Routing, Network Coding and TCP Support for Wideband HF Communications

Tugba Erpek*, Yalin E. Sagduyu*, Yi Shi*, Maice Costa*, Kemal Davaslioglu*, Hao Han*,
Daniel Fayette*, Eric Johnson†, Elizabeth Bentley‡, and Clif Banner§
* Intelligent Automation, Inc., Rockville, MD 20878, USA
† New Mexico State University, Las Cruces, NM 88003, USA
‡ U.S. Air Force Research Laboratory, RITF, Rome, NY 13441, USA
§ AFLCMC/HNAA, Hanscom AFB, MA 01731, USA
Email: {terpek, ysagduyu, yshi, mcosta, kdavaslioglu, hhan, dfayette}@i-a-i.com, ejohnson@ad.nmsu.edu,
{elizabeth.bentley.3, clifden.banner.ctr}@us.af.mil

***Abstract*—High frequency (HF) communications provide beyond line-of-sight (BLoS) links for robust and reliable delivery of critical messages over long distances. HF communications suffer from low data rates and are susceptible to highly dynamic HF channel effects. Wideband HF (WBHF) communication technologies aim to support higher throughput rates that can be achieved with higher bandwidth. While there has been progress in the design of WBHF technologies at the physical layer, there is a lack of network and transport layer solutions that can support WBHF communications with improved end-to-end network performance in terms of throughput, reliability, delay, and overhead. First, a high-fidelity WBHF network simulator is presented to implement, test, and evaluate network protocols under realistic HF channel conditions that show unique characteristics compared to other spectrum bands. The WBHF network simulator runs real packet traffic with the real network protocol stack on virtual HF radios. The integrated HF channel simulator models long, intermediate, and short time-scale HF channel effects. Second, routing is combined with packet-level coding to support reliable communications for unicast and multicast traffic while limiting the need for end-to-end feedback messages. To support interoperability with transmission control protocol (TCP) applications, a performance-enhancing proxy (PEP) is used to improve the throughput achieved with TCP for WBHF communications. These network solutions are evaluated with the high-fidelity WBHF network simulator and their feasibility to improve the WBHF network performance is demonstrated.**

***Index Terms*—HF communications; WBHF; network simulator; routing; packet coding; TPC; PEP; multicast.**

I. Introduction

High Frequency (HF) communications (in the 3-30 MHz frequency band) provide beyond line-of-sight (BLoS) communication capabilities for robust and reliable delivery of messages such as Emergency Action Messages (EAM). As the satellite communications (SATCOM) spectrum saturates, HF communications become viable alternatives to costly per-byte subscriptions to SATCOM, especially in satellite-limited or denied environments. Building upon the improved understanding of HF ionospheric conditions, HF communications find applications ranging from amateur radios to Global Maritime Distress and Safety System (GMDSS).

Traditionally, HF communications have been mostly used for voice, due to the low bandwidth, i.e., 3 kHz, and the low throughput, i.e., 9,600 bps, over HF links. However, in recent years, with the development of the high throughput HF technology (HTHF), also known as wideband HF (WBHF), higher throughput can be achieved with higher bandwidth (such as 48 kHz). Moreover, these improved waveforms can support reliable digital data transmission in addition to voice. Most of the research on HF communications, up to now, has focused on waveform development at the physical layer to increase the throughput on HF links. This paper tackles two challenges to overcome, in order to realize the benefits of WBHF communications, namely, the need for a high-fidelity simulation environment and for network protocols optimized for WBHF communications.

WBHF network simulator integrated with a HF channel simulator: HF propagation channel behavior varies significantly over location, time of day and adverse environments, and it is difficult to guarantee a certain signal-to-noise ratio (SNR) at the receiver. Therefore, WBHF network protocols need to adapt to SNR dynamics. Systematic evaluation of WBHF network performance needs a high-fidelity network simulator that accounts for HF channel characteristics, real packet traffic and full network protocol stack for HF radios. This simulator is critical in assessing novel network designs for WBHF communications before integrating with HF radios.

Network protocols for WBHF communications: Despite the recent waveform developments, potential improvements mostly at the physical layer are not translated to network protocol performance since higher network layer protocols have not been designed yet to protect HF communications against its susceptibility to various conditions such as seasons, day/night, and space weather (e.g., solar activity). These factors cause communications to become degraded or denied, and raise the need for full protocol stack networking solutions to pass information reliably from sources to destination. Since bandwidth is precious for WBHF communication networks, novel mechanisms across the network protocol stack are needed to increase the reliability and reduce the *overhead* messages by limiting the feedback messages and retransmissions. In

978-1-5386-7185-6/18/$31.00 ©2018 IEEE

addition, WBHF communications need to support *multi-hop networking* where relay nodes (WBHF radio transceivers) are used to forward packets from sources to destinations when the direct BLoS link is not available due to channel conditions.

This paper describes novel solutions to address these two main challenges. First, a *high-fidelity WBHF network simulator* is presented to implement, test, and evaluate network protocols under realistic HF channel conditions. This simulator integrates an *HF channel simulator* that accounts for long, intermediate and short time-scale channel effects in HF spectrum, and uses *real packet traffic* with the *real network protocol stack*, each running on a virtual WBHF radio.

Second, algorithms at the network and transport layers are presented to improve the throughput, delay, and reliability of WBHF network communications. These solutions are evaluated with the developed WBHF network simulator. Two routing protocols are compared in terms of the overhead, delivery ratio, delay, and throughput. Then, two packet coding schemes are evaluated on top of routing to reduce the overhead (by limiting the end-to-end feedback messages) and increase the throughput and reliability of unicast or multicast packet deliveries. Some legacy applications run on transmission control protocol (TCP) that relies on end-to-end feedback messages to ensure reliable communications and thus incurs high level of overhead that is not suitable for HF communications with low link rates. A *performance-enhancing proxy* (PEP) is added on top of WBHF routing to improve the performance on network paths where native performance suffers due to characteristics of a link or subnetwork on the path.

The rest of the paper is organized as follows. Section II presents the high fidelity WBHF network simulator. Section III and Section IV evaluate routing protocols and packet coding schemes for WBHF network communications. Section V evaluates PEP in supporting TCP for WBHF network communications. Section VI concludes the paper.

II. WBHF Network Simulator

The network simulator architecture is shown in Figure 1. This architecture uses real packet traffic and a real IP stack. Linux containers are used to mimic WBHF radios with having their own protocol stack for each radio. RFview scenario planner (the integrated GUI of network channel emulator, RFnest [1]) is used to specify the number of radios, positions, radio parameters, and mobility patterns [2], [3]. HF channel simulator is based on the Walnut Street model [4] and provides the path loss information based on the radio and scenario parameters received from RFview. The packet-level network simulator provides SNR vs. packet error rate curves based on the real measurements from HF communications and determines whether a transmitted packet is received without error or not. The medium access control (MAC) is integrated through Extendable Mobile Ad-hoc Network Emulator (EMANE) [6]. With EMANE, network protocol and application software can be experimentally subjected to the same conditions that are expected in real-world wireless network systems. This WBHF network simulator covers a realistic WBHF communication

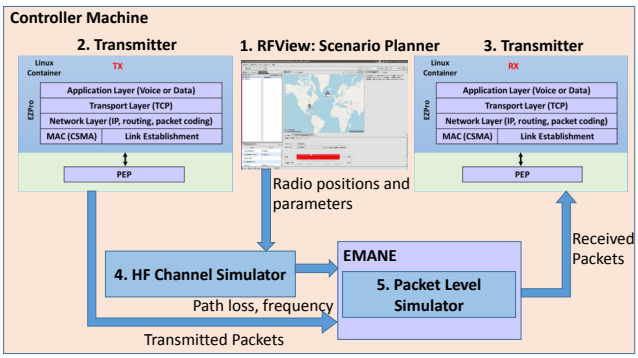

Fig. 1. Network simulator system architecture.

network setting with unicast, multicast and broadcast messages, point-to-point and multi-hop communications, voice and data communications, and fixed or mobile radio positions.

A. High-fidelity HF Channel Simulator

The statistical properties of the ionospheric propagation, HF surface-wave and skywave channels (different from L and S band characteristics typically used in wireless simulators and emulators [5]) are captured in the HF channel simulator. The refractive and absorptive characteristics of the channel depend on the radio frequency, location, time of the day, season, solar weather, and atmospheric effects. These factors cause the ionospheric channel to vary over a wide range of time scales, including *milliseconds* (multipath spread that produces intersymbol interference), *seconds to minutes* (fading), *hours* (diurnal variations), and *years* (11-year sunspot cycle). The Walnut Street HF model [4] is used in the HF channel simulator and ionospheric path loss is computed as the superposition of long, intermediate and short time scale effects:

- *Long time scale*: Path geometry relative to the sun and other slowly varying factors. Ionospheric propagation prediction programs such as Ionospheric Communications Analysis and Prediction Program (IONCAP), Ionospheric Communications Enhanced Profile Analysis and Circuit Prediction Program (ICEPAC), and Voice of America Coverage Analysis Program (VOACAP) are used to capture these effects.
- *Intermediate time scale*: Fading effects due to ionospheric motion and similar phenomena.
- *Short time scale*: Multipath interference. The Watterson model [7] is used to capture short time scale effects.

The HF channel simulator models the HF propagation that reflects channel variations in time. The simulator leverages `pythonProp` [8], which is a wrapper to the VOACAP HF Circuit propagation prediction engine [9] to model the long time scale effects. The `pythonProp` library consists of VOACAPL (a command line driven engine that performs the predictions) and `pythonProp` (a set of tools supporting the graphical input and output). The `pythonProp` library (that uses `voacapgui` as a front end to the VOACAP engine) is applied to characterize HF propagation between two locations (*point-to-point mode*), and generate HF propagation plot over a user-defined area from a fixed transmit mode (*area mode*).

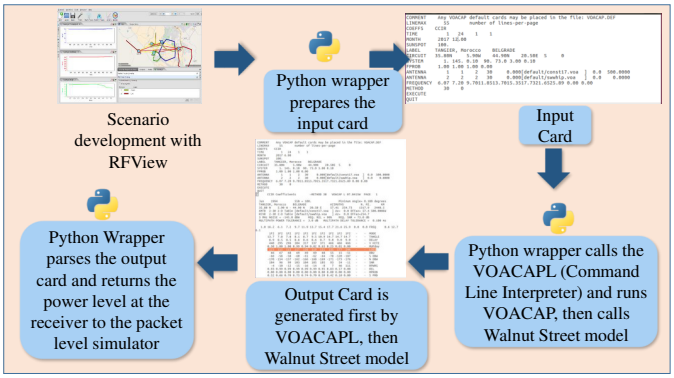

Fig. 2. HF channel simulator integration with WBFH network simulator.

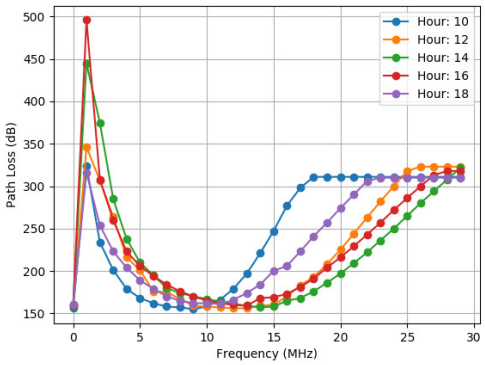

Fig. 3. Frequency vs. path loss.

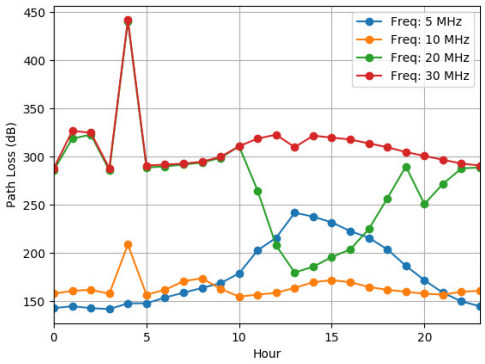

Fig. 4. Hour vs. path loss (hour 0 represents midnight).

The `pythonProp` library calculates path lengths up to 7,000 km and uses an empirical formulation for the distances beyond 9,000 km [10]. The inputs to `pythonProp` are transmitter location (latitude and longitude), frequency, power in the units of Watts, antenna type, gain and height, receiver location (latitude and longitude), noise (in dBW/Hz), minimum take-off angle (in degrees), reliability (in percentage), required SNR (in dB/Hz), multipath tolerance (in dB), maximum delay (in of milliseconds), multipath tolerance (in dB), time window and smoothed sunspot number (SSN). The graphical output provides the propagation results and performance of frequencies in the range of 2-30 MHz over a 24-hour period. For the point-to-point mode, the year and month of the transmissions can be defined. Different transmission times and frequency bands can be evaluated at the same time. For the area predictions, the year and month of the transmissions, as well as the frequency and time zone are defined. The area and grid size can also be defined. A Python-based wrapper is built to leverage `pythonProp` by calling its command line interface VOACAPL and integrate the path loss results into the Walnut Street channel model.

B. Integration with WBHF Network Simulator

The HF channel simulator is integrated with the network simulator, as shown in Figure 2. Parameters (such as location of transmitter and receiver, antenna types, and bandwidth) are input from the RFView software and sent to the Python-based wrapper on the local socket. Using these parameters, the Python-based wrapper prepares the input card for VOACAP command line interpreter (VOACAPL) and runs VOACAPL to obtain the long time scale path loss at each HF frequency. The output is parsed by the Python-based wrapper. Figure 3 and Figure 4 show path loss vs. frequency and hour, respectively, for March 5, 2018.

A RESTful web service is implemented in a virtual machine. The web service takes source and destination locations as input and returns path loss and best operating frequency. Next, the scenario is set up in RFview, which automatically updates the xml, where the scenario details are stored. A new message is created to broadcast the path loss from source to destination. The new HF path loss values are broadcast from a new multicast IP to the EMANE packet level simulator.

III. Routing for WBHF Communications

When the point-to-point HF links between source-destination pairs are not available due to sudden changes in the HF channel, packets can be delivered through relay nodes. Performance is compared for Wireless Address Resolution and Routing Protocol (WARRP) [11] (specifically designed for HF networks) and Optimized Link State Routing (OLSR) [12] for a WBHF communication network with ground nodes and airplanes. WARRP is designed for wireless networks with subnetworks, where the response types to a node's routing request can include: *identity* (the responder is the desired destination), *direct connectivity* (the responder can relay to the destination), *gateway* (the responder has no direct connectivity information, but is connected to other wide-area network(s) that may be able to reach the destination), and *relay to gateway* (the node can relay traffic to a gateway as the best option). A node selects its next hop node from one of the response types. OLSR is a proactive link-state routing protocol, which aims to minimize the number of hops towards the destination and uses hello and topology control (TC) messages to discover and disseminate link state information in the network. As the topology information becomes easily stale with network mobility, OLSR's attempts to collect and exchange excessive network information may result in significant overhead [13].

The WBHF network simulator in Section II is used to generate two subnetworks, one with three ground nodes 1, 2, 3 (positioned at fixed locations $(100, 100, 0)$, $(100, 700, 0)$,

$(700, 100, 0)$ miles) and the other one with two mobile nodes 4 and 5 (moving at speed 540 mph within $[0, 800] \times [0, 800] \times [5, 10]$ miles) such as airplanes. Nodes 1 and 4 are gateways and the transmission between them is always successful. For transmissions between other nodes, data cannot be received if the distance is more than 1000 miles (when the signal level drops below the noise floor), and can be received with otherwise certain probability (according to the SINR computation in WBHF channel simulator). Two cases are considered:

- Case (1): two nodes in different subnetworks are connected if their distance is less than 1000.
- Case (2): subnetworks are connected by gateways only.

The link capacity of 120 kbps is shared by data and control messages. OLSR is implemented with information exchanged every minute to reduce its overhead similar to the level incurred by WARRP. Routing results are shown in Table I.

TABLE I
PERFORMANCE OF ROUTING.

Case	Routing	Overhead (bps)	Delivery ratio	Delay (ms)	Throughput (kbps)
(1)	OLSR	3.33	81.04%	2.7	97.25
	WARRP	5	81.04%	2.7	97.24
(2)	OLSR	0.73	29.00%	3.1	34.80
	WARRP	5	80.93%	4.5	97.11

TABLE II
PERFORMANCE OF ROUTING AND PACKET CODING COMBINED.

Case	Coding	Routing	Throughput (kbps)
(1)	ACO	OLSR	46.82
		WARRP	46.82
	CC	OLSR	38.41
		WARPP	38.41
(2)	ACO	OLSR	21.95
		WARRP	46.82
	CC	OLSR	17.15
		WARPP	38.18

TABLE III
PERFORMANCE FOR MULTICAST.

Case	Routing	Throughput without coding (kbps)		Throughput with coding (kbps)
		$D = 2$	$D = 3$	($D = 2$ or $D = 3$)
(1)	OLSR	78.81	63.87	97.25
	WARRP	78.81	63.86	97.24
(2)	OLSR	10.09	2.93	34.80
	WARRP	78.59	63.61	94.30

For Case (1), OLSR and WARRP have similar performance (success ratio, delay, and throughput). For case (2), OLSR's overhead is reduced with fewer links. However, its success ratio and throughput become much worse than WARRP, while its delay performance (average delay among successful end-to-end deliveries) is better, since packets transmitted along longer paths are likely lost. On the other hand, WARRP can reliably deliver packets to destinations over longer paths.

IV. PACKET CODING FOR WBHF COMMUNICATIONS

Coding at packet level such as network coding has been successfully applied to improve the achievable throughput in wireless networks [14], [15]). Using receiver feedback, packet coding is known to optimize the throughput over wireless links [16]. In addition, packet coding shapes the spectrum and makes it easier to discover the spectrum opportunities [17]. Since WBHF communications need to operate with small overhead for efficient use of link capacity for data communications, feedback mechanisms such as end-to-end acknowledgements cannot be effectively applied for reliability assurance. Instead, packet coding is applied at the source by generating different coded combinations of original packets. The number of these coded packets is slightly larger than the number of original packets in the batch (or generation) considered for transmission. This controlled redundancy increases the reliability and limits the end-to-end feedback. The coding coefficients are selected to ensure linear independence of coded packets. Routing mechanisms in Section III carry the coded-packets to the destination(s) that decode these packets to recover source messages. Reliability is achieved though coding diversity while using the User Datagram Protocol (UDP). Some applications need to run with TCP that requires end-to-end feedback. The interoperability with TCP will be addressed in Section V. Reliability of end-to-end packet delivery is considered at two levels: 1) all packets need to be successfully received by destinations to contribute to the throughput; 2) each packet delivered to destinations counts towards the throughput. The first level is assumed for command and control (C2) applications, where all data components (e.g., mission orders) are critical for mission success.

In packet coding, no feedback is sent from destinations to sources until the complete message is decoded. Hence, intermediate acknowledgement (ACK) messages are eliminated, limiting the feedback to a single bit transmitted at the end, to acknowledge reception of all packets in an entire message. Two low-complexity packet coding schemes are applied to improve reliability of WBHF communications: adaptive coding optimization (ACO) [18], and chunked coding (CC) [19].

ACO optimizes the number of coded packets to be transmitted, K, for a given number of original packets, M, considering the estimated probability of packet erasure on the end-to-end path. The big block of coded packets is then transmitted once, and this transmission should suffice to decode the original packets at the destination. The value of K is adapted over time using the limited number of end-to-end feedback messages.

CC aims to improve the computational efficiency of random linear codes. The original message comprising M packets is first split into L chunks and encoding only uses packets within the same chunk. While chunks used to generate an encoded symbol can be adapted with end-to-end delivery feedback on individual chunks, a random selection of chunks is more suitable for WBHF communications, because it avoids the necessary feedback messages used in other chunked schemes to notify the sender that a given chunk has been decoded and the sender should avoid sending packets from that particular chunk. CC chunks the coded transmissions, where the total number of coded transmissions is optimized to K by ACO.

In both schemes, the coding coefficients (smaller in size compared to packets) are carried along with coded packets. The overhead of joint routing and packet coding consists of

1) *routing overhead* (already taken into account in the results presented in Table I of Section III);
2) *redundancy in coded packet transmissions* ($K - M$ additional packets);
3) *coding coefficients* (the number of symbols per packet is $8Bq$, where q is the coding field size such that each symbol is represented by q bits and each packet in the original message has fixed length of B bytes).

Packet coding (ACO or CC) is combined with routing algorithms (OLSR or WARPP) for performance evaluation of WBHF network communications. Throughput is measured by accounting for all three types of overhead and averaged over 100 simulations. For $M = 100$, $L = 50$, $q = 8$, and $B = 1000$, results are shown in Table II for the two cases from Section III. Note that the performance of CC is expected to improve as the chunk size L is optimized. Table II of Section III shows the throughput at the routing layer (namely, it counts all packets delivered to destinations). However, at the application level, packets contribute to the throughput only if all the M packets in original message are delivered to the destination. Thus, the throughput shown in Table II is smaller than the throughput shown in Table I, but it is achieved with reliability assurance.

Next, multicast traffic is considered from a source to a set of D destinations. A packet contributes to the throughput only if it is received by all destinations. Coding improves the throughput of multicast communications over HF channels significantly, as shown in Table III. As D increases, throughput without coding drops significantly, while coding sustains its throughput. The reason is that coding across packets allows error/erasure correction at multiple destinations at the same time, while uncoded transmissions may ensure reliable delivery of packets to some destinations but they are unlikely to achieve reliable packet delivery to all destinations.

V. PEP to Support TCP for WBHF Communications

While reliability can be supported with packet coding as discussed in Section IV, various applications such as multimedia run on TCP to support reliable communications at the transport layer of network stack. TCP over HF links tends to work successfully only under a limited set of channel conditions because of its slow start and congestion avoidance mechanisms. The TCP performance is severely limited by the HF link characteristics. To provide an acceptable throughput through TCP, a PEP [20] is added on top of WBHF routing to improve the performance on network paths, where native performance suffers due to characteristics of a link or subnetwork. The PEP is used to isolate TCP from links over which it cannot be supported directly (due to error rate, latency, or other issues). The PEP intercepts IP packets as they are being routed, and masquerades as the TCP final destination to carry on the transfer with the origin TCP node. Packets are then transported through the degraded link (typically using a different protocol specifically adapted to the physical channel), where the receiving end of the PEP establishes a separate TCP connection with the real destination node, completing the transfer. TCP at both the source and destination nodes are thus isolated from the difficulties of the intermediate channel and are provided with the illusion of a normal direct connection.

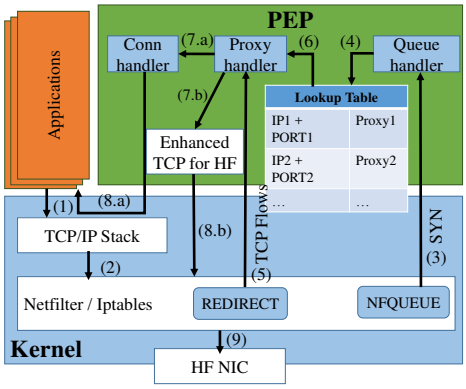

Fig. 5. Architecture of PEP implementation for WBHF communications.

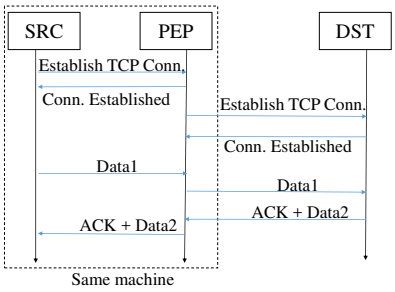

Fig. 6. Data flow of PEP.

PEP implementation for WBHF communications is outlined in Figure 5. A *split, integrated, and asymmetric PEP* is developed by tailoring a simple open-source implementation of PEP for satellite links (with long propagation delays similar to HF) called PEPsal [20]. The data flow of PEP is shown in Figure 6. The developed PEP implementation seeks to improve the performance by adopting the TCP Hybla [21] for TCP connection 2. TCP Hybla aims to eliminate penalization of TCP connections that incorporate a high-latency terrestrial or satellite radio link, due to their longer round-trip times (RTTs). By setting up `netfilter/iptables` framework [22], PEP splits the TCP connection between the source and the destination into two separate connections, namely the connection between the source and the PEP, and the connection between the PEP and the destination.

Figure 5 shows nine steps of PEP implementation for WBHF communications. When a local application (the sender) tries to make a TCP connection with a remote server (the receiver), the TCP packages first go through the TCP/IP stack (step 1) and are captured by `netfilter/iptables` framework in the kernel (step 2) before being sent out through the network interface. A `netfilter` "NFQUEUE" target is created to match the incoming TCP segments having their SYN flag set, to allow the queue handler to communicate with the `netfilter` by the `libnetfilter_queue` library [22] for getting information about the TCP connection source/destination IPs and ports (step 3). For each TCP connection, the queue handler creates a new proxy with two

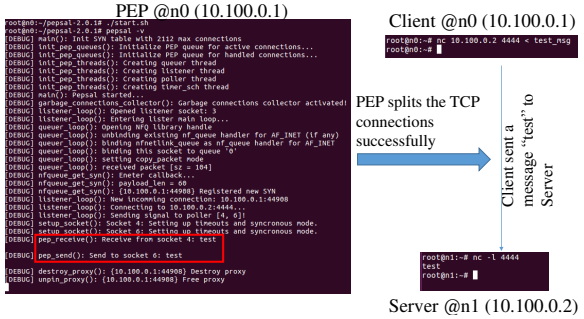

Fig. 7. Demonstration of PEP to split TCP connections.

endpoints (i.e., one for the connection between the sender and the PEP, and the other for the connection between the PEP and the receiver) and stores the proxy information into a shared-memory lookup table with IPs and ports as the search key (step 4). Then, the SYN packet is released and redirected to TCP port 6001, where the proxy handler is listening for it (step 5). If the IPs and ports are found in the lookup table (step 6), the proxy handler triggers the connection handler, which pretends to be the destination and acknowledges the sender application (step 7.a and step 8.a). Meanwhile, a new TCP connection is attempted towards the real destination optimized by our enhanced TCP protocols (step 7.b), and re-injected into the kernel (step 8.b). Finally, the TCP packets are sent out through the network interface controller (NIC) in step 9.

The performance of PEP is evaluated in the WBHF network simulator. Two network nodes (n0 and n1) are created, each running in a Linux container. The link between them is simulated by EMANE. The IP address of n0 and n1 is 10.100.0.1 and 10.100.0.2, respectively. On n1, a TCP server is set up using Linux `netcat` tool (`nc`), which listens on port 4444. On n0, PEP splits every outgoing TCP connection into two and copy-and-forward data from one connection to the other. `nc` is used to send a test message with content "test" to the server on n1. As shown in Figure 7, PEP can correctly intercept the message and transmit the message out.

`iPerf3` is used to measure the available TCP/UDP bandwidth on a path between two nodes. As shown in Figure 7, two HF nodes (n1 and n2) are created using the WBHF network simulator. On node 1, `iPerf3` is run in server mode. On node 2, `iperf3` is configured to client mode and traffic is generated to measure the bidirectional bandwidth between two nodes. The average bandwidths of the sender-to-receiver and receiver-to-sender links measured at n1 (the receiver) are 27.9 Kbps and 22.3 Kbps, respectively. PEP is run on node 2 (the sender). The average bandwidth of the sender-to-receiver link increases to 31.3 Kbps. PEP improves the TCP performance by 12%. PEP is implemented at the sender. Thus, the performance of the backward link from n1 to n0 is slightly impaired due to the copy-and-forward overhead of PEP. The improvement of PEP will increase by tuning other PEP parameters.

VI. Conclusion

This paper discussed challenges for WBHF communications and presented a high fidelity WBHF network simulator and novel network design solutions to improve the performance of WBHF network communications. The WBHF network simulator is used to implement, test, and evaluate network protocols with real packet traffic and under realistic HF channel conditions that are generated by an integrated HF channel simulator, accounting for long, intermediate and short time scale channel effects in the HF spectrum. The WBHF network design combines routing with packet coding or PEP enhancement of TCP to support reliable WBHF network communications. The feasibility of these design solutions is demonstrated with realistic tests in the WBHF network simulator using both unicast and multicast packet traffic.

References

[1] J. Yackoski, B. Azimi-Sadjadi, A. Namazi, J. H. Li, Y. E. Sagduyu, and R. Levy, "RF-NEST: Radio Frequency Network Emulator Simulator Tool," *IEEE Military Communications Conference (MILCOM)*, 2011.
[2] K. J. Kwak, Y. E. Sagduyu, J. Yackoski, B. Azimi-Sadjadi, A. Namazi, J. Deng, and J. H. Li, "Airborne Network Evaluation: Challenges and High Fidelity Emulation Solution," *IEEE Comm. Magazine*, Oct. 2014.
[3] Y. E. Sagduyu, S. Soltani, T. Erpek, Y. Shi, and J. Li, "A Unified Solution to Cognitive Radio Programming, Test and Evaluation for Tactical Communications," *IEEE Comm. Magazine*, Oct. 2017.
[4] E. E. Johnson, "The Walnut Street Model of ionospheric HF Radio Propagation," *NMSU Technical Report*, 1997.
[5] S. Soltani, Y. E. Sagduyu, Y. Shi, J. Li, J. Feldman, and J. Matyjas, "Distributed Cognitive Radio Network Architecture, SDR Implementation and Emulation Testbed, *IEEE MILCOM*, 2015.
[6] J. Ahrenholz, T. Goff, and B. Adamson, 'Integration of the CORE and EMANE Network Emulators, *IEEE MILCOM*, 2011.
[7] C. Watterson, J. Juroshek, and W. Bensema, "Experimental Confirmation of an HF Channel Model," *IEEE Trans. Comm. Tech.*, Dec. 1970.
[8] Pythonprop [Online]. Available: https://github.com/jawatson/pythonprop.
[9] VOACAP Quick Guide [Online]. Available: http://www.voacap.com.
[10] ITU-R, "Software, Data and Validation Examples for Ionospheric and Tropospheric Radio Wave Propagation and Radio Noise," [Online]. Available: http://www.itu.int/en/ITU-R/study-groups/rsg3/Pages/iono-tropo-spheric.aspx.
[11] E. Johnson, Z. Tang, M. Balakrishnan, H. Zhang, and S. Sreepuram, "Routing in HF Ad-Hoc WANs," *IEEE MILCOM*, 2004.
[12] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)", IETF: The Internet Engineering Taskforce RFC 3626, 2003.
[13] Y. E. Sagduyu, Y. Shi, T. Erpek, S. Soltani, S. J. Mackey, D.H. Cansever, Mitesh P. Patel, B. F. Panettieri, B. K. Szymanski, and G. Cao, "Multilayer MANET Routing with Social-Cognitive Learning," *IEEE MILCOM*, 2017.
[14] Y. E. Sagduyu and A. Ephremides, "On Joint MAC and Network Coding in Wireless Ad Hoc Networks," *IEEE Trans. Inform. Theory*, Oct. 2007.
[15] Y. E. Sagduyu and A. Ephremides, "Some Optimization Trade-offs in Wireless Network Coding," *Conference on Information Sciences and Systems (CISS)*, 2006.
[16] Y. E. Sagduyu, L. Georgiadis, L. Tassiulas, and A. Ephremides, "Capacity and Stable Throughput Regions for Broadcast Erasure Channel with Feedback -An Unusual Union-," *IEEE Trans. Inform. Theory*, May 2013.
[17] A. Fanous, Y. E. Sagduyu, and A. Ephremides, "Reliable Spectrum Sensing and Opportunistic Access in Network-Coded Communications," *IEEE J. Sel. Areas Commun.*, Mar. 2014
[18] Y. Shi, Y. E. Sagduyu, J. Zhang, and J. H. Li, "Adaptive Coding Optimization in Wireless Networks: Design and Implementation Aspects," *IEEE Trans. Wireless Comm.*, Oct. 2015.
[19] P. Maymounkov, N. J. A. Harvey, and D. S. Lun, "Methods for Efficient Network Coding," *Allerton Conf. Commun., Control. Comput.*, 2006.
[20] C. Caini, R. Firrincieli, and D. Lacamera, "PEPsal: A Performance Enhancing Proxy designed for TCP satellite connections," *IEEE Vehicular Technology Conference (VTC)*, 2006.
[21] C. Caini and R. Firrincieli, "TCP Hybla: A TCP Enhancement for Heterogeneous Networks," *Int. Jour. of Sat. Comm. and Netw.*, 2004.
[22] "Netfilter: Firewalling, NAT, and Packet Mangling for Linux," [Online]. Available: http://www.netfilter.org.