Towards a Context Aware Multipath-TCP

Richard Withnell
School of Computing and Communications
Lancaster University
r.withnell@lancaster.ac.uk

Christopher Edwards
School of Computing and Communications
Lancaster University
c.edwards@lancaster.ac.uk

***Abstract*—Multi-homing has become ubiquitous in the mobile domain. Despite this, mobile devices are not yet capable of fully realizing the potential of the increasing variety and availability of network resources. Multipath-TCP (MPTCP) is one of the most prominent technologies for multi-homed mobile devices, that is able to effectively tackle both resource pooling and mobility. Despite the success of MPTCP we believe that the implementation can be enhanced for the mobile domain, by increasing the granularity of control a device has over how its network interfaces are used. In this paper we present and evaluate MPTCP - Context Aware (MPTCP-CA), which introduces a policy oriented approach to the management and control of MPTCP connections. Our evaluation demonstrates three use cases that leverage context awareness to improve the utilization of multiple network interfaces with respect to the users requirements.**

***Index Terms*—Mobile Devices, Multipath-TCP, Network Management, Context Aware, Cross Layer**

I. Introduction

In the mobile domain demand for a variety of high bandwidth applications such as media streaming, cloud storage and real time video conferencing has seen a vast increase in recent years and is predicted to quadruple by 2019 [1]. This increase in demand for bandwidth while mobile can not be met by the cellular network alone, leading to a number of off-loading solutions, combining a range of access technologies such as WiFi and Femto Cells. Despite the presence and availability of access points enabling off-loading the network stack is still not equipped to effectively balance and migrate traffic between the available network connections. Typical policies used by smart devices are based on the availability of an access medium, as opposed to dynamically allocating the network resource based on quality or application demand. For example, the cellular network is commonly used as a fall back when no WiFi connectivity is available. To improve the current mobile connectivity model we propose three requirements for the network stack;

1) ***Mobility*** – Both horizontal and vertical handover should be efficient, seamless and have minimal impact on the users quality of experience.
2) ***Resource Pooling*** – Combine the devices available network resource to help meet the current demand for high bandwidth applications as proposed in [2].
3) ***Intelligent Resource Usage*** – Adaptively allocate network resource based on the requirements of the applications and the mobile device.

The most prolific effort to date in addressing these issues is Multipath-TCP (MPTCP) [3]. Current MPTCP research has mainly focused on the resource pooling aspect of using multiple network interfaces simultaneously[4], as well as the benefits of enabling seamless and efficient mobility [5] [6]. MPTCP in its current form however fails to address our final goal for mobile connectivity, this can lead to sub-optimal usage of the available network resource in regards to the users requirements. To address the problem of intelligent resource usage we propose Multipath-TCP - Context Aware (MPTCP-CA). MPTCP-CA provides a new cross-layer approach to the management and utilization of a mobile devices network resource, with a core focus on leveraging context awareness combined with a high level policy definition. We believe that by exploiting context and taking into account user policies for network resource usage, we can not only improve the quality of experience of the users applications, but also improve the ability for a smart device to meet the users needs.

In this paper we first provide a background of the current state of Multipath-TCP. We then go on to present our proposed architecture for MPTCP-CA, including a discussion on how context and policy decisions can be applied to improve the efficient utilization of network resources. Finally, we present a preliminary evaluation, demonstrating three use cases in which we use MPTCP-CA to exploit multi-homing and context awareness; including application priority, power consumption and wireless link quality.

II. Multipath-TCP

Multipath-TCP [3] is the foremost attempt to both standardize and implement a deployable multipath extension to TCP. MPTCP functions by creating a number of TCP subflows between two remote end-points. MPTCP has two key advantages over single path TCP; multiple network interfaces can be used simultaneously to increase throughput and the use of multiple independent paths improves resiliency, in the event of failure. Additionally MPTCP is transparent to the application, requiring no modification to take advantage of multipath communications.

A. Interface Management

The current implementation of MPTCP provides a simple interface controlling the MPTCP capability of each network interface. Three different options are currently available, a specific interface can either be set to enabled, disabled or to

978-1-4673-6770-7/15/$31.00 ©2015 IEEE

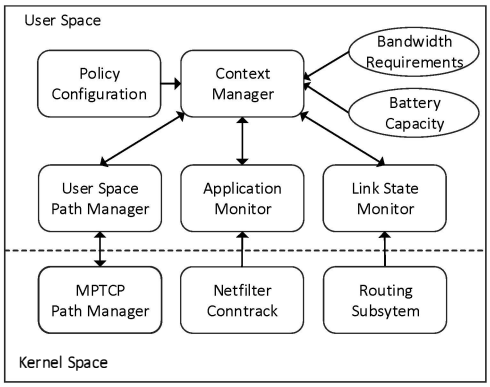


Fig. 1: Architecture for the MPTCP-CA implementation.

act as a backup. In backup mode the interface is only used in the event no other network interface is available to be used.

B. Path Managers

The path manager module of the Linux MPTCP implementation, is responsible for controlling the establishment of TCP subflows. This section discusses some of the key implementations of alternative MPTCP Path Managers.

1) Full Mesh: The full mesh path manager creates a TCP subflow for every pair of IP addresses present at the two end points. The server advertises its available IP addresses to the client and for each advertised address the path manager creates a subflow for each of its own addresses. This path manager attempts to take advantage of both resource pooling and path diversity, to improve both bandwidth and resilience.

2) ndiffports: The ndiffports path manager is very simple, it creates a pre-determined number of subflows between a single pair of IP addresses between the hosts. This path manager can be beneficial in a data center environment, in which the hosts are single-homed but the network load balances TCP flows across redundant paths.

3) Cross Layer: To the best of our knowledge at present there are only two other proposals for cross-layer MPTCP path managers. In [7] the author focuses on using link quality information at the MAC layer to improve the decisions about how subflows are managed. Based on link quality estimation subflows are labeled either active or inactive. If the link quality is poor the subflow becomes inactive and its in-flight packets are re-transmitted using other available subflows. In [8], the authors propose that information from the user space can be used to inform the number of subflows that are created in a cloud environment. For example, if there are multiple redundant paths in a cloud infrastructure the user space application informs the MPTCP path manager of the available paths, leading to the optimal number of subflows being created.

III. Multipath-TCP - Context Aware

At the core of MPTCP-CA is a new cross layer path manager that is able to create and destroy subflows over specific network interfaces or paths as the context of the network or mobile device changes. When we discuss context we take this to mean the state of the device, the active applications and the available network connectivity. In order to benefit from this awareness we propose a policy oriented model that describes how the network interfaces should be used in any given context and how they should react upon change. These policies can be described in terms of; application priorities, bandwidth limits per interface or by reacting to changes in the devices state such as battery capacity. For the remainder of this section we describe the proposed policy definition and architecture for MPTCP-CA. An overview of the proposed design can be seen in Fig 1.

A. Policy Definition

Policies are defined and processed as a set of conditions and actions to execute when the conditions are met. Each condition is defined using a key, value and comparator. The key indicates the context that is to be monitored, while the value and comparator define when the associated actions are to be triggered. For example; when the battery voltage is less than 3.4V, disable the cellular interface. The context modules are dynamically loaded at run time, providing a flexible and extensible policy definition. Additionally, an individual policy may contain multiple conditions and actions, such that all conditions must be true before the actions are executed.

B. Architecture

The core split of the implementation is between the kernel space and the user space, as shown in Fig 1. The main path manager module resides in the kernel space and provides hooks via a netlink interface to allow communication with the user space component. When a new MPTCP connection is established, the path manager notifies the user space application, providing it with a unique token for the connection. This token identifies which TCP subflows belong to the same MPTCP connection. The user space application will then respond by making the appropriate netlink calls, which contain the necessary information to establish or close subflows for a specified interface, depending on the defined policies.

The core logic of the proposed path manager resides in user-space, this is to maximise the configurability and flexibility of the solution. It will also allow the path manager to be extended with an API, that will enable applications to provide the path manager with their specific requirements. These requirements include network metrics such as throughput or latency. The user space is made up of a number of modules;

1) Context Manager: The context manager, is responsible for parsing the user defined policies, and ultimately making the decisions about how network interfaces are used based on the information it receives from the monitoring and context modules.

2) Context Modules: The context modules integrate with the mobile device, obtaining the contextual information for the manager to use. This could represent network metrics such as the strength of a wireless signal or network layer metrics

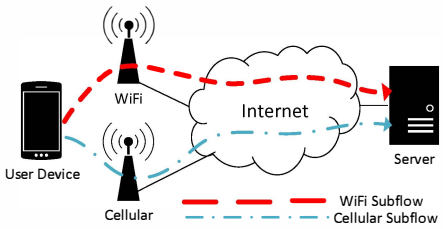


Fig. 2: Evaluation topology used when comparing MPTCP to MPTCP-CA.

such as available bandwidth, latency and loss. Alternatively this could also include device specific metrics such as the remaining capacity of the battery.

3) Link Monitor: The link monitor hooks into the routing subsystem using netlink in order to receive updates when ever network interface availability changes. This initially provides the context manager with the set of IP addresses that can be used to create additional subflows.

4) Application Monitor: The application monitor uses the netfilter connection tracking library to obtain a constant feed of active connections, this includes information for all transport protocols. This will provide a complete view of the current network application context, supplementing the MPTCP based information received from the kernels path manager.

IV. Evaluation

To evaluate the initial prototype of MPTCP-CA, we have implemented a subset of the proposed system described in Section III, including the kernel and user space path managers, which incorporates a simple policy engine to inform when to create or close MPTCP subflows. The evaluation system consist of a real world test-bed based on a Raspberry Pi acting as a mobile device, shown in Fig 2. The Raspberry Pi is powered by a 2000mAh battery and equipped with both WiFi and Cellular network interfaces as well as a smart battery monitor that is able to provide voltage and current readings. We compare the performance of MPTCP with the full mesh path manager to our implementation of MPTCP-CA, across three different use cases. The use cases consist of; application prioritisation, adjusting the network interface usage for energy consumption and pre-empting handover as signal strength decreases.

A. Application Policies

In the experiment shown in Fig 3, an initial application (App One) is in the background actively downloading content, this could represent receiving software updates or syncing files with cloud storage. After 20 seconds the user starts a second application (App Two), App Two then proceeds to download a small 10MB file that the user is interested in, so should be downloaded as quickly as possible. In Fig 3a, both App One and App Two establish MPTCP connections, creating subflows for both the WiFi and cellular network interfaces. When App Two becomes active, both applications share the available

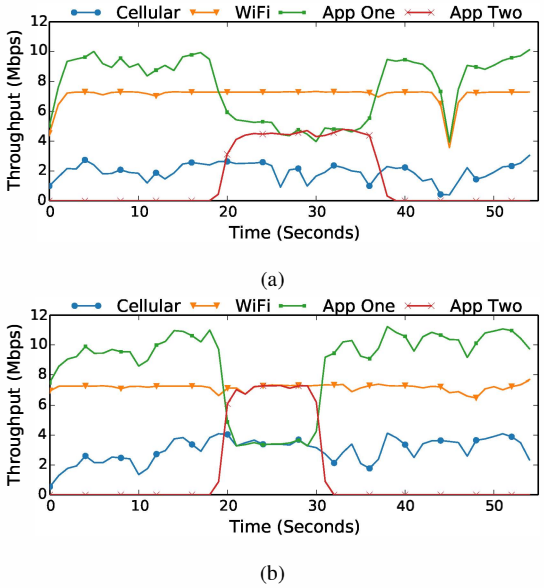
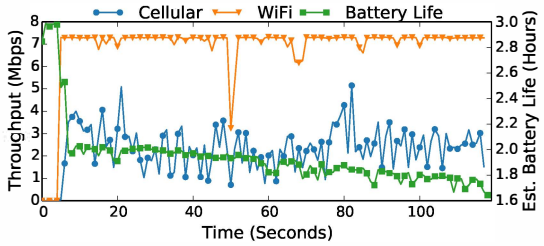


Fig. 3: TCP flows competing for network resource for (a) MPTCP and (b) MPTCP-CA. App One represents the throughput of background activity and App Two represents the throughput of the users active application. WiFi and Cellular refers to the throughput achieved over the individual links.

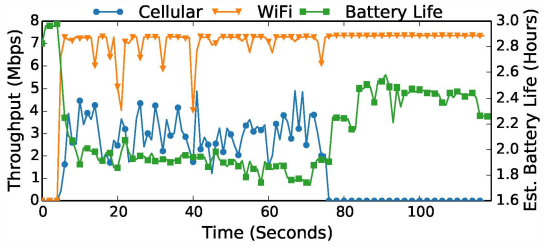
bandwidth equally, leading to App Two taking 18 seconds to download its content. In Fig 3b, a policy has been defined that states App Two should receive priority over App One in terms of bandwidth. When App One starts, the user space path manager is aware that both network interfaces are free to be used, and subsequently creates subflows for each path. When App Two starts, MPTCP-CA recognizes the conflict and gives App Two priority by removing App One's WiFi subflow. This forces App One to only use the cellular interface, giving App Two exclusive access to the higher bandwidth WiFi interface. This reduces the time App Two takes to download the users file to 11 seconds, which is 38% faster.

B. Energy Consumption

Mobile devices are limited in terms of battery capacity, therefore it is important to not only maximize battery life, but to also effectively balance this while trying to maximize the users quality of experience. In the experiment shown in Fig 4, the user is assumed to be watching a movie using Adaptive HTTP Streaming while mobile. As the amount of bandwidth changes the quality of video that the user receives adapts to minimize re-buffering. Both the WiFi and cellular interfaces have been enabled to maximize the quality of the video stream that is being viewed. This increases the total energy consumption required to view the video, and depending on the length, the battery capacity could be run down prematurely. When using the full mesh path manager as shown in Fig 4a, both interfaces are used indefinitely for the duration of the video. In this state, the total current draw for the Raspberry Pi, including WiFi and Cellular is approximately 0.96 Amps. At this rate the device would only remain turned on for a little

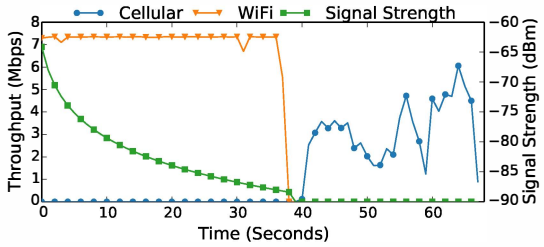


(a)

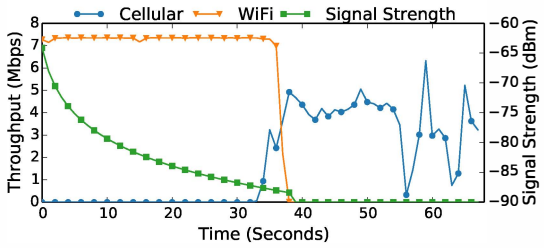


(b)

Fig. 4: Energy consumption while streaming video for (a) MPTCP and (b) MPTCP-CA.



(a)



(b)

Fig. 5: Handover of WiFi to Cellular with (a) MPTCP and (b) MPTCP-CA.

over 2 hours, given the 2000mAh battery used. In Fig 4b, MPTCP-CA identifies that the battery capacity is too low and the current draw is too high to maintain, this causes the Cellular subflow to be closed, allowing the network interface to fall into a sleep state, dropping the average current draw from 0.96 Amps to 0.73 Amps, extending the lifespan of the battery. Due to the adaptive application context, reducing the available bandwidth and subsequently lowering the video quality to increase battery life may be beneficial to the user.

C. Link Quality

In the experiment shown in Fig 5, the cellular interface is set to act as a backup to be used when the WiFi interface disconnects. To determine disconnection, we emulate signal strength using the model proposed in [9], with the user moving at 1.4 Meters per second away from the WiFi access point. When the signal strength reaches approximately -88dBm, the WiFi interface is disabled. In Fig 5a, the behavior of the stock MPTCP implementation is shown. When the WiFi interface disconnects it takes up to two seconds for the cellular subflow to start transmitting data, leading to the throughput dropping to zero for approximately two seconds, corroborating the results presented in [6]. In Fig 5b, MPTCP-CA is monitoring the signal strength, as it approaches the point of disconnection a new subflow is preemptively created over the cellular interface. As the cellular subflow is created before the WiFi disconnects the throughput does not drop to zero, improving the seamless nature of the handover.

V. Conclusion

In this paper we have proposed the need for more intelligent usage of the available network resource to assist in meeting application requirements. To meet this demand we have implemented MPTCP-CA which leverages context awareness to improve the flexibility and management of Multipath-TCP subflows. The implementation has been evaluated, demonstrating the benefits of improved subflow management in three different use cases. The evaluation has shown that using MPTCP-CA we are able to; increase throughput for prioritized application, use battery capacity to determine which network interfaces to use and improve handover by pre-emptively creating new subflows on secondary interfaces.

References

[1] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2013-2018," Feb. 2014. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.pdf

[2] D. Wischik *et al.*, "The resource pooling principle," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 5, pp. 47–52, Sep. 2008.

[3] C. Raiciu *et al.*, "TCP Extensions for Multipath Operation with Multiple Addresses," IETF - RFC 6824, January 2013, http://www.rfc-editor.org/rfc/rfc6824.txt.

[4] C. Paasch, R. Khalili, and O. Bonaventure, "On the benefits of applying experimental design to improve multipath tcp," in *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '13. New York, NY, USA: ACM, 2013, pp. 393–398. [Online]. Available: http://doi.acm.org/10.1145/2535372.2535403

[5] C. Raiciu *et al.*, "Opportunistic mobility with multipath tcp," in *Proceedings of the Sixth International Workshop on MobiArch*, ser. MobiArch '11. New York, NY, USA: ACM, 2011, pp. 7–12.

[6] C. Paasch *et al.*, "Exploring mobile/wifi handover with multipath tcp," in *Proceedings of the 2012 ACM SIGCOMM workshop on Cellular networks: operations, challenges, and future design*, ser. CellNet '12. New York, NY, USA: IEE, 2012, pp. 31–36.

[7] Y. sup Lim, Y.-C. Chen, E. Nahum, D. Towsley, and K.-W. Lee, "Cross-layer path management in multi-path transport protocol for mobile devices," in *INFOCOM, 2014 Proceedings IEEE*, April 2014, pp. 1815–1823.

[8] M. Coudron, S. Secci, G. Pujolle, P. Raad, and P. Gallard, "Cross-layer cooperation to boost multipath tcp performance in cloud networks," in *Cloud Networking (CloudNet), 2013 IEEE 2nd International Conference on*, Nov 2013, pp. 58–66.

[9] Q. Dong and W. Dargie, "Evaluation of the reliability of rssi for indoor localization," in *Wireless Communications in Unusual and Confined Areas (ICWCUCA), 2012 International Conference on*, Aug 2012, pp. 1–6.