2009

K-bass: A Knowledge–Based Access Security System For Medical Environments

George Vakaros
Informatics Lab, Faculty of Technology, Aristotle University of Thessaloniki, vakaros@arrow.com.gr

George Pangalos
Informatics Lab, Faculty of Technology, Aristotle University of Thessaloniki, pangalos@auth.gr

Arrow Technologies s.a

K-BASS: A KNOWLEDGE–BASED ACCESS SECURITY SYSTEM FOR MEDICAL ENVIRONMENTS

Vakaros, George, Informatics Lab, Faculty of Technology, Aristotle University of Thessaloniki, 54006, Greece, vakaros@arrow.com.gr
Pangalos, George, Informatics Lab, Faculty of Technology, Aristotle University of Thessaloniki, 54006, Greece, pangalos@auth.gr
Eleftheriadis, Viktor, Arrow Technologies s.a, I. Tsalouhidi 16-20, Thessaloniki, 54248, Greece, bikdarw@yahoo.gr

Abstract

Enforcing security requires the application of an access control model. The access control models used today have limitations that become evident when applied in collaborative environments, such as medical environments. To overcome these problems, a system has been developed in order to introduce dynamic access security. The system at hand combines effectively (C-TMAC) Team-based access control using contexts model and knowledge base technology. The system's security scheme fine-grains the users' access rights by integrating the Role Based Access Controls (RBAC) model and the (C-TMAC) model through knowledge-based systems technology. The originality lies on the fact that the users in the system are authenticated by combining their individual access rights (RBAC), their team's access rights (C-TMAC) and the context information associated with the team they belong to.

Furthermore, knowledge-based technology is used for the representation of knowledge and reasoning. The system initiates with some facts and rules and is able to learn, infer knowledge and produce meta-knowledge. Therefore the system can train itself and respond in non-deterministic way to user requests. Any change in context information fires a new rule in the knowledge base. The proposed system is an automated and self-controlled system called (K-BASS) Knowledge-based Access Security System that may be used in medical environments, to dynamically assign permission rights and to add new medical staff and patients.

Keywords: *Knowledge-based systems, K-BASS, TMAC, C-TMAC, Meta-Knowledge, IDS*

1 INTRODUCTION

Fast and reliable access to information is becoming one of the major factors that decide on the success of a medical system. Most access control models that are available today are of a relatively static nature and make it difficult to express access control requirements that are dependent on time or the occurance of events. Especially in the medical domain, the value of information protection is very high, and on the other hand medical security systems need to be flexible. Systems without any type of learning mechanisms need someone to expand and refine their knowledge on a regular basis, otherwise systems become out of date and useless. Knowledge acquisition mechanisms will make systems more effective and powerful.

The most widely used access control nowadays is Role Based Access Control (RBAC). At first it was integrated in database systems, but is now used in many applications, including Microsoft Windows platform [Dieter Gollmann, 2003]. RBAC has been so popular because of its core concept: roles. Assigning the organisation roles to the roles of the access control system is an easy and relatively effortless task for any system security designer. However RBAC has certain limitations.

In an effort to overcome RBAC's limitations, other access control models have been designed including Team-based Access Controls (TMAC) [Roshan, 1997], C-TMAC [Georgiadis et al.,2001] and SESAME [Zhang, Parashar, 2003]. In this work, the hybrid access control model C-TMAC is used as proposed in [Georgiadis et al.,2001]. C-TMAC has been preferred over other models, because it produces the fine-

grained access control policy that is needed in any collaborative environment. C-TMAC refines the permissions granted by RBAC, to ensure that only authorised users will be able to modify patients' medical records.

The objective of this study is to design and implement a system for any medical environment that will be able to be rule learning and dynamic. K-BASS uses knowledge-base technology and has meta-knowledge which allows it to learn and thus to be dynamic and non-deterministic. Furthermore, to enforce the strong access control policy necessary in medical environments, K-BASS incorporates context information. The context information used up to this point of research is the time and place of request and the team that the requestor belongs to. Granting or denying the access to the requestor is the combination of all of the above.

2 BACKGROUND

2.1 Role Based Access Control (RBAC)

To implement RBAC three concepts have to be defined: Users, Roles and Sessions. Every user has a role that may not be unique. Roles and Users are assigned to Sessions. Roles are but a representation and a grouping of access rights. Two Users that have the same Role, also share the same access rights. RBAC focuses on users and on the jobs they perform [Dieter Gollmann, 2003]. The administration of an RBAC system is less complex, since the granting and the revocation of a user's access rights is only a matter of changing the user's role. Additionally in RBAC the internal hierarchy that is formed is quite similar to the hierarchy found in the organization itself.

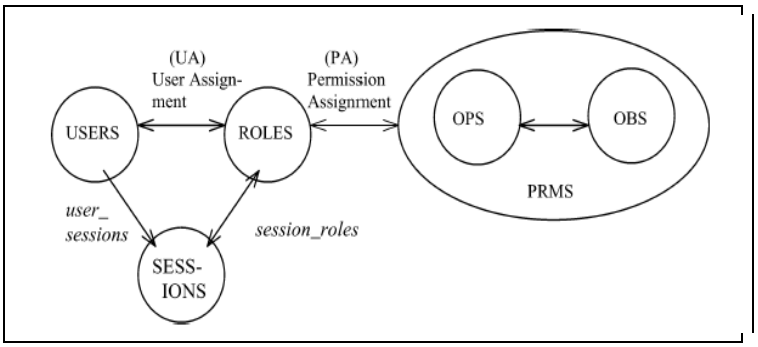

Figure 51. RBAC

In Figure 1, a schematic representation of RBAC can be found. RBAC includes sets of five basic data elements called users (**USERS**), roles (**ROLES**), objects (**OBS**), operations (**OPS**), and permissions (**PRMS**). The RBAC model as a whole is fundamentally defined in terms of individual users being assigned to roles and permissions being assigned to roles. As such, a role is a means for naming many-to-many relationships among individual users and permissions. In addition, the RBAC model includes a set of sessions (**SESSIONS**) where each session is a mapping between a user and an activated subset of roles that are assigned to the user. Permission is an approval to perform an operation on one or more RBAC protected objects. An operation is an executable image of a program, which upon invocation executes some function for the user. An object is an entity that contains or receives information. The function session roles gives us the roles activated by the session and the function user sessions gives us the set of sessions that are associated with a user. The permissions available to the user are the permissions assigned to the roles that are activated across all the user's sessions.

Nevertheless, RBAC is inadequate for medical systems. Two limitations stand out when applied to a collaborative environment; the first one being that RBAC is a passive model. Its outcome relies on static rules that are imposed on groups and users, resulting in passive access control. Even though the static

rules may change, RBAC lacks the ability to be dynamic and flexible to the changing conditions of everyday reality in a collaborative environment, such as a medical environment.

The second limitation when using RBAC in collaborative environments is that RBAC assigns permissions to groups of users and not per case. When using RBAC, all doctors will have the same permissions over the patients' medical records, regardless of whether they are treating them or not. This is a highly undesirable vulnerability when enforcing security in sensitive data, such as patients' records.

2.2 Team-based access control using contexts (C-TMAC)

C-TMAC is based on TMAC that was originally proposed by Thomas [Roshan, 1997]. He was the first one to recognise the importance of the context information associated with collaborative tasks and the ability to apply this context to decisions regarding permission activation. The collaboration context of a team contains two pieces: the user context, which could be the current members (users) of a team, and the object context, which could be the set of object instances required by the team to accomplish its task. TMAC allows us to create a general structure (class/definition) of a team with role-based permission assignments to object-types. However, when a team is instantiated, the user context can be used to tailor the role-based permissions defined on object types to user-specific permissions on individual object instances considered to be part of a team's resources. By aligning access control to the metaphor of teams, TMAC can provide a paradigm for access control that is natural and non-intrusive to the way users work in collaborative environments. In TMAC users are assigned roles and teams to best implement access control policy.

C-TMAC is based on the integration of RBAC and TMAC. It provides a framework to integrate TMAC concepts with RBAC and it also establishes the use of other context information such as the time of access, the location from where access is attempted, the location the object that is to be accessed resides in, transaction-specific values that dictate special access policies, etc.

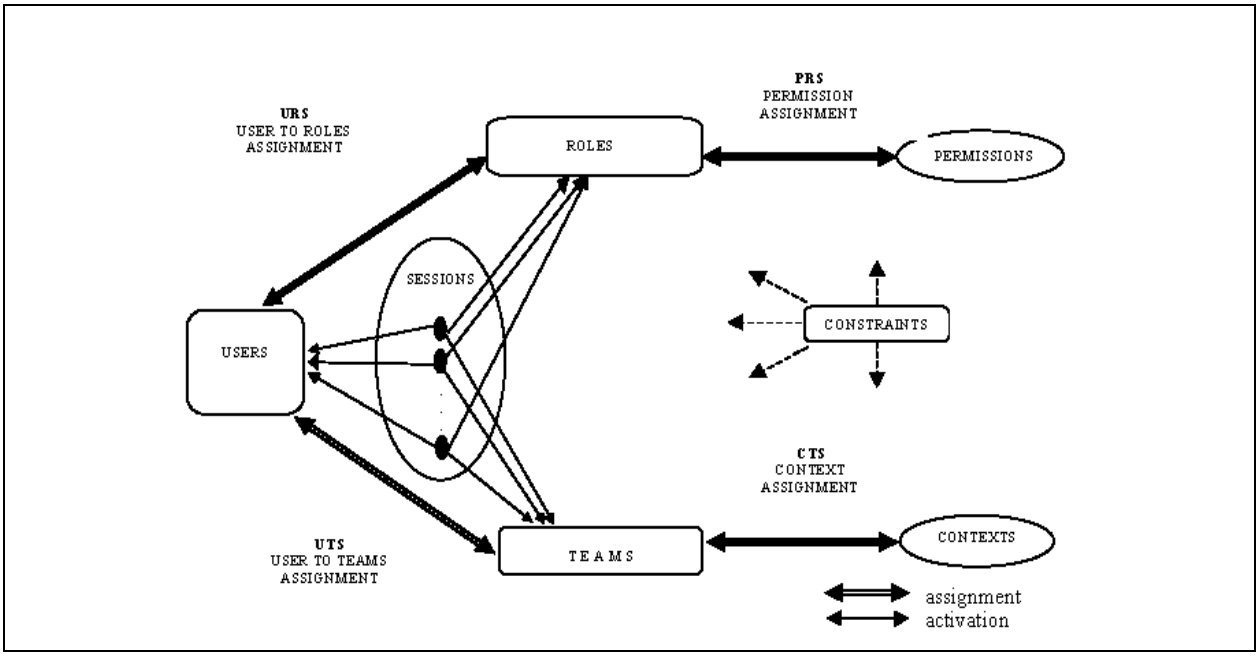

Figure 52. C-TMAC

C-TMAC is outlined in Figure 2. In this approach every user is assigned to one or more roles, as in RBAC but his permission rights are fine-grained by the team he belongs to. Context information is imposed for every team and defines the access rights permissions for that team.

In C-TMAC, a user may have one or more roles and may also be part of one or more teams. In the case that the user is acting alone he may be considered as a team of only one person. C-TMAC consists of five entities, users, roles, teams, contexts and permissions and a collection of sessions [Ferraiolo et al.2001]. The use of a team as an intermediary to enable a user to obtain a context is similar to the use of roles as an intermediary between users and permissions. An important property of a Session is that the user associated with a session, cannot change. The association remains constant for the life of a session.

3 DESIGN

K-BASS consists of two modules, the interface and the kernel. 'Interface' refers to the front – end used by the user to interact with the system. Interface presents data to user and uses the typed in data to interact with the knowledge base and the working memory. It does not insert, update or modify data and knowledge itself; this is a task of the kernel module.

As we see in Figure 3, the kernel module consists of the knowledge base, the inference engine and the working memory. The **knowledge base** [Lakner, 2004] contains the domain knowledge useful for problem solving. The knowledge is represented as a set of rules. Each rule specifies a relation, recommendation, directive, strategy or heuristic and has the IF (condition) THEN (action) structure. When the condition part of a rule is satisfied, the rule is said to ***fire*** and the action part is executed. The **working memory** includes a set of frames and instances used to match against the IF (condition) parts of rules stored in the knowledge base. The **inference engine** carries out the reasoning whereby K-BASS reaches a solution. It links the production rules given in the knowledge base with the procedural knowledge provided in the working memory. The **explanation facilities** enable the user to ask the system ***how*** a particular conclusion is reached and ***why*** a specific fact is needed. The **knowledge acquisition facilities** [Hasnah, 1994] check the consistency of the knowledge base (verification, validation) and extract knowledge.

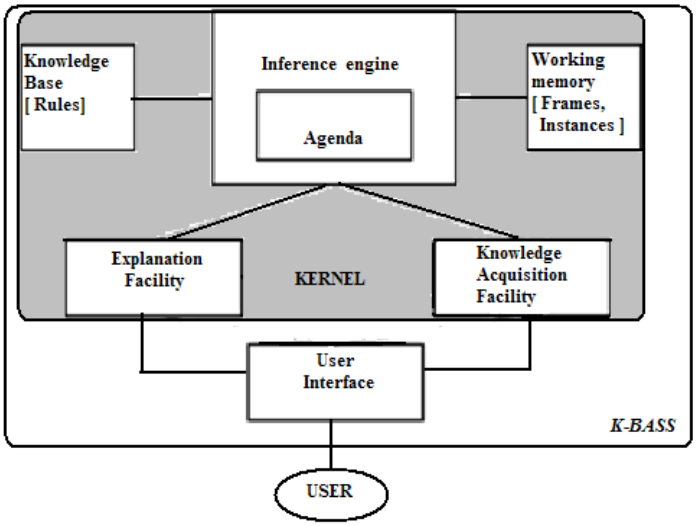

Figure 3. K-BASS Design

In Figure 4, we see the cycle of K-BASS. It finds the rules that are eligible to be fired by matching left-hand side of the rules to the procedural knowledge in the working memory (pattern matching) and then it selects which rule to fire (conflict resolution).

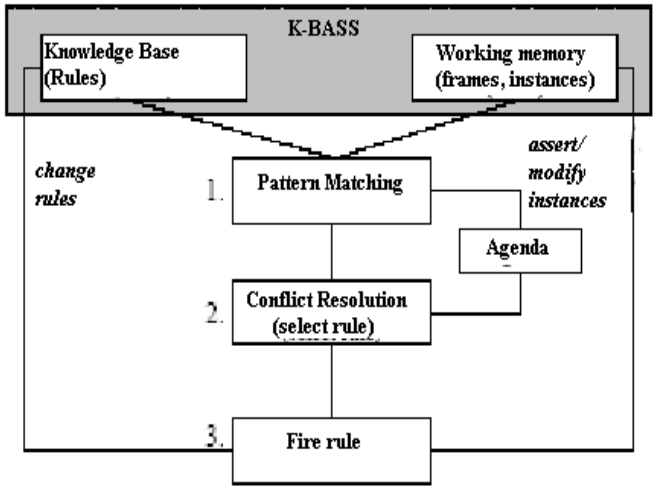

Figure 4.K-BASS cycle

The inference engine of K-BASS follows the forward chaining strategy [Turban, Aronson, 2001] and matches the datas in working memory against 'conditions' of the production rules in the knowledge base. When a rule fires, this is liable to produce more data and the cycle continues as we see in Figure 5.

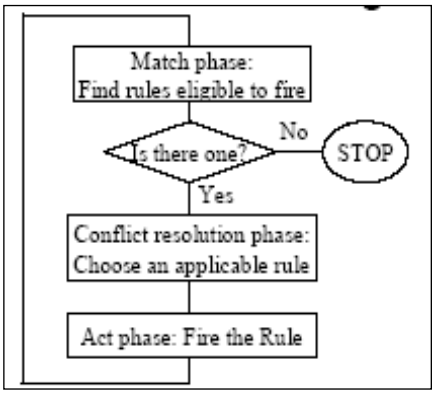

Figure 5. Forward chaining

A production rule example of the inference engine of K-BASS is:

```
{
rule_access_positive        /*(read only rights)*/
 if   the identity`s name is nurse and the id_number is less than 8040 and time(H,M,S,T)
 then echo('The system welcomes its.....', identity`s name)
},
```

that checks the identity of a nurse.

A frame is similar to an object and is a complex data structure, which provides a useful way of modelling-world data objects. Frames are analogous to records within a database but far more powerful and expressive. Each individual frame has a name by which it is referred, details of its parent(s) frame, and a collection of slots or attributes, which will contain values or pointers to values. Slot values can be explicitly defined locally, or implicitly inherited from an ancestor frame further up the

hierarchy [Vasey, 1989]. Frame hierarchies are similar to object-oriented hierarchies that allow knowledge to be stored in an abstract manner within a nested hierarchy with common properties automatically inherited through the hierarchy. This avoids the unnecessary duplication of information, simplifies code and provides a more readable and maintainable system [Pangalos et al., 2004]. Each frame has a set of slots (which are analogous to fields within records, except that their expressive power is greatly extended) that contain attributes describing the frame's characteristics.

In K – BASS four entities have been defined: *Role, Team, User* and *Patients*. Every entity has been implemented as a frame. The Role frame consists of the medical, ward and admin instances that inherit the attributes of Role. Likewise, medical consists of head_doctor, special_doctor and physician. The frame hierarchy for the entity Role can be found in Figure 6.

Apart from frames the working memory also contains instances. Instances are used to assign objects to frames. Instances respectively inherit the attributes of parent frame. Instances are used in K-BASS to define medical staff entities and users.

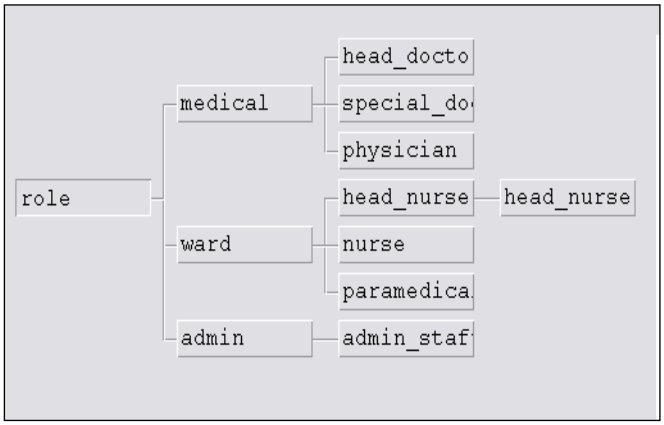

Figure 6. Frame Role

The core of the K – BASS system is production rules in (if-then) form. Rules, a part of the knowledge base, are the means used to assign permission rights to objects. Rules define what a member of the staff can or cannot do.

Nevertheless, rules are not static information. They can be updated [Dhamija, Tygar, 2005]. The knowledge base itself has the ability to be updated due to meta-rules that it has, resulting in the **learning** of the system (Figure 7). The meta-rules alter the reasoning process and determine the strategy for the use of task specific rules that make K-BASS **dynamic** and flexible. The inference engine initiates the update of rules [Buchanan,Duda, 1982] and knowledge base. K – BASS has the ability to interact with the user and use data and knowledge to produce meta–knowledge [Garnier] and expand its knowledge. The meta-knowledge is the knowledge about how the system reasons, is knowledge about the use and control of domain knowledge in K-BASS and it allows the system to examine the operation of the declarative and procedural knowledge in the knowledge base. Over time, metaknowledge will allow K-BASS to create the rationale behind individual rules by reasoning from first principles. Every time a new rule is created, it is asserted to the knowledge base and becomes part of the system's rules.

```
Sequential-Covering(Target_attribute, Attributes, Examples, Threshold)
Learned_rules <-- { }
Rule <-- Learn-one-rule(Target_attribute, Attributes, Examples)
While Performance(Rule, Examples) > Threshold, do
Learned_rules <-- Learned_rules + Rule
```

Examples <-- Examples -{examples correctly classified by Rule}

Rule <-- Learn-one-rule(Target_attribute, Attributes, Examples)

Learned_rules <-- sort Learned_rules according to Performance over Examples
Return Learned_rules

Figure 7. Meta-rule for rule learning

4 METHODOLOGY

The methodology used in K-BASS is divided in procedures and scenarios. Procedures are used to describe the actions that need to be taken by the system to interact with the users and meet their demands. In other words, procedures include interface's and kernel's mechanism. Scenarios on the other hand, represent the internal mechanism implemented in K-BASS. Respectively, scenarios are kernel's mechanism.

4.1 Procedures

In this section the login procedure is presented. The procedure initiates with the typing of the system's password that is used to ensure that the users are authenticated entities of the system. Should the password be typed wrong thrice, the users are locked out from their account and the system. In case the system password is correct, they are welcomed and the kernel engine verifies their status.

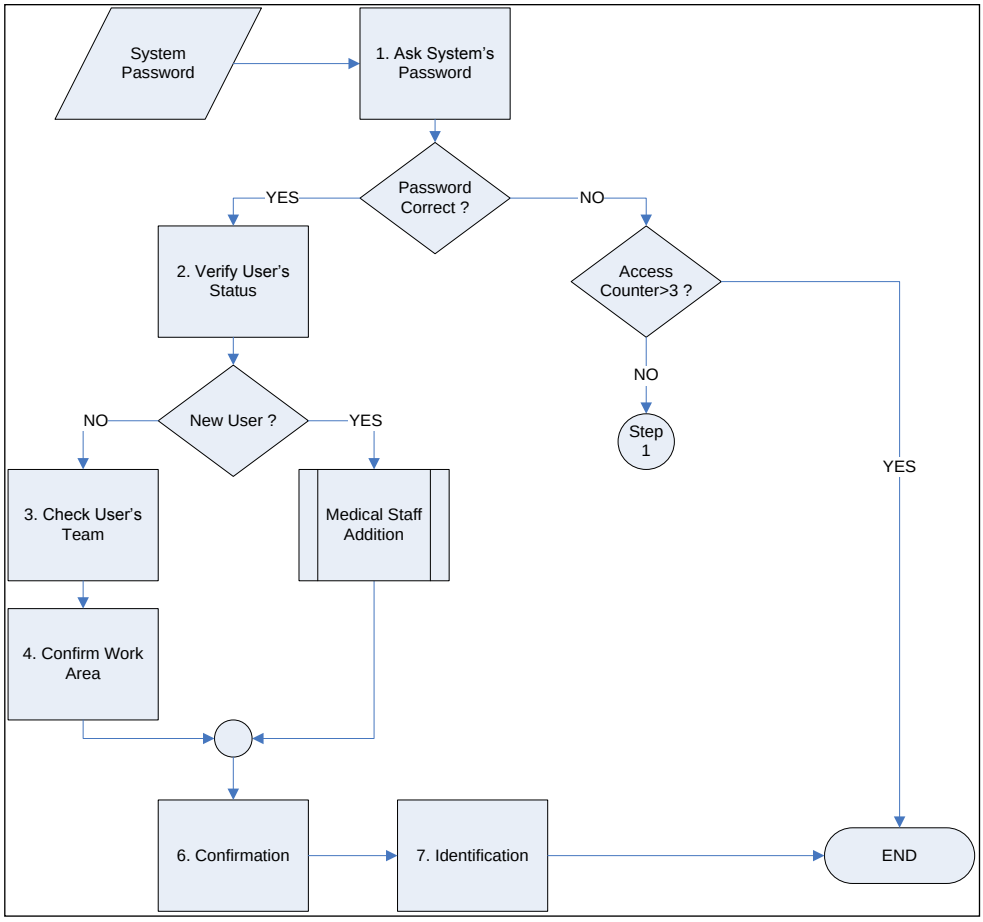

Figure 8. Login

Two cases can be established when verifying users' status, the addition of new users or the login of already registered users. If the users are new, the system asks for the users ' detailsand the kernel carries on with the procedure described in section 4.2.1. If the users are registered, system retrieves users' team and confirms their work area. Both attributes are necessary to define their access rights. Afterwards the confirmation and identification rulesets are triggered and users log in to the system and are granted access to patients' medical records.

It has to be noted that users have no access to the system up or to patients' files up to this point. The login procedure is outlined in Figure 8.

4.2 Scenarios

4.2.1 Medical Staff Addition

As already mentioned, scenarios are used to describe the actions taken by the kernel engine. The scenario discussed in this section refers only to the addition of medical staff; it does not apply when adding patients to the system. The later scenario is described in Section 4.2.2.

Upon adding medical staff to K-BASS knowledge base (Figure 9), the system initiates by verifying users' status by interacting with the knowledge base of the kernel engine. Once it has been verified that the users are new to system, the interface prompts users' for Name, Profession, Surgery, Specialty and Responsibility. The kernel engine processes the information provided by the users. If the information is correct, the inference engine generates a new instance of user in the system. The system restarts and the users can login with their credentials, user name and password.

For new users to login in the system, they have to be informed of the system's password. This can be given to them when employed. Naturally the system's password has to be updated in regular, predefined intervals to protect from information leakage. The use of this password guarantees that only legitimate users can interact with the system. It is also a countermeasure in the case of identity theft.

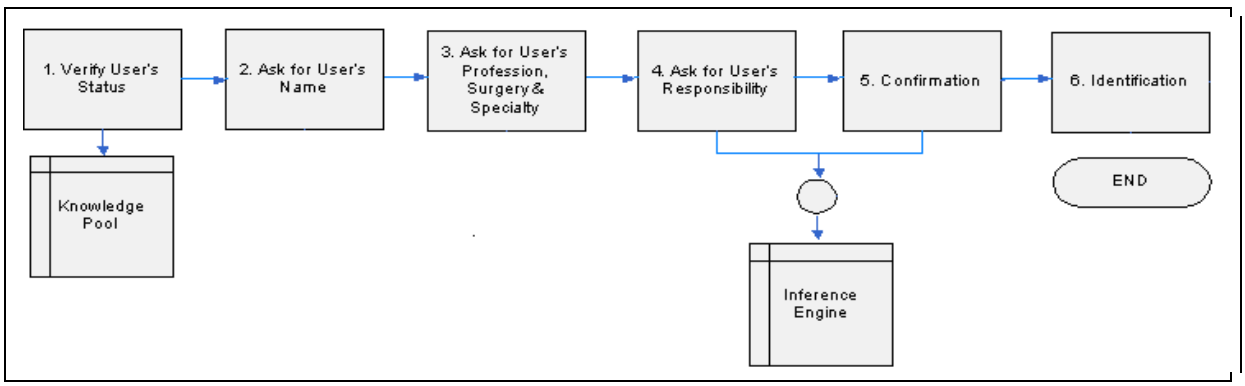

Figure 9. Medical Staff Addition

4.2.2 View / Insert / Modify Patients' Medical Records

This section describes (as we see in Figure 10) the actions taken by the system to View/Insert/Modify Patient's Medical Records. This scenario assumes that the users have already been authenticated. Users choose the action they want to take on patients' medical records [Cody, 2003]. Once they have entered their choice, the kernel engine verifies their permission rights against the knowledge base. The possible actions are: viewing patients' records, modifying them or adding new ones.

In the case the users wish to view medical records they are retrieved from the knowledge base and they are presented to the user. In case the users need to modify existent records, the knowledge base

retrieves them and allows them to edit them. If the users wish to add new record in patients' files, the inference engine is initiated and the knowledge base is updated.

In addition, users who are registered to the system, have the right to enter new rules, due to the dynamic rule learning nature of it and the metaknowledge attributes that has the K-BASS' s kernel. These rules are inserted into the knowledge base, they become part of it and in that way the system's workflow changes partially.

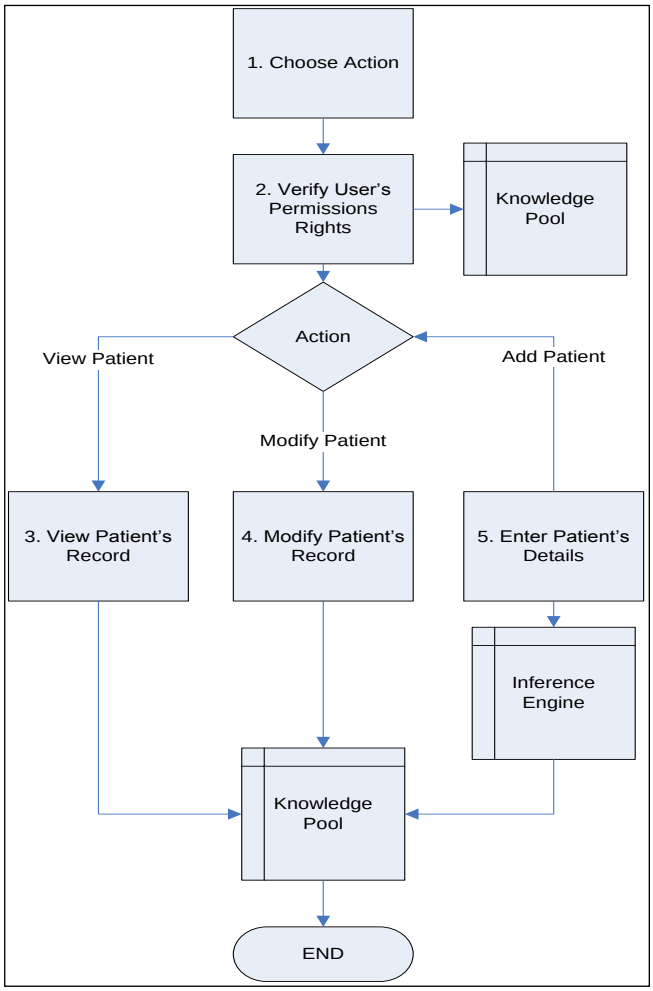

Figure 10. View/Insert/Modify Patient

5 APPLICATION

The use of knowledge-based systems guarantees that the system will be able to enforce existing access rules and expand its knowledge base. The decision is reached in a non-deterministic way based on facts and rules that lie on the evolving knowledge base. Dynamicity in the area of access control is implemented through the use of context information. Any change in context information fire a new rule in the knowledge base.

We use Rule-Based representation techniques because they are popular for storage and manipulation of domain knowledge. Two important reasons for this popularity are 1] the modularity of the rule-based framework and 2] the ability to use knowledge stored as rules in a non-procedural manner. Rule-Based analysis relies on sets of predefined rules that are provided by an administrator, automatically created

by the system, or both. The early access control research efforts realized the inefficiency of any approach that required a manual review of a system audit trail. While the information necessary to identify attacks was believed to be present within the voluminous audit data, an effective review of the material required the use of an automated system.

To design K-BASS and to define the necessary roles, the eMEDAC model has been used. The structure and the properties of the medical staff have been derived with this model [Mavridis et al., 1999]. K-BASS has been programmed with a hybrid intelligent system, which is expressive and powerful and supports frame-based reasoning with inheritance, rule-based programming and data driven procedures fully integrated within a logic-programming environment.

The context information that is associated with the teams is information about the location and time. For every team the locations the user could be in are included in the context information as well as the time slots the user can make such a request. This information is cross-checked before the user is allowed access to the system. In K-BASS a legitimate user will not be granted access if the request takes place when the system is aware of the fact that the user should not currently be working or if the request is originated from a place outside his predefined working space. This is extremely important since it ensures that even if identity theft occurs, access to sensitive data additionally requires illegal physical entry. Furthermore, in such cases the system acts as an intrusion detection system (IDS).

Initially the user needs to be authenticated. To do so he needs to enter the system's password, his username, his id number and his working area. The system examines his credentials and assigns him to the corresponding team. An example of a dialog box that the user is presented during the authentication is presented in Figure 11.

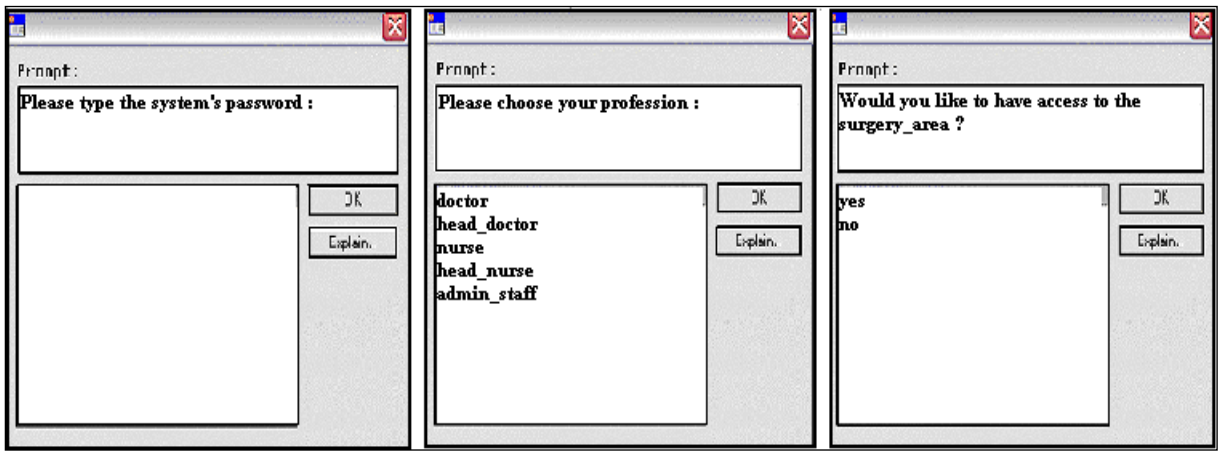

Figure 11. Dialog boxes of K-BASS

In the effort of making K-BASS able to detect intrusion attempts in the system the ability of keeping log files has been introduced. Every time a user logs in a record is being kept with all session details, including the time and the place of the login.

The system accepts data that the users insert, but in order to make them part of its dynamic working memory, a dynamic control has to be done. This happens with an interactive procedure (with the form of questions and answers) between the users and the system. When new users try to register to the system, their access rights are being given to them after their successful authentication. At this point, take place the attributes of meta-knowledge, and the users can insert rules and entities to the system's kernel, influencing its function.

read_and_modify rights to all of the data. The users' teams work as filters for the access to the patients' datas. For example, two doctors with the same access rights may differ, as the one doctor may

modify the datas of a particular patient and the other may not modify the datas of the same patient. In Figure 12, we can see an example of read-only rights to a patient's datas.

CURRENT PATIENT		
1]-----	The name is :	xara_konti
2]-----	The day of insertion is :	22/12/2005
3]-----	The day of arrival is :	29/12/2005
4]-----	His/her location is :	[surgery_area,hall _2]

Figure 12. Read_only_rights

6 RELATED AND FUTURE WORK

According to Clancey and Shortley (1984), the medical artificial intelligence is related with the construction of AI systems that help the application of diagnosis and make therapy propositions. Many medical systems are based on other programming methods, like on simple statistical and probabilistic methods. In addition, the medical AI systems are based on symbolic models of patients' entities and they are related with the factors of the patients and the clinical data.

The first ten years of AI in medicine, most systems or research were developed to help the medical staff with the procedure of diagnosis. Most of these systems were not developed more than the phase of the laboratory research. But some other systems continued to be developed and they were transformed into important parts of educational systems. They follow examples of expert medical systems:

MYCIN: At his moment, is probably the most well known medical expert system that has ever been developed (Shortliffe, 1976), although it has not been posed in real practice till now. It was developed in the University of Standford as a research effort to provide help to the medical staff with the diagnosis.

PUFF: It was developed in 1979, using the shell **EMYCIN**. Its purpose is to interprete measurements and to recognise pneumonic disorders. It is used in routine basis.

HELP: It is a medical information system that is based on knowledge and it was developed in 1980. It provides to the medical staff warnings and recommendations, explanations of medical datas, configuration of the patients' datas and clinical protocols. It is used in 6 big hospitals in Utah and in a lot of locations in USA.

PEIRS: It is an interpreting research system, which was used till 1994, and it was interpreting about 80-100 references per day with a diagnostic accurancy of 95%.

Today, a few number of medical research systems are being performed from medical staff and academic staff. As the medical expert systems are a real power and not just an academic idea, we must work for the constant establishment of their technology in the medical sector, in general.

K-BASS is an automated and self-controlled system used to dynamically assign permission rights and to add new medical staff and patient information. This is accomplished via symbolic processing and non-deterministic reasoning (that AI offers us), as we have to do with entities that give us new entities and new rules, and not with numeric values. In this way the system learns with the use of meta-knowledge. It interacts with users to authenticate them and grants them access rights according to C-TMAC. Furthermore, it ensures that only the doctors treating patients can modify their medical records. K-BASS can also learn and expand its knowledge base by adding new medical staff and patient information.

Currently experiments are conducted to study the behaviour of K-BASS when adding more context information. At this point, location and time are the constraints associated with the teams. The next step in this research area is to conduct experiments to evaluate the error rate of the system and retrieve values. The experiments will examine how the change of context information affects the system and what impact it has on the system's overall performance. Future work includes adding IP address to deal with mobile workstations and defining special access rights for medical staff previously treating a patient.

References

Ahmad Hasnah (1994) IEEE. Knowledge Acquisition for Computer-Based Medical Expert Systems.

Brian J.Garnier. Meta-Knowledge Acquisition Strategies in Asynchronous Learning Frameworks.

Bruce G. Buchanan and Richard O.Duda (August 1982). Principles of Rule-Based Expert Systems, M. Yovits (ed.) Advances in Computers, Vol.22, New York: Academic Press.

David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn, Ramaswamy Chandramouli (August 2001), Proposed NIST Standard for Role-Based Access Control, ACM Transactions on Information and System Security, Vol. 4, No. 3, Pages 224–274.

Dhamija Rachna, J.D.Tygar (2005). The Battle Against Phishing: Dynamic Security Skins.

Dieter Gollmann (2003). Computer Security, John Wiley and Sons.

Efraim Turban and Jay E. Aronson (2001). Decision Support Systems and Intelligent Systems, 6th ed, Prentice Hall, Upper Saddle River, NJ.

Georgiadis C., Mavridis I., Pangalos G. and Thomas R. (2001). Flexible Team-based Access Control Using Contexts, Proceedings of the Sixth ACM Symposium on Access Control Models and Technologies (SACMAT2001), USA.

Guangsen Zhang, Manish Parashar (2003). Dynamic Context-aware Access Control for Grid Applications., Fourth International Workshop on Grid Computing, 11 17 - 11, Phoenix, Arizona.

G.Pangalos, G.Vakaros, Ch. Georgiadis, I. Nestori, K.Kemalis SETN (2004). Dynamic Access Control Management Using Expert System Technology, Third Hellenic Conference on Artificial Intelligence.

Mavridis I., Pangalos G. and Khair M. eMEDAC (1999): Role-based Access Control Supporting Discretionary and Mandatory Features, in Proceedings of 13th IFIP WG 11.3 Working Conference on Database Security, Seattle, Washington, USA.

Patrick M. Cody (August 28, 2003). Dynamic Security for Medical Record Sharing.

Phil Vasey (April 1989). Flex Expert System Toolkit, Version 1.2.

Roshan K. Thomas. Team – based Control (TMAC) (1997): A primitive for Applying Role-based Access Controls in Collaborative Enviroments, Proceedings of the Second ACM Workshop on Role-based Access Control, Fairfax, VA USA.

Rozália Lakner, University of Veszprém (2004), Department of Computer Science. Knowledge-based systems, Engineering Applicationof AI.