Information Flow Control for Distributed Usage Control

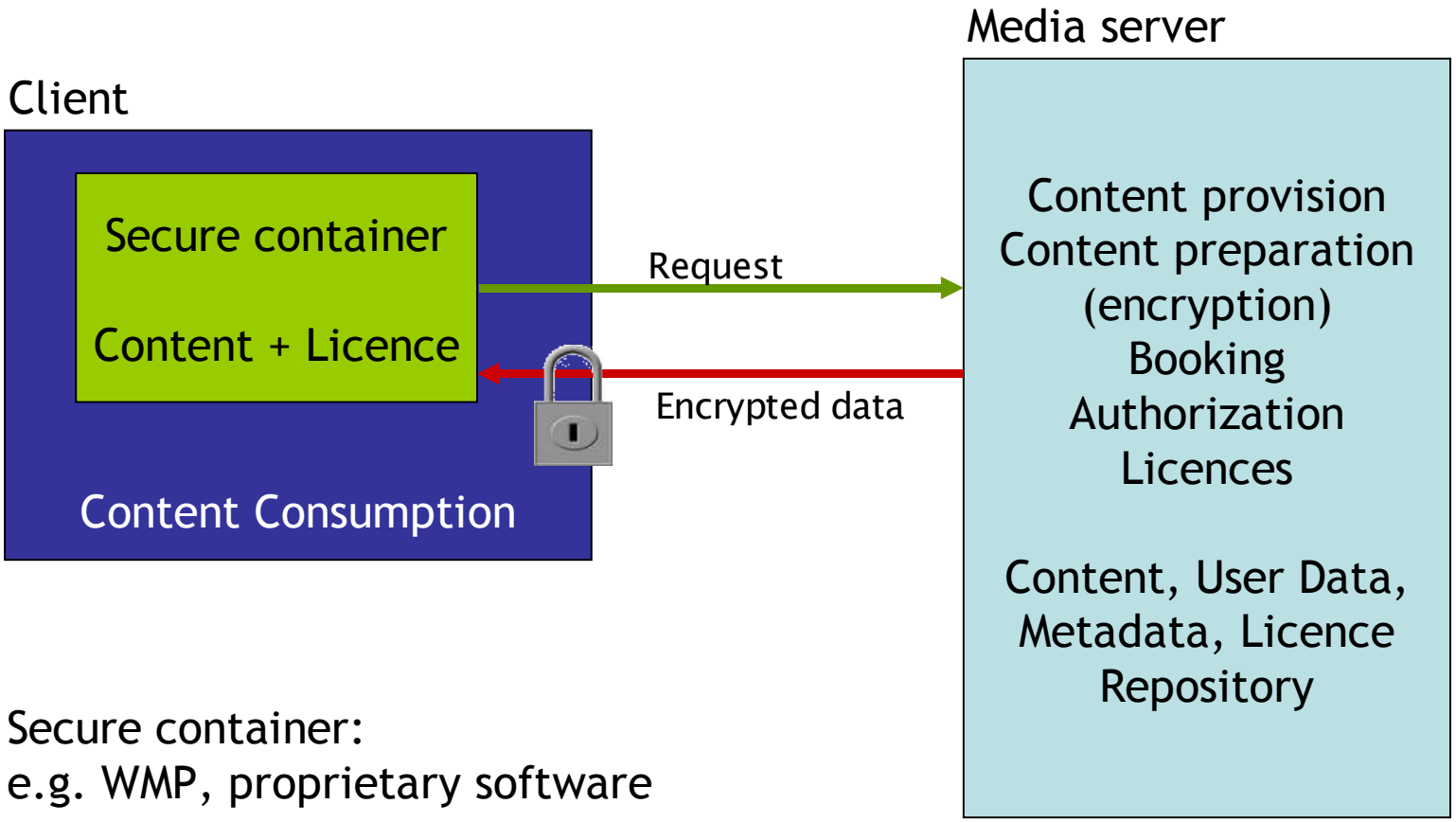
Dieter Hutter

German Research Center for Artificial Intelligence
Saarbrücken, Germany

Digital Right Management

- Content provider has intellectual rights on virtual goods
- Provider likes to restrict the usage of the content to individuals
- Required:
  - technical mechanisms to enforce intellectual rights
  - flexible mechanism to deal with content processing
  - Inheritage of intellectual rights during processing

Classical Technical Solution



Secure container:
e.g. WMP, proprietary software

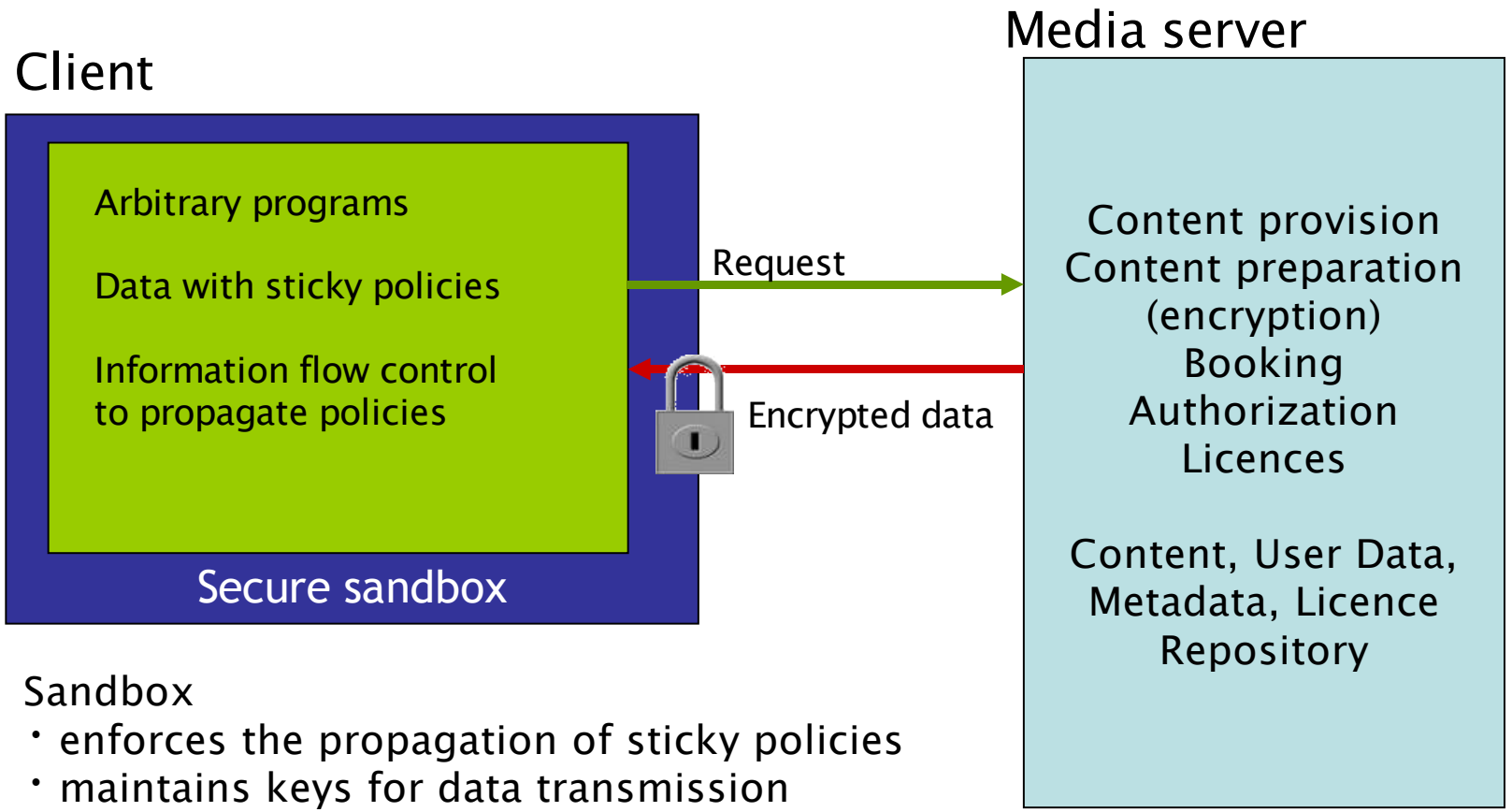
Classical Technical Solution – Problems

- Digital Right Enforcement by the use of trusted software
  - e.g. Windows Media Player
  - trusted software maintains digital rights
  - access control mechanisms
- Trusted software incorporate key management
- Trusted software is typically proprietary
- Usage of provided data is restricted to the abilities of trusted software

Usage Control of Data

- Access control operates on containers for information
  - containers have to be known before
- Usage control is a property of the information itself
- Sticky policy:
  - policy is attached to information
  - inheriting information requires inheriting security policies
  - problem of information flow control

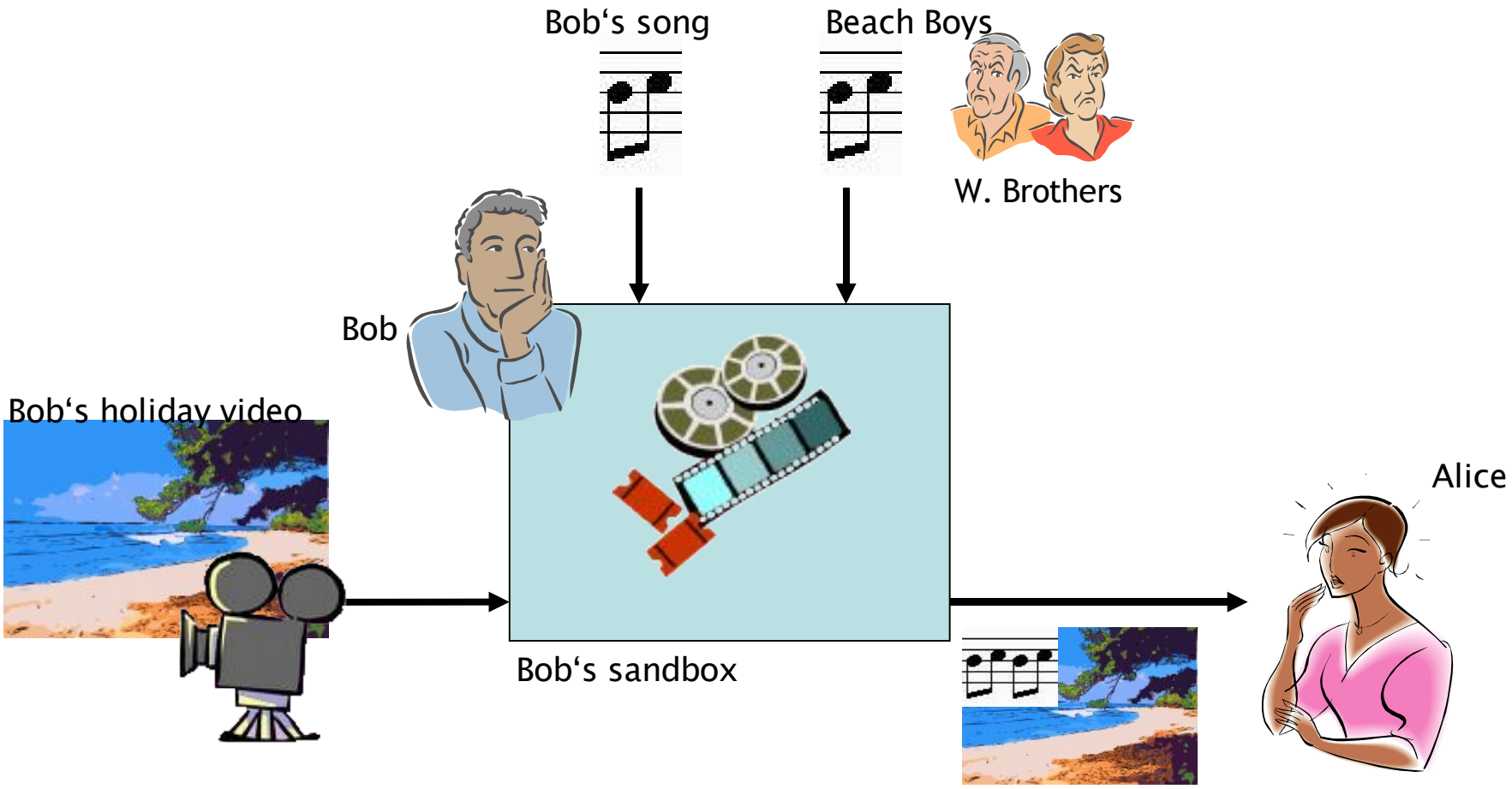
General Approach



Sandbox

- enforces the propagation of sticky policies
- maintains keys for data transmission

An Example


Bob‘s song
Beach Boys
W. Brothers
Bob
Bob‘s holiday video
Bob‘s sandbox
Alice

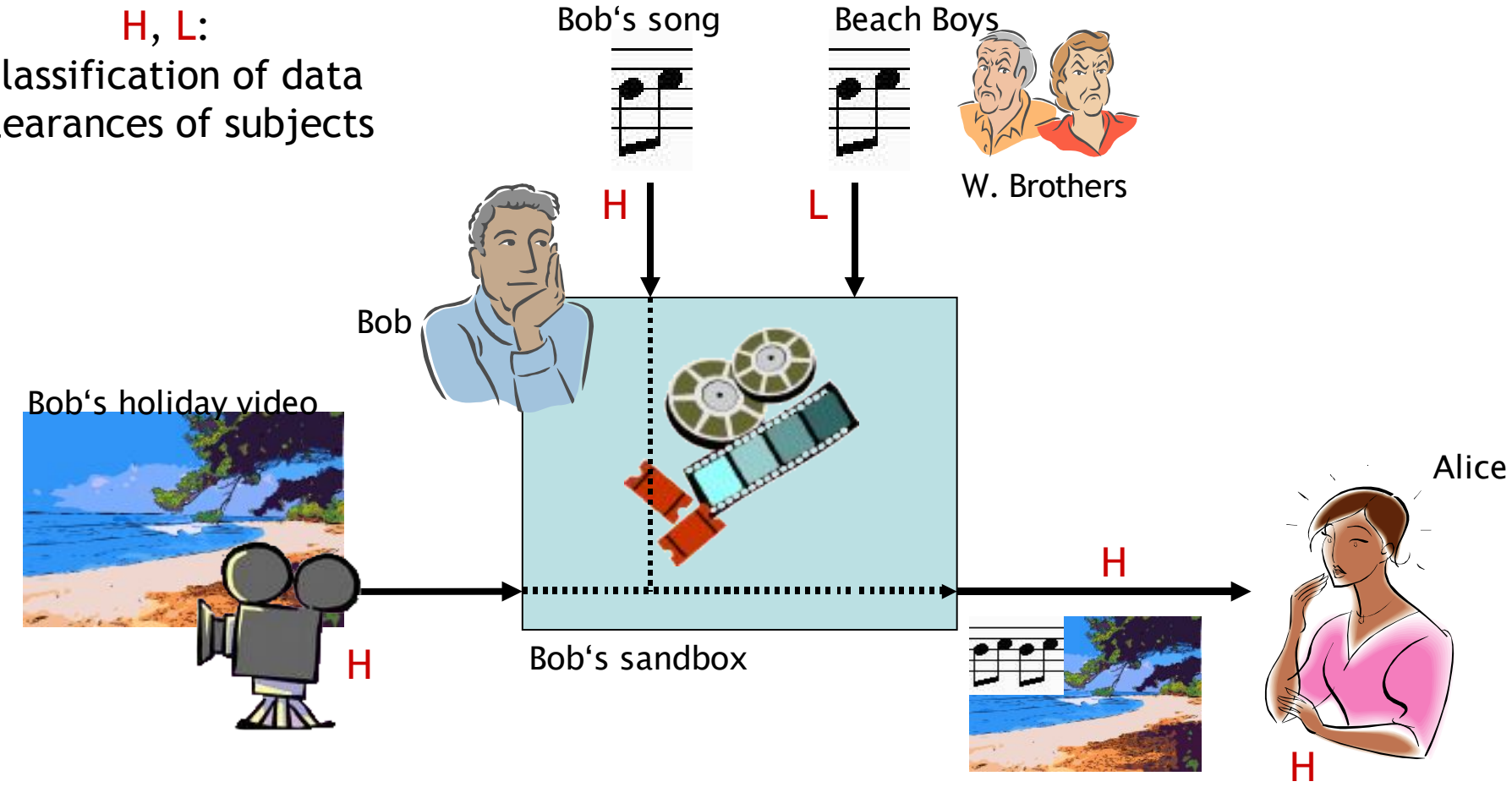
Security Policies

- Specification of a policy which web services are authorized to obain which information
- Approach from mandatory access control (Bell/LaPadula, Biba)
- Security types :
  - Lattice: lower upper bound, greatest upper bound
  - Classification of information
  - Clearances of web services

- Access to information is granted if clearance of web service is greater or equal then classification of information

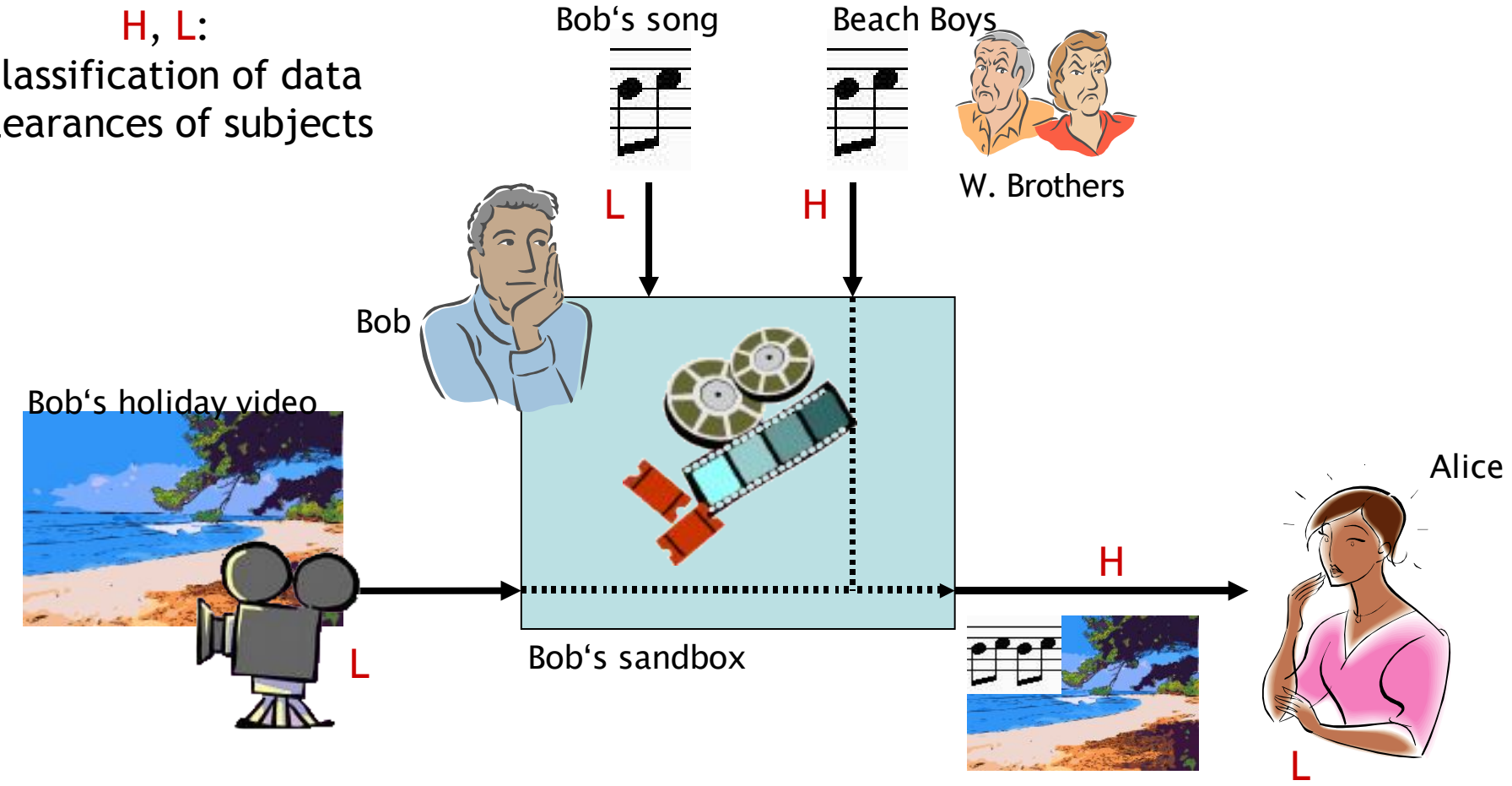
An Example: Bob‘s Privacy Issues

H, L:
classification of data
clearances of subjects



An Example: W. Brothers' issue

H, L:
classification of data
clearances of subjects



Privacy Policies

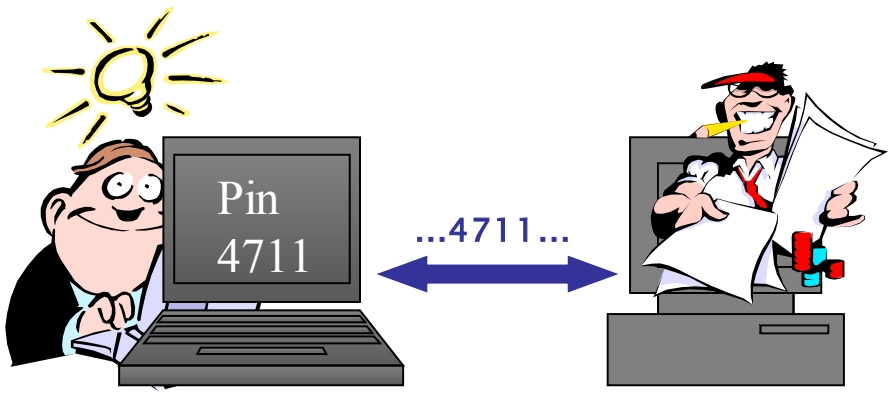
- Policies allow one to distinguish different „roles“
  - e.g. Bob‘s concerns, copy rights of W. Brothers
- Content provider defines access to his information by
  - classification of his data for each role
  - clearence of customers for each role

SP_{Data}	Bob’s priv.	W.Brother
Bob’s song	H	L
Beach Boys	L	H

SP_{WS}	Bob’s priv.	W. Brother
Bob	H	H
~~Alice~~	~~H~~	~~L~~
~~W. Broth.~~	~~L~~	~~H~~
	…	…

How to Detect Information Flow ?

Confidentiality as a property of dependencies:

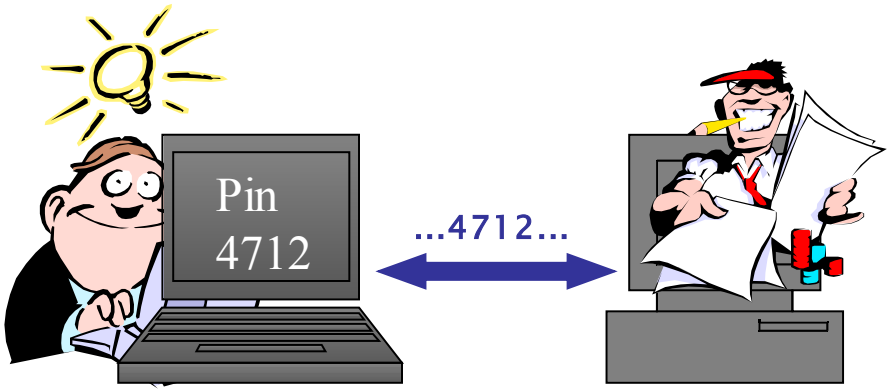


- The PIN 4711 is confidential.
- The information on PIN 4711 must not leave Bob‘s computer.
- First idea: PIN 4711 does not appear (explicitly) on the output line.
  - too strong, too weak
- Instead: The output of Bob‘s computer does not **depend** on the (setting of the) PIN.

Note: Confidentiality is formalized as a notion of dependency.

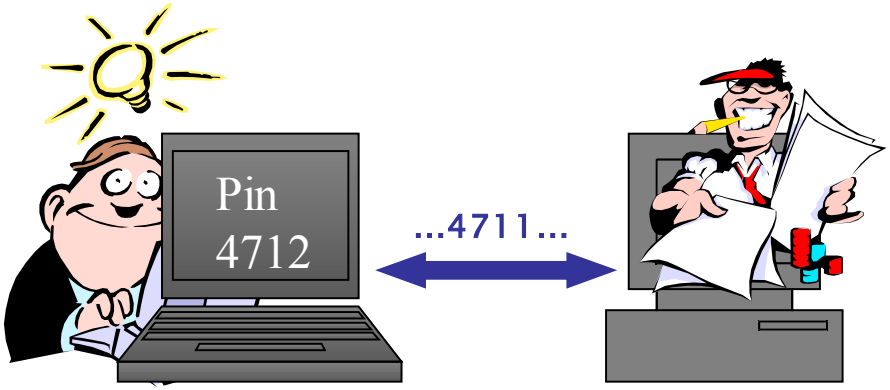
Confidentiality as Dependability

Confidential message: changing PIN (from 4711) to 4712



Insecure system:
output 4712 depends on PIN

Secure System:
output 4711 does not depend on PIN



In terms of web services:
Output to unauthorized web services does not depend on private data

Enforcing Information Flow Control

- Language based Information Flow Control
  - System is „specified“ in a programming language typically: sequences, conditionals, loops, procedures
  - Approaches by Volpano & Smith, Sabelfeld, …
  - Expressions and statements are labeled by types
  - Calculus rules define how to propagate types along the program
  - Program is secure if the program can be typed
  - Approaches are incomplete

Direct Information Flow

- Attaching types to all constructs that stores or provide information
  - controlling direct information flow
  - variable to store data has to have sufficient clearance (type)
  - $x_H := a_L$ is allowed
  - $x_L := a_H$ is not allowed !

Constraint: clearance of variable has to dominate the classifiction of the expression

Indirect Information Flow

- Attaching types to all program fragments
  - to control indirect information flow
  - who can observe the run of this fragment ?
  - if x_H = true then y_H := true else y_H := false; is allowed
  - if x_H = true then y_L := true else y_L := false; is not allowed !
    because observer knows which branch has been executed
  - y_L := false has type „L cmd“ while y_H := false has „H cmd“

Constraint: guard can only be as high as the minimum of the branches

The Role of Trust

Content provider has to *trust* the sandbox that it enforces the security policies

- Certificates (TTP, digital signatures) for web services incorporating security type checks
- Cryptographical means to link input and security policy and secure transmission between web services

Conclusion

- Ideas for realizing a fair DRM
  - incremental definition of a common security policy
  - enforcing security policies by security type calculus

- Future work:
  - Development of common ontologies for “roles” and lattices
  - Extending language to concurrency and shared memory