

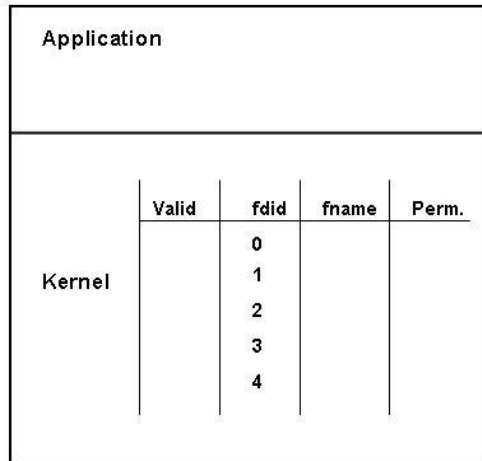
Topic: Access Control: Revocation and Authentication**Access Control: Revocation**

Capabilities Implementation: Two ways

1) *Use Unforgeable Pointer/Handles*

Handles that only the OS can create and pass to applications.

E.g. UNIX File Descriptors



- Application Source can pass the handle to the OS kernel with each request.
- In this method, application source have to pass the file descriptor to the OS for read/write requests.
- Read (fdid,)
 - This command identifies the objects.
 - represents the application's right to act on that object.
 - can only create entry in table via syscall that does some permission check.
- Capabilities are non-transferable.
- Revocation is easy: All you need to do is to remove the entry from the application's file table.

2) *Cryptography*

Message Authentication Code (MAC)

- similar to an Error Correcting Code.
- But you need to know secret key to compare MAC.

Illustration: Alice and Bob want to share a message. Both share a secret key 'K'.

Alice computes MAC (K, M) where 'M' is the message to be transferred.

Since, Bob shares the same secret key; he can verify that MAC (K, M) is still correct.

Example: HMAC SHA-256 Algorithm

The communication between the application and the OS can be shown as follows:



- Capability Transfer
 - Capability can be copied without OS intervention.
 - Capability is forever
 - Revocation is bad

- Capabilities can expire
Applications must renew capabilities
 $M' = (\text{foo}, R, \text{Exp})$

- Blacklist capabilities
To maintain a list of revoked capabilities

- Add process ID to capability
 $M' = (\text{foo}, R, \text{Exp}, \text{PID})$
Example: Kerberos

Authentication

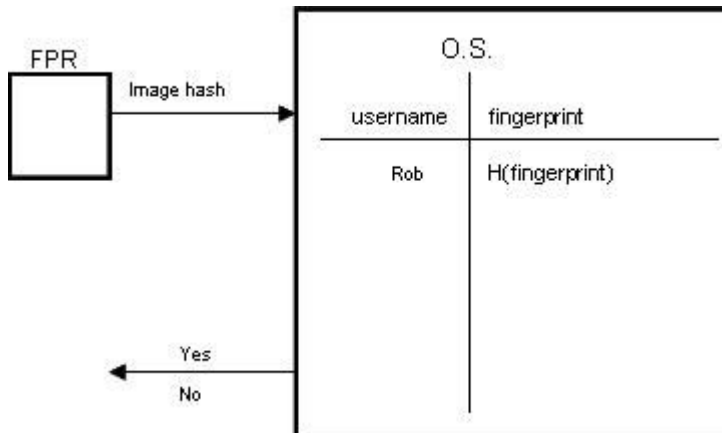
Question: How does the computer know you are who you say you are?

Answer: There are three factors for the computer to determine your identity

- 1) By something you know.
E.g.: Passwords
- 2) Something you are.
E.g.: Biometrics
- 3) Something you possess.
E.g.: Secure tokens

BIOMETRICS:

- Biometrics are a very simple password scheme
- Each reading differs slightly from the original reading, so we can only have approximate match.
- Cannot use fingerprints as keys.



Example: Iris Scans

- Iris Scan is converted to a 2048-bit string
- Two scans of the same iris agree on ≥ 1600 bits
- Two scans of different iris agree on ≤ 1200 bits

Biometric Attacks

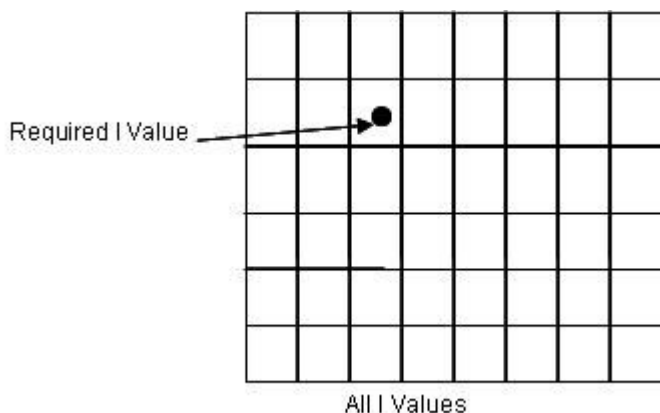
- A photograph of the iris can be used to fool/misguide the system.
However, liveness detector can be used to detect such behavior and verify proper iris.
- In case of thumbprints, gummi bears can be also used apart from photographs to misguide and attack the system.
Even in these cases, liveness detectors can be used to detect such behavior and prevent attack on the system.

There is an Error Correcting Code (ECC) that can correct 400-bit errors in a 2048-bit message and no more.

The iris scan digitized value 'I' has the above ECC applied to form the message with the error correcting code (I, CRC_I) which is stored in the system.

- 1) When the iris is scanned, we get the value I' which is passed to the system for verification.
- 2) The O.S. stores the username along with the CRCI value for each user.
- 3) Now, the value I' obtained from the present iris scan and the CRCI value is applied to the reverse ECC algorithm, ECC-1 to obtain the value of I.
- 4) If the value of I' is valid then the value obtained is of I, else if value of I' is invalid, the value obtained is J and not I.
- 5) If only I is obtained, then the user can have access the files in the file system. The file system is encrypted with the key $K = H(I)$.

- The value of CRCI is public.
- CRCI is useful in the above scenario to determine the value of I from all possible I values.



- The required I value will be at the center of one of the cells.