TEKNILLINEN KORKEAKOULU

Access Control

Authentication and Authorization

- Authentication, Authorization (and Accounting), AAA
  - AAA services are especially important for commercial services
- User **authentication** means verifying the identity of a user
- Based on the authentication the user may be **authorized** to perform some actions
  - Access files, run programs
  - Access cellular network, make phone calls
- The usage may be logged, creating **accounting** information
  - Which can also be used for billing
- Suomeksi: todentaminen, valtuuttaminen/oikeuttaminen

Case telnet

- The user connects to a Unix server

```
10$ telnet server.foo.fi
login: kiravuo
Password:
Last login: Fri Aug 7 19:03:50 1998 from jalopeno
1$ _
```

- The "telnet" program opens a TCP connection to the server
- The user replies to the login query with his username
  - This provides the **identification** of the user
- The user is prompted for the secret password shared by the user and the server
  - The password provides the **verification** for the user’s identity
- The user is now **authenticated**
- Based on the authentication the starting of a shell for the user may be authorized
- Accounting information is created and saved

Basis for User Authentication

- Something the user **knows** (e.g. a password)
- Something the user has in his **possession** (e.g. a smart card)
- Something the user **embodies** or is (biometrics, e.g. user fingerprint)
- Sometimes also other information
  - User’s location
- Usually at least two of these methods are required for the user authentication to be considered strong

Passwords

- Currently user names and passwords are most common authentication methods
  - Smart cards with cryptographic keys or biometric systems may change this in the future
- Passwords should not be easily guessable
  - Brute force attacks on Unix's /etc/passwd are common
  - LAN file system passwords are another common target
- Passwords are hard to remember
  - Passwords can be stored in encrypted format in a non-networked device and be protected by a single password
- Experimental options to passwords exist
  - Select certain faces from a set of pictures
  - Click correct places in a picture
  - User's choices are predictable, so computer generated selections fare better in tests

TEKNILLINEN KORKEAKOULU

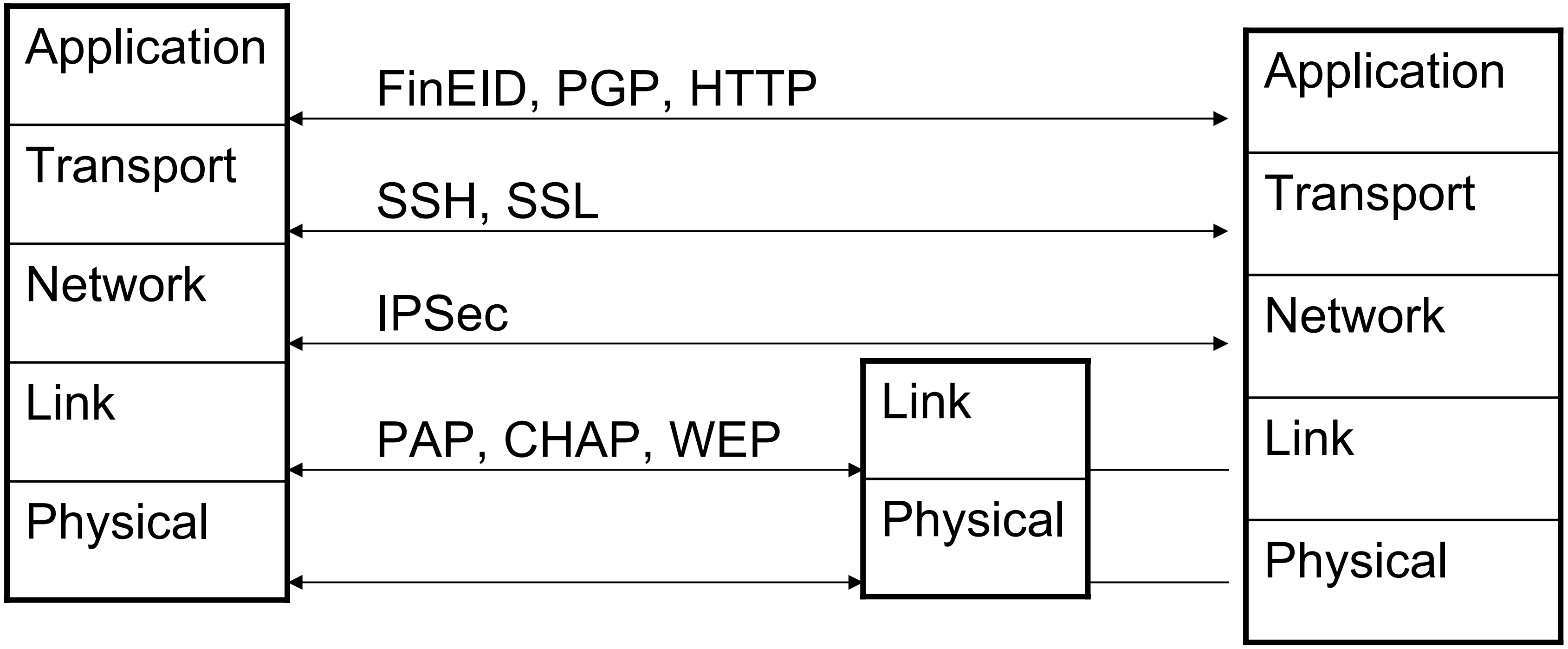
Alternatives to passwords

- One-time password systems
  - Each password is used only once
  - E.g. S/Key, banking passwords
- Password token
  - A physical device that can uniquely authenticate itself
  - Smartcards keep the passwords or crypt keys on the card
  - Securid is a popular product that has a clock and is synchronized with the authentication server
- Fingerprint readers
  - Require control of the device
  - Handy on laptops
- Methods are often used together with a password or PIN-code

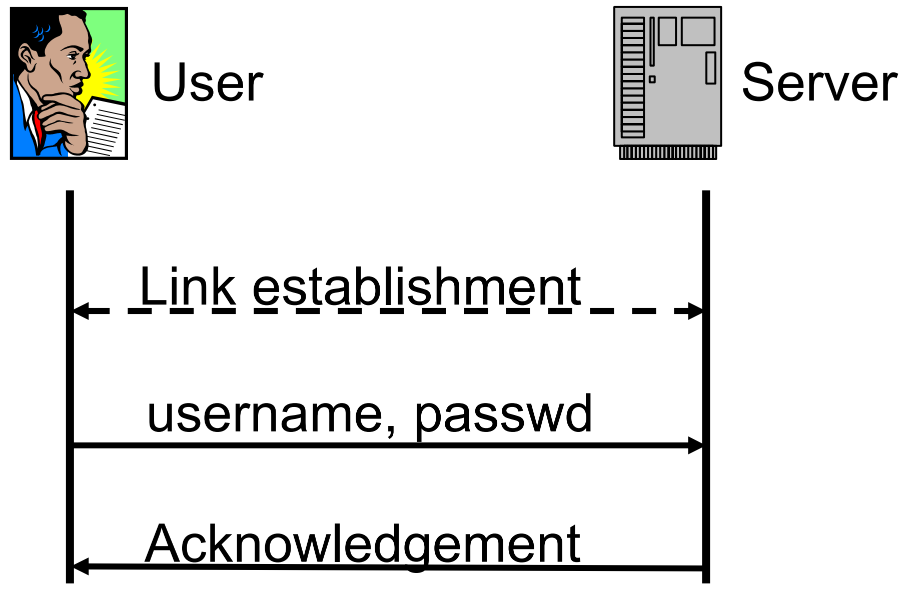
TEKNILLINEN KORKEAKOULU

Authentication in the Internet

- Authentication may be performed at various layers of the Internet protocol stack



Basic clear text authentication



TEKNILLINEN KORKEAKOULU

Attacking the Telnet

- The attacker may gain the user name and password by different means
  - Overseeing the user at the keyboard
  - Eavesdropping to the connection
  - Asking the user
- Having the username and password enables the attacker to present the user’s credentials to the server and thus to masquerade as the user
- The attacker may also capture the TCP connection after authentication
  - Continue the TCP connection towards the server, distract the client
- Or the attacker may pretend to be the server and ask for the user’s name and password

Authentication using usernames and passwords

- Cleartext passwords are open for many attacks
- The cleartext channel may be encrypted
  - Encryption prevents eavesdropping
  - MAC codes (message hashes) prevent connection capture
  - SSH, IPSec, SSL...
- The problem of encrypted channel is that the server side may be an attacker
  - The protocols above use public key (SSH) or PKI (IPSec, SSL) based server side authentication
  - Otherwise ”man in the middle” attack is a real threat

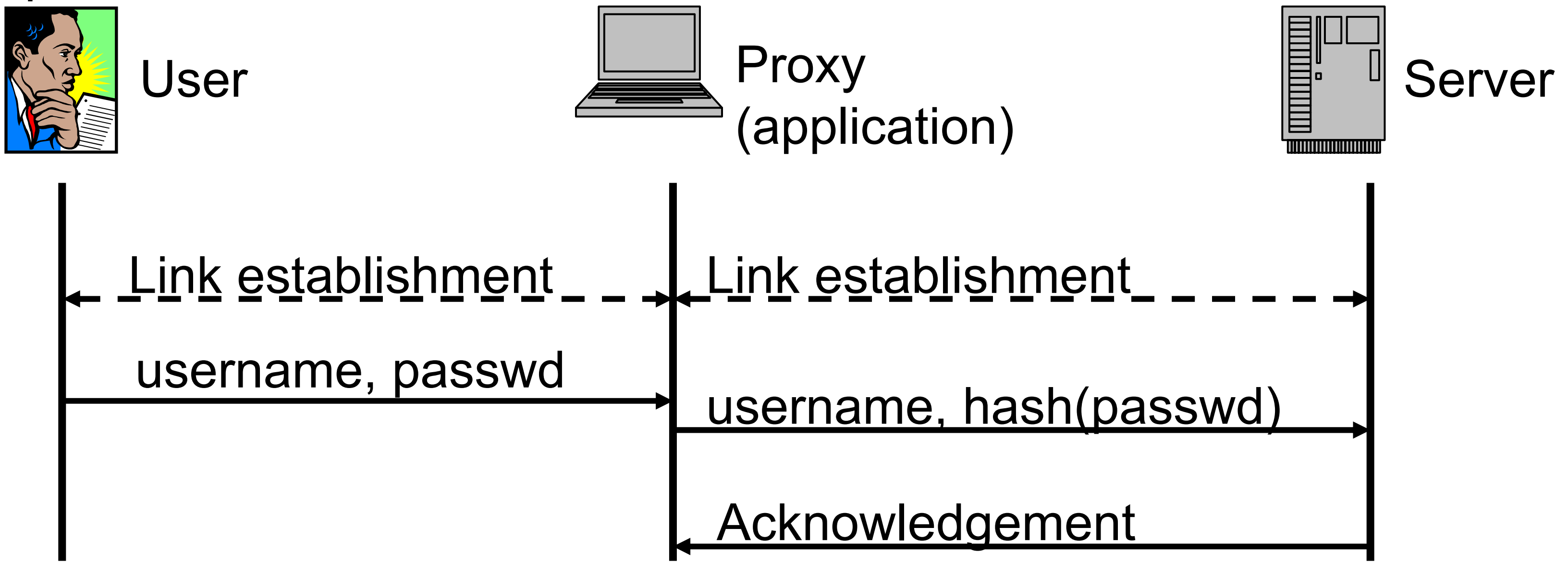
Hash functions

- A hash is a cryptographic **one way function** that produces a record from the plaintext
  - Sometimes called a fingerprint
- If y=f(x) is the hash, it should be impossible beyond attacker's resources to produce:
  - Inverse function g(y)=x
  - Chosen counterfeit message g(x')=y
- Suomeksi: tiiviste

TEKNILLINEN KORKEAKOULU

Authentication over a hostile network

- One way to improve security is to cryptographically hash the password

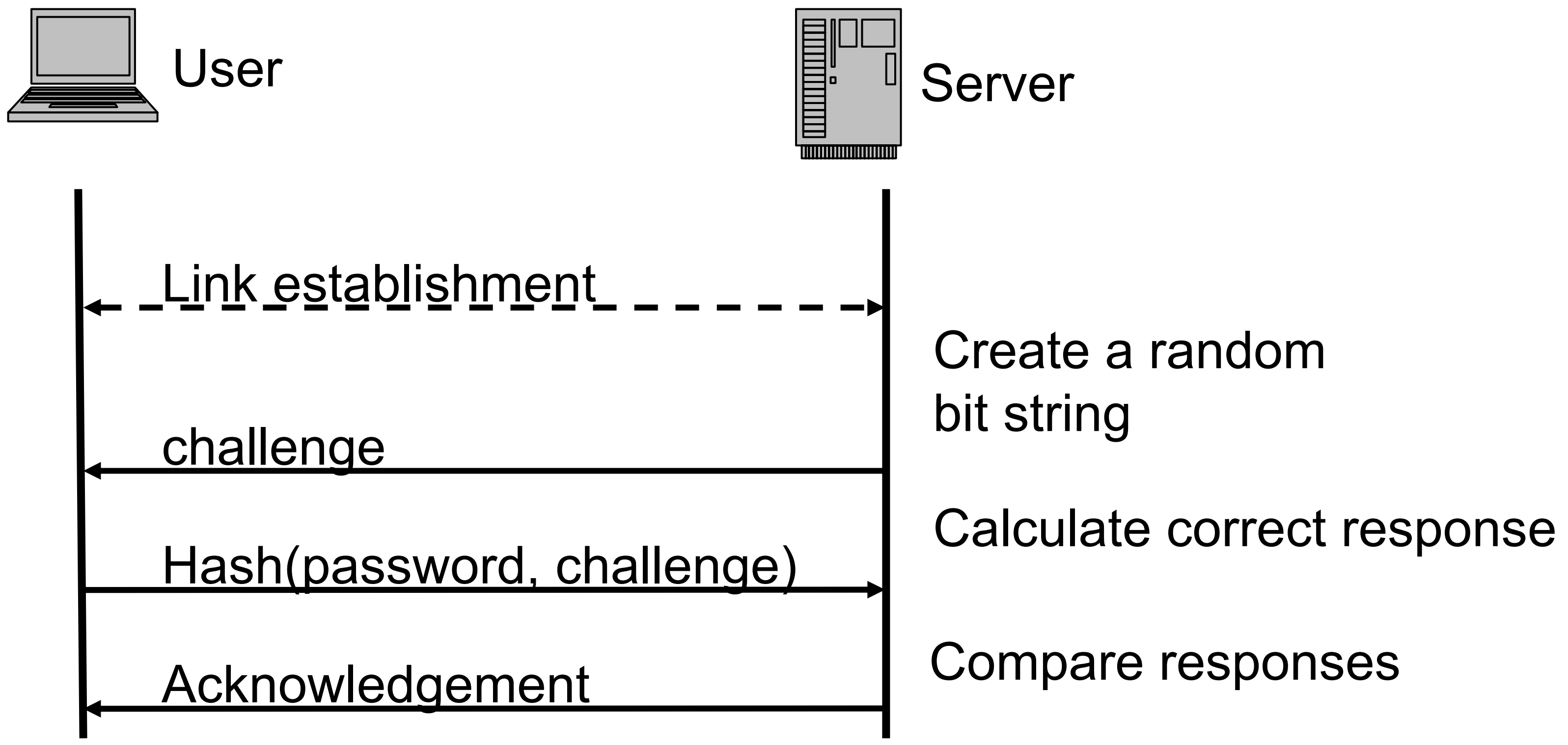


  - This refinement does not overcome the vulnerability for replay attacks
  - Password guessing is relatively easy unless passwords are long enough
    - The attacker knows the correct password, when he reaches a correct hash

Challenge-Handshake Authentication Protocol (CHAP)

- Used for authenticating dial-in users over a PPP link
- Based on the use of shared secrets
- Avoids sending passwords over a network
- The knowledge of the password is proved indirectly, using a one-way hash function
- RFC 1994 defines the packet format for CHAP messages sent encapsulated in PPP frames

The CHAP 3-Way Handshake


User
Server
Link establishment
Create a random bit string
challenge
Calculate correct response
Hash(password, challenge)
Compare responses
Acknowledgement

CHAP Security

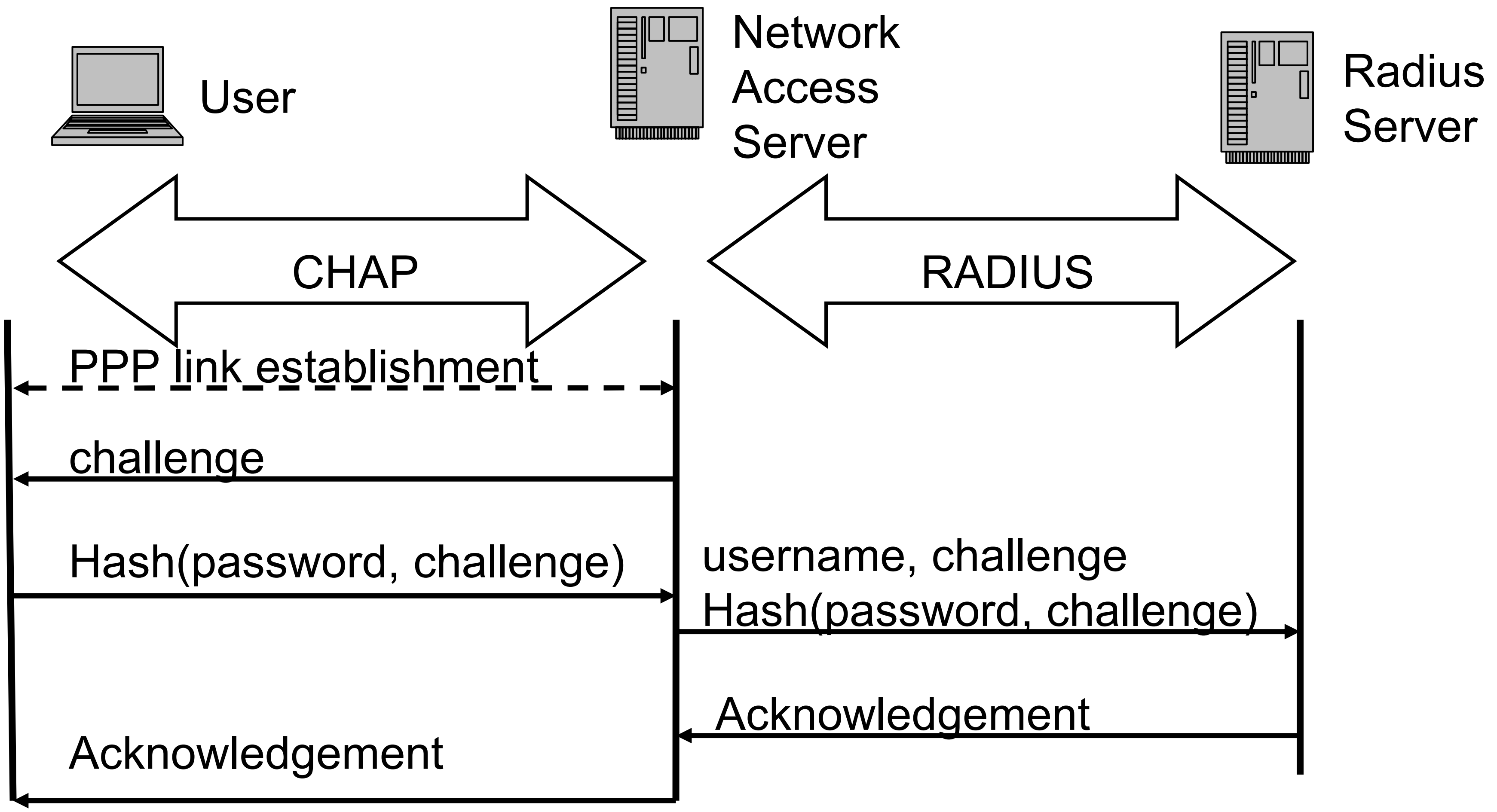
- The 128-bit MD5 algorithm is the default hash function used with CHAP
  - Without knowing the secret password, it is practically impossible to create a valid response to a given challenge
  - Password guessing is still possible
- The use of a random challenge eliminates the possibility for a replay attack
- Incapable of protecting against connection capture
  - The CHAP handshake procedure may be periodically repeated to limit the time of exposure to any single attack
- If desired, another CHAP handshake may be performed to authenticate the server to the dial-in user
  - However, it is ok to use different authentication protocols in each direction

Remote Authentication Dial-In User Service

- RADIUS is a protocol designed for transmitting authentication data between a Network Access Server (NAS) and a centralized authentication server
  - Used for dial-in modem servers and also in the 3G mobile networks
- It is desirable to have only one centralized user database for storing passwords and other sensitive user information
  - No management overhead in configuring many user databases
  - No need to distribute passwords to every access providing NAS
  - Centralized security management is easier and more cost effective
- A centralized server creates a requirement for low overhead
  - RADIUS packets are encapsulated in UDP datagrams
- RADIUS is used in combination with an authentication protocol, such as PAP or CHAP

TEKNILLINEN KORKEAKOULU

Authentication Using RADIUS and CHAP



TEKNILLINEN KORKEAKOULU

Using RADIUS and CHAP (cont.)

- As with an ordinary CHAP authentication, the procedure starts by NAS creating a challenge
- The dial-in user responds using his/her password with a one-way hash function
- After receiving the user's response, NAS wraps the challenge and the response with user ID in a RADIUS access-request and forwards this to a RADIUS server
- The RADIUS searches the password corresponding to the user ID and computes the hash values corresponding to the user password and the challenge
- If the two hash values match, the RADIUS server returns a access-accept message to the NAS
- NAS sends a successful CHAP acknowledgement to the dial-in user

TEKNILLINEN KORKEAKOULU

Properties of RADIUS

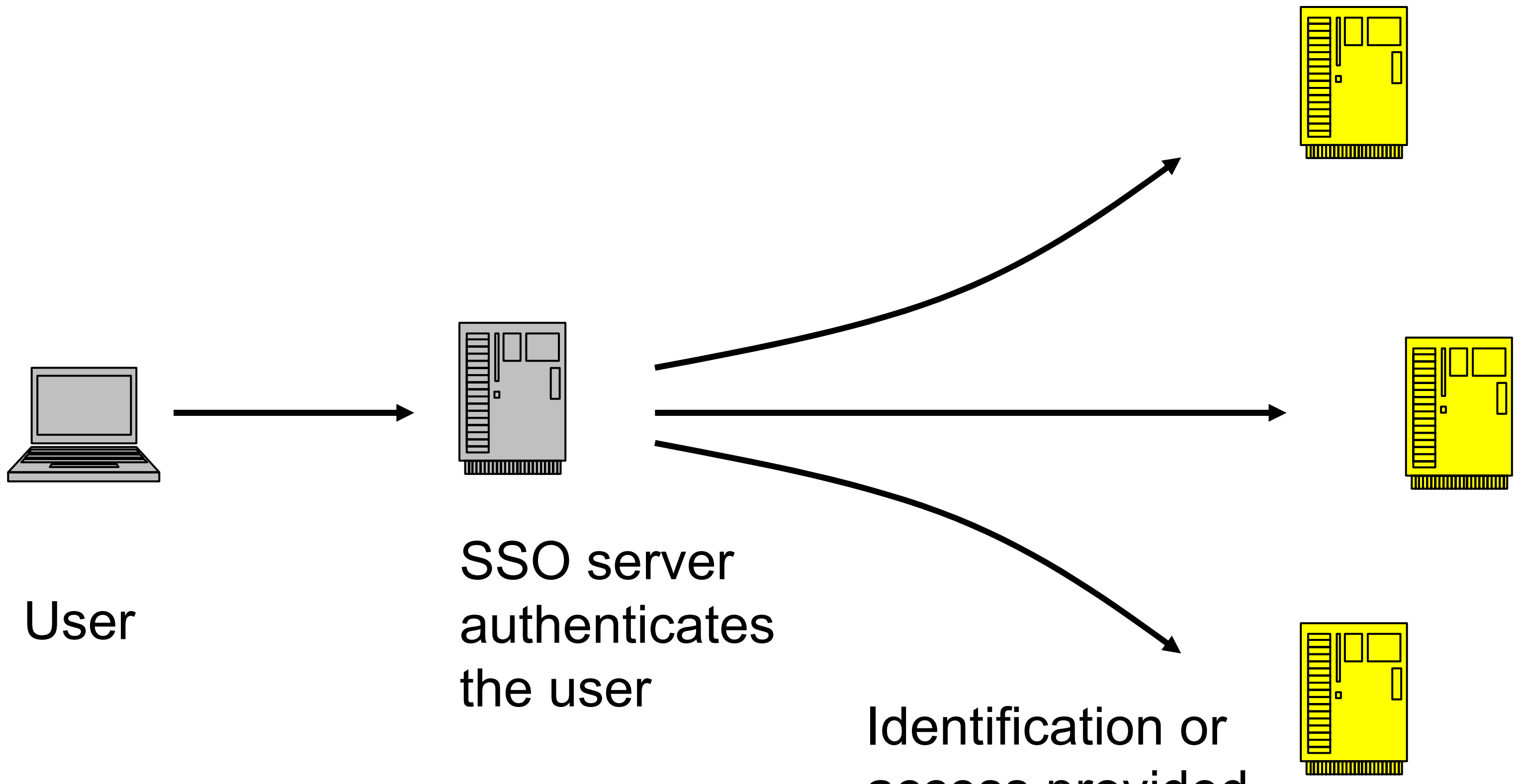
- RADIUS may also be used for sending user configuration information
  - For example, RADIUS server may manage in its database a static list of IP-addresses for PPP users which are assigned to users in the authentication procedure
  - Alternatively, RADIUS server may inform the NAS to dynamically allocate an IP-address from its pool to a requesting user
- RADIUS may also authenticate users of other services, for example Telnet
- The NAS and the RADIUS server use a shared secret for providing integrity and authenticity for RADIUS traffic between them
- The RADIUS protocol supports a wide variety of authentication methods
- As an extension, RADIUS may be used to transmit accounting information

Single Sign On (SSO)

- A concept for having the user to sign on (log in) to the network only once
  - Also being able to use one username and password for many different services
- The authentication information would be passed on to different network entities
- In theory this is a nice and workable idea
- In practice this nice idea has been difficult to reach
  - Everybody should trust everybody else for this to work
  - Crossing "domains of trust" is difficult
  - Different protocols, services, organizations, needs
- Global SSO is difficult but it is being worked on and might happen some day
- Local SSO is easier, and in use especially for WWW services

TEKNILLINEN KORKEAKOULU

Gateway style Single Sign On


User
SSO server authenticates the user
Identification or access provided to servers
Contents and services

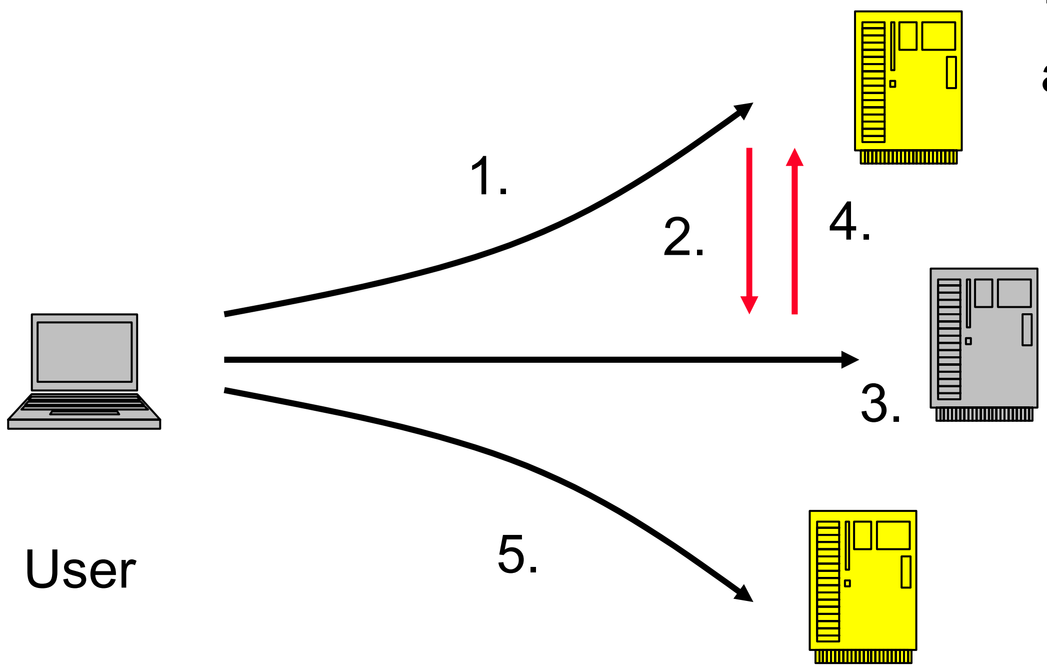
TEKNILLINEN KORKEAKOULU

Federated Authentication

- Federated authentication is currently used mostly with WWW services
- A set of services trust a set of authenticators
- When authentication is required a service in a web server passes the user to an authentication server
  - Session information is included in cookies or form *post* data
  - Some services can act also as authentication services
- The user is authenticated using some standard method (password over TLS etc.)
- Authentication information is passed forward with some credentials and additional information
- Cryptography is used to secure the data

TEKNILLINEN KORKEAKOULU

Federated SSO or Authentication



1. User accesses a service

2. User is redirected

3. SSO server authenticates the user

4. User returns to original service

5. Same authentication is valid for other services

Building a Global, Distributed SSO System

- You need a trusted third party that authenticates the users
  - Should it also authenticate the services?
- How to pass the authentication info to the service?
  - Potential for replay attacks
  - Timestamps are one way, but create synchronization problems
- The client's behavior should be controlled by the user
  - Having a smart card that authenticates any request is not a good idea

From Authentication to Authorization

- Authorization means controlling the actions of the user
  - Typically based on the authentication information
- Generally there is a need for some kind of mapping from the user identity to the authorized actions
- Also called Access Control
- Suomeksi: pääsynhallinta

TEKNILLINEN KORKEAKOULU

Other forms of authorization

- Physical presence may be used for authorization
  - Old single user operating systems used this principle
- Token based authorization
  - A colored string used at a rock festival authorizes the wearer to be present
- PKI based authorization
  - Instead of authentication certificates, give out authorization certificates

TEKNILLINEN KORKEAKOULU

What to Protect and How?

- Confidentiality: only authorized entities can read
- Integrity: only authorized entities can change
- Availability: authorized entities can use

⇒ **Access control**

TEKNILLINEN KORKEAKOULU

Access Operations

- What subjects can do to objects?
- In Unix, typical file access types are read, write, execute
- The file owner can control the permissions to these operations
- Windows has, in addition to these, permissions for delete, change ownership and change permissions
- Some systems define write so that it includes reading rights. These systems often have one operation more, append (blind write)

Access Control Matrix

- Access rights can be defined in form of access control matrix, where rows correspond to subjects and columns to objects. The matrix entry (subject, object) then has the allowed access rights
- Example:

	doc_1	passwd	progr_1
Alice	rw	r	x
Bob	r	r	-
Ronald	rw	rw	rwx

- In real systems, however, access control matrices are not very practical :
  - the matrix is usually sparse and there is a lot of redundancy
  - new subjects and objects can be added or removed easily, but the centralized matrix could become a bottleneck

Access Control Lists (ACL)

- We take objects as the starting point
- ACL is a column of the access control matrix
- Typically stored with the object itself
- Example:
  ACL for doc_1: {Alice: r,w; Bob: r; Ronald: r,w}
- Simple to implement, not so simple to manage
  - E.g. when a user is removed from the system, have to go through every ACL
- Unix file permissions can be seen as a simplified version of ACL
  - Note that Unix uses the file system to control access to other objects, too, like physical devices and processes
    - /dev , /proc

Capabilities

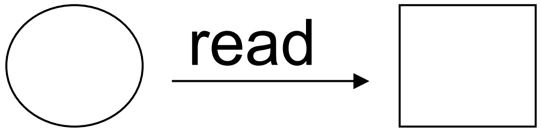
- Take subjects as the starting point
- Rows of access control matrix
- Example:
  Alice’s capability: {doc1: r,w; passwd: r; progr_1: x}
- Strengths:
  - Delegation of rights is easy
  - Runtime checking is more efficient compared to ACLs
- Weaknesses:
  - Difficult to find out which subjects have access rights to a specific object
  - Revoking delegated rights can be difficult
- Attribute or authorization certificates are capabilities for distributed systems (”holder of this key has the right to do x”)

Other Ways to Grant Access

- A *group* is a list of subjects with similar access rights. It is used as an intermediate access control layer
- A *role* is a collection of access permissions that a user or a set of users have in some context. Many systems implement roles as groups
- Some systems may use additional information for making access control decisions, e.g. what program the subject is using to access an object

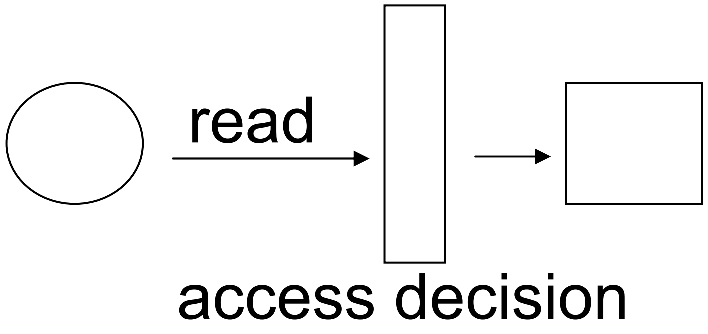
Subjects and Objects

- In computer security passive resources are called objects and active entities that utilize the resources are called subjects
- Typical *objects* include: files, directories, memory, printers, …
- Typical *subjects* include: users, processes, …
- The roles depend on situation: For example, a process can request access to some resource (act as a subject) and later be a target of access request (act as an object)
- Common practice in diagrams: subjects are depicted as circles and objects as squares



Reference Monitor

- A Reference monitor is the mechanism that uses (authenticated) identity, type of access requested and the object to allow or deny access
- Most operating systems implement a reference monitor at the kernel
  - When a process asks to open a file, the access is checked



Access Controls at Different Levels

Applications
Services/Middleware
Operating system
Hardware

- Access control and reference monitors can be implemented at many levels:
- Upper levels use the mechanisms of the level below
- Higher levels tend to have richer and more complex mechanisms, while lower level mechanisms tend to be more basic and reliable
- Also different on the network and inside one computer

Access Control and Time

- What happens if the configuration is changed and the user is logged in?
- Many systems cache some decisions
  - A Unix user can stay in the system even if the user is deleted from /etc/password
  - Open files can be accessed even if file permissions are changed
- TOCTTOU, Time Of Check To Time Of Use
  - A real problem
- Solutions need not be technical
  - E.g. a process that requires killing all of a user's processes if that user is removed from the system

Operating System Access Control

- Most computers have an operating system
- Most Internet services are application processes running on a multipurpose computer over an operating system
- The original purpose of the operating system is to
  - Provide common utilities
  - Abstract hardware and provide I/O interfaces
  - Limit users' and processes' access
- From security point of view we are studying the implications of
  - Rogue user
  - Misbehaving process

Operating System Security Paradigm

- Current multipurpose OSes have two divisions for software to run in
  - Kernel
  - User space
- Any programming code in the kernel space has full access to the computer it is running on
- A process running in the user space has access rights based on the User ID it is running under
  - On Unix UID 0 is reserved for the superuser or root and the kernel automatically gives this UID complete access
- Notice the difference between kernel and superuser access
  - Kernel processes can access anything
  - Root processes can order the kernel to access anything

TEKNILLINEN KORKEAKOULU

Unix Users and User Groups

- The system must be able to tell users apart
- User number (user id, uid) identifies an user
  - Internal information for the operating system
  - Usually not visible for users
  - Ordinary users have number > 0 (1 to 99 are used for different system accounts such as mail, news and sync)
  - Largest available user number on many systems is 60000
- Every user belongs also to at least one group
- Group number (group id, gid) identifies the group
  - Also internal information

User names and /etc/passwd file

- Users are referred to by user names
- Every user name points to a user number
- Several user names can have the same user number
  - The operating system holds these accounts (almost) identical
- /etc/passwd connects user names to numbers
  - Also NIS, Kerberos...
  - Passwords are often held in file /etc/shadow
  - Other mechanisms exist

TEKNILLINEN KORKEAKOULU

Format of /etc/passwd

- Fields are separated by colon

```
username:password:uid:gid:comment:home directory:shell
```

- Samples

```
root:z83c0d958LH2E:0:0:The Superuser:/:/bin/sh
bin:*:2:2:0000-Admin:/usr/bin:
ali:WiqIjY17WlPk6:202:100:Ali Baba:/home/ali:/bin/bash
jim:0w93R8u6nJ2NT:205:200:JimJones:/home/jim:/opt/bin/db-app
```

- Separate accounts for administrator's own use and administrator as root (shadow password system used here)

```
root:x:0:0:Big Brother Watching You:/:/bin/sh
rckent:x:0:108:Clark Kent as root:/:/bin/sh
ckent:x:108:108:Clark Kent:/home/ckent:/usr/bin/bash
```

/etc/passwd

- User name (login name)
- Password, encrypted
  - usually modified DES (MD5 is becoming quite common)
  - one way function, it is impossible to decrypt the password
  - at login the entered password is encrypted and compared to file
- User id (number)
- Login group id (number)
- GCOS (Comment, usually real-life name)
- Home directory
- Program to be executed at login, usually shell

Using Permissions

- A file has owner and group id (sometimes several)
- A process has owner and group id (sometimes several)
- Kernel verifies permissions before executing system calls
  - If owner uid=0 (root), everything is allowed
  - Otherwise the uid and gid of the process and object are compared in this order and permission for the operation is searched for based on owner, group and other (world) rights

Unix file permissions

- Permissions
  - r Read
  - w Write
  - x Execute or reference (for directories)
- For
  - Owner (User)
  - Group
  - World (Others)
- This file may be edited (rw) by its owner, read by the members of the "titu" group and not read by others

```
-rw-r----- 1 kiravuo titu 7627 Oct 1 12:50 exam
```

- This file may be edited by anybody:

```
-rw-rw-rw- 1 root root 12987 Sep 7 19:34 /etc/passwd
```

Unix permission levels and bits

- The information is coded in two bytes in the inode of the file

```
Bit 15                    Bit 0
xxxx s s t r w x rwx rwx
     |   |   |    |  permissions for others
     |   |   |     permissions for group
     |   |   | owner's x-permission
     |   |   owner's write permission
     |   | owner's read permission
     |   sticky bit
     | sgid bit
     suid bit
file type (-, d, c, b, l, s, p)
```

Processes

- A process is a data structure in the computer's memory
- It has
  - Program code to be executed
  - Data area
  - Stack
  - Associated data (owner, open files etc.)
- The operating system tells the CPU to execute the program code
- If the CPU architecture supports it, the process can usually access only its own memory area
  - Interprocess communications require that a specific mechanism is set up
- The process communicates to the kernel through system calls

Suid and Sgid Mechanisms

- Usually a process (program) inherits the properties of the parent process
  - In most cases shell
- Suid and sgid bits in file permissions allow executing a program with the file owner and group IDs
  - Necessary for Unix to work, for example the passwd command, which is owned by root and allows an user to change his or her password

Creating a Suid Program

- Easy:

```
chmod +s program
```

- or

```
chmod 4111 program
```

    - no read permission
- The program is responsible for limiting the acts of the user
- A shell script should never be made a suid program
    - Read it again: NEVER
    - This a hole that is trivial to exploit

TEKNILLINEN KORKEAKOULU

Changing passwords

- Passwords are changed using the program /bin/passwd
  - Users can change only their own password, requires old password
  - root user can change any password

```
# passwd username
```

- Password files not writable (not even root), how does changing password work?

```
-r-s--x--x 1 root sys 15688 Oct 25 1995 /bin/passwd
-r--r--r-- 1 root sys 7627  Oct 1 12:50 /etc/passwd
-r-------- 1 root sys 3292  Oct 2 15:14 /etc/shadow
```

Windows security

- Windows NT family
  - 2000, XP, Vista, 7
  - Windows 3.1, 95, 98 etc. are different
- Here compared to Unix
- Windows NT design team has famous VMS background
- While Windows is not related to Unix, the differences are mostly in details
- Also, Unix has minicomputer/server background, Windows microcomputer/networked workstation background

Windows Configuration

- Unix configuration is spread in many text files
  - Many, not all in the /etc directory
  - Some have their own APIs (like /etc/passwd)
- Windows has the registry database that is accessed through an API
- The centralized registry makes it easier to manage many hosts through the Windows Active Directory system
  - Similar tools exist for Unix, but the implementation is different

Windows Users

- Users are identified by a Security ID (SID)
  - A pseudorandom user ID and
  - A *domain* identifier
- Each user belongs to a particular domain, an administrative area
  - Domains are an important concept in Windows regarding networks and administration of networked machines
- Groups are identified by SIDs, contain SIDs (users and other groups)

Windows Processes (Subjects)

- A process has owner SID and additional data, like Unix
  - Group SIDs
  - Privileges related to Windows services, like backups and logging

Windows Files (Objects)

- Files' access rules are described in a security descriptor
  - Owner SID
  - An Access Control List (at user's discretion), DACL
  - A System Access Control List that can enforce actions, like generating an audit log
- The DACL provides great granularity in control
  - A list of user and group SIDs (Unix provides only one group)
  - Read, write and execute rights
  - Also access rights to attributes, append-only write, delete

TEKNILLINEN KORKEAKOULU

Windows Reference Monitor

- Checking the access rights is also more complex than in Unix
  - As the rights control are more detailed
- The "object manager" has to parse the request in regard of the DACL
- Like Unix, the reference monitor functionality is implemented in the kernel

Summary

- User authentication, especially in a distributed system is not trivial
- Authorization is usually based on the identity information verified by authentication
- Both authentication and authorization require information to be stored by both the subject and the object system
- Time is an important dimension for many issues

Sample Exam Questions

- Define: reference monitor, ACL
  - Some kind of definition that makes sense
- Justify T/F:
  - An ACL is an easy way to find one user's all access rights.
    - No way, requires tree traversal (if you know your algorithms, tree traversal really sucks)
  - Eavesdropping one time passwords is not a problem
    - Right, that's about the point of having a password that is used only once

Cont.

- How would you design an authentication system that protects against passive attackers (eavesdropping) and assumes that the user's computer has only username, password and a hash function?
  - You can get about 0.5 p for saying: "hire somebody who knows"
  - Think about the question, there is a user, but we have already assumed that the uid and pwd are inputted, thus we are assuming all the info is at the client computer
  - Apply one of the protocols in the lecture: Send RND1 from the server, send "UID, hash(RND1, password)"
  - Note: we assume that the server has password stored in clear
  - Note: on this course the crypto protocols do not have to be secure against all attacks, having a basic understanding is enough
  - Note: you can get points if you show that you don't have the actual answer, but you can discuss the **relevant** issues constricting the answer, e.g. "I send the UID and hash(PWD) and I know Eve can send the same, but I don't know how to protect against that" will not get you full points, but will get you some points

Cont.

- What is the Unix SUID mechanism (1p)
- Why is the SUID mechanism needed (2p)
  - to allow user perform actions, that are necessary, but would break the security model
  - to extend the security perimeter to individual pieces of software, that act as gatekeepers to allow certain actions
- Please note that exact details, like the message names in CHAP or the names of the fields in /etc/password are not required in the exam