Protecting Inappropriate Release of Data from Realistic Databases

Gio Wiederhold and Michel Bilello

Dep. of Computer Science
Stanford University, Stanford CA 94305
{gio, michel}@cs.stanford.edu

Abstract

In databases that are used for internal operations the data are not organized according to external access criteria. When collaboration with external customers is required the common tools of authentication, authorization, and secure transmission are inadequate to protect against release of inappropriate data. The approach used in the TIHI/SAW projects at Stanford adds a release filter. Such a release filter can be awkward and costly. By driving the filtering primitives through simple rules we allow a security officer to manage the institution policy and balance manual effort and complexity. A byproduct of the approach is a lowered dependence on perfect data management.

1. Introduction

There are data sources that are primarily intended for external access, such as public web pages, reports, bibliographies, etc. These are organized according to external access criteria. If some of the information is not intended to by public, then security provisions are put into place. Customers will have to authenticate themselves, typically through the use of passwords, and after authentication will be authorized to gain certain access and computational privileges [CastanoFMS:95]. The databases – we use the term here to include semi-structured information sources [LuniewskiEa:93] as well -- are logically partitioned according to the types of authorizations that external customers can receive. In some business settings some system customers may have access to all data, others may not have access to, say internal cost information of products, and others may not have access to sales volumes. Such restrictions can be encoded in view definitions of relational databases, and are managed by a database administrator and the associated staff [GriffithW:76]. Secure transmission is the responsibility of the corporate networking staff [He:97].

There are many situations where this scenario is not realistic [Bellovin:97]. A corporation may have large, existing databases which were established before external access needed to be considered. A firewall provides isolation of the enterprise from external attacks, but hinders legitimate accessors [CheswickB:94]. In modern environments convenient access will be needed for off-site staff, corporate salespersons, vendors which have contract relationships, government inspectors, and an ever-increasing number of collaborators [Bellovin:96]. Reorganizing corporate databases to deal with developing needs for external access is a challenge that traditional approaches do not easily handle. The staff concerned with security will also be concerned with entrusting the protection of valuable corporate data to database administrators and networking staff. These people are promoted to their positions because they have a helpful attitude and know how to overcome problems of system failures and inadequacies. This attitude is inherently in conflict with corporate and legal concerns for the protection of data. It is not surprising that security concerns were the cited as the prime reason for lack of progress in establishing *virtual enterprises* [HardwickS:96].

We encountered the problem initially in the manufacturing area, where security concerns caused the interchange of manufacturing data to take many weeks, although they had installed compatible CAD systems. All drawings had to printed, inspected, verified, and edited if the design contained information inappropriate for the subcontractor. The edited drawings could then be copied and shipped to the contractor, who had to scan them into their systems. The source of the problem is, of course, that the design engineer makes drawings of the equipment to be built, with justifications, finishing information, and explicit and implicit performance criteria. The drawings are not initially produced to satisfy the capabilities of an unknown subcontractor.

Our actual initial application domain was actually in healthcare. Medical records are needed for many purposes: diagnosis, care delivery, drug supplies, infection control, room assignments, billing, insurance claims, validation of proper care, research, and public health records. Patient care demands that the record be accessible in a comprehensive form and up-to-date [Rindfleisch:97]. Historical information is important for disease management, but not for many billing tasks. It is obviously impossible to split the record into access categories that match every dimension of access. Even if that would be possible, the cost and risks to the internal operations in a hospital or clinic would be prohibitive.

The solution we provide to this dilemma is *result checking* [WiederholdBSQ:96]. In addition to the conventional tasks of access control we check the results of information requests before releasing them to the customer, as well as a large number of parameters about the release. This task mimics the manual function of a security officer in checking the briefcases of collaborating participants in a secure meeting, when the attendees leave the secure facility. We incorporate result checking in a *security mediator* workstation, managed by a security officer [WiederholdEa:96].

2. System Architecture

The *security mediator* system interposes security checking between external accessors and the data resources to be protected. Physically a security mediator is designed to operate on a distinct workstation, owned and operated by the enterprise security officer. It is positioned as a pass gate within the enterprise firewall, if there is such a firewall. In our initial commercial installation the security mediator also provided traditional firewall functions, by limiting the IP addresses of requestors [WiederholdBD:98]. The components are shown in Figure 1.

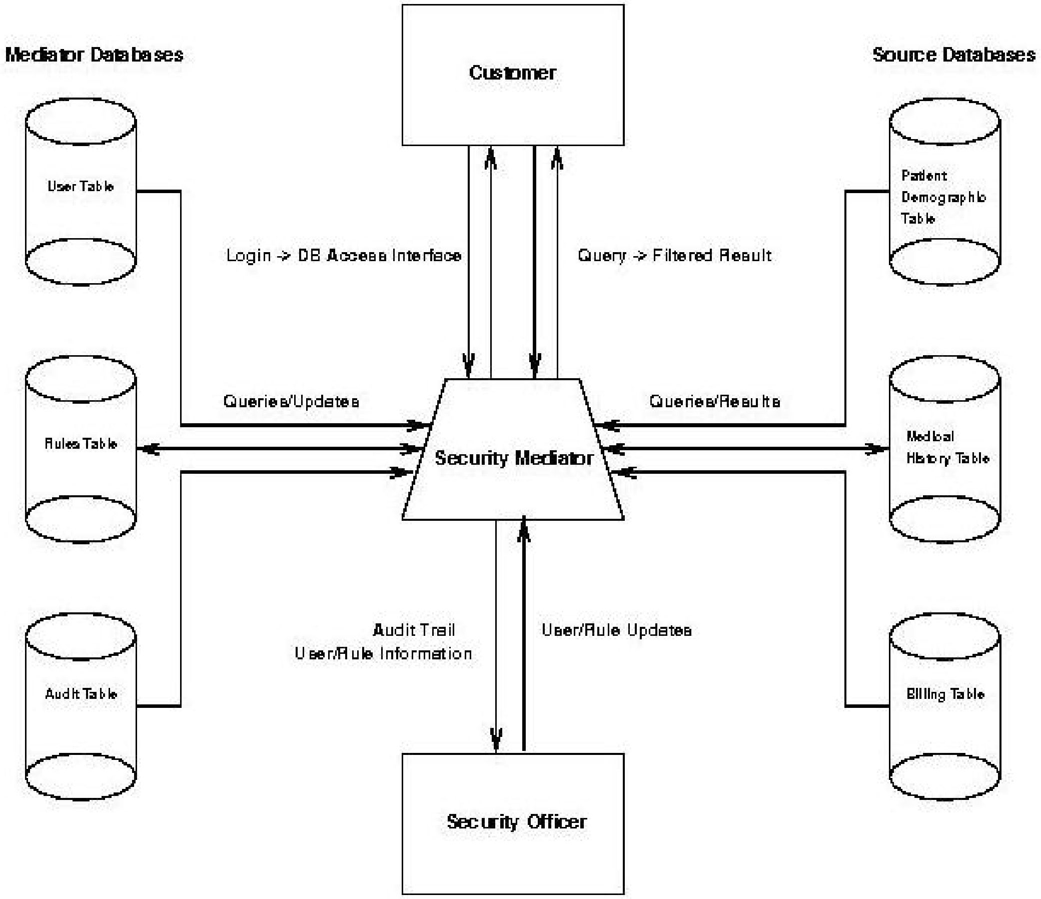
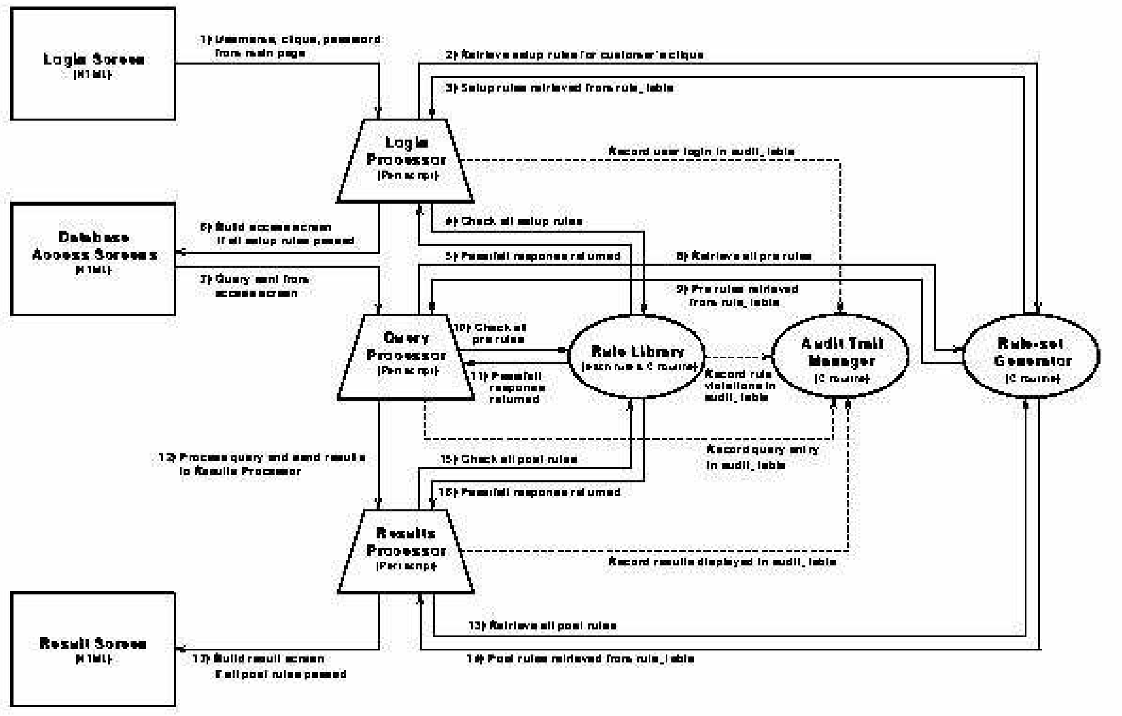


Fig.1: Components of a Security Mediator.

The mediator system and the source databases typically reside on different machines. Thus, since all queries are processed by the mediator, the database behind a firewall need not be secure unless it operates in a particularly high security setting. When combined with an integrating mediator, a security mediator can also serve multiple data resources behind a firewall [Ullman:96]. Combining multiple sources prior to result checking improves the scope of result validation. Of course, a minimum level of reliability is required for effective processing. For instance, the security mediator cannot protect from inadvertent or intentional denial of information by a mismanaged database system.

Operation

Within the workstation is a rule-based system which investigates queries coming in and results to be transmitted to the external world. Any request and any result which cannot be vetted by the rule system is displayed to the security officer, for manual handling. The security officer decides to approve, edit, or reject the information. An associated logging subsystem provides an audit trail for all information that enters or leaves the domain. The log provides input to the security officer to aid in evolving the rule set, and increasing the effectiveness of the system.



Login Screen (HTML)
1) Username, clique, password from main page
2) Retrieve setup rules for customer's clique
3) Setup rules retrieved from rule_table
Login Processor (Perl script)
Record user login in audit_table
Database Access Screens (HTML)
6) Build access screen if all setup rules passed
4) Check all setup rules
5) Pass/fail response returned
8) Retrieve all pre rules
7) Query sent from access screen
9) Pre rules retrieved from rule_table
Query Processor (Perl script)
10) Check all pre rules
11) Pass/fail response returned
Rule Library (each rule a C routine)
Record rule violations in audit_table
Audit Trail Manager (C routine)
Rule-set Generator (C routine)
12) Process query and send results to Results Processor
15) Check all post rules
Record query entry in audit_table
16) Pass/fail response returned
Results Processor (Perl script)
Record results displayed in audit_table
Result Screen (HTML)
13) Retrieve all post rules
17) Build result screen if all post rules passed
14) Post rules retrieved from rule_table

Figure 2: Software Interactions in a Security Mediator.

The software of our security mediator is composed of modules that perform the following tasks

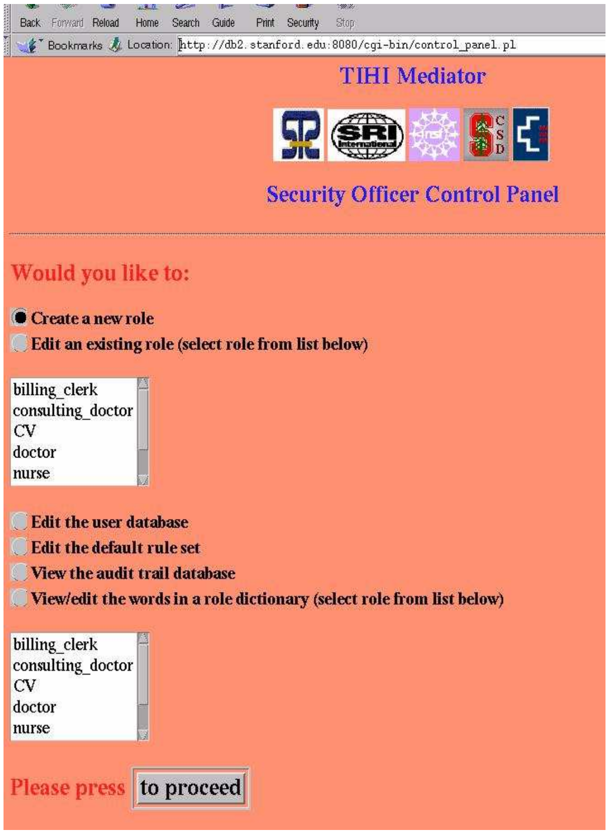
1. Optionally (if there is no firewall): Authentication of the requestor
2. Determination of authorization type (clique) for the requestor
3. Processing of a request for information (pre-processing) using the policy rules
4. If the request is dubious: interaction with the security officer
5. Communication with internal databases (submission of request and retrieval of results)
6. Processing of results (post-processing) using the policy rules
7. If the result is dubious: interaction with the security officer
8. Writing into a log file
9. Transmission of vetted information to the requestor

The mediator system can operate fully interactively or partially automatically, depending on the coverage of the rule-set. A reasonable goal is the automatic processing of say, 90% of queries and 95% responses, but even an empty, rule-less security mediator can greatly improve operations. Without rules, all interactions are presented to the security officer, but requests and results are now viewed on-line, and immediately editable, if needed. In time, simple rules can be entered to reduce the load on the officer. Currently, the secure paper-based systems we encountered often take weeks for turn-around. In many other situation we are aware of, security mechanisms are bypassed when requests for information are deemed to be important. Keeping the security officer in control allows any needed bypassing to be handled formally. This capability recognizes that in a dynamic, interactive world there will always be cases that are not foreseen or situations the rules are too stringent. Keeping the management of exceptions within the system greatly reduces confusion, errors, and liabilities.

Even when operating automatically, the security mediator remains under the control of the enterprise since the rules are modifiable by the security officer at all times. In addition, logs are accessible to the officer, who can keep track of the transactions. If some rules are found to be to liberal, policy can be tightened. If rules are too stringent, as evidenced by an excessive load on the security officer, they can be relaxed or elaborated.

3. The Rule System

The rules system is composed of the rules themselves, an interpreter for the rules, and primitives which are invoked by the rules. The rules embody the security policy of the enterprise. They are hence not preset into the software of the security mediator.



http://db2.stanford.edu:8080/cgi-bin/control_panel.pl

TIHI Mediator

Security Officer Control Panel

Would you like to:

● Create a new role
○ Edit an existing role (select role from list below)

billing_clerk
consulting_doctor
CV
doctor
nurse

○ Edit the user database
○ Edit the default rule set
○ View the audit trail database
○ View/edit the words in a role dictionary (select role from list below)

billing_clerk
consulting_doctor
CV
doctor
nurse

Please press [to proceed]

Figure 3. Security Officer Initial Screen.

In order to automate the process of controlling access and ensuring the security of information, the security officer enters rules into the system. These rules are trigger analyses of requests, their results, and a number of associated parameters. The interpreting software uses these rules to determine the validity of every request and make the decisions pertaining to the disposition of the results. Auxiliary functions help the security officer enter appropriate rules and update them as the security needs of the organization change. Access for the security officer is provided as an active local browser form, using Netscape facilities. Figure 3 shows the initial screen display.

The rules are simple, short and comprehensive. They are stored in a database local to the security mediator with all edit rights restricted to the security officer. Some rules may overlap, in which case the most restrictive rule automatically applies. The rules may pertain to requestors, cliques, sessions, databases tables or any combinations of these.

Rules are selected based on the authorization clique selected for the requestor. All the applicable rules will be checked for every request issued by the requestor in every session. All rules will be enforced for every requestor and the request will be forwarded to the source databases only if it passes all tests. Any request not fully vetted is posted immediately to the log and sent the security officer. The failure message is directed to the security officer and not to the requestor, so that the requestors in such cases will not see the failure and its cause. This prevents that the requestor could interpret failure patterns and make meaningful inferences, or rephrase the request to try to bypass the filter.

Result Checking

The novel aspect of our approach is that security mediator checks outgoing results as well. This is crucial since, from the security-point-of-view, requests are inclusive, not exclusive selectors of content and may retrieve unexpected information. In helpful, user-friendly information systems getting more than asked for is considered beneficial, but from a security point-of-view it is not. Thus, even when the request has been validated, the results are also subject to screening by a set of rules. As before, all rules are enforced for every requestor and the results are accessible only if they pass all tests. Again, if the results violate a rule, a failure message is logged and sent to the security officer but not to the requestor.

Primitives

The rules invoke executable primitive functions which operate on requests, data, the log, and other information sources. As new security functions and technologies appear, or if specialized needs arise, new primitives can be inserted in the security mediator for subsequent rule invocation. In fact, we do not expect to be the source of all primitives. We do hope that all primitives will be sufficiently simple that their correct function can be verified.

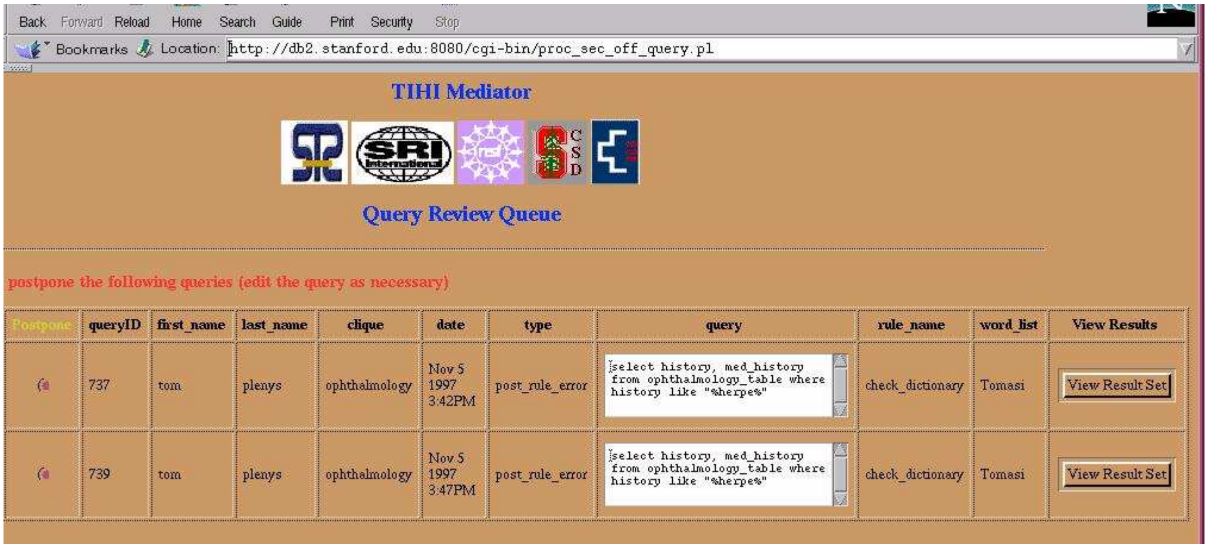
Primitives which have been used include:

- Assignment of a requestor to a clique
- Limit access for clique to certain database table segments or columns
- Limit request to statistical (average, median, ..) information
- Provide number of data instances (database rows) used in result
- Provide number of tables used (joins) for result
- Limit number of requests per session
- Limit number of sessions per period
- Limit requests by requestor per period
- Block requests from all but listed sites
- Block delivery of results to all but listed sites
- Block receipt of requests by local time at request site
- Block delivery of results by local time at delivery site
- Constrain request to data which is keyed to requestor name
- Constrain request to data which is keyed to request site name
- Filter all result terms through a clique-specific good-word dictionary
- Disallow results containing terms in a clique-specific good-word dictionary
- Convert text by replacing identifies with non-identifying surrogates [Sweeney:96]
- Convert text by replacing objectionable terms with surrogates
- Randomizing responses for legal protection [Leiss:82]
- Extract text out of x-ray images (for further filtering)
- Notify the security officer immediately of failure reports
- Place failure reports only in the log

Not all primitives will have a role in all applications.

Primitives can vary greatly in cost of application, although modern technology helps. Checking for terms in results is costly in principle, but modern spell-checkers show that it can be done fairly fast. For this task we create clique-specific dictionaries, by initially processing a substantial amount of approved results. In initial use the security officer will still get false failure reports, due to innocent terms that are not yet in the dictionary. Those will be incrementally added, so that in time the incidence of such failures will be minimal.

For example, we have in use a dictionary for ophtamology, to allow authenticated researchers in that field to have access to patient data. That dictionary does not include terms that would signal, say HIV infection or pregnancies, information which the patients would not like to see released to unknown research groups. Also, all proper names, places of employment, etc. are effectively filtered.



http://db2.stanford.edu:8080/cgi-bin/proc_sec_off_query.pl

TIHI Mediator

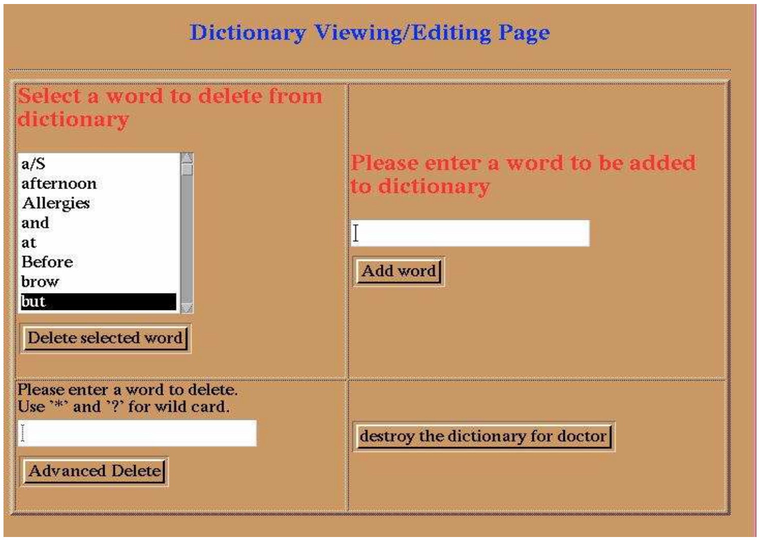
Query Review Queue

postpone the following queries (edit the query as necessary)

Postpone	queryID	first_name	last_name	clique	date	type	query	rule_name	word_list	View Results
●	737	tom	plenys	ophthalmology	Nov 5 1997 3:42PM	post_rule_error	select history, med_history from ophthalmology_table where history like "%herpe%"	check_dictionary	Tomasi	View Result Set
●	739	tom	plenys	ophthalmology	Nov 5 1997 3:47PM	post_rule_error	select history, med_history from ophthalmology_table where history like "%herpe%"	check_dictionary	Tomasi	View Result Set

Figure 4. Failure Reports Received by the Security Officer The impatient requestor entered the request twice.

The security officer must deal with such a failure. It turns out that a colleague's name appeared in the record. The security officer can now decide to omit that name from the result, to protect the privacy of the record, allow the name to appear in this instance, add the name to the dictionary so further occurrences of that name will not cause failure as shown in Figure 5.. The default is conservative: the researcher will not receive this particular record.



Dictionary Viewing/Editing Page

Select a word to delete from dictionary

a/S
afternoon
Allergies
and
at
Before
brow
but

Please enter a word to be added to dictionary

Please enter a word to delete.
Use `*' and `?' for wild card.

Figure 5: Dictionary edit screen

We have also experimented with a tool, SCRUB, which automatically removes identifying information, as names, places of residence and employment, and similar information from medical record results [Sweeney:96]. While in our ophtamology filter some names will pass, say a Dr. Iris, SCRUB uses linguistic clues to located identifiers.

Several of these primitives are designed to help control inference problems in statistical database queries [AdamW:89]. While neither we, nor any feasible system can prevent leaks due to inference, we believe that careful management can make reduce the probability, and dealing with it specifically, as logging all accesses will reduce the practical threat [Sweeney:97]. These will often have to refer to the log, so that efficient access to the log, for instance by maintaining a large write-through cache for the log will be important. Here again there the function of traditional database support and security mediation diverges, since database transaction are best isolated, where as inference control requires history maintenance.

Rule types

Rules relate requestors, or more often their cliques, to sets of primitives to be executed. The rules are hence easy to comprehend, although limited in complexity. Currently none of our rules invoke directly other rules, although linkage can occur due to the result of their actions.

The rules can be classified as *set-up* or maintenance rules, *pre-processing* (request) rules and *post-processing* (result) rules. Some primitives may be invoked by any class, but most primitives are specific to pre- or post-processing rules. Examples of primitives for pre-processing rules include the imitation of the number of queries per session, the number of queries for the clique, the session duration, etc. Post-processing rules can check against a minimum number of instances retrieved, restrictions on intersection of queries.

Not only are the rules easy to comprehend and to enter into the system as shown in the figure of the interface for the security officer, they are also powerful enough to enable the officer to specify requirements and criteria accurately, so that whenever requestors may see all information, they should be allowed to do so and whenever information is restricted, they should have no access to it. The requestors in the system are grouped as cliques and rules may apply to one or more cliques. The security officer has the authority to add or delete requestors from cliques and to create/drop cliques. Similarly, columns in tables can be grouped into segments and request/results validations could be performed on segments. tables can be grouped into segments and request/results validations could be performed on segments.

4. Logging

Throughout, the failures, as well as the request text and source, and actions taken by the security officer, are logged by the system for audit purposes. Having a security log which is distinct from the database log is important since:

- A database system logs all transactions, not just external requests, and is hence confusingly voluminous
- Most database systems do not log attempted and failed requests fully, because they appear not to have affected the databases
- Reasons for failure of requests in database logs are implicit, and do not give the rules that caused them.

We provide user-friendly utilities to scan the security log by time, by requestor, by clique, and by data source. Offending terms in results are marked.

No system, except one that provides complete isolation, can be 100% foolproof. The provision of security is, unfortunately, a cat-and-mouse game, where new threats and new technologies keep arising. Logging provides the feedback which converts a static approach to a dynamic and stable system, which can maintain an adequate level of protection. Logs will have to be inspected regularly to achieve stability.

Bypassing of the entire system and hence the log remains a threat. Removal of information on portable media is easy. Only a few enterprises can afford to place controls on all personnel leaving daily for home, lunch, or competitive employment. However, having an effective and adaptable security filter removes the excuse that information had to be downloaded and shipped out because the system was to stringent for legitimate purposes. Some enterprises are considering limiting internal workstations to be diskless. It is unclear how effective this approach will be outside of small, highly secure domains in an enterprise. Such a domain will then have to be protected with its own firewall and a security mediator as well, because collaboration between the general and highly secure internal domains must be enabled.

5. Current State and Further Work

Our initial demonstrations have been in the healthcare domain, and a commercial version of TIHI is now in use to protect records of genomic analyses in a pharmaceutical company. As the expectations for protection of the privacy of patient data are being solidified into governmental regulations we expect that our approach will gain popularity [Braithwaite:96]. Today the healthcare establishment still hopes that commercial encryption tools will be adequate for the protection of medical records, since the complexity of access requirements has not yet been faced [RindKSSCB:97]. Expenditures for security in medical enterprises is minimal [ClaytonEa:97]. Funding of adequate provisions in an industry under heavy economic pressures, populated with many individuals who do not attach much value to the privacy of others, will remain a source of stress.

We are now going back to our original motivating application area, manufacturing information [QianW:97]. Here the simple web-interfaces which are effective for the customer and the security officer interfaces in health care are not adequate. We are preparing interfaces for the general viewing and editing of design drawings and attached textual information. The text may not only appear in ASCII, but may also be incorporated in the drawings themselves. When delivering an edited drawing electronically, we also have to assure that there is no hidden information. Many design formats allow *undo* operations, which would allow apparently deleted information to reappear.

Before moving to substantial automation for collaboration in manufacturing, we will have to understand the parameters for reliable filtering of such information better. However, as pointed out initially, even a fully manual security mediator will provide a substantial benefit to enterprises that are trying to institute shared efforts rapidly.

6. Conclusions

Security mediation provides an architectural function as well as a specific service. Architecturally, expanding the role of a gateway in the firewall from a passive filter to an active pass gate service allows concentration of the responsibility for security to a single node, owned by the security officer [DAllotto:96]. Existing services, as constraining views over databases, encryption for transmission in networks, password management in operating systems can be managed via such a node.

The specific, novel service is, of course, result checking, which complements traditional access control. We have applied for a patent to cover the concept. Checking results introduces practicality into securing data-intensive systems. The requirement that systems that are limited to access-control impose, namely that all data are correctly partitioned and filed is not achievable in practice. The rules balance the need for preserving data security and privacy and for making data available. Data which is too tightly controlled reduces the benefits of information in collaborative settings. Rules which are too liberal can violate security and expectation of privacy. Having a balanced policy will require directions from management. Having a single focus for execution of the policy in electronic transmission will improve the consistency of the application of the policy.

A side-effect of result checking that it provides a level of intrusion detection. Intruders that successfully penetrate the system may well be caught when removing results. For instance, if a terminology filter over results is in use, any removal of passwords will be caught. We have not investigated primitives and rules specifically to detect intrusion. Here the system may want to respond seemingly normally, in order to gather more information for pursuit and prosecution.

Acknowledgements

Research leading to security mediators was supported by an NSF HPCC challenge grant and by DARPA ITO via Arpa order E017, as subcontracts via SRI International. Steve Dawson is the PI at SRI. The commercial transition was performed by Maggie Johnson, Chris Donahue, and Jerry Cain under contracts with SST (www.2ST.com). Work on editing and filtering graphics is due to Jahnavi Akella and James Z. Wang, The images were extracted by Yan Tan.

References

AdamW:1989] Adam,N.R. and Wortmann,J.C.: Security-Control Methods for Statistical Databases: a Comparative Study; *ACM Computing Surveys*, Vol. 25 No.4, Dec. 1989.

[Bellovin:97] Steven M. Bellovin, "Network and Internet Security", in Peter Denning and Dorothy Denning, eds., *Internet Besieged: Countering Cyberspace Scofflaws*, ACM Press, 1997.

[Bellovin:96]Steven M. Bellovin, "Network Security Issues", in A. Tucker, ed., *CRC Computer Science and Engineering Handbook*, CRC Press, 1996.

[Braithwaite:96] Bill Braithwaite: "National Health Information Privacy Bill Generates Heat at SCAMC"; *Journal of the American Informatics Association*, 1996:3(1):95-96.

[CastanoFMS:95] S.Castano, M.G. Fugini, G.Martella, and P. Samarati: *Database Security*; Addison Wesley Publishing Company - ACM Press, 1995, pp. 456

[CheswickB:94] W. R. Cheswick and S. M. Bellovin. *Firewalls and Internet Security: Repelling the Wily Hacker*. Addison-Wesley, 1994.

[ClaytonEa:97] Paul Clayton (chair): For the Record: *Protecting Electronic Information*; National Academy Press, 1997.

[DAllotto:96] L. J. D'Alotto: "Internet Firewalls Policy Development and Technology Choices"; *Proc. 19th National Information Systems Security Conference*, Nist and National Computer Security Center, Baltimore, MD, 1996, pp.259-266.

[GriffithW:76] Paula P. Griffiths, B.W. Wade: "An Authorization Mechanism for a Relational Database System"; *ACM Transactions on Database System*, 1976:1(3):242-255.

[HardwickSRM:96]Martin Hardwick, David L. Spooner, T. Rando, KC Morris: "Sharing Manufacturing Information in Virtual Enterprises';*Comm. AC*M, 1996, Vol.39 No.2, pp.46-54.

[He:97] J. He: "Performance and Manageability Design in an Enterprise Network Security System"; *IEEE Enterprise Networking Miniconference 1997 (ENM-97*), Montreal, Canada, June, 1997.

[Leiss:82] E. Leiss.: "Randomizing, A Practical Method for Protecting Statistical Databases Against Compromise"; in *Proceedings of the Conference on Very Large Databases (VLDB) 8*, McLeod and Villasenor(eds), Morgan Kaufman pubs., August 1982, pp 189--196.

[LuniewskiEa:93] Allan Luniewski et al.: "Information Organization Using Rufus"; *ACM SIGMOD 93 Internat'l Conf. on Management of Data, SIGMOD Record*, June 1993, Vol.22 No.2 p.560-561

[QianW:97] XioaLei Qian and Gio Wiederhold: "Protecting Collaboration"; abstract for *IEEE Information Survivability Workshop, ISW'97*,Feb.1997, San Diego.

[RindEa:97] David M. Rind; Isaac S. Kohane; Peter Szolovits; Charles Safran, Henry C. Chueh, and G. Octo Barnett: "Maintaining the Confidentiality of Medical Records Shared over the Internet and the World Wide Web"; *Annals of Internal Medicine*, 15 July 1997. 127:138-141.

[Rindfleisch:97] Thomas C. Rindfleisch: Privacy, Information Technology, and Health Care; *Comm. ACM*; Vol.40 No. 8 , Aug.1997, pp.92-100.

[Sweeney:96] Latanya Sweeney: "Replacing Personally-identifying Information in Medical Records, the SCRUB System"; in Cimino, JJ, ed.: AMIA *Proceedings, Journal of the American Medical Informatics Association*. Washington, DC, Inc, 1996, pp.333-337.

[Sweeney:97] Latanya Sweeney: "Guaranteeing Anonymity When Sharing Medical Data, the DATAFLY System"; *Proceedings AMIA, Journal of the American Medical Informatics Association*, Washington 1997

[Ullman:97] Jeffrey Ullman: "Information Integration Using Logical Views; *International Conference on Database Theory (ICDT '97)* Delphi, Greece, ACM and IEEE Computer Society, 1997.

[WiederholdBD:98] Wiederhold, Gio, Michel Bilello, and Chris Donahue: "Web Implementation of a Security Mediator for Medical Databases"; in T.Y.Lin and Shelly Qian:*Database Security XI, Status and Prospects*, IFIP / Chapman & Hall, 1998, pp.60-72.

[WiederholdBSQ:96] Wiederhold, Gio, Michel Bilello, Vatsala Sarathy, and XiaoLei Qian: A Security Mediator for Health Care Information"; Proceedings AMIA, *Journal of the , Journal of the American Medical Informatics Association*, Oct. 1996, pp.120-124.

[WiederholdEa:96] Gio Wiederhold, Michel Bilello, Vatsala Sarathy, and XiaoLei Qian: "Protecting Collaboration"; *Proceedings of the NISSC'96*, Baltimore MD, Oct. 1996, pp.561-569.