ACCESS CONTROL, AUTHENTICATION AND MONITORING IN MASARYK UNIVERSITY COMPUTER NETWORK

Jan Konečný[1], Martin Osovský[2], Michal Ordelt[3]

[1]Institute of computer science, Masaryk University, Botanicka 68a Brno, konecny@ics.muni.cz,
[2]Institute of computer science, Masaryk University, Botanicka 68a Brno, osovsky@ics.muni.cz,
[3]Institute of computer science, Masaryk University, Botanicka 68a Brno, ordelt@ics.muni.cz

Keywords:
Identities, authentication, authorization, monitoring

1. EXECUTIVE SUMMARY

1.1. Background

The system described in this paper was implemented for Masaryk University Brno. It serves the purposes of authorization, authentication and monitoring for a wide variety of software services used at the university. It's was made out of the need to have an unified system like this for the sake of greater security and maintainability. It is based on continual actualization of personal and other data in the common data model based on data and relation stored in particular external information systems (Human Resources Management, Study Evidence, other databases). The data from the common model are distributed to other services and systems through a well defined interface. Everything the system and the concurrent services do is monitored and logged, so that errors can be fixed and the services can be better administered.

1.2 Alternatives

At the time of creation, there were no real alternatives for such a system and there still are no alternatives for some parts (like the integrated monitoring). The automatic creation of user accounts can be done using other methods like those based on Microsoft Identity Integration Server but our system is better suited for the university and requires less programming and configuration for the end users.

1.3 Conclusions

We created a system that serves its purpose very well and has a good potential of evolving and being offered and then implemented in other environments. It enables us to unify authentication and authorization services at the university and to make a lot of other services more usable and user friendly. It also has some problems but these can be fixes in the future when the system is extended and made more flexible and usable.

2. INTRODUCTION

In this article we would like to describe the solution for access control, monitoring and other similar tasks that we have implemented in the computer studies environment at Masaryk University. We have been implementing and improving this system over several years and it evolved from a relatively simple form to a much more general and complex system with many usages. This however sometimes means that the architecture of some parts of the system is somewhat less flexible then it would have been, if we had designed these usages in mind. In the article we first describe the needs that led us to developing the system, than in short the solution itself, the benefits of the system, some problems and what we would like to do in the future.

3. NEEDS FOR THE SOLUTION

Before we implemented the new system our infrastructure was relatively small and there was no real need to have a sophisticated access control and monitoring system. So we just imported the user data from personal databases and recreated all user accounts each night and we monitored just a few things. Then three years ago we were forced to revise this simple yet functional system, because a bunch of new needs arose. It was the massive extension of the infrastructure form one computer study with approximately 80 computers to more than twenty rooms with few hundred computers that have very different access control and monitoring requirements. Another impulse was the need to integrate and unify access control to a lot of services at the whole university not only the originally supported student workstations authentication (like federated services, various web pages, library system and so on) and to extend it for use of persons extern to the university that are not present in the original personal databases. So we needed a system that would collect all the information about persons in various source databases (study database, human resources database, guests database and so on) and convert it into a common data model later reflect changes without need of recreating the data every time and to convert the common data into specific authentication and authorization data for a unrestricted number of services and systems (like different physical access control systems, active directory domains, federation identity provider and so on) with the same automatic change reflection capabilities. These requirements were connected with the need to completely monitor the whole system to log all sorts of events (like accesses through the physical control, logon events on workstations or web pages and so on) top find out errors and comfortably administer all the services.

4. THE SOLUTION

The name of the system is AccMgr (Account Manager) and its purpose is simple – to keep up to date data in all extern service databases (we call the interface between the system and these services a replication module). It is implemented in the .NET framework 3.5 (most of it is written in the C# programming language). All the data is stored in an Oracle database and the communication extensively uses SOAP web services.

The data originate from several extern sources; we will call it the extern data. To achieve this goal AccMgr stores its version of data, we will call them the common data. The common data consist of the actual data (or at least the data that were actual when AccMgr was last executes) and various flags used for change tracking and executing replication actions.

The normal execution of AccMgr consists of two parts: the propagation and the replication. The propagation compares extern data with common data, marks changes accordingly and actualizes the common data. The main entity for the propagation process is a Person – it contains personal information such as surname, academic titles and so on, list of studies with their states, list of

active employments and list of active university chip cards. The person is created from extern data and compared with the corresponding records in the common data, every item being compared (chip card number for instance) is marked as valid in this run. If there is a change in data, the Person object and that data is marked as changed. Then all conditions for access privileges are evaluated and the result is compared with stored state. Again, if there is a difference, it is marked. Finally all current access privilege bans are checked, whether they have expired. If person has no active study or employment, no change in the common data is made.

After checking all persons the new values of AccMgr data are committed to the common data database. This 'atomicity' of propagation is needed because some extern systems hold only active data , so the inactivation of value is detected by the fact, that it is not marked as been propagated in this execution run.

The replication reflects changes to extern system. It is divided to 3 steps. The steps have common scenario, the objects which have any changes are passed to replication modules along with the required replication operation. The available operations are: activate, delete, change, create, none. The operation is selected based on the propagation status of given object. In the first step inactive chip cards are replicated, in second step the inactive organizaton units (workplace, field of study), in the last, most important step, the system replicates persons with changes in data or in access privileges. If person is not being propagated in this run, but was propagated in last run, it is considered to be inactive and as is replicated with the delete replication action.

There is also functionality for defining additional replication actions for persons. Each additional action has its own condition. The additional action can also have undo action. So long there is only one pair – compression of a user roaming profile directory on the profile server - with the condition that the person was inactive for more than 14 days. The undo action is of course decompression with condition that the person is currently being activated and its user profile directory was compressed.

Replication modules perform requested operation and return values that indicate whether their operations were successful. If all replication modules replicated successfully, the values in the common data are marked as replicated in this run and the change mark are cleaned. Otherwise the object is replicated during the next run. There is no problem to add another replication module, it is just an object that implements an abstract replication module interface and fulfils some documented conditions.

As an example of a replication module and extern service we will describe the Physical Access Control system based on the Honeywell Access Systems' WIN-PAK system. The system itself consist of a lot of physical entities like chip card readers, turnstiles, doors with magnetic locks all driven by a so called controller. When a person wants to enter a study, he or she puts his university chip card on the card reader. The reader reads then the unique identification number stored on the card's chip and sends it to the controller. The controller checks its internal database for that number and if it's present it allows the person to enter by sending an electric signal to the turnstile or door lock. It also sends an xml message to the monitoring system indicating that an access check action occurred in the controller and the result of the action, the message is stored in the database and sent to various other receivers to give feedback to the entering person (a monitor near the turnstile) and to inform study operators (a monitor in their office).

The controller is connected via a local LAN network to a server that receives and resends the monitoring messages, is capable of controlling the controller and through it the doors and turnstiles programmatically and. There is also a copy of the controllers' local database of chip cards and a configuration database of all controllers in the system.

All controllers are connected to the AccMgr system. As stated above, all controllers need a database of cards and access rights so that it could decide whether to grant access to a particular person (or more precisely card). The actual access data are generated by AccMgr via a replication module. It is a class called `PACReplicationModule` that basically uses two methods of the `IReplicationModule` interface - `ReplicateCard` and `ReplicatePerson` (the interface

contains more methods to replicate other entities managed by the system). First all inactive cards are deleted from the system by the first method (it's called for each card with action delete), then each person is updated in the system (it's name, surname, card numbers, access privileges and bans are mapped to corresponding data in the controller's database) by the second with possibly all replication actions. The updating itself is done either by calling a COM API supplied with the system or by updating the database directly. The most important part is that the only the module knows how exactly to map the common data to its specific records and can be configured and reconfigured to do so without the need for the AccMgr system to know about it.

Other thing to mention is the monitoring action that the controller (or to be completely honest – the server) performs – every other entity connected to the system does the same thing (windows workstations, doors and so on). There is a central monitoring server that collects such messages and stores them to a database in a way determined by their contents. It's possible to connect almost any monitoring service to this server, it must only be capable to send a message via RPC call or SOAP web service.

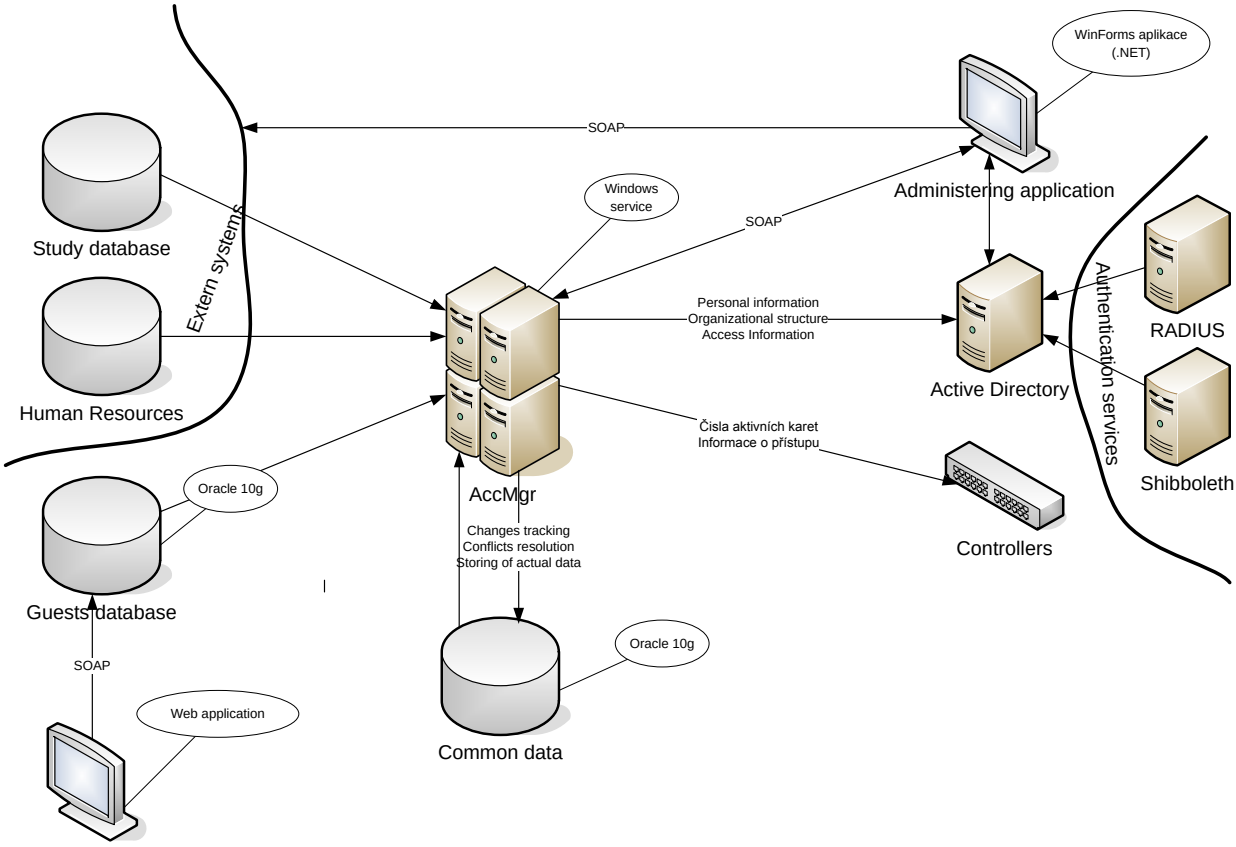

Figure 1Current state of the system (simplified)

5. BENEFITS OF THE SOLUTION

In the current state the system enables us to provide authentication and authorization services for a lot of very different systems, currently it is an Active Directory domain, A Shibboleth identity provider, the University Virtual Private Network and Eduroam, Library system and Physical Access

Systems some connected with safeguard system. All the data is guaranteed to be actual and most importantly no access is possibly granted to a person that no longer fulfills a condition (like having an active study or job at the university, it can also be belonging to a particular department or studying a particular field of study) or when a time restriction expires. It is easy for the administrators of the services to configure the accesses and also to define the algorithm for converting the personal data to the service specific format (mostly a different database or directory service), so it is very easy to use the system for a completely new service. We also provide a set of tools (graphical user interface applications and web pages) for comfortable configuration and monitoring of the services. One of the greatest benefits is the common data model that removes possible inconsistencies and superfluous dependencies in and between the source systems.

6. PROBLEMS

There are also some problems connected with the current solution. All the data is updated in the batch fashion, so it is actualized only every a day or so. It enables the system to be completely independent of its data sources but it brings also nontrivial problems. The common data model was originally designed for use only at the university, so its structure is less general than it should be if we wanted to use the system in a completely different environment. Also the performance is not as good as we would like it to be.

7. FUTURE WORK

In the future we would like to further generalize the common data model for use at different universities and companies. We would also like to make the actual data retrieval more flexible in particular to enable to reflect changes immediately (but to leave the batch data processing in the system) which could also lead to much better performance (we are usually processing data of approximately 50 000 people). And we would like to connect all relevant services used at the university to the system to make it better by implementing various users' needs and requirements.