SEER4US, Secured Energy Efficient Routing for UAV Swarms

Benoît Fournier, Gilles Guette, Valérie Viet Triem Tong, Jean-Louis Lanet

SEER4US, Secured Energy Efficient Routing for UAV Swarms

Benoît Fournier, Gilles Guette, Valérie Viet Triem Tong and Jean-Louis Lanet
CIDRE research group Rennes France
Univ Rennes, Inria, CentraleSupelec, CNRS, IRISA
name.surname@inria.fr

***Abstract*—This article introduces SEER4US a secured routing protocol designed for UAV swarm networks. SEER4US is the first protocol providing integrity of routing messages and authentication of their sender with low energy consumption for battery preservation. SEER4US prevents the UAV swarms from usual routing attacks enabled when routing messages are modified or replayed by nodes external to the swarm. SEER4US is an extension of the pro-active routing protocol OLSR designed for mobile ad-hoc networks. SEER4US is also inspired from TESLA, a protocol designed for stream signature allowing authentication of stream messages sender. After specifying SEER4US, we evaluate here the energy requirements of this protocol and we show that the use of lightweight cryptography allows a significant energy saving compared to traditional cryptographic schemes.**

***Index Terms*—Network security, UAV swarm, energy efficiency, ad hoc routing protocol**

I. Introduction

Unmanned aerial vehicle (UAV) applications and development have increased over the past few years as this technology has become more accessible and less expensive. Its primarily purpose is divided in two main sectors: transport and coverage. Drones are used for the transport of goods, or people and serve as interconnected air vehicles.

Coverage applications can take different aspects. Basically UAV will move in an area and carry a payload that interacts with the environment (monitoring for instance).

Nowadays, most applications rely on only one single device, but it is natural to think that in the near future, swarms will be able to cooperate to achieve a mission. A swarm will increase reliability, as it is tolerant to individual failure. It will also be faster in discovery mission by splitting workload among several devices. We focus here on autonomous UAV, able to dynamically change their behavior and adapt to unplanned event. This is not a reality yet, but we already see swarm that execute full-planned mission successfully. Some important issues remain to be solved: secured communications, control of the swarm, resilience to malicious behaviors.

On a single UAV scenario, communication is a keystone to transmit commands and retrieve data from UAV sensors. It is even more critical in swarm where cooperation and inter messaging is fundamental. The communication between the nodes of a swarm is based on a suitable routing algorithm. The routing must allow each node to send messages to each other, by successive hops between different neighbors.

A UAV swarm is a particular mobile ad-hoc networks where nodes run independently but form a cooperative communication newtwork. Routing information in these networks is a broadly explored domain. Many classes of routing protocol exists, most popular are reactive and proactive protocol (AODV [16] and OLSR [7]). A UAV swarm shares common characteristics with VANET (vehicular ad hoc network), sensors network or mobile phone network but also strongly differs on specific points. The mobility model of UAV swarm is in a 3D space, since they do not follow road or marked path contrary to vehiculars. VANET routing protocols [4] rely on specific characteristics of the vehicle *i.e.* positioning system, infrastructure access, almost unlimited energy and high computing capacities. On the contrary, any computation on a UAV is a permanent trade off between volume, weight and power consumption, with no infrastructure access.

It also differs from sensor networks that are mainly stable networks where communication is made from the sensor to the sink with more or less stable routes. In UAV networks, communication is more critic thus latency and disconnection have to be minimized. Mobility of nodes is also quite dissimilar. And to finish, it differs from mobile phones network mainly with the network infrastructure, cellular network are not part of our assumption for UAV deployment.

UAV network have always one or two major differences with other kind of MANET and cannot use off the shelf technologies to achieve best effectiveness. Existing routing protocols are not well suited but attack scenarii are. UAV network is also subject to many known routing attacks: black hole [2], wormhole [13], Sybil attacks [8] ... Attacker's goal is to disturb or completely stop communication between specific nodes of the network while remaining stealth. Such attacks can lead to disconnection from the swarm, false environment awareness, incorrect behavior. This can therefore be exploited in conjunction with other attacks. For example to take control of UAV, retrieve data from it, capture it or even just crash it. As part of a swarm, a compromised node can threaten the others. Security issues need to be addressed with minimal cost, to not impact flight autonomy.

In this article, we aim to address current threats on routing in UAV swarm. More precisely, we presents three contributions:

- SEER4US the first routing protocol dedicated to UAV swarm ensuring integrity and authentication of routing messages with low energy consumption.

- A measure of theoretical latency threshold we can achieve with SEER4US.
- An evaluation of energy consumption of SEER4US protocol with present-day hardware

In the remainder, Section II discusses existing protocols dealing with routing and security in mobile ad-hoc networks, Section III introduces two protocols namely OLSR and TESLA [17] that inspired SEER4US. The core of our protocol is detailed in Section IV, latency is deeply studied in Section V, security properties are detailed in Section VI. Finally, in Section VII we compare energy consumption with other solutions.

II. Related works

Our problem covers both routing and security issues in mobile ad-hoc networks. Different approaches exist, covering each side, but rarely both. Routing in UAV network has been addressed on the literature.

In [20], authors focus on airborne network, comparing their protocol to OLSR and AODV. They conduct their evaluation on a defined scenario with four nodes that have circular trajectory with fixed radius and speed. They focus on latency and throughput. Their protocol assumes that flight plans of all nodes are common knowledge.

In [18] they emulate two different scenarii: one with four nodes, and one with 32, in both cases most of them are not moving. They focus on increasing routing performance by taking GPS position and movement of nodes.

In [6] they evaluate best routing protocols for another use case. They used fixed path for UAV movement and simulate 40 nodes. They focus on latency and network overhead, moreover they choose to have heterogeneous network composition: some nodes will be dedicated for long range communications.

In [1], Adjih *et al.* proposed a secured proactive routing protocol based on OLSR. It provides integrity, authentication and timestamping of routing messages. It can use either symmetric or asymmetric signature algorithms. They assume that necessary keys are available on each node to perform suitable verifications.

Very closely, in [21], Zapata proposed a secured reactive routing protocol based on AODV. It provides integrity and authentication of routing messages. It uses asymmetric signature and a key management scheme. On each hop, message signature is verified and if valid the message is processed.

Ariadne protocol [11] is based on DSR a reactive routing protocol. In order to provide integrity and authentication of routing messages the protocol uses a signature mechanism. The choice of this mechanism is left to end user among private/public key, secret shared key and TESLA mechanism. The setup of those keys needs a specific initialization and during execution some nodes act as trusted key managers.

To our knowledge there is only one previous work that explore both routing and security on UAV network, the SUANET project [14]. They propose a routing protocol based on AODV. They use public key cryptography to provide confidentiality, integrity and authentication of routing messages. They also include packet leashes security mechanism. To conduct their simulation they used scenarii with 4 nodes extracted from a single UAV real traces. Energy consumption overhead is out of scope of their studies.

III. Building Blocks

SEER4US, is based on two existing protocols: a pro-active ad hoc routing protocol (OLSR) and an authentication protocol (TESLA) mainly dedicated to stream signature.

OLSR is a proactive routing protocol, developed to address routing problem in mobile ad hoc network and other wireless ad hoc networks [7]. When running OLSR, all nodes periodically exchange `Hello` message to advertise neighbours of their presence. `Hello` messages sent by a node also contains the list of the neighbours learned from previous received `Hello` messages. `Hello` messages are one hop message, they are broadcast but not retransmitted. The nodes are able to construct a 2-hops view of the network topology thanks to `Hello` messages,. To access the entire network, a node elects some other nodes as Multi Point Relay (MPR). For a given node the set of its MPR is the minimum set allowing to reach all the two hops neighbors. MPR sends periodically Topology Control (`TC`) messages. A `TC` message contains information about the MPR initiator of the message and nodes that elected it. When a MPR A receives a `TC` message from one node B that has elected it as MPR, A forwards it to its one hop neighbors. With `TC` and `Hello` messages all nodes are able to construct the topology of entire network. As messages are periodical, network topology is updated accordingly. With OLSR, nodes are able to route message in ad hoc network, assuming execution is correct. Unfortunately anybody can inject malicious node in the network and perform incorrect behavior. Those kinds of attack is made possible because routing messages are not protected. They are received and processed without any form of checking about integrity or authentication. That means even a single malicious node can block a large amount of traffic, forbidding communications between nodes [2]. This lack of security has to be addressed to deploy a reliable network.

TESLA [17] is a security scheme initially proposed in the constrained context of continuous authentication of data stream needing strong packet-loss robustness, high scalability, and minimal overhead (as radio and TV Internet broadcasts, and authenticated data distribution by satellite). TESLA relies on hash function and global time. To use TESLA on a message stream, a sender owns an arbitrary long hash chain. Elements of hash chain will be used as keys in reverse order. We use last chain element $h^n(S_0)$ as first key K_0. Any message m_i broadcasted by the source node is composed of three elements: $p_i, K_{i-1}, H_{\text{MAC}}(p_i, K_{i-1}, K_i)$. p_i is the data payload contained in the packet. K_{i-1} is the previous key used in the chain, disclosed in m_i. $H_{\text{MAC}}(p_i, K_{i-1}, K_i)$, H_{MAC} (Keyed-hash message authentication code) of m_i, using yet undisclosed key K_i. To process m_i through verification steps, receivers have to know if it was received before sending of m_{i+1} and then disclosure of K_i. To ensure this property,

keys revelation is scheduled and node are synchronized with a bounded error. On m_i reception, m_{i-1} verification will be allowed and m_i will be buffered, waiting for m_{i+1} reception in order to be verified. To verify m_i, receivers check that $h(K_i) = K_{i-1}$ with K_i disclosed in m_{i+1}. This step will ensure that m_i belongs to message stream. Additionally receivers will compute $H_{\text{MAC}}(p_i, K_{i-1}, K_i)$ and compare it with H_{MAC} in m_i. By assuming that h cannot be reversed only the owner of the key chain can produce legitimate message on time. Those message cannot be stealthy tampered due to H_{MAC}. That ensures integrity of message and authentication of sender. The very first message needs an external method of authentication or the prior transmission of K_0.

OLSR alone can be used for UAV swarm but, according to its latest definition [16], it is strongly recommended to add mechanism to ensure message integrity and authentication. Many attacks are, indeed, based on unsecure routing message: replay attack [9], blackhole attack [2]. On the other hand, TESLA alone allows authentication of sender of a data stream but is not a routing protocol. It relies on hash function and loose synchronization rather than heavy computational cryptographic functions and bring two good properties we want: low energy consumption and less computation. In the rest of this article, we design SEER4US, a secure ad hoc routing protocol providing authentication and integrity of routing messages, based on OLSR and TESLA, well suited to UAV swarm due to its low energy consumption (see section VII).

IV. SEER4US

SEER4US is designed for autonomous UAV swarm where UAVs share a common knowledge initialized offline.

1) Common knowledge: We assume that all nodes share a global clock: each node has its own local clock but maximum deviation is supposed to be bounded by a fixed parameter ϵ. In SEER4US, each node A of the swarm is equipped with an arbitrary fixed-length key chain $(K_0^A, \ldots, K_n^A)$ where the ending value K_n^A is randomly chosen by each node and $K_{j-1}^A = h(K_j^A)$ where h is a one way hash function. The set of all the starting values $\{K_0^N\}$ (where N is in the set of nodes) is part of the common knowledge of the swarm. Each node will disclose the keys of its key chain according to a frequency that is assumed to be known by all nodes in the swarm. Starting form a arbitrary value k, it is easy to verify that k is the an element of the chain by checking if $h^x(k)$ is equal to a previously disclosed key for some value of x in $\{0, ..n\}$. One or more key may be missing in the disclosed key chain. Each node collects information about the network by exchanging `Hello` and `Topology Control` (`TC`) messages with the same purpose as in OLSR. `Hello` messages will be periodical 1-hop message and `TC` will be periodical message broadcasted in the entire network through MPR mechanism. `Hello` will contains addresses of neighbours nodes and `TC` addresses of nodes choosing `TC`'s senders for MPR.

2) Messages formatting: In SEER4US, each message is linked to its predecessor through the key chain of their sender, in order to ensure the integrity of message and authentication of the sender. More precisely, both `Hello` and `TC` messages contain two additional fields: a *MAC* field and a *key* field. Thus, instead of sending a series of messages $\{m_1\}, \{m_2\}, \ldots, \{m_i\}, \ldots$ of common `Hello` or `TC` messages, a node A will send:

$$\{m_1^A\} = (p_1^A, K_0^A, H_{\text{MAC}}(p_1^A, K_0^A, K_1^A)),$$
$$\{m_2^A\} = (p_2^A, K_1^A, H_{\text{MAC}}(p_2^A, K_1^A, K_2^A)),$$
$$\ldots,$$
$$\{m_i^A\} = (p_i^A, K_{j-1}^A, H_{\text{MAC}}(p_{i-1}^A, K_{j-1}^A, K_j^A)),$$
$$\ldots,$$

3) Verification step: When a node B receives a series of messages supposedly sent by A, he can, at each message reception, check the integrity of previous message and that A is indeed the sender of the message. On reception of a message m_i^A two verifications are performed: first that m_i^A is received on time according to K_j^A disclose time and B internal clock.

Secondly that K_{j-1}^A belongs to the key chain of A, to verify this property, B applies h on K_{j-1}^A until it can compare it with the last revealed K^A known, or ultimately K_0^A. If one of the verification failed the message is discarded, otherwise the message is stored awaiting for authentication and the revealed key chain of A is updated.

On reception of a next message from A, m_{i+1}^A, which disclose K_j^A further verification are performed. Firstly same two verifications are done and will update A's revealed key chain. As m_{i+1}^A is sent after m_i^A key revealed on it is the next one or is the same as the key used to sign m_i^A. So B is able to retrieve K_j^A by applying h on K_{j-1}^A. With this new knowledge B can computed if $H_{\text{MAC}}(d_i^A, K_j^A)$ is same as received or not (for convenience we note $d_i^A = p_i^A, K_{j-1}^A$). If it is the same then m_i^A is authenticated and p_i^A can be transmit to the routing algorithm. If verification is incorrect then payload of m_i^A is discarded. Fig 1 displays few round of communication from A to B. Where p_i^A is the payload of the routing message, K_j^A is the key used to sign m_i^A and K_{j-1}^A is is the disclosed key.

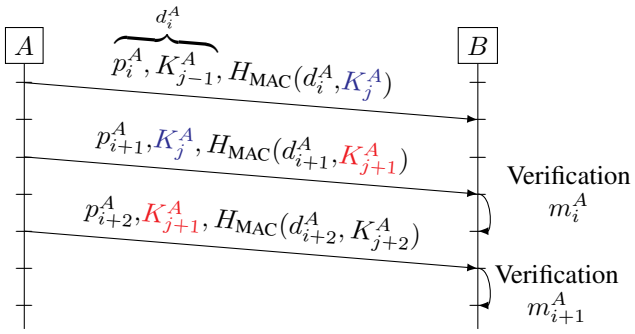

Fig. 1. Messages Exchanges diagram

With this verification process we ensure integrity and authentication of routing messages.

SEER4US scheme as previously exposed in Section IV uses fixed frequencies for message sending. But in fact OLSR is much more flexible and can overload frequency by sending

message sooner in case of topology updates. This feature better copes in UAV swarm where low latency routing is critical and topology update are potentially high. To include this reactivity, we can use a faster key disclosure frequency in SEER4US.

Since a node has to wait upon key disclosure, the key lifetime has to be long enough for message to be received, a balance for key disclosure rate is needed. In the following, we discuss the minimum value for key lifetime.

`Hello` messages will be send every Δ_{Hello} and `TC` messages every Δ_{TC}. Those periods do not form an exact schedule nor are respected all the time. First because of common wireless link, packets can be delayed on sending, to avoid collision. Second because when a node received a new `Hello` or `TC` it can sooner advertise its others neighbours by immediately sending a new message. Still, to avoid complete flooding, there is a minimum time to wait before sending another message.

To comply with this dynamic rate we set minimum validity time of key $\delta = \Delta_{tt} + \varepsilon + g$ as depicted in Figure 2. Where Δ_{tt} is an upper bound of transmission time, ε an upper bound of synchronization error and g an upper bound of jitter time. Even in worst case, a message is valid at reception if it is sent δ before its key expiration. The minimum time to wait before sending another message has to be greater or equal to δ.

Keys used to sign the chain of message will not be necessarily direct successor in the keys chain. So verification step will repetitively apply h on the key to link it to a previous disclosed one.

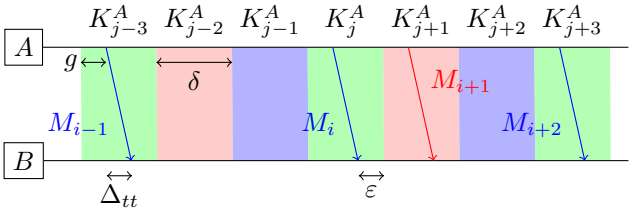

Fig. 2. Key disclosure

Base routing and signature are now in place but we aim to deploy our protocol on a fast and high dynamic network. Waiting for upcoming message introduce extra latency to reduce.

V. Addressing latency

As authentication of a message implies to wait the next one, extra latency is introduced by SEER4US. Being able to communicate flawlessly is critical for an UAV, as without communication it is more vulnerable, perform worst or even being a danger for the swarm. Latency of routing message can lead to partial disconnection when node will no longer being able to communicate with others. To ensure that our protocol is suitable for UAV swarm we aim to reduce this extra latency to the minimal as described in the following Section.

In Figure 3, we see journey of a TC^A_i from A to D.

On the first hop, B will authenticate TC^A_i with TC^A_{i+1}, introducing one round of latency. On TC^A_{i+1} reception, if we

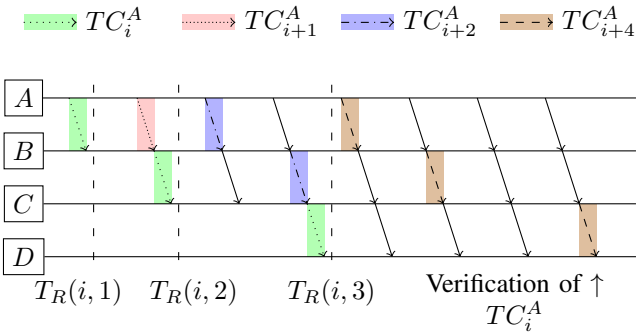

Fig. 3. Latency over hop

neglect verification time, B will re emit TC^A_i immediately after verification. So C will received TC^A_i with two rounds of latency. Only TC^A_{i+2} (or later message) can allows C to authenticate TC^A_i. But TC^A_{i+2} will first pass trough B and so have additional latency. C will only be able to authenticate TC^A_i with three rounds of latency. Finally, three-hop node D will receive TC^A_i from C but will wait until reception of TC^A_{i+4} to authenticate it. TC^A_{i+4} will pass trough B and C leading to a final seven round latency for TC^A_i authentication. Authentication latency seems to growth exponentially with number of hops.

We discretize here the time to interval of equal length so that there is one `TC` sent per unit of time. We define Δ_{ll}, the link latency, the time for a message to be received by another node, $0 < \Delta_{ll} \leq 1$, as communication are not immediate but shorter than message period. Let $T_R(i,n)$ time to receive `TC` sent at i, that is emitted from n hops. Let $T_A(i,n)$ time to authenticate `TC` sent at i, that is emitted from n hops. `TC` is broadcast as soon as they are authenticated and we assume validation time is negligible: $T_R(i,n) = T_A(i,n-1) + \Delta_{ll}$. `TC` is authenticated when a message sent after its reception is received: $T_A(i,n) = T_R(\lceil T_R(i,n) \rceil, n)$. Easily we have: $T_R(i,n) \geq i + 2^{n-1} - 1 + n \cdot \Delta_{ll}$ and $T_A(i,n) \geq i + 2^n - 1 + n \cdot \Delta_{ll}$.

So the farer a `TC` will be broadcast, the longer it will take to authenticate it. The exponential growth of this latency does not suit our short latency constraint.

To reduce latency we make every node re authenticate `TC` message. By replacing message signature on each hop we reduce authentication to one hop waiting. On Figure 4 we can see the same scenario as on Figure 3 but with each node applying is own signature before re emission.

Actually we have $T_A(i,n) = T_R(\lceil T_R(i,n) \rceil, 1)$. As all messages will be authenticated with direct neighbors key chain, that leads to $T_R(i,n) \geq i + n - 1 + n \cdot \Delta_{ll}$ $T_A(i,n) \geq i + n + n \cdot \Delta_{ll}$.

So now latency to authenticate and receive message grows linearly with the size of the network.

Verification time can also be speed up by introducing a new mechanism: `Validation` message. This new kind of message will be sent after all message after δ time. `Validation` message will not carry any routing information and their only

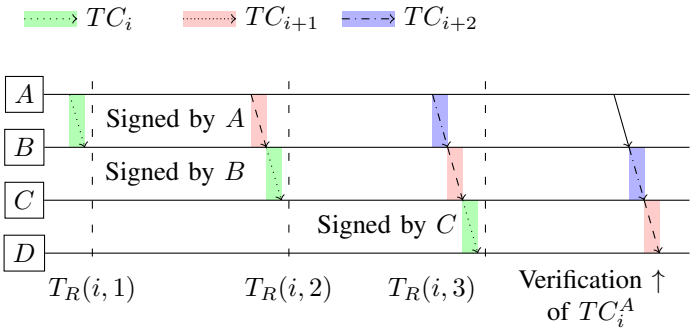

Fig. 4. Latency over hop with re-signature

purpose is to allow receivers to authenticate message with routing payload.

Fig 5 shows on the same timescale messages propagation as in Fig 4, we can see that `TC` is propagated faster tanks to `Validation` messages.

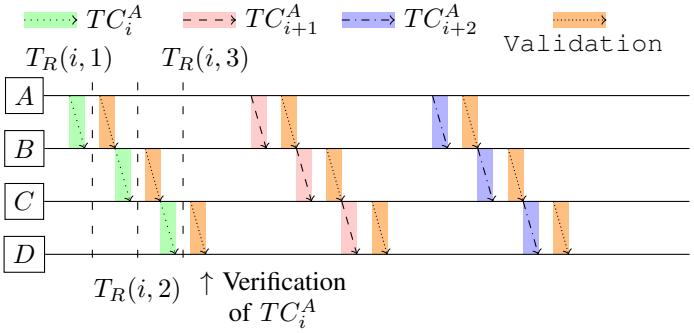

Fig. 5. New message to speed up authentication

With this modification we reach the theoretical threshold $2 * (\delta_{tt} + g) + \epsilon$ of minimum latency for one hop signature. After every `Hello` or `TC` message a `Validation` message is sent as soon as possible for validation process.

VI. Security discussion

Our threat model supposed an attacker that can interact with the transmission layer. It can effortlessly record all messages sent by any node. It can also emit all messages it can forged with its current knowledge and send them to any set of nodes he wants. We assume that an attacker may be deployed on the entire network. In its initial knowledge are nodes addresses. The attacker is neither able to reverse the hash function nor to forge a valid signature. We assume that one goal of the attacker is to modify routing information while leaving stealth, so jamming attack is out of scope of our study.

Before being process by routing routine we choose to buffer all messages and wait for authentication. One policy can be to directly use those routing information and make a proper validation after. This behavior can lead to energy depletion attack. One attacker can send a lot of incorrect messages to force heavy and useless topology processing. Same choice can be applied on `TC` broadcasting, we choose to wait to be sure `TC` messages are correct before MPR spread them. An hybrid policy approach may also be applied: at the beginning node can process message immediately and check authentication later but in case of error they switch in a vigilant mode in which they buffer and wait for verification.

On reception two checks are performed before message is buffered for authentication. First lifetime of the packet is checked, if the key that is used to sign it is already disclosed (or disclosed in a time inferior than the synchronization error ϵ) the packet is discarded. Second the revealed key in the packet is verified, by iteratively computing hash function on it. The key must be part of a known key chain, otherwise the packet is discarded. If the key is correct then packet is stored for future authentication. The verified disclosed key may be use to authenticate buffered packets from the same transmitter.

VII. Evaluation

In order to evaluate our protocol we conduct a theoretical comparison with OLSR and SOLSR, a secured version of OLSR, using RSA 2048 signature for routing messages and SEER4US. These experiments highlight that SEER4US protocol consume less energy for the same security properties and same information exchanged.

Most evaluations on MANET protocols use either real data or dynamic simulation, while others choose static configurations [12], [20]. Mobility models range from extremely specific (one node move at a time) to common and consensual model like random walk, random waypoint. Those two last models have become quite popular in evaluation of MANET protocols but for our use case, in which we want to simulate behaviour of coordinated swarm, those models cannot render coordination between nodes as explain in [10] and [5]. Other model begin to emerge for this specific use case: in [3] they render swarm mobility by combining tool for flight planning and markov chain. In [5] by combining existing basic mobility model they construct a more suitable mobility model for group mobility. Unfortunately even if they are quite promising we choose to do not use them yet: according to [3] there is no consensual method to evaluate those models and measure if they correctly render reality. For our first evaluation we choose to stick to a simple setup: fixed position, we choose best case option for all the evaluated protocols, ideal link layer, no collision, ideal topology and so no movement.

We study 3 protocols: OLSRv1 as a baseline, SEER4US and SOLSR. For this experiments, we used the recommended configuration parameters from OLSR and TESLA. Adresses are ipv4 on 4 bytes. We use SHA-256 hash function for both key chain and hmac. We truncate hashes on message to reduce its size to 10 bytes as recommended in [17]. `TC` period is 5s and `Hello` period is 2s. Timestamp is on 4 bytes. SOLSR uses RSA2048 with 256 bytes signature. We count energy consumption on a ten seconds time frame.

There is four ways to spend energy: sending messages, receiving messages, signing and verifying a message received. We consider no energy consumption for OLSR on sign and verify operation. We also do not take into account energy

spent to perform routing calculation like MPR set election, route processing and so on, as such operation will be the same on each protocol and do not introduce useful element of comparison.

As our model is simple enough, we don't need an external simulation tool, we will add cost of every tasks performed by all nodes successively over the time frame. All 49 nodes form the best case topology for OLSR. For each of them we have S_n, R_n the set of messages sent, received respectively. We can construct and measure size of all message that will give us the overhead. We re-use values published in [19] about energy cost for sending and receiving. We took experimental values from [15] for hash/signature consumption, by adding each verify and signing operation for each protocol we can then have energy consumption for both SEER4US and SOLSR. For SEER4US, messages are signed at every hop but for RSA signature only the first sender performs signature.

For SOLSR, another cost per message is the access to the key for verification but we neglect this cost here since it highly depends of the infrastructure.

	OLSR	SOLSR	SEER4US
nb of message per second	5	5	10
total power consume (mW)	3184	11882	6104
Kbyte/s sent in average	3.500	13.791	6.952
Average Overhead (Kbytes/s)	-	10.291	3.452

TABLE I
COMPARISON OF ENERGY CONSUMPTION IN SEER4US, OLSR, SOLSR

Results detailed in Table I give a hints on how cryptography drastically increased energy consumption. So signatures come at a cost, but we see that our protocol cost nearly twice less than SOLSR still guaranteeing integrity and authentication. We can also see that even if our protocol sends more messages due to double sending mechanism, SEER4US still have a smaller overhead than SOLSR compared to baseline.

VIII. CONCLUSION

We have presented SEER4US a proactive routing protocol for UAV swarm based on OLSR and TESLA, that provides integrity of routing message and authentication of senders. We show in our evaluation that this protocol consumes less energy and has significantly less overhead than SOLSR, another proactive routing protocol. Future work will focus on improving SEER4US by handling dynamic arrival and departure of node in the network and cooperation with other networks. At last, we will propose a proper mobility model that we will use to measure throughput, latency or packet drop.

REFERENCES

[1] Cedric Adjih, Thomas Clausen, Philippe Jacquet, Anis Laouiti, Paul Muhlethaler, and Daniele Raffo. Securing the olsr protocol. In *Proceedings of Med-Hoc-Net*, pages 25–27, 2003.

[2] Anu Bala, Munish Bansal, and Jagpreet Singh. Performance analysis of MANET under blackhole attack. In *2009 First International Conference on Networks & Communications*. IEEE, 2009.

[3] Ouns Bouachir, Alinoe Abrassart, Fabien Garcia, and Nicolas Larrieu. A mobility model for UAV ad hoc network. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 383–388. IEEE, IEEE, may 2014.

[4] Timothy Brown, Brian Argrow, Cory Dixon, Sheetalkumar Doshi, Roshan-George Thekkekunnel, and Daniel Henkel. Ad hoc UAV ground network (AUGNet). In *AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit*, page 6321. American Institute of Aeronautics and Astronautics, sep 2004.

[5] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing*, 2(5):483–502, 2002.

[6] Bow-Nan Cheng and Scott Moore. A comparison of MANET routing protocols on airborne tactical networks. In *MILCOM 2012 - 2012 IEEE Military Communications Conference*, pages 1–6. IEEE, IEEE, oct 2012.

[7] T. Clausen and P. Jacquet. Optimized link state routing protocol (olsr). RFC 3626, RFC Editor, October 2003.

[8] John R. Douceur. The sybil attack. In *Peer-to-Peer Systems*, pages 251–260. Springer Berlin Heidelberg, 2002.

[9] Priyanka Goyal, Vinti Parmar, and Rahul Rishi. Manet: vulnerabilities, challenges, attacks, application. *IJCEM International Journal of Computational Engineering & Management*, 11(2011):32–37, 2011.

[10] Xiaoyan Hong, Mario Gerla, Guangyu Pei, and Ching-Chuan Chiang. A group mobility model for ad hoc wireless networks. In *Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems - MSWiM '99*, pages 53–60. ACM, ACM Press, 1999.

[11] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. *Wireless Networks*, 11(1-2):21–38, jan 2005.

[12] Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael Degermark. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 195–206. ACM, 1999.

[13] Reshmi Maulik and Nabendu Chaki. A study on wormhole attacks in manet. *International Journal of Computer Information Systems and Industrial Management Applications*, 3(1):271–279, 2011.

[14] Jean-Aimé Maxa, Mohamed-Slim Ben Mahmoud, and Nicolas Larrieu. Survey on uaanet routing protocols and network security challenges. *Ad Hoc & Sensor Wireless Networks*, 2017.

[15] Jose A. Montenegro, Monica Pinto, and Lidia Fuentes. What do software developers need to know to build secure energy-efficient android applications? *IEEE Access*, 6:1428–1450, 2018.

[16] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (AODV) routing. Technical report, jul 2003.

[17] A. Perrig, R. Canetti, J.D. Tygar, and Dawn Song. Efficient authentication and signing of multicast streams over lossy channels. In *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, pages 56–73. IEEE Comput. Soc, May 2000.

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

[19] Swetank Kumar Saha, Pratik Deshpande, Pranav P. Inamdar, Ramanujan K Sheshadri, and Dimitrios Koutsonikolas. Power-throughput tradeoffs of 802.11n/ac in smartphones. In *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, apr 2015.

[20] Abhishek Tiwari, Anurag Ganguli, Ashwin Sampath, D. Scott Anderson, Bao hong Shen, Niyant Krishnamurthi, Joseph Yadegar, Mario Gerla, and David Krzysiak. Mobility aware routing for the airborne network backbone. In *MILCOM 2008 - 2008 IEEE Military Communications Conference*, pages 1–7. IEEE, IEEE, nov 2008.

[21] Manel Guerrero Zapata. Secure ad hoc on-demand distance vector routing. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(3):106, jun 2002.