On the Definition and Policies of Confidentiality

Johs Hansen Hammer
NAV, Oslo, Norway
johs.hammer@nav.no

Gerardo Schneider
Dept. of Informatics, University of Oslo, Norway
gerardo@ifi.uio.no

Abstract

In this paper we propose a more general definition of confidentiality, as an aspect of information security including information flow control. We discuss central aspects of confidentiality and their relation with norms and policies, and we introduce a language, with a deontic flavor, to express such norms and policies. Our language may be regarded as a first step towards a formal specification of security policies for confidentiality. We provide a number of examples of useful norms on confidentiality, and we discuss confidentiality policies from real scenarios.
Keywords: *confidentiality, norms, policies.*

1 Introduction

Confidentiality is defined in ISO/IEC-17799 [8] as "ensuring that information is accessible only to those authorized to have access". The references to this definition have been rather frequent, though it has been recognized long ago, even before the standardization, that the definition is incomplete and does not capture all the aspects of confidentiality. Indeed, some early papers have defined confidentiality not only including "those who have authorized access", but also some kind of restriction to information flow (see for instance discussions presented in [2] and references therein).

The above definition does not comprise information flow control as an aspect, which we regard as a severe weakness. Moreover, it only focuses on what is authorized, without mentioning normative aspects as entitlement and permission. Shifting focus from what is authorized to what is entitled, both with respect to access and information flow, raises the ambition dramatically for ensuring confidentiality.

In order to understand the above we need to give a more precise definition of *authorization*, *entitlement*, and *permission*. These concepts are often used as synonyms in the daily language, but they are in fact quite different concepts from a normative and security point of view. Saying that somebody is *authorized* to do something means that somebody else, with the power to do so, has decided to grant this individual the ability to do such a thing. In this paper in particular, to authorize shall be interpreted as a decision enabling access. It is assumed that there exists an *authorization system* which produces such decisions. It is also assumed that there exist *access control systems* controlling access requests according to the registered decisions. *Entitled* accessibility shall here be understood as complying with certain given policies, constraining information flow and access to information. In general, entitlement should be judged relative to some existing normative system. *Permission*, on the other hand, is the right stipulated in a normative system of an individual (organization, software, etc), or a group of individuals, to perform some action (including accessibility).

Let us consider the following example, to show how the above three concepts are intertwined in a confidentiality policy. If a subject is authorized to access a source containing information stipulated to be confidential by a confidentiality norm, then the subject is not entitled to access such information unless it is permitted by the normative system. If the subject access such information, there is a violation of confidentiality, which might be caused by different reasons: (i) the information should not have been stored in the source, (ii) the authorization was inappropriate since the source was susceptible of containing confidential information.

As motivated by the example presented above, if a person has authorized access to a copy that should not have existed, or should have not been stored where it is, this may be regarded as a violation of confidentiality. That is, misplacing information is also a problem, not only insecure storage. Similarly, confidentiality should also take into account the use of channels since one may utilize inappropriate means for sending information. Approved storing (determining which objects must be stored, conditions of the environment, etc), sending (channels, messages, etc.), deletion, and destroying are also aspects of confidentiality. While it is the unentitled reading we want to avoid, controlling confidentiality may also involve controlling writing. It is controlling the proliferation of information that is the goal

of confidentiality. It is not to control that the integrity of the information is preserved. This is a separate goal of information security, not covered in this paper.

Authorizations are often given to *types* of information, not to specific information. Actual accessibility to information may not only be the consequence of authorization decisions but of other kinds of decisions, e.g. the decision to send information to a file catalogue where a person is given authorization. Concerning information transmission, it is important to inspect whether the sending complies with the confidentiality policy at the sending part, not only to inspect whether the receiver's authorization comply with the confidentiality policy at the receiving part, although this is also a matter. Thus, confidentiality compliance should, neither for the manual nor for the automatized part, be demonstrated only against decisions about (local) accesses. It should be compared with the confidentiality policy of the organization, and in more general scenarios against inter-organizational policies.

Aligned with these motivations, safeguarding confidentiality should include aspects such as access control, controlling flow of information (copying and moving information, controlling external and internal exchange of information), storing, destroying and deleting information, and surveillance (using logs, alarms and warnings to detect and follow up events and incidents). There are certainly many other challenges arising from the attempt to define a wide and precise definition of confidentiality from the operational point of view. One example is on which level of detail access to information should be given, another is what precision to grant access with, and a third how to judge the sensitivity of information compiled from various sources. It is not the intent to discuss every aspect of confidentiality here, but rather to focus on a number of general aspects.

Though we are interested both in organizational and software confidentiality aspects, we will focus in this paper on the former. We claim that any aspect of confidentiality should be found in the organization's norms, agreements, declarations, authorizations, habits and practices, all together constituting a *confidentiality policy*.

Briefly, any security system aiming at preserving aspects related to confidentiality should: (1) ensure that accessibility to information is entitled, not only authorized; (2) focus on confidentiality norms in general and not only authorizations *qua* decisions; (3) comprise both access *and* flow control, including the control of where the information can be stored and of deletion of information; (4) ensure that the use of the information which is accessible is entitled. We argue that the notion of confidentiality should be generalized to include the above-mentioned aspects.

The contributions of this paper are a proposal of a more general definition of confidentiality and a discussion of certain aspects related to confidentiality, hoping to shed some light on the correct use of relevant terminology. We also present a language to express confidentiality policies.

In this paper we only give the syntax of the language, and we use it to clarify the notion of confidentiality and to illustrate how to describe useful policies taken from real-life scenarios. Semantics is left for future work.

The paper is organized as follows. In next section we provide a new definition of confidentiality, and we briefly discuss its impact and difference with the ISO definition. In section 3 we give a language for writing confidentiality norms, and we illustrate its use with few examples, also used to expose important aspects of confidentiality. In section 4 we give some real scenarios from the Norwegian Public Administration Act, showing how our language can be used to represent real confidentiality policies. In the last two sections we list some related work and we conclude.

2 On a New Definition of Confidentiality

The main message from the previous section is the imperative need to redefine the notion of confidentiality, and of a new language for writing normative statements for guaranteeing safeguarding of information confidentiality. We address the former in this section.

We propose the following definition of confidentiality:

Definition 1 Confidentiality *is the concept of: 1) ensuring that information is accessible for reading, listening, recording or physical removal only to subjects entitled to it, and 2) that subjects only read or listens to the information to the extent permitted.*

A *subject* may be a person, a process or an organization. By definition, the *entitlement* should be derivable from the confidentiality policy. Furthermore, as we discussed in the introduction, from any explicit authorization it does not follow that one has permission to read all the information one has access to. The condition about authorization in the ISO definition is thus regarded neither as a necessary nor a sufficient condition for confidentiality. The following example illustrates this.

Example 1 Assume that a person s_1 is authorized to access an information storage where information i is stored, and that s_1 at a certain moment shares this information with s_2 storing i in a shared file catalogue where both s_1 and s_2 are authorized to access. Since s_2 is authorized to access the file catalogue, no violation of confidentiality may be observed due to a dogmatic application of the ISO definition, even if s_1 is strictly forbidden to share i with s_2. This should clearly be regarded as a breach of confidentiality, showing the weakness of the ISO definition. The problem above is avoided with Def. 1 since s_1 is not entitled to perform the information transfer of i to the shared file catalogue. □

$$
\begin{array}{rcl}
Policies & ::= & Policy \mid Policy \wedge Policies \\
Policy & ::= & Domain \cdot Modal\ Action \mid \\
 & & Domain \cdot \textbf{if}\ Cond\ \textbf{then}\ Modal\ Action \\
Modal & ::= & Modality \mid \textbf{not}\ Modality \\
Modality & ::= & \textbf{obligatory} \mid \textbf{permitted} \mid \textbf{prohibited} \\
Action & ::= & Subject\ .\ Activity \mid \textbf{not}\ Action \\
Activity & ::= & get(i)from(k) \mid delete(i)from(k) \mid \\
 & & send(i)from(k)to(d)through(c) \mid \\
 & & give(i)to(d) \mid put(i)in(k) \mid \\
 & & Activity\ ;\ Activity
\end{array}
$$

Figure 1. Syntax of the Policy Language

3 A Notation to Illustrate Aspects, Norms and Violations of Confidentiality

We introduce now a language which will be used to illustrate aspects of confidentiality, and to express confidentiality norms, facilitating the detection of violations of confidentiality. The language must be seen as a first step towards a formal language for expressing confidentiality policies. Actual implementation of the language would demand both the specification and development of a semantics. The language is meant to have expressive power comparable to first order predicate logic, while including also a kind of *deontic modalities*.[1]

3.1 A Confidentiality Policy Language

Obligatory and qualificational norms Norms may be divided into *obligatory* and *qualificational* norms (see for instance [14]). Obligatory norms may be characterized by four elements: (i) a condition for the norm to be applicable, (ii) subjects addressed by the norm, (iii) an action, and (iv) a deontic modality telling if the action is permitted, prohibited or obliged. Qualificational norms, on the other hand, merely state that some phenomenon is qualified as something else. Norms of competence are regarded as a certain type of qualificational norms, where power plays a central role as normative modality. An authorization is also a qualifying activity, since it qualifies certain activities as accessible for a subject, not necessarily making them permitted.

The activities introduced in our language are information processing activities, so authorizations are restricted to those kind of activities. We do not treat here how power is managed (e.g qualifying power to give and withdraw power to authorize, power to establish and remove rules, like access rules, rules for firewalls, and others), though they are important from a normative and security perspective.

[1] We use the term here informally since we are not giving a logical semantics at this stage.

The Language We present in Fig. 1 the syntax of our policy language, where we assume we have the following *sort declarations*: $information(i)$, declares that i is of type information; $source(k)$ and $destination(d)$ says that k and d can act as the source and destination of a given information flow, respectively; $subject(s)$ and $object(o)$ declare the sender (or actor) and the receiver (or receptor) of a given action, respectively; finally, $channel(c)$ declares c to be the means through which information may flow. A source (and a destination) may be a subject, object or a channel. A subject may either be a person, a process, or an organizational unit. We will use the above lowercase letters, and sub-indexed versions, as representatives of the corresponding sorts.

A set of *policies* is then defined as consisting of one or more policies (being *Policy* the syntactic category). These comprise *unconditional* and *conditional* policies. Each policy takes the form of a *modality* (including its negation) "applied" to an *action*, under a certain *domain*. Domains are underspecified here, but are supposed to range over the subset of natural numbers or any enumerative type denoting in which context the policy is applied. For instance, the set of domains may include: government, local community, university, and so on. Conditions are also underspecified; we will see later some examples of their use.

The syntactic category *modality* includes the usual deontic normative notions of obligation, permission and prohibition. An *action* is defined to consist of a subject performing an *activity*. For sake of space, we do not list all the activities in Fig. 1, and we only consider here a few to illustrate our language. The syntax is self-explanatory, but for completeness we only explain $s.send(i)from(k)to(d)through(c)$. It stands for s sends information i from source k to destination d through c. It is possible to define new actions from the primitive ones. For instance, $s.copy(i)from(k1)to(d1)$ is a shortcut for $s.get(i)from(k1); s.give(i)to(d1)$, and $s.move(i)from(k1)to(d1)$ is a shortcut for $s.copy(i)from(k1)to(d1); s.delete(i)from(k1)$, where ";" glue activities to form sequences of activities.

Moreover, we assume we have the following types of information:

$$types\_of\_inf ::= public(i) \mid internal(i) \mid classified(i) \mid declassified(i, l)$$

where $public(i)$ means that information i is public, $internal(i)$ that i is internal to an organization, $classified(i, l)$ that i is classified to level l with respect to national security, and $declassified(i, l)$ that information i is declassified at level l.

Besides the policies as described above, we consider that *authorizations* are described as follows:

$$
\begin{array}{rcl}
Authorizations & ::= & Auth \mid Auth \wedge Authorizations \\
Auth & ::= & Domain \cdot Ok\ Action \mid \\
 & & Domain \cdot \textbf{if}\ Cond\ \textbf{then}\ Ok\ Action \\
Ok & ::= & Subject\ .\ \textbf{OK} \mid \textbf{not}\ Subject\ .\ \textbf{OK}
\end{array}
$$

where $Action$, and $Activity$, are as in Fig. 1. Notice that the deontic notions do not need the specification of who is granting the obligation, permission or prohibition; it is not the case for **OK** : by definition of authorization, a subject is always responsible for the authorization.

3.2 Examples

The following examples are presented to show the use of our notation, and also as a mean to discuss important aspects of confidentiality.

Example 1 Assume the following new predicates: $employee(s, b)$ means that s is an employee within organization b. Let $employee(s, b)$, $employee(s_2, b)$, and **not** $employee(s_1, b)$, then

$$\textbf{if}\ public(i)\ \textbf{then permitted}\ (s.give(i)to(s_1))$$

expresses that every employee may convey open information to externals. Similarly,

$$\textbf{if}\ internal(i)\ \textbf{then permitted}\ (s.give(i)to(s_2))$$

expresses that every employee may convey internal information to employees. If $cleared\_for\_level(l, s)$ means that subject s is cleared for level l, then

$$\textbf{if}\ classified(i, l) \wedge cleared\_for\_level(l_1, s_3) \wedge l_1 < l\ \textbf{then prohibited}\ (s_1.give(i)to(s_3))$$

expresses that it is prohibited to convey information to a person with a lower clearance than the classification of the information. These are widespread restrictions for information flow. The conditions are quite general and may express requirements and restrictions on, for instance, storage and security at the receiving side of the channel.

Accordingly, an obligation to delete certain information may be expressed as:

$$\textbf{if}\ < condition > \textbf{then obligatory}\ (s.delete(i)from(o)).$$

Example 2 The following example shows a case of access without permission but with authorization inside a given organization. Assume that an organization has a confidentiality policy stating that

$$\textbf{not permitted}\ (s_1.get(i)from(k)).$$

Let us assume now that s authorizes s_1 to access i, i.e.

$$s.\textbf{OK}\ (s_1.get(i)from(k)).$$

The above shows how a given confidentiality policy can enter into conflict with certain authorizations, giving place to a violation of confidentiality. The above case where s_1 is not permitted but is authorized access to a certain information arises when the access rules do not reflect, and are not in accordance with, the confidentiality policy.

Example 3 Let us assume now that a certain organization b has a norm stating that

$$\textbf{if}\ employee(s_1, b) \wedge \textbf{not}\ employee(s, b)\ \textbf{then not permitted}\ (s.get(i)from(s_1)).$$

Assume that s is authorized to get the information i by his organization b_1, that is b_1.**OK** $(s.get(i)from(b))$. This is properly allowed by the ISO definition of confidentiality, since it only considers authorization as a condition to get access. The key is, however, to determine whether s is *entitled* to get such information, and whether s_1 is permitted to deliver it. This is not a criterion at all using the ISO definition, it is however properly handled by Def. 1. Authorized, but not permitted, access to information implies a violation of confidentiality.

Example 4 Sometimes a policy simply stating that something is permitted is not enough. Assume the following unconditional policy: **permitted** $(s_1.give(i)to(d))$.

If d is an external destination where information is compromised, a violation occurs in d's organization. But, does the delivery of information to d imply a violation of confidentiality in s_1's organization, in the presence of insufficient security in d? It depends on the confidentiality policy! It should not be considered a violation by s_1 if there is a policy as above, and there is no explicit assumption about d. In some cases, however, a delivery is not permitted because the security at the receiving party is insufficient. That is, whether the above policy will trigger a violation of confidentiality or not, depends on what conditions about security at the receiving party there exist as criteria for permitting the sending of information. Conditions about security, processing, and other matters at the receiving part, affect confidentiality and should be carefully stated in the security policies of the organization.

4 Real Scenarios

In this section we present a norm taken from a real normative system, to show the difficulty in unambiguously interpreting policies written in natural languages. We also illustrate with the examples how a more formal language helps to refine a norm, and to detect possible flaws in the normative system.

Scenario 1 The following is the main secrecy norm of the Norwegian Public Administration Act.

> Everybody working or performing services for a public administration unit, is obliged to prevent others getting access or knowledge to what he in the service or work get knowledge about "§13-information".

In order to formalize it in our language, we introduce the following predicates: $organization(b)$, $employee(s, b)$, $consultant(s, b)$, $i \in I\S13$ (i is of type $I\S13$). We assume we also have in our language the following activity: $s.protect(i)from(s_2)$, meaning that the subject s prevents s_2 to have access to i. The norm takes the form:

if $(employee(s, b) \vee consultant(s, b)) \wedge i \in I\S13 \wedge$
not $employee(s_2, b)$
then obligatory $(s.protect(i)from(s_2))$.

Though the above seems to correctly formalize the norm, it does not make explicit the concrete operational ways s_2 has of getting to know i. In other words, the activity $protect(i)from(s_2)$ is too high-level. We propose to refine the above to the following:

if $(employee(s, b) \vee consultant(s, b)) \wedge i \in I\S13 \wedge$
not $employee(s_2, b) \wedge s.get(i)from(b)$
then prohibited $(s.give(i)to(s_2))$.

The above is not, however, a complete formalization of the norm since we should also avoid the indirect flow of information. We need thus to add the following:

if $(employee(s, b) \vee consultant(s, b)) \wedge i \in I\S13 \wedge$
not $employee(s_2, b) \wedge s.get(i)from(b) \wedge$
$s.send(i)from(b)to(d)through(c)$
then obligatory $(s.protect(d)from(s_2) \wedge$
$s.protect(c)from(s_2))$.

Actually, the norm also restricts internal information flow (within b), which we have not expressed in the above specification.

Scenario 2 The norm "§13-information" referred in the above example allows in fact the supposed secret information to flow. In §13b-3 of the above norm, it is stated that:

> Secrecy according to §13 does not forbid [...] that the information is accessible for other employees within the unit or the governmental agency the unit belongs to, to the extent needed for an appropriate arrangement of work and archive [...].

Notice that the norm stipulate an exception to the secrecy, in case the work (or archive) arrangement is *appropriate*. If a work arrangement is judged to be appropriate, then there is an opening to legally let information flow. The main problem here is, what does it mean to be "appropriate"? There is no strict definition in the normative system of "appropriateness", letting some space for leaking information without infringing the law. To avoid this, the normative system needs to provide a precise definition of "appropriate work and archive arrangement". We do interpret the norm as a qualificational norm, giving competency to the body deciding upon the question of appropriateness to decide upon the "embedded" flow and access rules. In effect, this is a competency to establish flow and access rules.

Despite the complexity of the norm we can represent it in our language, with the addition of few extra predicates. Due to space limitation we do not present the formalization here. This example clearly exposes the imperative need of a formal definition of all the terminology used in real normative systems.

5 Related Work

There are many other definitions of confidentiality in different important national organizations. For instance, NCSC[2] defines confidentiality as " the concept of holding sensitive data in confidence, limited to an appropriate set of individuals or organizations" [11]. This definition is stronger than the ISO definition, describing confidentiality as a restriction to flow of data/information, even though the definition is somehow circular, and uses the term *data* instead of *information*.

The language presented in section 3 is based on an experimental tool to express legal norms [5, 7], a more detailed account of which only exists in an unpublished document [6] discussing a broad scope of questions like the feasibility of formalizing legal norms, and of deducing normative positions from certain actions, events and states within a normative system. It also discusses the role of general formalizations of deontic operators in normative systems and other applications. There, also behavior controlling aspects of norms is emphasized, and a form of "adherence semantics" for norms is discussed.

Despite the fact that confidentiality is not mentioned as such in [9], McLean has already identified that security should include access control and information flow control.

The idea of using deontic logic as a formal framework for security policies in general is not new [4]. In particular, deontic logic has been used in combination with epistemic logic for formalizing confidentiality policies [2, 3]. In this context, confidentiality is defined as "If A knows φ then A should have the permission to know φ". Cuppens and colleagues' work is based on logics and is thus amenable to formal reasoning, with application in software security (computer systems). Though their definition must be understood as an abstraction to be properly tractable in a logical

[2]National Computer Security Center, USA.

system, we believe that from a general (organizational) perspective confidentiality should not be defined using knowledge as a key concept. We argue that confidentiality should be defined in terms of access and flow control. Moreover, their definition is not applicable in organizational systems (i.e., not computerized) where determining what is entitled may involve a legal decision process.

In this paper we deal only with the specification of confidentiality policies and we have not discussed how to enforce or to monitor such policies. Other works dealing with the formal definition and enforcement of particular aspects of security, e.g. information flow, can be found in the quite complete survey [13] and references therein.

See [10] for a clear presentation of Standard Deontic Logic, its paradoxes, and the usual problems arising when formalizing deontic notions.

It is worth noticing the report [1] of the Ambient Networks project, which includes an interesting roadmap of research directions on policy management in general.

6 Conclusion and Further Work

In this paper we have argued for a more general definition of confidentiality, and we have presented a language for expressing confidentiality policies. This is only a first step towards a formal definition of such policies, but we believe our language is a good starting point. Though the main focus of the language presented in [12] is the formalization of electronic contracts, it could be possible to use it as a basis for giving semantics to our language.

While formalizing confidentiality policies for a given organization is certainly crucial, the main challenges concern the definition of policies acting at different levels of abstraction, and in inter-organizational activities. This comprises the detection of contradictory policies, resolution of priorities, and refinement of policies from high to more concrete levels. The conformance of inter-business contracts w.r.t. existing relevant policies is also a very interesting research direction. We expect to be able to address some of the above issues whenever the language is provided with a logical semantics.

We have not dealt with management of power, including delegation, though actions for such activities might be represented in our language. For instance $s.givepower(p)to(s_1)within(sc)$ would represent that s gives power p to subject s_1 within scenario (domain) sc; and $s.giverule(r)for(s_1)within(sc)$ would represent that s *enacts* rule r for subject s_1 within scenario sc. Similarly for withdrawal of power. One could also think of a more abstract definition of qualificational activity, using an expression like $qualify(x)as(y)$, with x as an action and y as the authorization operator **OK** . The expression should represent the fact that action x is authorized. Similarly, where x is a policy expression, and y is the policy type. The effect will be that the policy expression is taken as a policy element. Surely, there should also be an abstract activity, $disqualify(x)as(y)$, which brings the qualification of x to y to an end. We are currently working on extending our language with such meta-language constructors.

Whether the norms are manually or automatically controlled, vague or even implicit norms may be a poor source for good behavior. Clearly stated norms are, therefore, welcomed. Systems, with the capability to control information flow according to flow control authorizations and norms, would be a great achievement.

References

[1] Ambient Networks Phase 2. D10-d.1 integrated design for context, network and policy management, December 2006.

[2] P. Bieber and F. Cuppens. *Expression of confidentiality policies with deontic logic*, pages 103–123. Deontic Logic in Computer Science: Normative System Specification. John Wiley & Sons, 1993.

[3] F. Cuppens and R. Demolombe. A modal logical framework for security policies. In *ISMIS*, volume 1325 of *LNCS*, pages 579–589. Springer, 1997.

[4] F. Cuppens and C. Saurel. Specifying a security policy: a case study. In *CSFW*, pages 123–134. IEEE Computer Society, 1996.

[5] J. Hansen. Simulation and automation of legal decisions. CompLex 6/86, Norwegian University Press, 1986.

[6] J. Hansen. Formalisering av rettslige normer (Formalizing legal norms). In Norwegian, 1990.

[7] J. Hansen and E. Ødegaard. SMARN –A language for modelling legal norms, with an example of modelling parts of the Norwegian Inheritance Act. CompLex 8/85, Norwegian University Press, 1985.

[8] ISO/IEC 17799. Information technology – Security techniques – Code of practice for information security management. `http://www.iso.org/iso/en/prods-services/popstds/informationsecurity.html`, 2000. Rev. 2005.

[9] J. McLean. Security models and information flow. In *IEEE Symposium on Security and Privacy*, pages 180–189, 1990.

[10] P. McNamara. Deontic logic. In D. M. Gabbay and J. Woods, editors, *Handbook of the History of Logic*, volume 7, pages 197–289. North-Holland Publishing, 2006.

[11] NCSC-TG-004. [Aqua Book] Glossary of Computer Security Terms. `http://www.marcorsyscom.usmc.mil/sites/ia/references/national/NCSC-TG-004%20Glossary.html`, 1988.

[12] C. Prisacariu and G. Schneider. A formal language for electronic contracts. In *FMOODS'07*, volume 4468 of *LNCS*, pages 174–189. Springer, 2007.

[13] A. Sabelfeld and A. C. Myers. Language-based information-flow security. *IEEE Journal on Selected Areas in Communications, special issue on Formal Methods for Security*, 21(1):209–239, 2003.

[14] N. K. Sundby. Om normer. Dr. Juris thesis, University of Oslo, 1974. In Norwegian (in English: On Norms).