Reducing Keepalive Traffic in Software-Defined Mobile Networks with Port Control Protocol

Kamil Burda, Martin Nagy[✉], and Ivan Kotuliak

Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Ilkovičova 2, 842 16 Bratislava, Slovakia
{xburdakamil,martinko.nagy}@gmail.com, ivan.kotuliak@stuba.sk

Abstract. User applications, such as VoIP, have problems traversing NAT gateways or firewalls. To mitigate these problems, applications send keepalive messages through the gateways. The interval of sending keepalives is often unnecessarily short, which increases the network load, especially in mobile networks. Port Control Protocol (PCP) allows the applications to traverse the gateways and to optimize the interval. This paper describes the deployment of PCP in software-defined networks (SDN) and proposes a method to measure keepalive traffic reduction in mobile networks using PCP. The proposed solution extends the battery life of mobile devices and reduces the traffic overhead in WCDMA networks.

Keywords: Middleboxes · Keepalives · Port Control Protocol · Mobile networks · Software-defined networking

1 Introduction

User applications that require long-term connections, such as Voice over IP (VoIP), Instant Messaging or online gaming, may have problems establishing connections if hosts running the applications are located behind network address translation (NAT) gateways or firewalls, hereinafter referred to as *middleboxes*.

For each connection, a middlebox contains a mapping entry that is manually configured or dynamically created when the connection is being established. In case of NAT gateways, the mapping entry usually consists of the following fields: internal IP address, external IP address, internal port, external port and mapping lifetime.

If a connection is idle for longer than the corresponding mapping lifetime, the middlebox blocks the connection without notifying the communicating hosts. To keep the connection alive, the application sends keepalive messages (such as empty TCP or UDP datagrams) toward the destination host. Because the application does not know the exact connection timeout, keepalives are sent in very short intervals, which increases the network load. The unnecessarily high amount of the keepalive traffic reduces battery lifetime on mobile devices, especially those connected to mobile networks, where each message sent imposes additional overhead in the form of signaling traffic.

This paper proposes a network architecture to deploy the Port Control Protocol (PCP) in the core of software-defined mobile networks and a method to measure the keepalive traffic reduction with PCP in WCDMA networks.

The rest of this paper is structured as follows. Section 2 briefly reviews existing NAT traversal and keepalive reduction methods. Section 3 describes the basics of the PCP protocol and the advantages of the deployment of PCP in SDN networks. Section 4 describes the architecture of the core network and its components. Section 5 describes the method to measure the battery life extension of mobile devices and signaling traffic reduction in WCDMA networks [1,8]. The final section provides concluding remarks and challenges for future work.

2 Related Work

Protocols such as Session Traversal Utilities for NAT (STUN) [2], Traversal Using Relays around NAT (TURN) [3] or Interactive Connectivity Establishment (ICE) [4] can resolve NAT traversal issues for user applications. Additional methods for proper NAT traversal are defined for IPSec ESP [5] and mobile IP [6].

A method proposed in [7] aims to reduce the keepalive traffic in mobile IPv4 networks and in IPSec communication by replacing UDP keepalives with the so-called TCP wake-up messages, given the considerably greater mapping lifetime for TCP connections on NAT and firewall devices from popular vendors [8]. The results of the experiments conducted suggest that the keepalive traffic reduction is significant in 2G (GSM) and 3G (WCDMA, HSDPA) networks, but not in IEEE 802.11 Wireless LAN [7].

3 Port Control Protocol

PCP [9] allows IPv4 and IPv6 hosts to determine or explicitly request network address mapping, port mapping and mapping timeout (also called *mapping lifetime*) directly from middleboxes. From this information, a host behind a middlebox can establish communication with a host in an external network or in another internal network behind another middlebox and can optimize the interval of sending keepalives. PCP does not replace the function of proxy or rendezvous servers to establish connections between hosts in different internal networks. PCP requires that hosts run a PCP client and middleboxes run a PCP server [9].

Based on the existing research [7], the reduction of the keepalive traffic in mobile networks can be considerable. PCP introduces a more universal approach that allows to optimize keepalive traffic for multiple transport protocols (any protocol with 16-bit port numbers) and other upper-layer protocols, such as ICMP or IPSec ESP [9].

PCP may be vulnerable to security attacks such as denial of service or mapping theft [9]. The security of PCP is currently under discussion [10]. An RFC draft specifies an authentication mechanism to control access to middleboxes [11].

3.1 Port Control Protocol in Software-Defined Networks

Software-defined networking (SDN) [12–14,19] is a novel approach to managing computer networks which separates the control and data planes of network devices to controllers and forwarders, respectively, and achieves greater network flexibility by allowing to program the network behavior. Existing networks are expected to migrate to SDN given the aforementioned advantages.

With SDN, a PCP server can run on a controller, thereby reducing the processing overhead on middleboxes, increasing vendor compatibility and avoiding the need to upgrade the middleboxes to support PCP server functionality. If multiple middleboxes are placed in an SDN network, mapping lifetime can be determined from the controller instead of every middlebox separately. There is an ongoing effort to support advanced firewall functionality in SDN networks by introducing new PCP message types [15].

4 Network Architecture

This section describes the architecture of the SDN-based mobile core network, which incorporates PCP to reduce the signaling traffic. The essential components of the architecture are shown in Fig. 1. For the implementation, OpenFlow [14,16] is used as the communication protocol between the controller and the forwarders.

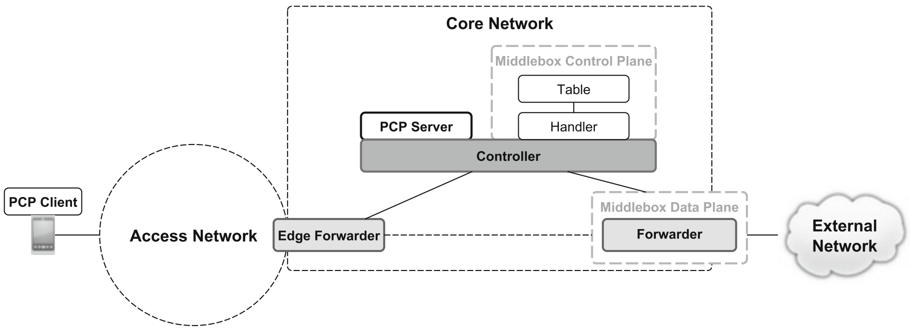

Fig. 1. Architecture of the proposed network.

An end host running a user application with a PCP client is located behind an existing access network (such as UTRAN in case of 3G networks). The access network connects to the core network via the edge forwarder. The edge forwarder forwards PCP requests to the controller, PCP responses from the controller back to the PCP client and other traffic further through the core network.

In the proposed architecture, the control and the data plane of a middlebox are decoupled. The middlebox data plane resides on another forwarder, placed between the core and the external network (the Internet). The middlebox data plane executes the rules installed by the control plane, such as overwriting IP addresses and transport protocol ports in packets in case of NAT.

The controller runs the middlebox control plane, which is responsible for maintaining mappings stored in a table. The handler accepts requests from the PCP server to create or remove a mapping and instructs the controller to add or remove the corresponding rules on the forwarder.

The PCP server running on the controller receives PCP requests, instructs the middlebox control plane to create a mapping for the client and sends PCP responses back to the client once the middlebox control plane successfully creates a mapping. The PCP server address is assumed to be the address of the default gateway, so PCP clients must use this address to communicate with the PCP server. Dynamic PCP server discovery options [9,17,18] are currently not considered.

In order to verify the proper traversal of packets behind middleboxes and the keepalive traffic reduction, a custom, simple NAT gateway is implemented in the network that supports only IPv4 addresses and TCP and UDP as the upper-layer protocols. A custom firewall, IPv6 or other upper-layer protocols are not implemented in the network, as the verification and evaluation method of the proposed solution is identical and would not affect the results.

The control plane of the NAT is responsible for creating NAT table entries from the configured pool of external IP addresses and ports. Each NAT table entry contains the following items: internal IP address, internal port, external address, external port, upper-layer protocol and mapping lifetime. The NAT data plane is represented as a set of flow tables and entries shown in Fig. 2.

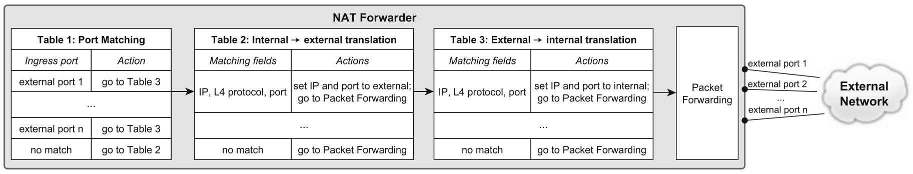

Fig. 2. Flow entries in the NAT forwarder.

The design does not address the security of PCP. The authentication mechanism specified in [11] could be used to control the access to the controller running the PCP server.

5 Evaluation

This section describes a method to measure the battery life extension and signaling traffic reduction based on the values of keepalive intervals to prove the feasibility of the deployment of PCP in WCDMA networks. The method is based on the measurements performed by Haverinen et al. [8] and Signals Research Group, LLC [1], both performed in WCDMA networks.

5.1 Requirements

No other network data, except keepalives, are sent over the network. This is done to isolate the useful network traffic that is usually unpredictable in practice, which would distort the results.

CELL_PCH and CELL_FACH Radio Resource Control (RRC) states are assumed to be enabled in the WCDMA network. When sending a keepalive, the mobile device uses the following RRC state transition: CELL_FACH → CELL_PCH → CELL_FACH → CELL_PCH → ...

The WCDMA inactivity timers are assigned the values used in the first measurement in [8]. In particular, the T2 inactivity timer (causing transition from CELL_FACH to CELL_PCH) is set to 2 s. Keepalives must be sent one at a time, until the mobile device re-enters the lower RRC state (with lower power consumption).

To quantify the battery life saving and signaling traffic reduction, reference values must be defined. For example, suppose that an application currently uses a keepalive interval of 20 s (such as IPSec ESP [5]). If the keepalive interval is increased, the mobile device consumes that much less battery charge compared to the original (reference) keepalive interval. Likewise, the network and the device generate fewer signaling messages.

5.2 Battery Power Saving

Battery Consumption Figures. The first experiment in [8] consisted of sending one keepalive at a time. In the experiment, the inactivity timers and the RRC state transitions were identical to those specified in the Sect. 5.1. The results showed that the average current in the CELL_FACH state is 120 mA (disregarding the negligible variance of the current due to the actual data transmission), and the cost of a single keepalive in the 3G WCDMA network ranged from 0.15 to 0.6 mAh.

Method. Let T be the time period over which the measurement is performed. Over time period T, n_{ref} keepalives are sent given the original (reference) keepalive interval t_{ref} (i.e. the user application originally used the interval t_{ref}). Likewise, n keepalives are sent given the new keepalive interval t_{new}. The number of keepalives sent can be computed as $n = T/t$, where $1/t$ is the number of keepalives sent per second.

The amount of battery consumption saved (in mAh) can be determined as follows:

$$reduction\,(t_{new}) = k_{ref} - k = (n_{ref} - n) \cdot cost = cost \cdot T \cdot \left(\frac{1}{t_{ref}} - \frac{1}{t_{new}} \right) \quad (1)$$

where $k = n \cdot cost$ is the total cost of keepalives over time T.

In order to determine the battery power saving given the desired and reference keepalive intervals (t_{new} and t_{ref}, respectively), the battery capacity C of the mobile device (in mAh) must be known. The relative amount of battery life consumed by sending keepalives can be determined as k/C.

By increasing the keepalive interval to t_{new}, the amount of the battery life saved, given the battery capacity C, can be determined as follows:

$$battery\ power\ saved = \frac{k_{ref}}{C} - \frac{k_{new}}{C} = \frac{reduction\,(t_{new})}{C} \tag{2}$$

From the Eq. (2), one can conclude that, by using a higher keepalive interval t_{new}, such percentage of battery consumption was saved over time T.

From the end-user perspective, an alternative measure may better indicate the power consumption reduction: how much longer the battery will last before recharging it. Suppose that the cost of a single keepalive ($cost$) and the average current while sending a single keepalive ($\bar{I}_{keepalive}$) are known. The total time of the battery life saved can then be computed as follows:

$$battery\ life\ saved = (n_{ref} - n) \cdot \frac{cost}{\bar{I}_{keepalive}} = \frac{reduction\,(t_{new})}{\bar{I}_{keepalive}} \tag{3}$$

Results. Figure 3 shows the battery power saving with increasing keepalive interval, given the time period, cost of a single keepalive, battery capacity and the reference keepalive interval of 20, 40, 80 and 120 s, respectively.

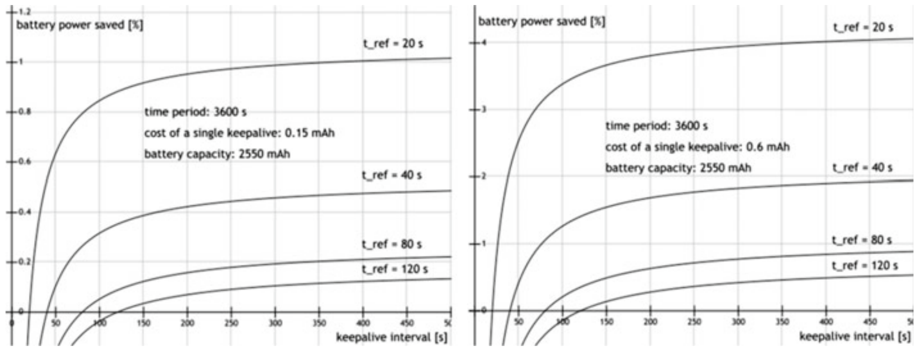

Fig. 3. Battery power saving based on keepalive intervals relative to reference values.

The percentage of the battery power saving increases significantly when the keepalive interval is increased by the first few tens of seconds from the reference interval. Above 400–600 s, the difference in the increase starts to be negligible.

Table 1 shows the percentage of the battery power saving for the chosen representative values of battery capacity, reference values and the keepalive interval of 400 s.

If the application running on a smartphone with battery capacity of 2550 mAh (Samsung Galaxy S6) originally used the keepalive interval of 20 s, 1–4 %

Table 1. Battery power saved for a mobile device connected a WCDMA network given battery capacity and the following reference values: $t_{ref} = 20$ s, $t_{new} = 400$ s, $T = 3600$ s, $cost : 0.15 - 0.6$ mAh.

Battery capacity [mAh]	Battery power saved [%]
300 (Samsung Gear S smart watch)	8.5–34.2
2550 (Samsung Galaxy S6 phone)	1–4
7340 (iPad Air 2 tablet)	0.35–1.4

of the battery life can be saved over 3600 s for the cost ranging from 0.15 to 0.6 mAh. For the reference interval of 40 s, the battery power saving is halved. The battery power saving proves to be significant for devices with relatively low battery capacity, such as smart watches (provided that they support WCDMA), and less significant for devices with higher battery capacity, such as tablets.

If the battery lifetime saving is considered, approx. 13–52 min of battery life can be saved for the keepalive cost ranging from 0.15 to 0.6 mAh, the average current of 120 mA (CELL_FACH state) [8], the reference keepalive interval of 20 s and the new keepalive interval of 400s .

5.3 Signaling Traffic Reduction

Signaling Traffic Figures. In [1], several measurements were performed in two 3G WCDMA networks, observing the number of signaling messages generated in the networks and the battery power consumption in mobile devices. In one of the measurements, the mobile devices sent keepalive messages to the network. In the observed networks, the mobile devices entered the CELL_DCH state when sending a keepalive. According to the results, sending one keepalive causes 40–50 signaling messages to be exchanged between a mobile device and the network (referred to as "observed" messages), and estimated 20 signaling messages generated in the network not captured on the mobile device (referred to as "unobserved" messages).

Method. Let s be the number of signaling messages sent per a single keepalive. The total number of signaling messages sent over time T given keepalive interval t_{new} is $S = n \cdot s$. The reduction of signaling messages in the network with increased keepalive interval can then be computed as:

$$reduction\,(t) = S_{ref} - S = (n_{ref} - n) \cdot s = \left(\frac{1}{t_{ref}} - \frac{1}{t_{new}}\right) \cdot s \cdot T \qquad (4)$$

Results. As seen in Fig. 4, the reduction of the number of signaling messages grows rapidly up to the keepalive interval of approx. 400 s. The growth of the reduction starts to be negligible from approx. 1800 s, which can be considered an acceptable keepalive interval for WCDMA networks. Table 2 quantifies the results for reference keepalive intervals of 20 and 120 s.

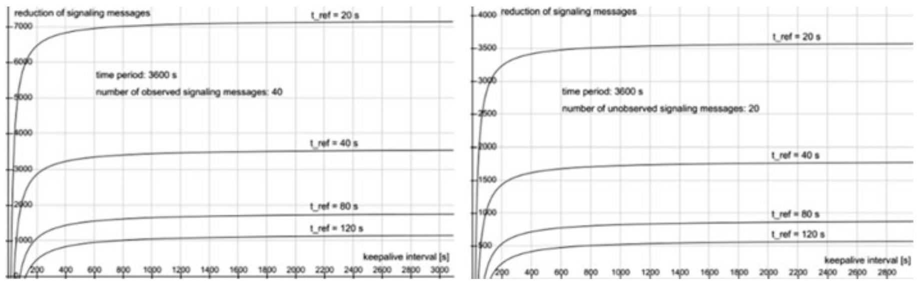

Fig. 4. Number of signaling messages reduced based on keepalive intervals and reference values.

Table 2. Number of signaling messages reduced given the following reference values: $t_{new} = 1800$ s, $T = 3600$ s.

Number of signaling messages per keepalive	Reference keepalive interval [s]	Number of signaling messages reduced	Reference keepalive interval [s]	Number of signaling messages reduced
40 (observed)	20	7120	120	1120
50 (observed)	20	8900	120	1400
20 (observed)	20	3560	120	560

It should be noted that the reduction of the number of signaling messages was computed for one mobile device running a single application. Considering that hundreds of thousands of mobile devices are connected to a network, each running one or more always-on applications, the decrease in the network load on elements in the network core may prove to be significant.

5.4 Determining PCP Mapping Lifetime

From the perspective of a mobile device and its battery life, the keepalive interval of 400–600 s is suitable for most applications. When considering the amount of signaling traffic generated in a mobile network, the keepalive interval of approx. 1800 s is sufficient to greatly reduce the signaling traffic.

For mappings created by PCP requests with the MAP opcode (i.e. user applications function as servers), user applications must send PCP MAP requests at the interval of at least 1/2 of the mapping lifetime [9]. In order to sustain the interval of 1800 s, the mapping lifetime for PCP MAP mappings should be doubled, i.e. set to 3600 s. Beside PCP MAP requests, applications may still have to send keepalives to the destination host to maintain the end-to-end connectivity. In order to keep the number of RRC state transitions to a minimum, applications should send PCP MAP requests to the PCP server and keepalives to the destination host at the same time.

For mappings created by PCP PEER requests, and given the relatively high keepalive interval of 1800 s suitable for WCDMA networks, it may be sufficient for the application to send the keepalives 7/8 of the mapping lifetime. Therefore, the mapping lifetime for PCP PEER mappings could be approx. 2100 s.

6 Conclusions

This paper described the architecture for the deployment of the Port Control Protocol in software-defined networks. Using the SDN approach, this architecture separates the control and the data plane of middleboxes and allows to run the PCP server outside the middleboxes according to SDN principles. This improves vendor device compatibility and avoids the processing overhead imposed by running the PCP server.

With PCP deployed in the network, mobile devices connected to mobile networks can reduce the amount of keepalive traffic sent, which results in extended battery life of mobile devices. Additionally, the network throughput is increased due to signaling traffic reduction in access networks. The keepalive interval of approx. 1800 s proves to be suitable for most applications in WCDMA networks. The battery power saving by using higher keepalive intervals is more significant in devices with relatively small battery capacity, such as smartphones and smart watches. Given the recommended keepalive interval, PCP server should assign mapping lifetime of at least 3600 s for PCP MAP mappings and 2100 s for PCP PEER mappings.

Acknowledgments. This work is a result of the Research and Development Operational Program for the projects Support of Center of Excellence for Smart Technologies, Systems and Services, ITMS 26240120005 and for the projects Support of Center of Excellence for Smart Technologies, Systems and Services II, ITMS 26240120029, co-funded by ERDF. This project was also partially supported by the Tatra banka Foundation under the contract No. 2012et011.

References

1. Signals Research Group, LLC: Smartphones and a 3G Network, reducing the impact of smartphone-generated signaling traffic while increasing the battery life of the phone through the use of network optimization techniques, May 2010
2. Wing, D., et al.: Session Traversal Utilities for NAT (STUN). RFC 5389, October 2008
3. Matthews, P., et al.: Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN). RFC 5766, April 2010
4. Rosenberg, J.: Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Offer/Answer Protocols. RFC 5245, April 2010
5. Huttunen, A., et al.: UDP Encapsulation of IPsec ESP Packets. RFC 3948, January 2005

6. Levkowetz, H., Vaarala, S.: Mobile IP Traversal of Network Address Translation (NAT) Devices. RFC 3519, April 2003
7. Eronen, P.: TCP Wake-Up: Reducing Keep-Alive Traffic in Mobile IPv4 and IPsec NAT Traversal. Nokia Research Center (2008)
8. Haverinen, H., et al.: Energy consumption of always-on applications in WCDMA networks. In: 2007 IEEE 65th Vehicular Technology Conference, VTC2007-Spring, pp. 964–968, April 2007
9. Boucadair, M., et al.: Port Control Protocol (PCP). RFC 6887, April 2013
10. Wing, D.: Port control protocol. The Internet Protocol Journal **14**(4)
11. Reddy, T., et al.: Port Control Protocol (PCP) Authentication Mechanism. Internet-Draft draft-ietf-pcp-authentication-07, December 2014
12. Open Networking Foundation: Software-Defined Networking: The New Norm for Networks. https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf. [Online; Accessed 30th November 2014]
13. Nadeau, T.D., Gray, K.: SDN: Software Defined Networks. O'Reilly Media, Sebastopol (2013)
14. Nagy, M., Kotuliak, I.: Utilizing openflow, SDN and NFV in GPRS core network. In: Leung, V.C.M., Chen, M., Wan, J., Zhang, Y. (eds.) TridentCom 2014. LNICST, vol. 137, pp. 184–193. Springer, Heidelberg (2014)
15. Reddy, T., et al.: PCP Firewall Control in Managed Networks. Internet-Draft draft-reddy-pcp-sdn-firewall-00, December 2014
16. Open Networking Foundation: OpenFlow Switch Specification, version 1.3.0. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf. [Online; Accessed 5th December 2014]
17. Boucadair, M., et al.: DHCP Options for the Port Control Protocol (PCP). RFC 7291 (2014)
18. Penno, R., et al.: PCP Anycast Address. Internet-Draft draft-ietf-pcp-anycast-02, August 2014
19. Skalný, J. et al.: Application of software defined networking (SDN) in GPRS network. In: IIT SRC 2014: Student Research Conference in Informatics and Information Technologies, pages 553–558, Bratislava. Nakladateľstvo STU, April 2014