Modeling and formal verification of implicit on-demand secure ad hoc routing protocols in HLPSL and AVISPA

MIHAI-LICA PURA, VICTOR-VALERIU PATRICIU, ION BICA

***Abstract*—** Ad hoc networks are a relatively new and promising communication technology. Its key aspect is represented by the specific routing protocols that assure the ad hoc manner of inter node message exchange. But like any other communication technology, ad hoc networks raise specific security problems, especially related to the routing protocols. Although researchers had been very kind with this field and lots of papers were written regarding this aspect, we see no use of these networks in real life applications. A possible explanation would be the lack of user confidence in the security of these special wireless networks. As a countermeasure we propose the use of formal validation methods, model checking in particular, to formally prove the security properties of these protocols. The idea is not necessary new. What represent the novelty are the used tools: HLPSL and AVISPA. Until now researchers used in this matter only mathematical methods or tools like SPIN, that cannot be automated or the possible automation degree is very low. On the contrary, AVISPA offers the possibility to highly automate the modeling the model checking of these protocols.

***Keywords*—**formal verification, model checking, HLPSL, AVSIPA, implicit on-demand secure ad hoc routing protocol, ad hoc networks, ARAN.

I. Introduction

The existence and the functioning of ad hoc networks are based on specific routing protocols. Unlike in the wired networks, were special nodes exist that perform routing, in the ad hoc networks every node must act as a router for the messages of the others. Because no infrastructure exists, the nodes that want to communicate but are not in one another's communication area must rely on the intermediate nodes (with whom both share direct link) to forward their messages. This means that when a node wants to send some data to another one, it first has to determine the path in the network that the data must follow in order to reach the destination. This is called the route discovery phase. After that follows the data forwarding phase, in which the actual message is sent to the destination using the path previously determined.

These actions are performed by the routing protocols and are transparent for the user. By the way in which the routing protocols accomplish their task, they are divided into three categories: pro-active, re-active and hybrid. Pro-active routing protocols are characterized by the fact that they initiate and complete the route discovery phase for every node in the network, after the network initialization. Re-active routing protocols are also called on-demand and they execute the route discovery phase only when a node demands for a path (when the node needs so send data to another node). Hybrid protocols use both previous techniques, but for different partitions of the network. We will focus in our paper only on on-demand ad hoc routing protocols.

The information about the forwarding path is called routing information. It consists from the next node to which the current node must send the data in order to get to the destination. Taken into consideration the response of the route discovery phase, on-demand ad hoc routing protocols can be classified in implicit and explicit. The route discovery phase of an explicit on-demand routing protocol, returns to the source node the whole path from it to the destination. In the case of implicit on-demand routing protocols, the route discovery phase returns to the source node only the next node to whom it should send a packet in order to reach the destination. In fact, this kind of protocols modifies the ARP tables of all the nodes in the corresponding path with this next node information.

The routing protocols which are concerned with security of routing information are called secure routing protocols. For an in-depth analysis of the vulnerabilities and possible attacks on routing protocols please see [14, 15]. Before using such a protocol in a real application, it should be verified if it accomplishes the intended security properties. In [3] are presented the current evaluation techniques for secure routing protocols. The author goes from visual inspection, through simulation and analytical proof systems to formal verification through model checking, presenting the advantages that the former technique has. This is also the field of our work: using model checking tools to verify the security properties of implicit on-demand secure routing protocols.

The rest of the paper is organized as fallows. In section 2 we

Manuscript received May 21, 2010: Revised version received May 21, 2010.

F. A. Mihai-Lica Pura is with the Military Technical Academy, Bucharest, ROMANIA (corresponding author to provide phone: +40-021-335-64-60; fax: +40-021-335-64-60; e-mail: puramihai@yahoo.com).

S. B. Victor-Valeriu Patriciu is with the Military Technical Academy, Bucharest, ROMANIA (corresponding author to provide phone: +40-021-335-64-60; fax: +40-021-335-64-60; e-mail: vip@mta.ro).

T. C. Ion Bica is with the Military Technical Academy, Bucharest, ROMANIA (corresponding author to provide phone: +40-021-335-64-60; fax: +40-021-335-64-60; e-mail: ibica@mta.ro).

present the formal framework of formal analysis that has been already used to verify secure routing protocols. In the third section we present AVISPA model checker and the models that it uses for the components necessary for the verification of secure routing protocols, in relation to the models used in formal analysis and we prove the correspondence between them. In section 4 we present ARAN, an implicit on-demand secure routing protocol and the way we had modeled it for AVISPA's model checker. Section 5 presents the results of formal verification and the last section contains some conclusions.

II. Related work

When developing a secure routing protocol is important to be able to prove that it has the desired security properties. One of the most used techniques is formal analysis. Formal analysis is based on a mathematical modeling of the analyzed system. To model the system means to model its components. For secure routing protocols the system is represented by the network, the adversary and by the routing protocol itself. So the analysis starts by specifying the models used for the network, for the adversary and for the secure routing protocol. Having established this formal framework, one can specify what are the security properties required for the secure routing protocol. After establishing the objectives, the model allows the proving or the disproving of the validity of the properties for the secure routing protocol. Examples of this approach can be found in [1], with the case study of TinyOS which is proved to be insecure, in [2], with the analysis of SRP, DV-SRP and SLSP, and in [4], with the analysis of SRP and Ariadne.

In the formal framework enumerated above, the network is considered to be the collection of all the honest nodes and the adversary. The nodes are considered static ([1]), and they are considered to have a single antenna and a unique name in the network. If the adversary has more than one antenna, it is modeled as more than one node. Given the fact that the ad hoc networks are wireless networks, when a node sends a message, all the nodes in its communication range receive that message too. By analyzing the contents of the message, every node will then establish if it should drop or process the message ([3]). Generally, it is assumed that the wireless links between the nodes are symmetric, which means that if a node A can receive a transmission form a node B, node B can also receive a transmission form node A ([4]).

The adversary is a node of the network that deviates and actively disrupts the network operation ([2]). From the point of view of its resources, the adversary is considered as the most powerful node in the network: it uses a device with a powerful antenna and an unconstrained energy supply ([1]). If more than one adversary exists in the network, it is assumed that they can communicate in out-of-the-band channels (using different frequencies or a wired connection). Regarding the secure routing protocol, it is assumed that the adversary nodes can generate any messages and can replay or modify any received messages ([2], [3]). Also, in [1] it is proved that an adversary can also delete original messages and inject its own fabricated messages instead. Based on these two operations, the adversary can also re-order message sequences. The only restriction imposed for the adversary is that it has a finite processing power and so it cannot mount cryptanalytic attacks in order to compromise a symmetric or an asymmetric key, nor can it inverse one-way and hash functions ([2]). The actions that the adversary will perform are of course dependent on the secure routing protocol targeted, but, generally speaking, it will try to shorten the network life time, degrade the packet delivery ratio, increase its control over traffic and increase network delay ([1]).

The routing protocol is modeled as a distributed algorithm that operates on the collection of the nodes and their direct communication links ([4]). The input of the protocol is a pair of identities: the identity of the source node, and the identity of the destination node. After a finite time interval, the routing algorithm outputs the route between the two nodes, to the source node. When the output is obtained by the source node, it is said that the protocol had discovered the route ([2]).

III. AVISPA formal analysis framework

Our purpose was to automate the formal analysis of secure routing protocols. It was not hard to find the start point in this research. To conclude the previous section about formal analysis frameworks, we highlight the fact that formal analysis is in fact a formal verification process. The system that needs to be analyzed (verified) is first formalized (modeled) according to some previous established assumptions. Then, the intended properties for the system are also formalized. The actual analysis step follows, in which, based on the made assumptions and on the models of the system and of the properties, the validity of the properties is established. It is important to note that the results depend on the considered model.

For the automated formal verification we used AVISPA tool. AVISPA project was developed based on EU founds in FET Open program, IST priority. Its purpose was to build an industrial scale formal verification technology for Internet protocols and applications, with a special emphasis on security. The importance of AVISPA is given by the fact that it has a high capacity of developing new network protocols and of securing already proposed protocols, making them easier to accept by the users. The key element of the project is the formal specification and deduction technology that automate the analysis of the security protocols. According to [12] there is no other formal verification tool that has the same applicability and robustness, by offering in the same time high scalability and performance. A special characteristic of AVISPA is that a same specification can be validated by four tools: OFMC, CL-AtSe, SATMC and TA4SP. For more information regarding the characteristics of these tools, please consult AVISPA user manual [6], and [16].

In order to highlight the parallelism with formal analysis, we will next present the models that AVISPA uses for the

network, for the intruder and the way in which it permits the modeling of a secure routing protocol. In AVISPA the network is represented as a collection of processes that communicate. Each of these processes represent a node in the network, and the special process named i represents the intruder. Each of these processes can communicate with the others using variables of type channel ([6]): one for sending messages and one for receiving messages. The model of the network is directly linked to the model of the intruder. And we will next show how. The model of the intruder in AVISPA is determined by the model of the channels used by the nodes for communications. Currently, the only supported channel model is Dolev-Yao. Under this model, the intruder is assumed to have full control over the network. In consequence, all the messages sent by the nodes go through the intruder. So it can intercept, analyze, and / or modify any message (as far it knows the required keys), and it can send any message that it has composed to any node that it wants, posing as any other node [5]. This type of model for the intruder (and for the channel) means that it is not really important the channels that a node uses for communications: the network is the intruder, and the intruder is the network. An intended connection between certain channel variables (e.g. a node A send messages on a channel SndA for the node B who receives them on the channel RcvA) is irrelevant.

Being given this network-channel-intruder dependence, any message sent by a node in the network will be in fact broadcasted to all the other nodes. Of course, the message can be received only by the nodes for which such a message was specified as a possible received message. We state that this model is a perfect model for the verification of secure routing protocols for ad hoc networks. And we will next prove why. First of all, in the ad hoc network a message emitted by a node will be received by all the other nodes that are in the communication area of the source node. This request is accomplished in AVISPA network model and even more: we cannot specify the exact topology of the network (in the sense that which nodes are in which nodes communication area) but the fact that all the nodes can receive a sent message is even better, because this way, when generating the states for the system, the model checker will generate all the possible combinations of source node-destination node pairs. This means that the verification will be made over all the possible topologies that can be built with the specified number of nodes.

Comparing the modes used by AVISPA for the network and for the intruder, with the ones described in the previous section one can observe the concordance between them. This means that the results obtained based on AVISPA models will have the same generality as the results obtain by formal analysis.

IV. MODELING ARAN IN HLPSL

To show how AVSIPA can be used to verify the security properties of implicit on-demand secure routing protocols, based on the previous presented models used by this formal verification tool, we have chosen ARAN (Authenticated Routing for Ad hoc Networks) secure routing protocol, introduced in [7], and proved to be insecure in [11] and [17].

In order to verify ARAN using AVISPA, it has to be modeled using a special input language called HLPSL (High Level Protocol Specification Language). All the nodes in the network that perform the same actions regarding the modeled protocol have to be grouped together in what HLPSL calls 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. Taken into consideration the authenticated route discovery part of ARAN, the nodes of a wireless ad hoc network (by the actions that they perform) can be grouped in three roles: source, destination and intermediate. The source node initiates the authenticated route discovery process by broadcasting a route discovery packet (RDP) which contains the route discovery packet identifier (RDP), the IP address of the destination to which the route is needed, the certificate of the source node, a nonce and the current time. All this information is signed with the private key of the source node.

A -> broadcast: [RDP, IPX, certA, NA, t]privA

Depending on the topology of the network, this packet will reach directly the destination, or will reach an intermediate node. When a node receives a RDP packet directly from the source node, it will validate the certificate of the source node (A in our example); it will then extract the public key and will check the signature of the source node over the message. If the signature is valid, the intermediate node will check to see if it has not already processed this RDP by looking at the tuple (IPA, NA). If it has already processed the RDP, the package is discarded. Otherwise the intermediate node will sign the received packet, will append its own certificate and then will broadcast the new composed packet.

When a node receives a RDP packet from another intermediate node, it will only validate the certificate and the signature of the previous intermediate node. Then it will perform the same check as previous described. Before broadcasting the packet, this new intermediate node will remove the signature and the certificate of the previous intermediary and will add its own signature and certificate.

Eventually the packet will arrive at the destination node. This node will validate the certificate of the source node and its signature and / or the certificate of the last intermediate node and its signature, depending if the RDP has reached directly the destination or it has traveled through one or more intermediate nodes. Then, the destination will verify if it had not already responded to this RDP, by looking also at the tuple (IPA, NA). If the message was already processed, it is discarded. Otherwise the destination composes a replay packet (REP) that contains the identifier of the packet type (REP), the IP address of the source from which the corresponding RDP had came, the certificate of the destination node, the nonce and the time from the RDP package. All the information is signed

with the private key of the destination node.

The destination node sends the packet not by broadcast, but by unicast to the node from which it has received the associate RDP packet. Dependent on the topology, the packet will arrive directly to the destination, or it fill first travel through one or several intermediate nodes. Again, these intermediate nodes will not broadcast the REP, but it will send it only to the node from which they received the corresponding RDP. The first intermediate node will validate the certificate and the signature of the destination, it will sign the REP, add its own certificate and then unicast it. The other intermediate nodes will do the same, but they will validate only the certificate and the signature of the previous intermediate node. As in the case of RDP, the intermediates will remove the signature and the certificate of the previous intermediate, it will then add their own signature and certificate and only then they will unicast the packet. When the source receives the REP it will have to validate the certificate and the signature of the destination node and / or of the previous intermediary.

In order to verify the security properties of the route discovery phase of ARAN, we had to model three roles: the role of the source node, the role of the intermediate nodes, and the role of the destination node. As described above, the source node performs two actions: it sends the RDP for a given destination node and it receives the REP for the associate RDP. Regarding the receiving of the REP, it can be received directly from the destination, or from an intermediate node. In the two cases, the REPs will have different formats. The HLPSL code for the source node is:

```
role source(S, D        : agent,
            Ks, Kd      : public_key,
            KeyRing     : (agent.public_key) set,
            Snd, Rcv    : channel(dy))

played_by S def=

local State         : nat,
      H             : agent,
      Na            : text,
      Kh            : public_key,
      RDP, REP      : text

init State := 0

transition

rreq. State = 0 /\ Rcv(start)
      =|> State' := 2 /\ Na' := new() /\ RDP' := new() /\
Snd({RDP'.D.S.Ks.Na'}_inv(Ks))

confirmi. State = 2 /\
Rcv({{REP'.S.D.Kd.Na}_inv(Kd)}_inv(Kh').H'.Kh') /\
in(H'.Kh',KeyRing)
          =|> State' := 3 /\ request(D,S,shd,Na)

confirmd.   State = 2 /\ Rcv({REP'.S.D.Kd.Na}_inv(Kd))
          =|> State' := 3 /\ request(D,S,shd,Na)

end role
```

The parameters of the role represent the initial knowledge of a source node: its identity S in the network, the identity of the destination node to which it will have to send messages, the public key of the destination node, and two variables of type channel for sending and receiving messages. The variable KeyRing is in fact an array that contains the certificates of all the nodes in the network (we modeled the certificates as pairs of identities and public keys, but without being signed by the certification authority). We have chosen to give to each role the knowledge of all the agents and their public keys in order to eliminate the need to model the central authority and the request / receive of certificates for a given identity, and thus reducing the state space. Of course, this will eliminate the possibility to model key revocation, but it will not have any effect on the security properties of the actual secure routing protocol.

As can be seen from the HLPSL code, the source node is requested to initiate the route discovery by sending it the start message when it is in the initial state. After sending the RDP, the source node changes the state to a one in which it just waits for the REP. The validation of the certificate is made by searching it in the KeyRing array. The validation of the signature is implicit: it could have been made only by the inverse key of the public key. So if the corresponding key is found in KeyRing, the signature is also validated.

Another interesting modeling aspect is related to the timestamps. As one can observe from the HLPSL code of the source node, the RDP and REP packets do not contain the timestamp information, as specified by ARAN's authors. The fact is that HLPSL (and in particular AVISPA's back-ends) do not support time. But what timestamps should achieve is to limit the time window in which a message is accepted by a recipient and thus limiting the replay of messages. So AVISPA project team propose and suggest the use of weak authentication security goal instead of the standard authentication, in which such attacks are ignored ([13]).

The intermediate nodes perform also two actions: the forwarding of a RDP and the forwarding of a REP. In both cases, the packets can have two formats, depending if they were received directly from the source/destination or from another intermediate node. The HLPSL code for an intermediate node is:

```
role node(N         : agent,
          Kn        : public_key,
          Memory    : (text.text.agent) set,
          KeyRing   : (agent.public_key) set,
          Snd, Rcv  : channel(dy))

played_by N def=
```

```
local State          : nat,
    Ks, Kd, Kh   : public_key,
    Na               : text,
    S, D, H          : agent,
    RDP, REP     : text

init State := 1

transition

forward10. State = 1 /\ Rcv({RDP'.D'.S'.Ks'.Na'}_inv(Ks'))
/\ not(in(RDP'.Na'.D', Memory))
        =|>     State'  :=  1   /\  Memory'  :=
cons(RDP'.Na'.D',Memory)                          /\
Snd({{RDP'.D'.S'.Ks'.Na'}_inv(Ks')}_inv(Kn).N.Kn)

forward11.          State       =       1       /\
Rcv({{RDP'.D'.S'.Ks'.Na'}_inv(Ks')}_inv(Kh').H'.Kh')      /\
not(in(RDP'.Na'.D', Memory))
        =|>     State'  :=  1   /\  Memory'  :=
cons(RDP'.Na'.D',Memory)                          /\
Snd({{RDP'.D'.S'.Ks'.Na'}_inv(Ks')}_inv(Kn).N.Kn)

end role
```

An intermediate node can only be in one state: the one in which it receives and forwards RDP and REP packets. Because the format of the two packets is the same and because the actions involved by forwarding are the same, we only modeled the code for the RDP, but it is actually used for REP also. When receiving a packet, the node will first search the tuple (Na,D) in the array Memory. If the tuple is found there, it means that the request was already processed, and the message is discarded. If the tuple is not in Memory, it means that the node has not already seen the same message and will process it. After that it will save the tuple in the array for the next verifications. Regarding the validation of the signatures and of the certificates in the messages, the observations made for the source node apply here too.

The destination node has to perform a single action: to respond to a received RDP. The RDP can have two formats, depending if it was received directly from the source, or if it has traveled through one ore mode intermediates. The HLPSL code for the destination node is given next.

```
role destination(D          : agent,
                 Kd         : public_key,
                 Memory     : (text.text.agent) set,
                 KeyRing    : (agent.public_key) set,
                 Snd, Rcv   : channel(dy))

played_by D def=

local State      : nat,
    S, H         : agent,
    Na           : text,
    Ks, Kh       : public_key,
    RDP, REP     : text

init State := 4

transition

respondd.   State = 4 /\ Rcv({RDP'.D.S'.Ks'.Na'}_inv(Ks'))
/\ in(S'.Ks',KeyRing) /\ not(in(RDP'.Na'.S',Memory))
        =|>     State'  :=  6   /\  Memory'  :=
cons(RDP'.Na'.S',Memory)    /\  REP'    :=  new()   /\
Snd({REP'.S'.D.Kd.Na'}_inv(Kd)) /\ witness(S',D,shd,Na')

respondi. State          =          4          /\
Rcv({{RDP'.D.S'.Ks'.Na'}_inv(Ks')}_inv(Kh').H'.Kh')       /\
in(H'.Kh',KeyRing) /\ not(in(RDP'.Na'.S',Memory))
        =|>     State'  :=  6   /\  Memory'  :=
cons(RDP'.Na'.S',Memory)    /\  REP'    :=  new()   /\
Snd({REP'.S'.D.Kd.Na'}_inv(Kd)) /\ witness(S',D,shd,Na')

end role
```

The destination node waits for RDP packets and responds to them according to the protocol. Like an intermediate node, it will first determine if it has already responded to the request using the same mechanism: the tuple (Na,D) for an answered packet is saved for future verifications. The validation of certificates and signatures is done in the same way as described for the source node.

Based on these three roles, the scenario (in fact the network) in which the protocol will be verified, can be modeled. The model of the network is made in a composed role. We give below an example of such a role. One can observe that the network is modeled by instantiating one process of type source role, none, one or several processes of type intermediate role, and a single process of type destination role. To reduce the state space, we modeled the source node and the destination node for a single route discovery. That is why the network should contain only one source node and one destination node. Besides the instantiations of the processes that represent the nodes of the network, the composed role that we named session contains the declarations and the initializations of some of the parameters for the roles: the channels, the arrays used for transmitting the certificates, and the arrays used for storing the identifiers for the processed packets.

```
role session(T, U, V, W      : agent,
             Kt, Ku, Kv, Kw  : public_key) def=

  local S1, R1,
        S2, R2,
        S3, R3,
        S4, R4          : channel(dy),
```

```
        Mem1, Mem2, Mem3  : (text.text.agent) set,
        KS1, KS2, KS3, KS4  : (agent.public_key) set

    init Mem1 := {} /\ Mem2 := {} /\ Mem3 := {} /\ KS1 :=
{T.Kt,U.Ku,V.Kv,W.Kw}          /\          KS2          :=
{T.Kt,U.Ku,V.Kv,W.Kw}          /\          KS3          :=
{T.Kt,U.Ku,V.Kv,W.Kw}          /\          KS4          :=
{T.Kt,U.Ku,V.Kv,W.Kw}

    composition

      source(T,W,Kt,Kw,KS1,S1,R1)
    /\node(U,Ku,Mem1,KS2,S2,R2)
    /\node(V,Kv,Mem2,KS3,S3,R3)
    /\destination(W,Kw,Mem3,KS4,S4,R4)

  end role
```

The security properties of ARAN, as is stated in its presentation in [7], are based on the fact that all the packets are exchange in an authenticated manner. That is why the security property that we formal verified was the authentication of the destination by the source node when it receives the REP for the RDP that it has sent. In order to verify this property, it had to be formally specified. An authentication security goal consists out of two goal facts (witness and request) and a goal (authentication). The goal facts are used to augment the transitions of the basic roles, and the authentication goal is used to assign a meaning to them. Witness and request are used to check if an instance of a role is right in believing that its peer is present in the current session, has reached a certain state, and agrees on a certain value, which typical is fresh ([5]). They always appear in pair and have the same third parameter. This third parameter is the identifier of the authentication goal and it is used in the goals section of the HLPSL code. In our protocol, the destination node D has to authenticate to the source node S and they both should agree on the value of Na, and vice versa. That is why the goal facts were used to augment the transition in which the destination node receives, process and responds to the RDP (witness), and the transition in which the source nodes receives the REP answer (request).

V. FORMAL VERIFICATION RESULTS

For the verification we used a personal computer running openSuse 11.0, with 2 GB of RAM and an Intel DualCore 2 GHz processor. After completing the specification as described above, we divided the verification process into two steps. The first step was to validate the specification using OFMC back-end tool of AVISPA, and the second step was to use ATSE back-end. For both steps we prepared the same input files. These input files were in number of four. The first input file represented an ad hoc network formed by only two nodes: the source node and the destination node (from HLPSL point of view, this meant the composition of a source role with a destination role). The second file added an intermediate node in order to represent a three node ad hoc network (a source role was composed with a node role and a destination role). Each of the two remaining files added another node role to the specifications, representing this way a four, and respectively, a five node ad hoc network.

OFMC is the back-end tool of AVISPA that has the highest speed in detecting attacks. This is the reason for which we started the verification with it. The first specification was successfully validated and OFMC report that it is safe. For the other three specifications, OFMC did not complete the validation, not even after 72 hours. So we decided to perform a validation with a depth bound for the search. This means that when expanding the state space of the model, OFMC does a depth-first search (for which it only goes to a depth level that is less of equal to the bound), while the standard search strategy is a combination of iterative deepening search with breadth-deep search. This bounding resulted in a smaller validation time. We considered the bounds varying from two to twelve. The time needed for the validation in each case is given in the table below. The cells that contain a dash represent an incomplete validation, which was interrupted after one hour. It is worth mentioning that for every bounded validation, OFMC reported the protocol to be safe, according to the specified goals. According to AVISPA user manual, this bounded search does not affect the validity of the results, because the messages an intruder can generate are not bounded. In fact, this is the only way in which OFMC can be used not to find attacks, but to validate a protocol.

Tabel 1 Validation time for OFMC

Bound	OFMC validation time (s) for		
	3 nodes	4 nodes	5 nodes
2	0.03	0.03	0.04
3	0.04	0.1	0.32
4	0.14	0.75	2.9
5	0.57	4.93	26.34
6	2.48	34.59	195.92
7	10.56	190.08	1660.05
8	44.20	1211.83	-
9	176.09	-	-
10	228.59	-	-
11	1256.4	-	-
12	4185.8	-	-

AVISPA's ATSE back-end tool provides a translation for

the protocol specification into a set of constrains. In particular, each step of the protocol is modeled by using constraints on the adversary's knowledge. Then, these constraints are used to find attacks against the specified security objectives. The algorithm that this tool is using is designed for a bounded number of protocol steps. But, according to the manual, if the protocol is loop free, all the specification will be verified. When feeding the input files to ATSE, it completed the verification process and reported that the protocol is safe, only for the first three network topologies. For the five node ad hoc network the validation was uncompleted even after three hours, and it was interrupted. The time consumed for the verification of each of the four specifications is given in the table below.

Tabel 2 Validation time for ATSE

ATSE validation time (s) for			
2 nodes	3 nodes	4 nodes	5 nodes
0.01	0.25	14.26	-

One can observe that ATSE back-end is faster than OFMC. Nevertheless, both of the two tools should be used together, because they apply different algorithms. So if both report that the protocol is safe, the probability that the actual system accomplishes the security objectives is higher.

We restricted the verification to an ad hoc network of maximum 5 nodes. If it is necessary to perform the verification for a higher number of nodes, the tables can help us in approximating the amount of time needed for such an attempt. For OFMC, it can be approximated that in an hour it can validate the protocol for a network of maximum 8 nodes with a session number bound of 2. We appreciate that for any other network larger than 8 nodes, the verification process will last too much to worth expecting its end.

VI. Conclusion

In our paper we presented how AVISPA formal verification tool can be used to validate the security properties of implicit on-demand ad hoc secure routing protocols. In order to prove the technique, we demonstrated it in a case study: formal verification of ARAN.

From our knowledge and as stated in [3] there were very little attempts to apply model checking techniques for secure routing protocols in general, and for implicit on-demand secure routing protocols in particular. We only know the attempts of using SPIN model checker (for DSR, SRP, Ariadne and endairA [3] and [10]), HOL theorem proofer ([8]), CPAL-ES (for SRP [9]), and Uppaal (for ARAN [11]). But we did not find any reports of using AVISPA in this purpose. So we consider that our work is an important step forward in using formal verification for secure routing protocols too.

The major advantage of our approach is the use of AVISPA toolkit and its specification language. The reason is related to the models that this tool uses for the network and the intruder, as shown in section 3. The way these models are implemented, assure that a verification process is performed for all the possible topologies that can exist, being given the number and the type of the roles specified. We consider this a major improvement of the formal verification of ad hoc routing protocols, because it is specific for them to have an evolving topology. For example, in the solution proposed and used by Andel in [3] (SPIN and Promela to formally verify DSR and SRP), one has to explicitly specify each of the network topologies for which the validation has to be made.

Our work was focused only on how to specify in HLPSL implicit on-demand ad hoc routing protocols. We have highlighted in section 5 the special problems raised by this language in order to model the entire specification of ARAN. Based on this example, any other implicit on-demand routing protocol based on PKI security, can be easily model and verified. In our future work we intend to complete the formal verification of ad hoc routing protocols by taken into consideration explicit on-demand ad hoc routing protocols. Also, we intend to see how this technique can be applied for the latest propositions for secure routing protocols that are based on identity-based cryptography ([18]).

References

[1] Gergely Acs, Levente Buttyan, Istvan Vajda, "Modelling Adversaries and Security Objectives for Routing Protocols in Wireless Ad hoc Networks", in *Proceedings of the Fourth ACM Workshop on Security of ad hoc and sensor networks*, 2006

[2] Panagiotis Papadimitratos, Zygmunt J. Haas, Jean-Pierre Hubaux, "How to Specify and how to Prove Correctness of Secure Routing Protocols for MANET", in *IEEE-CS Third International Conference on Broadband Communications, Networks and Systems*, 2006

[3] Todd R. Andel, Alec Yasinsac, "Automated Security Analysis of Ad Hoc Routing Protocols", in *Proceedings of the Joint Workshop on Foundations of Computer Security and Automated Reasoning for Security Protocol Analysis*, 2007

[4] Levente Buttyan, Istvan Vajda, "Towards Provable Security for Ad Hoc Routing Protocols", in *Proceedings of the Second ACM Workshop on Security of ad hoc and sensor networks*, 2002

[5] The AVISPA team, "HLPSL Tutorial", 2006

[6] The AVISPA team, "AVISPA v1.1 User Manual", 2006

[7] Kimaya Sanzgiri, Bridget Dahill, "A Secure Routing Protocol for Ad Hoc Networks", in *Proceedings of the 10th IEEE International Conference on Network Protocols*, 2002

[8] Karthikeyan Bhargavan, Davor Obradovic, Carl A. Gunter, "Formal verification of standards for Distance Vector Routing Protocols", in *Journal of the ACM (JACM),* 2002

[9] John D. Marshall, II, "An Analysis of the Secure Routing Protocol for Mobile Ad Hoc Route Discovery: Using Intuitive Reasoning and Formal Verification to Identify Flaws", MD Thesis, the Florida State University, 2002

[10] Todd R. Andel, "Formal Security Evaluation of Ad Hoc Routing Protocols", PhD Thesis, the Florida State Univeristy, 2007

[11] Jens Chr. Godskesen, Olena Gryn, "Modeling and Verification of Security Protocols for Ad Hoc Networks using Uppaal", in *Proceedings 18thNordic Workshop on Programming Theory*, 2006

[12] A. Armando, D. Basin, J. Cuellar, M. Rusinowitch, L. Vigano, "AVISPA: Automated Validation of Internet Security Protocols and Applications", ERCIM News – Online Edition, no. 64, January 2006

[13] AVISPA mailing list, http://avispa-project.org/avispa-users-old/2006-May/000219.html

[14] Ali Ghaffari, "Vulnerability and Security of Mobile Ad Hoc Networks", in *Proceedings of the 6th WSEAS International Conference on Simulation, Modelling and Optimization*, 2006

[15] B. Malarkodi, B. Venkataramani, X.T. Pradep, "Modified AODV Protocol for Prevention of Denial of Service Attacks in Wireless Ad Hoc Networks", in *Proceedings of the 5th WSEAS Int. Conf. on Applied Informatics and Communications*, 2005

[16] Abdelilah Tabet, Seonghan Shin, Kazukuni Kobara, Hideki Imai, "On Formal Verification Methods for Passwords Based Protocols: CSP/FDR and AVISPA", in *Proceedings of the 4th WSEAS International Conference on Information Security, Communications and Computers*, 2005

[17] Abdalla Mahmoud, Ahmed Sameh, Sherif El Kassas, "Authenticated Routing for Ad Hoc Networks Protocol and Misbehaving Nodes", in *Proceedings of the 4th WSEAS International Conference on Telecommunications and Informatics*, 2005

[18] Jue-Sam Chou, Chu-Hsing Lin, Chia-Hung Chiu, "An Identity-Based Scheme for Ad Hoc Network Secure Routing Protocol from Pairing", in *Proceedings of the 5th WSEAS International Conference on Applied Computer Science*, 2006