digital certificates.
  - does not dictate the use of a specific public-key cryptographic algorithm but recommends RSA, nor does it define a specific hash algorithm.
  - used in S/MIME, IP Security, SSL/TLS and SET.

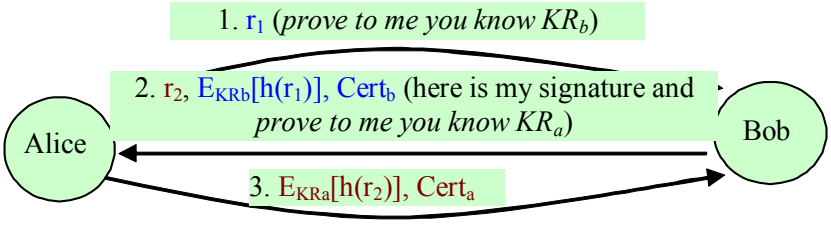
X.509 (2) - Authentication using public keys

- As only *B* can decrypt the random number (nonce), r_1, correctly, message (2) authenticate *B* to *A*.
- Message (3) authenticates *A* to *B*.
- 3 messages are needed for mutual authentication between two parties.
- This is called *challenge-response authentication method*.



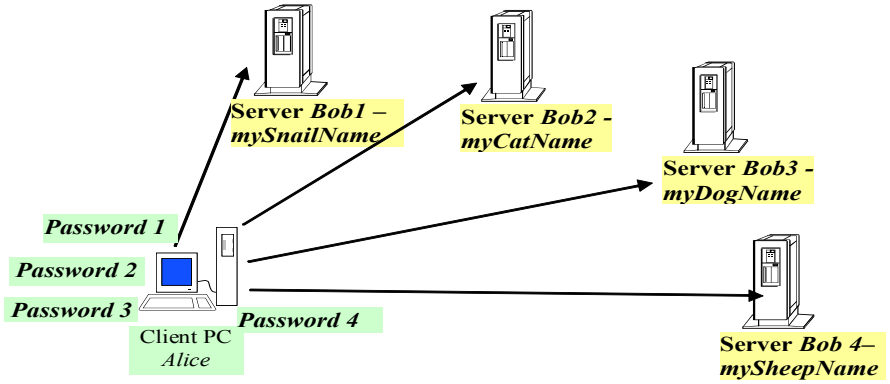
X.509 (3) - Authentication using digital signatures

- As only *B* can generate the rightful signature, message (2) authenticates *B* to *A*.
- Message (3) authenticates *A* to *B*.
- This is also called *challenge-response authentication method*.
- Can you tell me why r_1 is sent by A, instead of from B? Why B's public key certificate, $Cert_b$, is necessary here?



Enterprise authentication (1)

We have seen the single system story, as shown earlier. Now we have more systems, and more passwords!

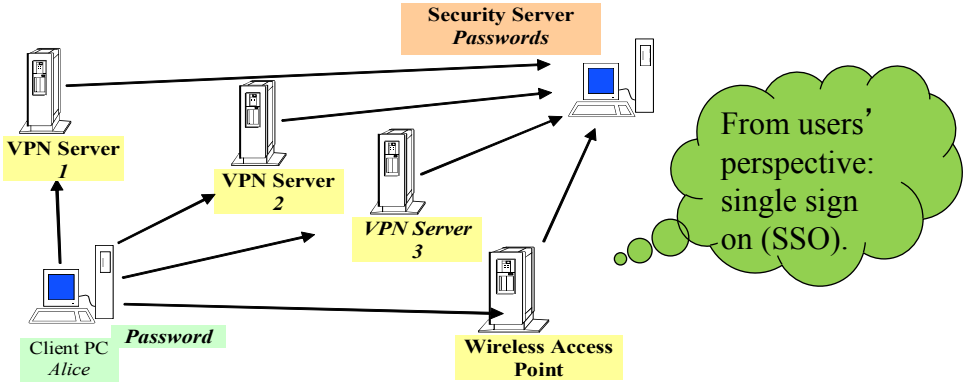


Enterprise authentication (2)

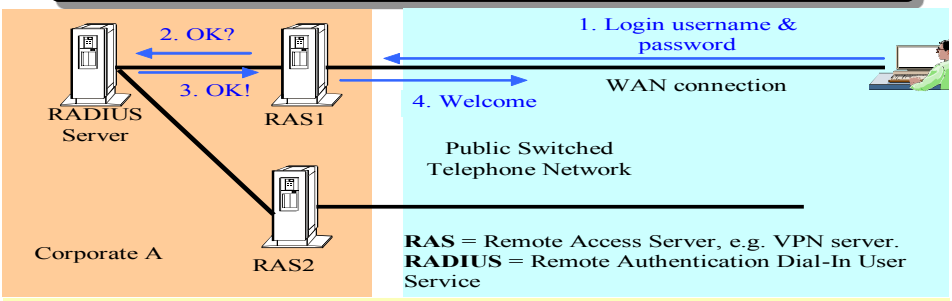
- Central authentication for a number of systems in an organisation
  - Let **one central authority** (usually called **security server**) at an organisation/site to manage your passwords instead of each computer having its own.
    - By this, we can enforce organisation wide security policies, including authentication, authorisation and accounting (i.e. the AAA services)
- A number of systems exist, e.g.
  - Radius - Remote authentication for dial-in user service
    - Initially used for providing authentication services for one or more access servers
    - Later extended to handle enterprise AAA services…..
  - Kerberos

Enterprise authentication (3)

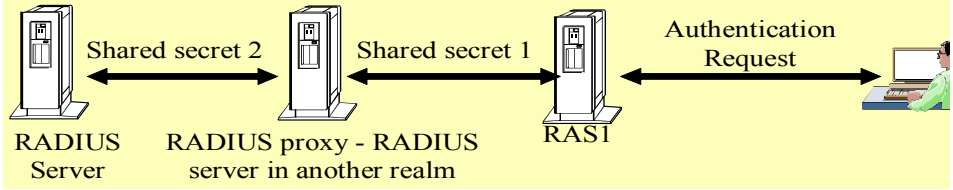
- The security server is responsible for providing AAA services to edge devices, e.g. VPN servers and wireless access points.



Enterprise authentication (4) – RADIUS

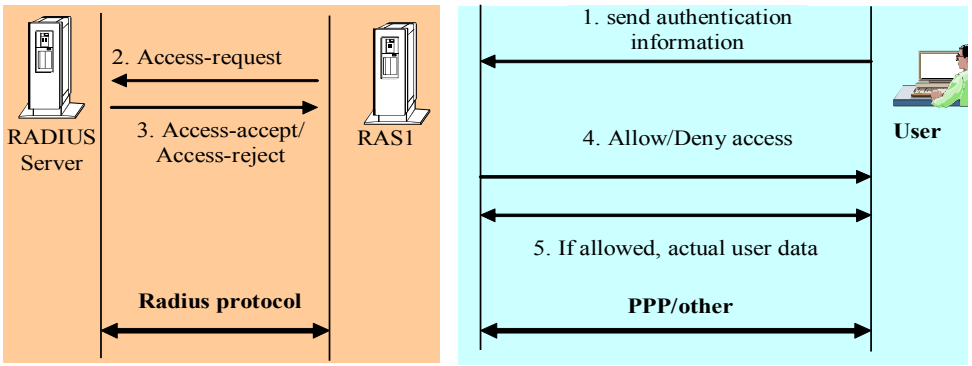


The model of trust: RADIUS uses shared secrets to achieve mutual authentication and confidential communication between RADIUS entities.



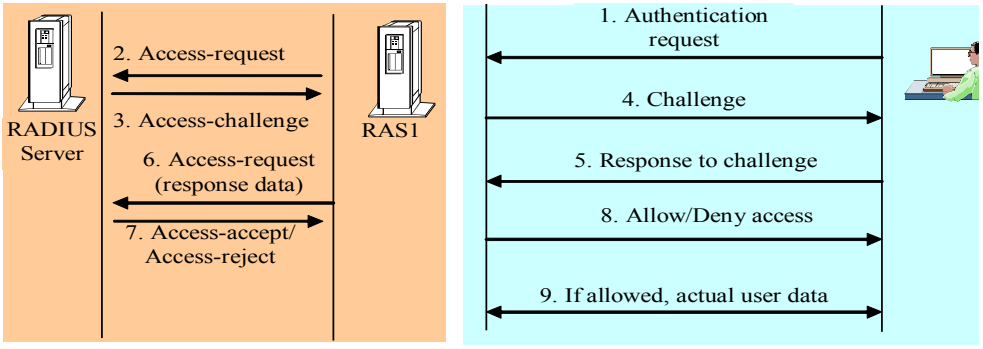
Enterprise authentication (5) - RADIUS

- Simple user authentication and authorisation using RADIUS



Enterprise authentication (6) - RADIUS

- Challenge-response authentication and authorisation using RADIUS



Enterprise authentication (7) - RADIUS

- RADIUS protocol:
  - Client forwards the user access request to a RADIUS server
  - Server
    - Replies with *reject access* or *allow access* based upon a user supplied password/credential.
    - Challenge (when challenge-response protocol is used, e.g. CHAP).
    - If challenge-response is used, client forwards Challenge to the user, and the user sends their Response to the client that then forwards it to the server.
    - One RADIUS server may act as a client to another RADIUS server for consultation, etc.

Exercise 9 (a)

- (a) You are given a hash function, and are asked to design a password based OTP (One Time Password) solution. You are not supposed to use random numbers, counter or timing information in this design.
- (b) Comment on any strength or limitation of the solution in (a).

Exercise 9 (b)

- Read this article and let us know what your thoughts are: **NIST Special Publication 800-63-2:** Electronic Authentication Guideline; available here at: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-2.pdf

Conclusions (1)

- Passwords are the most basic authentication mechanism
  - The security level is only as good as the passwords you select.
  - They are vulnerable to guessing unless precautions ensure there is a large enough set of possible passwords and each potential one in the set is equally likely to be chosen.
  - Challenge-response techniques allow the system to vary the password therefore less vulnerable to guessing attacks; OTPs, an example of this technique, are particular effective against guessing and replaying attacks.
- Authentication can also be achieved with public-key cryptography for which public key certificates are needed - X.509 standard.

Conclusions (2)

- There are also other forms of authentication: biometrics measures physical characteristics of the user; location requires the verifier to determine the location of the user.
- In practice, some combination of these methods can be used. This depends on the resources available to the verifier and the user, the strength of the authentication required, and external factors such as lows and customs.
- System designers have to balance convenience and security. Ease-of-use is an important factor in IT systems. However, convenient practices may introduce new vulnerabilities.
- There are authentication issues when multiple systems or multiple sites are involved.