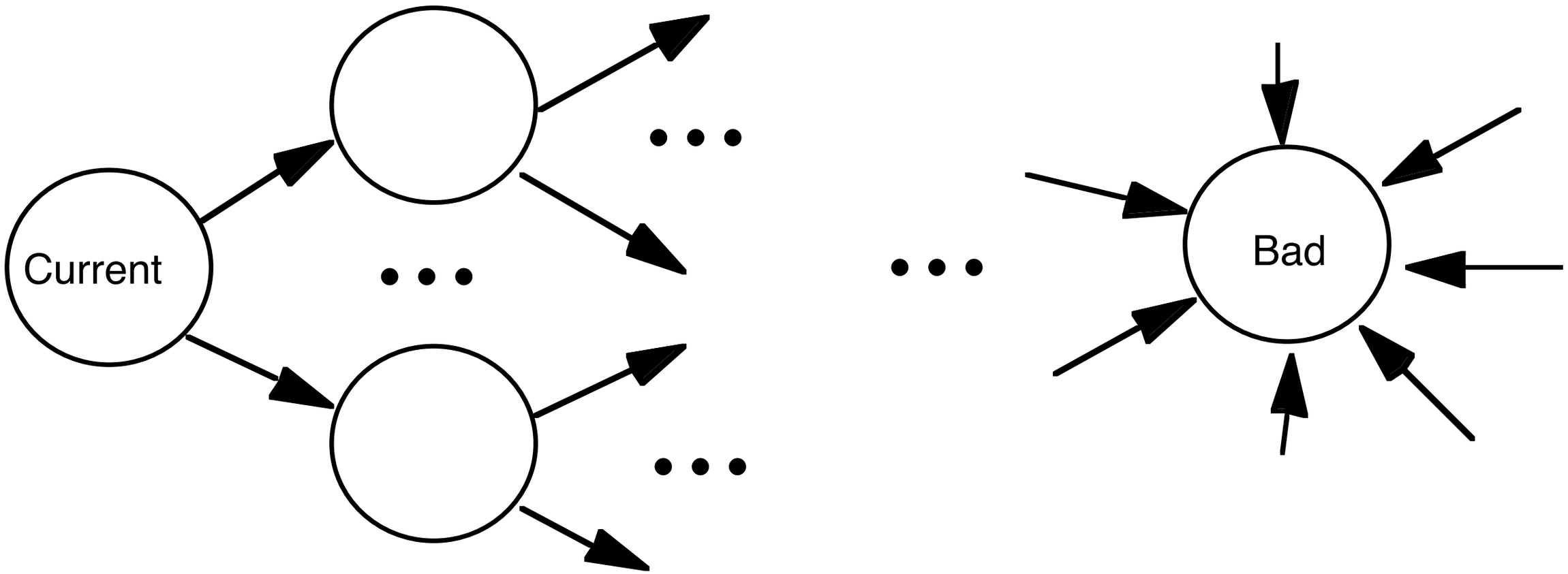
Information security. The protection of information and information systems against unauthorized access or modification of information, whether in storage, processing, or transit, and against denial of service to authorized users.

Information Operations. Joint Chiefs of Staff of the United States Armed Forces, Joint Publication 3-13 (13 February 2006).

Complete mediation

For every requested action, check authenticity, integrity, and authorization.



Open design principle

Let anyone comment on the design. You need all the help you can get.

Minimize secrets

Because they probably won't remain secret for long.

Economy of mechanism

The less there is, the more likely you will get it right.

Minimize common mechanism

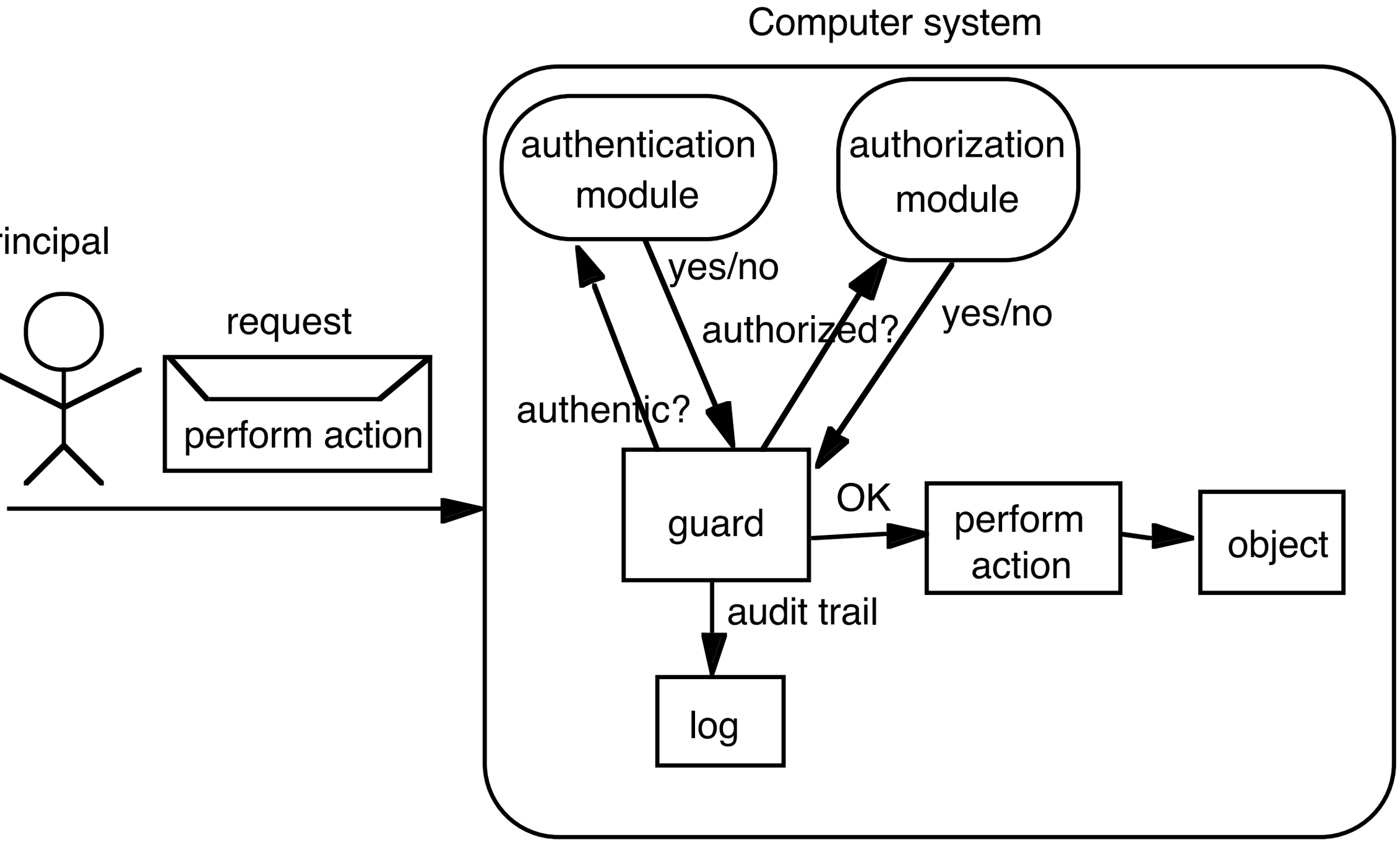
Shared mechanisms provide unwanted communication paths.

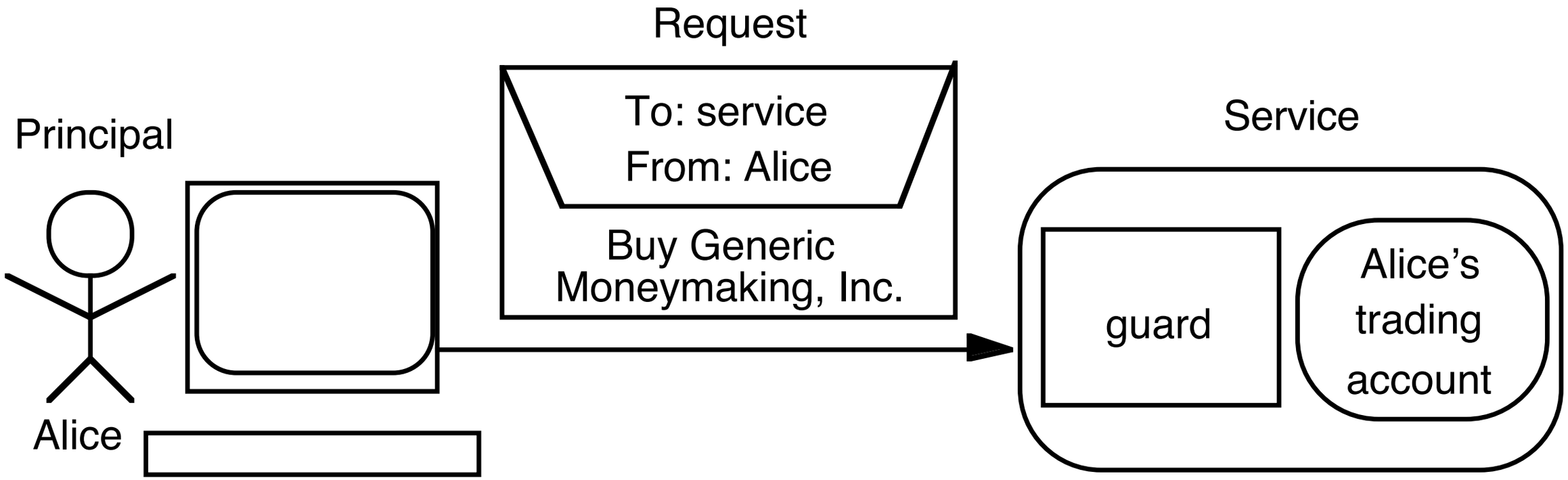
Fail-safe defaults

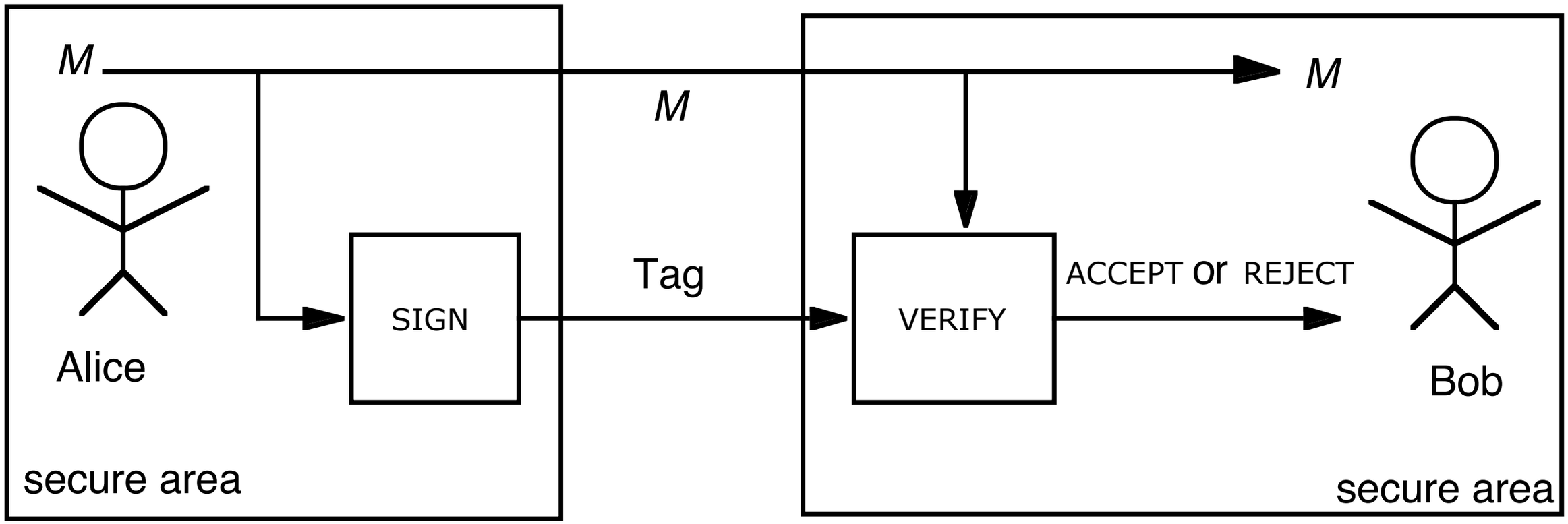
Most users won't change them, so make sure that defaults do something safe.

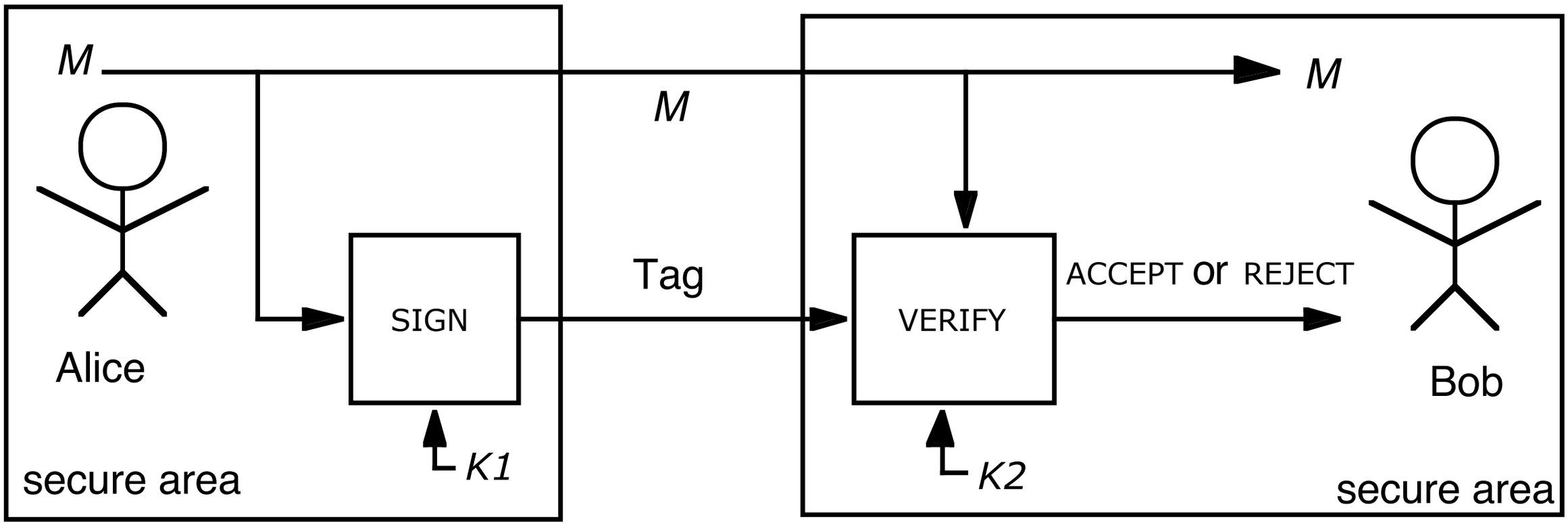
Least privilege principle

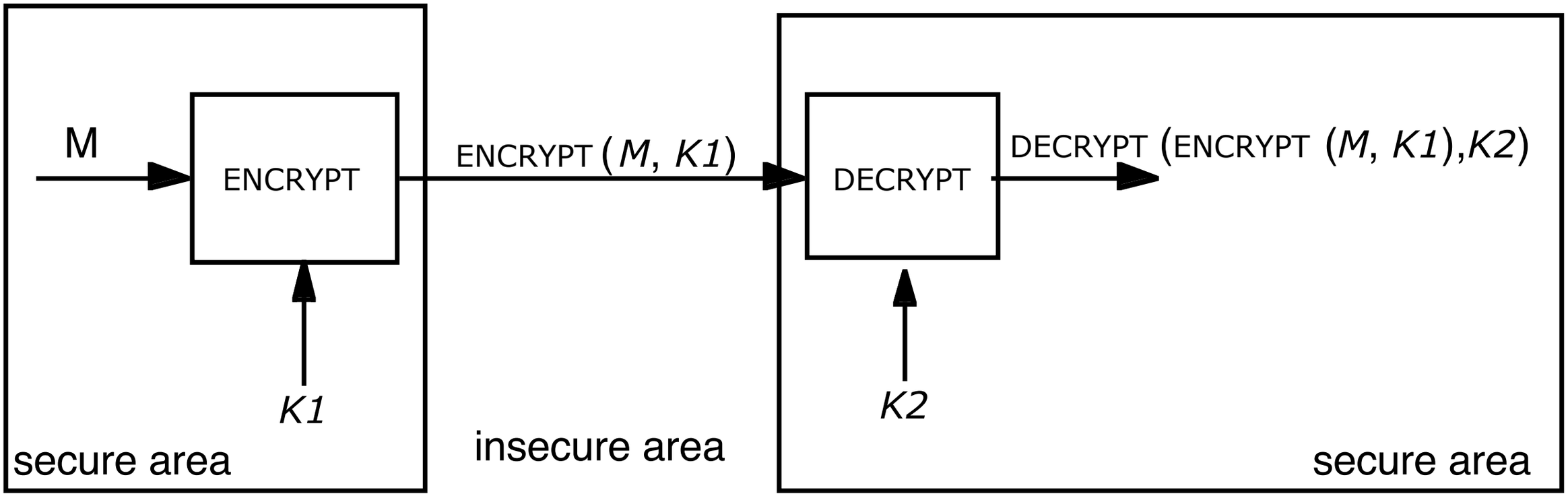
Don’t store lunch in the safe with the jewels.





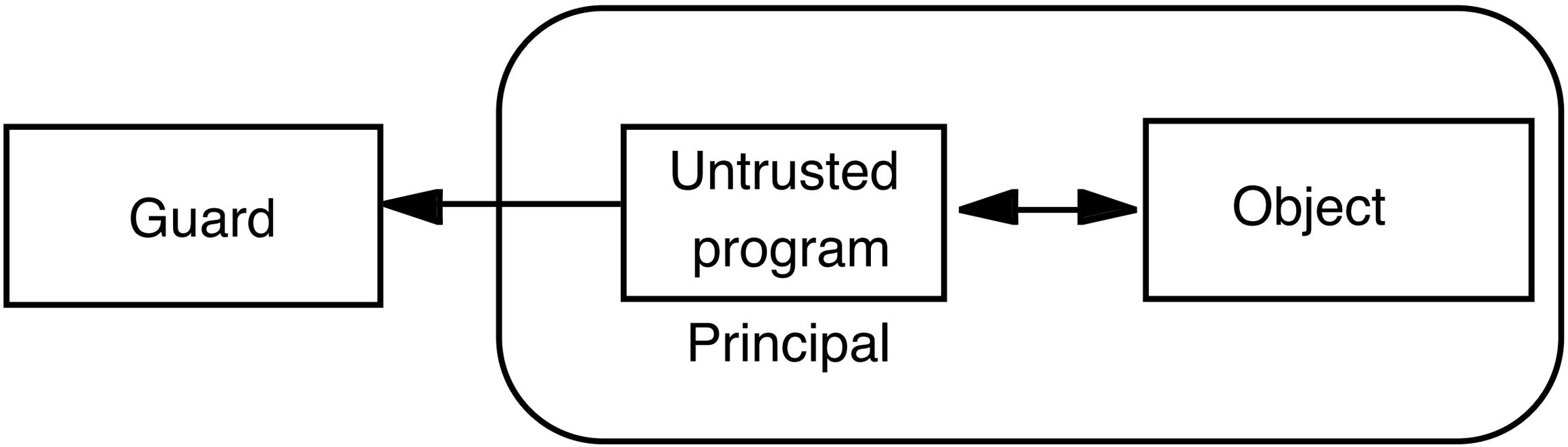


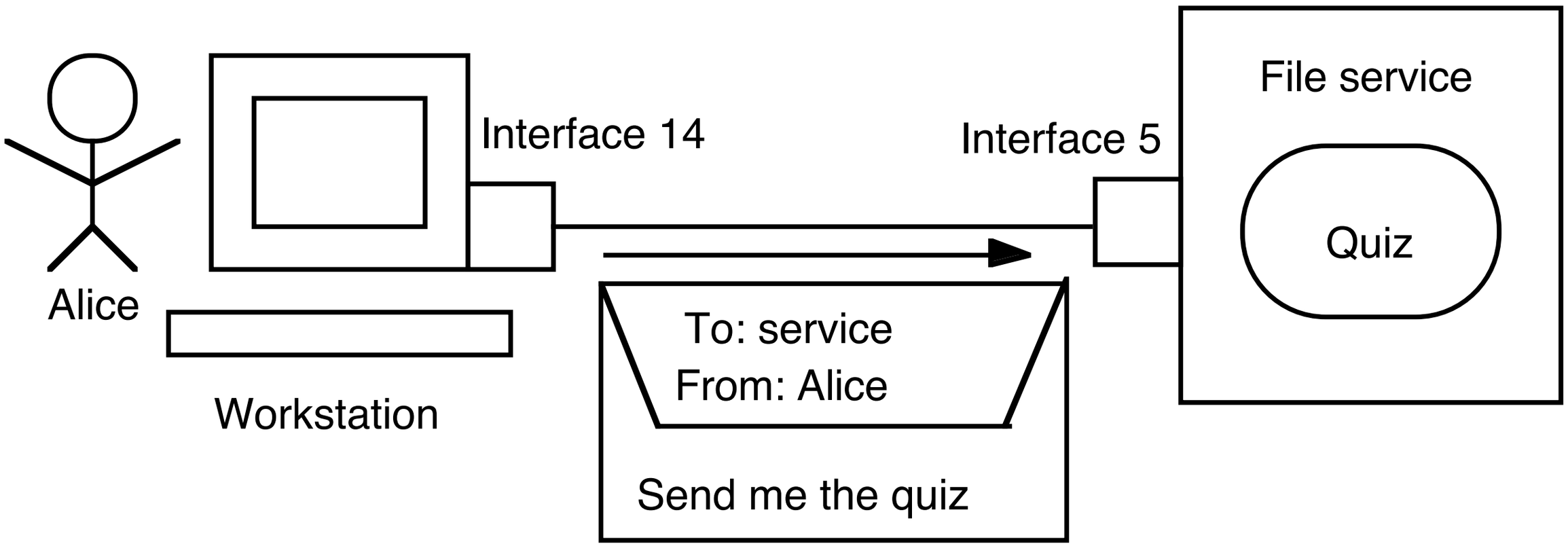




Comparison of access control systems

System	Advantage	Disadvantage
Ticket	Quick access check	Revocation is difficult
	Tickets can be passed around	Tickets can be passed around
List	Revocation is easy	Access check requires searching a list
	Audit possible	
Agency	List available	Revocation might be hard





Rule 1: Delegating authority:

If	A **says** (B **speaks for** A)
then	B **speaks for** A

Rule 2: Use of delegated authority.

If	A **speaks for** B
and	A **says** (B **says** X)
then	B **says** X

Rule 3: Chaining of delegation.

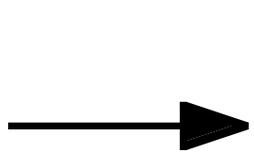
If	A **speaks for** B
and	B **speaks for** C
then	A **speaks for** C

```
procedure RC4_GENERATE ()
    i ← (i + 1) modulo 256
    j ← (j + S[i]) modulo 256
    SWAP (S[i], S[j])
    t ← (S[i] + S[j]) modulo 256
    k ← S[t]
    return k
```

```
procedure RC4_INIT (seed)
    for i from 0 to 255 do
        S[i] ← i
        K[i] ← seed[i]
    j ← 0
    for i from 0 to 255 do
        j ← (j + S[i] + K[i]) modulo 256
        SWAP(S[i], S[j])
    i ← j ← 0
```

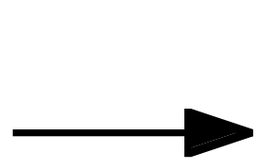
input

i_0	i_4	i_8	i_{12}
i_1	i_5	i_9	i_{13}
i_2	i_6	i_{10}	i_{14}
i_3	i_7	i_{11}	i_{15}



state

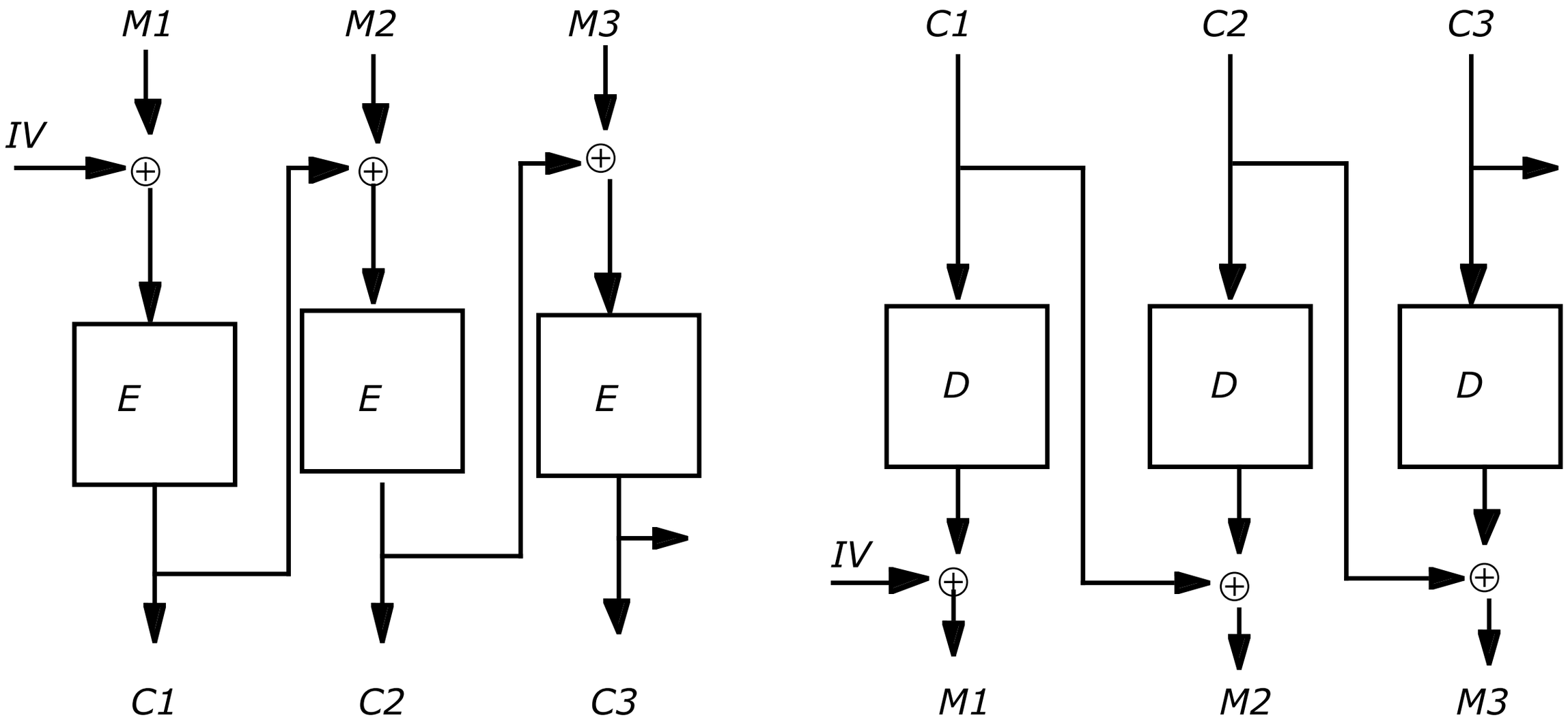
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1.2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2.3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$



output

o_0	o_4	o_8	o_{12}
o_1	o_5	o_9	o_{13}
o_2	o_6	o_{10}	o_{14}
o_3	o_7	o_{11}	o_{15}

```
procedure AES (in, out, key)
    state ← in                              // copy in into state
    AddRoundKey (state, key)                // mix key into state
    for r from 1 to 9 do
        SubBytes (state)                    // substitute some bytes in state
        ShiftRows (state)                   // shift rows of state cyclically
        MixColumns (state)                  // mix the columns up
        AddRoundKey (state, key[r×4, (r+1)×4 – 1])    // expand key, mix in
SubBytes (state)
ShiftRows (state)
AddRoundKey (state, key[10×4, 11×4 – 1])
out ← state                                 // copy state into out
```



(a) Encipher

(b) Decipher

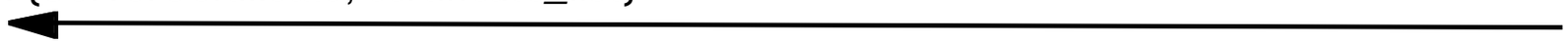
Client

Service

1. {ClientHello, client_version, randomclient, session_id, cipher_suites, compression_f}

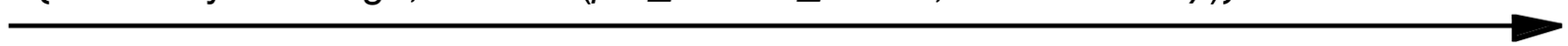
2. {ServerHello, server_version, randomserver, session_id, cipher_suite, compression_f}

3. {ServerCertificate, *certificate_list*}



4. {ServerHelloDone}

5. {ClientKeyExchange, ENCRYPT (*pre_master_secret*, *ServerPubKey*)}



6. {ChangeCipherSpec, cipher_suite}

7. {Finished, MAC (*master_secret*, messages 1, 2, 3, 4, 5)}$_{\text{client\_write\_MAC\_secret}}^{\text{client\_write\_key}}$

8. {ChangeCipherSpec, cipher_suite}

9. {Finished, mac (master_secret, messages 1, 2, 3, 4, 5, 7)}$_{\text{server\_write\_MAC\_secret}}^{\text{server\_write\_key}}$

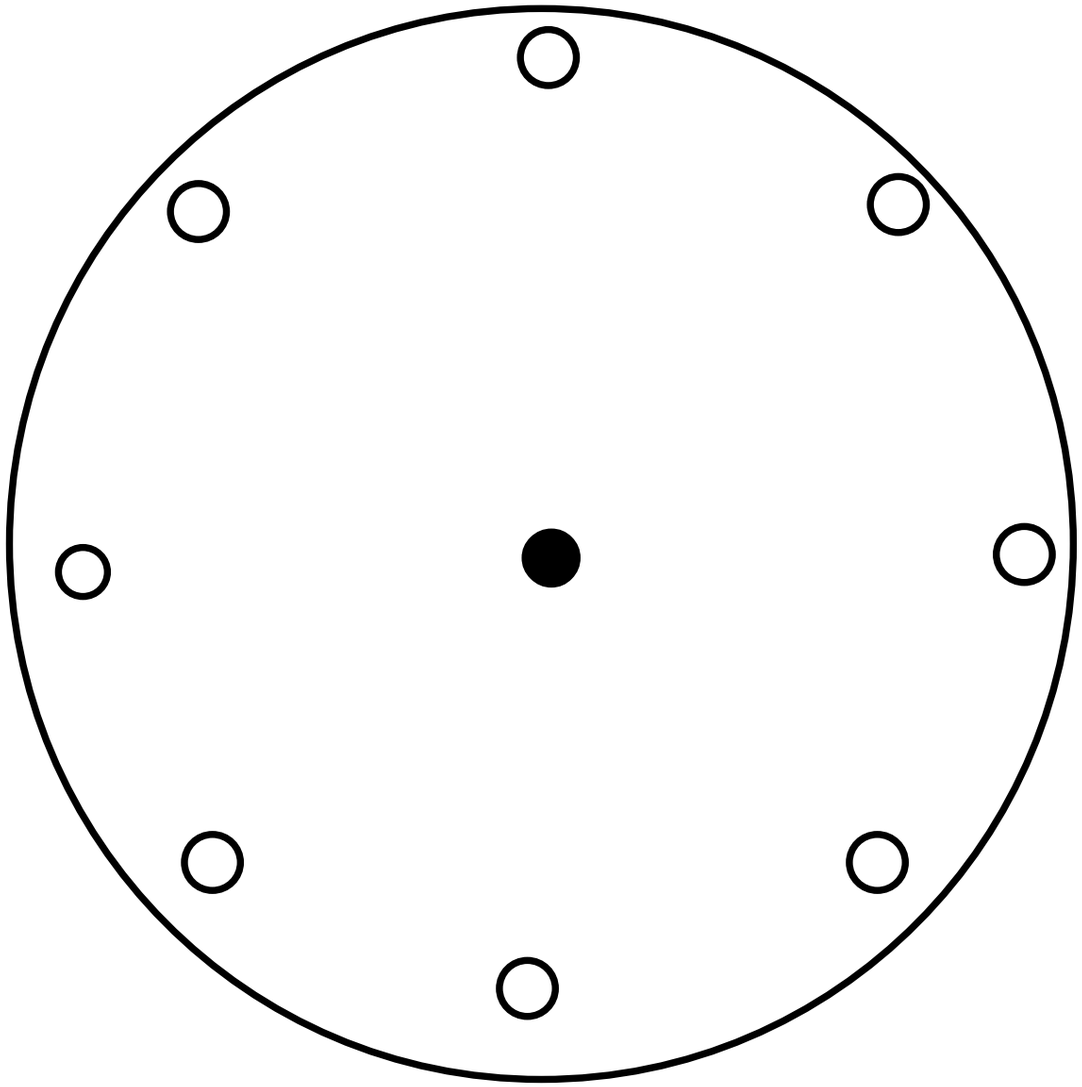
10. {Data, plaintext}$_{\text{client\_write\_MAC\_secret}}^{\text{client\_write\_key}}$

structure X_509_v3_certificate

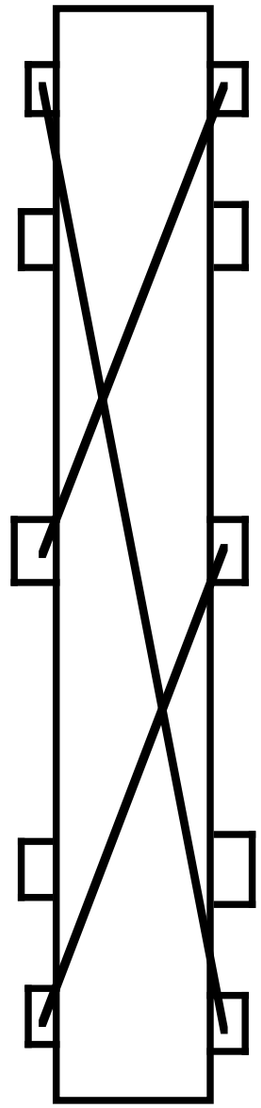
version
serial_number
signature_cipher_identifier
issuer_signature
issuer_name
subject_name
subject_public_key_cipher_identifier
subject_public_key
validity_period

```
procedure DELETE_FILE (file_name)
    auth ← CHECK_DELETE_PERMISSION (file_name, this_user_id)
    if auth = PERMITTED
        then DESTROY (file_name)
        else signal ("You do not have permission to delete file_name")
```

Enigma Rotor with eight contacts



Side view, showing contacts.



Edge view, showing some connections.