Modeling and Evaluation of Address Resolution Scalability in VPLS

Dominik Klein, Rastin Pries
University of Würzburg
{dominik.klein,pries}
@informatik.uni-wuerzburg.de

Michael Scharf, Michael Soellner
Alcatel-Lucent Bell Labs Germany
{michael.scharf,michael.soellner}
@alcatel-lucent.com

Michael Menth
University of Tübingen
menth@uni-tuebingen.de

***Abstract*—More and more services are provided by large data centers with a potentially very large number of physical or virtual hosts. As the number of hosted services and service consumers increases, also the number of hosts inside a data center raises to cope with the increasing end-user demand. Current data center networks are usually based on Ethernet and mechanisms like load balancing or redundancy between data centers require a transparent connection of these Ethernet networks over a Wide Area Network (WAN). Due to the large number of hosts, these interconnected data center networks face scalability problems on different protocol layers. One such issue, which is currently discussed within the IETF, is the scalability of the link layer Address Resolution Protocol (ARP).**

This paper studies the control traffic caused by address resolution for interconnected data centers. We develop an analytical model for the ARP traffic between data center locations that takes into account the number of hosts and connected sites. This model can then be used to quantify the ARP traffic for a data center interconnect solution. As an example, we apply our model to Virtual Private LAN Services (VPLS). In addition, we study how an ARP proxy can improve the overall scalability, and we show that a proxy significantly reduces the ARP traffic at VPLS switches.

I. Introduction

Current and emerging Internet applications are hosted in large data centers. Data center operators use several geographically dispersed locations for load sharing and resilience reasons. This, however, requires synchronization of the different data centers, and an efficient connection between the different locations is necessary.

Data centers often employ Ethernet as networking technology, and use server virtualization to run several virtual machines on one physical host. If a large number of nodes is attached to a data center network, the broadcast traffic caused by the Address Resolution Protocol (ARP) can result in scalability issues [1]. Although the scalability of address resolution in Ethernet networks can be improved by partitioning the network, e. g., into smaller Virtual Local Area Networks (VLANs), mechanisms like redundancy, load sharing, or virtual machine mobility require that a large subset of nodes is in the same VLAN ([1], [2]). Due to virtualization, more than 10000 nodes (either physical or virtual) may be connected to the same VLAN, which may also span more than one location. This requires a transparent interconnection of different data center sites, i. e., the transport of Ethernet frames over a Wide Area Network (WAN).

This work was conducted within the Internet Research Center (IRC) at the University of Würzburg. The authors alone are responsible for the content of the paper.

The address resolution scalability problem for large data center networks is currently discussed within the Internet Engineering Task Force (IETF) [3]. For data center interconnect solutions it is important to quantify the impact of broadcast traffic due to link layer address resolution. However, the authors are not aware of an analytical model that quantifies the amount of ARP traffic. Hence, in this paper, we model the address resolution traffic and then study the signaling load caused by this traffic on data center interconnect solutions. There are many different solutions to tunnel Ethernet frames over a WAN [4] and also ongoing standardization activities on further possibilities to exchange MAC address reachability information, but they are beyond the scope of this document [5]. We concentrate on Virtual Private LAN Services (VPLS) [6] as example and quantify the amount of ARP traffic at a VPLS edge switch. In addition, we show how an ARP proxy can improve the overall scalability at a VPLS edge switch.

The paper is structured as follows. In Section II, we give a brief introduction to VPLS and explain how VPLS handles ARP traffic. We then describe our analytical model for ARP traffic in Section III. In Section IV, we quantify the ARP traffic for VPLS and in Section V, we describe an ARP reduction mechanism and show how it improves the overall scalability for a VPLS edge switch. Finally, we give a short conclusion in Section VI.

II. Virtual Private LAN Services

In this section, we give an overview of VPLS, explain how unicast traffic is handled in general and then describe the broadcast handling by means of an ARP resolution process between two nodes in different data center Ethernet networks.

A. Architecture Overview

VPLS is a standardized mechanism to connect Ethernet domains over a Multiprotocol Label Switching (MPLS) core. VPLS is implemented in Provider Edges (PEs) and the PEs are connected via a full mesh of MPLS tunnels among each other.

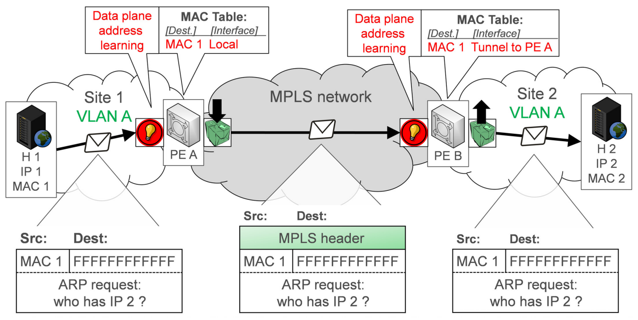

Fig. 1. ARP request between VPLS Ethernet domains.

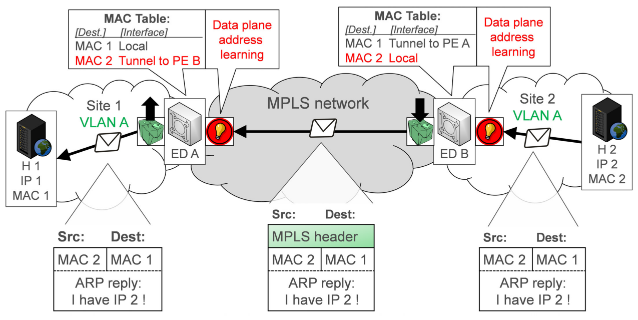

Fig. 2. ARP reply between VPLS Ethernet domains.

The provider edges apply data plane learning on all interfaces to learn the mapping from destination MAC address to outgoing interface. If the destination of an outgoing packet is not known, the packet is flooded via the full mesh to all other connected provider edges. To avoid loops in the full mesh of MPLS tunnels, a provider edge does not forward incoming packets from one MPLS tunnel to another MPLS tunnel. This is called the "split horizon" rule.

B. *Forwarding Procedure for Unicast Traffic*

In the following, we explain the forwarding procedure for unicast traffic at provider edges. We distinguish between outgoing and incoming Ethernet frames.

1) Data Plane Learning: If a provider edge receives an Ethernet frame on one of its interfaces, it first adds or updates the entry for the source MAC address in its local MAC table. The table stores the mappings from MAC address to outgoing interface for a specific VPLS instance. The frame is then further processed depending on the communication direction.

2) Outgoing frames from internal nodes: For outgoing frames received via one of the local interfaces, the provider edge checks in its internal MAC table, whether there is an entry for the destination MAC address. If there is an entry in the table, the provider edge forwards the Ethernet frame on the associated MPLS tunnel. In case there is no entry, the Ethernet frame is broadcast on all MPLS tunnels belonging to this VPLS instance. Therefore, the provider edge replicates the packet and forwards it on the appropriate MPLS tunnels.

3) Incoming frames from external nodes: For incoming frames received via one of the MPLS tunnels, the provider edge also checks in its internal MAC table, whether there is an entry for the destination MAC address. If the entry points to another MPLS tunnel, the Ethernet frame is discarded to avoid a possible loop in the full mesh of MPLS tunnels. If the entry points to an internal interface, the MPLS header is removed and the frame is forwarded on the internal interface.

In case there is no entry for the destination MAC address, the packet is broadcast on all interfaces except the interfaces pointing to an MPLS tunnel. This again avoids loops in the full mesh of MPLS tunnels.

C. *ARP Traffic Handling*

In this scenario, we describe the ARP traffic handling of VPLS by means of a communication example between two endhosts (H 1 and H 2) located in different customer sites, see Figure 1. Endhost H 1 has IP address IP 1 and a MAC address MAC 1. Endhost H 2 has IP address IP 2 and a MAC address MAC 2. H 1 knows the IP address of H 2 and the first step of the well-known ARP resolution is to get the MAC address for H 2. Therefore, H 1 sends an ARP request to discover the MAC address of H 2. The ARP request is sent to the broadcast MAC address FF:FF:FF:FF:FF:FF and the source address is the MAC address of H 1 (MAC 1). The frame arrives at PE A which then performs the VPLS forwarding process. First, PE A learns that MAC 1 can be reached locally and stores the appropriate entry in its MAC table. The destination MAC address is the broadcast MAC address, hence PE A floods the packet to all other PEs (PE B). PE B learns that MAC 1 can be reached via the MPLS tunnel to PE A and stores this information along with the MAC address in its MAC table. PE B then removes the MPLS label and floods the frame on its site-faced local interfaces. It does not flood the packet over MPLS tunnels because of the split horizon rule. Eventually, the broadcast frame arrives at H 2.

H 2 now responds to the ARP request with an ARP reply, see Figure 2. Therefore, H 2 sends a frame addressed to MAC 1 and uses its own MAC address MAC 2 as source address. The frame arrives at PE B, which learns that MAC 2 can be reached locally and stores this information in its MAC table. PE B already knows that MAC 1 can be reached via the MPLS tunnel to PE A and adds the appropriate MPLS header. The frame is then only forwarded to PE A, which receives the frame and learns that MAC 2 can be reached via the MPLS tunnel to PE B. PE A removes the MPLS header and forwards the frame according to the entry in its MAC table. Eventually, H 1 receives the frame.

III. Analytical Modeling for ARP Broadcasts

In this section, we present an analytical model for the rate of generated ARP broadcasts per node in an Ethernet domain. Even though ARP is a standard protocol, we are not aware of models for the resulting traffic. Therefore, we first describe the behavior of state-of-the-art ARP implementations. Then, we introduce our assumed scenario and our analytical model.

A. ARP Implementation Characteristics

In this paragraph, we explain some details of the ARP implementation of current operating systems. We verified the following behavior for Linux (Ubuntu 10.10) and Microsoft Windows 7 Professional. The current implementation of the ARP kernel module both in Windows [7] and Linux [8] follows RFC 4861 [9], which describes the network to link layer address resolution (Neighbor Discovery) in IPv6.

If a source host initiates a connection to a destination IP address with an unknown MAC address, ARP is used to obtain the MAC address. The source broadcasts an ARP request and the owner of the destination IP address answers with an ARP reply, which contains the mapping and is sent back to the source. The source host stores this information in an ARP cache. The destination host also stores the mapping for the source host in its own cache, but it first checks whether this mapping is valid. For UDP, this induces an ARP resolution in the opposite direction. For TCP, the ACK packet is sufficient and no additional ARP resolution process is initiated.

The ARP cache inside both hosts is used to reduce the number of ARP broadcast requests. To avoid outdated entries, each entry is assigned with a timeout. Once the timeout expires, the entry is removed from the ARP cache. In addition, each entry has a certain state which indicates the validity of the entry before the entry is eventually deleted. A newly created entry gets the *REACHABLE* state and can be used by higher layer applications without a refresh of its validity.

If the entry is not used by higher layer applications for a random time between $\frac{base\_reachable\_time}{2}$ and $3 \cdot \frac{base\_reachable\_time}{2}$, the state of the entry is changed to *STALE*. The parameter *base_reachable_time* is usually set by default to 30 s [9]. *STALE* entries need to be refreshed if they are used again. For connectionless protocols like UDP, this again requires an ARP request. However, this ARP request is then sent by unicast and not by broadcast. For connection-oriented protocols like TCP, ACK packets of successful connections can be used to refresh the entry. Refreshed entries get the *REACHABLE* state again. If a *STALE* entry is not refreshed, it is deleted after a certain time span. The duration depends on the configuration but a common value is 300 s.

In summary, for connection-oriented protocols like TCP, the above described behavior results in a timeout for cache entries of about 330 s. A new outgoing TCP connection induces an ARP broadcast request if this specific address has not been used for 330 s. For connection-less protocols like UDP, the timeout for cache entries is 30 s and entries need to be refreshed with an ARP request although they are in use. Hence, an outgoing UDP stream induces an ARP request every 30 s but only the first ARP request, if there is no entry in the ARP cache, is broadcast. Consecutive ARP requests are then sent per unicast. Regarding the ARP broadcast requests, UDP and TCP thus show the same behavior and an ARP broadcast request to a specific destination is only sent if the address has not been used for 330 s. Nevertheless, in the following sections we only consider the TCP behavior because most traffic in large data centers usually uses TCP. However, the model could also be adapted for UDP traffic.

B. Scenario and Assumptions

We assume a geographically dispersed data center with D locations and a maximum number of N connected nodes per entire VLAN. For the numerical evaluations, N is set to 10000. This results in $\frac{N}{D}$ nodes per data center location. Furthermore, we suppose that each node initiates flows to random destinations which leads to a certain rate of outgoing flows per node. This workload model is of course simple and it neglects many details about the complex load distribution mechanisms inside a data center, but it is difficult to provide a better model that is still generally applicable. Figure 2 in [10] confirms that a random workload scenario is a reasonable assumption. We vary the rate of outgoing flows per node λ_{node} between 10^{-4} and 10^{4} flows per second, in order to consider a wide range of possible load situations. Regarding the flow duration, we assume short flows that are at least one order of magnitude smaller than the cache timeouts in hosts. This assumption is valid because according to [11], 99 % of the flows in a data center transmit less than 100 MB, which results in a flow duration in the order of seconds.

C. Analytical Model: ARP Rate per Node

In this section, we model the ARP rate per node λ_{ARP} dependent on the rate of outgoing flows per node λ_{node}. We assume that the time A between two consecutive flows a node initiates towards another node is exponentially distributed with rate λ_{node}. According to [12], this is a reasonable approximation for the traffic within a data center. As there are $N-1$ other nodes in the VLAN, the considered node contacts a specific other node with a rate of $\frac{\lambda_{node}}{N-1}$. If the node does not find an entry in the ARP cache, it broadcasts an ARP request in the VLAN. This happens if the node initiated no other flow to that same destination for more than T_{host} time. Thus, the corresponding probability is

$$p_{ARP} = P(A > T_{host}) = e^{-\frac{\lambda_{node}}{N-1} \cdot T_{host}} \qquad (1)$$

so that the overall rate of ARP messages issued by a node is

$$\lambda_{ARP} = \lambda_{node} \cdot p_{ARP}. \qquad (2)$$

In addition to the ARP rate due to outgoing flows, we also assume a certain base rate λ_{base} of outgoing ARP requests per node. This base rate may be caused by configuration or signaling protocols like DHCP, automated updates, or administration tasks. It can be seen as a kind of ARP background noise and we assume a rate of 10^{-3} ARP requests per second per node. This rate is added to Equation 2 and we then get a total rate of ARP broadcasts per node as

$$\lambda_{ARP} = \lambda_{node} \cdot p_{ARP} + \lambda_{base}. \qquad (3)$$

In Figure 3, we plot λ_{ARP} against the rate of outgoing flows λ_{node} for $T_{host} = 330$ s. We also present results of a hypothetical value of $T_{host} = 1$ s to show how our model behaves for very

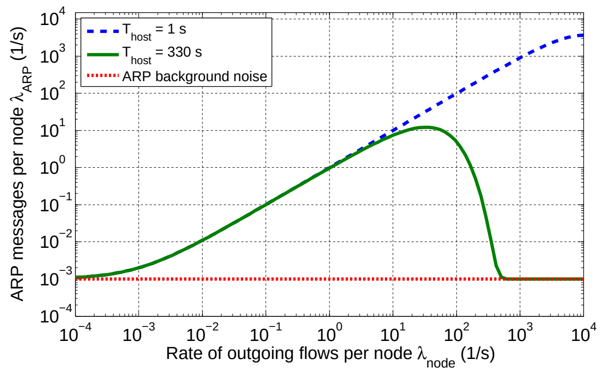

Fig. 3. Rate of outgoing ARP messages per node.

short timeout values. The x-axis shows the rate of outgoing flows per node λ_{node} and is logarithmically scaled. The y-axis shows the ARP rate per node λ_{ARP} and is also logarithmically scaled.

The ARP rate per node increases about linearly with the rate of outgoing flows per node λ_{node} for values below $\lambda_{node} < 10$ flows per second. It is almost identical because in that range, the node finds almost never a matching entry in its ARP cache as entries are deleted from the cache before the same destination is contacted again. For larger rates of outgoing flows per node, the ARP rate depends on the value for the timeout interval T_{host}. For $T_{host} = 330\,\text{s}$, the rate of ARP broadcasts rapidly decreases for rates λ_{node} larger than 10 flows per second. The smaller the caching duration T_{host}, the later the ARP rate drops to the base rate λ_{base}.

IV. Performance Evaluation

In this section, we combine the ARP handling for VPLS explained in Section II and the ARP model introduced in Section III to quantify the amount of received ARP request and ARP reply messages per VPLS switch that are caused by the address resolution mechanism.

A. *Total Address Resolution Traffic for a VPLS Switch*

In the following, we assume a benign environment in which ARP traffic is mainly triggered by the steady-state communication between the nodes inside the data center. We thereby neglect specific situations such as broadcast storms caused by failures, which are more difficult to model.

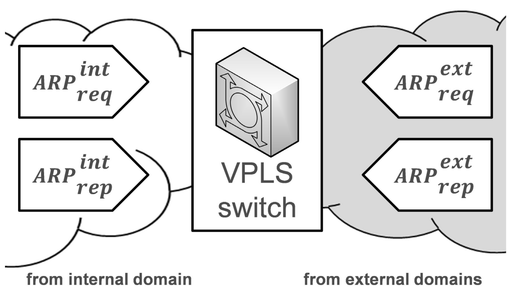

Fig. 4. Rate of received ARP messages per VPLS switch.

For the evaluation, we divide the total rate of received ARP messages per VPLS switch into four distinct rates (see Figure 4). The rates ARP_{req}^{int} and ARP_{rep}^{int} denote the number of outgoing ARP requests and outgoing ARP replies per second sent from internal nodes. The rates ARP_{req}^{ext} and ARP_{rep}^{ext} denote the number of incoming ARP requests and incoming ARP replies per second sent from external nodes.

In total, a VPLS switch receives all four rates which results in

$$ARP_{total} = ARP_{req}^{int} + ARP_{rep}^{int} + ARP_{req}^{ext} + ARP_{rep}^{ext} \tag{4}$$

ARP messages per second.

Concerning the ARP broadcast requests sent from internal nodes within one domain, the total rate is the individual rate per node times the number of nodes per location. The number of nodes per location is $\frac{N}{D}$ and hence the total rate of outgoing ARP requests per domain is

$$ARP_{req}^{int} = \frac{N}{D} \cdot \lambda_{ARP}. \tag{5}$$

These ARP requests are broadcast within the own location and arrive at the VPLS switch which further broadcasts the ARP requests to the different locations belonging to the same data center. Hence in addition, the VPLS switch receives the rate according to Equation 5 from each of its $D-1$ connected neighbors which results in

$$ARP_{req}^{ext} = \frac{N \cdot (D-1)}{D} \cdot \lambda_{ARP}. \tag{6}$$

incoming ARP requests per second sent from external nodes.

Concerning ARP unicast replies, a VPLS switch receives only those replies that are destined to an internal node in its own served location or those replies that are sent from an internal node in its own location. As we assume a random distribution of destinations, $\frac{1}{D}$ of the rate ARP_{req}^{ext} of incoming ARP requests are destined for an internal node. Hence, these nodes answer with an ARP reply which results in

$$ARP_{rep}^{int} = \frac{1}{D} \cdot ARP_{req}^{ext} \tag{7}$$

outgoing ARP replies per second sent from internal nodes.

In addition, as we again assume a random distribution of destinations, $\frac{(D-1)}{D}$ of the rate ARP_{req}^{int} of outgoing ARP request are destined for external nodes located in another domain. Hence, these nodes answer with an ARP reply which results in

$$ARP_{rep}^{ext} = \frac{(D-1)}{D} \cdot ARP_{req}^{int} \tag{8}$$

incoming ARP replies per second sent from external nodes.

B. *Numerical Results*

In Figure 5, we plot $ARP_{req}^{int} + ARP_{req}^{ext}$ and $ARP_{rep}^{int} + ARP_{rep}^{ext}$ for $D = 4$ domains. The dashed line denotes the ARP requests, the dotted line the ARP replies, and the solid line denotes the sum of both message types received at a VPLS switch.

For a low or very high load, i. e., for rates λ_{node} smaller than 0.1 or larger than 200 flows per second, the total rate of ARP messages received at a VPLS switch is below 1000

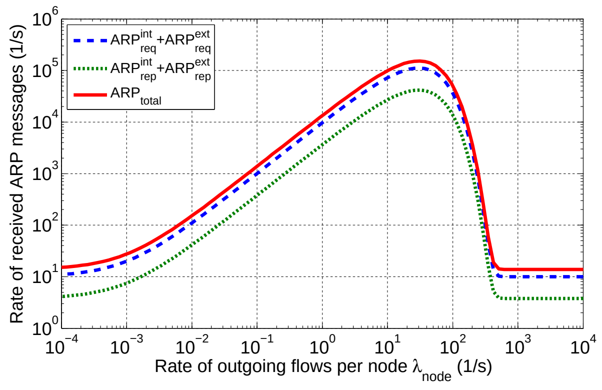

Fig. 5. Rate of received ARP messages per VPLS switch (D=4).

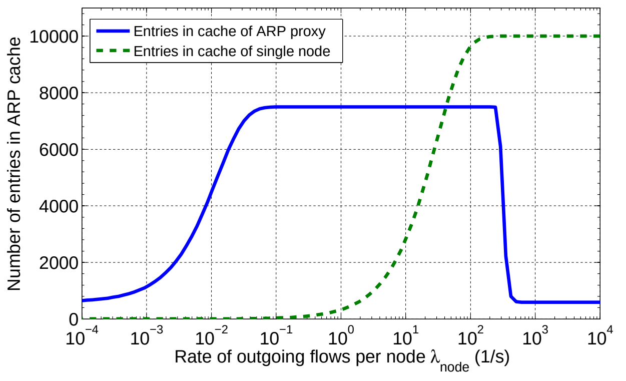

Fig. 6. Entries in cache of ARP proxy and in cache of single node.

ARP messages per second in our numerical example. However, between rates λ_{node} of 0.1 and 200 flows per second, the rate of ARP messages polynomially increases and reaches a maximum of $1.5 \cdot 10^5$ message for a rate λ_{node} of 31 flows per second.

V. Improvement by an ARP Proxy

An ARP proxy is a well-known solution to improve the scalability of Ethernet address resolution. In this section, we model an ARP proxy and we show how such a proxy in a VPLS switch reduces the number of ARP broadcast requests between data center sites.

A. ARP Proxy Basics

ARP proxies are usually implemented in customer edge switches. They snoop ARP traffic and cache the mappings from IP to MAC address seen in the ARP reply packets. The ARP proxy sees the ARP replies from nodes outside its own domain as these ARP replies pass through its interfaces. The cache inside the ARP proxy can thus be seen as an aggregate of the ARP caches of the nodes in the local domain. If a node in that domain asks for an already cached IP address, the ARP proxy generates an ARP reply locally rather than broadcasting the ARP request to other domains. As a result, the ARP proxy reduces the number of ARP broadcast requests between the different domains. A more detailed description of an ARP proxy can be found for example in [13].

B. Entries in an ARP Proxy Cache

In the following, we describe a model how to estimate the number of entries inside an ARP proxy. First of all, we need to calculate the rate of outgoing ARP requests λ_{dest} that arrive at an edge switch for an individual address from internal nodes. Since there are in total N nodes in this distributed data center, λ_{dest} can be calculated by dividing the total rate (see Equation 5) by N:

$$\lambda_{dest} = \frac{1}{D} \cdot \lambda_{ARP}. \tag{9}$$

The number of mapping entries inside the ARP proxy cache depends on the timeout T_{cache} for ARP cache entries inside the proxy. For the numerical results presented in the following, we assume the same value as for the ARP cache timeout in the nodes ($T_{cache} = 330\,\mathrm{s}$). To calculate the number of entries inside the cache, we first need the probability p_{entry} that a specific external address is stored in the cache. A mapping for a specific external address is stored in the cache if and only if the time A between two consecutive ARP requests for this address is smaller than the timeout interval T_{cache}. If we again assume that the time A is exponentially distributed with rate λ_{dest}, the corresponding probability is

$$p_{entry} = P(A \leq T_{cache}) = 1 - e^{-\lambda_{dest} \cdot T_{cache}}. \tag{10}$$

The total number of mapping entries is then

$$n_{proxy} = \frac{N}{D} \cdot (D-1) \cdot p_{entry}. \tag{11}$$

In Figure 6, we plot the number of mapping entries inside the proxy ARP cache for $D = 4$ domains.

As a reference, we also depict the number of entries in the ARP cache of one individual node. This value can be calculated similarly to Equations 10 and 11 by replacing λ_{dest} by $\frac{\lambda_{node}}{N-1}$ and by multiplying with $(N-1)$ instead of $\frac{N}{D} \cdot (D-1)$. This is because a cache in a node contains entries for both the external and the internal nodes. The number of entries in the ARP cache of a single node is thus

$$n_{node} = (N-1) \cdot (1 - e^{-\frac{\lambda_{node}}{N-1} \cdot T_{host}}). \tag{12}$$

For a low load, i. e., for rates λ_{node} smaller than 0.1 flows per seconds, the number of entries inside the ARP proxy cache slowly increase until they reach the maximum of $\frac{N}{D} \cdot (D-1)$ entries. In this parameter range, the increasing rate λ_{node} leads to an increasing rate λ_{dest} at the ARP proxy (see Equation 3 and 9) which then leads to an increasing number of cache entries in the ARP proxy (see Equation 10 and 11). For a medium load, i. e., for rates λ_{node} larger than 1 but smaller than 100 flows per second, the number of cache entries inside the node cache increase because of the increasing rate λ_{node} (see Equation 12). Hence, the number of sent ARP broadcasts per node decreases and the cache entries for external nodes shift from the ARP proxy cache to the node cache. For a high load, i. e., for rates λ_{node} larger than 100 flows per second,

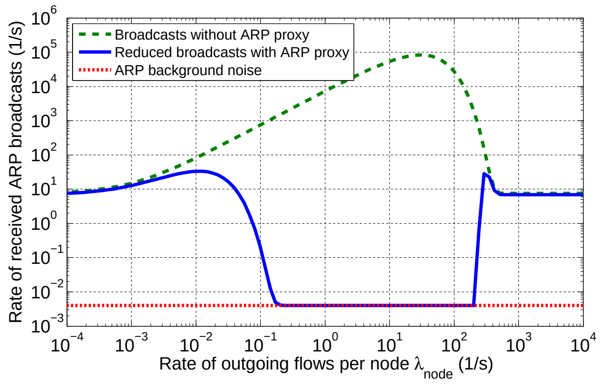

Fig. 7. Effect of proxy ARP mechanism on incoming ARP requests.

the ARP caches inside the nodes nearly contain all possible N destination addresses and hence no ARP broadcasts are sent anymore. The small number of cache entries for rates larger than 100 flows per second come from the assumed ARP background noise λ_{base} (see Equation 3).

C. ARP Broadcast Reduction with ARP Proxy

Now, we apply the model for the ARP proxy to the receiving rate of incoming ARP requests ARP_{req}^{ext} at a VPLS edge switch. The outgoing rate ARP_{req}^{int} of ARP requests sent from internal nodes cannot be reduced with an ARP proxy because these messages originate within the local domain and always arrive at the VPLS Switch. Hence in the following, these messages are not shown. The ARP proxy of a site broadcasts an ARP request to the other sites only if it cannot find a matching entry for that request in its ARP cache. Hence, the probability $p_{broadcast}$ is complementary to the probability p_{entry} that there is an entry inside the ARP proxy cache. The reduced received rate is then

$$ARP_{req}^{ext\prime} = (1 - p_{entry}) \cdot ARP_{req}^{ext}. \tag{13}$$

In Figure 7, we plot the receiving rate with and without ARP proxy mechanism for $D = 4$ domains. The dashed line shows the received rate without ARP proxy and the solid line shows the received rate with ARP proxy. Again, we assume some kind of ARP background noise per domain.

According to Figure 7, there is a significant reduction of the ARP traffic for values of λ_{node} between 0.01 and 100 flows per second. In this interval, nearly all external destination mappings are stored in the ARP cache and hence no interdomain ARP broadcasts are necessary, except for ARP background traffic. For lower and higher rates, not all entries are cached in the ARP proxy and hence the probability to broadcast an ARP request increases. For these rates, the ARP proxy is not as efficient as for the interval between 0.01 and 100 flows per second.

In summary, our analytical model confirms that an ARP proxy is an effective method to reduce the ARP traffic between data center sites, and that it may make sense to deploy such proxies in VPLS edge switches.

VI. Conclusion

In this paper, we investigated the scalability of the link layer to network layer address resolution mechanism in distributed data centers. This issue is discussed in the research community and also in the IETF, but good models for this traffic are still missing. Therefore, we first gave a brief overview of current ARP implementations and proposed an analytical model for the ARP traffic between data center locations that takes the number of nodes and connected sites into account.

As an application scenario, we applied this model to VPLS to quantify the amount of ARP request and ARP reply messages received at a VPLS switch. In addition, we studied how an ARP proxy can improve the overall scalability by reducing the ARP broadcast traffic. However, our proposed model is not specific to VPLS and could be adapted also for other data center interconnect solutions or for ARP traffic prediction within a single data center.

Acknowledgements

The authors would like to thank Prof. Tran-Gia for the stimulating environment and fruitful discussions.

References

[1] L. Dunbar, S. Hares, M. Sridharan, N. Venkataramaiah, and B. Schliesser, “Address Resolution for Large Data Center Problem Statement,” IETF Internet-Draft, work in progress, Mar. 2011.
[2] Y. Li, “Problem Statement on Address Resolution in Virtual Machine Migration,” IETF Internet-Draft, work in progress, Mar. 2011.
[3] IETF Working Group , “Address Resolution for Massive numbers of hosts in Data center (Active WG),” http://tools.ietf.org/wg/armd/, Oct. 2011.
[4] P. Knight and C. Lewis, “Layer 2 and 3 Virtual Private Networks: Taxonomy, Technology, and Standardization Efforts,” *Communications Magazine, IEEE*, vol. 42, no. 6, pp. 124 – 131, 2004.
[5] R. Aggarwal, A. Sajassi, W. Henderickx, A. Isaac, J. Uttaro, N. Bitar, R. Shekhar, F. Balus, S. Boutros, and K. Patel, “BGP MPLS Based Ethernet VPN,” IETF Internet-Draft, work in progress, Sep. 2011.
[6] M. Lasserre and V. Kompella, “Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling,” IETF RFC 4762, Jan. 2007.
[7] Microsoft Knoweledge Base, “Description of Address Resolution Protocol (ARP) Caching Behavior in Windows Vista TCP/IP Implementations,” Jan. 2010.
[8] Linux Man-Pages Project, “Description of ARP Kernel Module in Release 3.24.”
[9] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, “RFC4861: Neighbor Discovery for IP version 6 (IPv6),” IETF RFC 4861, Sep. 2007.
[10] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, “The Nature of Data Center Traffic: Measurements & Analysis,” in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*. New York, NY, USA: ACM, 2009, pp. 202–208.
[11] A. Greenberg, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, “VL2: A Scalable and Flexible Data Center Network,” in *ACM SIGCOMM*, Barcelona, Spain, Aug. 2009.
[12] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, “Data center TCP (DCTCP),” *ACM SIGCOMM Computer Communications Review*, vol. 40, pp. 63–74, August 2010. [Online]. Available: http://doi.acm.org/10.1145/1851275.1851192
[13] K. Elmeleegy and A. Cox, “EtherProxy: Scaling Ethernet By Suppressing Broadcast Traffic,” in *IEEE Infocom*, Rio de Janeiro, Brazil, Jun. 2009, pp. 1584 – 1592.