Enhancing Security in ETSI Open Source MANO with Usage Control Capability

A. La Marra, A. Lunardelli, F. Martinelli, P. Mori, A. Saracino
IIT-CNR
Pisa, Italy
{antonio.lamarra, alessio.lunardelli, fabio.martinelli, paolo.mori, andrea.saracino}@iit.cnr.it

P. Castoldi[1,2], F. Marino[1], B. Martini[1,2]
[1]*Scuola Superiore Sant'Anna*, [2]*CNIT*
Pisa, Italy
{castoldi, fr.marino}@santannapisa.it
barbara.martini@cnit.it

***Abstract*—In this paper, we present and discuss the extension to the Open Source MANO software framework with advanced authorization capabilities able to enhance the security support in terms of preservation of virtual resources and slices integrity in a dynamic context of users, services and resources. The proposed extension consists of a Usage Control system integrated with different OSM subsystems (i.e., lifecycle management, service front-end, and monitoring components) able to enhance the traditional authorization operations with on-going usage control on established slices and virtual resources by continuously reconsidering the granting of resources in light of mutable attribute of users, resources and environment (e.g., presence of viruses, set-up of weak passwords). We provide a description of the ongoing integration with the Open Source MANO software. We also discuss the main issues we addressed in the software integration process and how the Open Source MANO could further evolve to fully encompass a security support with Usage Control.**

I. Introduction

Technologies like Sofware-Defined Networking (SDN) [1] and Network Function Virtualization (NFV) [2] are moving networks toward software-based automation and higher network flexibility by exploiting software virtualization and programmability [3]. Thus, network providers will be able to partition network infrastructures into virtual networks (i.e., slices) with characteristics tailored to meet the specific needs of applications and services built on top of it [4].

In such scenario, the adoption of some degree of automation and orchestration is key for convergent resource management and service delivery operations [5], as addressed by the MANagement and Orchestration (MANO) component within the ETSI NFV Architectural framework [6]. The NFV MANO provides the orchestration and lifecycle management of physical and/or software resources that support the infrastructure virtualization, including computing, networking, storage, and virtual machine resources (i.e., Virtual Network Functions (VNFs)). As an operator-led community, Open Source MANO (OSM) is an ETSI-hosted initiative to develop an open source MANO software stack aligned with ETSI NFV aiming at automatically deploying slices with independent characteristics and taking care of lifecycle management and scaling operations [7]. In OSM slices are deployed as Network Services (NSs) specified in terms of component VNFs, Forwarding Graphs, and inter-VNF Links.

978-3-903176-15-7 © 2019 IFIP

Although OSM provides a production-quality MANO software stack, effective NFV/SDN deployments and infrastructure slicing in a dynamic service context must consider the interdependence of multiple slices and services and any security threats that could come from the malicious or improper use of assigned virtual resources. Indeed, the underlying resources are eventually shared between slices and services and highly-changing context of users, resources and services poses new security challenges to preserve the integrity of the physical and virtual infrastructure while assuring the adequate user experience of services delivered on top of slices [8]. In fact, the shared computing resources which NFV allow to access in an ubiquitous and convenient fashion are typically not designed to support multi-tenancy requirements. Sharing physical resources such as hard disks, RAM, CPUs, GPUs can allow a compromised VNF to access other victim VNFs through shared memory, network connections, and other shared resources, leading to the disruption of the entire cloud infrastructure and services [9].

In order to address the security challenges posed by such a novel service ecosystem, in this paper we propose to extend the OSM with advanced authorization functionality suitable for dynamic service context to avoid the improper access to virtual resources and slices while they are used by the vertical to run services. In this work, we propose a Usage Control system integrated into OSM subsystems (i.e., lifecycle management, service front-end, and monitoring) to enable OSM to reconsider already made resource granting decisions in light of changes in attribute values relevant for security (e.g., the integrity status of the virtual machines or the reputation of the user) according to specified policies. In case of policy violations, the proposed system is able to react by enforcing the countermeasures defined in the policy itself (e.g., the slice including a corrupted virtual machine is suspended/terminated). The proposed security support in not currently included in OSM, still the need for proper authorization mechanisms are envisaged into the NFV MANO framework as stated in [10].

In terms of authorization capability, OSM currently supports the definition of different roles for users with different sets of

privileges to apply consistent access to resources that OSM manages. Indeed, the orchestration of virtual resources and slices requires a significant set of capabilities and privileges to perform all its required tasks, e.g., VNF on-boarding, NS on-boarding, NS deployment, NS termination, addition of new resources (e.g., datacenters, networks), etc. To this purpose, OSM release FIVE [11] supports a basic concept of access control, which is closely related to the Role Based Access Control (RBAC) model [12]. However, this is not sufficient to guarantee a proper and secure control on virtual resources and slice usage, mainly for two reasons.

At first, the role assigned to the requesting user could not be sufficient to express the constraints that are necessary to regulate the access to the resources. For instance, the RBAC model does not allow to express the following policy example: a slice can run as long as all its virtual machines (VMs) are not corrupted (e.g., infected by a virus) and the local users' passwords of such VMs are not weak. On the other hand, the adoption of an access control system following the Attribute Based Access Control (ABAC) model [13] instead of the RBAC one would ensure the enforcement of more expressive access control policies, such as the one previously listed, because attributes can be also managed.

As a matter of fact, the access context could evolve in such a way that the policy which initially granted the access is not satisfied anymore (e.g., after the slice creation, a VM gets infected by a virus, or a new local user is created on a VM with a weak password). Such changes of the context would not be detected by any traditional access control system.

For these reasons we claim that more adequate authorization support would be needed in OSM to address the previous points and, ultimately, to cope with the mutability of the reference high-dynamic scenario. Thus, we propose to integrate the usage control functionality into the OSM software framework in line with Usage Control model with benefits that can be illustrated with the following example. A vertical V (e.g., content provider) requests to the network provider (directly or mediated by an OSS/BSS) to instantiate a slice to run a service (e.g., content delivery). Accordingly, an OSM is triggered with a NS instantiation request to set-up a slice composed of VMs to host content delivery network components into VNFs (e.g., transcoders, caches, contents) interconnected through a virtual network able to properly transfer content data according to specified SLA specifications. In order to guarantee the security of such a slice, the network operator can assign to each VM of the slice a criticality level (e.g., critical, medium, low) expressing the security classification of the data stored in and processed by such VM, and enforces the following usage control policy: "*A slice can run as long as all its VMs are not corrupted (e.g., infected by a virus) and the local users' passwords of such VMs are not weak. In case of violation, if the corrupted VM is classified as Critical, the slice is suspended, otherwise the corrupted VM is interrupted and the slice is properly modified* ". Such an expressive policy can be only processed and enforced by a usage control system because it implies the continuous evaluation of the mutable attributes describing the VM integrity and local user passwords strength.

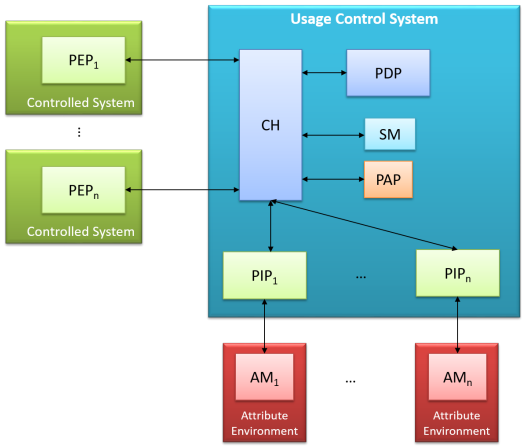

Fig. 1. Architecture of the Usage Control System

II. The Usage Control System

The proposed functionality is aligned with the Usage Control (UCON) model introduced in [14]. The main difference w.r.t. traditional access control models is that UCON is meant for dynamic scenarios, where resources are accessed by subjects while the attributes describing the features of such subjects, resources and environment change their values over time, as a normal effect of the system operation and resource usage. UCON enhances traditional access control by introducing policies that are evaluated continuously while the resources are used, in order to execute proper countermeasures in case of policy violation. Our implementation of the Usage Control System (UCS) is based on the XACML standard [15]. The UCS software architecture, shown in Figure 1, has been defined in [16]. According to the XACML standard, the UCS includes a Context Handler (CH), which is the frontend and receives access requests from Policy Enforcement Points (PEPs), which are embedded within the resource to be protected. The Policy Information Points (PIPs) are the components invoked by the CH to retrieve the attributes required by the Policy Decision Point (PDP) for the execution of the decision process, i.e., for the evaluation of the policy retrieved from the Policy Administration Point (PAP). Attributes are stored and managed by Attribute Managers (AMs), which provide the interfaces to retrieve their current values. Each specific scenario where the Usage Control system is exploited has its own set of AMs to manage the attributes required for the policy evaluation. Finally, the Session Manager (SM) keeps track of the ongoing accesses by managing a table including one record for each of the accesses that are currently in progress (i.e., active sessions).

The phases of the Usage Control decision process are regulated by the interactions between the PEP and the UCS as follows (according to [17]):

- *TryAccess*: the TryAccess message is sent by the PEP to the UCS when a subject requests to execute the access.

- *StartAccess*: sent by the PEP to the UCS when the access that has been authorized has actually started, to trigger the continuous mutable attribute monitoring.
- *RevokeAccess*: executed every time an attribute exploited in the ongoing policy changes its value to perform the re-evaluation. If a violation occurred, the RevokeAccess message is sent by the UCS to the PEP.
- *EndAccess*: sent by PEPs to the UCS when the access ends normally, for instance because the user closes it.

When a subject tries to execute an action A, the PEP suspends its execution and retrieves the information related to this access (subject and resource IDs, etc.). Hence, the PEP sends the *TryAccess* message with the data previously collected to the UCS, which performs the pre-decision process, i.e., the traditional access control phase, and returns the result to the PEP, which enforces it. If the execution of A is permitted, the UCS starts the ongoing-decision phase as soon as it receives the *StartAccess* message, by performing the first evaluation of the ongoing policy. From this moment on, as long as the action A is in progress, the Usage Control service evaluates the ongoing policy every time an attribute changes its value, and as soon as such policy is violated, the UCS sends a *RevokeAccess* message to the PEP, in order to take proper countermeasures.

III. Integrating Usage Control in OSM

This section describes the integration of the UCS into the OSM framework based on current release version available, i.e., OSM release FIVE. We aim at a not invasive integration with respect to what already exists in order to promote a smooth UCS adoption in production environments. Figure 2

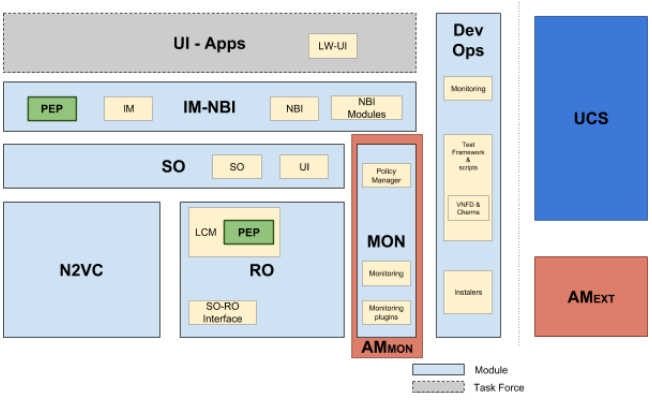

Fig. 2. Integration of the Usage Control System with OS MANO

reports the extension needed into the OSM framework, in which we highlighted (in green the PEPs, in red the AMs, and in blue the UCS) the components of the UCS that are being embedded. For a detailed description of the OSM components see [11].

A. PEP Deployment

The first integration step is the embedding of the PEP in the OSM framework. The main task of the PEP is the implementation of the previously described protocol to interact with the UCS.

First of all, the PEP must intercept the security relevant actions submitted by the users, e.g., the action considered critical because targeting the creation and usage of slices. To this aim, a PEP is being embedded in the front-end component, i.e., IM-NBI, as shown in Figure 2, by modifying API code in order to send the TryAccess message to the UCS. The PEP can be readily integrated into the OSM release FIVE by exploiting the configuration mechanisms OSM provides to exploit external authorization systems.

Then, the PEP is responsible to intercept both the beginning and the end of the execution of an access, in order to send, respectively, the StartAccess and the EndAccess messages to the UCS. In the OSM context, this means that the PEP must be able to understand when a slice has been deployed and the requestor has been given the control on it, and when the requestor terminates such slice. These tasks can be accomplished by embedding a PEP within the Life Cycle Manager (LCM) component of the OSM framework. In particular, our implementation is being leveraging the *charm* mechanism provided by the LCM component to run a sequence of ad-hoc codes. As a matter of fact, a *charm* is activated, according to the reactive programming paradigm, to respond to changes in state, including lifecycle events.

Finally, the forth PEP task is to receive (asynchronous) requests from the UCS, represented by the *RevokeAccess* message, and to ask the OSM framework to enforce the countermeasures specified in the received message during the slice usage. This PEP functionality is being embedded in the IM-NBI component as well, and it is being implemented as a service waiting for a message from the UCS, and invoking the OSM API in order to enforce the countermeasures decided by the UCS. The first version of our prototype is implementing the following countermeasures: *i)* slice termination; *ii)* resource reduction.

B. Attributes Identification and Extraction

The second integration step concerns the identification of the attributes for subjects, resources and environment that are relevant for the scenario of interest, and the development of the pieces of code required to get these attributes from the providers. In particular, in order to enable the UCS to retrieve the current values of the required attributes from the AMs, a specific plugin module, called Policy Information Point (PIP), must be implemented to manage the interactions with each of them. Each of these PIP implements the specific protocol supported by the AM for attribute retrieving and update, and provides a predefined API that is exploited by the UCS to call it. These attributes are the ones that will be exploited in the Usage Control policy to decide whether the right to perform the access holds, both when the access is requested and while the access is in progress.

With reference to the example of Section I, the AMs to be taken into account are the antivirus software which control the integrity of the VMs belonging to the slice and the password

strength checker which verifies the passwords of the local users of the VMs. Moreover, the OSM Monitoring Module can be exploited as provider of a considerable set of attributes as well, concerning resource utilization.

C. UCS Deployment

The last step is the deployment of the UCS component as stand-alone service, which stores and evaluates the usage control policies for each access request and during each access. The integration of the UCS into the OSM framework is actually implemented through the PEP and the PIPs, which manage the interactions with OSM as described in the previous points. In particular, the UCS is configured to exploit the previously defined PIPs to retrieve attributes data from the identified AMs, and it interacts with PEP components embedded in OSM through the previously described predefined protocol (*TryAccess,StartAccess,RevokeAccess,EndAccess*).

IV. Discussion

In this section, we provide observations, open points and call for actions on how to enhance OSM in light of enhancing security (i.e., authorization) support. In particular, we discuss the several issues we encountered during the integration of UCON into OSM.

Support of only a basic authorization system: currently OSM supports the RBAC authorization paradigm by providing the necessary abstract APIs and models, but it does not include a complete implementation yet. However, the RBAC could be inadequate in the service scenarios envisioned for 5G mainly for two reasons: the RBAC access control policies are not adequately expressive because they considers only the role of users; the policy evaluation is confined only at the time of resource request. We envision the need of adopting other access control models, such as Mandatory Access Control [18], or more advanced ones, e.g., UCON.

The lack in the IM-NBI of APIs enabling a coherent interaction with a sophisticated access and usage control framework: the current OSM implementation supports only basic functionalities in this regard, and it requires an extension. New APIs would be needed in order to allow the smooth integration of the UCON frameworks and the co-existence of multiple access and usage control frameworks, in order to meet in a fine-grained fashion the needs of each service in terms of required level of control and overheads introduced by the access and usage control system;

The urgency to provide UCON with the capability of triggering lifecycle management actions to react to policy violations: in this respect, the *charm* mechanism provided by OSM has been being assessed as a mean to perform the countermeasures that are required in this context, e.g., the dynamic reduction of the granted resources, in a consistent and non-invasive manner. In this regard, a more in-built support would be needed in terms of APIs to enhance the level of control on resources and slices.

The need to make available to UCON a wide range of attributes which can be relevant in the policy evaluation process: while the monitoring module provided by OSM can represent a significant source of information in this regard, AMs can be fruitfully deployed also in other functional blocks of the OSM architecture in order to provide useful higher lever information (i.e., the number of active services for each users).

V. Conclusion

In this paper, we discussed the extension to the Open Source MANO software framework we propose to add ongoing control operations according to Usage Control model to continuously reconsidering the decision of granting the resources in light of mutable attribute of users, resources and environment (e.g., presence of viruses, set-up of weak passwords). We described the integration steps and the main issues we addressed to embed such a security support in a smooth way considering the production-level of OSM software. We envisage that OSM core design should evolve to enlarge the cases where lifecycle management operations need to be taken, beyond the scale in/out, in order to include also security-level actions.

Acknowledgment

This work was partially supported by the EIT Digital project HC&IoT and by the SLICENET-5G project approved within the First Open Call of the 5GINFIRE project.

References

[1] B. A. A. Nunes *et al.*, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[2] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, 2015.

[3] B. Martini *et al.*, "A service-oriented approach for dynamic chaining of virtual network functions over multi-provider software-defined networks," *Future Internet*, vol. 8, no. 2, 2016.

[4] F. Paganelli, M. Ulema, and B. Martini, "Context-aware service composition and delivery in ngsons over sdn," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 97–105, 2014.

[5] C. J. Bernardos *et al.*, "5gex: realising a europe-wide multi-domain framework for software-defined infrastructures," *Transactions on Emerging Telecommunications Technologies*, vol. 27, no. 9, pp. 1271–1280.

[6] ETSI Industry Specification Group, "Network functions virtualisation (nfv): Architectural framework," *ETSI GS NFV 002 v1.1.1*, 2013.

[7] ETSI, https://osm.etsi.org/.

[8] Huawei, "One slice at a time: Sdn/nfv to 5g network slicing," 2018, https://www.huawei.com/en/about-huawei/publications/communicate/81/sdn-nfv-to-5g-network-slicing.

[9] M. Pattaranantakul, R. He, Q. Song, Z. Zhang, and A. Meddahi, "Nfv security survey: From use case driven threat analysis to state-of-the-art countermeasures," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 3330–3368, Fourthquarter 2018.

[10] ETSI, "Network functions virtualisation (nfv); nfv security; problem statement," ETSI, October 2014, https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/001/01.01.01_60/gs_NFV-SEC001v010101p.pdf.

[11] ETSI Open Source MANO, "Osm release five documentation," ETSI, https://osm.etsi.org/wikipub/index.php/OSM_Release_FIVE_Documentation.

[12] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38–47, Feb. 1996. [Online]. Available: https://doi.org/10.1109/2.485845

[13] F. Vincent C. Hu, David, K. Rick, S. Adam, M. Sandlin, K. Robert, and S. Karen, "Guide to attribute based access control (ABAC) definition and considerations," 2014.

[14] J. Park and R. Sandhu, "The $UCON_{ABC}$ usage control model," *ACM Transactions on Information and System Security*, vol. 7, no. 1, pp. 128–174, 2004.
[15] OASIS, "eXtensible Access Control Markup Language (XACML) version 3.0," OASIS XACML TC, January 2013.
[16] A. Lazouski, F. Martinelli, and P. Mori, "A prototype for enforcing usage control policies based on XACML." in *Trust, Privacy and Security in Digital Business (TrustBus 2012)*, ser. LNCS, vol. 7449. Springer Berlin Heidelberg, 2012, pp. 79–92.
[17] X. Zhang, F. Parisi-Presicce, R. Sandhu, and J. Park, "Formal model and policy specification of usage control," *ACM Transactions on Information and System Security*, vol. 8, no. 4, pp. 351–387, 2005.
[18] P. Samarati and S. Capitani de Vimercati, "Access control: Policies, models, and mechanisms," in *Foundations of Security Analysis and Design*, R. Focardi and R. Gorrieri, Eds. Springer Berlin Heidelberg, 2001, pp. 137–196.