Binder: A System to Aggregate Multiple Internet Gateways in Community Networks

Luca Boccassi*
Cisco Systems
lboccass@cisco.com

Marwan M. Fayed
University of Stirling
mmf@cs.stir.ac.uk

Mahesh K. Marina
University of Edinburgh
mmarina@inf.ed.ac.uk

ABSTRACT

We present a novel system termed `Binder` that seamlessly aggregates multiple geographically distributed Internet gateways in community networks. The proxy based approach taken in `Binder` allows for applications on end-user devices to enjoy the benefits from gateway aggregation without requiring any modifications. `Binder` makes novel use of multipath TCP (MPTCP), and additionally leverages OpenVPN tunneling and loose source routing (in a limited form that avoids security concerns) as part of an easily deployable implementation. `Binder` supports flexible gateway aggregation without negative effects from packet reordering through its use of MPTCP and generalizes link aggregation mechanisms to handle distributed links. Our proof of concept evaluation of `Binder` using a real implementation over an emulation based lab testbed demonstrate its benefits in terms of bandwidth aggregation, load balancing and fault tolerance relative to common practice.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]:

Keywords

Gateway aggregation; MPTCP; community networks.

1. INTRODUCTION

Community networks have emerged in recent years as a viable grassroots based alternative to the traditional Internet Service Provider (ISP) led approach to deliver Internet connectivity to those yet to be connected by any means. They have sprung up in almost every part of the world as evident from [1] and are typically based on wireless technologies (often some form of WiFi). Motivations for community networks differ depending on the operational setting. In rural areas and the developing world, it is the unavailability of alternate means for Internet access that has largely been the driving force behind community network deployments (e.g., [2, 3, 4, 5]), whereas in urban areas it is people in low-income neighborhoods finding it unaffordable to subscribe to ISP broadband offerings (e.g., [6, 7]). Regardless of the operational setting, community networks are commonly characterized by organic growth, and cooperative deployment and management.

Another common feature is of key relevance to this paper: In all but very small community network deployments is the presence of multiple, often geographically distributed, Internet gateways (upstream connection points to the public Internet). A typical example is the Tegola network [4, 8], which started out as an experimental rural wireless network and has since evolved into a larger community wireless network serving more than 300 households in a remote part of Scotland. The Tegola network currently has three different Internet gateway locations with seven 4Mbps DSL lines and a 34Mbps E3 line among them. Multiple gateway configurations in Tegola, and in almost every other community network, are limited to providing failover in the event a preferred gateway (or path to it) fails. Given that the cost associated with Internet gateways amounts to a significant fraction of the deployment and operational expenditure of community networks, using them for failover alone is indeed suboptimal. This, however, is the case in practice mainly due to the lack of an easily deployable solution to seamlessly aggregate distributed Internet gateways. Filling that void is precisely the aim of this paper.

We present `Binder` (§3), a novel system for aggregation of multiple geographically distributed Internet gateways in a community network. Besides supporting fault tolerance for resilience against failure of gateways (and paths leading to them), `Binder` enables bandwidth aggregation and load balancing at the per user and per flow level. Specifically, with `Binder`, a user can potentially consume aggregated bandwidth across multiple gateways for one or more flows. User traffic can be split across the available Internet connections thus gaining the benefit of load balancing. `Binder` takes a proxy based approach to realize the above benefits. A *relay* inside the community network, co-located with the access point serving user devices, and handles the distribution of traffic across multiple gateways in conjunction with a *proxy* situated outside the community network. Each relay establishes with the proxy a virtual channel built upon a virtual private network (VPN) tunnel using multipath TCP (MPTCP) [9]. We have implemented loose source and record route (LSRR) compatibility into MPTCP so that

*This work was done at the University of Edinburgh, prior to joining Cisco Systems Inc.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
LCDNet'13, September 30 2013, Miami, Florida, USA
Copyright 2013 ACM 978-1-4503-2365-9/13/09 ...$15.00.
http://dx.doi.org/10.1145/2502880.2502894.

`Binder` may direct each sub-flow over the backbone of the community network to a unique gateway.

Practicality and ease of deployment considerations heavily influenced the design and implementation of `Binder`. `Binder` does not require any modifications to end-user devices or applications. Moreover, its relay and proxy components can run on commodity hardware. Our implementation of `Binder` is based on open-source software – OpenVPN [10] and our modified version of a publicly available MPTCP implementation [11]. Source code is available from [12]. Evaluations of `Binder` (§4) over a Dummynet [13] based emulation testbed show that it effectively meets its design objective of gateway aggregation to realize the benefits of bandwidth aggregation, load balancing and fault tolerance.

`Binder` has several novel aspects when compared to the state of the art (§2). Firstly, it presents a new use case for Multipath TCP (MPTCP) [9]. Until now, applications of MPTCP have been limited to use in data centers and in mobile devices [14, 15]. Secondly, `Binder` generalizes link aggregation mechanisms, exemplified by LACP in 802.1ax [16], to aggregate geographically distributed links. Thirdly, compared to network layer approaches that target routing modifications (e.g., [17, 18, 19]), `Binder` supports aggregation of gateways in a flexible manner, and avoids packet reordering related performance degradation commonly associated with traditional bonding techniques. Finally, `Binder` differs from other proxy based approaches for bandwidth aggregation in the literature (e.g., [20, 21]) in terms of the target application scenario and its design constraint to avoid end-user device/application modifications.

2. RELATED WORK

The aggregation of bandwidth across multiple channels has origins at the link layer with protocols such as MPPP [22], and culminates in the Link Aggregation Control Protocol (LACP) and IEEE 802.1ax standard [16]. Link-layer aggregation is achieved by 'bonding' together multiple ports on a point-to-point link. FatVAP [23] provides link-layer bonding in 802.11 in a novel client-side driver architecture that connects and schedules a single wireless interface with multiple access points. Since bonding assumes co-located links, it is inadequate for our purpose of aggregating across geographically distributed Internet gateways. Moreover, the throughput increases derived from bonding only materialize with multiple flows. A single flow with link bonding is bounded by the capacity of the port to which a flow is assigned, a typical practice to avoid penalties induced on TCP performance by reordered packets. The FLARE algorithm [24] addresses reordering by scheduling bursts of packets in a flow across different paths when the time between bursts is less than the round-trip time. While the intention behind this scheduling is to balance load across paths, it also leads to bandwidth aggregation as a positive byproduct. However, the route to implementation of the approach proposed in [24] is unclear as it was evaluated via analysis and trace-based simulation.

Several network layer approaches are designed to exploit the presence of multiple gateways in wireless mesh networks. Gateway Aware Routing [17] associates a gateway for each router in the network independently based on a composite metric. Despite the use of multiple gateways there is no aggregation. The Multi-Gateway Association (MGA) model [18] proposes to abstract multiple gateways in a network via a 'super-gateway'. Super-gateways split each flow over a subset of gateways selected using a greedy algorithm. While novel, the super-gateway imposes location constraints on gateways, and more importantly assumes a re-assembly process at the receiver/host side. By contrast Plasma [19] suggests an 'anypath' routing model where the path and gateway selection is determined by routers per-hop and per-packet. The independent selection of gateways for each packet in Plasma gives the illusion of aggregation. Evaluations of Plasma, however, were limited to UDP traffic, leaving open questions about the possibility of packet reordering and its effect on TCP performance. Our proposal, Binder, is routing agnostic. It leverages the standard loose source routing options in a limited sense — inside the community network for the specific purpose of forwarding different MPTCP sub-flows to different gateways. Binder may, however, deliver improved performance in general multihop community network settings if a traffic-adaptive routing protocol (e.g., [25, 26]) is used alongside.

The Binder architecture is more closely matched by designs in [20] and [21]. Their common attribute is the use of intermediary nodes. In [20], a bandwidth aggregation mechanism is implemented by use of a proxy to split, assemble, and schedule traffic over multiple gateways simultaneously. Given its focus on one-way streaming UDP traffic, TCP behavior remains an open question. The Distributed Link Bonding mechanism proposed in [21] aggregates available bandwidth with nearby mobile devices by injecting relays. Relays in this system capture and forward client traffic over tunnels to a proxy. Evaluations demonstrated merit of the architecture although TCP throughput is seen to suffer from reordering issues. The design goals in the above mentioned two systems [20, 21] are different from ours, and crucially require modifications at each client device. This is a real-world obstacle to deployment that Binder avoids.

3. DESIGN AND IMPLEMENTATION

`Binder` is both an architecture and an implementation designed for community access networks. This section describes its design and implementation.

3.1 Overview

The primary goal of `Binder` is to aggregate multiple Internet gateways, potentially geographically distributed at different locations. In community networks, where Internet connections are typically the most expensive elements, the benefits of doing so include the following:

- *Bandwidth aggregation*: Each user or flow may see bandwidth that is the aggregate of gateways, rather than be bound by the capacity available at an assigned gateway.
- *Load balancing*: The load of the network is balanced proportionally across gateways, and according to available bandwidth at each gateway.
- *Fault tolerance*: Each connection is more resilient against failures in subsets of gateways and the paths leading to them.

The remaining practical considerations address deployment constraints. Specifically, `Binder` is designed to be easy to deploy using commodity hardware, require no modifications to end-hosts, and be agnostic to routing protocols, thereby surviving them if changes occur.

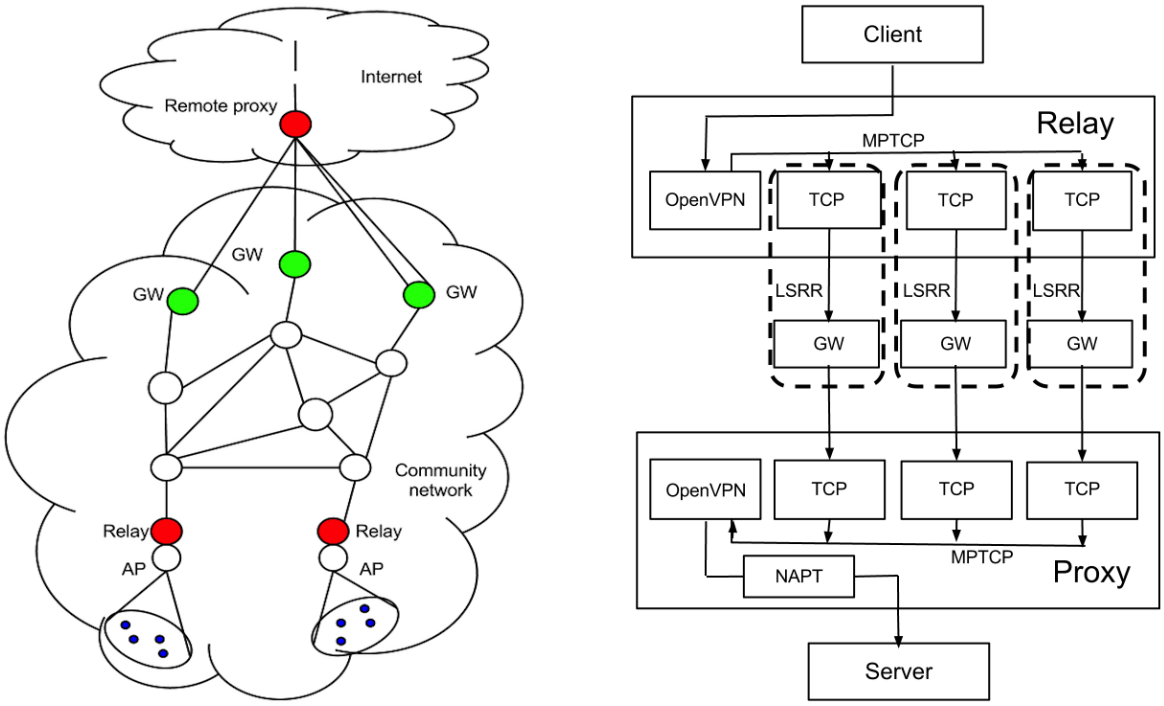

Figure 1: (a) `Binder` components in a community network context; (b) `Binder` software architecture.

From a high-level perspective `Binder` is made up of three components: relays, proxy and a multipath component. Each is illustrated in Figure 1a, where components inside the lower 'cloud' represent the community network that is operated independently from the rest of the Internet. User (at the bottom of the figure) packets in the network will first encounter `Binder` at a *relay* that is co-located with the access point that serves them. The relay captures and redirects traffic to a remote *proxy* that sits outside of the network. Finally, relays and the proxy communicate via a *multipath* component. Packets returned from the Internet are treated similarly, and follow a reverse path from the proxy to the relays. Collectively these components provide the tunneling and aggregation that define `Binder`.

The architecture itself is agnostic to the implementation choices. In our implementation of `Binder` these functions are composed by building on well-known standards and implementations. Specifically, it relies on OpenVPN [10] and a modified implementation of Multipath TCP [27]. The interaction of components in our implementation is represented by the diagram in Figure 1b. Their function is described in the remainder of this section.

3.2 Capture and Redirection of Traffic

In the `Binder` system, packets from an end-user device are captured at an associated access point (AP) and redirected to a proxy outside the network. Referring to Figure 1a, this is accomplished by a set of relays co-located with APs that intercept, split, and forward traffic to a proxy outside the network for re-assembly. This relationship is used to establish a single tunnel from each relay to the proxy, irrespective of the number of gateways in aggregate use.

This approach fits well with the stated goals. In particular it avoids modifications to end-hosts, it enables both incremental and partial deployment, and makes no changes to core network behaviour and functionality.

In addition the single-tunnel relationship is easier to manage at the relay, and adds flexibility and scalability to the proxy. For example, multiple servers may form a 'proxy farm' to handle high aggregate bandwidth or offer fault-tolerance, and even establish a cloud service to tunnel with Binder instances in other community networks.

The Tunneling Implementation

`Binder` captures and redirects packets by use of OpenVPN tunnels. OpenVPN was determined to be the best choice following extensive and systematic evaluation against alternatives in a lab testbed environment. It is worthwhile summarizing the experimental results that determined this choice. In selecting the best tunneling solution we compared the following options at the relay nodes and proxy:

- a custom-written 'filter-and-forward' daemon using `libpcap` [1] to filter packets;
- a custom-written 'filter-and-forward' daemon using the newer `libtrace` [2] to filter packets;
- a custom-written 'capture-and-forward' daemon using the Linux `TUN` driver and IPTables.
- OpenVPN TCP tunnels to capture and forward, configured without encryption or compression.

Experiments were performed on two testbeds. In all experiments the throughput of OpenVPN redirection met baseline forwarding. In retrospect this should be unsurprising:

[1] http://www.tcpdump.org/
[2] http://www.wand.net.nz/trac/libtrace/

OpenVPN is a well maintained, highly optimized, and configurable package. Further description of each option with comparative experimental evaluation is available in [28].

3.3 Multipathing and Aggregation

Our implementation of `Binder` builds upon MPTCP [27] to provide multipathing. Recently approved by the IETF [9], MPTCP proved to be the appropriate starting point, requiring no network-layer support. MPTCP manages multiple subflows by maintaining a master window at each of sender and receiver, and additional windows for each subflow. Its congestion control algorithm links master windows with subflow windows to provide congestion control across multiple paths, respect TCP-fairness across all bottleneck links, and gracefully handle re-ordering. Current applications include data centers, and mobile devices that have multiple interfaces. `Binder` presents a new use case for MPTCP.

What MPTCP lacks is the ability to direct its own subflows. By relying instead on the routing layer to select paths to the destination, there is no guarantee that sub-flows will traverse all available gateways. MPLS offers a possible solution, and is standard practice in larger network operations, yet is overly complex for our domain, and requires expensive high-powered hardware. Binder instead fills this gap by implementing a novel composition of MPTCP, and loose source and record routing (LSRR).

MPTCP with Loose Source and Record Routing

We use LSRR options to influence both outbound and inbound paths within the network. To the application, MPTCP sockets appear as regular TCP sockets. In the case of standard TCP there is support for IP Loose Source and Record Routing (LSRR), which allows TCP flows to influence the chosen path by visiting a set of given IP addresses in the order given. `Binder` adds this facility to MPTCP to ensure that all available gateways are used by MPTCP sub-flows.

Figure 1b illustrates the `Binder` software architecture. At the relay, using a callback mechanism a list of available gateways is made available to MPTCP at runtime. One TCP socket to the proxy (a 'subflow') is then opened for each available gateway. Loose source routing support injects the gateway into the IP header. At the proxy, packets are reassembled before network address and port translation (NAPT) and forwarding. Return packets from servers to clients are handled in the reverse fashion. We note that LSRR is contentious in the public Internet, and that packets with LSRR options may be ignored or dropped by other networks. To ensure delivery to the proxy we make LSRR options in IP headers invisible from outside networks via an `iptables`[3] extension at each gateway. This extension allows gateways to strip and store LSRR options from outgoing packets, then re-insert them on return packets.

Our implementation adds flexibility to the system. Compatibility with loose source routing ensures `Binder` can use all available gateways, while leaving room to benefit from traffic-adaptive routing protocols if appropriate.

4. EVALUATION

In this section, we evaluate the effectiveness of `Binder` with respect to bandwidth aggregation, load balancing and fault tolerance aspects.

[3] http://www.netfilter.org/projects/iptables/

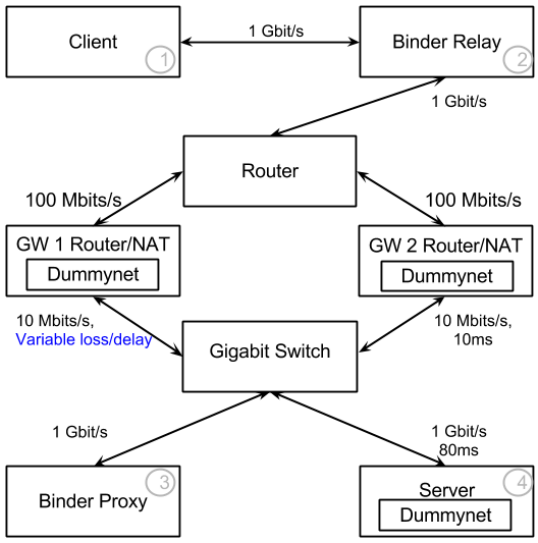

Figure 2: Illustration of testbed setup.

4.1 Experimental Setup

Our evaluations are conducted over an emulation based controlled laboratory testbed using the Dummynet emulator [13] to realize diverse path characteristics in terms of packet loss rates and round-trip latencies drawn from observations in the field.

Figure 2 depicts the testbed setup, and represents how a small `Binder` deployment would look in practice. In the outbound direction clients devices (marked as '1') in the home connect to a nearby Access Point with co-located `Binder` relay (marked as '2'). Packets are then routed to the `Binder` proxy (marked as '3') via the rural backbone network and through available, potentially disparate, gateways labeled GW1 and GW2 in Figure 2. The proxy reassembles, NATs, and forwards traffic to the Internet. Return traffic follows the reverse path. We note that while a `Binder` proxy may have multiple interfaces, our deployments are more likely to sit behind a switch in a single network. A separate router is injected between the `Binder` relay and gateways to validate our loose source routing implementation.

Each machine in the testbed is a dual core 2Ghz processor with 2GB of memory, running a linux 2.6.32.5 kernel. Bottleneck link capacities are 10Mbps. To model paths of differing quality in terms of loss and latency, a dummynet emulators at GW1, GW2, and the server. We use GW1 to subject the corresponding path with varying packet loss rate and latency while leaving the other path via GW2 undisturbed. Round trip times are set to reflect the RTT characteristics measured from within Tegola [4, 8] when compared to remote hosts on the Internet, as shown in Table 1. Default values of 10ms are set for each sub-path in the `Binder` tunnel, with an additional 80ms RTT between the proxy and the server, for a total RTT of 90ms from the client.

We focus on TCP to evaluate end-to-end characteristics from effects of MPTCP tunnels in `Binder`. The average goodput for 30 second long `iperf` TCP flows between the client and server (beyond the proxy) is the key performance metric when comparing `Binder` against the baseline single gateway case.

Src - Dst	min	avg	max	mdev
(i) default gateway	2.32	3.69	8.43	1.08
(ii) alternate gateway	6.93	15.88	71.17	11.22
(iii) remote host	23.28	25.35	37.96	2.88

Table 1: RTT values in *ms* as reported by *ping* from residential machine inside Tegola [4, 8] to (i) default and nearby gateway, (ii) alternate and furthest gateway, (iii) remote host via default gateway.

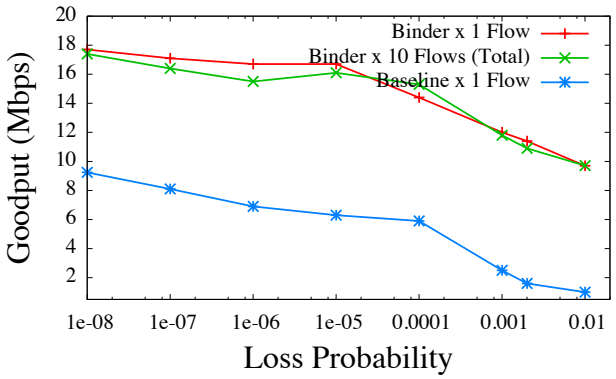

Figure 3: `Binder` goodput when one subpath has no loss, while loss rate on other subpath varies.

4.2 Varying Loss Rate

Figure 3 shows goodput for `Binder` against baseline TCP as loss probabilities on one of the paths (via GW1 in Figure 2) is varied.We see in Figure 3 that baseline single-path flows behave as expected in response to packet loss variation, while the `Binder` flow maintains aggregate capacity throughout. We have omitted loss rates greater than 1%, at which point TCP baseline flows are unusable.

By contrast, `Binder` maintains service at all levels of loss. With low loss rates, Binder exhibits gains from perfect bandwidth aggregation. As one of the paths get increasingly lossy goodput with `Binder` correspondingly gets closer to the capacity of the other non-lossy path. Figure 3 also shows the total goodput measured across sets of 10 simultaneous flows. In all cases the total goodput matches the aggregate goodput of both subpaths. The coefficients of variation of the mean goodput of individual flows range from 5% in most cases to a maximum of 20%. This demonstrates that, unlike traditional bonding techniques which bind each flow to a distinct pipe in a bonded channel, a single `Binder` tunnel benefits multiple flows equally.

4.3 Varying Latency

In the presence of lossy links `Binder` is able to maintain a high level of service. A similar trend emerges as the latency between subpaths increases. This trend is summarized by Figure 4, in which TCP goodput over Binder is plotted for both single, as well as 10 simultaneous flows, as the difference in round trip time (RTT) between subpaths increases. Initially both paths share the baseline RTT of 80ms. RTTs along the higher latency path through GW1, and represented along the x-axis in Figure 4, range from 80-120ms. Selected values maintain the ratios between RTTs to available gateways shown in Table 1.

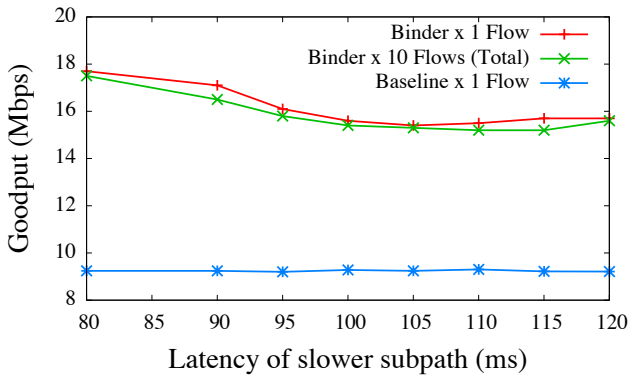

Figure 4: `Binder` goodput when one subpath is 80ms, while other subpath latency varies.

Time (s)	High RTT path (ms)	Loss Prob.
0	85	1e-07
3	90	1e-06
6	95	1e-05
9	100	1e-04
12	105	1e-03
15	85	1e-07
18	90	1e-06
21	95	1e-05
24	100	1e-04
27	105	1e-03
33	110	1e-02
36	110	1

Table 2: Time varying changes in path characteristics to evaluate `Binder` adaptability and load balancing.

Along a single path TCP, with its dynamic window scaling implementation, operates at full capacity. `Binder` maintains high aggregate throughput. Goodput diminishes slightly as the difference in RTTs increases. This stems from MPTCP which has to manage three congestion windows, and retransmissions between them, effectively. Total goodput measured across sets of 10 simultaneous flows is also shown in Figure 4. In all cases mean the total goodput exceeds 16Mbps. Coefficients of variation from mean goodput of individual flows range from 17-25%, further demonstrating the benefit of a single `Binder` tunnel to multiple flows.

4.4 Time Varying Loss and Latency

We now consider a dynamic scenario intended to simulate a slow decline of a subpath. In this scenario loss rate and latency on the subpath via GW1 are made to change according to the times and values shown in Table 2. This sequence of events cycles twice through a slow decline, followed by a failure. Figure 5 shows the result comparing `Binder` with the baseline case for TCP traffic between the client and server. Note that the x-axis in Figure 5 corresponds to the time component in Table 2.

As expected, soon after 30 seconds when loss probability reaches 0.01, single-path TCP is no longer able to cope. By contrast the `Binder` handles the rate of change in path characteristics, and maintains appropriate balance across paths of differing quality, to consistently provide higher goodput. Interestingly, `Binder` seems to mirror TCP single-path reactions, while being more sensitive to the causes. We note, for example, the drops in goodput at 15 and 26 seconds

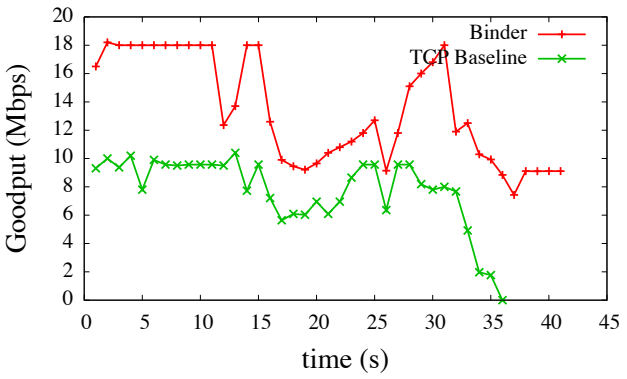

Figure 5: `Binder` goodput (1s averages) relative to baseline TCP in a dynamic network scenario described by Table 2

in Figure 5. Rather than a decrease in performance by a *magnitude* equal to the drop in TCP, `Binder` and MPTCP appear to be penalized by a *proportion* equal to that in the drop of TCP. This is a subject currently under investigation.

5. CONCLUSION

In this paper we introduced `Binder`, a system to aggregate multiple Internet gateways in community networks. `Binder` is novel in that it is designed to aggregate gateways that are geographically distributed, and that have varying characteristics. As a result `Binder` provides two immediate benefits. First, it balances load proportionally across gateways, and enables aggregation of available gateway capacity at the per flow level. Second, it provides fault tolerance against failure in any subset of gateways and their associated paths while avoiding the need for a failover mechanism, instead reducing service to the levels surviving gateways.

`Binder` aggregates gateways by using Multipath TCP to communicate over VPN tunnels that join relays inside the community network with an outside proxy. Loose source and record routing functionality maps each MPTCP subflow within the community network to a unique gateway. `Binder` runs on commodity hardware, requires no network-layer or end-host modifications, and allows incremental or partial deployment. Proof of concept evaluations of `Binder` using a real implementation over an emulation based lab testbed demonstrate that it provides gateway bandwidth aggregation, load balancing and fault tolerance benefits in diverse network conditions. Our on-going work focusses on a more comprehensive evaluation of `Binder` in a real-world community network setting, and characterizing the performance benefits to real application workloads.

6. ACKNOWLEDGMENTS

We would like to thank Olivier Bonaventure and the MPTCP Linux kernel implementation team, in particular Christoph Paasch for his insight and observations.

7. REFERENCES

[1] Wikipedia. List of wireless community networks by region. http://tinyurl.com/5efgye.
[2] Airjaldi. http://drupal.airjaldi.com.
[3] S. Surana et al. Beyond pilots: Keeping rural wireless networks alive. In *Proc. USENIX NSDI*, 2008.
[4] G. Bernardi, P. Buneman, and M. K. Marina. Tegola tiered mesh network testbed in rural scotland. In *Proc. ACM MobiCom 2008 Workshop on Wireless Networks and Systems for Developing Regions (WiNS-DR'08)*, 2008.
[5] R. Flickenger et al. Wireless Networking in the Developing World. http://wndw.net/download.html.
[6] Technology For All. http://tfa.rice.edu/.
[7] Bristol Wireless. http://www.bristolwireless.net/.
[8] Tegola network. http://www.tegola.org.uk/.
[9] A. Ford, C. Raiciu, M. Handley, S. Barre, and J. Iyengar. Architectural guidelines for multipath tcp development. *Internet Engineering Task Force (IETF) , Request for Comments*, (6182):28, March 2011.
[10] OpenVPN. https://community.openvpn.net.
[11] Multipath TCP Linux kernel implementation. http://mptcp.info.ucl.ac.be.
[12] L. Boccassi. Binder: MPTCP-LSRR git repository. https://github.com/bluca/mptcp.
[13] M. Carbone and L. Rizzo. Dummynet revisited. *ACM SIGCOMM Comput. Commun. Rev.*, 40(2), 2010.
[14] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley. Improving datacenter performance and robustness with multipath TCP. In *Proc. ACM SIGCOMM*, 2011.
[15] C. Paasch, G. Detal, F. Duchene, C. Raiciu, and O. Bonaventure. Exploring mobile/WiFi handover with multipath TCP. In *Proc. ACM SIGCOMM Workshop on Cellular Networks (CellNet'12)*, 2012.
[16] 802.1AX-2008 - IEEE Standard for Local and metropolitan area networks – Link Aggregation. Technical report, IEEE, 2008.
[17] P. Acharya, D. L. Johnson, and E. M. Belding. Gateway-aware routing for wireless mesh networks. In *Proc. IEEE Int'l Conf. on Mobile Ad hoc and Sensor Systems (MASS)*, 2010.
[18] S. Lakshmanan, R. Sivakumar, and K. Sundaresan. Multi-gateway association in wireless mesh networks. *Ad Hoc Networks*, 7(3):622–637, 2009.
[19] R. P. Laufer, P. B. Velloso, L. F. M. Vieira, and L. Kleinrock. Plasma: A new routing paradigm for wireless multihop networks. In *Proc. IEEE INFOCOM*, 2012.
[20] K. Chebrolu and R.R. Rao. Bandwidth aggregation for real-time applications in heterogeneous wireless networks. *IEEE Transactions on Mobile Computing*, 5(4), 2006.
[21] Yin Wang. *Distributed diversity in hybrid wireless networks*. PhD thesis, Northeastern University, Boston, MA, USA, 2010.
[22] The PPP Multilink Protocol, RFC 1990. Technical report, IETF, 1996.
[23] Srikanth Kandula, Kate Ching-Ju Lin, Tural Badirkhanli, and Dina Katabi. Fatvap: aggregating ap backhaul capacity to maximize throughput. In *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2008.
[24] Srikanth Kandula, Dina Katabi, Shantanu Sinha, and Arthur Berger. Dynamic load balancing without packet reordering. *ACM SIGCOMM Comput. Commun. Rev.*, 37(2):51–62, March 2007.
[25] V. Raghunathan and P. R. Kumar. Wardrop routing in wireless networks. *IEEE Transactions on Mobile Computing*, 8(5):636–652, 2009.
[26] L. Qiu, R. Y. Yang, Y. Zhang, and H. Xie. On self adaptive routing in dynamic environments - an evaluation and design using a simple, probabilistic scheme. In *Proc. IEEE ICNP*, 2004.
[27] C. Raiciu et al. How hard can it be? designing and implementing a deployable multipath TCP. In *Proc. USENIX NSDI*, 2012.
[28] L. Boccassi. Multi-gateway support for long-distance WiFi mesh networks. Master's thesis, The University of Edinburgh, 2012.