Implementation of a formal security policy refinement process in WBEM architecture

Laborde R., Kamel M., Barrère F., Benzekri A.

Université Paul Sabatier - IRIT/SIERA
118 Rte de Narbonne F31062 Toulouse Cedex04
Phone: +33 (0) 5 61 55 60 86 - Telecopy: +33 (0) 5 61 52 14 58
{laborde, mkamel, barrere, benzekri}@irit.fr

Abstract. Policy based network management (PBNM) is now a recognized standard approach for network management. Its particularity comes from the separation of the rules that govern the system and the functionalities provided. Even so, policy refinement - i.e. translating high level policies into concrete ones - remains a key problem in the policy based network management. In previous papers [3,4], we have proposed a formal framework that focuses on network security information management refinement. It allows the expression of network security goals and abstract tactics. An associated evaluation method guarantees the consistency and the correctness of specified network security tactics. However the tactics are expressed using data flow-based language that is independent from the technologies. Therefore, tactics cannot be directly enforced by technologies. In this paper, we complete the previous framework. We consider the enforcement of the tactics and we implement this refinement process in WBEM architecture.

1 Introduction

The policy based network management (PBNM) proposes an approach to bridge the gap between the management goals and the associated configurations. In the management context, *policies are rules governing the choices in the behaviour of a system* [9]. It allows the separation of the rules that govern the system from the functionalities provided by it. The policy rules can be specified at different abstraction levels from the goals to the configurations [9,10]. A process, called refinement, transforms high level policies into lower ones [8].

Therefore, the refinement process is a key problem in PBNM approaches. How to translate high level into low level policies? Works in [8,1] have proposed to refine goals. The method [1] uses the goal oriented requirement engineering (GORE) approach [5]. Goals are specified in KAOS [2]. Each goals is refined into sets of sub-goals based on predefine formal schemes until they can be directly enforced. Goal refinement formalization is an interesting approach. Nevertheless, it is not sufficient to automate the refinement task because it only deals with the behavioural aspect. It does not consider the informational part of the refinement problem.

The model-based management approach [6,7] utilizes object-oriented models of a managed system to support the derivation which is divided in three abstraction levels: Roles & Objects, Subjects & Resources, and Processes & Hosts. Each level is a refinement of the upper one. The designer graphically defines the three abstraction level models and the tool guides the derivation. This tool defines clearly the different abstraction levels of the models. So, it facilitates the design and the deployment of network security policies However, it does not formally guarantee that the security policy is consistent and correct, i.e., the carried out decisions are relevant.

Our work focuses on the *formal derivation of the management information* from the security goals to the security configurations. What is the management information model to be used at each derivation step? Is the management information valid? What is the formal relation between these information models?

The Laborde et al.'s framework presented in [3,4] proposes a formal language for the expression of RBAC-based *network security goals*, and abstract *network security tactics*. It includes a formal *evaluation* method that guarantees the *consistency* and the *correctness* of security tactics regarding RBAC-based goals (both upper layers in fig1).

Nevertheless, this framework cannot be used by a management platform because the network security tactics are technology independent. It is then not possible to enforce them directly into technology specific configurations (the lower layer in fig. 1). First, we propose a formal method to prove that the network security tactics are enforceable by real devices. Second, we present a management architecture in compliance with this formal framework. We have chosen the WBEM initiative because it focuses, like our approach, on information models.

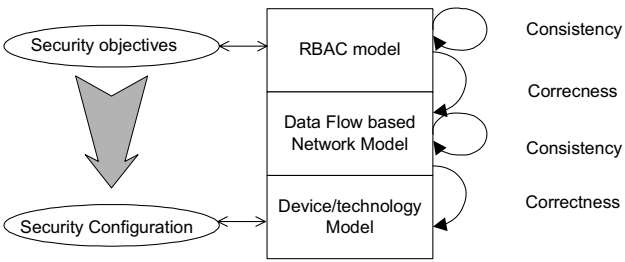

Fig. 1. The proposed framework decomposition

Consequently, the paper is organized as follow. Section 2 introduces the Laborde et al.'s framework. It presents the language and the associated evaluation method. Then, section 3 exposes a formal model that guarantees the technology independent tactics to be enforceable into real technologies configurations. Section 4 defines the WBEM-based architecture that implements all the concepts introduced in the formal refinement framework. Finally, we conclude with directions for future works.

2 The Laborde et al. framework

The Laborde et al. framework [3,4] constitutes a first step towards the security derivation process taking into account network environments. It proposes a formal way for the expression of *network security tactics* considering network topologies and their validation against RBAC [11] policies.

2.1 Network security tactics expression

The notion of data flow is at the heart of the network security management problem. Data flows are not restricted to a set of IP addresses, application ports, etc. Here, data flows represent the data exchanged between the entities that perform given actions and the entities that store information (e.g., client and server communication or users and files). In a network, each intermediate device acts on data flow according to its own capabilities and its configuration. The Laborde et al. language allows the expression of the possible treatment according to four technology independent atomic functionalities: end-flow, channel, transform and filter functionalities. Then, the definition of security tactics consists in the interconnections of these configured atomic functionalities.

The mechanisms that *consume/produce* data flows such as the end-systems are represented by the *end-flow functionalities*. Moreover, they constitute the link between the service management layer model, i.e. RBAC, and the network management layer model. We call active end-flow (AEF) functionality (resp. passive end-flow functionality - PEF), an end flow connected to one or more subjects (resp. objects) of the top layer RBAC policy. In the RBAC model, the set of roles constitutes the relation between the set of subjects and the set of objects – i.e., a user can access an object if being assigned to the role that is associated to the permission granting the access to object. So, we append a list of roles to each EF to indicate the flows that the EF can produce. When the users launch their authorized services, this implies a communication between all the AEF and the PEF corresponding to the associated role (fig. 2). We represent a data flow as (source_EF, destination_EF, role, <transformation_list>). If no transformation is applied, the transformation list only contains the keyword *any*.

The mechanisms that *propagate* data flows such as physical supports and relay devices are represented by the *channel functionalities* (fig. 3).

The mechanisms that *transform* data flows such as cryptosystems are represented by the *transform functionalities* (fig. 4). The transform functionality receives a data flow (ex: (ef_1,ef_2,role1,<Transf_list>)) from one of its two interfaces. According to the transformation rules conditions represented by a list of triple <set_of_source_EFs, set_of_destination_EFs, role> which identifies the data flows that must be transformed, it sends to the other interface the same data flow or the data flow transformed represented by the parameter "Group" that identifies the transformation applied (ex: (ef_1,ef_2,role1,<Group•Transf_list>)).

The mechanisms that *filter* data flows such as firewalls are represented by the *filter functionalities* (fig. 5). The filtering rules explicitly express the permitted flows

between the two interfaces by a tuple <set_of_source_EFs, set_of_destination_EFs, role, transformation_list>.

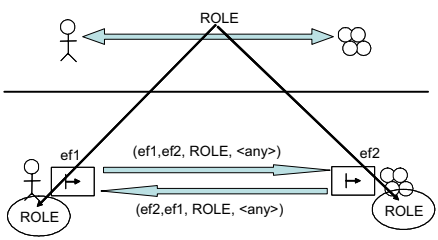

Fig. 2. End-flow functionality

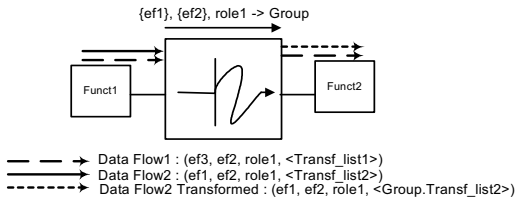

Fig. 4. Transform functionality

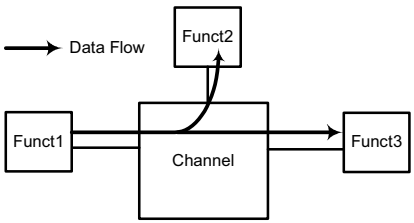

Fig. 3. Channel functionality

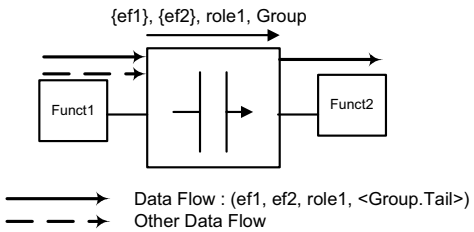

Fig. 5. Filter functionality

2.2 A VPN example case study

The following example explains how the language is used to implement an IPsec/VPN case study tactic definition (fig. 6). As in a traditional enterprise network, this example considers an edge router interconnecting a private network and a DMZ. The *App_Server* and the FTP servers are respectively installed in the private network and in the DMZ (fig. 6). The application level security policy is an RBAC one, without hierarchy, where two user groups *VPNmembers* and *Others* are defined. This organization is only based on the granted privileges. The *App_Server* server is dedicated only to the services usable by the *VPNmembers* group. The *FTP_Server* has two directories: /confidential and /pub. The directory "confidential" contains data only accessible to the *VPNmembers* users group. Data of the "pub" directory is accessible to everyone. $User_1$, $User_2$, $User_3$ and $User_4$ belong to *VPNmembers* and *Others* groups. $User_5$ is only member of the *Others* group.

The objective is to specify a VPN security tactic. The Private Network, the DMZ and the Internet interconnection infrastructures are specified thanks to channel functionalities because we use their transmission functionality. This approach of specification with a large granularity only considers the minimum set of functionalities provided by these infrastructures: their interconnection capability.

On the contrary it is possible to refine a specification as the edge router shows it. It has obviously the interconnection functionality (the channel functionality), setting as a security gateway with filtering capabilities (the three filter functionalities) and encryption mechanisms (the transform functionality, for example an IPsec module is installed). The modelling of the routing is carried out by filtering rules on the filter functionalities.

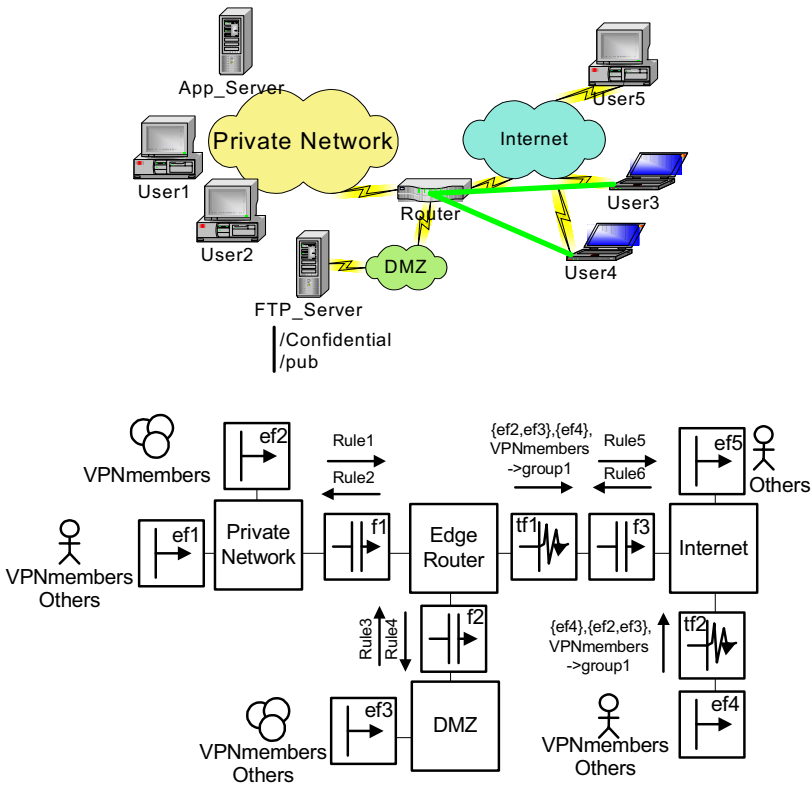

Fig. 6. Architecture and graphical specification of the VPN example

The servers are specified by two PEF. The *App_Server* server (ef_2) has the *VPNmembers* role because only the users with the *VPNmembers* role have the access rights. The PEF corresponding to the FTP server (ef_3) has the roles *Others* and *VPNmembers* because the permission (+all_access, FTP_Server/pub) is assigned to the *Others* role and (+all_access, FTP_Server/confidential) to the *VPNmembers* role.

The devices of $user_1$ and $user_2$ are represented by a single AEF (ef_1) because $user_1$ and $user_2$ have the same roles (*Others* and *VPNmembers*) and are connected to the

same channel functionality thanks to the concept of role which reduces the overall size of the specification. In the same way, the devices of $user_3$ and $user_4$ are specified by only one AEF (ef_4). The device of $user_5$ is specified by a different AEF (ef_5) because the *VPNmembers* role is not assigned to him.

Two transform configurations are defined on the transform functionalities tf_1 and tf_2 that add security properties - according to the *group1 confidentiality* service transform actions - to the communication between ef_2, ef_3 and ef_4 with the role *VPNmembers* (fig. 6). Moreover, the following filtering rules associated with the filter functionalities are specified:

- Rule1 = <{ef_1}, {ef_3}, VPNmembers, any>, <{ef_2}, {ef_4}, VPNmembers, any>, <{ef_1},{ef_3}, Others, any>
  This rule permits the untransformed data flows from ef_1 to ef_3 with the roles VPNmembers and Others, and the untransformed data flows from ef_2 to ef_4 with the role VPNmembers.
- Rule2 = <{ef_3}, {ef_1}, VPNmembers, any>, <{ef_4}, {ef_2}, VPNmembers, any>, <{ef_3},{ef_1}, Others, any>
  This rule grants the reverse data flows permitted by rule1 in order to enable bidirectional communications between the end-flows.
- Rule3 = <{ef_3},{ef_1, ef_4}, VPNmembers, any>, <{ef_3},{ef_1, ef_4, ef_5}, Others, any>
  This rule permits the untransformed data flows from ef3 to ef1 and ef4 with the roles VPNmembers and Others, and the untransformed data flows from ef3 to ef5 with the role Others.
- Rule4 = <{ef_1, ef_4},{ef_3}, VPNmembers, any>, <{ef_1, ef_4, ef_5}, {ef_3}, Others, any>
  This rule grants the reverse data flows permitted by rule3.
- Rule5 = <{ef_2, ef_3}, {ef_4},VPNmembers, group1>, <{ef_3}, {ef_4, ef_5}, Others, any>
  This rule permits the data flows transformed according to group1 from ef2 and ef3 to ef4 with the role VPNmembers, and the untransformed data flows from ef3 to ef4 and ef5 with the role Others.
- Rule6 = <{ef_4}, {ef_2, ef_3},VPNmembers, group1>, <{ef_4, ef_5}, {ef_3}, Others, any>
  This rule grants the reverse data flows permitted by rule5.

This specification approach facilitates the security network management layer expression since it is technology independent and since many management information are aggregated within the concepts of basic functionalities and roles.

In addition, as we will now present, the framework includes a formal verification process which allows validating the specifications.

2.3 Network security tactics validation

The first step consists in expressing the security tactics based on atomic functionalities. The interactions between the specified atomic functionalities should

be validated in order to prove that the security tactic involves no conflict and corresponds to the goal requirements. The Laborde et al. validation process allows the proof of network security mechanisms consistency and their correctness against RBAC policies [4]. First, a specification is transformed into the corresponding Coloured Petri Net. It is produced by interconnecting each CPN sub-model of the basic functionalities in the specification. Then, the dead state of the reachability graph produced by the CPN model is computed and analyzed thanks to the set of security properties such as confidentiality and availability and new configuration properties introduced in [4]. For example, the property of confidentiality prohibits an active end-flow functionality from receiving at any time a data flow, that is not transformed with confidentiality property, with any unassigned role. Finally, the model is checked, i.e. the dead state satisfies or not all the security properties. If it does not satisfy them then the mechanisms hence defined do not fulfill the requirements otherwise the specification is considered to be secure. The formal security properties definitions, the theorem proofs, the analysis process and its applicability in complex studies are given in [4].

3 Translation into technology specific information models

On the one hand, the Laborde et al. framework language allows the expression of network security tactics using a data flow based approach and regardless of the technologies specificities. The language permits a high level of abstraction in the data flows definition. On the other hand, each technology used for enforcing the network security tactics has its own capabilities. A technology capability means:

1. the possible actions (i.e., the treatments that can be applied on the data flows),
2. and the discrimination criteria to differentiate the data flows (i.e., the set of data flow value types that the device/technology can perceive).

Examples of the discrimination criteria are:

- HTTP proxies can differentiate data flows based on keywords in HTML pages.
- Stateless firewalls can only differentiate data flows according to IP addresses, transport layer protocol and port numbers.
- Switches view data flows as MAC addresses, Source Service Access Points and Destination Service Access Points numbers.

The problem of management refinement at this layer is to determine if the technologies are able to enforce the associated security tactics or not. By nature, the atomic functionalities represent the actions capabilities of the technologies. Then, the action part does not represent a possible refinement problem. Nevertheless, since the language permits a high level of abstraction in the data flows definition, a distinction between two data flows made at the network security tactics abstraction level by an atomic functionality does not imply that the corresponding technology is able to do it. This discrimination criteria problem is formalized as follows.

Let :

- D, the set of possible values characterizing data flows,
- T, the set of types of values (e.g., IP address, transport protocol, port number),

- $C_X \subseteq T$, the distinction capability of device X (e.g., routers perceive the IP addresses, transport protocol, port numbers, etc.). The distinction capability of a device is modelled as the set of types of values that it can distinguish.
- $f : T \rightarrow \mathcal{P}(D_T)$ a data flow where D_T is the set of values of type T. A data flow is modelled as a set of functions which return for each type of values a set of values of this type.
- F, the set of data flows,
- $\theta_G : 2^F \rightarrow 2^F$, the function associated to the transform group G with θ_{any} = identity,
- $\delta : EF \times EF \times ROLE \rightarrow 2^F$, the function that creates the associated flows (i.e., the set of values) associated to an untransformed data flow in the Laborde et al model.

Definition 1:
The derivation function between the network security tactics abstraction and device abstraction is defined as:
$\Delta ((ef_1, ef2, role, <G_1 \bullet G_2 \bullet \ldots G_n \bullet any>)) \equiv \theta_{G1} \circ \theta_{G2} \circ \ldots \theta_{Gn} \circ \theta_{any} \circ \delta(ef_1, ef2, role)$

Definition 2:
We call the technology X perception of the data flow f: $\mathcal{V}_X(\Delta (f)) = \Delta(f)_{|Cx}$.

Definition 3:
We say that technology X confuses the data flows f_1 et f_2 if $\mathcal{V}_X(\Delta(f_1)) \cap \mathcal{V}_X(\Delta (f_2)) \neq \varnothing$ that we note $\mathcal{V}_X(\Delta(f_1)) = \mathcal{V}_X(\Delta (f_2))$

Definition 4 – Loose property of derivation capability:
Technology X is said able to enforce a network security tactics:
1. if the tactic of functionality F associated to technology X states two different actions for two distinct data flows f_1 and f_2, and (f_1 and f_2 pass through F)
2. it implies that $\mathcal{V}_X(\Delta (f_1)) \neq \mathcal{V}_X(\Delta (f_2))$

The loose property of derivation capability, contrary to the strict property of derivation capability, considers that if X never sees f_1 and f_2, X can confuse both data flows and X is able to apply the network security tactics.

In the example of fig 6, the transform functionality tf_2 has the following configuration {ef_4}, {ef_2, ef_3}, VPNmembers → $group_1$. Both data flows <ef_4, ef_3, VPNmembers, any> and <ef_4, ef_3, Others, any> pass through tf_2. We recall that the directory "confidential" on *FTP_Server* contains data only accessible to the *VPNmembers* users group and the data of the "pub" directory is accessible to *Others*. We consider also that the security group $group_1$ represents an IPsec tunnel. The distinction capability of IPsec C_{IPsec} is the set of types IP address, port number and transport protocol. Both *VPNmembers* and *Others* role use the same transport protocol TCP and protocol numbers 21 and upper than 1024.
case 1: *The address space used for the VPN architecture is private*. So, the IP address of *FTP_Server* for the *VPNmembers* role is different from its IP address for the *Others* role. Consequently, $\mathcal{V}_{IPsec}(\Delta(<ef_4, ef_3, VPNmembers, any>)) \neq \mathcal{V}_{IPsec}(\Delta(<ef_4, ef_3, Others, any>))$. Then, the tactics can be enforced by IPsec.

case 2: *The address spaces used for the VPNmembers and Others roles are not different*. So, the IP address of *FTP_Server* for the *VPNmembers* role is the same as its IP address for the *Others* role. Consequently, $\mathcal{V}_{IPsec}(\Delta(<ef_4, ef_3, VPNmembers, any>)) = \mathcal{V}_{IPsec}(\Delta(<ef_4, ef_3, Others, any>))$. Then, the tactics cannot be enforced by IPsec because IPsec confuses $\Delta(<ef_4, ef_3, VPNmembers, any>)$ and $\Delta(<ef_4, ef_3, Others, any>)$.

This formalization allows the establishment of the network security tactic enforceability. To summarize, we formalize the management information derivation from RBAC objectives, through the Laborde et al. framework tactics, to the technologies configuration. We have implemented our methodology to demonstrate its feasibility using WBEM architectures.

4 Implementation of the derivation process for WBEM management platforms

Our methodology only deals with management information transformation through different models. The implementation needs a representation of the information for each management layer model. Several classical device information models used by management platforms exist: MIB/SMI, PIB/SPPI, MIB/GDMO, proprietary management information model. Nevertheless, RBAC and Laborde et al. information models cannot be easily represented by them because they are specific to the associated distribution protocols. The "Common Information Model" (CIM [12]) meta-model, which is part of the WBEM initiative [13], has been selected to implement this derivation process because the three management layers (top-down: RBAC, data flow based and device based) can be specified in a CIM single formalism. In addition, CIM is widely used and its standard schemas are very rich (e.g. it includes both the RBAC and the IPsec technology models).

4.1 Expression of the three models

CIM already includes RBAC (Role, AuthorizedPrivilege and Service classes) and lots of technologies information models - end-systems network information (IPProtocolEndPoint, TCPProtocolEndPoint), IPsec technology configurations (SecurityAssociationEndPoint, and FilterList) and firewall rules (FilterList). However, CIM, which is a device-based management information model, does not include any information on data flows. Therefore we have extended the core model with the Laborde et al. framework functionalities.

We have specified all the basic Laborde et al. schema via the new class "GenericFunctionality" which inherits from the class "LogicalElement" (fig. 7). The "GenericFunctionality" class specializes into the three classes: "ConfigurableFunctionality" that corresponds to a transform and filter functionality, "EFFunctionality" that is an active or passive end-flow functionality and "ChannelFunctionality".

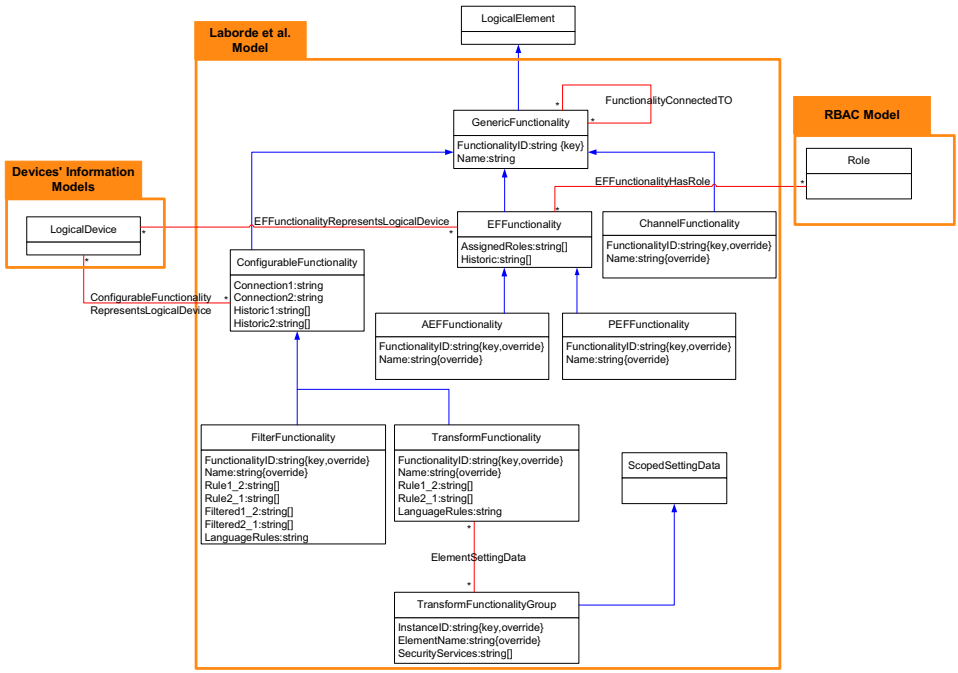

Fig. 7. Laborde et al. model scheme in CIM

The connections between the basic functionalities are specified by the "FunctionalityConnectedTo" association. The connection between the RBAC model, our data flow based network model and the devices information model is specified by the associations:

– "EF_FunctionalityHasRole" allows the specification of roles assignment to end-flow functionalities,
– "ConfigurableFunctionalityRepresentsLogicalDevice" and "EF_Functionality-RepresentsLogicalDevice" represent the relation between the network management layer model and the real devices. This relation is divided into two associations because an end-flow functionality represents a logical device according to a specific role. For example, it allows to consider that a device is member of different addressing spaces according to different activities (e.g. VPN architectures - cf. the case 1 of the example given in section 3). Thus, the association "EF_FunctionalityRepresentsLogicalDevice" has a property "RoleName" that identifies the involved role in the relation end-flow functionality/logical device. The configurable functionalities (transform and filter functionalities) represent the logical devices through the "ConfigurableFunctionalityRepresentsLogicalDevice" association.

4.2 Definition of the technologies perception in CIM

We restrict the representation of the function Δ, which defines the correspondence between the data flow models in the Laborde et al framework and the element layer model to the previous IPsec VPN case study. The Laborde et al. model defines the data flows by the tuple (ef source, ef destination, role, <transformation list>). For example, the capability of the IPsec and the filtering routers technologies is: the set of IP addresses, the set of transport protocols and the set of application ports. Then, the IPsec and filtering router perception of a data flow is:

- For active end-flow functionalities, the IP addresses may depend on:
  - The IPProtocolEndPoint address of the LogicalDevice instances associated to the active end-flow functionality with a specific role. In fact, one active end-flow functionality can represent one or more LogicalDevice instances.
  - *Or* the IPProtocolEndPoint address of the LogicalDevice instance associated to the transform functionality that has transformed the data flow with "transformation_group". This represents the IPsec tunnel end-point.
- For active end-flow functionalities, the IP addresses may depend on:
  - The IPProtocolEndPoint address of the LogicalDevice instances associated to the passive end-flow functionality with the specific role *and* the IP address of the LogicalDevices that implement the services associated to a role.
  - *Or* the IPProtocolEndPoint address of the LogicalDevice instance associated to the transform functionality that has transformed the data flow with "transformation_group". This represents for example an IPsec tunnel end-point.
- The transport protocols and the port numbers may depend on:
  - The transport protocol and the port number (TCPProtocolEndPoint or UDPProtocolEndPoint) associated to the services authorized by a role.
  - *Or* the transport protocol and the port number associated to the transformation protocols that implement the transformation group.

We have developed a tool (using Java and CIM over HTTP [14] queries) which implements the *loose property of derivation capability*. It checks if all the devices can enforce the network security tactic. It also generates the associated firewalls and IPsec configurations using the client of the WBEM services platform [14].

5 Conclusion

In this article, we have presented a new generic framework for security derivation in a network environment. The three models indicate how to express management information at the different levels. The RBAC model allows the expression of security objectives using users/subjects, roles and permissions notions. The Laborde et al. model formulates network security tactics in terms of data flows constraints. And the technologies layer is modeled by an abstraction view of the technologies capabilities. Moreover, formal evaluation techniques are included in the framework allowing intra and inter formal analysis.

This formal framework is completely implemented in the CIM/WBEM initiative architecture. The RBAC and the devices information models were previously

specified in CIM. We have included the Laborde et al. language and all the dependencies between the different models.

The presented work, formalizing the information model derivation, constitutes one more step towards the automation tasks. Nevertheless, there is still work to do. First, we will focus our work on defining algorithms that can generate correct network security tactics based on an RBAC policy, a basic functionalities specification and the derivation rules. The GORE approach [5] seems to be an interesting lead. In the same way, we should enhance our CIM algorithm in order to aggregate the configuration of the devices and limit the size of the configuration.

Moreover, we will define a device specifications database in Laborde et al. language. Hence, a device plugging in the network implies to interconnect its specification to the global network specification. Device unplugging or crashing implies to disconnect its specification. Any modification of the global network security tactic specification implies the calculus of a new basic functionalities configuration or, if it is impossible, a monitoring message. Then, the real device management layer can act on the network tactics management layer and conversely.

Acknowledgments
We are grateful to M. Sibilla and E. Lavinal for their helpful comments on CIM and the English writing.

6 References

[1] A. Bandara, E. Lupu, J. Moffet, A. Russo,. “A Goal-based Approach to Policy Refinement”, in: Policy 2004.
[2] Dardenne A., Van Lamsweerde A., Fickas S., “A goal directed requirements acquisition”, Science of computer programming, vol. 20, 1993.
[3] Laborde R., Nasser B., Grasset F., Barrère F., Benzekri A. “Network Security Management: A Formal Evaluation Tool based on RBAC Policies”. IFIP NetCon'2004.
[4] Laborde R., Barrère F., Benzekri A., “A security management information model derivation framework: from goals to configurations”, In IFIP FAST2005, to appear.
[5] Van Lamsweerde A., “Goal-Oriented Requirements Engineering: A Roundtrip from Research to Practice”, RE’04, 2004.
[6] Lück I., C. Schäfer, H. Krumm, “Model-based Tool-Assistance for Packet-Filter Design”, In: Policy 2001, LNCS 1995, 2001.
[7] Lück I., S. Vögel, H. Krumm, “Model-based configuration of VPNs”, in Proc. 8th IEEE/IFIP NOMS 2002, 2002.
[8] Moffet, J., Sloman M. S., “Policy Hierarchies for Distributed Systems Management”, IEEE JSAC 11 - Special Issue on Network Management, 1993.
[9] Sloman M., “Policy Driven Management for distributed systems”, Journal of Network and Systems Management, vol 2, no. 4, Dec. 1994.
[10] Westerinen A., Schnizlein J., Strassner J., Scherling M., et al., “Terminology for Policy-Based Management”, RFC 3198, 2001.
[11] ANSI, “Role-Based Access Control”, ANSI/INCITS 359-2004, February 2004.
[12] URL: http://www.dmtf.org/standards/cim
[13] URL: http://www.dmtf.org/standards/wbem/
[14] URL: http://wbemservices.sourceforge.net/