Extended Verification of Secure UAANET Routing Protocol

Jean-Aimé Maxa, Mohamed-Slim Ben Mahmoud, Nicolas Larrieu

Extended Verification of Secure UAANET Routing Protocol

Jean-Aimé Maxa[1 2] , Mohamed Slim Ben Mahmoud[1] and Nicolas Larrieu[1]
[1] ENAC, TELECOM/Resco, F-31055 Toulouse, France
[2] Univ de Toulouse, F-31400 Toulouse, France
maxa@recherche.enac.fr
slim.ben.mahmoud@gmail.com
nicolas.larrieu@enac.fr

***Abstract*—**

UAV Ad hoc Network (UAANET) is a wireless ad hoc network composed of Unmanned Aerial Vehicles (UAVs) and Ground Control Station (GCS). It requires an efficient and secure routing protocols to find accurate and secure route between nodes to exchange data traffics. There have been several secure routing proposals to ensure data authentication and integrity services of ad hoc routing protocols. However, most of them are vulnerable against wormhole attacks and therefore cannot be used for UAANET directly without amendment. The wormhole attack involves two attackers who perform a colluding attack.

In this paper, we present a new UAANET secure routing protocol called SUAP (Secure Uav Ad hoc routing Protocol). It ensures message authentication and provides detection and prevention of wormhole attacks. SUAP is a reactive protocol using public key cryptography, hash chains and geographical leashes. We have carried out a formal verification analysis of SUAP security properties using the AVISPA tool, an automated model checker for the analysis of security features. We have also validated our security proposal through formal model checking using Simulink and Stateflow tools. Additionally, we use a hybrid experimental system (based on virtual machines and a virtual mesh framework) under a realistic UAANET scenario to evaluate SUAP routing performances and validate its security properties.

***Index Terms*—UAV Ad hoc NETwork, Security Architecture, AVISPA, Model Driven Development, Routing Protocol.**

I. Introduction

Technological and research advances in embedded systems help to produce small UAVs with highly effective capacities. Recently, their reputation has drastically increased as their use is not limited to the military domain anymore but extended to civilian applications to perform automated surveillance with minimal human intervention [1]. In order to enable the scalability and the duration of Unmanned Aerial System (UAS[1]) missions, it is possible to collaborate between autonomous teams[2] of UAVs and GCS through an ad hoc wireless network called UAANET. Ad hoc networks are considered suitable for UAV based networks because of their self-forming, self-healing and self-organizing features. Once UAVs have been configured, they can form their network structure with the guidance from the GCS. Thus, the network becomes resilient to eventual failures of nodes. Ad hoc networks have been largely investigated by the research community for a bunch of mobile systems such as sensors, cars, or civil aircraft. However, much of the work carried out in these areas does not take into account some specific features of UAANET (detailed in section 2) which raises some networking issues.

Furthermore, from a security perspective, UAANET routing protocol must also be secured to protect network and data (payload and command & control) traffic from malicious attackers. Hence, control packets need to be authenticated to verify both the identity of the message originator and the message integrity. Security of routing protocols has been widely investigated in wired networks and Mobile Ad hoc Networks (MANETs), but as far as we know, there is no existing work dedicated to UAANET routing protocols with security features. There have been some proposals to ensure the security of ad hoc routing protocols in MANET but most of them are vulnerable against wormhole attack [2]. The wormhole attack involves two attackers who perform a colluding attack. One attacker record packets at a particular location and replay them at another attacker by using a high-speed private network.

In this paper, we present the formal verification of our new secure routing protocol for UAANET called SUAP (Secure Uav Ad hoc routing Protocol). This protocol ensures message authentication, data integrity and provide detection and prevention of wormhole attacks. SUAP is based on SAODV routing protocol [3] and use a public key cryptography, hash chains and geographical leashes. It uses digital signatures for non-mutable fields (e.g. destination IP Address) and hash chains for mutable fields (i.e the hop count). Two distinct mechanisms are combined to provide security against wormhole attacks. The first one is used during route maintenance in which hello and route error packets are sent to reckon one hop neighbors and to detect link breaks. We use a mechanism that mathematically analyzes the correlation between the hop count and the distance traveled by the packet (detailed in section 3). The second mechanism is used during the route discovery process in which each node computes a hash hop by hop by taking into account the respective address of the previous node

[1]A Unmanned Aircraft System is composed of UAVs, communication links, ground control stations, launch and recovery system, and any other system elements that may be required during flight operation

[2]also called UAV swarms

and next hop to which the message is forwarded (detailed in section 3).

Additionally, in order to validate SUAP security properties, we use a formal verification through model checking with Embedded coder formal verification tools [4] and AVISPA (Automated Validation of Internet Security Protocols) tool [5]. AVISPA is a modular and expressive formal language used to verify SUAP security properties. With regards to Embedded coder, it is a code generator containing several integrated model and code consistency verification tools used with Simulink & Stateflow framework to validate the conformance between SUAP high-level models (security requirements) and its associated source code. This latter is performed to contribute to UAS certification as initially presented in our previous work [6].

Finally, we used a hybrid experimental system [7] (based on virtual machines and the Virtualmesh framework to emulate physical aspects) to evaluate SUAP routing protocol effectiveness under a realistic UAANET scenario. We compared SUAP and AODV [8] in a realistic scenario with three UAVs in flight.

This paper is organized as follows. In section 2, we highlight the state of the art and the existing work on UAANET secure routing. Section 3 describes the SUAP routing protocol by specifying the modified structure of route discovery and route maintenance packets along with its implementation methodology. The formal verification with AVISPA and Embedded coder will be detailed in section 4. In section 5, we present our performance evaluation of SUAP protocol with a hybrid testbed presented in [7] to add an overlay of security properties validation. Finally, we draw our conclusions and future works in section 6.

II. Secure Routing For UAANET

A. UAANET state of the art

UAANET is a specific case of UAS. It is composed of UAVs, payloads (e.g. digital cameras) several communication links, ground control stations, launch and recovery system, and any other system elements required during flight operation. The architecture of a typical UAS is shown in Figure 1. For a detailed description of each of those subsystems, interested readers should check the paper in [9]. It is important to emphasize the existence of several kinds of UAVs, each designed for a different purpose. It is possible to categorize them in many ways using different metrics. In this paper, we mainly focus on civil, commercial and research-dedicated UAVs. Military ones are out of the scope of the paper due to their specific requirements.

Moreover, similar to MANETs, the UAANETs architecture is an infrastructure-less network which uses multiple nodes to forward data packets. It also shares characteristics such as self-organized abilities, self-managed information, communications and cooperation between nodes to perform data delivery. However, UAANETs have some specific features that differentiate it to MANET [1] as explained in the following:

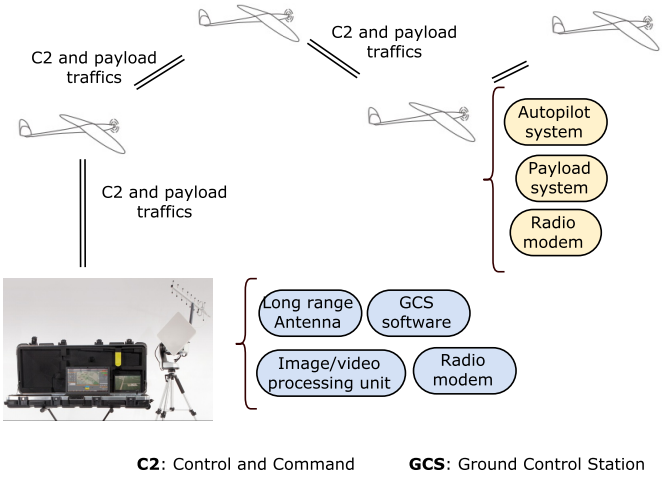


Figure 1 Example of UAS Architecture

1) **Network connectivity**: The intermittent degree of UAANET network connectivity is more important than in MANETs or VANETs [10]. This is mainly driven by the UAV mobility degree, and the UAV mobility pattern. Such communication interruption could be critical when they are transmitting important information (control/command traffic). In addition, UAV failure may cause connectivity failure, which results routing failure, and therefore communication failure or longer delay. Another aspect that affects the connectivity is the connection outages. Due to the UAV movements and variations of distances between UAVs, link quality fluctuates and may cause loss of connectivity and performance degradations.
2) **Number of nodes** : When UAV deployed in a given mission has a relatively high speed, it can be sufficient to cover a restricted mission areas. 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 [11].
3) **Sufficient energy**: Depending on their sizes and types, UAVs are usually assumed to have enough energy and computing power compared to certain nodes in MANETs. This is driven by the fact that the energy needed to move the UAV is much greater than the energy needed to compute data. Similarly, as nodes in Wireless sensor network (WSN) UAVs are on batteries, and certain mission can last long.
4) **Mobility (3D)**: Mobility model plays a significant role in designing network protocols for UAANETs. UAV mobility patterns are a lot different from any other vehicle. A 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). Indeed, several existing

misbehavior detection techniques often rely on nodes position. Thus, a study must be carried out to take into account the altitude information into the misbehavior detection algorithm. Furthermore, note that based on the mission, the UAV movement can follow different types of mobility pattern. It can be straight following a waypoint, circular staying in a specific zone, oval and scan when patrolling around a given circuit.

5) **Strict delay constraints**: Generally, UAANETs are used for real-time applications, such as aerial photography and video capture. Accordingly, the control/command traffic should arrive in time to avoid the loss of control of the whole UAS.

B. Routing protocol for UAANET

Routing is an essential parts of the UAANET communication architecture as UAVs must relay control and data traffics between them to the GCS. It must take into account the UAANET specific features to accurately find routes between UAVs and the GCS. Particularly, a noticeable delay during transmission must be avoided as control traffics flows in real-time between entities. Once a control packet process is delayed, it can induce network instability and an unexpected behavior of the UAVs. Generally, in addition to the requirements present in the generic wireless ad hoc networks (such as: finding the most efficient route, allowing the network to scale, controlling latency, etc.), UAANETs also require some requirements such as: location-awareness algorithms (due to the possibility to perform routing based on geographical information), energy awareness (due to the small size of power unit), and most importantly, a strong robustness mechanism to the intermittent links and constant changing topology features caused by the high speed of UAVs. Thus, the conventional ad hoc routing protocols designed for MANET are not necessarily suitable for UAANET.

Typically, most of existing UAANET routing protocols are an extension of the following well-known MANET routing protocols: AODV, OLSR [12] and DSR [13]. Geographical routing, as surveyed in [14], could also be efficient in specific contexts. The question is now: which of these routing protocols fits UAANET environments. To provide an answer to this question, in [7], we introduced an emulation-based performance evaluation of MANETs routing protocols for UAANETs. This realistic study considers the Linux kernel networking stack requirements, the protocol implementations, the background traffics, the real time execution features and a realistic UAVs mobility model that was deduced from real UAS flights. Also, we used real-time data traffic based on actual data acquired from real UAS experiments. Our results showed that AODV suits better in UAANET compared to OLSR and DSR. Our mobility pattern and emulation system certainly affected the measures, but we found similar results to those exposed in [15]. Accordingly, in [6], we introduce the model design of SUAP (Secure Uav Ad hoc routing Protocol). Also, a first preliminary outdoor experiments have been carried out in [16] to evaluate its network requirements and performances.

C. Security Challenges For UAANET

Generally, in ad hoc networks, the wireless links are prone to link attacks, which consist of passive eavesdropping and active interfering. These attacks are strengthened by the fact that there is not a clear secure boundary within the network, which can be compared with the clear line of defense in the traditional wired network. UAVs can join, leave and move inside the network. Consequently, with a commercial high-gain antennas configured in specific frequencies, anyone can listen to the frequency and receive the signals (e.g. GPS signal) send by the GCS and the UAVs. Moreover, the absence of centralized management infrastructure makes the detection of attacks difficult as it is not easy to monitor the traffic in a highly dynamic network. In addition, due to the intermittent connectivity caused by UAVs mobility, differentiate malicious failures from network disruption caused by the dynamic environment is challenging, especially, if the adversaries modify their attack pattern.

Also, the fact that UAANETs has restricted power supply features also contribute to its vulnerabilities. Indeed, depending on UAVs sizes, UAV can have limited storage capacities which make easier to launch a denial-of-service attacks. An attacker can launch sleep deprivation attack [17] which aims to drain the battery power of a UAV by sending meaningless packets.

Accordingly, to protect UAANET from attacks, the routing protocol must provide a set of security properties to provide an accurate and secure path from the source to the destination. These requirements consist of:

1) Node authentication: only authorized nodes should perform route computations and discovery; minimal exposure of network topology;
2) Message authentication: detection of spoofed and fabricated routing messages;
3) Message integrity: detection of altered routing messages;
4) Avoiding formation of routing loops and redirection of routes from shortest paths.

III. SUAP Routing protocol

There have been several proposals to enhance security in AODV and these include SAODV [3], ARAN [18], SEAR [19], SEAODV [20]. These routing protocols ensure message authentication but fails when facing wormhole attacks. They rely on cryptographic mechanisms to secure routing packets. They prevent forgeries by allowing intermediate nodes to check the packets' authenticity. Specifically, with SAODV, the hop count is protected by hash chains preventing nodes from modifying the hop count value. However, its security features have some flaws as attackers can leave the hop count unchanged to falsify how a path is selected. Generally, the kinds of attacks that are prevented by SAODV are limited to those involving impersonation of nodes or manipulation

of routing control messages, and it remains vulnerable to wormhole attacks.

A. Wormhole Attacks

Wormhole attacks also known as tunneling attack is a colluding attack led by one or two attackers working together to create a wormhole link. One attacker forwards control messages to the second one (usually located several hops away) without changing the packets through a high quality out-of-band link. This means that distant nodes are considering themselves as neighbors since the hop count is not increased. Thus, this attack cannot be detected even with a strong authentication mechanism. Once attackers have successfully attracted routes through them, they can degrade the network performance by dropping or selecting data packets and create a DoS attacks. Figure 2 demonstrates an example scenario of this attack within UAANETs, where A1 and A2 are the colluding attackers while the node N0 and N3 are the victim node. In this example, the GCS send a route discovery packet to node N0 which forward the request to N1. By putting A1 in the vicinity of N0, it can receive the information and replays it to A2 placed next to destination N3. As a result, N0 and N3 assume that they are neighbors.

B. SUAP security features

SUAP is an extension of the SAODV routing protocol that can be used to protect the route discovery mechanism providing integrity, authentication, non-repudiation security services and an additional security features against wormhole attacks through geographical leashes approach.

The following assumption is considered by SUAP network and attacker model:

- Nodes are homogeneous. UAVs and GCS are coming from the same manufacturer (in our case Delair Tech) which allows us to put aside the security issue from selfish nodes. Apart from being captured or controlled by an attacker, a node will not change its behavior and will always cooperate with its neighbors;
- Nodes have sufficient energy power and network resources (i.e bandwidth) to perform cryptographic computation;
- All UAVs utilizes omnidirectional antennas. Communication range is r and cannot exceed Dmax (r < Dmax). Dmax is a maximum one-hop range;
- Nodes rely on efficient symmetric and asymmetric cryptography algorithm for encryption/decryption, authentication and hashing. RSA and SHA-256 algorithms have used respectively for message authentication and hashing. It should be noted that SHA-512 can also be used. Both belongs to the SHA-2 functions which are preimage-resistance and collision-resistance. The hash algorithms in this standard have a secure features as for a given algorithm, it is computationally infeasible to find a message that corresponds to a given message digest. Also it is not computationally unfeasible to find two distinct messages producing the same digest.
- All nodes are clocks synchronized. This is possible with the presence of GPS on-board of UAVs and GCS;
- We assume that there is an efficient and reliable key management within the network to share, to manage and to revoke node cryptographic keys;
- Routing control packet confidentiality is not insured. Indeed, routing packets are processed in real time by the flying UAVs. As such, even if an attacker is able to eavesdrop the message, its action is limited in passive mode because in the future, the past information is no longer valuable.
- Hash function H is only known by legitimate nodes and pre-loaded with keys at the bootstrapping. The selected function is included in route discovery packet and signed with a digital signature. This leads to the protection of the hash function from attacker modification. In order to breach the hash function, any attacker must hack the digital signature. Attack on cryptography algorithm is not considered in our research.
- In regards to attackers capabilities, we consider the work of Cordasco et al. in [21] based on Dolev-Yao model in which they present a topology and protocol agnostic model that takes into account a real-world scenario. Accordingly, we considered that the following attacks are possible: data traffic disclosure, routing information disclosure, performance degradation and topology modification.

By taking into account those requirements, we propose the SUAP routing protocol which is a security extension of the SAODV protocol, based on public key cryptography, hash chains and geographical leashes [22]. It uses digital signatures for non-mutable fields and hash chains for mutable fields (i.e the hop count). A node that generates a routing message signs it with its private key, and the nodes that receive the message verify the signature using the sender's public key. The hop count cannot be signed by the sender, because it must be increased at every hop.

Therefore, a mechanism based on hash chains is used. A hash function maps a variable-length message into a fixed-length hash value, or message digest. This function has the property that the results of applying the function to a large set of inputs will produce distributed and random outputs. Virtually, the hash function use an iterative use of a compression function such as SHA [23]. Particularly, hash chains are based on one-way hash functions. A hash chain of length N is constructed by applying a hash function "h" N times to a random value "seed".

$$\left\{ \begin{array}{c} h_i(Y) = h(h_{i-1}(Y)) \\ h_0(Y) = seed \end{array} \right\}$$

Intrinsically, the routing protocol is still vulnerable against wormhole attacks. Accordingly, a version of geographical leashes based security algorithm is used to estimate the correlation between the traveled distance and the hop count value. In order to do so, SUAP requires each node in the network

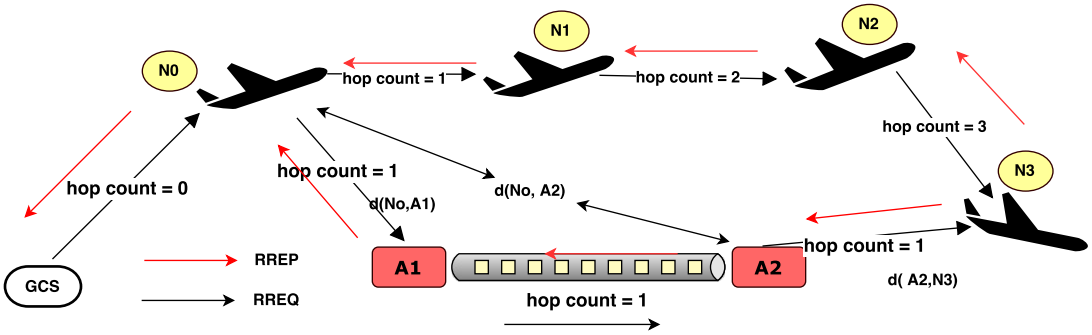


Figure 2 Illustration of Wormhole Attack in UAANETs

Table I Notation Table

Paremeter	Description
hc	Hop count
Dmax	One hop distance maximum
Rij	Distance between nodes i and j
c(i,j)	connectivity state between node i and node j
d(No, A1)	Distance between the first target and the first attacker
d(A1,A2)	Distance between the two attackers
d(No, A2)	Distance between the first target and the second attacker
d(A2,D)	Distance between the second target and the second attacker
T	The total distance of the legitimate route
Dw	The total length of the path through the wormhole link

to be tightly synchronized and maintains a local connectivity with its direct neighbor. It should be noted that, when it comes to node synchronization, a synchronization upper bound and lower bound must be set up. This should be measured in our upcoming real world experiments.

C. Enhanced Beacon Message

Beacon message is sent by broadcast to one-hop neighbors to maintain the local connections updated. Our objective with SUAP is to protect these packets from wormhole attacks. Hence, besides signing all the data fields, we use a mechanism that analyzes the correlation between hop count and distance traveled by the packet. When sending messages, each node includes its actual location information. To protect from malicious modification, message fields are signed (including the geographical position).

Figure 3 illustrates the format of modified beacon messages.

To illustrate our proposition, we considered the Figure 2 and the notation in table I. The connectivity between two legitime nodes can be expressed by (1), with $R_{i,j}$ is the current communication range.

$$c(i,j) = \begin{cases} 1 & if \quad R_{i,j} \leq Dmax \\ 0 & if \quad R_{i,j} > Dmax \end{cases} \tag{1}$$

The presence of wormhole link modifies this condition to (2).

$$c(i,j) = \begin{cases} 1 & si \quad R_{i,j} \leq Dmax \\ 1 & si \quad R_{i,j} > Dmax \end{cases} \tag{2}$$

Furthermore, we have :

$$d(No, A1) \leq Dmax$$
$$d(A2, N3) \leq Dmax$$
$$d(No, A2) > Dmax$$
$$d(A1, N3) > Dmax$$

It results that.

$$d(No, A1)^2 + d(A1, A2)^2 > Dmax^2$$

thus,

$$d(A1, A2) > Dmax - d(No, A1)$$

We have (3) :

$$Dw = d(A1, A2) + d(No, A1) + d(A2, N3)$$

$$Dw > Dmax \tag{3}$$

Knowing that

$$T = \sum_{i=0,j=0}^{n} R_{i,j}$$

- When the node N0 send the packet, $To = Ro1$ that corresponds to hc = x + 1 with $x \in \mathbb{N}$;
- When the node N1 send the packet, $T1 = To + R12$ that corresponds to hc = x + 2;
- When the node N2 send the packet, $T2 = T1 + R23$ that corresponds to hc = x + 3;

thus,

$$\frac{T}{Dmax} - 1 \leq hc < \frac{T}{Dmax} + 1 \tag{4}$$

By taking into account the inequality (3), we can compare the hop count value present in the packet and the hop count value computed on the traveled distance by following the corresponding value depicted in the Table II and the formula (4). If there is a difference, the wormhole link is detected, and the packet is rejected. Otherwise, the link is considered as free

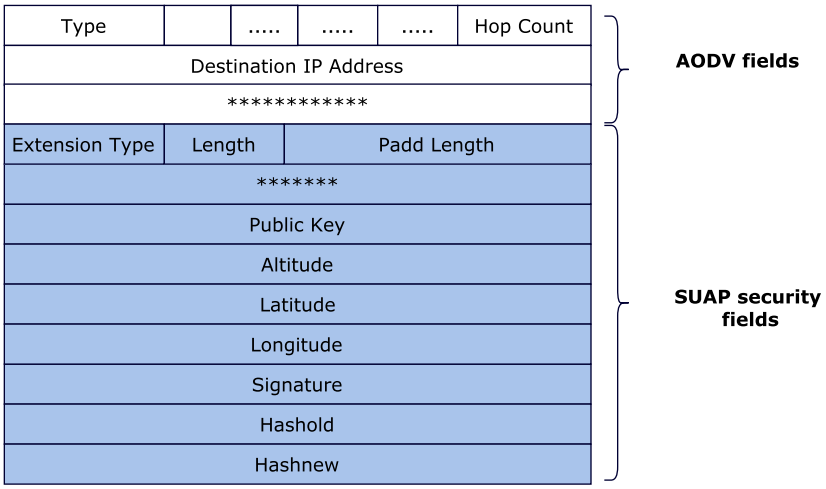


Figure 3 Format of SUAP Modified Beacon Message Extension

Table II Mapping Table Between the Distance Traveled and Hop Count

T	Hop count (hc)
$0 < To \leq Dmax$	0
$Dmax < T1 \leq 2Dmax$	1
.....	.....
$(n-1)Dmax < T_{n-1} \leq (n+1)Dmax$	n-1

of the wormhole attack and the signature verification process begins. Note that because of the position information included in the packet, it is possible to compute the relative distance between 2 neighbor nodes using 3D Euclidean distance.

D. Securing Route Discovery

In order to implement a hop-by-hop authentication, each node must verify the incoming message from its one-hop neighbors before sending it by unicast to its neighbors. Each node must ensure that the packet is authenticated and was not forwarded through wormhole link. Thus, each node should make sure that it has a trust relationship with its neighbors. Such secure relationship between each pair of nodes relies on the exchange of beacon messages between neighbors as explained previously. In SUAP, the route discovery process is similar to that of standard AODV, but two hash extension is appended to the end of route discovery packets as explained in the following.

During the route discovery process route request and response are exchanged. In this mechanism, nodes do not need to send its geographical position. Instead, each node assumes that its local connectivity is secure thanks to the neighbor information provided by the previous mechanism. Each node then sends in unicast all route discovery packets to its direct neighbors. Each node also includes the address of the next hop to which the message is forwarded and apply a hash chain to the packet mutable fields. The non-mutable fields are signed as stated previously. An illustration of the request message is shown in Table III. The source node appends its own address and the next node address to the hash chain called Hashnew. It also includes the Hashold (which is the previous Hashnew) within the packet. When an intermediate node says Ni receives a request, it checks its signature and verify the Hash chain. It re-computes the Hash chain with H [previousnode, MyIPAddress, Hashold] and verify if it has the same result as the one included in the packet.

Table III SUAP Request Packet Signature Extension Fields

Field	Value
Type	64
Hash function	hash function selected by the sender node. It is used to compute the hash chain field
Signature	The signature of all the non-mutable fields
Hashnew	Hashnew = H [CurrentNode, NextNode, Hashold] CurrentNode is the address of node sending the request packet. It can be its public key or its IP Address. The Nextnode is the next node public key or IP Address. Hashold is the previous chain element received from the previous node
Hashold	It is the previous chain element received from the previous node. When receiving packets, nodes change the value of Hashnew into Hashold
Hop Count	The actual hop count of the packet. It is the number of times the hash is performed

Considering the Figure 2, we compute the hash chain as the following:

The GCS node executes the following operation:

- Select a H hash function;
- Compute $Oldhash = H(seed)$, seed is a value selected randomly by the sender;
- Compute Hashnew = H(GCS, N0, Oldhash), N0 is the next node address;
- Compute message S to N0: [64, H, signature, Hashnew, Oldhash]

When the node N0, receives the packet, it processes the following operation:

- Integrity verification by computing Hashverifier = H [previousnode, actualnode, Oldhash] and verify the result compared to Hashnew. Since, we use a one way hash function, the slightest change would lead to difference. Besides, the hash function is only known by legitimate node.
- If Hashverifier = H[GCS, N0,Oldhash] = Hashnew, it indicates that the link is free of wormhole attack. Otherwise, it means that the packet has been transmitted via a wormhole link. The packet is then rejected.
- Assign the new Oldhash = Hashnew;
- Compute the new Hashnew with Hashnew = H[N1, N2, Oldhash].

The operation is repeated until the packet reaches the destination. The same mechanism is also used for the response packet. As regards the exact value of the hop count values, it can be inferred from the number of times that the hash was used for verification. It can also be included in the hash chain computation. Note that this mechanism can also be efficient against man in the middle attacks in which malicious nodes tries to breach route discovery mechanisms by forwarding control packets from one point to another.

E. Security Block Design Architecture and Model Checking

UAS needs to be certified to have an airworthiness approval in order to be used in real world deployment. Accordingly, on a regional French scale, our UAS communication system needs to be validated by the French UAV professional civil federation. Although a specific validation and certification standards for UAANETs is yet to be shaped, the process and safety standards followed by the aeronautical industry for embedded software design can be applicable as explained in [24]. Accordingly, several standards have to be met. The most closely related to our topic is DO 178C Europe version EUROCAE ED-12C [25] and DO 331 [26]. On the one hand, the DO 178 standard put emphasis on model driven approaches that can generate high-level software code through high-level models (which represent the features of the final system) as inputs. It also recommends the use of formal methods along with the methodology to contribute to the validation of the final system. On the other hand, the DO 331 standard deals with tools used to generate software code. In this standard, the RTCA[3] introduce the use of three different advanced verification methodologies: model checking, formal proofs and code assertions [27].

Model-driven development methodologies (i.e. MDD) claim the use of models as primary artifacts in the development process. A system model composed of block diagrams and state charts is the focus of the development process, from requirement specifications to simulation testing and integration. When used with a code generator, it generates a high level code which facilitates the early verification of the design through formal verification tools. It also execute model-and-code consistency checking for system verification purposes.

[3]Radio Technical Commission for Aeronautics

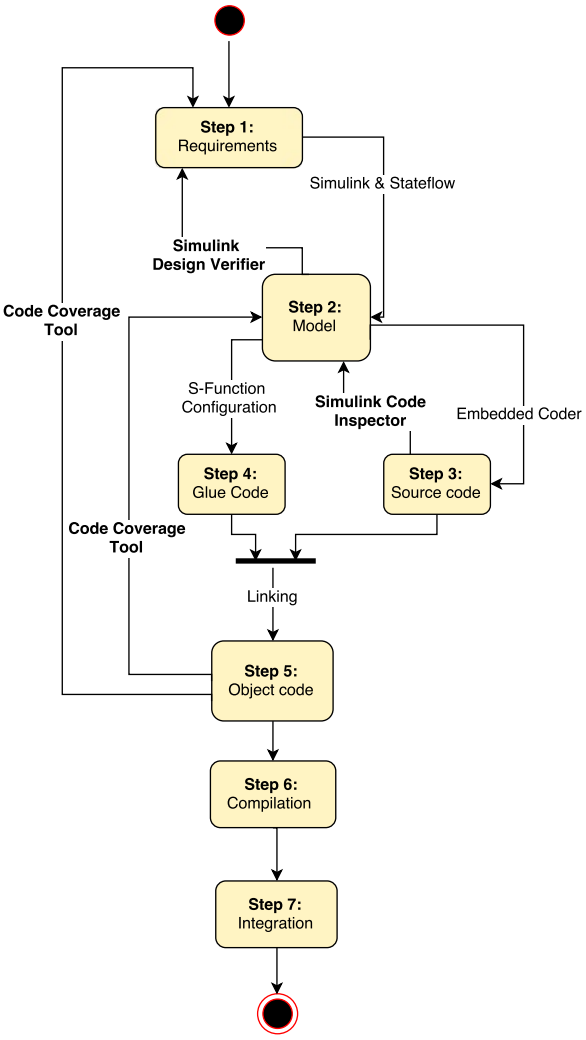


Figure 4 Set of MDD Tools Used to Design This Secure Routing Protocol

Accordingly, our design is based on Matlab Simulink and Stateflow frameworks. Simulink is composed of a block-diagram environment which allows us to accurately design the routing protocol while the stateflow framework allows us to define a finite number of states in the algorithm which are changed based on a defined condition. Figure 4 represents the model driven development workflow and schematizes the integration process. It is composed of the seven different steps as explained in [6].

In order to specify our specification into high-level models, we use a dedicated security design architecture for the security part as depicted in Figure 5. To have an idea of the routing architecture design, we advised interested readers to check our paper in [16]. The specification follows the IETF draft [28] in which each requirement (messages, tables, parameters, extension specification) has been meticulously respected. Our system includes several blocks transmitting signals between them denoting the security and network requirements spec-

ifications. They also contain several instances of stateflow graphic representation. These blocks describe the possible protocol behavior as will detail in the following. To give a general idea of the complexity of the SUAP model specification, we present in the Table IV some significant metrics of the global system.

Table IV Metrics of the SUAP Protocol Specification

Code lines	6236
Blocks	30
Procedures	188
States	30
Signals	36
Macro definitions	14

In the security architecture design, each packet received from the link layer must be verified by a module called "Packet identifier" to check whether or not the packet has a security extension. In case, it does not contain a sufficient security extension, the packet is rejected by the Packet rejector module.

The packet rejector module role is to delete suspicious packets. If the packet has its security extension activated, its content is extracted from the Content extractor module. This module explicitly identifies which part contains the signature, the hash chains and the location information for the wormhole detector. Then, the packet is verified by the wormhole tester module which mainly compute either the association between the hop count and the distance traveled by the packet (if beacon and route error messages are exchanged), or compute the actual hash chains for the message. In this step, if the packet fails to prove its required security level against wormhole attacks, it is redirected to the packet rejector module. Otherwise, it is sent to Authentication and Integrity tester module. In this block, the authentication of non-mutable fields and integrity of mutable fields is verified. If all the information within the security extension is valid, the packet is directed to the routing module to check whether or not the packet has reached its final destination.

IV. SUAP Formal Verification Analysis

A. The AVISPA tool

AVISPA is an automated formal verification tool used to verify the security properties of secure routing protocols. It consists of various modules as shown in Figure 6. It specifies a security problem associated with one or more security properties in the High-Level Protocol Specification Language HLPSL. The HLPSL is an expressive role-based formal language which allows specifying flow patterns, data structures, intruder models, complex security properties and various cryptographic primitives along with their algebraic properties. Furthermore, the AVISPA tool also comprises a low-level specification language, called Intermediate Format IF. It contains infinite-state transition systems which will be given as input to the various back-ends of the AVISPA tool. These back ends implement a process of automatic analysis techniques in order to find anomalies by simulating a hidden attack on the input protocol.

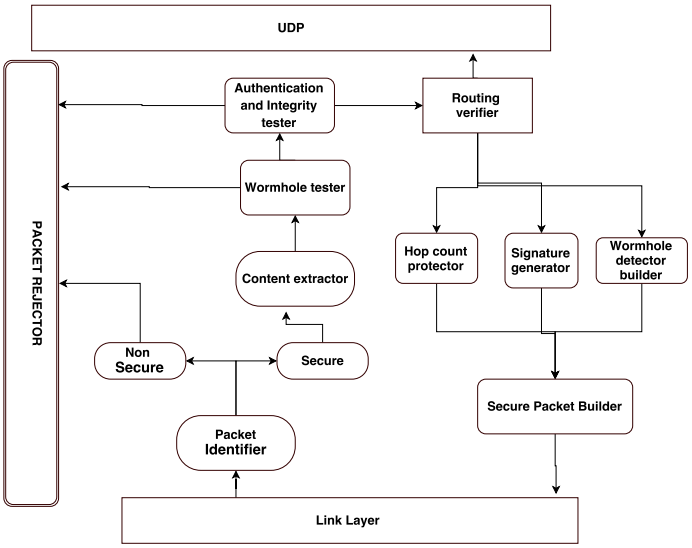


Figure 5 Secure UAANET Routing Protocol Design Architecture

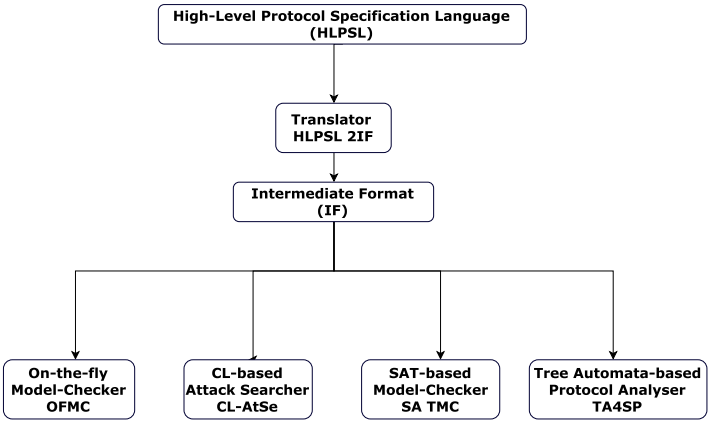


Figure 6 Architecture of AVISPA tool

It should be noted that the security property verification is processed under the assumptions of perfect cryptography and that the network is under the control of a Dolev-Yao intruder [29]. All back-ends analyze a protocol by considering the asynchronous model of an active intruder who controls the network but cannot break cryptography keys. Upon completion, each back-end outputs the result and states whether or not the protocol meets his security requirements. The output gives a description of the protocol goal and in case of a security violation, presents the related attack trace.

In our case of SUAP analysis, as we will see in the next section, the AVISPA tool has not revealed any attacks on message authentication but it has discovered attacks on confidentiality properties by means of back ends On-the-fly Model Checker OFMC. However, confidentiality attacks of network control packets are not critical within UAANETs as network control packets do not contain sensitive information that could be used later by the attackers. They are executed in real time and as such does not have a significant value

when processed later in time (on the contrary of payload traffics). Furthermore, through AVISPA, we were also able to demonstrate that even if the attacker does not have the cryptographic keys, a wormhole attack can be launched into the network. Note that in this paper, we only consider some of the most common execution scenarios of SUAP, consisting of a finite number of protocol sessions executed in parallel. For instance, the scenario in which beacon messages containing location information is not expressed.

B. Analysis of SUAP

In order to verify SUAP using AVISPA, it has to be modeled using HLPSL specifications. All nodes within UAANET that executes the same actions are grouped together within basic roles. A basic role is a module in which can be specified what information a class of nodes can initially use (as parameters of the role), their initial state, and the ways in which this state can change. Note that the network is seen as a collection of agents. The behavior of each of these agents is defined within a basic role module. These agents exchange messages through variables of type channel. Since AVISPA consider a Dolev-yao model for attacks modeling, this implies that channels have no protection and that intruder can have complete information about the network topology.

Our HLPSL specification of SUAP formalizes three protocol roles that can be considered as equivalence classes between nodes having the same behavior for sending and receiving messages: a source node **A**, a destination node **F**, and three intermediate nodes **B, C, D** in which **B** communicates with **A** and **C**, **C** then communicate with **B** and **D**, and **D** communicates with **C** and **F** as depicted in Figure 8. The node **A** send a route request to node **F** through node **B**, **C**, **D**. The message contains mutable fields and static fields. Since AVISPA does not allow the modeling of mutable elements, all the data are considered static throughout the packet lifetime and signed with the signature of the sender node. The notion of hop count update is specified with oldhash and newhash fields which are always modified before packet forwarding. For illustration purposes, the HLPSL code for the source node is depicted in Figure 7.

The properties of the SUAP specification that we have to analyze are the following:

- Authentication of NA, Tophash and F during the request phase: the receiving node checks (Na, f) to verify if that is already processed this request.
- Authentication hop by hop of Hashnew : each neighbor node compute a Hashverifier and compare it to Hashnew
- Authentication of ND and A in the reply phase: the receiving node checks (ND, A) to verify that it has not already processed the response.

We have formalized two HLPSL specifications, representing two different network topologies corresponding to two attacks. The first contains an attacker that can launch black hole while the second illustrates a wormhole attack and its effect.

The authentication verification is both end-to-end and hop by hop, checked by each node during route discovery process. Our HLPSL specification formalizes all this as follows. We simulate both the request phase and reply phase: the source node **A** starts the session and send a message containing the packet identifier, data payload, Hashnew and hashold and the top hash. This message is forwarded to nodes **B, C** and **D** to reach **F**. Similarly, **F** sends a reply that reaches **A** through **B**, **C** and **D**.

When a node receives a request packet from another intermediate node, it will only validate the certificate and the signature of the previous intermediate node. Eventually, the packet will arrive at the destination node. This node will validate the certificate of the source node and its signature. Then the destination will verify if it had not already processed the request packet by looking to the tuple (NA, A). If the message was already processed, it is discarded, otherwise, the destination processes the packet and generates a route reply that contains the packet identifier, the IP address of the source from which the corresponding request had arrived, the certificate of the destination node, a timeout value, a seed, and a hash value. All those information is signed with the private key of the destination node. Afterwards, it sends the packet by unicast to the node from which it has received the associate request packet and also verify if these nodes are free of the wormhole attack. The intermediate nodes will unicast the packet hop by hop to reach the sending node. Their role is to validate the certificate and verify the signature.

C. Analysis of the first HLPSL specification of SUAP

The analysis with the AVISPA tool of our first HLPSL specification of SUAP prove that message authentication is ensured. However, we also found that SUAP suffers from spoofing attack, where the intruder pretends to be a valid intermediate node. The node **A** send a secure packet by broadcast to node **B** which is intercepted by the intruder **I**.

1) $A \rightarrow I : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA$
2) $A \rightarrow B : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA$
3) $I \rightarrow B : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA$
4) $B \rightarrow I : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA, certB$

The intruder **I** relay the request to node **B**, who thinks that the intruder **I** is the node **A**. Note that **I** can move from his initial position to be in the vicinity of **B** and **C**. As such, the intruder **I** can overhear the message and send it to **C**. **I** can perform the same operation during reply phase. When the intruder executes the sequence, an incorrect routing state attack can be launched. This attack can be categorized as rushing attacks in which the intruder was able to rush its control packets to disrupt the path selection algorithm.

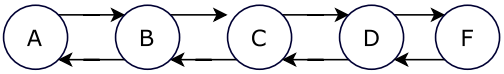
Furthermore, the SREQ fields indicate the non-mutable fields of the control packets while the notation F is the destination node. The notation *inv(Ka)* indicates the private key used by the sender node **A**, and finally, Hash and cert illustrate respectively the hash and certificate fields.

Moreover, our results also show that entities that are in the vicinity of valid nodes can easily intercept the message.

```
role sourceNode (
A,B,C,D,F: agent,
Ka, Kb, Kc, Kd, Kf : public_key,
Hf : function, %Hash function
KeySet : (public_key) set,
RCV, SND : channel(dy)
)
played_by A def=
local
State : nat,
REQstatic, REPstatic : text,
SREQ, SREP : protocol_id,
Hashnew, Oldhash, Na, Nd : message %Na, et Nd a enlever peut être, KHC = Tophash%%
init
State := 0

transition
step1.
State = 0  /\ RCV(start) =|>
State':= 8 /\ Na' := new()
 /\ Hash_new' := new()
 /\ Hash_new' := Hf(A,B,Oldhash)
 /\ SND(SREQ.REQstatic.Na'.Oldhash.Hashnew'.{Na'.SREQ.REQstatic}_inv(Ka))
 /\ witness(A,B,Na,Na')
step2.
State = 8 /\ RCV(SREP.REPstatic.Nd'.Oldhash.Hashnew'.{Nd'.SREP.REPstatic'}_inv(Kb).H) /\ in(Kb, KeySet)/\ Hashnew'=Hf(A,B,Oldhash)=|>
State':=9 /\ wrequest(A,B,Na,Na') %weak authentication%
end role
```

Figure 7 HLPLS Source node role code



A ----> B: { SREQ, NA, A, id}_inv(Ka),
oldhash, newhash}

B ----> C: { SREQ, NA, A, id}_inv(Ka),
oldhash, newhash}
.....
D----> F: { SREQ, NA, A, id}_inv(Ka),
oldhash, newhash}

B ----> A: { SREP, NF, F, id}_inv(Kf),
oldhash, newhash}

Figure 8 The SUAP Protocol (an example with five nodes

This implies that SUAP does not take into account the routing packet confidentiality. Nonetheless, during communication between UAVs, for instance, during surveillance mission, network control packets do not contain sensitive information that could be used later by the attackers. Their encryption is thus unnecessary. However, some messages from the application layer such as c2 and setting traffics must be kept confidential by way of encryption schemes. Confidentiality is mostly achieved by using public or symmetric key encryption to ensure secure communications.

D. Second HLPSL Specification of SUAP and Its Analysis

To consider wormhole attacks, we created a second HLPSL specification of SUAP protocol. Accordingly, we created a chain of collaboration between intruders to develop a parallel path from the source node **A** to the destination **F**. Consequently, since the intruders are not required to perform cryptographic operations, the request messages may reach the destination faster than in other authorized network path. This operation can be executed within only a portion of the network. The same operation is executed during reply phase.

To model this tunneled attack, we formalized a second HLPSL specification in which, as depicted in Figure 9, we added a polymorphic role that we called node **J**. **J** can reach three nodes that are geometrically aligned. It breaches the normal execution of the protocol to simulate intruder tunnel.

1) $A \rightarrow I : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA$
2) $A \rightarrow B : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA$
3) $I \rightarrow J : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA$
4) $J \rightarrow D : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA$
5) $D \rightarrow F : \{SREQ, F, Na'\}\, inv(Ka), Hash, certA, certD$
6) $F \rightarrow D : \{SREP, A, Nd'\}\, inv(Kf), Hash, certF$
7) $D \rightarrow J : \{SREP, F, Nd'\}\, inv(Kf), Hash, certF, certD$
8) $J \rightarrow I : \{SREP, F, Nd'\}\, inv(Kf), Hash, certF, certD$
9) $I \rightarrow A : \{SREP, F, Nd'\}\, inv(Kf), Hash, certF, certD$

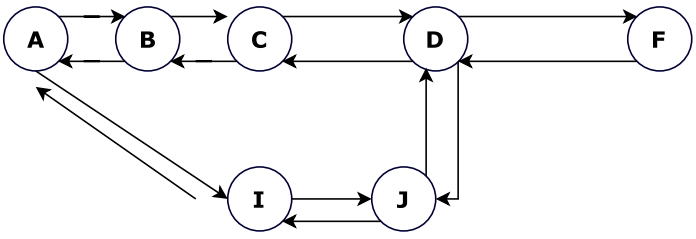


Figure 9 Specification of wormhole attacks with SUAP

Specifically, when the intruder **I** receive a message from **A**, it starts the attack by making use of the intruder **J** (other polymorphic role of the intruder **I**) to simulate the intruders

tunnel. **J** sends then the exact similar packet to node **F**. When receiving the packets, **F** assume that **A** is its neighbor, it then sends an identical packet to **A** on the same path. Node **F** then sends the packet to **J**, which forward it to its intruder associate **I**, which can now send the packet to node **A**. As a result, the path is then constructed through nodes **I** and **J**. Upon termination, the intruders can decrease the network performance by selection routing control packets or deleting some of them.

As described previously, SUAP adds geographical leash properties to check if a wormhole link exists within the network. It is because of this additional information added to the data packets that SUAP is robust against wormhole attacks. Expressing our mechanism into HLPSL specification was not worth it since AVISPA does not allow integer operations. Consequently, we have modeled this property through Simulink and Stateflow framework. We were able to model and generate code through embedded coder. A model-and-code consistency checking through formal verification tool has been performed.

As shown in Figure 4, simulink code inspector examines blocks, state diagrams, parameters, and settings in the model to determine whether they are structurally equivalent to the operations, operators, and data in the generated code. Then it generates a traceability documentation with respect to the EUROCAE ED-12C reference document that can be used for certification purposes . Regarding simulink Design Verifier: it identifies model design errors. It detects, blocks that lead to errors, such as dead logic, integer and fixed-point overflows, division by zero. Finally, with regarding to model coverage tool, it gives line coverage information. It indicates which part of the model is depicted by a given line of code. This is useful when an error has been identified by the Simulink Design Verifier to locate which part of the model is faulty.

As we can see in the Figure 11, we explicitly validate our wormhole attack security solution through model checking by using simulink and stateflow properties.

Those two figures illustrate that our modeling went well. Our code and model objectives are joined. The Figure 10 indicates the model verification. It reports that all the block models and charts diagram are properly diagram and not faulty. The Figure 11 indicates the code generation report of our high-level models. Note that this high-level models represent the different features and behaviors of the SUAP protocol as depicted in Figure 5.

One important feature of Embedded coder is that if the high-level models contains design errors in which the formal verification tools testing fails, it does not generate software codes and generate errors instead. In our case, all the verification processed by the set of formal verification tools mentioned in the Figure 4 went well. It reports that embedded coder was able to generate a high level code for the input model requirements. This proves that our design is validated and respond to our security specific needs. To validate its effectiveness, the performance evaluation is carried out and demonstrated in the next section.

V. SUAP performance validation

A. Testbed Architecture

In this section, we will evaluate SUAP performances. It should be noted that there have been some studies [15] conducted to measure routing protocols within UAANETs. Most of them are simulation-based and as such do not take into account the inherent features of UAS embedded systems and software[4]. Consequently, the simulation-based performance evaluation may not be the best solution in UAANET paradigm since it hides several important parameters due to the lack of OS-based implementations. These limitations could induce significant differences between simulations and real test-bed results especially in our case in which we have to deploy our software based routing protocol in real flight outdoor experiments. Accordingly, we use a hybrid experimental system to combine the low cost of a simulation with the accuracy of a real protocol stack. We use virtual machine implementations to deal with the entire complexity of Linux operating system. The traces used to generate UAVs mobility patterns were extracted from real traces so that physical related factors could be as realistic as possible. 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, measurement tools and a framework to allow virtual machines to communicate through a virtual wireless medium. We chose to use VirtualBox [30] as a virtualisation tool because it is an easy-to-use and efficient hypervisor. The virtualized system is a 12.04 version Ubuntu, working with the 2.6.38 version of the Linux kernel. An illustration of this system is depicted in Figure 12.

B. The virtual mesh framework

Our testbed architecture uses a Virtualmesh framework that has been proposed initially by [31]. It is a framework which interfaces a Linux-based system with an OMNeT++ [32] simulation. Omnet++ is a powerful network simulator which simulates several systems and normalized protocols. Using Virtualmesh could be summed up in the following steps :

- A virtual wireless interface is created on the Linux system we want to include in the simulation;
- The OMNet++ simulation is launched (which has to include some modules supplied by the framework);
- 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

[4]Such as the Linux kernel networking stack, the network protocol implementation or the real time execution of background traffic.

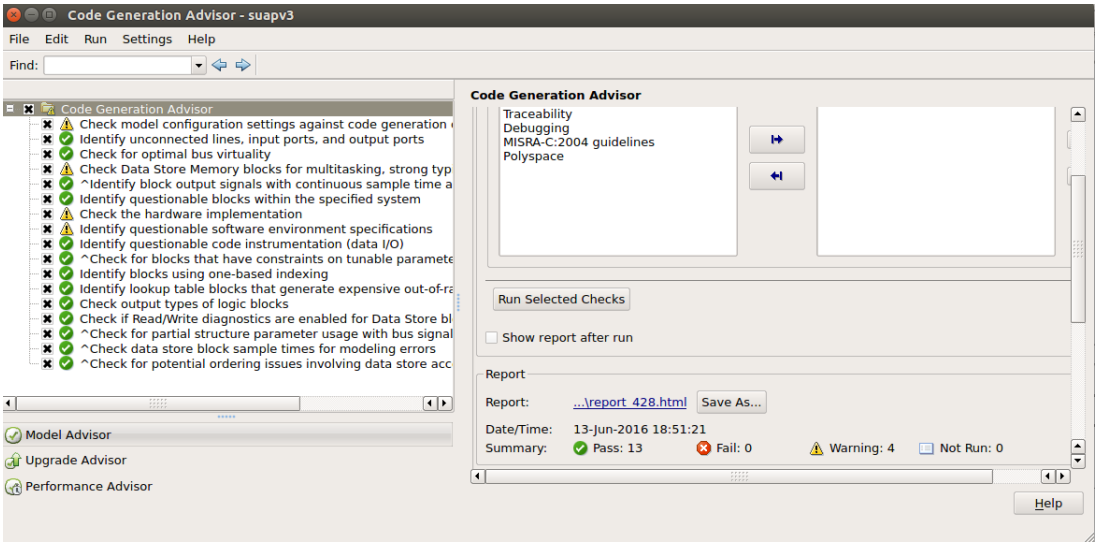
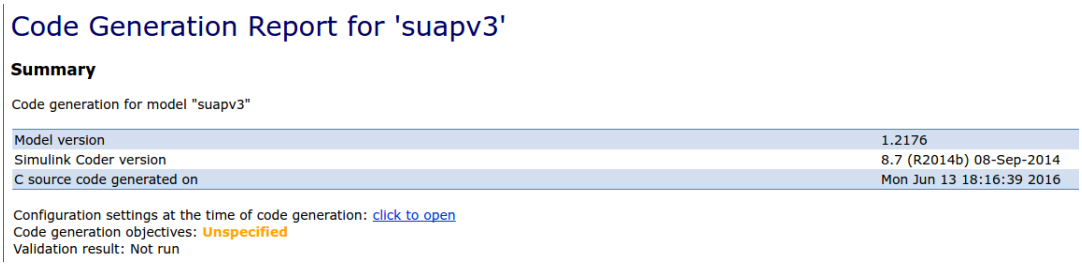


Figure 10 Design verification check with Embedded coder



Code Generation Report for 'suapv3'

Summary

Code generation for model "suapv3"

Model version	1.2176
Simulink Coder version	8.7 (R2014b) 08-Sep-2014
C source code generated on	Mon Jun 13 18:16:39 2016

Configuration settings at the time of code generation: click to open
Code generation objectives: Unspecified
Validation result: Not run

Figure 11 Design verification check with Embedded coder

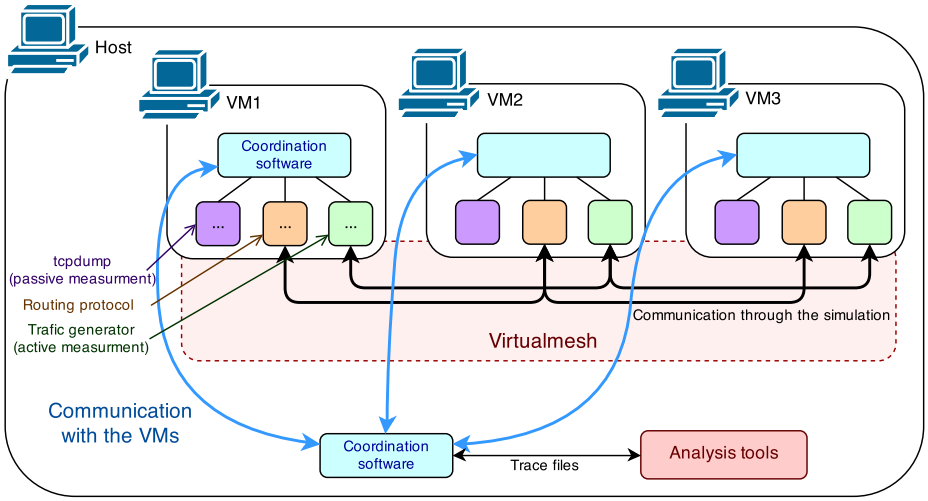


Figure 12 Testbed implementation

simulation is relayed to the virtual interface. An illustration of this system is exposed in Figure 13.

Furthermore, in order to test protocols 'performances, we wanted to generate a realist traffic. Using traces supplied by Delair-Tech[5], from a mission with only three UAVs, we extracted control traffic characteristics. As we supposed a

[5]This is the French company we work with for the Secure Uav Ad hoc NETwork (SUANET) research project, more information available on http://www.delair-tech.com.

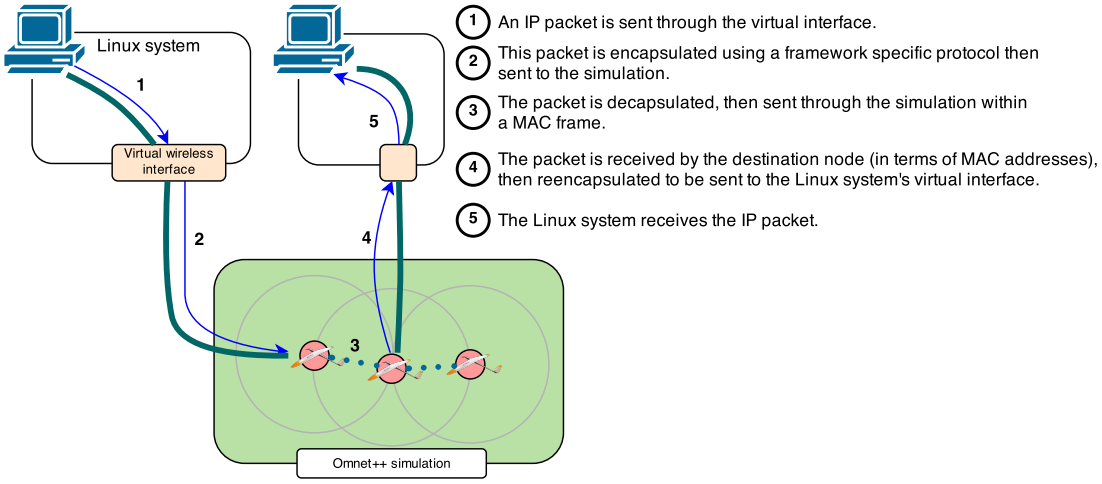


Figure 13 The virtual mesh functioning

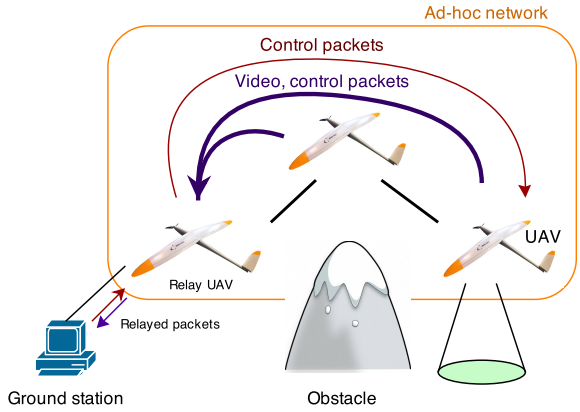


Figure 14 Our Video Surveillance Scenario

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 supposed to be 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 (these characteristics were extracted from a promotional video from Delair-Tech including a lot of images taken by in-flight UAVs). 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.

In order to evaluate routing protocols, we create an experimental test according to what could be a real UAV swarm mission. It consists of three drones scanning an area for video surveillance purposes. We suppose that an obstacle blocks traffics from two of the drones to the ground station, so the third one should be responsible for relaying packets. This scenario is illustrated in Figure 14.

Regarding mobility model, we use a realistic mobility scenario through real traces supplied by Delair-Tech, corresponding to three UAVs scanning a specific area.

C. Evaluation scenario

In this section, our objective is to assess the security properties of SUAP that was not verified with AVISPA tool. This security particularly concerns the authentication of mutable fields which changed every time an eligible node processes the packet. For instance, it was not possible with AVISPA to check the hop count authentication. Furthermore, we also want to go further in the verification by assessing the control packet end-to-end delay both with a non-secure routing protocol and SUAP.

Accordingly, we simulate blackhole attacks within UAANET environment[6]. In blackhole attack, an attacker breach the route discovery process by forging false routing packets to intercept data packets exchanged between neighbors. It then drops some or all data packets instead of forwarding them to the next node. The ultimate objective of this attack is to degrade the routing performance which often results in a very low packet delivery ratio. The evaluation objective is to analyze the behavior of AODV routing protocol and SUAP routing protocol under this attack. The analyzed parameters are shown in Table V. We examined the effects of black hole attack on UDP traffic. Our Linux implementation of SUAP is designed through MDD approach. Libgcrypt encryption library [33] is used for digital signature creation and hash chain generation. For the purpose of securing non-mutable fields, RSA algorithm [34] was used while SHA-1 [35] is used for hash chain generation.

During the emulation, we evaluate the following network parameters:

[6]Because, the implementation time of wormhole attack and security solution within OMNET++ was too important and complex, we decided to simulate only blackhole attack which is already enough to assess authentication and integrity services.

Table V Generated traffic

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	

1: UAV1; 2: UAV2; 3: UAV3.

- Packet Delivery Ratio (PDR): the ratio of packets that are successfully delivered to a destination compared to the total number of packet that has been sent. This gives us an idea of how successful each protocol is in delivering data packets.
- End-to-End Delay (EED): the average time taken for a packet to be transmitted across the network from the source to the destination.
- Routing overhead: the size of routing packets required to build routes. This allows us to measure the quantity of control packets exchanged to maintain connectivity. It also indicates the total amount of secure overhead added onto the routing.
- Loss rates: the rate of packets lost when a route cannot be established. It gives us an idea of how much data packet is lost during breakages. If the routing protocol is able to recover rapidly, the loss rate is not significant.
- Connectivity rates: the percentage of connectivity during the whole communication duration. It indicates the stability of each routing protocol to keep routes alive.

D. Evaluation Results

In order to evaluate the security properties of our secure routing protocol, we executed a single and a collaborative blackhole attacks [36] both against AODV routing protocol and SUAP routing protocol. We remind that blackhole attack is a sharp attack executed against ad hoc routing protocols. In a single blackhole attack, a malicious node forge routing control packets and advertise for an active route to a specific destination. It then drops all received data packets without forwarding them. The attack can be more severe when two or more blackhole nodes cooperate with each other to create a DoS and to degrade network performances, this is called collaborative blackhole attack. Blackhole attack is a less complex attack than wormhole already introduced in this paper to justify and design the security mechanisms embedded in the routing protocol. However, by simulating blackhole attacks we are able to validate and verify the different additional security mechanisms introduced in this paper.

Table VI illustrates experiment results over a one-hour test confront to a single blackhole and a collaborative blackhole attack.

- **Average loss duration:** the effect of a single black hole attack clearly affects the loss burst duration. With AODV, the route loss lasts approximately 7 seconds with single black hole and 15 seconds with collaborative black hole, which indicates that during this period, there is no communication between the GCS and the UAVs. This is caused by the omnipresent malicious agents within the network. From time to time, the attacker leaves the vicinity of the GCS which allows data packets from the GCS to be exchanged on the legitimate route. Regarding SUAP, it behaves correctly as delays narrow 2 seconds for single blackhole attack and 5 seconds for collaborative attack. Typically each time a packet is sent, it is authenticated. As a result, nodes do not send data packets to non-authorized nodes. Consequently, this 2 s and 5 s delay is not caused by the attacks but caused by the dynamic nature of UAANET environment which breaks UAVs connectivity uniformly.
- **Loss rate:** because of the previous metric, we notice that the loss rate is very important with AODV. All route discovery and route maintenance packets are not processed by the intended destination or neighbors nodes. The attacker which is considered as regular node drops all data packets received. On the contrary, because SUAP data packets are authenticated, they do not suffer from a long disconnection. Hence, the loss rate is acceptable which merely corresponds to the loss caused by UAVs mobility.
- **Connectivity during unstable states:** we extracted connected and disconnected states on the links GCS-UAV2 and UAV2-UAV1. To prevent short unstable states to disturb the measurement, two losses that are too close in time (less than 0.1 s) are merged. We extracted what we called ”unstable states” in which, we took away states during which connectivity or non-connectivity were stable for each protocol. We only focus on connectivity or dis connectivity that happens intermittently. Our objective is to assess the protocol behavior in those cases. This unstable state extraction is shown in Figure 15. As we can see in Table VI, SUAP stands out from AODV making connectivity up to 90 % and 88 % compared to 9.8 % and 7.7 %.
- **Packet delivery ratio:** SUAP allows to deliver more data than with AODV. Thanks to the hop count hash chains, SUAP has fresher routes than AODV, hence UAVs has more up-to-date routing tables, implying more number of packets being delivered. With single black hole attacks, we obtain 95 % of PDR and 88.5 % for collaborative attacks. This difference lies on the attacker mobility pattern which shadows connectivity between two legitimate nodes. Regarding AODV, the beacon message is not protected which creates non-existing routes to a malicious node. The routing table is

Table VI Experiment results over a one-hour test with single and collaborative blackhole attacks

Parameter with black hole attack	AODV (single)	SUAP (single)	AODV (collaborative)	SUAP (collaborative)
Average loss duration	**7.34 s**	**2.04 s**	**15 s**	**5 s**
Loss rate	**55 %**	**2.8 %**	**94 %**	**5.8 %**
Connectivity during unstable state	**10 %**	**88 %**	**6 %**	**82 %**
Packet delivery ratio	**40.5 %**	**95 %**	**5.68 %**	**88.5 %**
Average end-to-end delay	**5.11 ms**	**35.11 ms**	**7.3 ms**	**45 ms**
Number of route loss	**68**	**21**	**129**	**30**
Connectivity percentage	**9.8 %**	**90 %**	**7.7 %**	**88.68 %**
Routing overhead (percentage compared to all traffic)	**0.6 %**	**1.1 %**	**0.8 %**	**1.5 %**
Security overhead (compared to routing overhead)		**48 %**		**74 %**

not maintained updated. Thus, most of the data packets are lost (more than 50 % of packets are lost in both single and collaborative attack simulations).

- **Average end-to-end delay:** the blackhole attack does not have an effect on the end-to-end delay because malicious nodes immediately drop data packets. When a packet is sent through the blackhole nodes, it will never reach the destination. So, there is no question of delay. Nonetheless, the important delay noticed with SUAP is explained by the time required to add the signature and hash each hop. A RSA signature verification last approximately 5 ms, the signing 10 ms and the decryption about 10 ms [37].
- **Number of route loss**: we noticed more stability with SUAP when facing blackhole attacks. In SUAP, the route is lost because of the mobility. From time to time, UAVs does not have a direct communication. In case of AODV, even when there is UAV to UAV communication, the attacker was able to disrupt the route by publishing a close neighbor thanks to non-authenticated hello messages.
- **Connectivity percentage**s: we noticed that we get more connectivity rates with SUAP than in AODV. This is because each control packets with AODV is dropped by the attacker(s) whereas in SUAP, if the hello message is not properly authenticated, the node is not included in routing table.
- **Overhead**: the blackhole does not affect AODV because when packet is dropped, the sender node does not have a feedback. An ACK is not required during route discovery. As a result, it will retry sending a route request RREQ until a certain threshold (16 in AODV) and approximately 60 bytes each time. The total amount of packet overhead slightly increases when the number of attackers increases because the disconnection rate is increased which induces more route maintenance packets. Regarding SUAP, it generates a significant amount of control packets compared to AODV. SUAP uses both digital signatures and hash chains to provide security. The extension header is about 64 bytes on top of its non-secure routing protocol parts. As a result, the overhead size per control message is higher in SUAP than in AODV. This extension header also increases depending upon the number of nodes. Compared to AODV, SUAP transmits less routing messages in number but more in size.

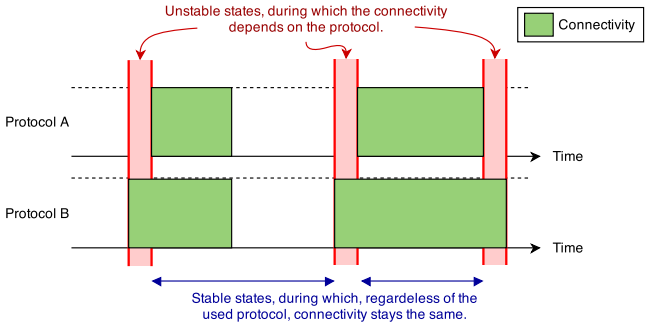


Figure 15 "Unstable states" extracted from the measurement

VI. Conclusion and future research

In this paper, we have presented an extended approach to security analysis and validation of a new secure UAANET routing protocol (SUAP). SUAP is a reactive routing protocol that uses digital signatures for authentication, hash chains for data integrity and geographical leashes to counter blackhole and wormhole attacks. In order to verify its security properties, we have specified and verified its security properties through AVISPA tool. We found that message authentication is properly insured as OFMC back-end states that it is safe. We also discovered that SUAP is vulnerable against confidentiality attacks but these kinds of attacks are not critical for UAANETs due to the real-time characteristics of routing control packets. Furthermore, to complete our security property validation, we carried out a model driven development of our security requirements with Simulink and Stateflow tools in which we were able to check the model and code consistency through a set of integrated formal verification tools included within Embedded Coder. In addition, to evaluate the effectiveness of SUAP, both security property validation and performance evaluation have been carried out with a hybrid testbed associating a virtual mesh emulation and OMNET++ simulations. We found that compared to AODV, SUAP improves the UAANETs quality of service in many aspects.

In regards to our short-term perspectives, our next long-run project would be to deploy our secure routing protocol

into real world outdoor experiments with three UAVs and one GCS. This last part of our project will allow us to validate and compare the different security mechanisms that have been developed so far. Furthermore, even though the hash function is protected by a digital signature, we would also like to add a mechanism is which we periodically change the hash function used throughout the mission to avoid reverse engineering attacks. Such a mechanism adds a security fence when the signature is somehow hacked.

References

[1] I. Bekmezci, O. K. Sahingoz, and Ş. Temel, "Flying ad-hoc networks (fanets): A survey," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1254–1270, 2013.

[2] R. Maulik and N. Chaki, "A study on wormhole attacks in manet," *International Journal of Computer Information Systems and Industrial Management Applications ISSN*, pp. 2150–7988, 2011.

[3] M. G. Zapata, "Secure ad hoc on-demand distance vector routing," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 6, no. 3, pp. 106–107, 2002.

[4] E. Coder, "Generate c and c++ code optimized for embedded systems, 2014 the mathworks."

[5] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuéllar, P. H. Drielsma, P.-C. Héam, O. Kouchnarenko, J. Mantovani *et al.*, "The avispa tool for the automated validation of internet security protocols and applications," in *Computer Aided Verification*. Springer, 2005, pp. 281–285.

[6] J.-A. Maxa, M. Slim Ben Mahmoud, and N. Larrieu, "Secure routing protocol design for uav ad hoc networks," in *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*. IEEE, 2015, pp. 4A5–1.

[7] J.-A. Maxa, G. Roudiere, and N. Larrieu, "Emulation-based performance evaluation of routing protocols for uaanets," in *Communication Technologies for Vehicles*. Springer, 2015, pp. 227–240.

[8] C. Perkins, E. Belding-Royer, S. Das *et al.*, "Rfc 3561-ad hoc on-demand distance vector (aodv) routing," *Internet RFCs*, pp. 1–38, 2003.

[9] E. Pastor, J. Lopez, and P. Royo, "Uav payload and mission control hardware/software architecture," *IEEE Aerospace and Electronic Systems Magazine*, vol. 22, no. 6, pp. 3–8, 2007.

[10] J. Li, Y. Zhou, and L. Lamont, "Communication architectures and protocols for networking unmanned aerial vehicles," in *Globecom Workshops (GC Wkshps), 2013 IEEE*. IEEE, 2013, pp. 1415–1420.

[11] K. Daniel, B. Dusza, A. Lewandowski, and C. Wietfeld, "Airshield: A system-of-systems muav remote sensing architecture for disaster response," in *Systems Conference, 2009 3rd Annual IEEE*. IEEE, 2009, pp. 196–200.

[12] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," in *Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century. Proceedings. IEEE International*. IEEE, 2001, pp. 62–68.

[13] D. B. Johnson, D. A. Maltz, Y.-C. Hu, and J. Jetcheva, "The dynamic source routing (dsr) protocol for mobile ad hoc networks," *IETF Draft, draft-ietf-manet-dsr-009. txt*, 2003.

[14] A. Maghsoudlou, M. St-Hilaire, and T. Kunz, "A survey on geographic routing protocols for mobile ad hoc networks," *Carleton University, Systems and Computer Engineering, Technical Report SCE-11-03*, 2011.

[15] M. Hyland, B. E. Mullins, R. O. Baldwin, and M. A. Temple, "Simulation-based performance evaluation of mobile ad hoc routing protocols in a swarm of unmanned aerial vehicles," in *Advanced Information Networking and Applications Workshops, 2007, AINAW'07. 21st International Conference on*, vol. 2. IEEE, 2007, pp. 249–256.

[16] J.-A. Maxa, M.-S. B. Mahmoud, and N. Larrieu, "Joint model-driven design and real experiment-based validation for a secure uav ad hoc network routing protocol," in *ICNS 2016, 2016 Integrated Communications Navigation and Surveillance Conference*, 2016.

[17] T. Bhattasali, R. Chaki, and S. Sanyal, "Sleep deprivation attack detection in wireless sensor network," *arXiv preprint arXiv:1203.0231*, 2012.

[18] K. Sanzgiri, D. LaFlamme, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer, "Authenticated routing for ad hoc networks," *Selected Areas in Communications, IEEE Journal on*, vol. 23, no. 3, pp. 598–610, 2005.

[19] Q. Li, M. Zhao, J. Walker, Y.-C. Hu, A. Perrig, and W. Trappe, "Sear: a secure efficient ad hoc on demand routing protocol for wireless networks," *Security and Communication Networks*, vol. 2, no. 4, pp. 325–340, 2009.

[20] M. Mohammadizadeh, A. Movaghar, and S. M. Safi, "Seaodv: secure efficient aodv routing protocol for manets networks," in *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*. ACM, 2009, pp. 940–944.

[21] J. Cordasco and S. Wetzel, "An attacker model for manet routing security," in *Proceedings of the second ACM conference on Wireless network security*. ACM, 2009, pp. 87–94.

[22] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: a defense against wormhole attacks in wireless networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3. IEEE, 2003, pp. 1976–1986.

[23] T. Polk, L. Chen, S. Turner, and P. Hoffman, "Security considerations for the sha-0 and sha-1 message-digest algorithms," Tech. Rep., 2011.

[24] N. Larrieu and A. Varet, "Methodology for rapid prototyping avionic software," *Rapid Prototyping of Software for Avionics Systems*, pp. 23–60.

[25] C. M. Holloway, "Towards understanding the do-178c/ed-12c assurance case," in *IET Conference Proceedings*. The Institution of Engineering & Technology, 2012.

[26] L. Rierson, *Developing safety-critical software: a practical guide for aviation software and DO-178C compliance*. CRC Press, 2013.

[27] G. Gigante and D. Pascarella, "Formal methods in avionic software certification: the do-178c perspective," in *International Symposium On Leveraging Applications of Formal Methods, Verification and Validation*. Springer, 2012, pp. 205–215.

[28] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (aodv) routing," Tech. Rep., 2003.

[29] I. Cervesato, "The dolev-yao intruder is the most powerful attacker," in *16th Annual Symposium on Logic in Computer Science—LICS*, vol. 1. Citeseer, 2001.

[30] V. Oracle, "Virtualbox," *User Manual–2013*, 2015.

[31] R. Gantenbein, T. Braun, and T. Staub, "Virtualmesh: An emulation framework for wireless mesh networks in omnet+," in *in The 2nd International Workshop on OMNeT++ (OMNeT++ 2009) held in conjuction with the 2nd International Conference on Simulation Tools and Techniques. ICST (Institute for Computer Sciences, SocialInformatics and Telecommunications Engineering), March 6-*, 2010.

[32] A. Varga *et al.*, "The omnet++ discrete event simulation system," in *Proceedings of the European simulation multiconference (ESM'2001)*, vol. 9, no. S 185. sn, 2001, p. 65.

[33] gnu, *Cryptography implementation C*, 2014 (accessed February 3, 2014). [Online]. Available: https://www.gnu.org/software/libgcrypt/

[34] J. Jonsson and B. Kaliski, "Public-key cryptography standards (pkcs)# 1: Rsa cryptography specifications version 2.1," 2003.

[35] X. Wang, A. C. Yao, and F. Yao, "Cryptanalysis on sha-1," in *Cryptographic Hash Workshop hosted by NIST*, 2005.

[36] F.-H. Tseng, L.-D. Chou, and H.-C. Chao, "A survey of black hole attacks in wireless mobile ad hoc networks," *Human-centric Computing and Information Sciences*, vol. 1, no. 1, pp. 1–16, 2011.

[37] M. Stam and A. K. Lenstra, "Speeding up xtr," in *Advances in Cryptology—ASIACRYPT 2001*. Springer, 2001, pp. 125–143.