Implementing the Domain Title Service atop OpenFlow

Flávio de Oliveira Silva[1][2], Maurício Amaral Gonçalves[2], João Henrique de Souza Pereira[1], Lásaro Jonas Camargos[2], Rafael Pasquini[2], Pedro Frosi Rosa[2], Sérgio Takeo Kofuji[1]

[1]Escola Politécnica da USP (EPSUP) – PSI – Universidade de São Paulo (USP)
Caixa Postal 61.548 – 05.424-970 – São Paulo – SP – Brazil

[2]Faculdade de Computação (FACOM) – Universidade Federal de Uberlândia (UFU)
Caixa Postal 593 – 38.400-902 – Uberlândia – MG – Brazil

{flavio,lasaro,pasquini,frosi}@facom.ufu.br, joaohs@usp.br

mauricio@algartelecom.com.br, kofuji@pad.lsi.usp.br

Abstract. *In previous works we have introduced the Domain Title Service (DTS) for a Future Internet. The DTS allows communicating entities to create or find and join ongoing conversations while minimizing costs and providing the exact QoS needed. For example, the DTS may aggregate multiple unicast streams of the same contents into one multicast and switch cryptography in mid-communication once a conversation becomes critical. In this paper we present a first DTS prototype implementation, based on the OpenFlow Software-Defined Networking approach.*

1. Introduction

Internet design started in the 70's and, after four decades and a huge success, most of that initially designed is still in place. However, applications vastly different from those that initially used the network are now being developed and bring a new set of requirements that the Internet is not able to satisfy in a proper way [Zahariadis et al. 2011].

Researchers from all over the world are engaged in designing a new Internet, from the ground up. Two results of such an effort are most relevant to this work. First, the advent of Software-Defined Networking (SDN)[Greene 2009] and, second, the proposal of Title Model and the Domain Title Service [Pereira et al. 2011, Silva et al. 2011].

SDN allows new networking protocols to be defined and readily experimented in real conditions in production networks. SDN is materialized in OpenFlow [McKeown et al. 2008], a network switch that externalized the control plane to a *controller* station.

The Title Model is a vision of how entities should be able to semantically specify their requirements and capabilities in order to communicate to each other [Pereira et al. 2011]. The Domain Title Service (DTS), is a distributed system over the network elements responsible for aiding communicating entities in locating their peers and in negotiating the establishment and maintenance of their conversations, accordingly to the Title Model [Silva et al. 2011].

In this paper we put these two results together and present a proof of concept OpenFlow-based implementation of the DTS. We do so by further describing the DTS and OpenFlow in Section 2. In Section 3 we discuss how this results is important to both the DTS and the OpenFlow development as well as make some concluding remarks and discuss our next steps.

2. Domain Title Service using Software-Defined Networking

The Domain Title Service is a distributed system over the network elements and communicating entities of a network. The DTS' main purpose is providing the means for the entities to find, establish and maintain communication sessions with each other. A *workspace* (a communication session in the DTS) consists of a logical bus that spans the network elements required to support the communication of the entities, such as the entities themselves and the switches connecting them.

A workspace is created by when an entity needs to communicate with a another for specific purpose, such as video-conferencing or file sharing. In order to create a workspace, the entity must specify the requirements it has and capabilities it may offer in conversing with other entities in the workspace. For example, the entity may require secrecy and delivery guarantees from its peers, while also offering a maximum bandwidth value. Another entity may join the workspace as long as it satisfies the requirements of the entities already within and requires no more that they offer. If the requirements change during the conversation, the DTS brokers the renegotiation of the requirements.

All entities that share a workspace see the same message exchange. That is, any message sent by one entity is multicast to all the other entities in the workspace. Delivery, ordering or other guarantees are provided only if required, thus making an efficient use of the physical layer. A workspace normally accommodates two entities, but if another one is interested in a conversation going in an existing workspace, the DTS provides the tools to discovering such workspaces, and this entity may join the conversation at any time. That is, given that entity passes any authentication and authorization restrictions associated to the workspace.

The DTS may be broken down in several parts, being the Domain Title Service Agents (DTSA) the system's cornerstone. The DTSA is responsible for both keeping the entities' and workspaces' metadata, and for coordinating the network elements to implement workspaces. Hence, implementing the DTSA and, therefore, the DTS requires control of network elements, which is not feasible in the in the current networks unless we use a software defined-based approach such as implemented by OpenFlow [McKeown et al. 2008].

2.1. DTSA as the Controller

As the DTSA's task of coordinating network elements is closely related to that of managing flows by an OpenFlow controller, we have decided to implement the first on top of the latter. In a nutshell, we extended the FloodLight open-source OpenFlow controller [Big Switch 2012] to closely work with the DTSA.

The extensions to the Floodlight controller consisted in a new module that instantiates the DTSA and handles the exchange of DTS control messages.

As a extented *IOFMessageListener*, this module it is able to listen incoming messages. By default, all messages that do not match any of the rules in the switch flow table are sent to the DTSA, as ilustrated by Figure 1. When a message is received, the listener is called and checks if the message is a defined primitive, as detailed in table 1. If so, the message is delivered to DTSA that process it and modify the switches using a *flow_mod*.

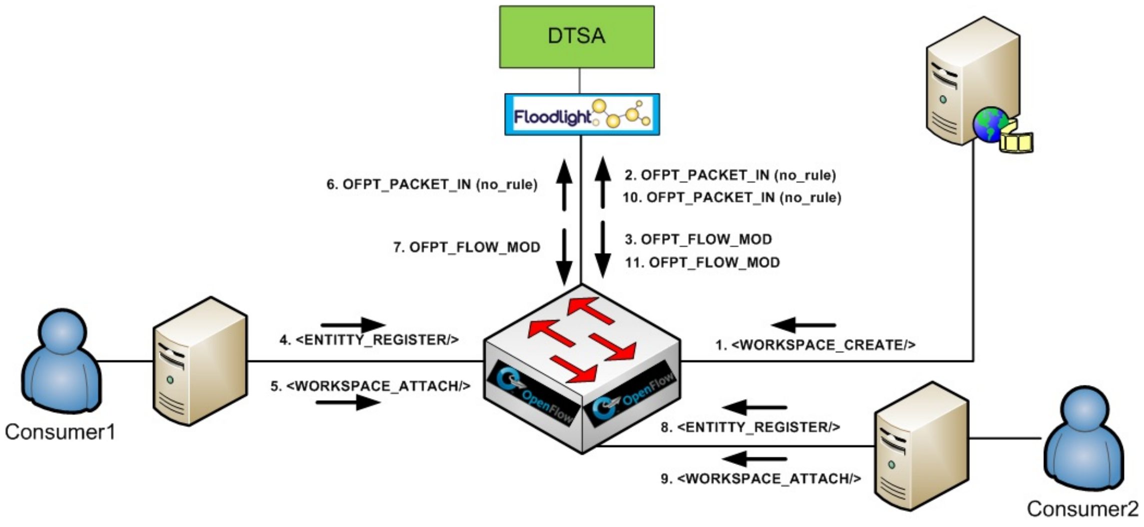

Figure 1. Control Messages Exchanged to Create and Adapt the Workspace

Initially an entity willing to provide a media requests the *WORKSPACE_CREATE* message. This primitive will be forwarded by the switch to the DTSA, using the OpenFlow *OFPT_PACKET_IN* message. DTSA will receive this indication and it will assign a VLAN id regarding this new workspace. By using the OpenFlow *OFPT_FLOW_MOD* message, this rule will be added to the flow table.

Table 1. Messages and their semantic regarding OpenFlow

Message	Meaning
ENTITY_REGISTER	Registers an entity at the DTS. To be registered an entity must present its title and communication requirements
WORKSPACE_CREATE	Creates the workspace. Using a *flow_mod* message adds a new flow identified by a specific VLAN id
WORKSPACE_ATTACH	Attaches an entity in a workspace and using a *OFPT_FLOW_MOD* message, updates the output ports to include this entity
ENTITY_UNREGISTER	Removes an entity from the DTS and updates the necessary flow tables
WORKSPACE_DETACH	Removes an entity from a existing workspace and updates flow tables accordingly
WORKSPACE_DELETE	Removes the flow entries regarding a workspace

A registered entity that wants to receive the media provided by the workspace, should attach to it by requesting a *WORKSPACE_ATTACH* message. This primitive also will be forwarded to the DTSA and in the same manner, by using the OpenFlow *OFPT_PACKET_IN* and *OFPT_FLOW_MOD* messages will modify the flow table to include the physical port of the requesting entity into the current workspace. Another entity could be attached to the workspace by pursuing the same procedure and becoming part of the sharing entities.

3. Concluding Remarks and Future Work

Considering the new set of requirements, Internet architecture must be reviewed. This process of revision using a *clean slate* can free researchers of current shortcomings, providing a rich environment for experimentations.

In this paper we have presented an SDN-based implementation of the DTS atop OpenFlow. Besides validating the DTS design, this implementation also shows how an IP focused OpenFlow switch, compliant with OpenFlow 1.0 specification, may be used in networks that completely drop the IP stack from the data plane by using a new semantic for the flow table.

We have started experimenting this prototype using the OFELIA testbed [OFELIA 2011] and reporting on these experiments is the subject of future work. Moreover, DTS plays an important role at networks aspects like naming, addressing and routing and an intelligent use of unicast and multicast in a end-to-end communication, provided by the workspace. This work has not covered such aspects of the DTS and discussing how they may integrate in this OpenFlow design is also the focus of future work.

References

[Big Switch 2012] Big Switch (2012). Developing floodlight modules. floodlight OpenFlow controller. http://floodlight.openflowhub.org/developing-floodlight/.

[Greene 2009] Greene, K. (2009). TR10: Software-Defined Networking. *MIT - Technology Review*.

[McKeown et al. 2008] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). OpenFlow: enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74. ACM ID: 1355746.

[OFELIA 2011] OFELIA (2011). OpenFlow in europe - linking infrastructure and applications.

[Pereira et al. 2011] Pereira, J. H. d. S., Silva, F. d. O., Lopes Filho, E., Kofuji, S. T., and Rosa, P. F. (2011). Title model ontology for future internet networks. In *Future Internet Assembly 2011: Achievements and Technological Promises*, volume 6656 of *LNCS*, page 465. Springer-Verlag, Future Internet: Achievements and Promising Technology.

[Silva et al. 2011] Silva, F. d. O., Pereira, J. H. d. S., Kofuji, S. T., and Rosa, P. F. (2011). Domain title service for future internet networks. In *Anais do II Workshop de Pesquisa Experimental na Internet do Futuro (WPEIF)*, Campo Grande. SBC.

[Zahariadis et al. 2011] Zahariadis, T., Papadimitriou, D., Tschofenig, H., Haller, S., Daras, P., Stamoulis, G. D., and Hauswirth, M. (2011). Towards a future internet architecture. In Domingue, J., Galis, A., Gavras, A., Zahariadis, T., and Lambert, D., editors, *The Future Internet. Future Internet Assembly 2011: Achievements and Technological Promises*, volume 6656 of *LNCS*, page 7–18. Springer-Verlag, Berlin, Heidelberg.