EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH
CERN — BEAMS DEPARTMENT

CERN-BE-Note-2010-011 (CO)

Performance of the CMW-RBAC at CERN

Ilia Yastrebov

Abstract

The distributed control system of the LHC presents many challenges due to its inherent heterogeneity and highly dynamic nature. One critical challenge is providing access control guarantees within Controls Middleware (CMW). Role-based access control (RBAC) project at CERN was designed to protect from accidental and unauthorized access to the LHC and injector equipment. CMW-RBAC library is deployed as a part of every device server that imposes very tight constraints in terms of performance, stability and security. In this paper we present comprehensive performance evaluation of the CMW-RBAC system. We also estimate how different aspects of the access control influence the performance.

Geneva
April 2010

Performance of CMW-RBAC at CERN

Table of Contents

1. Introduction 3

2. Authorization 3

3. Experimental Performance Evaluation 4

3.1 Architecture 5

3.2 Logging 5

3.3 Size of the access map 6

3.4 Number of the user roles 6

4 Conclusions 7

5 References 7

Geneva
April 2010

1. Introduction

Controls Middleware (CMW) is a software infrastructure delivered and managed by CERN Beams Department/Controls group. Its goal is to provide a generic way of accessing accelerator devices [1]. CMW provides access to devices from application programs in a distributed heterogeneous control system. It allows interconnecting applications and devices implemented in Java or C++, and running under Unix or Windows platforms.

The CMW design reflects the Accelerator Device/Property Model in which devices, named entities in the control system, can be controlled via properties. Each device belongs to a Device Class and it is the Device Class, which defines the properties which can be used to access the device. CMW implements this model in a distributed environment with devices residing in servers that can run anywhere in the controls network. It provides a location-independent and reliable access to the devices from control programs. By invoking the device access methods, clients can read, write and subscribe to device property values [2].

From the point of view of the CMW, each accelerator device is a device server. Control applications maintaining communication with accelerator devices are viewed as CMW clients. The following figure shows the overall architecture of CMW.

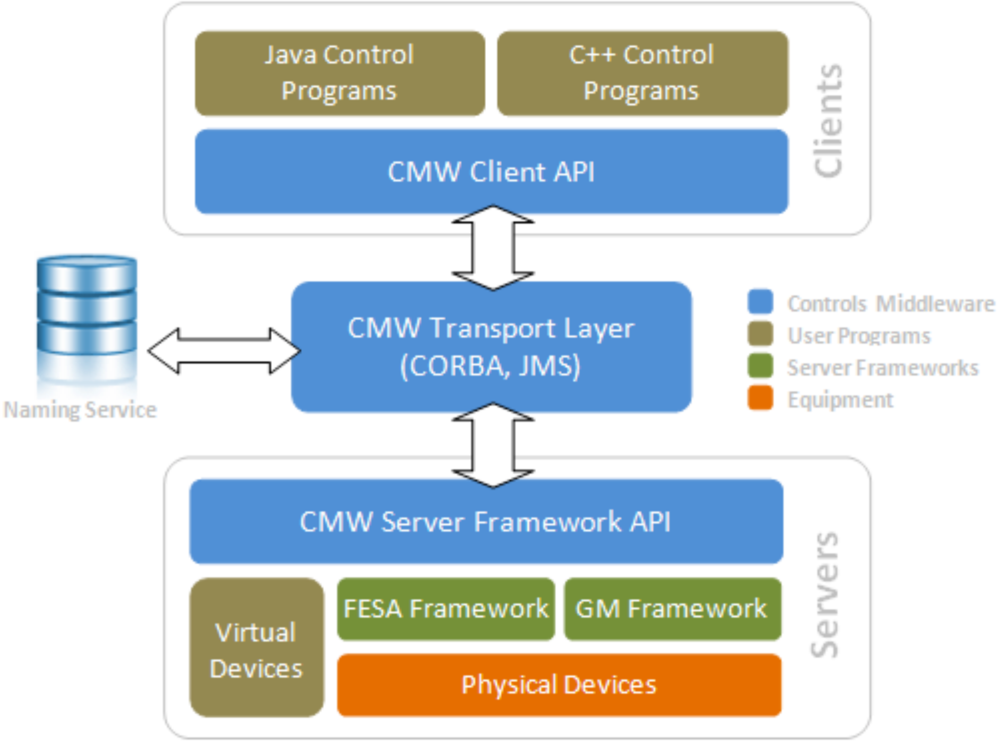

Figure 1: Controls Middleware Architecture

2. Authorization

Authorization is the function of specifying access rights to resources or services. The process of authorization is performed for each CMW request by CMW-RBAC authorization library. CMW-RBAC library is the part of each device server. It is implemented in C++ and compiled for all platforms used at CERN (LynxOS, Scientific CERN Linux, and Windows). Authorization server library is called by CMW server to perform authorization for equipment transactions. Every operation (get, set or subscription demand) is subject to authorization process and its execution can be denied in case of requester having insufficient privileges. Other types of operations (e.g. reboot, configuration change) can also be controlled with CMW-RBAC by introducing additional server properties and limiting their access with appropriate access rules.

Transaction request is made from the application via the CMW client to the CMW server. The CMW-RBAC token obtained at authentication is passed to the CMW client. There the digital signature is verified, and if valid, the token is sent to the CMW server. If the token is not valid a meaningful exception message is returned to the sender. The CMW looks up the permission in the access map, and depending on access rights either grants access or blocks the request. [3].

Figure 2 demonstrates the general overview of the authorization process.

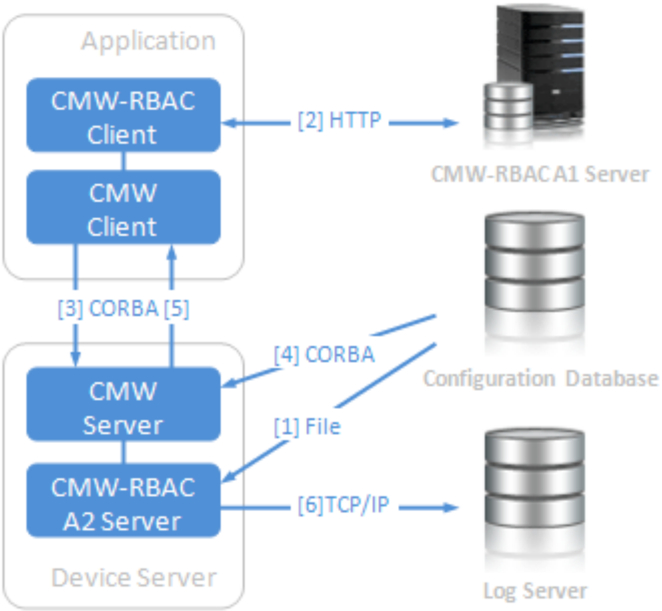

Figure 2: Authorization process

1. Access map loaded from the local file on device server startup.
2. Client application authenticates on the CMW-RBAC A1 server and obtains a valid token.
3. Token is passed to the CMW client and then to the device server via CMW protocol.
4. CMW server receives the accelerator mode from timing source.
5. Dynamic authorization algorithm verifies user permissions and either allows to execute operation or throws an exception.
6. Result of authorization process logged for auditing.

3. Experimental Performance Evaluation

An important concern with any design is its performance. CMW-RBAC performs authorization for every transaction within control system and thus authorization algorithm must work as fast as possible. It should not hinder the performance of the middleware. In order to estimate how much different factors affect the performance we made several experiments.

3.1 Architecture

LHC controls can run in 2-tier mode, meaning the application and the client are on one machine and the database and devices are accessed directly on the second machine. This configuration is used when in developer testing mode. However when the system is in operation the configuration is usually 3-tier, meaning the client is on a middle tier. A common client is shared by several applications and consolidates requests. In 2-tier, a dedicated client ensures that each request is made from the same application. The CMW-RBAC token can be passed once per session and the credentials can be used for each subsequent request.

In 3-tier, the requests can come from any application at random times. Thereby token must be passed for each transaction. This slows down processing because of additional operations of token serialization and de-serialization.

In order to measure the performance we accessed test device server from remote client. The authorization time is the time measured on the server side for the following operations: token serialization, de-serialization and authorization process. The request time is the duration of the whole equipment call. This includes operations of CORBA processing, sending over the network, authorization, forming response etc. Table 1 demonstrates difference in terms of performance between 2-tier and 3-tier architecture.

Table 1. Comparison 2-tier vs. 3-tier

	Linux (SLC 4, L864) cs-ccr-dev1 2.7 GHz, 4xCPU	LynxOS (ppc4) cfv-864-cdv26 400 MHz, 1xCPU	Windows XP SP3 abpc11036 3.0 GHz,1xCPU
CMW Authorization 2-tier	12,83 μs	201,5 μs	13,44 μs
CMW Request 2-tier	190 μs	2000 μs	200 μs
CMW Authorization 3-tier	171,73 μs	2111 μs	182,4 μs
CMW Request 3-tier	400 μs	4100 μs	410 μs

As expected 3-tier token verification on every request has a large impact on performance. Authorization in 3-tier mode takes almost 45% of the whole request time.

3.2 Logging

The purpose of this test is to estimate how logging of authorization impacts on performance. In order to evaluate the impact we measured the whole time of the equipment call when logging was disabled and when it was enabled. When access is granted only some user information (token, location, transaction) is being sent to the log server. But when access is denied then device server also logs the extract of the access map with the access rules that protect given transaction. Throwing a detailed exception to the client is also expensive. The results of this test are presented in Table 2.

Table 2. Logging of authorization result

	Linux (SLC 4, L864) cs-ccr-dev1 2.7 GHz, 4xCPU	LynxOS (ppc4) cfv-864-cdv26 400 MHz, 1xCPU
Policy: no-check. Log: off. Access: granted	140 μs	1900 μs
Policy: no-check. Log: on. Access: granted	200 μs	5400 μs
Policy: strict. Log: off. Access: granted	150 μs	2000 μs
Policy: strict. Log: on. Access: granted	220 μs	5800 μs
Policy: strict. Log: on. Access: denied	390 μs	7400 μs

3.3 Size of the access map

Authorization process is performed for each transaction and therefore should work as fast as possible and should not slow down the performance of the system significantly. As an authorization time is a concern for the CMW operation, it was decided to represent the access map in the CMW server in the form of trees, a separate one for every operation type. Nodes placed with the same distance from the tree root group authorization argument of one type. Authorization is performed by traversing the tree, appropriate in respect to the operation type, from its root to a leaf, trying at each level to match exact authorization parameter or to find a wildcard '*'. If this succeeds, the access is granted.

Such a structuring of the access map allows us to assure authorization with time complexity of $O(\mathrm{lo}$ instructions, where n is the number of access rules. Figure 3 demonstrates how size of the access maps impacts on performance.

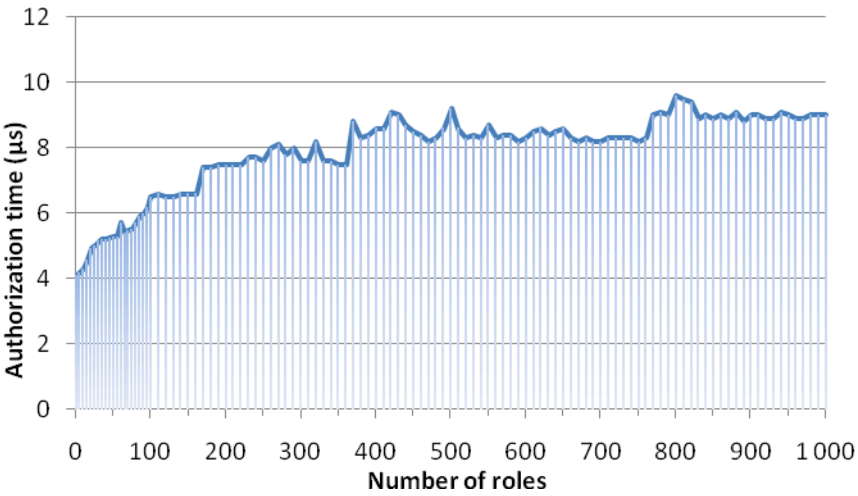

Figure 3. Size of the access map

3.4 Number of the user roles

Number of the user roles may also affect the performance, because authorization algorithm might perform search in the tree for all of them in worst case.

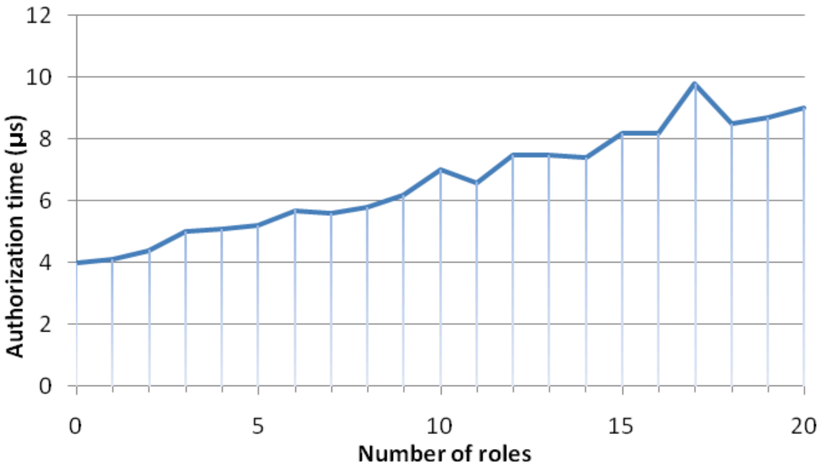

Figure 4. Number of the user roles

In order to test how this factor impacts on performance we measured authorization time for the request made by user with several roles. The last role in the list was the target one. Figure 4 demonstrates how authorization time depends on number of the user roles.

4 Conclusions

In this paper we presented experimental performance evaluation for the CMW-RBAC system at CERN. Our results lead to the following conclusions:

1. 3-tier token verification on every request has a larger impact on performance than the other concerns. In a 3-tier configuration the CMW authorization takes 2.1 ms at most. At this time, this is acceptable according to the requirements. The performance decrease is caused mainly by additional operations of token de-serialization and digital signature verification. Since de-serialization algorithm is quite optimal and efficient, there is possibility to increase performance by choosing another digital signature algorithm or implementation. In order to improve the performance there is need to investigate other algorithms and solutions in this field.
2. Logging of each request has a very significant effect on the performance. Our tests show the difference between having logging on versus having it off is roughly 100% when access is granted and even more when access was denied. However typical device server configuration disables log output for authorized requests, this is only enabled during server debugging. Logging of unauthorized requests is always enabled and cannot be turned off. The reason for large performance decrease is that authorization algorithm produces detailed explanation why access was denied (including excerpt of the access map for the given transaction), and throws this as exception to the client. In the future CMW team plans to revise logging mechanism in order to improve performance and maintainability.
3. The size of the access map has little effect on performance due to sophisticated and optimized search algorithms. Time complexity of the authorization algorithm is logarithmic, because of efficient binary search in the access tree. Our test result shows a double increase in authorization time between one rule and 200 rules access map. Further increasing of the access map size has no significant impact on authorization time.
4. The number of roles in the client token influences the performance more significantly than the access map size. Authorization time almost doubled when number of roles increased from 1 to 20. However from the client side the performance decrease was only 5%. Thus this factor has little impact on the performance of the entire system.

5 References

1. K. Kostro, V. Baggiolini, F. Calderini, F. Chevrier, S. Jensen, R. Swoboda, N. Trofimov, “Controls Middleware – the New Generation”, EPAC’02, Paris, France, p. 2028.
2. N. Trofimov, V. Baggiolini, S. Jensen, K. Kostro, F. Di Maio, A. Risso: Remote Device Access in the New Accelerator Controls. – ICALEPCS 2001, San Jose, USA, 27-30 Nov. 2001.
3. P. Charrue et al.: Role Based Access Control for the Accelerator Control System in the LHC Era - Design. EDMS Id 805654, (2007).