

Accountability

Daniel Bosk

Department of Information and Communication Systems,
Mid Sweden University, SE-851 70 Sundsvall.

1st May 2017

1 Book-Keeping

- Double-Entry Book-Keeping
- Separation of Duties
- Clark-Wilson Security Policy Model

2 Logging

- Securing Logging Mechanisms
- Schneier-Kelsey Logs

1 Book-Keeping

- Double-Entry Book-Keeping
- Separation of Duties
- Clark-Wilson Security Policy Model

2 Logging

- Securing Logging Mechanisms
- Schneier-Kelsey Logs

- The banks are one of the oldest institutions with a need for strict accountability.
- The main tools developed for this purpose is double-entry book-keeping.

Definition (Double-entry book-keeping)

- Add one entry of x and one of $-x$.
- Invariant of zero ($x + (-x) = 0$).

Example

- All books should be balanced.
- A transfer from one account to another must be a credit in one account and a debit in the other.
- I.e. when adding them up they equal zero.



- This principle of keeping a balance of constant zero can be transferred to other principles.
- E.g. for each log-in there should be a log-out.
- If the difference of number of log-ins L_i for a user and the number of log-outs L_o is zero ($L_i - L_o = 0$), then the user is not currently logged-in.
- Hence, the user shouldn't be able to post a comment when the system is in this state.

Note

- Note that you shouldn't use the book-keeping system to keep track of whether a user is logged-in or not.
- You can use more efficient mechanisms for that.
- But the account should be kept for future reference, in case something bad happens, then you should be able to see what really happened.

Definition (Separation of duties)

- Two or more entities must collude to break the policy.
- Two classes: *dual control* and *functional separation*.

Example (Dual control)

- Two or more staff members must act together to authorize a transaction.

Example (Dual control on film)

- Two guys in a nuclear weapons silo.
- Two keys too far from each other for one to turn simultaneously.
- Both staffers must agree to turn the keys.

Example (Dual control)

- Two or more staff members must act together to authorize a transaction.

Example (Dual control on film)

- Two guys in a nuclear weapons silo.
- Two keys too far from each other for one to turn simultaneously.
- Both staffers must agree to turn the keys.

Example (Functional separation)

- Two or more staff members must act on the transaction at different points in the transaction path.

Example (Functional separation)

- Developer team writes the code.
- System administrators deploy it.
- Auditors verifies security.

Example (Functional separation)

- Two or more staff members must act on the transaction at different points in the transaction path.

Example (Functional separation)

- Developer team writes the code.
- System administrators deploy it.
- Auditors verifies security.

- The Clark-Wilson Security Policy Model is a model for securely implementing a security policy.
- It ensures *internal consistency*, i.e. properties of the internal state of the system.
- It also allows for *external consistency*, i.e. the relation of the internal state of the system to the real world. This must however be enforced by e.g. auditing.

- Mechanisms for enforcing integrity of the system are:
 - Well-formed transactions
 - Separation of duties

Definition (Well-formed transactions)

- A limited set of functions can manipulate an object.
- Users have access to these functions, not the objects.

- Mechanisms for enforcing integrity of the system are:
 - Well-formed transactions
 - Separation of duties

Definition (Well-formed transactions)

- A limited set of functions can manipulate an object.
- Users have access to these functions, not the objects.

Requirements

- 1 Subjects have to be identified and authenticated.
- 2 Objects can be manipulated only by a restricted set of functions.
- 3 Subjects can execute only a restricted set of functions.
- 4 A proper audit log must be maintained.
- 5 The system has to be certified to work properly.

Definition (Unconstrained data item, UDI)

- Input from outside the system.
- From outside the control of the system.
- It can be anything!

Definition (Constrained data item, CDI)

- Objects (data) inside the system.
- This is under the system's control.
- This is well-formed.

Definition (Unconstrained data item, UDI)

- Input from outside the system.
- From outside the control of the system.
- It can be anything!

Definition (Constrained data item, CDI)

- Objects (data) inside the system.
- This is under the system's control.
- This is well-formed.

Note

- UDIs must be converted to CDIs.
- This is a critical part of the system.

Definition (Transformation procedure, TP)

- Procedure which manipulates CDIs.
- Can take UDI as input, must convert to CDI.

Definition (Integrity verification procedure, IVP)

- Checks the integrity of a CDI.

Definition (Transformation procedure, TP)

- Procedure which manipulates CDIs.
- Can take UDI as input, must convert to CDI.

Definition (Integrity verification procedure, IVP)

- Checks the integrity of a CDI.

Certification rules

Should be checked so that the policy is consistent:

- CR1 IVPs must ensure integrity of CDIs when IVPs are run.
- CR2 TPs must be certified to be valid; valid CDIs transform into valid CDIs; each TP can access restricted set of CDIs.
- CR3 Access rules must satisfy separation-of-duties requirements.
- CR4 All TPs must write to an append-only log.
- CR5 Any TP handling UDI must convert it to a CDI or reject it.

Enforcement rules

Describes the mechanisms needed in the system:

- ER1 Must maintain and protect list of CDIs each TP can access.
- ER2 Must maintain and protect list of TPs each subject can access.
- ER3 The system must authenticate each subject requesting to execute a TP.
- ER4 Only a subject that may certify an access rule for a TP may modify the respective entry in the list. This subject must not be allowed to execute this TP.

- 1 Book-Keeping
 - Double-Entry Book-Keeping
 - Separation of Duties
 - Clark-Wilson Security Policy Model
- 2 Logging
 - Securing Logging Mechanisms
 - Schneier-Kelsey Logs

- Have a process write log messages to a file.
- Then the running process must access the file.
 - Could be done using append only access, thus no reading or rewriting.
 - Could trust the process to do a `setuid(2)` system call.
 - This saves us from trusting the user – but only if the user doesn't have access to the hardware.
- We could also log to this or another system via `syslog(3)`, this helps us if we don't trust the user or the process.
- However, the problem remains with the `sysadmin` who has superuser access to the system.

- The sysadmin problem can be solved using a clever setup of separation of duty.
- E.g. the logs of sysadmin *A* will be stored under the control of sysadmins *B* and *C*.
- This way sysadmin *A* can do everything except modify his own logging mechanisms.
- The downside of this is that all systems must be online for this to work.

- The Schneier-Kelsey logging scheme provides a secure logging mechanism for storing logs in an untrusted machine.
- The untrusted machine \mathcal{U} is expected to work correctly up to a time t when it is compromised by an attacker.
- The logging mechanism and the integrity of the logs L_1, \dots, L_{t-1} before t are provided with confidentiality and integrity.
- All logs L_t, L_{t+1}, \dots generated from this point, however, are under the influence of the attacker.

- The scheme consists of an untrusted principal \mathcal{U} and a trusted principal \mathcal{T} .

Schneier-Kelsey Logs

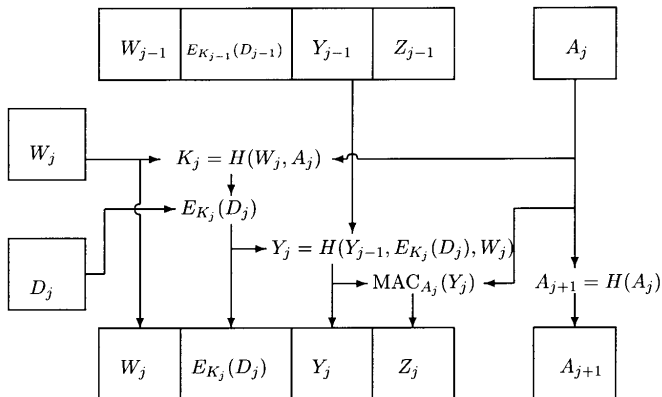


Figure: Overview of Schneier-Kelsey secure-log scheme; where W_j is the type of entry, D_j is entry data, K_j is entry key, A_j is authentication key, and H is a one-way function. Image: [SK99].

- One interesting property is that validation of logs can be delegated to a third party verifier \mathcal{V} .

- [SK99] Bruce Schneier and John Kelsey. 'Secure audit logs to support computer forensics'. In: *ACM Transactions on Information and System Security (TISSEC) 2.2 (1999)*, pp. 159–176.