A Model for Temporal Interval Authorizations

Francis B. Afinidad, Timothy E. Levin, Cynthia E. Irvine, and Thuy D. Nguyen
Computer Science Department, Naval Postgraduate School
Monterey, CA 93943, USA
{fbafinid, levin, irvine, tdnguyen}@nps.edu

Abstract

A new model for representing temporal access control policies is introduced. In this model, temporal authorizations are represented by time attributes associated with subjects and objects, in a "time interval access graph." The time interval access graph is used to define constraints on the temporal relations between subjects, objects, and the time of access. Interval algebra is used to precisely define and analyze the time interval access graph, and to specify the evaluation of access requests.[1]

1. Introduction

In many commercial and military environments, time may be a critical factor in authorizing access to information. For example, the value of the data or the criticality of the need to access it may vary over time. Thus, future information systems will need to support system-wide security policies that incorporate time as a decision factor. To this end, a Time Interval Access Control (TIAC) model has been developed.

A significant contribution of the TIAC model is that it provides formal semantics to express temporal authorization policies, in which temporal attributes of subjects and objects are used to determine authorized accesses. In one example, this model could be used to express a policy for a prepaid phone card, where the initial time of a request must occur before the expiration time of the card (the object), and the duration of the request must fall within the amount of minutes the user (the subject) has left. In another example, the TIAC model could specify an access control policy for the use of satellite imagery where the user has an authorized period of access and the imagery data (the object) may only be viewed during a given interval.

Interval algebra [2] provides the necessary expressive power to precisely describe a desired temporal access control policy, and an efficient way to computationally reason about whether a given access request may be acceptable within the constraints of that policy. Policy enforcement mechanisms and the modeling of the effectiveness of those mechanisms with respect to the type of temporal authorizations describable in TIAC are outside of the scope of this paper (see [1]).

A brief discussion of interval algebra is presented followed by our approach to modeling time and time intervals. A detailed description of the TIAC model is presented where we establish the formal semantics used for representing temporal authorizations and access requests. Section 5 discusses current and future research efforts related to TIAC, and finally, the conclusion is presented in Section 6.

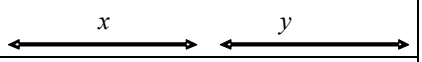
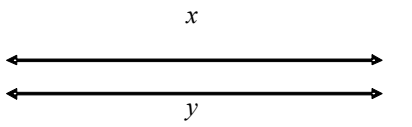
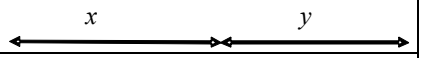
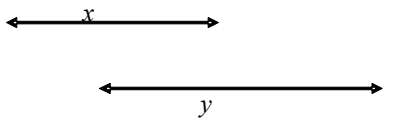
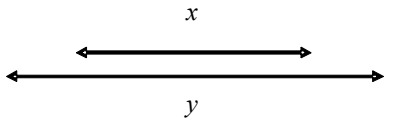
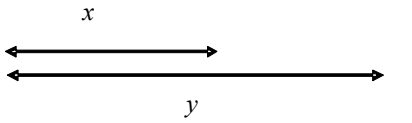
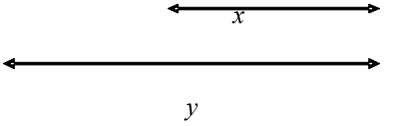
2. Background

Interval algebra [2] defines the relations that can hold between two time intervals (see Table 1). These relations are mutually exclusive, in that for any two intervals, only one relation can hold. Interval algebra assumes that the beginning and ending points (signified with "–" and "+" respectively) of an interval do not coincide. For each entry in Table 1, the first line shows the basic relation and the second line shows its inverse relation.

A set of time intervals and their required or allowed interrelationships can be represented using a directed graph - also known as an interval algebra (IA) network - in which each vertex represents an individual time interval and each directed edge represents one or more relationships between the connected vertices. For

[1] This material is based upon work supported by the National Science Foundation under Grant No. CNS-0430566 and CNS-0430598 with support from DARPA ATO. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or of DARPA ATO.

Table 1. Basic temporal interval relationships

RELATION	PREDICATE FORM	SYMBOL	RELATION ON ENDPOINTS	PICTORIAL MEANING
x before y y after x	BEFORE(x,y) AFTER(y,x)	< >	($x+ < y-$)	
x equals y y equals x	EQUALS(x,y) EQUALS(y,x)	= =	($x- = y-$) ∧ ($x+ = y+$)	
x meets y y met by x	MEETS(x,y) MET_BY(y,x)	m mi	$x+ = y-$	
x overlaps y y overlapped by x	OVERLAPS(x,y) OVERLAPPED_BY(y,x)	o oi	($x- < y-$) ∧ ($x+ > y-$) ∧ ($x+ < y+$)	
x during y y includes x	DURING(x,y) INCLUDES(y,x)	d di	($x- > y-$) ∧ ($x+ < y+$)	
x starts y y started by x	STARTS(x,y) STARTED_BY(y,x)	s si	($x- = y-$) ∧ ($x+ < y+$)	
x finishes y y finished by x	FINISHES(x,y) FINISHED_BY(y,x)	f fi	($x- > y-$) ∧ ($x+ = y+$)	

example, consider a set of time intervals {A, B, C}, where A is before B, and B is during C, then the corresponding graph could be drawn as shown in Figure 1, which will be abbreviated as: $A \text{—} (<) \rightarrow B \text{—} (\text{d}) \rightarrow C$. The notation *(B, C)* represents a directed edge connecting vertices *B and C* whereas *(B→C)* represents the relation between vertices *B and C. Thus, (B→C) = d, during(B,C)* and $B \text{—} (\text{d}) \rightarrow C$, are logically equivalent expressions.

The axioms in Appendix B define the transitive relations that result from the conjunction of any two adjacent interval relations. Each entry in the left column indicates a relation from node A to node B (A→B), each entry in the top row indicates a relations from node B to node C (B→C), and the row-column intersections show the resulting relations that are possible between A and C. Thus, the temporal relation from A to C in Figure 1 must be one of {<, o, m, d, s}, as shown in Figure 2. The derivation of transitive relations is generalized with the function R_T, which produces the union of the relations derived from two sets of adjacent interval relations, R_1 and R_2:

$R_T(R_1, R_2) =$

1. $R \leftarrow \varnothing$
2. **for each** r_1 in R_1
3. **for each** r_2 in R_2
4. $R \leftarrow R \cup T_r(r_1, r_2)$

5. **return** *R*

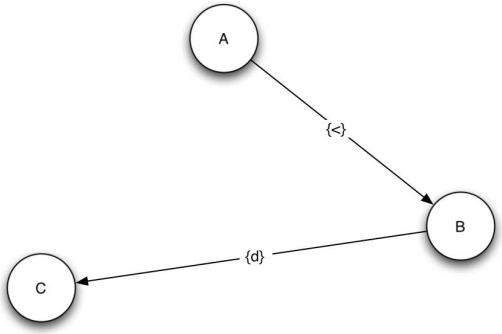


Figure 1. Initial graph representation of A before B and B during C

3. TIAC model

In this section, the TIAC model is described in detail. A discussion of time and intervals provides a foundation for the TIAC model. This discussion is followed by the elements that make up the TIAC model. These elements are: 1) Temporal Entities, 2) Temporal Authorizations, 3) Access Requests, and 4) Access Control.

3.1.Time and Intervals

In the TIAC model, time is represented by a set of points T = {t_0, t_1, t_2… } on a scale (i.e., a timeline), where T is linearly ordered with respect to the < relation such that $t_i < t_j$ iff $i < j$. The quantization of time provides the basis for reasoning about the relation between time intervals.

We wish to derive precise definitions such that it is clear that: 1) the interval between two adjacent points, t_a, t_{a+1}, is exactly one time unit, or a *unit interval*, 2) the duration of an interval consisting of two consecutive unit intervals is exactly two time units, and (3) there is no semantic ambiguity about the point where two intervals meet.

Let $\zeta(t_a, t_b)$ represent the set of the elements from T that lie between distinct points t_a and t_b, including t_a but excluding t_b. $|\zeta|$ represents the number of elements of ζ. ζ is not defined for $t_b \leq t_a$.

$$\forall\, t_a, t_b \in T\,(t_a < t_b$$
$$\zeta(t_a, t_b) = \{\, t_c \in T \mid t_a \leq t_c < t_b\}$$

Let a unit interval be associated with each $t \in \zeta(t_a, t_b)$. The number of units intervals, or the *temporal duration* of $\zeta(t_a, t_b)$ is $|\zeta(t_a, t_b)|$, to be designated as $\boldsymbol{\tau} = [t_a, t_b)$.

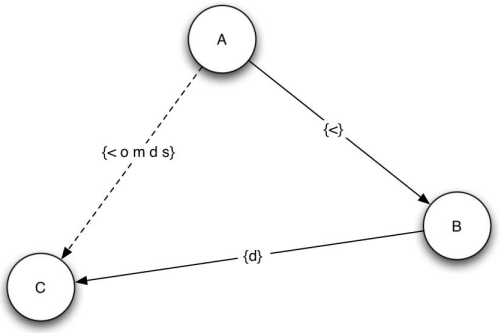


Figure 2. Derived graph showing possible relations between A and C

For example, if the interval $\tau_3 = [1, 6)$ is divided into equal halves $\tau_1 = [1, 3)$ and $\tau_2 = [3, 6)$, it is clear that point 3 belongs to τ_2, but not τ_1.

The model does not depend on or limit the granularity of intervals. In a system with multiple time sources, such as a network, mechanisms such as the *network time protocol* could be used to synchronize time. However, the granularity of time intervals would be limited by the degree of accuracy provided by the synchronization mechanism.

3.2.Temporal Entities

The model includes the concept of subjects and objects similar to those discussed by Graham et al., Lampson, and Weissman [11, 14, 17]. Subjects and objects each have an associated time interval (attribute), which is used for making authorization decisions.

In the following definitions, $\mathbf{S}_\tau = \{s_1, s_2, \ldots s_n\}$ is the set of temporal subjects, and $\mathbf{O}_\tau = \{o_1, o_2, \ldots o_n\}$ is the set of temporal objects (i.e., the passive entities that hold data or information and are accessed by temporal subjects).

Definition 1 (Temporal Object, Temporal Subject). *A temporal entity α is an object $o \in O_\tau$, or a subject $s \in S_\tau$, with which is associated a time* interval *$\tau = [t\text{-}, t+)$ where $\alpha.\tau$ designates the time interval and $\alpha.id$ indicates the identity of the entity.*

Useful temporal policy constraints may refer to a reference time such as the *current* time of day that is distinct from the attribute associated with a given subject or object, e.g., the current time must be within the subject's valid time. To reflect the notion that every activity will take at least one time unit (the minimal measurable duration), and to be consistent with the constructs of interval algebra, reference times will be expressed as intervals, e.g., based on the current time. t_{now} is a variable of the model that indicates the "current" time,

an element of T, and $now.\tau$ is the unit interval based on t_{now},:

$$now.\tau = [t_{now}, t_{now} + 1)$$

3.3. Time Interval Access Graph φ

The TIAC model introduces the time interval access graph, φ, which defines access constraints on the temporal relations between a subject, an object, and a reference time interval (τref), such as now.τ .

Definition 2 (Time Interval Access Graph φ). *The time interval access graph φ is a consistent instantiation of a three-vertex IA network $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{R}, \gamma)$ where*:

$V = \{s{:}\tau,\ o{:}\tau,\ \tau_{ref}\}$
$E = \{(s{:}\tau,\ o{:}\tau), (\tau_{ref},\ s{:}\tau), (\tau_{ref},\ o{:}\tau)\}$
$R = \{<, >, d, di, o, oi, m, mi, s, si, f, fi, =\} \cup \varnothing$
γ: $E \rightarrow \wp(R)$ = *a function that specifies the disjunctive set of the allowed relations between the vertices connected by a given edge.*

For example, φ could be instantiated as follows:

$s{:}\tau = [5, 20)$, $o{:}\tau = [10, 15)$, *and* $\tau ref = [11, 12)$
$\gamma(s{:}\tau,\ o{:}\tau) = \{includes\}$, $\gamma(\tau ref,\ s{:}\tau) = \{starts \vee during\}$, *and* $\gamma(\tau ref,\ o{:}\tau) = \{during\}$

The association of an empty set ∅ with an edge e means that there are no restrictions on the relationship between the adjoining vertices. Arbitrary relations in an IA network may be inconsistent, meaning a contradiction of relations exists.

Definition 3 (Inconsistent IA Network). *An IA network is inconsistent if there exists two adjacent edges $(u, v) \in E$ and $(v, w) \in E$ where $\gamma(u, w) \supset R_T((u \rightarrow v), (v \rightarrow w))$.*

A consistent representation of a three-node access graph can be efficiently determined [1, 2], as shown in the Appendix, by iteratively deriving transitive relations of the adjacent edges to achieve transitive closure of the graph.

3.4.1. Temporal Authorizations. Computer security policies often distinguish between different "modes" in which a subject may access an object (e.g., observe, modify, execute, append). A temporal authorization Aτ, is a mapping of a subject-object pair to a set of mode-ϕ pairs, which completely defines the temporal authorization policy for the subject and object. For simplicity of presentation, it is assumed herein that there is only one mode-ϕ pair per subject-object pair. Ωτ is the set of temporal authorizations for a given system.

Definition 4 (Temporal Authorization). *A temporal authorization $\mathbf{A}_\tau$ is defined as a 4-tuple (s, o, m, ϕ) where:*

s *is a subject identifier*
o *is an object identifier*
$m \subseteq M$ *is the allowed mode(s) of access (e.g., read, write, read-write, execute)*
ϕ *is the access graph for s and o*

A temporal authorization states that a subject *s* is allowed *m* access to object *o* as restricted by the access graph ϕ. The association of an empty interval (*) with a vertex in $\mathbf{A}_\tau \phi$ indicates that there is no restriction as to the value of that interval – otherwise, the authorization requires that exact interval (see Section 3.4.2, below). Clearly, authorizations for policy-based equivalence classes of subjects or objects could be similarly specified.

3.4 Access Requests

A temporal subject, to gain access to a temporal object, initiates an access request for a given mode of access to occur at a particular time. In the most general form, temporal requests would specify an arbitrary time in the past, present and future. For simplicity in this discussion, requests will be characterized relative to now.τ . There are two types of access requests: general access requests and *duration access requests,* which differ in the interpretation of the length of requested access.

Definition 5 (General Access Request). *A general access request $\mathbf{R}_{g\tau}$ is a 4-tuple (s, o, m, τref) where:*

$s \in S_\tau$ *is a temporal subject*
$o \in O_\tau$ *is a temporal object*
$m \subset M$ *is a mode(s) of access*
τ_{ref} *is the time of access: now.τ*

This form of request can be evaluated with respect to the unit interval τ_{ref}, as is shown below, or as an open-ended request. The former corresponds well to a memory request in a system (such as at the hardware level of abstraction) where the subject's access rights are verified on every use of the object. For example, in $\mathbf{R}_{g\tau}(s_1, o_1, \{r,w\}, [5,6))$, subject s_1 requests read-write access to object o_1 for the unit interval starting at time 5. Alternatively, $\mathbf{R}_{g\tau}$ could be interpreted as a request for access starting at τ_{ref} and continuing for the maximum duration allowed by the access graph [1].

Definition 6 (Duration Access Request). *A duration access request $\mathbf{R}_{d\tau}$ is a 5-tuple (s, o, m, τref, δ) where:*

$s \in S_\tau$ *is a temporal subject*
$o \in O_\tau$ *is a temporal object*
$m \subseteq M$ *is the mode(s) of access*

τref *is the time of access: now.* τ

δ *is the requested duration of access:* $\delta >= 1$

In a duration request, the subject requests access to object *o* for the interval $[t_{now}, t_{now} + \delta)$. For example, in $\boldsymbol{R_{d_\tau}}(s_1, o_1, \{r\}, [1,2), 5)$, subject s_1 requests read access to object o_1 for the interval [1, 6). This form of request would be useful to model a system interface where a continuing interval of access would be granted.

3.4.2. Evaluation of Access Requests. An access request is evaluated as follows: the set $\boldsymbol{\Omega_\tau}$ of temporal authorizations is searched for a matching subject-object pair. If no match is found, access is denied. If a match is found, the requested mode is compared to the allowed mode, and then the access graph is interpreted relative to the requested interval, to grant or deny access. This process is specified for general and duration requests in the boolean functions ***Eval_g*** and ***Eval_d***, which state that the corresponding ϕ *is* true when evaluated using s.$\boldsymbol{\tau}$, o.$\boldsymbol{\tau}$, and the time reference.

Eval_g $(R_{g_\tau}(s, o, m, now.\tau))$

$\exists\ (s', o', m', \phi) \in \Omega_\tau$:

$s.id = s'$ &

$o.id = o'$ &

$m \subset m'$ &

$(s.\tau \rightarrow o.\tau) \subseteq \phi.\gamma(s{:}\tau,\ o{:}\tau)$ &

$(now.\tau \rightarrow o.\tau) \subseteq \phi.\gamma(\tau_{ref},\ o{:}\tau)$ &

$(now.\tau \rightarrow s.\tau) \subseteq \phi.\gamma(\tau_{ref},\ s{:}\tau)$

Eval_d $(R_{d_\tau}(s, o, m, now.\tau, \delta))$

$\exists\ (s', o', m', \phi) \in \Omega_\tau$ (

$s.id = s'$ &

$o.id = o'$ &

$m \subseteq m'$ &

$(s.\tau \rightarrow o.\tau) \subseteq \phi.\gamma(s{:}\tau,\ o{:}\tau)$ &

$(now.\tau \rightarrow o.\tau) \subseteq \phi.\gamma t_{now}, t_{now} + \delta)),\ o{:}\tau)$ &

$(now.\tau \rightarrow s.\tau) \subseteq \phi.\gamma([t_{now}, t_{now} + \delta)),\ s{:}\tau)$

The relations illustrated in Table 1 are used to compute whether an access graph $\boldsymbol{\phi}$ is satisfied. For example, consider the following:

Let:

$s.\tau = [1, 20)$

$o.\tau = [5, 40)$

$now.\tau = [6, 7)$

$R_{g_\tau}(s, o, \{r\}, [6, 7))$

$(s', o', \{r, w\}, \phi) \in \Omega_\tau$

ϕ = OVERLAPS$(s{:}\tau,\ o{:}\tau)$ $\wedge$ DURING$(\tau_{ref},\ s{:}\tau)$ $\wedge$ DURING$(\tau_{ref},\ o{:}\tau)$

Authorization is allowed since the following conditions are met:

1) $s.id = s',\ o.id = o'$

2) $m \subseteq m' = \{r\} \subseteq \{r, w\}$

3a) OVERLAPS$(s.\tau, o.\tau)$

$= (s.\tau- < o.\tau-) \wedge (s.\tau+ > o.\tau-) \wedge (s.\tau+ < o.\tau+)$

$= (1 < 5) \wedge (20 > 5) \wedge (20 < 40)$

$= true$

3b) DURING$(now.\tau, s.\tau)$

$= (now.\tau- > s.\tau-) \wedge (now.\tau.t+ < s.\tau+)$

$= (6 > 1) \wedge (7 < 20)$

$= true$

3c) DURING$(now.\tau, o.\tau)$

$= (now.\tau.t- > o.\tau-) \wedge (now.\tau.t+ < o.\tau+)$

$= (6 > 5) \wedge (7 < 40)$

$= true$

4. Related Work

Various authorization models using temporal constraints or temporal attributes have been proposed in the past. Bertino et al. proposed a Temporal Authorization Model (TAM) [6,7] that associated temporal constraints with access authorizations. TAM models temporal dependencies among authorizations, allowing for the application of rules to derive authorizations from other authorizations.

The notion of associating temporal constraints with authorizations was extended in another access control model proposed by Bertino, et al. that supported discontinuous temporal constraints on authorizations [5].

Role-Based Access Control (RBAC) has also been extended to support temporal constraints to the activation and deactivation of roles [8] as well as user-role and role-permission assignments [13].

The Temporal Data Authorization Model (TDAM) proposed by Alturi and Gal [3, 11] is more closely related to TIAC, in that their access control constraints are based on temporal attributes associated with the data as well the time of the data access request.

An algorithm for analyzing an arbitrary IA network was developed by Allen[2]. A limitation to this algorithm is that it may not detect all inconsistencies in networks with more than three nodes. Vilain and Kautz [16, 17], showed that finding a consistent labeling for an arbitrary IA network is NP-complete. Golumbic and Shamir [12] also show that determining consistency in an IA network for a specific set of problems is NP-complete. Allen pointed out that Freuder's [10] techniques to ensure total consistency for larger IA networks has a complexity that is exponential with respect to the number of nodes. However, for an IA network consisting of three nodes, the algorithm guarantees total consistency: it is both useful

and practical for TIAC and other models that deal with constraints on three time intervals.

Chetcuti-Sperandio and Massacci [9] presented a framework utilizing interval algebra for reasoning about the consistency of temporal relationships of authorization and delegation certificates. Authorization certificates contain statements about actions that the associated subject may perform on certain objects during a given time interval. Delegation certificates provide a means for one subject to pass their authorizations to another subject, during a given interval. Their approach differs from that presented here in several ways. As discussed below, their model cannot describe temporal attributes of an object, such as a "validity period." While the revocation of authorization certificates is not discussed[9], subject-based certificate revocation is often difficult, whereas a repository of authorizations (i.e., the Ω_τ structure) lends itself to various effective and innovative revocation approaches [1]. Also, while their presentation describes how a consistent authorization structure may be ensured, it does not present a model for the evaluation of requests, as does the TIAC model.

None of the authorization models mentioned above support policies based on temporal attributes associated with both subjects (e.g., a process representing the user) and objects (e.g., data). As introduced by Graham et al., Lampson, and Weissman [13, 15, 18] authorizations for subjects to access to objects such as files, I/O devices, shared memory, etc. have been modeled in the form of a table or access matrix. Graham et al. [13], described how the access matrix was derived from the 3-tuple of subjects, objects, and a set of allowed modes of access. The TIAC model extends this approach by adding time as a decision variable in the form of a new concept called the access graph ϕ. ϕ restricts the accesses allowed by the subject, object, mode 3-tuple, just as in the Bell and LaPadula model[4] the relationship of subject sensitivity labels to object sensitivity labels further restricts the discretionary modes of access allowed.

5. Future Research

Several areas related to TIAC are still being investigated. We are considering the formal semantics for creating and deleting temporal authorizations, as well as the policy implications of modifiable temporal attributes associated with subjects and objects. We are investigating an extension to the semantics of a general request to allow an authorized access to continue for the maximum duration allowed by the access graph, and then, how access can be automatically revoked after that duration has expired.

In general, a set of mode-φ pairs can be associated with each subject-object pair in order to express a different policy for each mode of access, but that extension to the TIAC model is left for future work. Similarly, the model might be extended to provide multiple authorization statements applicable to a given subject/object/mode triple, for example to accommodate system-wide over-riding restrictions as well as allowing different policies during different time periods or environmental situations (e.g., emergencies).

We also plan to generalize this model to allow an access requests other than current time, which would allow the model to check for previous, current, and future authorizations. Our research is also exploring the range of useful temporal access control policies that can be expressed using the TIAC model, as well as the composition of the TIAC with other policy models, such as those for multilevel security. Finally, we are currently developing a prototype implementation based on the TIAC model to verify its effectiveness and efficiency in a simple processing environment.

6. Conclusion

In this paper, we have presented the TIAC model as a novel way to specify temporal authorization policies based on time attributes associated with subjects and objects, and time of access. Access constraints are represented as a special form of a three-node interval algebra network, called a Time Interval Access Graph. The model also presents two forms of access request, and a specification for the evaluation of access requests with respect to the Access Graph.

7. References

[1] F. B. Afinidad, "An Interval Algebra-Based Temporal Access Control Protection Architecture," Dissertation, Naval Postgraduate School, Monterey, CA, 2005.

[2] J. F. Allen, "Maintaining Knowledge About Temporal Intervals," Communications of the ACM, vol. 26, no. 11, pp. 832-843, November 1983.

[3] V. Atluri and A. Gal, "An Authorization Model for Temporal and Derived Data: Securing Information Portals," ACM Transactions on Information and System Security, vol. 5, no. 1, pp. 62-94, February 2002.

[4] D. E. Bell and L. LaPadula. Secure Computer Systems: Mathematical Foundations and Model. Technical Report M74-244, MITRE Corp., Bedford, MA, 1973.

[5] E. Bertino, C. Bettini, E. Ferrari, and P. Samarati, "An Access Control Model Supporting Periodicity Constraints and Temporal Reasoning," ACM Transactions on Database Systems, vol. 23, no. 3, pp. 231-285, September 1998.

[6] E. Bertino, C. Bettini, and P. Samarati, "A Discretionary Access Control Model with Temporal Authorizations," in Proceedings of the 1994 Workshop on New Security Paradigms, 1994, pp. 102-107.

[7] E. Bertino, C. Bettini, and P. Samarati, "A Temporal Authorization Model," in Proceedings of the 2nd ACM Conference on Computer and Communications Security, 1994, pp. 126-135.
[8] E. Bertino, P. A. Bonatti, and E. Ferrari, "TRBAC: A Temporal Role-Based Access Control Model," in Proceedings of the 5th ACM Workshop on Role-Based Access Control, July 26-28, 2000, pp. 21-30.
[9] N. Chetcuti-Sperandio and F. Massacci. "A semantics and a calculi for reasoning about credential-based systems," in Proceedings of the International Workshop Methods for Modalities, September 22-23, 2003, pp. 61-76.
[10] E. C. Freuder, "A Sufficient Condition for Backtrack-Free Search," Journal of the Association for Computing Machinery, vol. 29(1), pp. 24-32, January 1982.
[11] A. Gal and V. Atluri, "An Authorization Model for Temporal Data," in Proceedings of the 7th ACM Conference on Computer and Communications Security, November 1-4, 2000, pp. 144-153.
[12] M. C. Golumbic and R. Shamir, "Complexity and Algorithms for Reasoning about Time: A Graph-Theoretic Approach," Journal of the Association for Computing Machinery, vol. 40(5), pp. 1108-1133, November 1993.
[13] G. S. Graham and P. J. Denning, "Protection---Principles and Practice," in Proceedings of the Spring Joint Computer Conference, May 16-18, 1972, pp. 417-429.
[14] J. B. D. Joshi, E. Bertino, U. Latif, and A. Ghafoor, "A Generalized Temporal Role-Based Access Control Model," IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 1, pp. 4-23, January 2005.
[15] B. W. Lampson, "Protection," in Proceedings of the 5th Princeton Symposium on Information Sciences and Systems, March, 1971, pp. 437-443, reprinted in Operating Systems Review, vol. 8, no. 1, January 1974, pp. 18-24.
[16] M. Vilain and H. Kautz, "Constraint Propagation Algorithms for Temporal Reasoning," in Proceedings of the Fifth National Conference on Artificial Intelligence, 1986, pp. 377-382.
[17] M. Vilain, H. Kautz, and P. van Beek, "Constraint Propagation Algorithms for Temporal Reasoning: A Revised Report," in Readings in Qualitative Reasoning about Physical Systems. San Francisco, CA: Morgan Kaufmann Publishers Inc., 1989.
[18] C. Weissman, "Security Controls in the ADEPT-50 Time-Sharing System," in Proceedings of the Fall Joint Computer Conference, November 18-20, 1969, pp. 119-133

Appendix A

This example shows how an IA network is derived from a set of simple temporal policy statements, and is then transformed into a consistent instantiation, an access graph ϕ.

Assume that an informal temporal access policy is as follows: a temporal subject (s) can access a temporal object (o) only if the subject was created before the object and both the subject and object are *valid* at the time of access. By convention, the property *valid* means that $now.\tau$ falls within the time interval associated with the temporal entity. No generality is lost due to the use of $now.\tau$ as the time of access, since all intervals are at least a unit interval in length. Also by convention, creation time is associated with the beginning of the subject's or object's interval ($\alpha.t$-).

Now consider that $s.\tau$, $o.\tau$, and $now.\tau$ are nodes in an IA network. The sentence "subject was created before the object" implies that $s.t$- is before $o.t$-; this is interpreted as a disjunction of all of the interval relations where $s.t- < o.t-$, and forms an interval algebra (IA) relation as follows:

$\gamma(s.\tau \rightarrow o.\tau)$: MEETS($s.\tau$, $o.\tau$) $\vee$ BEFORE($s.\tau$, $o.\tau$) $\vee$ OVERLAPS($s.\tau$, $o.\tau$) $\vee$ FINISHED_BY($s.\tau$, $o.\tau$) $\vee$ INCLUDES($s.\tau$, $o.\tau$)

The sentence "both the subject and object are valid at the time of access" implies that at the time of access, $now.\tau$ falls within both $s.\tau$ and $o.\tau$. This is represented as two IA relations as follows:

$\gamma(now.\tau \rightarrow o.\tau)$: DURING($now.\tau$, $o.\tau$) $\vee$ STARTS($now.\tau$, $o.\tau$) $\vee$ FINISHES($now.\tau$, $o.\tau$)

$\gamma(now.\tau \rightarrow s.\tau)$: DURING($now.\tau$, $s.\tau$) $\vee$ STARTS($now.\tau$, $s.\tau$) $\vee$ FINISHES($now.\tau$, $s.\tau$)

However, without computing the transitive closure, an IA network with these relations would contain relations that could never be true (MEETS($s.\tau$, $o.\tau$) $\vee$ BEFORE($s.\tau$, $o.\tau$)) if the other relations were true ($\gamma(now.\tau \rightarrow o.\tau)$ and $\gamma(now.\tau \rightarrow o.\tau)$). Leaving untrue relations in the graph could lead to inconsistency, for example, if other parts of $\gamma(s \rightarrow o)$ were eventually removed, and would also adversely affect the efficiency of request processing. Another reason for performing the transitive closure is to detect policy incompleteness. The transitive closure results in these relations:

$\gamma(s.\tau \rightarrow o.\tau)$: OVERLAPS($s.\tau$, $o.\tau$) $\vee$ FINISHED_BY($s.\tau$, $o.\tau$) $\vee$ INCLUDES($s.\tau$, $o.\tau$)

$\gamma(now.\tau \rightarrow o.\tau)$:DURING($now.\tau$, $o.\tau$) $\vee$ STARTS($now.\tau$, $o.\tau$) $\vee$ FINISHES($now.\tau$, $o.\tau$)

$\gamma(now.\tau \rightarrow s.\tau)$:DURING($now.\tau$, $s.\tau$) $\vee$ FINISHES($now.\tau$, $s.\tau$)

This forms a consistent instantiation of the IA network, resulting in the desired access graph ϕ. Details of the transitive closure computation for this example are provided in the next section.

Derivation of the Access Graph ϕ. In this section, the IA relations in Example 1 are composed together into a simple IA network. As each IA relation is added to the IA network, the algorithm for computing the transitive closure is applied to ensure that the IA network is consistent and complete. In the details shown below, the intervals for the three nodes are: $s.\tau$ (subject), $o.\tau$ (object), and, $now.\tau$ (time of access). The time interval

relationship between each node represents the directional edge that connects each node.

Step 1: Convert $\gamma(s.\tau \rightarrow o.\tau)$ to its network representation

$\gamma(s.\tau \rightarrow o.\tau)$:

MEETS($s.\tau$, $o.\tau$) ∨ BEFORE($s.\tau$, $o.\tau$)
∨ OVERLAPS($s.\tau$, $o.\tau$) ∨ FINISHED_BY($s.\tau$, $o.\tau$) ∨
INCLUDES($s.\tau$, $o.\tau$)

$s.\tau$ — (m < o fi di) → $o.\tau$ **IA_n-1**

Step 2: Convert γ **(*now*.$\tau \rightarrow o.\tau$)** to the network representation of $\gamma(o.\tau \rightarrow now.\tau)$

$\gamma(now.\tau \rightarrow o.\tau)$: DURING($now.\tau$, $o.\tau$) ∨
STARTS($now.\tau$, $o.\tau$) ∨
FINISHES($now.\tau$, $o.\tau$)

$o.\tau$ — (di si fi) → $now.\tau$ **IA_n-2**

Note: In this and subsequent conversions, the inverse may be used to adjust the direction of the relationship from one interval to another.

Step 3: Add **IA_n-2** to **IA_n-1** to form a new network shown in Figure 3.

$s.\tau$ — (m < o fi di) → $o.\tau$ — (di si fi) → $now.\tau$

Figure A-1. Network representation after adding IA_n-2 to IA_n-1

Step 4: Compute the transitive closure of the IA network in Figure 3 to compute any consequences (inferred relation) between $s.\tau$ and $now.\tau$. Use the transitivity table (Appendix B) on each pair of basic relations $T(r_1, r_2)$ then take the union of all the results. As we refer to Appendix B, r_1 is a relation between $s.\tau$ and $o.\tau$ and r_2 is a relation between $o.\tau$ and $now.\tau$.

T (m, di) = (<)
T (m, si) = (m)
T (m, fi) = (<)
T(<, di) = (<)
T (<, si) = (<)
T (<, fi) = (<)
T (o, di) = (< o m di fi)
T (o, si) = (di fi o)
T (o, fi) = (< o m)
T (fi, di) = (di)
T (fi, si) = (di)
T (fi, fi) = (fi)
T (di, di) = (di)
T (di, si) = (di)
T (di, fi) = (di)

The union of the results is (< o m di fi). The network is updated to reflect the new inferred interval relation between $s.\tau$ and $now.\tau$ as shown in Figure 4.

$s.\tau$ — (m < o fi di) → $o.\tau$ − (di si fi) → $now.\tau$
↑
————————(< o m di fi)————————

Figure A-2. IA network after computing transitive closure

Step 5: Convert $\gamma(now.\tau \rightarrow s.\tau)$ to its network representation and add it to the network in Figure 4

$\gamma(now.\tau \rightarrow s.\tau)$: DURING($now.\tau$, $s.\tau$) ∨
STARTS($now.\tau$, $s.\tau$) ∨
FINISHES($now.\tau$, $s.\tau$)

$s.\tau$ — (di si fi) → $now.\tau$ **IA_n-3**

Two relations for $s.\tau$ — () → $now.\tau$ now exist in the network, their intersection is computed to derive the new inferred relation between $s.\tau$ and $now.\tau$.

($s.\tau$ — (di si fi) → $now.\tau$) ∩
($s.\tau$ — (< o m di fi) → $now.\tau$)
= $s.\tau$ — (di fi) → $now.\tau$

The IA network is updated with this new inferred relation as shown in Figure 5.

$s.\tau$ — (m < o fi di) → $o.\tau$ — (di si fi) → $now.\tau$
↑
————————*(di fi)*————————————

Figure A-3. IA network representation after adding IAn-3

Step 6: Since the interval relation between $s.\tau$ and $now.\tau$ has changed, the algorithm for determining transitive closure of the network is re-applied to the remaining interval relations.

Step 6a: Starting with the pair of relations $s.\tau$ — (di fi) → $now.\tau$ and $now.\tau$ — (d s f) → $o.\tau$ (inverted) we have:

T(di, d) = (o oi d s f di si fi =)
T (di s) = (di fi o)
T (di f) = (di si oi)
T (fi d) = (o d s)
T (fi s) = (o)
T (fi f) = (f fi =)

The union of the results is (o oi di si fi d s f =). A new relation between $s.\tau$ and $o.\tau$ is inferred:

$s.\tau$ — (o oi di si fi d s f =) → $o.\tau$

Since two relations for $s.\tau$ — () → $o.\tau$ now exist in the network, their intersection is computed to derive the new inferred relation between $s.\tau$ and $o.\tau$.

($s.\tau$ — (o oi di si fi d s f =) → $o.\tau$) ∩
($s.\tau$ — (m < o fi di) → $o.\tau$)
= $s.\tau$ — (o di fi) → $o.\tau$

The IA network is updated with this new inferred relation as shown in Figure 6.

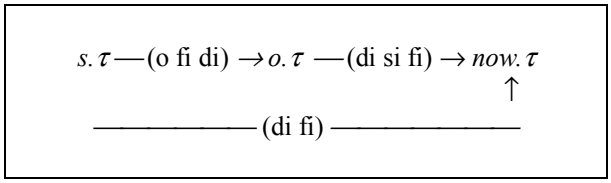


Figure A-4. A network with new inferred relation between $s.\tau$ and $o.\tau$

Step 6b: The process is continued with next pair of interval relations $o.\tau$ — (oi d f) → $s.\tau$ (inverted) and $s.\tau$ — (di fi) → $now.\tau$.

T (oi di) = (> oi mi di si)
T (oi fi) = (oi di si)
T (d di) = nothing
T (d fi) = (< o m d s)
T (f di) = (> oi mi di si)
T (f fi) = (f fi =)

The union of the results above is (< > o oi m mi d di s si f fi =) and a new relation between $o.\tau$ and $now.\tau$ is inferred:

$o.\tau$ — (< > o oi m mi d di s si f fi =) → $now.\tau$

Since two relations for $o.\tau$ — () → $now.\tau$ now exist in the network, their intersection is computed to derive the new inferred relation between $o.\tau$ and $now.\tau$.

($o.\tau$ — (< > o oi m mi d di s si f fi =) → $now.\tau$) ∩ ($o.\tau$ — (di si fi) → $now.\tau$)
= ($o.\tau$ — (di si fi) → $now.\tau$)

Since the relation between $o.\tau$ and $now.\tau$ remains the same ($o.\tau$ — (di si fi) → $now.\tau$), the network in Figure 6 does not need to be updated. The final network in Figure 7 is identical to the network in Figure 6.

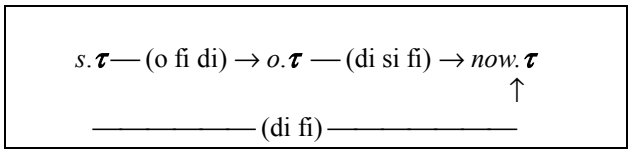


Figure A-5. Final network configuration with consistent labeling of relations

The final network represents the access graph ϕ (in network form) that is consistent and complete in its representation of the temporal authorization policy. Figure 8 shows the predicate form of the access graph derived from the final network in Figure 7.

$\phi(s.\tau, o.\tau, now.\tau)$ =
(OVERLAPS($s.\tau$, $o.\tau$) ∨ FINISHES($o.\tau$, $s.\tau$)
∨ DURING($o.\tau$, $s.\tau$))
∧ (DURING($now.\tau$, $s.\tau$) ∨ FINISHES($now.\tau$, $s.\tau$))
∧ (DURING($now.\tau$, $o.\tau$) ∨ STARTS($now.\tau$, $o.\tau$) ∨
FINISHES($now.\tau$, $o.\tau$))

Figure A-6. Predicate form of final access graph derived from Figure 7

Appendix B. Interval algebra transitivity table, showing relations A→C

A→B \ B→C	<	>	d	di	o	oi	m	mi	s	si	f	fi
<	<	no info	< o m d s	<	<	< o m d s	<	< o m d s	<	<	< o m d s	<
>	no info	>	> oi mi d f	>	> oi mi d f	>	> oi mi d f	>	> oi mi d f	>	>	>
d	<	>	d	no info	< o m d s	> oi mi d f	<	>	d	> oi mi d f	d	< o m d s
di	< o m di fi	> oi di mi si	o oi d s f di si fi =	di	o di fi	oi di si	o di fi	oi di si	di fi o	di	di si oi	di
o	<	> oi di mi si	o d s	< o m di fi	< o m	o oi d s f di si fi =	<	oi di si	o	di fi o	d s o	< o m
oi	< o m di fi	>	oi d f	> oi mi di si	o oi d s f di si fi =	> oi mi	o di fi	>	oi d f	oi > mi	oi	oi di si
m	<	> oi mi di si	o d s	<	<	o d s	<	f fi =	m	m	d s o	<
mi	< o m di fi	>	oi d f	>	oi d f	>	s si =	>	d f oi	>	mi	mi
s	<	>	d	< o m di fi	< o m	oi d f	<	mi	s	s si =	d	< m o
si	< o m di fi	>	oi d f	di	o di fi	oi	o di fi	mi	s si =	si	oi	di
f	<	>	d	> oi mi di si	o d s	> oi mi	m	>	d	> oi mi	f	f fi =
fi	<	> oi mi di si	o d s	di	o	oi di si	m	si oi di	o	di	f fi =	fi