Pub/Sub in the Air: A Novel Data-centric Radio Supporting Robust Multicast in Edge Environments

Mohammed Elbadry, Fan Ye, Peter Milder, Yuanyuan Yang
Electrical and Computer Engineering Department
Stony Brook University
New York, USA
{mohammed.elbadry, fan.ye, peter.milder, yuanyuan.yang}@stonybrook.edu

Abstract*—Peer communication among edge devices (e.g., mobiles, vehicles, IoT and drones) is frequently *data-centric*: most important is obtaining data of desired content from suitable nodes; who generated or transmitted the data matters much less. Typical cases are robust one-to-many data sharing: e.g., a vehicle sending weather, road, position and speed data streams to nearby cars continuously. Unfortunately, existing address-based wireless communication is ill-suited for such purposes. We propose *V-MAC*, a novel data-centric radio that provides a *pub/sub abstraction* to replace the point-to-point abstraction in existing radios. It filters frames by data names instead of MAC addresses, thus eliminating complexities and latencies in neighbor discovery and group maintenance in existing radios. V-MAC supports robust, scalable and high rate multicast with consistently low losses across receivers of varying reception qualities. Experiments using a Raspberry Pi and a commodity WiFi dongle based prototype show that V-MAC reduces loss rate from WiFi broadcast's 50–90% to 1–3% for up to 15 stationary receivers, 4–5 moving people, and miniature and real vehicles. It cuts down filtering latency from 20μs in WiFi to 10μs for up to 2 million data names, and improves cross stack latency 60–100× for TX/RX paths. We have ported V-MAC to 4 major WiFi chipsets (including 802.11 a/b/g/n/ac radios), 6 different platforms (Android, embedded and FPGA systems), 7 Linux kernel versions, and validated up to 900Mbps multicast data rate and interoperation with regular WiFi. We will release V-MAC as a mature, reusable asset for edge computing research.

***Index Terms*—Wireless edge Communication, data-centric networks, multicast, MAC protocol.**

I. INTRODUCTION

Peer communication among mobiles, vehicles, IoT and drones in the emerging edge computing environment is highly dynamic. Which nodes and what data exist nearby are usually unknown beforehand. Heterogeneous, densely deployed sensors (e.g., cameras, radars, lidars, IMUs, temperature/humidity/presence) produce rich varieties of data. Under constant and possibly high mobility (e.g., moving vehicles, flying drones), neighbors within communication range can change in seconds.

Such edge communication is frequently *data-centric* in nature: equally or more important is that the data is of desired content (e.g., continuous streams of nearby road, weather, and traffic conditions). Who generated, cached or transmitted the data is often lesser a concern, provided the data authenticity can be verified.[1] One-to-many sharing is common: multiple nearby cars all want to receive streams produced by a vehicle.

Unfortunately, current wireless technologies (e.g., WiFi [1], [2], DSRC [3], [4], [5] and V2X [6], [7]) remain largely ill-suited for such dynamic, data-centric, one-to-many communication. First, they use an *address-based, point-to-point* abstraction. E.g., a WiFi sender initiates transmission by explicitly specifying the destination group and node addresses (BSSID, MAC) in frames. A receiver's radio filters decoded frames first by the group and then node address, and retains only those carrying matching addresses. This *a priori, static binding* requires the intended receivers and their group and node addresses be decided before transmission. Mechanisms for group and node discovery and management (e.g., beacons, station profiles for joining and staying in proper groups) become inevitable. They incur complexity and latency excessive or infeasible in highly dynamic edge environments (e.g., joining a weather sensor's group may take a flying drone several seconds, much longer than that of the fly-by time or data downloading time).

Second, the support for robust multicast is almost non-existent. Per-frame acknowledgment and retransmission mechanisms are designed to deliver high data rates for unicast. Multicast can only use the lowest base data rate (e.g., 6Mbps using BPSK modulation and 1/2 coding rate [1], [2]), and has little specification on feedback mechanism for robustness, leading to severe losses [8], [9], [10]. Although there has been a plethora of WiFi multicast work [11], [12], [13], [14], [15], they do not modify and often rely on WiFi's point-to-point abstraction (e.g., unicast to one receiver and having others overhearing in promiscuous mode), thus retaining the baggage of address/group discovery and formation.

We propose a bold approach: using a *pub/sub* abstraction [16], [17], [18], [19] at wireless MAC layer to eliminate addresses, accompanying complexities and latencies. A receiver announces what data it needs by specifying the desired attributes of data. Any sender possessing respective data can transmit frames carrying respective data. The receiver examines whether the data's attributes match desired ones, and if so passes the frames to higher layers. Such *receiver-initiated pub/sub* communication achieves *late, on-the-fly binding*: each neighbor decides whether and how it should respond based on the desired data's attributes; no prior discovery nor determination of destination group or node addresses is needed.

To validate the feasibility of this approach, we have designed and implemented V-MAC, a data-centric radio supporting a topic based pub/sub abstraction. It filters incoming frames by *data names*, and offers robust multicast by an efficient, scalable feedback mechanism: no matter how many receivers exist, a few representatives (usually one) notify the sender of missing frames for retransmission. Thus all receivers have consistently low loss, despite varying reception qualities.

We have verified our clean slate V-MAC kernel modules on

---

[1] We will discuss this in Section VI.

top of 4 major WiFi chipsets, including Qualcomm Atheros ath9k_htc [20]/ath10k [21], RealTek [22] and MediaTek [23]. We have tested on 10 different (expected support > 60) commodity 802.11 a/b/g/n/ac radios, and reached up to 900Mbps multicast data rate on 11ac ones. We have ported V-MAC to 7 different Linux kernel versions, 6 different platforms (Jetson TX2, Rock64Pro, Raspberry Pi, Xilinx FPGA, Android and x86), and demonstrated co-existence and interoperability with regular WiFi (including 802.11n/ac). We want to make V-MAC a mature, low-cost, reusable research asset readily adoptable by the community. We make the following contributions in this paper:

- We propose a data-centric radio supporting a pub/sub abstraction fundamentally different from existing wireless communication. It filters decoded frames by comparing against the names of desired data at $O(1)$ amortized complexity. V-MAC eliminates the need, complexity, and latency in discovery and management of groups and nodes, including beacons, group formations, and address translations (e.g., ARP), providing much simpler and faster network stacks suitable for dynamic edge communication.
- We design a data-oriented acknowledgment (DACK) mechanism where consecutive frames are transmitted back to back in bursts, and usually only one receiver reports missing frames. Such feedback aggregated over multiple receiver-transmissions avoids expensive per-receiver-transmission feedback, and it is compatible with both the standard (address-based) and data-centric stacks.
- We develop a mature, low-cost commodity Pi, WiFi dongle based prototype and conduct extensive experiments in stationary scenarios (up to 15 indoor receivers) and three mobile scenarios (4 people, 4 miniature cars and 5 real cars outdoors), and find V-MAC reduces receiver losses from WiFi broadcast's 50–90% (stationary) or 80–90% (mobile) to about 1–3%. Compared to WiFi stacks, it cuts down average cross-stack TX/RX latency from 5–8ms to about 80μs (60–100× faster), and matching latency by half (from 20μs to 10μs). The code and documents will be made available for research and prototyping in vehicles, drones, IoT and mobiles.

V-MAC is *not* intended to replace WiFi in mostly stationary settings (e.g., infrastructure mode) where addresses and groups do not change often. It targets the mobile and dynamic edge setting where WiFi's point-to-point abstraction, thus address/group discovery and maintenance become undesirable and unnecessary. Our core contribution is the discovery that a pub/sub abstraction at radio level presents the right solution, and we undertake in-depth research to identify, justify and integrate multiple design/implementation techniques (e.g., backoff, hashing, packet injection, netlink) to produce a well-rounded, reusable asset on low cost hardware for the research community.

II. BACKGROUND

Address-based Wireless Communication. Existing wireless communication technologies (802.11, Bluetooth, Zigbee, DSRC, 5G) are all address-based. The MAC layer filters frames decoded by PHY, passing up frames only if the destination MAC address is that of the node, the broadcast address, or a multicast address to which the node belongs. 802.11 standards further require two nodes have the same BSSID (48 bit group address) before they can communicate (both infrastructure and ad hoc modes). Two nodes having the same SSID but different BSSID must discover each other from periodic beacon messages, then one (using a TSF counter mechanism [24]) adopts the other's BSSID, and each creates a "station" profile (e.g., in PHY) for the newly discovered neighbor [25]. Otherwise the PHY drops decoded frames carrying different group addresses. This two level filtering ensures complete separation across different groups.

Such filtering leads to several consequences: nodes must carry the same group address to communicate, otherwise even broadcast frames are dropped at PHY; the destination MAC address must be decided a priori, otherwise no node will retain the frame. Thus discovery of neighbors' group/MAC addresses, and which addresses as destinations must be decided before transmission. These cause inevitable complexity/latency (e.g., in periodic beacons, group address convergence, etc.), excessive and at times unacceptable in highly dynamic edge communication. Our measurements find that beacons can consume 30–40% of air time in office environments, and it takes tens of seconds for a few nodes to converge to the same group address. Also, a node joins one group at a time, thus it is unable to obtain desired data simultaneously from multiple groups.

Data-centric Wireless Communication. We propose to: 1) eliminate the concept of "group" and thus its accompanying complexities. Nodes are free to communicate with anyone within radio range without forming groups. WiFi's group concept arises from the assumption that a relatively stable set of nodes will communicate for extended periods of time. This no longer holds for the edge, where a node's neighbors and how they communicate change quickly.[2] 2) We will use a *pub/sub* abstraction to filter frames based on content. A consumer transmits a *"subscription"* carrying attributes of desired data; any overhearing neighbor can examine whether it has desired data, and if so, transmit *"publications."* The consumer retains those publications with matching attributes. This eliminates a consumer's need for discovering/deciding addresses before transmission. Each neighbor decides whether and how to respond individually, making decisions *late* and *on-the-fly*. A consumer can send multiple subscriptions, obtaining data from different "groups" simultaneously and instantly.

Data-centric wireless is a more general *superset*. It can fully support unicast, multicast and broadcast in address-based communication: nodes subscribe to respective addresses as "topics" and senders "publish" on them. Each node does pub/sub on a few mostly predetermined topics (i.e., addresses), while in data-centric it can use large numbers of dynamically created topics. Data-centric nodes may keep addresses as identity differentiators, but no longer for purposes of filtering or communication destinations.

Information-Centric Networking. Information-Centric Networking (ICN) has long advocated using content as a better alternative to addresses for communication. However, without a content-based MAC layer, ICN cannot achieve its full potential. We use NDN [26] to illustrate how V-MAC complements such ICN layers.

NDN has two types of packets: *Interest* and *Data*. A consumer sends an Interest (i.e., subscription) specifying the desired data's name, and whoever has matching content can

---

[2] *802.11p* in DSRC [3], [4], [5] introduces a single global group using *wildcard* BSSID of all '1's, but it is still address based and has no multicast support.

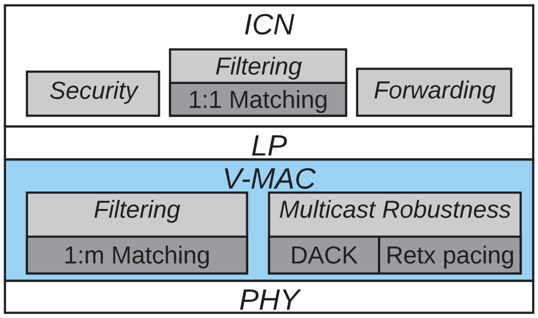


Fig. 1: V-MAC has name-based filtering for matching frames and DACK/Retransmission for multicast.

return Data (i.e., publication) packets. Each node has a *pending interest table* (PIT) to store all Interests received but whose matching Data have not come back yet, a *content store* (CS) to store cached data, and a *forwarding information base* (FIB) to decide to which neighbor(s) to further forward an Interest. An incoming Data packet will be matched against Interest names in the PIT. If matches exist, the packet will be passed to local applications and the neighbors that requested the data.

In NDN, data are named in a hierarchical form with "/" as boundaries, similar to topic-based pub/sub [27], [28], [19]. A one minute video clip of an accident happening at 12:30 on 5th Avenue between 25th and 30th Street in New York City may be named as *"nyc/5th-ave/25th/30th/ accident-video/ 05042018 / 12:30-12:40/"*. Naming conventions [29], [30], [31] are designed to ensure all parties derive the same name for the same data. Due to limitations in regular WiFi (e.g., low base rates and lack of robustness in multicast), vehicular experiments of NDN over WiFi [10], [31], [32], [33] had high losses, indicating the need for a data-centric wireless MAC layer.

V-MAC offers low-loss, high rate, inherently best-effort pub/sub communication *within one-hop*. Issues such as multi-hop routing (and state maintenance like FIB in NDN), data caching, and 100% reliability (if needed) belong to the ICN layer. V-MAC's robust radio level pub/sub allows ICN to address those issues much more easily.

III. V-MAC DESIGN

A. Design Goals

V-MAC has two goals: name-based frame filtering and multicast robustness (Figure 1).

Name-based Filtering. V-MAC works with ICN to filter incoming frames by name. To request desired data, a consumer's ICN layer sends an Interest packet carrying that data's name. V-MAC converts it into an Interest frame(s), passing it to the PHY for transmission, and records the data name. A neighbor that has that data sends back Data packets. Since a Data packet can be many MTU sizes, a link protocol (LP) between ICN and V-MAC breaks it into multiple MTU size units, and V-MAC converts each into a Data frame, carrying the same data name but a different sequence number denoting its position within the packet. The consumer's V-MAC receives and matches incoming Data frames' names against recorded names of desired data. Only frames of matching names are passed up to the ICN layer. The LP reassembles such frames into one Data packet when needed.

Multicast Robustness. To support robust multicast, we develop an efficient, scalable, data-oriented acknowledgment (DACK) aggregated over multiple receiver-frames. A sender transmits multiple data frames back to back (called a *"burst"*). This differs from WiFi A-MPDU aggregation [34] that uses one PHY preamble to send multiple frames in one transmission. If the single preamble does not properly synchronize the antenna circuits, all frames are lost. Our burst uses one preamble to transmit each frame, thus one synchronization error loses only one frame. This also allows other radios to compete and grab the medium, making it fair game instead of monopolizing the medium for lengthy periods (e.g., A-MPDU).

Feedback denoting the sequence numbers of missing frames is provided only after each burst. By doing so, we eliminate per-frame acknowledgment feedback and millisecond level backoff by the sender upon hearing such frames (without which the sender can grab the medium in tens of microseconds to transmit the next data frame) thus increasing air time for data transmission. To avoid per-receiver feedback, some representative(s) acts on behalf of all receivers, notifying the sender of the sequence numbers of missing frames. The sender then retransmits to make up for those losses. DACK is agnostic to the type of filtering, and can work in both data-centric and address-based networks.

In addition, we want a design that can deliver robust performance on even low-cost commodity hardware (e.g., Raspberry Pi, WiFi dongles). This minimizes the adoption barrier so researchers can use V-MAC easily in their testbeds.

Challenges. The V-MAC design must address the following questions: 1) how to eliminate address discovery and group formation for a data-centric MAC layer; 2) how to fit variable sized, possibly long hierarchical data names in limited size frames, while achieving fast matching against large numbers of data names; 3) how to find a suitable representative to send aggregated feedback without prior knowledge or explicit coordination between consumers, and how to ensure efficient air time utilization despite varying numbers of receivers and intensities of background traffic. We eliminate beacons and use a hash based encoding to address questions 1 and 2 (Section III-B), and use a backoff mechanism for question 3 (Section III-C).

B. Data-Centric Frame Filtering

In this section, we describe how we achieve *Beaconless Design* by leveraging V-MAC frames to carry the beacon's functionalities and how the *Lingering Encoding Table* works to support 1:m mapping of Interest/Data frames (instead of 1:1 Internet/Data packet mapping in NDN).

Beaconless Design. In WiFi radios, periodic beacon messages (transmitted at the lowest 1 or 6Mbps base rate, $\sim$10Hz) carry addresses and supported data rates of a node, fundamental for neighbors to discover this node and form groups [35]. They can consume significant airtime (up to 40% based on our measurements). Data-centric radios do not need MAC addresses or have any explicit notion of groups before transmission. They can eliminate beacons to use the most airtime for data, which is suitable for possibly short contact durations under high mobility.

We piggyback necessary information (e.g., supported data rates by consumers) in Interest frames to replace beacons. Such frames are transmitted at the most reliable base rate (1 or 6 Mbps). Producers obtain information on both desired data and supported rates (e.g., 2 bytes of header) in one frame, so they know what to send and at what rates (e.g., the highest rate supported by all neighboring consumers). This is more efficient because the overhead of conveying the existence of receivers (and their supported data rates, etc.) is paid only when nodes need to communicate. The ICN layer may need to send out Interests periodically to discover nearby data. (Note that this differs from node/group discovery in WiFi, which is a separate overhead that V-MAC eliminates. We discuss this further in Section VI.)

Lingering Encoding Table. Our data-centric MAC filters incoming frames using data names provided by the ICN layer. However, data names in ICN can vary in size and become very long, not suitable for direct embedding into limited size MAC headers. We use a hash function to hash the data name into a fixed size *encoding* field (e.g., 64 bits). The LP layer breaks a long ICN packet with one name into multiple MAC MTU size units, then V-MAC packages them into frames carrying the same encoding but different sequence numbers.

For each outgoing Interest frame, V-MAC adds its encoding in a lingering encoding table (LET) to record what data are requested. Each incoming Data frame is compared against the encodings in the LET. If a match is found, the frame will be passed to the LP for reassembly; otherwise it will be dropped.

We make two comments on the encoding matching: 1) One encoding is used to match and deliver multiple Data frames (1:m matching); whereas in NDN a matching Data packet immediately removes the Interest entry in the pending interest table (strictly 1:1 matching). Since many Data packets are multiple MTU sizes, a *"lingering"* encoding entry avoids repeated transmissions of the same Interest frame, thus increasing air time for Data frames. 2) Each encoding entry in the LET has an expiration time, after which it will be removed. Ideally, by the time the entry expires, all (or most) matching Data frames should come back. The expiration time should be set based on such estimation. If some Data frames still have not come back, the Data packet cannot be fully reassembled. The LP may send the Interest again right before the encoding expiration. This resets the LET entry timeout to allow more matching Data frames to come back.

The encoding size must be chosen properly to balance between matching speed and overhead. V-MAC targets edge applications of mostly local interaction, where both temporal and spatial localities exist: for Interests in the distant past or future, their encodings would have expired or not yet come; devices interact mostly with others in close proximity, limiting the scope of Interest propagation. Thus the number of encodings that exist concurrently in a node's LET is limited. Our current prototype chooses 64 bits and a hash map to store encodings. It achieves $O(1)$ complexity matching up to 2M encodings using 58MB kernel memory (see Section V). The size can be increased if needed. We discuss extension of prefix matching and possible encoding collisions in Section VI.

C. Robust Multicast

V-MAC uses a spatially and temporally aggregated feedback called data-oriented acknowledgment (DACK) to achieve robust multicast. Instead of per-frame feedback, the sender sends multiple frames back to back, which we call a "*burst*," then receives one feedback. One (or a few) representative, usually the receiver missing the most frames, sends the feedback. This is achieved by a backoff mechanism where receivers missing more frames back off less, thus transmitting the DACK earlier. A DACK contains sequence numbers of missing frames. Upon hearing DACK frames of the same burst, other receivers cancel transmitting their DACKs to avoid redundancy. The sender will retransmit those missing frames, then start the next burst.

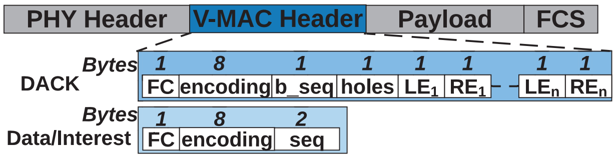


Fig. 2: Structure of a V-MAC header

DACK Frame Format. Figure 2 shows the format of the DACK header, which comes after the PHY (PLCP) header.[3] The first byte is FC (frame control), denoting the V-MAC frame type: Interest, Data, or Control (e.g., DACK). In a DACK, the next 8 bytes are the encoding of the packet's data name, informing the sender which packet's frames are missing when multiple packets are transmitted concurrently. The *b_seq* field denotes the sequence number of the most recent burst heard by this receiver, used to prevent redundant retransmissions (explained later). The next byte is the number of "holes" (frames missing in a row), then come pairs of left edge and right edge (LE, RE), denoting frame sequence numbers before and after each hole. Data/Interest frames have only a sequence number *seq* after *encoding*.

A DACK contains holes in a sliding window of multiple bursts. This allows receivers to request missing frames beyond the most recent burst, when they did not have a chance to request them immediately or successfully receive retransmissions. The size of this window is set comparable to and slightly smaller than the retransmission buffer, whose size is based on available memory and how long frames should be kept for retransmission.

Backoff Mechanism. After a burst, each receiver prepares a DACK frame. A receiver missing no frame also prepares a DACK, with 0 for the number of holes and no LE/RE pairs. Before transmitting its DACK, each receiver waits for a backoff time T

$$T = \alpha\tau, \tag{1}$$

where α is the slot size (described below), and τ is the number of successfully received frames in the last burst. Thus the receiver missing the most frames will back off the least and transmit its DACK earliest. When other receivers overhear a DACK, they cancel their own backoff timers, preventing excessive DACKs.

Our requirements do not guarantee that every receiver's missing frames are subsets of that of the one missing the most frames. We have experimented with having those receivers also send DACKs, but we found that this incurs extra processing time, preventing low-end platforms (e.g., Raspberry Pi) from keeping up with incoming WiFi frames. Given that this design already

---

[3] If WiFi interoperability is needed, a regular 802.11 header is inserted before V-MAC header. See Section V-C.

achieves very low loss in all receivers (see Section V), we decide that the current system achieves our goals without this feature.

When two receivers miss the same number of frames, tiebreaking is needed. We keep CSMA/CA [36] for this purpose; it prevents collisions by sensing the medium before starting any transmission. We find that due to slight differences in timing, the chance that two such receivers sending DACKs at the same microsecond granularity and colliding is very small. Thus CSMA/CA for tiebreaking is effective.

Slot Size. Parameter α denotes the backoff slot size, measured by each receiver and dynamically adapted to background traffic intensity. Ideally, α should be the minimum amount of time needed for a receiver to grab the medium for transmitting its DACK to maximize air time for data.

Because multiple factors can affect this medium grabbing time (e.g., intensity of background traffic, speed of *PHY* hardware interrupt handling by MAC), we leverage the sender's *back to back* Data frame transmissions to measure the slot size: a receiver uses the time difference between receiving two frames of consecutive sequence numbers, minus the Data frame transmission time (dividing the frame size by data rate, while accounting for preamble size and rate). This depicts how quickly the sender can grab the medium.

Asymmetry exists between the sender and each receiver, thus the perceived background traffic intensity can differ. Nevertheless, by extensive experiments we find our estimation is a simple, reasonable approximation and shows high efficiency (Section V-B). If a receiver does not receive any Data frame pairs with consecutive sequence numbers, it estimates α by calculating the average time for receiving the next frame. E.g., if frame 3 and 5 are received but not 4, α is the receipt time difference between frame 3, 5 divided by 2 (also minus Data frame transmission time).

Common Start of Backoff. We use a common event, the receipt of the last frame in the burst, as the common trigger on all receivers to start the backoff timer. This avoids complexities in accurate time synchronization among receivers. A receiver may miss the last frame in a burst, thus losing the common trigger. In such cases, the receiver estimates the expected receipt time of the last frame using its estimated α, and starts the backoff after the estimated receipt time.

DACK Cancellation Policy. We use a simple cancellation policy: neighboring receivers cancel their DACK timers upon hearing 2 DACKs with the same *b_seq*. This helps compensate DACK losses without incurring excessive redundancy, and help recover lost frames not reported in the first DACK. We have tried alternative approaches that examine overheard DACKs, canceling only if all or most locally missed frames are reported. This approach takes too much time and frequently finishes after DACK timer expiration, and deteriorates further under higher data rates (e.g., 900Mbps or higher in 802.11ac). Thus we adopt this simple yet effective policy.

We are aware of potential hidden terminal problems when receivers cannot hear each other to cancel DACKs. This may result in an increased number of duplicate DACKs. Because of the small DACK sizes ($<$70 bytes), this does not present a significant overhead, and our experiments also find minimal collision among DACKs, resulting in no perceivable impact on loss rate among receivers ($<1\%$).

Burst Size. The size of the burst (set to 5 empirically in prototype) represents a tradeoff between efficiency and latency. A larger burst aggregates more feedback using the same DACK overhead, but also incurs longer latency of feedback and retransmission. It should balance applications' tolerance to latency (e.g., more sensitive data should use smaller burst sizes) while not incurring excessive overhead in frame header fields (i.e. LE/RE) and retransmission buffer (explained next).

When a receiver loses a whole burst of frames, a DACK may not be triggered immediately. As long as the sender continues to send, upon reception of any frame in subsequent bursts, the receiver finds it has missed frames and can send DACKs. If the lost frames are the last from a packet and no further frames are transmitted, then the ICN layer will detect absence of complete packets, and trigger retransmission of packets (thus frames).

Retransmission Buffer. The sender keeps in a buffer a sliding window of frames transmitted in recent bursts. Upon hearing a DACK, it retrieves missing frames from the buffer and retransmits them. The window size is multiple bursts, thus allowing missing frames be requested and transmitted more than once to counter DACK or retransmission losses. The window size is limited by available kernel memory, and set to 10 bursts empirically in the prototype.

Retransmission Pacing. We observe that quite often more than one DACK is transmitted after each burst (e.g., earlier DACKs were not overheard to cancel other receivers' DACKs), and a DACK responding to earlier bursts may be transmitted very late due to delays in medium contention and queuing/processing across the radio hardware and network/OS stack.

Such DACKs frequently denote frames that are already reported missing. Although redundant DACKs are small, redundant retransmissions of long Data frames waste significant airtime. To avoid such waste, each DACK carries a *b_seq* field to indicate to which burst the DACK responds. Each receiver sets it as $seq\_latest\%B$, the sequence number of the latest received frame modulo the burst size B.

For each frame in the buffer, the sender keeps a monotonically increasing b_last, the most recent burst after which the frame was retransmitted. A missing frame is retransmitted again only upon a reporting DACK carrying a $b\_seq > b\_last + i$, where i is the *pacing size* and set empirically based on data rate and traffic intensity (see Section V-B). Thus DACKs reporting the same missing frames in the same or next i bursts do not generate retransmissions. We have validated this cuts redundant retransmissions from 40% to almost 0%.

IV. IMPLEMENTATION

We describe three aspects of the implementation on commodity hardware: 1) beaconless communication and firmware improvements; 2) V-MAC core functions and cross-stack latency; 3) co-existence and interoperation with WiFi. Commodity hardware (e.g., Raspberry Pi, WiFi dongles) is cheap and easily available in large quantities for experiments. However, it has significant constraints, posing non-trivial challenges for stable, high performance V-MAC implementation. Next we describe these challenges and how we combine multiple engineering techniques (e.g., intentional "mis-configurations," packet injection, netlink) to build a robust, low-cost system as a research asset to the community.

A. WiFi Radio Exploitation

By standard, WiFi radios do not send data frames unless they have joined a group and know at least one other "station" in that group. The logic is commonly inside PHY hardware/firmware, which create data structures for that group and station. This directly conflicts with V-MAC, which seeks to completely eliminate groups and addresses. All *ath9k_htc* radios have unicast frame rate adaptation control algorithms in firmware without options to disable them. Such algorithms cannot be used for multicast and they lead to uncontrollable fluctuations in experiment results.

Virtual Group and Station. We supply faked network information to a PHY API by calling lower level functions defined in the *ieee80211_ops* struct, so the PHY creates a network structure and believes it has joined that virtual, non-existent "group." Using another API, we make the PHY create a station data structure with all parameters set to its own capability and with the broadcast MAC address. This intentional misconfiguration "coaxes" the PHY to send data frames even though it has not joined any real group nor real neighbors (i.e., "stations").

Based on hardware capabilities, we take two different methods that enable radios to transmit without joining groups or sending beacons: 1) We switch the radio into *monitor* mode, and misconfigure it to belong to a group of a random BSSID. (Monitor mode captures frames from any network, and should not belong to any particular group/network.) This eliminates beacons because WiFi radios in monitor mode do not send beacons. However, monitor mode usually performs passive listening only. We can use a packet injection mechanism [37] to send frames, but only at the lowest 1 or 6Mbps base rate. We find that by "coaxing" the PHY to create a virtual station having broadcast (all 1's) as the MAC address and higher supported data rates, the PHY can transmit (thinking it is sending to that "station") at those rates (e.g., 54Mbps in 802.11n thoroughly tested, and up to 900Mbps in 802.11ac validated). This works on radios that support packet injection in monitor mode, and allows transmission above base rates.

2) If packet injection is not supported, we switch the radio into client mode, and "coax" the PHY to "join" a non-existent AP of the radio's own MAC as BSSID, then have the PHY create a virtual station with the radio's own MAC. We also supply the PHY with a beacon template of the AP's. Under these 3 intentional misconfigurations, the radio stops sending beacons, yet it can transmit data frames without joining any real network.

We have implemented and validated both methods. The first one leverages existing monitor mode and packet injection mechanisms; it requires fewer misconfigurations and is preferred if available. The second one needs more misconfigurations and thus is not as stable as the first.

Firmware Improvements. For open-source firmware (e.g., *ath9k_htc* used for 802.11 b/g/n), we modify the code to support per-frame rate specification from V-MAC; if the firmware source is not available (e.g., *ath10k* used in 802.11ac 5GHz), we enhance the community *ath10kct* driver that supports setting rates from the MAC layer so V-MAC can fix the rate.

We observe significant internal losses (up to 30%) when the MAC layer pumps frames to the PHY at high speeds (e.g., a batch of frames from a long TCP packet arriving at the PHY almost simultaneously). We find the PHY's internal buffer is overwritten by new frames, thus some frames are never transmitted. We add a back pressure mechanism so the PHY can signal the MAC to slow down when its buffer is full; this eliminates internal losses.

Fig. 3: *V-MAC* with an 802.11 header at beginning to allow 802.11 co-existence.

B. V-MAC Functions

V-MAC has two core functions: *Lingering Encoding Table* matching and *multicast robustness via DACK.* Both must be implemented efficiently without consuming excessive computing resources or kernel memory.

We leverage the standard kernel's *rhashtable* library to provide high search speed (O(1)) against millions of data name encodings. Each LET entry has a pointer, pointing to a separate data structure containing all information related to that encoding.This ensures a small entry size (10 bytes) so that multiple entries can be fetched in one page read to speed up search. Each entry has an expiration time and is removed upon timeout.

When only some but not all Data frames have arrived before an LET entry's expiration, the LP can send another Interest of the same name. This extends the expiration time to allow more Data frames to arrive. If the sender/receiver has moved out of range thus no further Data frame can come back, LP may stop trying after some attempts so the LET entry will expire.

DACK uses the back to back reception time difference of consecutive frames to estimate slot size α. However, we find occasionally (10% of the time) multiple frames come up with the same kernel timestamp. This happens because the kernel is not real time; its interrupt handling of a burst of PHY frame receptions operates at millisecond granularity. Thus these frames are processed around the same time and carry the same timestamp. In such cases, we use an empirical value of $50\mu s$ as the sender's medium grabbing time in α estimation.

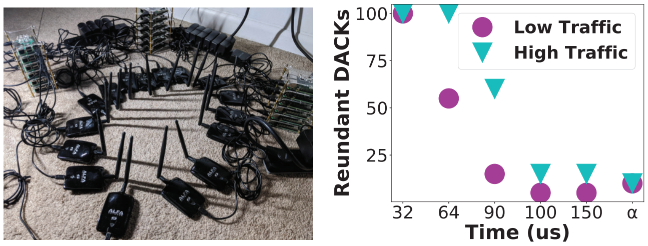
Fast Cross-Stack Paths. We address multiple inefficiencies in the existing Linux network/WiFi stack to improve the cross-stack path latency (100× TX path and 60× RX path): 1) We eliminate unnecessary encapsulations and decapsulations including the legacy 802.3 ones between network and MAC, and the memory remapping ones when frames are passed to/from userspace. 2) There are 5–6 different queues in kernel from userspace to the PHY; we cut down to 3 queues. 3) We use a more efficient *netlink* system call [38] instead of the old Unix socket to pass data from userspace to kernel. We cut down MAC layer code from 20K to 4K LoC (5× reduction): V-MAC has only one mode (WiFi has 6); data-centricity eliminates complexities in the address-based stack (e.g., beacons, stations). The consistent pub/sub abstraction across application, network and MAC enables a much cleaner and simpler stack.

C. Co-existence and Interoperation with WiFi

Co-existence. We find Wi-Fi radios (routers) cannot recognize V-MAC frames and may misinterpret the encoding as the source MAC of a new node. Thus the PHY may keep creating new station data structures, quickly exhausting its resources

Frame Rate	**54 Mbps**
Preamble type	**Long**
Number of frames per-run	**500**
Frame Payload Size	**1024 bytes**
Tx power	**20 dbm**
Interest Size	**70 bytes**
V-MAC burst size	**5 frames**

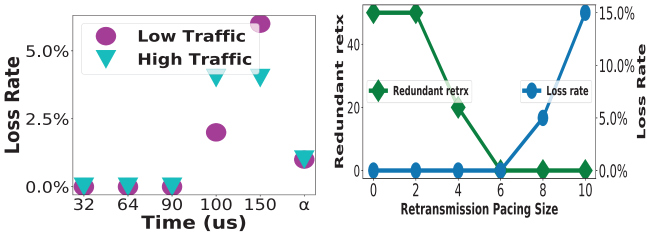
TABLE I: WiFi and V-MAC Configuration



(a) Raspberry Pi Testbed

(b) Slot Size Impact on DACK

Fig. 4: (a) shows V-MAC stationary testbed. (b) shows impact of the waiting slot size on the number of redundant received DACKs under low (10%) and high (40%–50%) medium environments.



(a) Slot Size Impact on Loss

(b) Impact of Retransmission Pacing size

Fig. 5: (a) shows impact of DACK slot size with loss rate under environments of low (10%) and high (40%–50%) medium utilization. (b) shows larger retransmission pacing sizes lead to more losses but less redundant retransmissions, with balance at 6.

and finally crashing. To avoid this, we add an 802.11 header carrying a BSSID different than nearby networks before the V-MAC header (Figure 3). These frames are ignored by WiFi radios because they are not in the same BSSID. V-MAC, on the other hand, is capable of recognizing WiFi frames, and can ignore them if configured so from userspace.

Interoperability. For WiFi radios to receive V-MAC frames, WiFi nodes' MAC, network, and transport layer must be able to parse them properly. In addition to the 802.11 header, we also add IP/UDP headers. Thus respective headers are parsed in these layers and the payload is passed to userspace. A V-MAC header is added at the end if concurrent V-MAC/WiFi reception is needed (validated in Section V-C). V-MAC radios by design are able to recognize all WiFi frames and can be configured to pass them to userspace.

V. EVALUATION

We evaluate V-MAC using Raspberry Pi 3 and Alfa AWUS036NHA WiFi dongles. We first evaluate stationary cases (up to 15 receivers indoors as in Figure 4a), then 3 different *mobility* cases: people walking around (4–6 mph), miniature cars (8–11 mph), and real vehicles (25–30 mph). We use 5 dBi antennas for all experiments except for *real cars* where we use 9 dBi antennas to increase the range.

A. *Experimental Methodology*

We first compare V-MAC broadcast against WiFi ad hoc broadcast. Table I shows the parameters used for both of them. We modify WiFi firmware to fix the data rate at 54Mbps (otherwise ad hoc can only broadcast at 1Mbps). This is a high enough and practically useful rate at which our implementation is robust enough; it also imposes much higher stress on implementation and algorithm robustness than the base rate 1Mbps. We further discuss the rate choice and future adaptation work at high rates in Section VI and VI-B, respectively.

For ad hoc, we ensure all nodes have joined the network under the same BSSID before data transmission. For V-MAC, we ensure all consumers have subscribed first by having the producer wait for an artificial 20 seconds before sending data. We also use identical userspace application programs to ensure fair comparison. Latency is calculated as the duration from the first node sending an Interest to it receiving the last data frame.

Each run consists of one Interest, and 500 Data frames of payload size 1024 bytes sent back to back. We collect 20 runs for each experiment. We first compare V-MAC broadcast and ad hoc broadcast. To minimize impact from varying background traffic, we alternate between V-MAC and ad hoc between runs. Due to a much simpler and cleaner data-centric stack, V-MAC is more efficient and outperforms ad hoc in broadcast. For stationary and mobile cases, we present V-MAC multicast against V-MAC broadcast, and mention briefly ad hoc broadcast results.

B. *V-MAC Benchmark*

We evaluate the impact of the waiting slot and retransmission pacing sizes on V-MAC performance, and stress test the prototype's stability.

Impact of Fixed vs. Adaptive Waiting Slot Size. We compare the adaptive α against fixed waiting slot sizes using one sender and three receivers. We artificially generate a "high traffic" scenario by having two more ad hoc mode Pi's transmitting 900-byte packets at random intervals, and low traffic is without the artificial transmissions. We ensure that the three receivers can hear each other and cancel each other's redundant DACKs. Figure 4b shows the average numbers of redundant *received* DACKs (i.e., not canceled, but transmitted and received by the sender) and loss rates (Figure 5a). We see that no fixed slot size can achieve low redundant DACKs and low losses concurrently. A larger fixed slot allows more time for canceling redundant DACKs, but provides slower feedback thus more losses. A smaller slot has opposite effects. Only the adaptive α achieves both regardless of background traffic intensity. The retransmission pacing (evaluated next) further ensures redundant received DACKs do not lead to redundant retransmissions (i.e., frames all receivers already have).

Retransmission Pacing Size i. We evaluate the impact of the pacing value i on loss rates of 3 receivers and the number of redundant data retransmissions (i.e., all receivers have the frame already). Figure 5b shows that smaller pacing sizes produce less loss yet more redundant retransmissions, and vice versa. We find a value of 6 achieves both low loss (close to 0) and zero redundancy (no wasted retransmitted data frames across all experiments). This best value may vary depending on conditions such as data rates. We also repeat with 10 receivers, and find only slightly increased loss but still zero redundancy. This shows that the DACK cancellation mechanism scales well with more receivers.

Stability under Multiple Concurrent Subscriptions. We stress test V-MAC stability by having 10 nodes each publishing under 10 different data names (500 frames per-data name) and subscribing to 10 other names concurrently. We give 3 minutes timeout between runs, enough for nodes to clean up internal states. We let the test run continuously for 7 days, and observe no kernel crash, and the average loss is 3% among all runs and all consumers across all data names. Besides demonstrating the implementation stability, this shows the design's capability in supporting multiple concurrent data streams while retaining the same low loss rates.

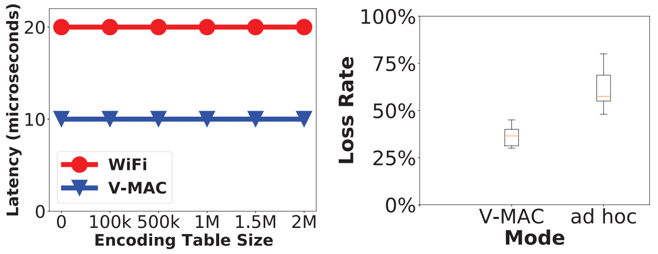
C. V-MAC vs. WiFi

We compare V-MAC against WiFi in matching and cross-stack TX/RX latencies, broadcast loss and latency, and interoperability with WiFi.

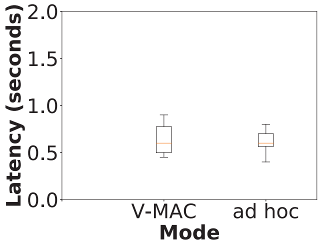
Matching Latency. While WiFi compares against two 6-byte MAC/group address, V-MAC needs to compare against possibly hundreds of thousands of entries in the LET. Figure 6a shows that LET comparison latencies remain constant as the table size increases (WiFi has constant comparison work thus also constant), and LET searching takes only half of WiFi's time (10 vs 20μs). This shows the LET hash table indeed delivers O(1) complexity. It is faster because in WiFi a 6-byte address is folded into pairs to align with physical memory boundaries for comparison, and done for both BSSID and MAC. There also exists redundancy in the WiFi stack (e.g., comparison done in *ath9k_htc* then again in *mac80211*). V-MAC just needs a hash table lookup and can easily map the 64-bit hash using the standard Linux data type u64.

Cross-Stack Latency. We compare the V-MAC stack to the standard WiFi stack and measure how long frames sent by a userspace program take to traverse all kernel layers to reach *ath9k_htc* for final transmission. We find that V-MAC takes a fraction of that of the standard stack (73–100μs vs. 7–10ms). On the reception path, it is 70–90μs vs. 3–7ms. Our clean, efficient stack is about 100× (TX) and 60× (RX) faster than the standard stack. A pure data-centric design allows us to eliminate unnecessary 802.3 encapsulation/decapsulation between network and MAC layers, and cut down the number of queues from 5–6 to 3. More efficient *netlink* instead of the old Unix socket for userspace/kernel communication also helps improve the speed.

Ad hoc vs. V-MAC Broadcast. We compare the loss and latency of V-MAC broadcast and WiFi ad hoc broadcast for 10 receivers. Figure 6b shows that *V-MAC* has about 30–40% loss while WiFi has 55–70%. This is due to a more efficient stack and PHY feedback signals reducing internal losses. The improved loss comes at similar latency (both around 0.6s in Figure 6c), showing V-MAC delivers more frames using similar airtime, thus utilizing the medium more efficiently.



(a) Filtering Latency **(b)** Loss Rate Comparison



(c) Latency Comparison

Fig. 6: (a) shows constant searching time (10μs) in V-MAC as the size of the encoding hashtable increases, smaller than WiFi comparing against one address (20μs). (b) Loss rate and (c) latency comparison for V-MAC broadcast and ad-hoc broadcast. V-MAC has 25% less loss at comparable latency.

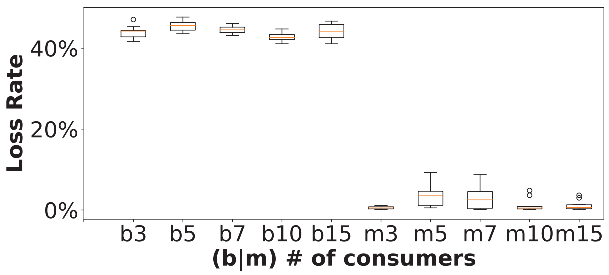
Interoperability. We have 2 WiFi ad hoc receivers and 3 *V-MAC* nodes (1 sender and 2 receivers). All receivers can successfully receive WiFi data frames mimicked by the V-MAC sender at latency similar to standard WiFi transmissions. There are DACKs from V-MAC receivers to cause retransmissions from the V-MAC sender, which benefit both WiFi ad hoc and V-MAC receivers to achieve <3% loss. This demonstrates the feasibility of V-MAC to interoperate with regular WiFi. Supporting sending/receiving with nodes of other WiFi modes requires mostly engineering changes but not much research, so we will do that when real needs arise.

D. Stationary Scenarios

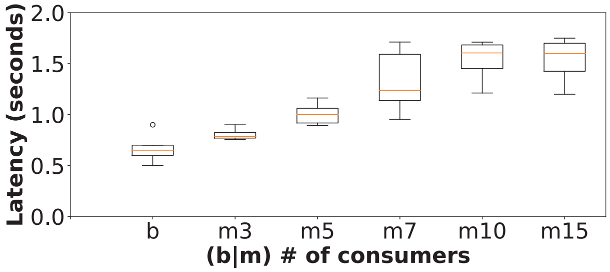
We evaluate how V-MAC scales as more receivers impact the loss rate and latency. Each result is based on 50–60 runs.

Low Loss. Figure 7a shows that V-MAC multicast (with DACK) achieves consistently low 1–3% loss (except a few outliers) as the number of receivers grows from 3 to 15, while V-MAC broadcast produces ∼45% loss. This shows DACK scales well with more receivers and delivers consistently low loss. WiFi ad hoc tested in the same environment exhibits much higher loss (50–90%) due to inefficient stacks and the lack of PHY back pressure signal.

Latency Plateau. We observe that (Figure 7b) at 3 receivers, multicast takes 30% longer than broadcast, but cutting losses from 45% to 3%, effectively delivering 76% more frames ((1-0.03) / (1-0.45) = 1.76) at only 30% longer time. This shows that DACK utilizes the medium much more efficiently for data transmissions. The multicast latency increases gradually as more consumers are added, and plateaus after 10. By this time the retransmissions are



(a) V-MAC Broadcast (b) vs Multicast (m) Loss Rate



(b) V-MAC Broadcast (b) vs Multicast (m) Latency

Fig. 7: (a) and (b) show that V-MAC multicast has low 1–3% loss for up to 15 consumers, yet latency increases gradually and becomes flat after 10 consumers.

sufficient to compensate losses of 5 additional consumers, and most redundant DACKs are canceled because transmitted DACKs are heard by consumers in dense environments (e.g., 10 and 15 consumers). This demonstrates the effectiveness of DACK cancellation policy to achieve few redundant DACKs and low loss.

Loss at Individual Receivers. Figure 8a shows that the broadcast loss rate varies greatly for different receivers by as much as 30%. We find varying the relative distance and angle between sender and receivers does not correlate to the loss rate. V-MAC delivers consistently low loss rate (<3%) across 10 receivers (Figure 8b). We also find more than 60% of DACKs are canceled; combined with retransmission pacing, we observe no wasted retransmission (i.e., sent when all receivers already have the frame). Node 6 with the highest broadcast loss (45–60%) produces the most DACKs, as intended by the backoff design. We test WiFi ad hoc broadcast and find much worse losses: receiver 6 has 80–90% loss in all runs, while all others are at 55–80%.

E. Mobility Scenarios

People Walking. We give each person a Raspberry Pi with a WiFi dongle and power bank to walk in a 24.24m × 9.5m^2 rectangular parking area. We do two scenarios: *platoon* where 3 people (receivers) line up after and follow a leading person (sender) at 1.5 meters spacing, walking along the perimeter of the area; *crossing* where 3 receivers and one sender start from opposite corners respectively, and walk to the other corner; the radios are in range for a duration (about 10 seconds) in the middle, and may be out of the range at the beginning and end of each experiment.

We observe that V-MAC multicast obtains a loss rate of ∼0.3% (platoon Figure 8c) and 2∼3% (crossing Figure 9a). We repeat with V-MAC broadcast, and find 30–40% (platoon) and 35–75% (crossing) losses, much worse due to lack of DACKs and retransmission. WiFi ad hoc produces 70–90% losses, worse than V-MAC broadcast, due to inefficient stacks, time wasted in joining/keeping the same BSSID in address based networks (communication cannot happen if not joined), and lack of PHY back pressure signals.

V-MAC multicast increases latency by 30% over V-MAC broadcast, quite similar to respective stationary results ('m3' vs. 'b' in Figure 7b). The above shows V-MAC utilizes the medium much more efficiently for data, and delivers low loss with moderate latency increase under low mobility.

Miniature Cars. We retrofit remote controlled (RC) cars with Raspberry Pi's, WiFi dongles and power banks (Figure 9b) to test how *V-MAC* performs under medium mobility (8–11mph). Four users control 4 cars, creating the same two mobility scenarios as people walking. We observe that V-MAC has about 0.2% loss in platoon (Figure 10a), and 2–4% in crossing (Figure 10b), comparable to people walking. We also test WiFi ad hoc, and find it has 80–90% loss because most of the time is wasted in group/network (re)formation and little time is spent in communication. This shows that eliminating beacon/group complexity is critical under mobility.

Real Vehicles. We use 5 vehicles (1 sender leading and 4 consumers following forming a platoon) at 25–30mph with 10–20m spacing. They go back and forth on a 1-mile road (Figure 11a) for four round trips. Figure 11b shows all consumers have low loss (1–4%), except vehicle 4 (∼8%) because it is the farthest and occasionally gets out of the range from the sender. The latencies are comparable to other experiments shown before (about 0.8s for sending 500 Data frames).

We observe that as mobility increases, the loss tends to increase moderately. We find that running RC cars at higher speeds (e.g., 20mph) yields similar loss characteristics as real vehicles. This may offer an easier, lower cost means to produce realistic vehicular results at much reduced costs and resources using miniature instead of real cars.

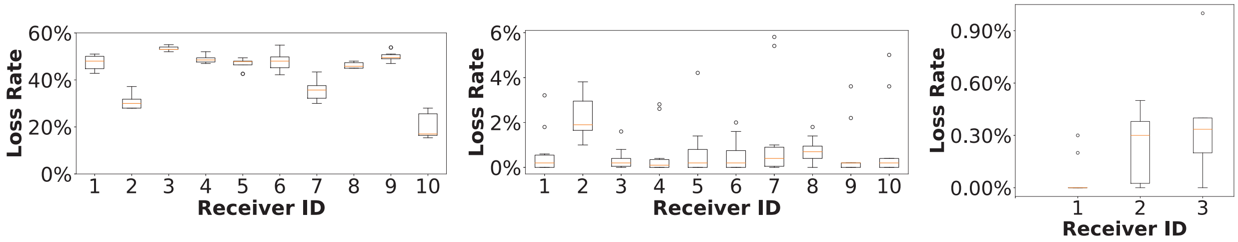
F. Video Multicast Demo

We test V-MAC in a video multicast application where one sender transmits a video to 10 receivers which play it back real time. Figure 12 shows sample video playback screenshots for each of the 10 receivers. In WiFi ad hoc broadcast (Figure 12a), only 2 out of 10 receivers can play back, and only garbled images at 68% and 70% loss, while the remaining 8 cannot receive sufficient data and show black screens. In V-MAC (Figure 12b), all 10 receivers can play the video smoothly, with highest loss at 0.6% and lowest 0.19%.

VI. DISCUSSION AND FUTURE WORK

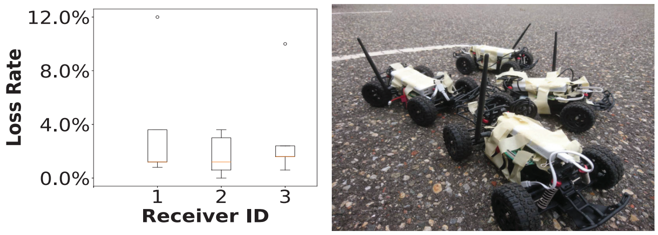
A. Discussion

Data-centricity Advantages. A pub/sub abstraction eliminates the need of any node or group addresses, and the mechanisms and complexities in their management. V-MAC removes beacons thus freeing up significant airtime for data transmissions, which is especially critical for densely deployed IoT devices. It gets rid of multiple layers of address translation (e.g., 802.3, ARP), greatly simplifying the stack and improving the latency. V-MAC uses pub/sub topics to replace group and multicast addresses. Thus it eliminates slow BSSID group convergence and multicast group joining. A node can obtain data



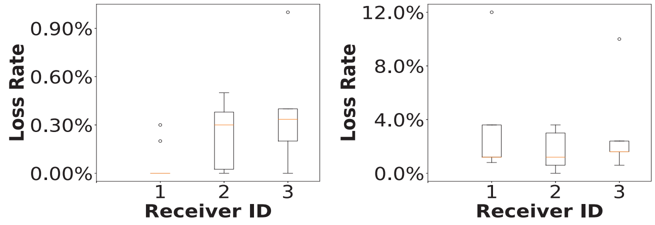
(a) Per-receiver Loss, V-MAC Broadcast **(b)** Per-receiver Loss, V-MAC Multicast **(c)** People Platoon

Fig. 8: (a) and (b) show the loss rate for V-MAC broadcast and multicast for 10 stationary receivers. (c) shows People Platoon scenario loss rate.



(a) People Crossing **(b)** Miniature Cars

Fig. 9: (a) shows Loss rate for people walking in crossing scenario. (b) shows minature testbed used for medium mobility.

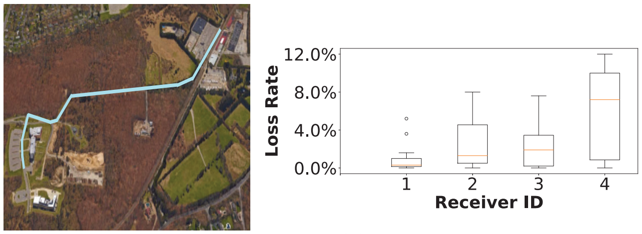


(a) Miniature Cars Platoon **(b)** Miniature Cars Crossing

Fig. 10: Loss rate for miniature cars in platoon (a) and crossing scenarios (b).

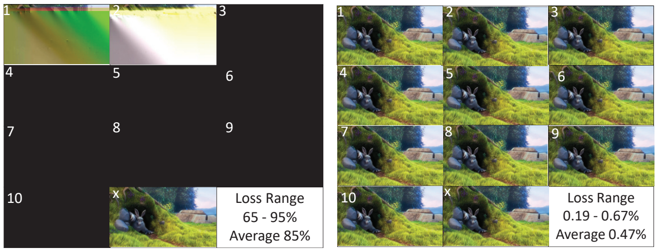
instantly and simultaneously from multiple “groups” by sending subscriptions, without the need of neighbor/data discovery or group joining overhead.

Beyond Nomadic Scenarios. Existing WiFi was designed for mostly *nomadic* mobility where mostly stationary nodes will stay at certain locations long enough (e.g., infrastructure mode). Thus there is sufficient time to discover each other and form networks. WiFi has a host of mechanisms (e.g., choosing SSID, merging groups based on highest *TSF*) to support these operations. However, in edge environments, high mobility (e.g., vehicles, drones) breaks this nomadic mobility and underlying assumptions. WiFi ad hoc mode carries baggage from nomadic mobility and still requires forming groups before communication, leading to unnecessary complexity and latency that render communication virtually impossible in fast-moving environments, as observed in our miniature car experiments. *V-MAC* removes the need of forming groups, thus eliminating those complexities and overheads to suit mobile and highly dense IoT edge communication scenarios.



(a) Real Vehicles Trajectory **(b)** Real Vehicles Loss

Fig. 11: (a) shows Real vehicles Path. (b) shows Real Vehicles Loss, V-MAC Multicast.



(a) Video Playback Screenshots for WiFi ad hoc Broadcast **(b)** Video Playback Screenshots for V-MAC Multicast

Fig. 12: (a) and (b) show representative screen shots for 10 receivers, in WiFi ad hoc broadcast and V-MAC multicast. In WiFi, only 2 out of 10 can display, and only garbled video; in V-MAC all 10 receivers can play smoothly with $<1\%$ loss. ‘X’ is the original screenshot.

Efficient Medium Utilization. V-MAC combines several techniques to deliver more data frames within similar airtime. It uses aggregate feedback to avoid per-frame acknowledgement; it eliminates beacons transmitted at slow base rates with long preambles consuming up to 40% airtime; it eliminates redundant retransmissions that no receivers need. Since both WiFi and V-MAC relies on CSMA, they can effectively share the medium without causing complications for each other.

Data and Channel Discovery. The elimination of address discovery does *not* mean no discovery is needed. In address-based stacks, discovery is done at both MAC and network layers for neighbors and data; in data-centric ones, only data discovery is needed, and by network layer only. V-MAC is designed to work together with an ICN layer, which needs to send discovery packets (e.g., possibly periodic Interests under high mobility) to find what data are available nearby.

V-MAC does not eliminate the need to discover channel frequencies and bandwidths used by neighbors for proper MAC. In WiFi this is done by a node scanning on different channels. We envision that some cross-layer mechanism can augment Interests sent on a commonly used channel for V-MAC nodes to coordinate with each other (e.g., PHY detects Interest frames and attaches frequency/bandwidth information).

Implicit Interests. When many receivers need the same data, having each sending the same Interest may be unnecessary. We can optimize with "implicit" Interests where V-MAC still creates LET entries when receiving Interest packets from ICN. But if it overhears the same Interest frames sent by others, it refrains from passing the Interest to PHY for transmission. This can help scale to very dense environments.

Prefix Matching. The current prototype supports exact name but not prefix matching because a prefix is hashed to a different encoding. One solution is for the producer's LP to add the prefix into replied Data packets, and for V-MAC to add a second encoding of the prefix so receivers can match that prefix's LET entry.

Hash Collisions. Using 64 bit encodings reduces collisions to very small probabilities. When hashes do collide, the LP cannot differentiate which frames belong to which packets, but it can tell a reassembled packet is incorrect from the network level checksum and drop it. LP may send the Interest again but instructing V-MAC to use another hash function so names will not collide again. The above two items require coordination with ICN layer, and will be our next steps.

DACK for Address-based Multicast. DACK can be used in address based multicast. This requires data frames in multicast streams to carry sequence numbers to trigger DACK frames. The backoff and retransmission mechanisms can be similar. Despite possible gains in robustness, complexities in node/group address discovery and maintenance will remain.

High Robustness vs Complete Reliability. V-MAC does not provide 100% reliability. Many applications do not require it (e.g., infotainment ones such as video streaming). For those that do, V-MAC will be integrated with LP, which can detect losses that DACK may miss and re-request to ensure 100% reliability. We are currently building such an LP layer.

Small or Time-sensitive Packets. DACK aggregates and delays feedback, trading latency for efficiency. Depending on application needs (e.g., IoT control systems require low-latency, while sensor data high robustness), this may be unsuitable for smaller than one MTU, or time sensitive packets. We tried a *slow start* where a packet's burst size starts from 1, then doubles at each subsequent burst, until finally reaching some default value. This enables small packets to also trigger DACKs when needed. We tested this method and found that it can retain high robustness ($<5\%$) for data packets of 1–4 frames' size.

Security. V-MAC is intended to work with an ICN layer, whose philosophy on security is to secure the data rather than the communication link/pipe: each data item carries a public key signature for anyone to verify authenticity. Because of this, standard WiFi security (grouping/association/authentication) designed to distinguish nodes based on MAC addresses is no longer applicable in data-centric stacks: ICN offers such functions, but based on data and possibly user identities. Thus the current V-MAC prototype focuses on name based filtering and robust multicast, and leaves security largely to the ICN layer. Nevertheless, we will study MAC level attacks (e.g., DoS by injecting fake Data frames) and investigate in-depth whether any link level security is still necessary.

B. Future Work

We describe a few immediate next steps that our *V-MAC* efforts are heading towards:

Other Physical Layers. We have ported V-MAC on top of 802.11 a/b/g/n/ac radios with minor implementation changes and the same conceptual design. Similarly, we expect with slight adaptations for V-MAC to run on DSRC 802.11p, which is largely a variant of 802.11a [39].

Large Scale Evaluation. Due to limits on our hardware resources and man power, we evaluated up to 15 receivers and 10 producers in experiments. For a one hop neighborhood, this is already a reasonably dense setting. We plan to further stress V-MAC with more nodes to see whether and where it may start to break down. Once integration with ICN is completed, we will conduct multi-hop evaluation, using a combination of experiments and simulations.

Software-defined Radios. To offer a low-cost yet robust research asset easily adoptable in large size testbeds, we use commodity hardware in the current prototype, and have to deal with fixed, undesirable mechanisms in WiFi radios (e.g., group, station and beacons). We plan to explore SDR, a much more expensive platform that offers a clean implementation free of unintentional consequences (e.g., instability) and difficulties (e.g., hacking firmware/registers without source code) dealing with rigid PHY in commodity radios. We are currently investigating OpenWiFi [40], an open source SDR project, as a start to study the ideas detailed below, to achieve the full potential of a data-centric stack.

Frame Rate Adaptation. In experiments, we use a fixed 54Mbps rate which stresses the system sufficiently while being practically useful in real applications (e.g., delivering a 720P HD video takes 8Mbps). V-MAC needs to adapt sender transmission rate based on receivers' capabilities. Frame rate adaptation algorithms in WiFi are based on explicit identities and states (e.g., "station" data structures in MAC/PHY) of receivers. However, V-MAC adopts a pub/sub abstraction where senders have no notion and do not even (need to) know who are receivers. Also the existence of multiple receivers makes the rate adaptation much more difficult: a rate best for one receiver may not be best for another.

We plan to investigate an approach based on feedback. Interest and DACK frames may be augmented with receiver states (e.g., reception quality, supported rates). A sender can gather such information to gain knowledge about receivers, thus making decisions to adjust the frame rate. The challenge is exactly what form of augmented states and adaptation algorithms can achieve the right balance between complexity, performance, and robustness. Mobility may impact the reception quality, thus the algorithms must be agile to react to changes in short time.

Producer Selection. Multiple producers that all possess the same requested data may respond to one Interest. Due to differences in relative distances, mobility, and interference, some producers' transmissions may generate better receptions at a receiver than others. V-MAC can exploit this to select the best producer per-receiver. This can solve issues of a few "straggler" receivers with bad reception or low data rate: other producers or receivers that have obtained the data can send to them, thus they do not slow down other receivers.

To deal with asymmetry of wireless links, receivers can provide "hints" so producers know who generates better reception. We have tested a simple idea that includes *Producer ID* in DACKs and found that this enables producers to know whose transmissions are received by which receiver, thus making better decisions of who should transmit. A complete solution requires consideration of many other factors (e.g., RSSI, reception rate, direction), and the possibility of dynamic selection of different producers at different times based on ever-changing reception qualities.

Link Layer Protocol (LP) and Applications. One important step in a full data-centric stack is to integrate ICN network layers and V-MAC. This enables multi-hop communication in edge environments. An adaptation layer (i.e., Link Layer Protocol) is needed in between to support ICN layer functions using V-MAC. There are a few challenges in building the LP layer: *i)* integrating two different designs of ICN and V-MAC to a coherent implementation (e.g., NDN leverages NACK packets while V-MAC does not); *ii)* fragmenting large ICN packets into multiple MTU size frames, and reassembling frames back into ICN packets; *iii)* supporting routing state maintenance. Currently, NDN can maintain FIB states using UDP in mobile environment. We expect V-MAC can offer a similar datagram transport like UDP, and more robustly than that of WiFi; *iv)* cross layer coordination between ICN and V-MAC to achieve the best performance (e.g., optimize for latency, loss, etc.). We are currently undertaking such efforts and expect a full integration of NDN and V-MAC in a few months.

Once integrated, we will study a series of questions in building applications: how often Interests should be sent to discover nearby data (or lack of it due to producers being out of range), how long a producer should retain data waiting for Interests, and how applications can send cross-layer hints.

VII. RELATED WORK

Publish/Subscribe. Pub/sub is an asynchronous communication abstraction used widely in social media (e.g., Twitter, Facebook) and business integration [16], [17], [18], [19], [41], [42], [43], [44]. It decouples sender/receivers: receivers (consumers) specify desired data in subscriptions using hierarchical topic names, multi-dimensional attribute predicates or queries like XPath. Senders' (producers) publications are matched against subscriptions at intermediaries (i.e., brokers) and delivered accordingly. Most pub/sub systems are built at application layer, and utilize point-to-point transport/networks (e.g., TCP/IP). V-MAC adopts this abstraction and is the first to offer it at the MAC layer.

Information Centric Networking. Among ICN proposals, NDN [45], [46], [47] is probably the most prominent. It takes a similar pub/sub abstraction and has been tested for video transmission [48] and vehicular networking [8], [9], [10] on top of and thus limited by WiFi ad hoc broadcast (e.g., lowest base rate, high loss). V-MAC pushes the pub/sub abstraction down to the MAC. It eliminates complexities in address-based WiFi stacks (e.g., unnecessary queues and en/decapsulations, beacons, group formation). V-MAC completes a full data-centric stack, which will benefit many edge applications (e.g., vehicles, drones, IoT and mobile [49], [50], [51], [52], [53], [8], [54], [55], [56]).

WiFi Multicast. Three common techniques have been explored for robust, low loss, address-based WiFi multicast: automatic retransmission request (ARQ) to trigger retransmission of missed frames [13], [57], [58], [59], [60], [61]; forward error correction (FEC) [62] to reconstruct the content despite losses [63], [64], [65], [66], [67], [68], [69], [70]; and network coding [71] to reduce the number of transmissions [72].

Some ARQ based designs adopt collective feedback [61], [73], [74], [14], [59], [15]). V-MAC utilizes a similar principle, but applied under a completely different pub/sub abstraction for highly dynamic edge environments. Those designs do not modify and rely on WiFi's point-to-point abstraction, thus they cannot eliminate the baggage of address/group discovery and formation. FEC and network coding are orthogonal techniques that V-MAC can also adopt.

Vehicular Networking. Dedicated Short-Range Communication (DSRC) [3], [4], [5] uses 802.11p (a variant of 802.11a) as MAC/PHY. Despite decades-long standardization, significant safety and throughput weaknesses still exist [75], [76], [77], [78], [79]. V-MAC provides a new pub/sub abstraction and robust, high rate multicast, features needed for vehicles but lacking in 802.11p.

VIII. CONCLUSION

We propose V-MAC, a novel data-centric radio providing a pub/sub abstraction to replace the point-to-point abstraction in existing wireless communication. It eliminates complexities such as beacons and groups in address-based radios, and offers natural one-to-many communication sorely needed in many edge computing applications. Compared to existing WiFi, the commodity hardware prototype can match millions of data names at O(1) complexity (10μs) and reduce losses from 50–90% to 1–3% or less across multiple receivers, for both stationary and mobile scenarios.

V-MAC on 802.11n radios is quite mature. We have ported V-MAC to 802.11ac (including NVIDIA Jetson TX2 GPU platform and RealTek dongles on Raspberry Pi 4). Preliminary experiments using 2 receivers and 1 sender show loss rate $<1\%$, and up to 900Mbps data rates. The current V-MAC prototype can run on different edge platforms (e.g., Android, embedded systems, and Xilinx FPGA PetaLinux) and three major WiFi chips (e.g., Qualcomm Atheros, RealTek, MediaTek), offering wide opportunities for adoption. We will further improve the code stability, validate the scalability at more receivers and higher data rates, then release it to the community. The change to a pub/sub abstraction opens up a slew of new research opportunities, which we will pursue to create a novel data centric wireless communication paradigm.

IX. ACKNOWLEDGMENT

We thank Aashish Ravindran, Christopher Moore, and Tejas Menon for their help with the experiments, NDN community for their support, and anonymous reviewers for their invaluable feedback. This work is supported in part by NSF grants 1513719, 1730291, and 1652276.

REFERENCES

[1] IEEE. IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control(MAC) and Physical Layer(PHY) Specifications. IEEE Standard 802.11-2012, March 2012.

[2] Matthew S Gast. *802.11 ac: a survival guide: Wi-Fi at gigabit and beyond.* " O'Reilly Media, Inc.", 2013.

[3] John B Kenney. Dedicated short-range communications (DSRC) standards in the united states. *Proceedings of the IEEE*, 99(7):1162–1182, 2011.

[4] Xinzhou Wu, Sundar Subramanian, Ratul Guha, Robert G White, Junyi Li, Kevin W Lu, Anthony Bucceri, and Tao Zhang. Vehicular communications using dsrc: challenges, enhancements, and evolution. *IEEE Journal on Selected Areas in Communications*, 31(9):399–408, 2013.

[5] Daniel Jiang and Luca Delgrossi. Ieee 802.11 p: Towards an international standard for wireless access in vehicular environments. In *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, pages 2036–2040. IEEE, 2008.

[6] Shanzhi Chen, Jinling Hu, Yan Shi, Ying Peng, Jiayi Fang, Rui Zhao, and Li Zhao. Vehicle-to-everything (v2x) services supported by lte-based systems and 5g. *IEEE Communications Standards Magazine*, 1(2):70–76, 2017.

[7] Rafael Molina-Masegosa and Javier Gozalvez. Lte-v for sidelink 5g v2x vehicular communications: A new 5g technology for short-range vehicle-to-everything communications. *IEEE Vehicular Technology Magazine*, 12(4):30–39, 2017.

[8] Giulio Grassi, Davide Pesavento, Giovanni Pau, Rama Vuyyuru, Ryuji Wakikawa, and Lixia Zhang. Vanet via named data networking. In *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pages 410–415. IEEE, 2014.

[9] Zhiwei Yan, Sherali Zeadally, and Yong-Jin Park. A novel vehicular information network architecture based on named data networking (ndn). *IEEE internet of things journal*, 1(6):525–532, 2014.

[10] Giulio Grassi, Davide Pesavento, Giovanni Pau, Lixia Zhang, and Serge Fdida. Navigo: Interest forwarding by geolocations in vehicular named data networking. In *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–10. IEEE, 2015.

[11] Varun Gupta, Yigal Bejerano, Craig Gutterman, Jaime Ferragut, Katherine Guo, Thyaga Nandagopal, and Gil Zussman. Light-weight feedback mechanism for wifi multicast to very large groups—experimental evaluation. *IEEE/ACM Transactions on Networking*, 24(6):3826–3840, 2016.

[12] Dimitrios Koutsonikolas, Y Charlie Hu, and Chih-Chun Wang. Pacifier: High-throughput, reliable multicast without "crying babies" in wireless mesh networks. *IEEE/ACM Transactions on Networking*, 20(5):1375–1388, 2011.

[13] C Perkins and M McBride. Multicast wifi problem statement. *Working Draft, IETF Internet-Draft*, 2015.

[14] José Villalón, Pedro Cuenca, Luis Orozco-Barbosa, Yongho Seok, and Thierry Turletti. Cross-layer architecture for adaptive video multicast streaming over multirate wireless lans. *IEEE Journal on Selected Areas in Communications*, 25(4), 2007.

[15] ALAY Ozgu, Thanasis Korakis, Shivendra S Panwar, and Yao Wang. Dynamic rate and fec adaptation for video multicast in multi-rate wireless networks, October 6 2011. US Patent App. 12/753,544.

[16] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, and Antony Rowstron. Scribe: A large-scale and decentralized publish-subscribe infrastructure. In *Third International Workshop on Networked Group Communication (NGC2001), London, UK*. Citeseer, 2001.

[17] Tova Milo, Tal Zur, and Elad Verbin. Boosting topic-based publish-subscribe systems with dynamic clustering. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 749–760, 2007.

[18] Peter R Pietzuch and Jean M Bacon. Hermes: A distributed event-based middleware architecture. In *Proceedings 22nd International Conference on Distributed Computing Systems Workshops*, pages 611–618. IEEE, 2002.

[19] Bill Segall, David Arnold, Julian Boot, Michael Henderson, and Ted Phelps. Content based routing with elvin4. In *Proceedings AUUG2k*, 2000.

[20] *ath9k_htc*, 2017 (accessed February 8, 2020).

[21] Linux Wireless. Linux Wireless: About ath10k. https://wireless.wiki.kernel.org/en/users/drivers/ath10k, 2019.

[22] Linux Wireless. Linux Wireless: Realtek. https://wireless.wiki.kernel.org/en/users/drivers/rtl819x, 2019.

[23] Linux Wireless. Linux Wireless: Mediatek. https://wireless.wiki.kernel.org/en/users/drivers/mediatek, 2019.

[24] Eldad Perahia. Ieee 802.11 n development: History, process, and technology. *IEEE Communications Magazine*, 46(7):48–55, 2008.

[25] IEEE 802.11 Working Group et al. Part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications: Higher-speed physical layer in the 5ghz band. *IEEE Std 802.11*, 1999.

[26] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang, et al. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.

[27] Ming Li, Fan Ye, Minkyong Kim, Han Chen, and Hui Lei. A scalable and elastic publish/subscribe service. In *Parallel & Distributed Processing Symposium (IPDPS), 2011 IEEE International*, pages 1254–1265. IEEE, 2011.

[28] Hongjun Yu, Rong Chen, and Zhenjun Du. Message oriented middleware support for mobile ad hoc data sharing. In *2010 IEEE 34th Annual Computer Software and Applications Conference Workshops*, pages 305–310. IEEE, 2010.

[29] Yi Wang, Keqiang He, Huichen Dai, Wei Meng, Junchen Jiang, Bin Liu, and Yan Chen. Scalable name lookup in ndn using effective name component encoding. In *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on*, pages 688–697. IEEE, 2012.

[30] Md Faizul Bari, Shihabur Rahman Chowdhury, Reaz Ahmed, Raouf Boutaba, and Bertrand Mathieu. A survey of naming and routing in information-centric networks. *IEEE Communications Magazine*, 50(12), 2012.

[31] Lucas Wang, Ryuji Wakikawa, Romain Kuntz, Rama Vuyyuru, and Lixia Zhang. Data naming in vehicle-to-vehicle communications. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 328–333. IEEE, 2012.

[32] Xiangshen Yu, Rodolfo WL Coutinho, Azzedine Boukerche, and Antonio AF Loureirol. A distance-based interest forwarding protocol for vehicular information-centric networks. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–5. IEEE, 2017.

[33] G. Grassi, D. Pesavento, L. Wang, G. Pau, R. Vuyyuru, R. Wakikawa, and L. Zhang. Vehicular inter-networking via named data. In *arXiv:1310.5980*, October 2013.

[34] Dionysios Skordoulis, Qiang Ni, Hsiao-Hwa Chen, Adrian P Stephens, Changwen Liu, and Abbas Jamalipour. Ieee 802.11 n mac frame aggregation mechanisms for next-generation high-throughput wlans. *IEEE Wireless Communications*, 15(1):40–47, 2008.

[35] Vishal Gupta and Mukesh Kumar Rohil. Information embedding in ieee 802.11 beacon frame. In *National Conference on Communication Technologies & its impact on Next Generation Computing CTNGC*. sn, 2012.

[36] Eustathia Ziouva and Theodore Antonakopoulos. Csma/ca performance under high traffic conditions: throughput and delay analysis. *Computer communications*, 25(3):313–321, 2002.

[37] M Vipin and S Srikanth. Analysis of open source drivers for ieee 802.11 wlans. In *2010 International Conference on Wireless Communication and Sensor Computing (ICWCSC)*, pages 1–5. IEEE, 2010.

[38] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. Tool release: Gathering 802.11 n traces with channel state information. *ACM SIGCOMM Computer Communication Review*, 41(1):53–53, 2011.

[39] Guido R Hiertz, Dee Denteneer, Lothar Stibor, Yunpeng Zang, Xavier Pérez Costa, and Bernhard Walke. The ieee 802.11 universe. *IEEE Communications Magazine*, 48(1):62–70, 2010.

[40] Jiao Xianjun, Liu Wei, and Mehari Michael. open-source ieee802.11/wi-fi baseband chip/fpga design, 2019.

[41] Abhishek Gupta, Ozgur D Sahin, Divyakant Agrawal, and Amr El Abbadi. Meghdoot: content-based publish/subscribe over p2p networks. In *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*, pages 254–273. Springer, 2004.

[42] David Tam, Reza Azimi, and Hans-Arno Jacobsen. Building content-based publish/subscribe systems with distributed hash tables. In *International Workshop on Databases, Information Systems, and Peer-to-Peer Computing*, pages 138–152. Springer, 2003.

[43] Hans-Arno Jacobsen, Alex Cheung, Guoli Li, Balasubramaneyam Maniymaran, Vinod Muthusamy, and Reza Sherafat Kazemzadeh. The padres publish/subscribe system. In *Principles and Applications of Distributed Event-Based Systems*, pages 164–205. IGI Global, 2010.

[44] E Fidler, HA Jacobsen, G Li, and S Mankovski. Publish/subscribe system. *Feature Interactions in Telecommunications and Software Systems VIII*, page 12, 2005.

[45] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12. ACM, 2009.

[46] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D Thornton, Diana K Smetters, Beichuan Zhang, Gene Tsudik, Dan Massey, Christos Papadopoulos, et al. Named data networking (NDN) project. *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 2010.

[47] Cheng Yi, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. Adaptive forwarding in named data networking. *ACM SIGCOMM computer communication review*, 42(3):62–67, 2012.

[48] Menghan Li, Dan Pei, Xiaoping Zhang, Beichuan Zhang, and Ke Xu. NDN live video broadcasting over wireless LAN. In *Computer Communication and Networks (ICCCN), 2015 24th International Conference on*, pages 1–7. IEEE, 2015.

[49] Swarun Kumar, Lixin Shi, Nabeel Ahmed, Stephanie Gil, Dina Katabi, and Daniela Rus. Carspeak: a content-centric network for autonomous driving. In *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 259–270. ACM, 2012.

[50] Marica Amadeo, Claudia Campolo, Antonella Molinaro, and Giuseppe Ruggeri. Iot data processing at the edge with named data networking. In *European Wireless 2018; 24th European Wireless Conference*, pages 1–6. VDE, 2018.

[51] Stefano Rosati, Karol Kružlecki, Louis Traynard, and Bixio Rimoldi Mobile. Speed-aware routing for uav ad-hoc networks. In *2013 IEEE Globecom Workshops (GC Wkshps)*, pages 1367–1373. IEEE, 2013.

[52] Mohammad Mozaffari, Walid Saad, Mehdi Bennis, Young-Han Nam, and Mérouane Debbah. A tutorial on uavs for wireless networks: Applications, challenges, and open problems. *IEEE communications surveys & tutorials*, 21(3):2334–2360, 2019.

[53] Ezedin Barka, Chaker Abdelaziz Kerrache, Rasheed Hussain, Nasreddine Lagraa, Abderrahmane Lakas, and Safdar Hussain Bouk. A trusted lightweight communication strategy for flying named data networking. *Sensors*, 18(8):2683, 2018.

[54] Tarak Gandhi and Mohan M Trivedi. Motion based vehicle surround analysis using an omni-directional camera. In *IEEE Intelligent Vehicles Symposium, 2004*, pages 560–565. IEEE, 2004.

[55] Hang Qiu, Fawad Ahmad, Ramesh Govindan, Marco Gruteser, Fan Bai, and Gorkem Kar. Augmented vehicular reality: Enabling extended vision for future vehicles. In *Proceedings of the 18th International Workshop on Mobile Computing Systems and Applications*, pages 67–72. ACM, 2017.

[56] Yurong Jiang, Hang Qiu, Matthew McCartney, Gaurav Sukhatme, Marco Gruteser, Fan Bai, Donald Grimm, and Ramesh Govindan. Carloc: Precise positioning of automobiles. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 253–265. ACM, 2015.

[57] Varun Srinivas and Lu Ruan. An efficient reliable multicast protocol for 802.11-based wireless lans. In *World of Wireless, Mobile and Multimedia Networks & Workshops, 2009. WoWMoM 2009. IEEE International Symposium on a*, pages 1–6. IEEE, 2009.

[58] Xiaoli Wang, Lan Wang, Yingjie Wang, Yongsheng Zhang, and Akira Yamada. Supporting mac layer multicast in ieee 802.11 n: Issues and solutions. In *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE*, pages 1–6. IEEE, 2009.

[59] Zhengyong Feng, Guangjun Wen, Chunlei Yin, and Hongsheng Liu. Video stream groupcast optimization in wlan. In *Internet Technology and Applications, 2010 International Conference on*, pages 1–4. IEEE, 2010.

[60] Xiaoli Wang, Lan Wang, Yingjie Wang, and Daqing Gu. Reliable multicast mechanism in wlan with extended implicit mac acknowledgment. In *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, pages 2695–2699. IEEE, 2008.

[61] Yigal Bejerano, Jaime Ferragut, Katherine Guo, Varun Gupta, Craig Gutterman, Thyaga Nandagopal, and Gil Zussman. Scalable wifi multicast services for very large groups. In *Network Protocols (ICNP), 2013 21st IEEE International Conference on*, pages 1–12. IEEE, 2013.

[62] Joachim Hagenauer and Erich Lutz. Forward error correction coding for fading compensation in mobile satellite channels. *IEEE Journal on Selected Areas in Communications*, 5(2):215–225, 1987.

[63] Munhwan Choi, Maria Samokhina, Kirill Moklyuk, Sunghyun Choi, Jun Heo, and Seong-Jun Oh. Vpal: video packet adaptation layer for reliable

[69] Ranveer Chandra, Sandeep Karanth, Thomas Moscibroda, Vishnu Navda, Jitendra Padhye, Ramachandran Ramjee, and Lenin Ravindranath. Dircast: A practical and efficient wi-fi multicast system. In *Network Protocols,*

video multicast over ieee 802.11 n wlan. *Computer Communications*, 33(18):2271–2281, 2010.

[64] Yeonchul Shin, Munhwan Choi, Jonghoe Koo, and Sunghyun Choi. Video multicast over wlans: Power saving and reliability perspectives. *IEEE Network*, 27(2):40–46, 2013.

[65] Hsin-Ta Chiao, Shih-Ying Chang, Kuan-Ming Li, Yi-Ting Kuo, and Ming-Chien Tseng. Wifi multicast streaming using al-fec inside the trains of high-speed rails. In *Broadband Multimedia Systems and Broadcasting (BMSB), 2012 IEEE International Symposium on*, pages 1–6. IEEE, 2012.

[66] Shiuan-Tung Chen, Hsin-Ta Chiao, Shih-Ying Chang, Hung-Min Sun, and Pei-Shan Zeng. An hd streaming system for wifi multicast channels based on application-layer fec. In *Consumer Electronics (ISCE), 2013 IEEE 17th International Symposium on*, pages 85–86. IEEE, 2013.

[67] Munhwan Choi, Weiping Sun, Jonghoe Koo, Sunghyun Choi, and Kang G Shin. Reliable video multicast over wi-fi networks with coordinated multiple aps. In *INFOCOM, 2014 Proceedings IEEE*, pages 424–432. IEEE, 2014.

[68] Tijs Van Dam and Koen Langendoen. An adaptive energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 171–180. ACM, 2003.

2009. ICNP 2009. 17th IEEE International Conference on, pages 161–170. IEEE, 2009.

[70] Philip K. McKinley, Chiping Tang, and Arun P. Mani. A study of adaptive forward error correction for wireless collaborative computing. *IEEE Transactions on Parallel and Distributed Systems*, 13(9):936–947, 2002.

[71] M Celebiler and G Stette. On increasing the down-link capacity of a regenerative satellite repeater in point-to-point communications. *Proceedings of the IEEE*, 66(1):98–100, 1978.

[72] Ayesha Naeem, Mubashir Husain Rehmani, Yasir Saleem, Imran Rashid, and Noel Crespi. Network coding in cognitive radio networks: A comprehensive survey. *IEEE Commun. Surveys Tuts.*, 19(3):1945–1973, 2017.

[73] Sayandeep Sen, Neel Kamal Madabhushi, and Suman Banerjee. Scalable wifi media delivery through adaptive broadcasts. In *NSDI*, pages 191–204, 2010.

[74] Nakjung Choi, Yongho Seok, Taekyoung Kwon, and Yanghee Choi. Leader-based multicast service in ieee 802.11 v networks. In *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE*, pages 1–5. IEEE, 2010.

[75] Francesca Martelli, M Elena Renda, Giovanni Resta, and Paolo Santi. A measurement-based study of beaconing performance in ieee 802.11 p vehicular networks. In *Infocom*, pages 1503–1511, 2012.

[76] Jijun Yin, Tamer ElBatt, Gavin Yeung, Bo Ryu, Stephen Habermas, Hariharan Krishnan, and Timothy Talty. Performance evaluation of safety applications over dsrc vehicular ad hoc networks. In *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 1–9. ACM, 2004.

[77] Zhigang Xu, Xiaochi Li, Xiangmo Zhao, Michael H Zhang, and Zhongren Wang. Dsrc versus 4g-lte for connected vehicle applications: a study on field experiments of vehicular communication performance. *Journal of Advanced Transportation*, 2017, 2017.

[78] Seungbae Lee and Alvin Lim. An empirical study on ad hoc performance of dsrc and wi-fi vehicular communications. *International Journal of Distributed Sensor Networks*, 9(11):482695, 2013.

[79] Lin Cheng, Benjamin E Henty, Daniel D Stancil, Fan Bai, and Priyantha Mudalige. Mobile vehicle-to-vehicle narrow-band channel measurement and characterization of the 5.9 ghz dedicated short range communication (dsrc) frequency band. *IEEE Journal on Selected Areas in Communications*, 25(8), 2007.