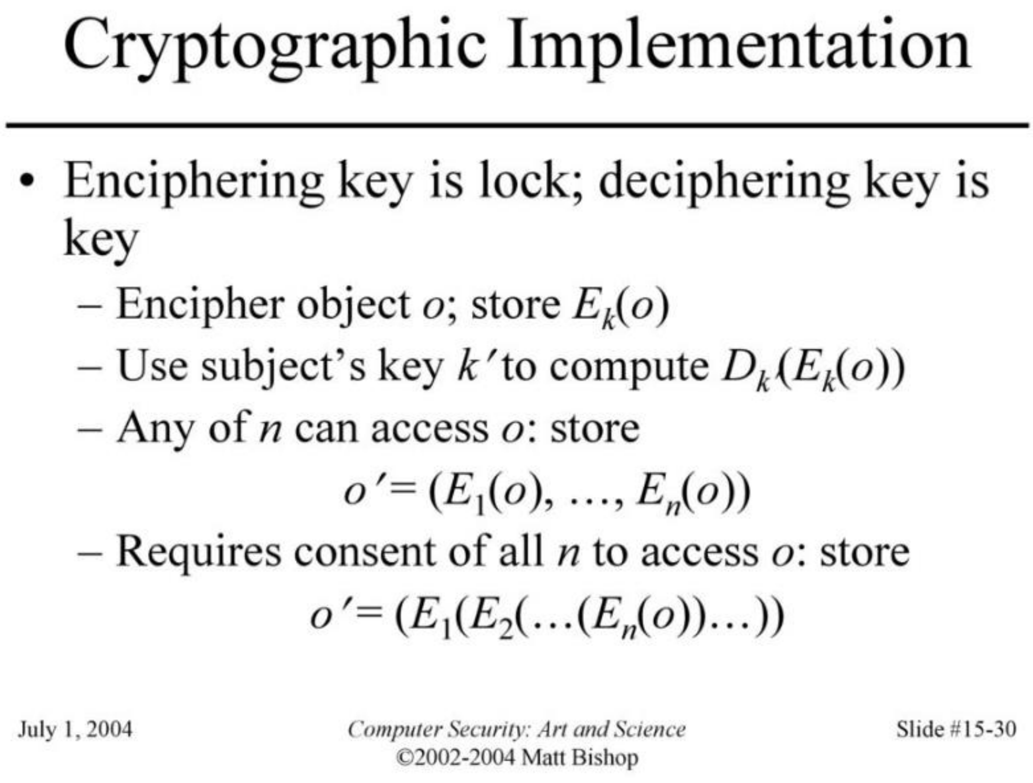
Locks and Keys

The locks and keys technique combines features of access control lists and capabilities. A piece of information (the lock) is associated with the object and a second piece of information (the key) is associated with those subjects authorized to access the object and the manner in which they are allowed to access the object. When a subject tries to access an object, the subject's set of keys is checked. If the subject has a key corresponding to any of the object's locks, access of the appropriate type is granted.

The difference between locks and keys and the other access control mechanisms is the dynamic nature of the former. An access control list is static in the sense that all changes to it are manual; a user or process must interact with the list to make the change. Locks and keys, on the other hand, may change in response to system constraints, general instructions about how entries are to be added, and any factors other than a manual change.

Type Checking

Type checking restricts access on the basis of the types of the subject and object. It is a form of locks and keys access control, the pieces of information being the type. Systems use type checking in areas other than security.



Cryptographic Implementation

- Enciphering key is lock; deciphering key is key
  - Encipher object o; store $E_k(o)$
  - Use subject's key k' to compute $D_{k'}(E_k(o))$
  - Any of n can access o: store

$$o' = (E_1(o), \ldots, E_n(o))$$

  - Requires consent of all n to access o: store

$$o' = (E_1(E_2(\ldots(E_n(o))\ldots))$$

July 1, 2004

Computer Security: Art and Science
©2002-2004 Matt Bishop

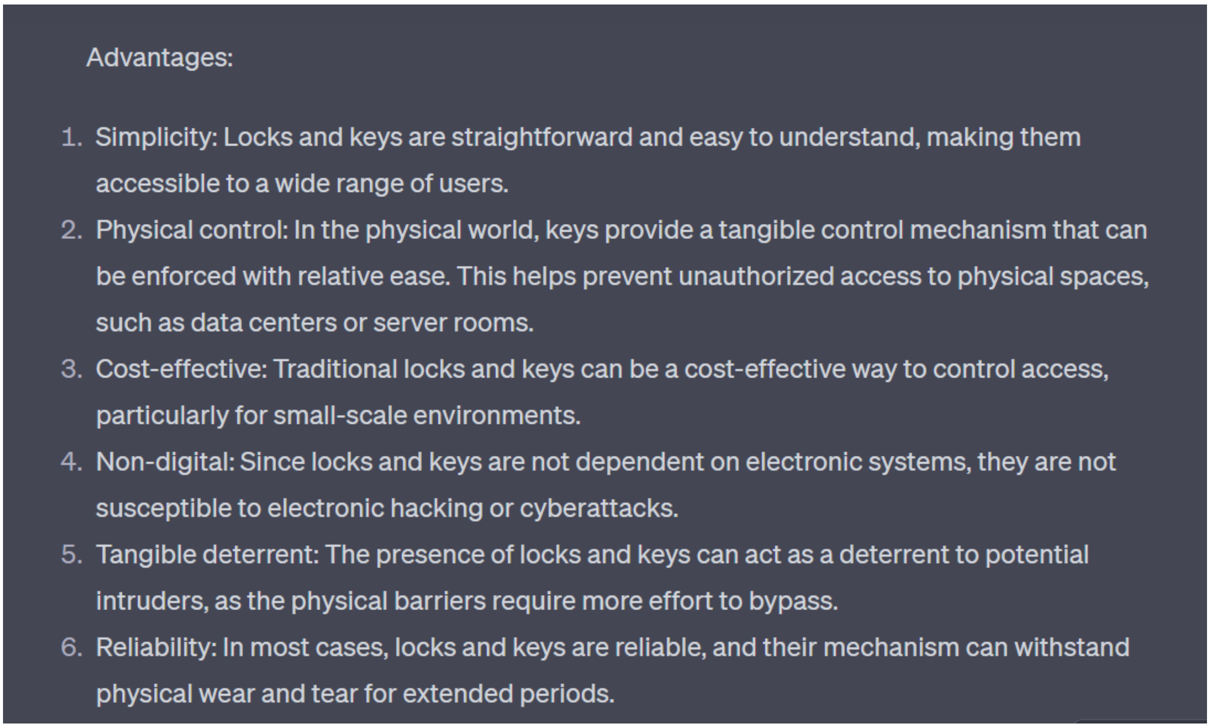
Slide #15-30

Example: IBM

- IBM 370: process gets access key; pages get storage key and fetch bit
  - Fetch bit clear: read access only
  - Fetch bit set, access key 0: process can write to (any) page
  - Fetch bit set, access key matches storage key: process can write to page
  - Fetch bit set, access key non-zero and does not match storage key: no access allowed

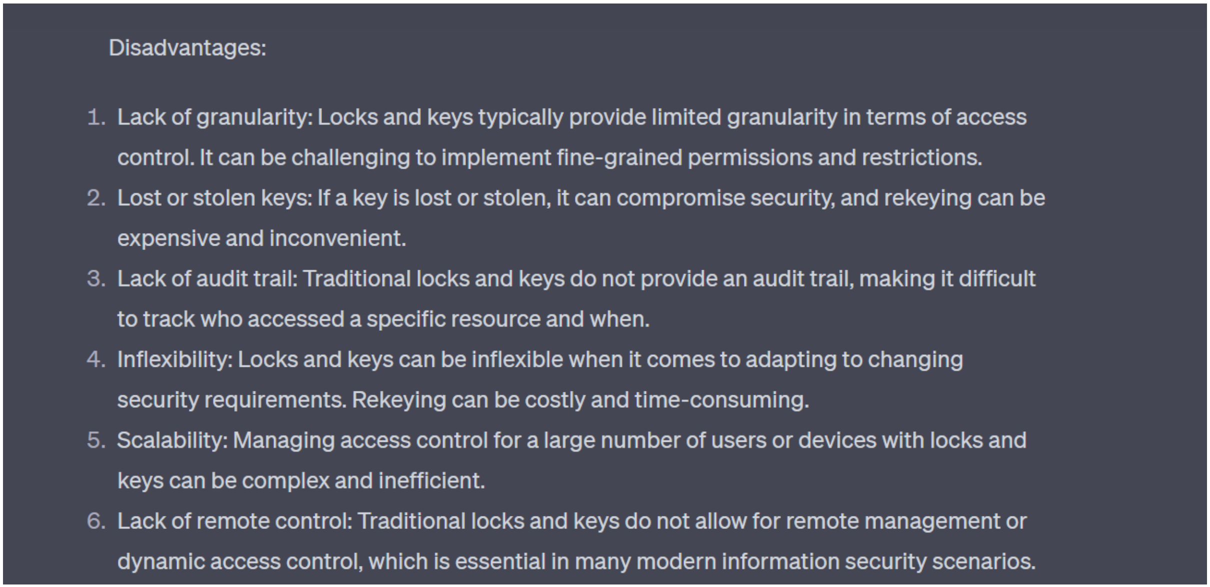
Sharing Secrets

- Implements separation of privilege
- Use (t, n)-*threshold scheme*
  - Data divided into n parts
  - Any t parts sufficient to derive original data
- Or-access and and-access can do this
  - Increases the number of representations of data rapidly as n, t grow
  - Cryptographic approaches more common



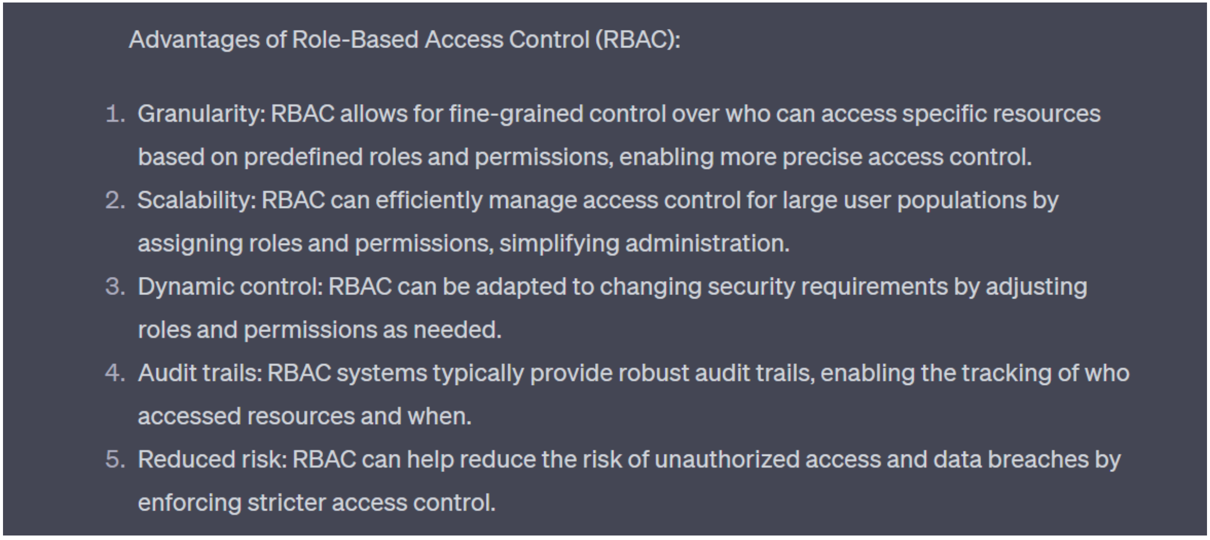
Advantages:

1. Simplicity: Locks and keys are straightforward and easy to understand, making them accessible to a wide range of users.
2. Physical control: In the physical world, keys provide a tangible control mechanism that can be enforced with relative ease. This helps prevent unauthorized access to physical spaces, such as data centers or server rooms.
3. Cost-effective: Traditional locks and keys can be a cost-effective way to control access, particularly for small-scale environments.
4. Non-digital: Since locks and keys are not dependent on electronic systems, they are not susceptible to electronic hacking or cyberattacks.
5. Tangible deterrent: The presence of locks and keys can act as a deterrent to potential intruders, as the physical barriers require more effort to bypass.
6. Reliability: In most cases, locks and keys are reliable, and their mechanism can withstand physical wear and tear for extended periods.



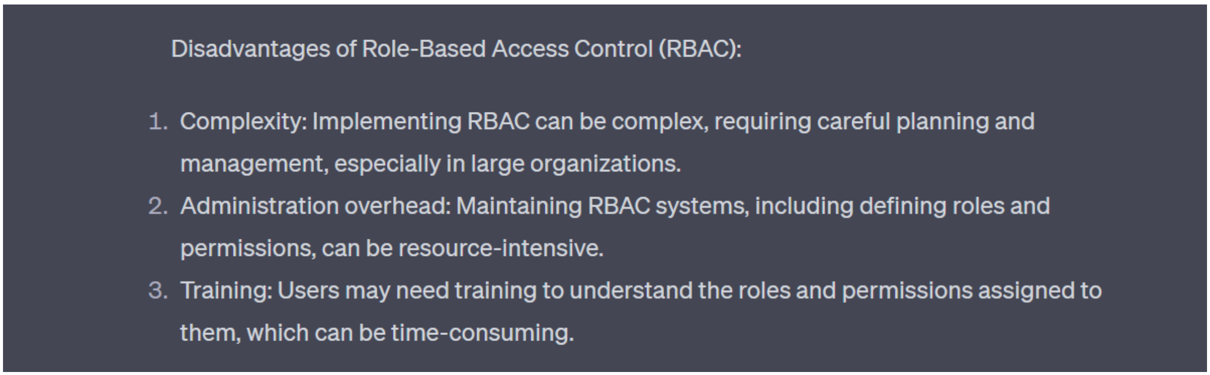
Disadvantages:

1. Lack of granularity: Locks and keys typically provide limited granularity in terms of access control. It can be challenging to implement fine-grained permissions and restrictions.
2. Lost or stolen keys: If a key is lost or stolen, it can compromise security, and rekeying can be expensive and inconvenient.
3. Lack of audit trail: Traditional locks and keys do not provide an audit trail, making it difficult to track who accessed a specific resource and when.
4. Inflexibility: Locks and keys can be inflexible when it comes to adapting to changing security requirements. Rekeying can be costly and time-consuming.
5. Scalability: Managing access control for a large number of users or devices with locks and keys can be complex and inefficient.
6. Lack of remote control: Traditional locks and keys do not allow for remote management or dynamic access control, which is essential in many modern information security scenarios.



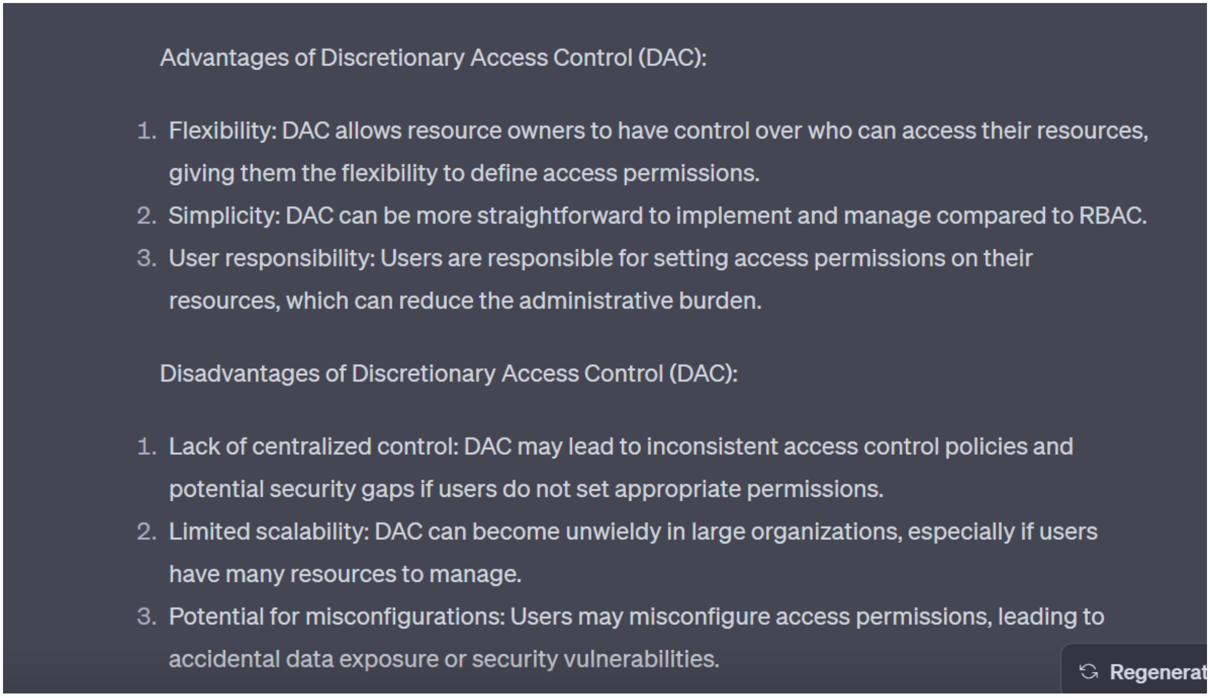
Advantages of Role-Based Access Control (RBAC):

1. Granularity: RBAC allows for fine-grained control over who can access specific resources based on predefined roles and permissions, enabling more precise access control.
2. Scalability: RBAC can efficiently manage access control for large user populations by assigning roles and permissions, simplifying administration.
3. Dynamic control: RBAC can be adapted to changing security requirements by adjusting roles and permissions as needed.
4. Audit trails: RBAC systems typically provide robust audit trails, enabling the tracking of who accessed resources and when.
5. Reduced risk: RBAC can help reduce the risk of unauthorized access and data breaches by enforcing stricter access control.



Disadvantages of Role-Based Access Control (RBAC):

1. Complexity: Implementing RBAC can be complex, requiring careful planning and management, especially in large organizations.
2. Administration overhead: Maintaining RBAC systems, including defining roles and permissions, can be resource-intensive.
3. Training: Users may need training to understand the roles and permissions assigned to them, which can be time-consuming.



Advantages of Discretionary Access Control (DAC):

1. Flexibility: DAC allows resource owners to have control over who can access their resources, giving them the flexibility to define access permissions.
2. Simplicity: DAC can be more straightforward to implement and manage compared to RBAC.
3. User responsibility: Users are responsible for setting access permissions on their resources, which can reduce the administrative burden.

Disadvantages of Discretionary Access Control (DAC):

1. Lack of centralized control: DAC may lead to inconsistent access control policies and potential security gaps if users do not set appropriate permissions.
2. Limited scalability: DAC can become unwieldy in large organizations, especially if users have many resources to manage.
3. Potential for misconfigurations: Users may misconfigure access permissions, leading to accidental data exposure or security vulnerabilities.