Emulation-Based Performance Evaluation of Routing Protocols for Uaanets

Jean-Aimé Maxa, Gilles Roudiere, Nicolas Larrieu

Emulation-based performance evaluation of routing protocols for UAANETs

Jean-Aimé Maxa[1 2], Gilles Roudiere[1], Nicolas Larrieu[1]

[1]ENAC, TELECOM/Resco, F-31055 Toulouse, France
[2]Univ de Toulouse, F-31400 Toulouse, France
maxa@recherche.enac.fr
{gilles.roudiere,nicolas.larrieu}@enac.fr

Abstract. UAV Ad hoc NETwork (UAANET) is a subset of the well-known Mobile Ad-hoc NETworks (MANETs). It consists of forming an ad hoc network with multiple small Unmanned Aerial Vehicles (UAVs) and the Ground Control Station (GCS). Similar to MANETs, the UAANET communication architecture is infrastructure-less and self-configuring network of several nodes forwarding data packets. However, it also has some specific features that brings challenges on network connectivity. Consequently, an adapted routing protocol is needed to exchange data packets within UAANETs. In this paper, we introduce a new hybrid experimental system that can evaluate different types of adhoc routing protocols under a realistic UAANET scenario. It is based on virtual machines and the Virtualmesh [1] framework to emulate physical aspects. We evaluated AODV, DSR and OLSR efficiency in a realistic scenario with three UAVs scanning an area. Our results show that AODV outperformed OLSR and DSR.

Keywords: UAV Ad hoc Network, Ad hoc Routing Protocol, VirtualMesh, Emulation, Simulation

1 Introduction

An Unmanned Aerial Vehicle (UAV) is a pilotless aerial vehicle that can be either controlled by an on-board computer or remotely piloted by a distant operator. As almost all of them are equipped with a wireless communication system, UAVs can be used to create a self-organizing and multi-hop network called UAV Ad-hoc NETwork (UAANET). This network architecture shares some common features with the well-known Mobile Ad-hoc NETwork (MANET), but also face distinctive issues. For instance, the relatively low number of nodes in UAANETs and their fast mobility leads to new challenges towards connectivity. Another challenge is also the QoS constraints of data traffic. UAVs missions often involve real-time data transmissions (video, real-time measurement), therefore, a certain QoS must be ensured to exploit the received information.

Furthermore, a routing protocol is needed to route data traffics through the multiple UAVs within UAANETs. Due to the several features of UAANETs (

detailed section II), an adaptive routing protocol is needed to properly exchange data traffics within UAANETs. To create such protocol, a possibility is either to create a routing protocol from scratch or to use an existing routing protocol proposed in the literature for MANETs and adapt them based on UAANETs requirements. The second case has particularly drawn our attention since it takes less effort and also because we would like to further concentrate on security aspect in our future work.

With this intention, a performance evaluation of the several MANETs routing protocols under UAANET scenarios must be carried out. In fact, there are only few studies that have been conducted to measure routing protocols for UAANETs [2] [3]. These studies are simulation-based and as such do not consider the linux kernel networking stack and a realistic mobility model. Indeed, simulation-based implementations might hide several important parameters (e.g, protocol implementations, background traffic, real time execution) due to the lack of OS-based implementations. These limitations could induce significant differences between simulations and real test-bed results.

Our emulation-based implementation allows to combine the low cost of a simulation with the accuracy of a real protocol stack. The traces used to generate UAVs mobility patterns were extracted from real traces so that physical related factors could be as realistic as possible. This article will present this testbed which could be used by the network community to carry out experimental evaluations of different types of Ad-Hoc Networks (e.g, VANET). It is also possible to evaluate different type of network protocols.

In the first part of this paper, we describe the different features of a UAANET architecture. The second part is dedicated to our experimental test protocol and results. Finally, the last part details the different software components of the testbed. Please note the entire framework is freely available and can be used by the research community.

2 UAANETs characteristics

Similar to MANETs, the UAANETs architecture is an infrastructure-less network which uses multiple UAVs to forward data packets. It shares some similarities with standard mobile ad hoc networks such as self-organized pattern, self-managed information and communication between nodes without a centralized authority. However, UAANETs also have specific features that can be listed as [4].

- **Number of nodes:** when an UAV deployed in a given mission has a relatively high speed, it can be sufficient to cover a restricted mission area. Then, the need for a large number of UAVs is not justified in such a case. Usually, UAV mission involves an average of 3 to 4 UAVs [5] [6] [7] [8] [9]. This has the advantage of reducing the impact of scalability issues which affects several MANET architectures.

- **Topology variability:** as the medium is shared between multiple agents, collisions can occur, and loss rate can raise on high load. Moreover, distance between UAVs can make them loose connectivity between each other. These issues impact how critical information (i.e. control packets) has to be managed in the network.
- **Mobility:** UAV mobility patterns are a lot different from any other vehicle. An UAV movement is above all 3D based. This brings a whole set of challenges on the physical layer, the antenna behaviors and the security aspect(e.g, misbehavior detection). Furthermore, UAVs are used for specific missions that can include several different mobility patterns like area scanning, reaching a way-point, staying at a position or even patrol around a circuit. Accordingly, an innovative approach has been proposed in [10] where the author provided a mobility pattern for UAVs based on real traces. The diversity of UAV moves leads to very varied connectivity patterns.
- **Energy:** energy and computing power are limited in UAANETs but not as much as in sensor networks. This issue is usually not considered as a determining factor in UAANETs. Indeed, the energy needed to move the UAV is much greater than the energy needed to compute data.
- **Propagation model:** in MANETs networks, nodes usually move close to the ground (like in VANETs or sensors networks). UAANETs are rather different as it is composed of flying node moving in large free space. Consequently, the free-space path loss model is often used to model the physical layer. Nevertheless it is advisable to take into account factors like large obstacles, ground reflections or weather conditions which can affect connectivity between UAVs.
- **QoS constraints:** UAVs mission usually need real-time services, from aerial video and photography to real-time monitoring. Consequently, delay constraints are stronger in UAANETs. Also, some control/command traffic should be guaranteed as high-priority traffic to avoid control losses.

3 Routing protocols for UAANETs

Several dynamic routing mechanisms are available for UAANETs. As exposed in [11], topology-based mechanisms, such as proactive, reactive or hybrid routing are the basis of numerous protocols. Nonetheless geographical routing, as surveyed in [12], could also be efficient in specific contexts.

Proactive routing: a proactive routing mechanism tries to establish a route from one node to another before it is needed. Each node in the ad-hoc network sends control messages at a fixed rate. They usually contain the node routing table and relayed information from other nodes. Step by step, routing information is relayed from the destination node to the source node, and a route can be established. As proactive routing protocols we can cite OLSR [13] and its extensions (like DOLSR [14], M-OLSR [15] or CE-OLSR [16]), DSDV [17] and B.A.T.M.A.N [18].

Reactive routing: in contrast with proactive protocols, reactive protocols establish a route when it is necessary. When a node wants to send a packet to a destination node, it first sends a route request packet which will be flooded through the whole network. When a node receives the route request packet, it adds its address to the list of the nodes that the packet went through. When one (or several) route request packets reach the destination node, a route response packet is sent back to the source using the shortest route discovered. The source uses this route to reach the destination. Several reactive protocols have been proposed in the literature : AODV [19] and its extensions (like AODVSEC [20], Time-slotted on-demand routing [37], or MAODV [21]) or DSR [22].

Geographical routing: geographical routing uses the nodes positions to find the best route from a source to a destination. Usually, it uses two distinct mechanisms: greedy forwarding and a backup mechanism in case where the former failed. The greedy forwarding consist of selecting as a next hop the closest node from the source node position. Alternatively, in case where no node within range closer to the destination is found, a backup mechanism is automatically launched. We can cite as an example ”Face Routing” used by GFG [23], a mechanism which consists in creating a planar graph of the network connections, then using the *right hand rule* [24] to reach the destination. Several geographical routing protocols have been proposed: GPSR [25], GPMOR [26], USMP [27] or MPGR [28].

Hybrid routing: hybrid routing is a generic term referring to a combination of two routing mechanisms. As an example, we could cite RGR [29], which is a reactive protocol using greedy forwarding as a backup mechanism.

3.1 Previous comparison studies

In [2], AODV, GPSR and OLSR has been compared with simulated systems. This study showed that with a large number of nodes (more than 50), GPSR outperformed AODV an OLSR in terms of delay and packet delivery ratio. AODV has slightly better performances than OLSR in terms of packet delivery ratio, but can create higher delays at a low workload. In [30], protocols are compared with a set of 19 nodes. In this situation, AODV performs better than other topology-based protocols, having a slightly better overall throughput and a lower end-to-end delay. However, it is important to underline that these evaluations are based on simulations and do not consider real systems related issues. Also, they usually consider a large number of nodes in an unrealistic mission scenario and based on inadequate mobility patterns. These limitations led us to create our own experimental test protocol, which is detailed in the following part.

Simulation	Real Word	Emulation
+ / − Variable experiment duration	= Real experiment duration	= Real experiment duration
+ Scalable	− Not easily scalable	+ Scalable
+ Mobility easy to set up	− Mobility difficult to set up	+ Mobility easy to set up
+ Reproducibility	− No reproducibility	= Reproducibility for physical factors
+ Mastered environment	− Undesirable interferences	+ Mastered environment
− Simulation specific implementation	+ Operating system implementation	+ Operating system implementation
− Whole system approximation	+ Real system	= Physical environment approximation

Table 1: Comparison of different kind of testbeds [1]

4 Emulation-based performance evaluation of routing protocols for UAANET

4.1 Using emulation for protocols evaluation

Some studies have already approached the comparison of routing protocols for UAANETs but they are usually based either on simulation or real flights experiments. Table 1 shows a comparison of each solution. Our proposal fits between these two solutions, it aims to evaluate protocol implementations compatible with a standard operating system without a need to perform a real environment test. We propose to use a real-time simulated physical environment in combination with virtual machines.

The main advantages of our hybrid experimental systems are:

- To perform a complete implementation of the different routing protocols that we want to analyse. Indeed, when a routing protocol is modelled inside a simulator, there is a need to simplify some parts of the behaviors and it is not obvious to predict that the obtained results will be consistant in real world. With the virtual machines implementation in this testbed, we can deal with the entire complexity of an Operating System such as Linux.
- To exchange information with a high number of nodes following a realistic flight plan. Indeed, it is expensive to gather enough UAVs, enough pilots and enough free space for the operation field. By using the *OMNeT++* [31] simulator, we can simulate a high number of nodes (each one being realistic thanks to the Linux virtual machine that we run). Furthermore, we have introduced real UAVs mobility pattern[1] into the OMNET simulation tool to be as close as possible to the UAANET outdoor experiments.

[1] provided by Delair Tech company, see http://www.delair-tech.com for more details about this company.

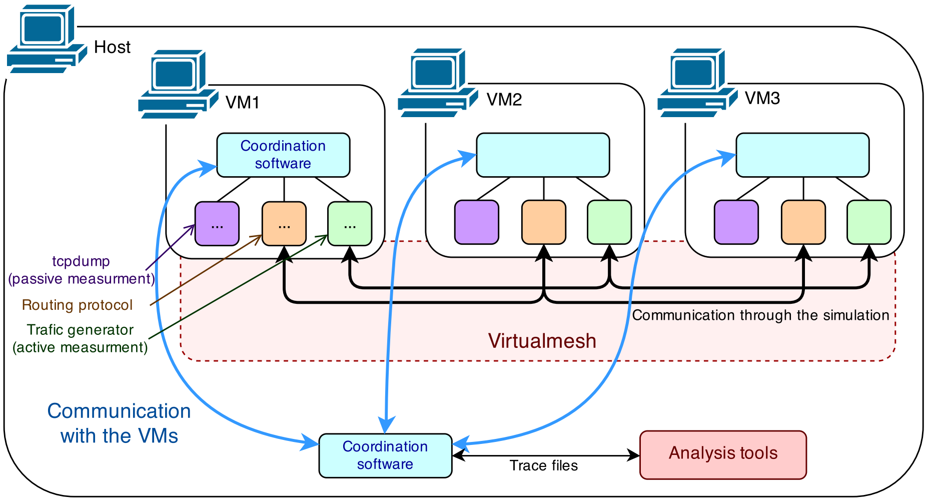

Host
VM1
VM2
VM3
Coordination software
...
...
...
tcpdump (passive measurment)
Routing protocol
Trafic generator (active measurment)
Communication through the simulation
Virtualmesh
Communication with the VMs
Coordination software
Trace files
Analysis tools

Fig. 1: Testbed implementation

4.2 Experimental test implementation

The system we used to evaluate protocols is divided in several parts. It includes a set of tools that can fit to several scenarios : An hypervisor to run the virtual machines, a measurement tools and a framework to allow virtual machines to communicate through a virtual wireless medium. An illustration of this system is on Figure 1

Virtualization: We chose to use *VirtualBox*[32] as a virtualisation tool because it is an easy-to-use and efficient hypervisor. The virtualized system is a 11.04 version Ubuntu, working with the 2.6.38 version of the Linux kernel. This version were chosen because it is the same than the one used on the development system. A higher linux kernel can be used for improvement purposes.

Traffic measure and analysis

Evaluated parameters: we decided to measure 3 parameters to evaluate routing protocols performances. First, the extra traffic generated by each protocol is evaluated. This overhead is caused by the control packets. This parameter impacts how much bandwidth can be allocated to applications. Secondly, we chose to evaluate the end-to-end delay for each protocol, even if with a low number of node this parameter becomes less interesting. Finally, the routing protocols abilities to find a new route after a route loss is evaluated. This is a really important matter in a high mobility context.

Active and Passive measurement: due to Virtual Machines (VM) implementations, we could use any software to generate real traffic. However, for measurement purposes, we created our own tool written in Python. This script is able to generate a realistic traffic (as described in 4.3) overloaded with a short header including an ID -incremented for each packet sent- and a timestamp. On the other hand, to get accurate data on the traffic going through the network during the test, we also used passive measurement tools. They aim at capturing traffic going through any interface without modifying the traffic. The tool used is the well-known *tcpdump*. Furthermore, *Wireshark* is used to analyse and extract several metrics (e.g, overhead) from the traces.

The Virtualmesh framework: the *Virtualmesh* framework has been proposed by [1]. It is a framework that interfaces a Linux-based system with an *OMNeT++* [31] simulation. *Omnet++* is a powerful network simulator which simulate several systems and normalized protocols. Using *Virtualmesh* could be summed up in these simple steps :

1. A virtual wireless interface is created on the Linux system we want to include in the simulation;
2. We launch the *OMNet++* simulation (which has to include some modules supplied by the framework);
3. The framework links the Linux virtual interface to the simulation. This uses UDP sockets, so we can use either real or emulated Linux systems.

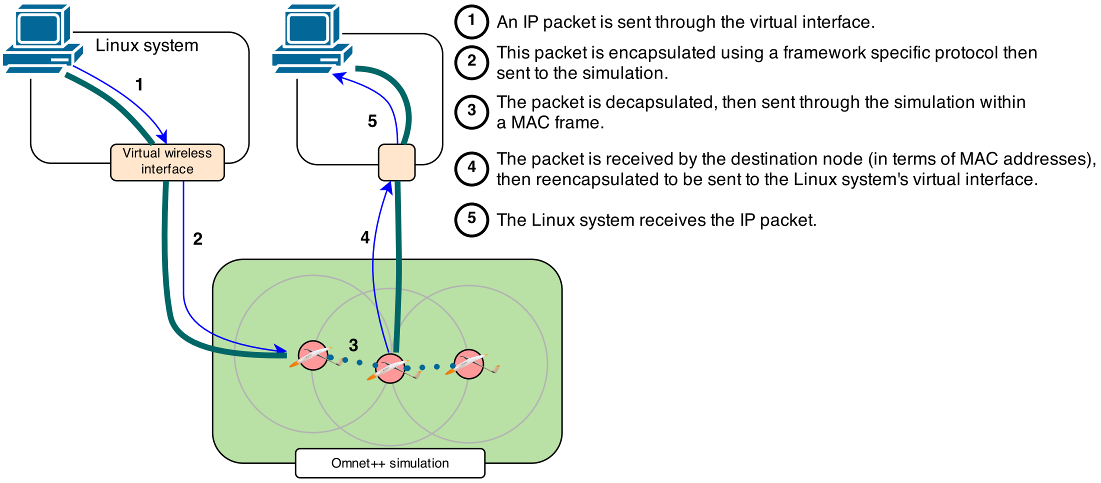
Once these steps are complete, a new node appears in the simulation. Any packet sent through the virtual interface is encapsulated in a UDP packet and sent to the simulation, which relays the packets depending on physical simulation parameters : nodes transmission range, signal attenuation, emission power, etc... Conversely, a packet received by an agent in the simulation is relayed to the virtual interface. An illustration of this system is exposed in Figure 2.

4.3 Test scenario

Realistic mission scenario: to evaluate our routing protocols, we chose to create our experimental test protocol according to what could be a real UAV swarm mission. The scenario selected to evaluate protocols performances has been proposed by the Delair-Tech company based on their expertise in this field. It consists of three drones scanning an area for video surveillance purposes. We suppose that an obstacle blocks the traffic between two UAVs and the ground station, so a third one is deployed to be responsible for relaying data packets. This scenario is illustrated in Figure 3.

Protocols implementations

Chosen protocols implementations: we selected three implementations to test our protocols :



1. An IP packet is sent through the virtual interface.
2. This packet is encapsulated using a framework specific protocol then sent to the simulation.
3. The packet is decapsulated, then sent through the simulation within a MAC frame.
4. The packet is received by the destination node (in terms of MAC addresses), then reencapsulated to be sent to the Linux system's virtual interface.
5. The Linux system receives the IP packet.

Fig. 2: The virtualmesh functionning

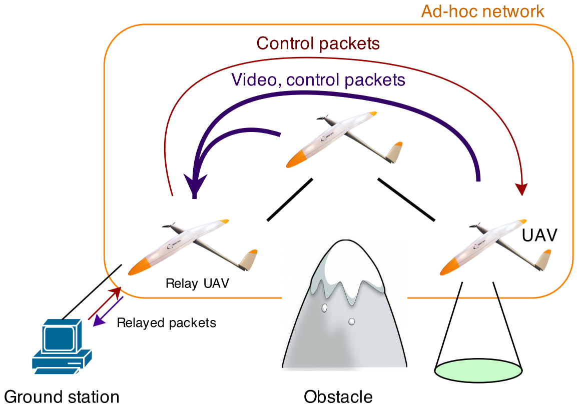


Fig. 3: Our video surveillance scenario

OLSR As a well maintained implementation of OLSR, we chose OLSRd [34]. It is an implementation compatible with several Linux version and is available in Ubuntu's packages repositories.

AODV We used AODV-uu [35] as the implementation of AODV. This is the only one we could find that is compatible with Linux. Nevertheless, it has been necessary to modify this version for it to be compatible with our Linux kernel version (2.6.38).

DSR As a Linux-compatible DSR implementation we used Piconet 2, as proposed in [36]. Unfortunately, this implementation was also not compatible with our kernel, and we chose to modify it to make it compatible with our kernel.

These implementations are parametrized according to the corresponding RFC recommended values.

Geographical protocols: concerning geographical protocols, their mechanisms are usually based on the following assertion : each node, to choose the next hop to the destination node, has to know the position of its close neighbours and the position of the destination. Thus, to deploy completely geographical protocols, we would have to implement an additional mechanism to exchange the different node positions through the Ad-Hoc Network (otherwise we would have to use an other communication medium, which is in our case not available). The main existing position sharing mechanisms are detailed in [33]. We did not choose to implement such a mechanism given that the evaluation of the routing protocol would have been deeply impacted by the specific additional mechanism we have selected. As future works, it could be interesting to analyse the different position sharing mechanisms available in the literature and implement one that is the most efficient to enhance this study.

Realistic traffic generation:

to test protocols performances, we generate a realist traffic. This is achieved by using real traces supplied by Delair-Tech, from a mission with only one UAV. We extracted control traffic characteristics and extrapolate these informations to a three UAVs mission. As a result, we have been able to recreate a realistic traffic scenario. As we supposed a video surveillance mission, we considered that a HD video would be the main applicative traffic from UAVs to the ground station. The used codec is the H264 for being a popular codec for this kind of video quality. We supposed a 4 Mbits throughput for a full HD image (1920x1080 pixels) at 30 images per seconds. H264 being a variable rate codec, we decided to include an arbitrary value of 50% variability for each image sent and separated in 1,000 bytes packet to avoid fragmentation. These different results are described in Table 2.

Type	Source→Destination	Paquet size	Rate
Tick	1→2,1→3	64 bytes	1.0 packet/s
Georef	2→1,3→1	64 bytes	1.8 packet/s
Command	1→2,1→3	64 bytes	0.034 packet/s
Video	2→1,3→1	4 Mbits/s	

Table 2: Generated traffic

	AODV	DSR	OLSR
Test duration (from the first to the last disconnection)	2h 57min 3s		
Disconnected state / Test duration	61.52 %	61.0 %	58.8 %
Unstable states / Test duration	3.78% (6 min 41s)		
Connectivity during unstable states	88.5 %	66.7 %	15.3 %

Traffic 3→1

	AODV	DSR	OLSR
Test duration (from the first to the last disconnection)	2h 58min 44s		
Disconnected state / Test duration	62.9 %	61.7 %	60.4 %
Unstable states / Test duration	3.79% (6 min 46s)		
Connectivity during unstable states	90.65 %	58.2 %	24.1 %

Traffic 2→1

Table 3: Connectivity results for a 3 hours test

4.4 Results and analysis

Connectivity

Connectivity analysis: we extracted disconnected states on the links 3→1 and 2→1, as being the most loaded traffics they allow an acurate active measurement. To prevent short unstable states to disturb the measurement, two losses that are too close in time (less than 0.1s) are merged. We performed the same mobility scenario with each protocol, so that the connectivity patterns would be similar. Thus, we synchronised each measurement on the first long loss (higher than 15s) to evaluate differences between each connectivity result. From this data, we extracted what we called "unstable states" which corresponds to states when all protocols do not behave the same way. By acting this way, we took away states during which connectivity, or non-connectivity, were stable for each protocol. This unstable states extraction is shown in Figure 4. We obtained the results exposed in Table 3. As we can see, the total connection time is slightly better for reactive protocols. In unstable states, AODV stand out from the others making able connectivity up to 90%. This means that AODV is the most reactive routing protocol to topology changes.

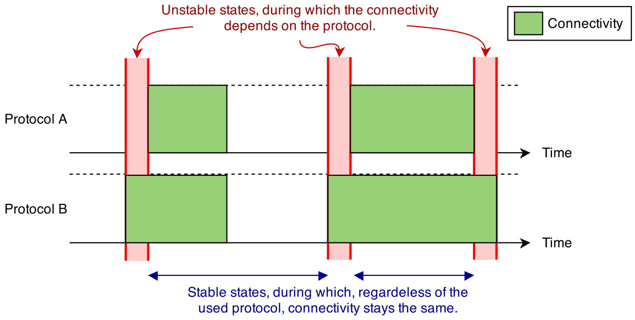

Unstable states, during which the connectivity depends on the protocol.
Connectivity
Protocol A
Time
Protocol B
Time
Stable states, during which, regardeless of the used protocol, connectivity stays the same.

Fig. 4: "Unstable states" extracted from the measurement

Overhead: we performed a one-hour test to evaluate the overhead created for each protocol and obtained the results exposed in Table 4. As we can see, OLSR is slightly better than AODV, but they both outperform DSR that has a really high overhead.

	AODV	DSR	OLSR
Control packets	501 kB	759,99 kB	438 kB
Traffic % (bytes)	0.034%	4.393%	0.027%
Headers added to data packets	0	26,723 kB	0
Average header length	0	17.8 B	0

Table 4: Protocol overhead over a one-hour test (captured from the interface of UAV 1)

Delays: concerning delays introduced by each protocol (exposed in Table 5), we notice that delays are really low. However we can see that DSR introduces a slightly higher delay than the other protocols but this is probably due to the DSR implementation. The fact that DSR needs to modify data packets by introducing a header, and that it uses acknowledgements for each packets sent, can also affect its performance.

	AODV	DSR	OLSR
Average delay	5.32 ms	10.15 ms	5.91 ms
Maximum delay	100.0 ms	100.0 ms	99.8 ms

Table 5: Average and maximum delays over a one hour test (captured from the interface of UAV 1)

Conclusion: as AODV being the protocol the most reactive to topology changes and having a limited overhead, we concluded that AODV was the most suitable routing protocol for our scenario. As several studies proved that AODV can be outperformed by proactive protocols with a large number of nodes, its on-demand mechanism allows here a faster response to routing needs. Our mobility pattern and our emulation system certainly affected the measures, but we found similar results to those exposed in [30].

5 Conclusion and future work

In this paper we introduce a hybrid experimental system which can be used for different types of Ad-Hoc Networks and to evaluate different types of network protocols. This evaluation framework is composed of a new set of tools to evaluate ad-hoc protocols thanks to virtual machines. Applied to UAANETs, we have been able to prove that, considering our realistic mobility scenario, AODV is a more suitable protocol than OLSR and DSR. Concerning further studies, the fact that we used an ideal physical model has certainly an impact over our results. It could be interesting to introduce a more realistic physical model to the simulation, and consequently adapt the experimental test protocol. As the Linux system used in our experimental test protocol is different from the real Delair-Tech embedded system, we would like to introduce a real system to the tests. These results could also be compared to results obtained with a real test bed, composed of multiple real UAVs. For further researches we decided to make available to download by the community the different tools we created for this experimental test protocol. This set of tools is available at the following link: http://www.recherche.enac.fr/resco/doku.php?id=emulationtestbed.

We hope that the tools we designed and shared online with the research community will be reused and extended for additionnal studies. Indeed, we think this work could help the design, the evaluation and the validation of future UAANET systems.

References

1. Staub, Thomas, Reto Gantenbein, and Torsten Braun. "VirtualMesh: an emulation framework for wireless mesh networks in OMNeT++." Proceedings of the 2nd

International Conference on Simulation Tools and Techniques. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009.

2. Hyland, M. T., et al. "Simulation-based performance evaluation of mobile ad hoc routing protocols in a swarm of unmanned aerial vehicles." Advanced Information Networking and Applications Workshops, 2007, AINAW'07. 21st International Conference on. Vol. 2. IEEE, 2007.
3. Hyland, Matthew T. Performance evaluation of ad hoc routing protocols in a swarm of autonomous unmanned aerial vehicles. No. AFIT/GCS/ENG/07-07. AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING AND MANAGEMENT, 2007.
4. Bekmezci, Ilker, Ozgur Koray Sahingoz, and amil Temel. "Flying ad-hoc networks (FANETs): a survey." Ad Hoc Networks 11.3 (2013): 1254-1270.
5. Chaumette, S., et al. "CARUS, an operational retasking application for a swarm of autonomous UAVs: first return on experience." MILITARY COMMUNICATIONS CONFERENCE, 2011-MILCOM 2011. IEEE, 2011.
6. Rosati, Stefano, et al. "Dynamic Routing for Flying Ad Hoc Networks." arXiv preprint arXiv:1406.4399 (2014).
7. Daniel, Kai, et al. "AirShield: A system-of-systems MUAV remote sensing architecture for disaster response." Systems Conference, 2009 3rd Annual IEEE. IEEE, 2009.
8. Cameron, Stephen, et al. "SUAAVE: Combining aerial robots and wireless networking." 25th Bristol International UAV Systems Conference. 2010.
9. Xu, Zanxin, et al. "Analyzing two connectivities in UAV-ground mobile ad hoc networks." Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on. Vol. 2. IEEE, 2011.
10. Bouachir, Ouns, et al. "A Mobility Model For UAV Ad hoc Network." ICUAS'14 Proceedings (2014).
11. Sahingoz, Ozgur Koray. "Networking models in flying Ad-hoc networks (FANETs): Concepts and challenges." Journal of Intelligent - Robotic Systems 74.1-2 (2014): 513-527.
12. Mauve, Martin, Jorg Widmer, and Hannes Hartenstein. "A survey on position-based routing in mobile ad hoc networks." Network, IEEE 15.6 (2001): 30-39.
13. Clausen, Thomas, et al. "Optimized link state routing protocol (OLSR)." (2003).
14. Alshbatat, Abdel Ilah, and Liang Dong. "Cross layer design for mobile ad-hoc unmanned aerial vehicle communication networks." Networking, Sensing and Control (ICNSC), 2010 International Conference on. IEEE, 2010.
15. Paul, Amrita Bose, and Sukumar Nandi. "Modified optimized link state routing (M-OLSR) for wireless mesh networks." Information Technology, 2008. ICIT'08. International Conference on. IEEE, 2008.
16. Belhassen, Mohamed, Abdelfettah Belghith, and Mohamed Amine Abid. "Performance evaluation of a cartography enhanced OLSR for mobile multi-hop ad hoc networks." Wireless Advanced (WiAd), 2011. IEEE, 2011.
17. Perkins, Charles E., and Pravin Bhagwat. "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers." ACM SIGCOMM Computer Communication Review. Vol. 24. No. 4. ACM, 1994.
18. Open-mesh. URL: http://www.open-mesh.org/projects/open-mesh/wiki (visited on 08/04/2014).
19. Perkins, Charles, E. Belding-Royer, and Samir Das. "Ad hoc on demand distance vector (AODV) routing (RFC 3561)." IETF MANET Working Group (August. 2003) (2003).

20. Aggarwal, Akshai, et al. "AODVSEC: A novel approach to secure Ad Hoc on-Demand Distance Vector (AODV) routing protocol from insider attacks in MANETs." arXiv preprint arXiv:1208.1959 (2012).
21. Royer, Elizabeth M. "Multicast ad hoc on-demand distance vector (MAODV) routing." IETF Internet Draft, draft-ietf-manet-maodv-00. txt (2000).
22. D. Maltz D. Johnson Y. Hu. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4. RFC 4728. Feb. 2007. URL: http://www.ietf.org/rfc/rfc4728.txt
23. Bose, Prosenjit, et al. "Routing with guaranteed delivery in ad hoc wireless networks." Wireless networks 7.6 (2001): 609-616.
24. John Adrian Bondy and Uppaluri Siva Ramachandra Murty. Graph theory with applications. Vol. 6. Macmillan London, 1976
25. Brad Karp and Hsiang-Tsung Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In: Proceedings of the 6th annual international conference on Mobile computing and networking. ACM. 2000, pp. 243254.
26. Lin, Lin, et al. "A novel geographic position mobility oriented routing strategy for UAVs." Journal of Computational Information Systems 8.2 (2012): 709-716.
27. Lidowski, Robert L., Barry E. Mullins, and Rusty O. Baldwin. "A novel communications protocol using geographic routing for swarming uavs performing a search mission." Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on. IEEE, 2009.
28. Lin, Lin, et al. "A geographic mobility prediction routing protocol for Ad Hoc UAV Network." Globecom Workshops (GC Wkshps), 2012 IEEE. IEEE, 2012.
29. Shirani, Rostam. Reactive-greedy-reactive in unmanned aeronautical ad-hoc networks: A combinational routing mechanism. Diss. Carleton University Ottawa, 2011.
30. Performance of mobile ad hoc networking routing protocols in realistic scenarios. In: Military Communications Conference, 2003. MILCOM 03. 2003 IEEE. Vol. 2. 2003, 12681273 Vol.2. DOI: 10.1109/MILCOM. 2003.1290408.
31. OMNeT++. URL: http://www.omnetpp.org/ (visited on 07/29/2014).
32. ORACLE VM Virtualbox. URL: https://www.virtualbox.org/ (visited on 07/28/2014).
33. Jorg Widmer et al. The Handbook of Ad Hoc Wireless Networks. In: ed. by Mohammad Ilyas and Richard C. Dorf. Boca Raton, FL, USA: CRC Press, Inc., 2003. Chap. Position-based Routing in Ad Hoc Wireless Networks, pp. 219232. ISBN: 0-8493-1332-5. URL: http://dl.acm.org/citation.cfm?id=989711.989724.
34. OLSRd, an adhoc wireless mesh routing protocol. URL: http://www.olsr.org/ (visited on 07/29/2014).
35. AODV-UU. URL: http://aodvuu.sourceforge.net/ (visited on 07/29/2014).
36. Alex Song. PicoNet II - A Wireless Ad-hoc Network For Mobile Handheld Devices. PhD thesis. University of Queensland, Oct. 2001.
37. Forsmann, J. Hope, Robert E. Hiromoto, and John Svoboda. "A time-slotted on-demand routing protocol for mobile ad hoc unmanned vehicle systems." Defense and Security Symposium. International Society for Optics and Photonics, 2007.