SIMULATION OF AN IDENTITY-BASED CRYPTOGRAPHY SCHEME FOR AD HOC NETWORKS

Pura Mihai-Lică, Patriciu Victor-Valeriu and Bica Ion
Department of Military Informatics and Mathematics, Military Technical Academy
81-83 George Coşbuc Boulevard, Bucharest, Romania

Keywords: Ad hoc network, Identity-based cryptography, Ns2, MIRACL.

Abstract: Ad hoc networks are a promising technology especially from the point of view of its aim: assuring connectivity. But communication cannot be separated from security without loosing a lot of its benefits. That is why research has to focus on security aspects of the ad hoc network too. The paper presents an implementation of a security scheme for ad hoc networks based on identity-based cryptography. This implementation was made in ns2, using MIRACL library to implement identity-based encryption. The solution focuses only on assuring confidentiality, but can be further developed to assure also authentication, integrity and non-repudiation. For the start, an ns2 implementation was developed to provide a simulation environment where different possible scenarios can be tested and the scheme can be improved according to the results, before the real implementation.

1 INTRODUCTION

Mobile ad hoc networks are self-managing networks formed of mobile routers that interconnect through wireless links and have an arbitrary topology. The routers are permitted to randomly move and to arbitrary organize themselves. The topology of such a network can rapidly change in an unpredictable way. The main purpose of such networks and of ad hoc networks in general is assuring connectivity between all the nodes. From this point of view, the routing algorithm is the heart of an ad hoc network. But, when it comes to communications, security is another factor that has to be taken into consideration, because it has become a constant demand. The proposed objective was to implement a security scheme for such networks that would correspond to their special characteristics.

The security schemes used in wired networks were designed to take advantage of all the benefits of this kind of networks: high speed, low cost, reliability, superior performance and others. That is why they cannot be applied in ad hoc networks without any change. But it would be easier to develop security solutions specially designed for the needs of ad hoc networks. Of course, these kinds of solutions have to focus on the special characteristics of wireless ad hoc networks: decentralized network architecture, transient node behavior, heterogeneity of network node resources and the self healing and managing ability of the network.

With these considerations in mind, let us take a look at the concept of identity-based cryptography. Identity-based cryptography is a particular case of public key cryptography; its main characteristic is the fact that the public key is chosen to be a string of characters that represents in a unique way the identity of the key holder (for instance name and address, e-mail address, telephone number or IP address). The private key is computed based on the public key. The computation of the private key is the job of a specialized network node called key generation center. This node also makes available to all the nodes the parameters required for the cryptographic operations, depending on the algorithm used. The main advantage of this scheme is that certificates are not needed; each node can compute the public key of the node it wants to communicate with, without being able to compute the private key too. Because there is no need to check the validity of the certificates, the key generation center is no longer involved in the communication, after it has provided the nodes with their private keys.

Comparing the characteristics of identity-based cryptography with the properties of an ad hoc

Mihai-Lică P., Victor Valeriu P. and Ion B. (2009).
SIMULATION OF AN IDENTITY-BASED CRYPTOGRAPHY SCHEME FOR AD HOC NETWORKS.
In *Proceedings of the International Conference on Security and Cryptography*, pages 135-139
DOI: 10.5220/0002183701350139

network it is obvious that identity-based encryption suits ad hoc networks' needs very well. It is like they were made for each other. Using identity-based cryptography for securing ad hoc networks is a lately preoccupation of researchers. An example can be found in Oliveira L.B.et al., 2007. A team from The University of Brazil had successfully implemented and tested a secure communication scheme for sensor networks based on Tate pairing and named TinyTate. The protocol developed focuses on the needs of resource constrained nodes, but this does not affect the generality of the implementation.

What our team proposed was to develop a simulation environment where the use of identity-based cryptography in ad hoc networks can be simulated and tested. The simulation of ad hoc routing protocols is very easy with the use of network simulators (like ns2), because ad hoc routing protocols implementations already exists. But simulating secure ad hoc networks is not as easy. This was the motivation of our work that tried to answer this issue.

2 IDEA

Imagine the scenario of a conference. When a conference is organized, several discussion topics are established, and for every one of the themes a separate room is assigned. People joining the conference can participate at discussions in all the rooms. This means that someone can arrive at the beginning and join discussion topic in room A. After an half an hour, say he or she gets bored and goes to room B. Then, after fifteen minutes, decides to go back to room A. And so on and so forth. A mobile ad hoc network would be the best solution to assure the communication between the laptops or the PDAs of the participants. But how can someone assure security? What are the characteristics of such a scenario? First of all, before the actual communication starts all the participants had arrived and all had checked in. After the conference began no one can check in anymore. Second, the security of communications has to be provided only for the duration of the conference, witch is a relative small time (several hours maybe). Third, once a person was selected for this conference, he of she cannot be excluded during the conference.

The following security scheme can be proposed for such scenarios, based on the utilization of identity-based cryptography. At checking in, every attendant of the conference is provided with an IP, in order to access the ad hoc network of the conference. At the reception there is also the key generation center. After the IP is assigned to the participant's device, it can ask the key generation center for the public variables needed for identity-based computations. After it receives them, it can ask for its private key. The key generation center computes the participant's private key based on the IP assigned to it and returns it. The private key is exchanged on a secure channel (for example Bluetooth) and the authentication of the requestor is made by physical contact. The participant will also receive a list of all the participants of the conference and their IP numbers. After all the participants check in, the key generation center, the only one that can generate the private keys is shutdown. So the participants can interchange secure messages based only on their IPs.

3 IMPLEMENTATION

When implementing all these in ns2, the work was divided in several steps. The first step was to determine how this security scheme can be implemented in ns2. The second step was to find an identity-based cryptography library that can be use with ns2 (this means that it had to be written in C or C++). The next step was the implementation of the objects involved (the key generation centre, the communicating nodes) through the implementation of the five needed algorithms (Boyen X., Martin L., 2007): initialization of the key generation centre, generation of the private keys, generation of the public keys, and encryption of a message and decryption of a message. The last step was to test the implementation and to conclude future development directions.

3.1 Security Scheme Implementation and Library Utilization

With ns2, ad hoc networks can be simulated using specific objects that represent the nodes of the network. For these nodes, among other things, the name of the ad hoc routing protocol needs to be specified. Of course, ns2 does not contain an implementation for all the routing protocols known, but only for the most important ones like AODV, DSR, and TORA. To simulate traffic in ns2, an agent object must be attached to the nodes. The agent object can act as a source or as a destination for the communication packets. New agents can be

implemented and added to ns2 in order to simulate new protocols. The implementation of the proposed secure communication scheme is based on such an agent (Leiming Xu, 2001, Ros F.J., Ruiz P.M., 2004).

The choice for the identity-based cryptography library was MIRACL (Multiprecision Integer and Rational Arithmetic C/C++ Library) from Shamus Software. MIRACL is an open-source Big Number Library who implements all the primitives necessary to design Big Number Cryptography into real-world application, and is free for educational use. This library already contains an implementation of the Boneh-Franklin identity-based encryption algorithm (Boyen X., Martin L., 2007). The implementation works with files, so code needed to be modified, but it was a good starting point.

3.2 Algorithms' Implementation

Next, the objects that were implemented in ns2 in order to create the proposed security scheme are presented. The code and the elements used are briefly discussed.

```
class KGCAgent : public Agent {
private:
   Big *p, *q, *xP, *yP, *xPpub,
*yPpub, *xcube, *ycube;
   Big *s;
   bool Set_KGC(void);
public:
   KGCAgent();
   virtual int command(int argc, const
char*const* argv);
   virtual void recv(Packet*,
Handler*);
};
```

As one can see from the KGCAgent presented above, a unitary approach was taken. This means that this agent can be used for the key generation centre node and for the communicating nodes also. The only difference is that for the key generation centre the Set_KGC() method must be called to set up the environment, prior to anything else. The s attribute represents the private key (of the KGC or of the communicating nodes, depending of the type of node that the agent is attached to). Attributes p, q, xP, yP, xPpub, yPpub, xcube, ycube represent the public parameters of the identity-based cryptography environment. One can observe the data type Big that was used to store the private key and also the public parameters. This is a data type specific to MIRACL and it is use to store very large numbers.

3.2.1 Key Generation Centre Initialization

Initialization of the identity-based cryptography environment consists in setting the private key for the key generation centre and constructing the public parameters needed in calculations. This is done only once, after the construction of the key generation centre node, by calling the method named Set_KGC(). The private key of the key generation center remains secret. No one can steal it from the key generation node: it is a private attribute of the KGCAgent class. This, of course, is valid also for the private keys of the communicating nodes that are stored in the same attribute.

3.2.2 Private Keys Generation

Generation of the private keys is made by the key generation centre only, based on its private key and on the public parameters and, of course, on the public key of the requestor. It is the time to mention that the public key selected to be used was the IP address of the nodes. A node requests the private key from the key generation centre node, and the key generation centre node computes is starting from its IP address and sends it back. The node stores its private key in the s attribute.

3.2.3 Public Key Generation

Each node can compute its public key or the public key of any other node in the network using the public parameters of the identity-based cryptography environment and the IP of the node. Public key computation is necessary for the generation of the private key for a node by the key generation centre node and for the encryption of a message for a certain node.

3.2.4 Message Encryption

When node A envisages sending an encrypted message to node B, it first randomly generates an AES symmetric key. This key is then used to encrypt the message for the node B. Then node A generates the public key of the node B based on its IP. The B's public key is then used to encrypt the AES key (Cooks Clifford, 2001). The encrypted AES key and the encrypted message are then sent to node B.

3.2.5 Message Decryption

When node B receives an encrypted message it first decrypts the AES key using its private key. Then,

using this AES key it decrypts the actual message. Because the AES key was encrypted with its pubic key, node B is the only node that can decrypt the symmetric key and, consequently, the message.

3.3 Communications

Communications taking place in the network can be grouped in three categories: private key request messages, public parameters request messages and text message delivering messages. The structure created to support all these messages is presented below:

```
struct hdr_kgc {
   char ret;
   char priv_key[400];
   char V[HASH_LEN],W[HASH_LEN];
   char msg[100];
   unsigned int msg_length;

   static int offset_;
   inline static hdr_kgc* access(const
Packet* p)
   {
      return
        (hdr_kgc*) p->access(offset_);
   }
};
```

3.3.1 Private Key Request Messages

This kind of message can only originate from a communicating node and can only be sent to the key generation centre node. The ret attribute of the message must have the value '0'. Using the IP of the source node and the parameters of the environment, the key generation centre computes the private key and sends it back in the priv_key attribute of the message, setting also ret attribute to the value '1'. The node that requested the private key extracts it from the corresponding attribute and stores it in the s attribute.

3.3.2 Public Parameters Request Messages

After the initialization phase takes place in the key generation centre node, the public parameters (that are 8 in number) are available to all the nodes in the network. Each one of these parameters must be requested separately from the key generation centre only, by setting the ret attribute of the message to one of the values: '2' for the first parameter, '4' for the second, '6' for the third and so one to '16' for the eight. When receiving such a message, the key generation centre node puts in the priv_key attribute of the message the corresponding parameter and change the ret attribute to a value of: '3' for returning the first parameter, '5' for returning the second parameter, '7' for returning the third parameter and so on to '17' for returning the eight parameter. The node that made the request extracts the parameter from the priv_key attribute and stores its value in its corresponding attribute.

3.3.3 Text Message Delivery Messages

After a node has requested and received the public parameters, it can now send and receive text messages to and from the nodes of the network. If node A wants to send the message "Hello" to the node B, it encrypts the message as shown at 2.2.4. Then it puts the encrypted message in the msg attribute of the message and the message length in the msg_length attribute of the message and the encrypted AES key in the attributes V and W. The ret attribute is set to the value '18'. The message is then sent to the destination. Here, in order to view the actual message received, the node must apply the decryption algorithm from 2.2.5.

4 SIMULATION

Let's now see the implemented agent at work. Only the part of a tcl script where our agent is used, is presented. First, let's consider the general simulation scenario. Because its purpose is the presentation of the usage of the KGCAgent object, it is a very simple one. The simulation has three nodes. One of them is the key generation centre. The other two are communicating nodes. After creating the three nodes, then the agent for the key generation centre node, the Set_KGC method is called, in order to set up the environment and it is attached to the first node.

```
set p0 [new Agent/KGC]
$p0 call-setkgc
$ns_ attach-agent $node_(0) $p0
```

Then the two agents for the communicating nodes are created and attached to the second and to the third node, respectively.

```
set p1 [new Agent/KGC]
$ns_ attach-agent $node_(1) $p1
set p2 [new Agent/KGC]
$ns_ attach-agent $node_(2) $p2
```

Subsequently, the key generation center agent and the agent of the second node are connected. Then the second node requests its private key and the eight public parameters.

```
$ns_ connect $p0 $p1
$ns_ at 0.30 "$p1 getk"
$ns_ at 0.32 "$p1 getp"
$ns_ at 0.34 "$p1 getq"
$ns_ at 0.36 "$p1 getxP"
$ns_ at 0.38 "$p1 getyP"
$ns_ at 0.40 "$p1 getxPpub"
$ns_ at 0.42 "$p1 getyPpub"
$ns_ at 0.44 "$p1 getxcube"
$ns_ at 0.46 "$p1 getycube"
```

The same thing is done for the third node.

```
$ns_ at 0.60 "$ns_ connect $p0 $p2"
$ns_ at 0.62 "$p2 getk"
$ns_ at 0.64 "$p2 getp"
$ns_ at 0.66 "$p2 getq"
$ns_ at 0.68 "$p2 getxP"
$ns_ at 0.70 "$p2 getyP"
$ns_ at 0.72 "$p2 getxPpub"
$ns_ at 0.74 "$p2 getyPpub"
$ns_ at 0.76 "$p2 getxcube"
$ns_ at 0.78 "$p2 getycube"
```

The agents of the communicating nodes are connected together. Then the nodes are moved. During their movement, the nodes exchange text messages.

```
$ns_ at 1.0 "$ns_ connect $p1 $p2"
$ns_ at 1.0 "$node_(1) setdest 105.0
200.0 60.0"
$ns_ at 1.0 "$node_(2) setdest 200.0
105.0 60.0"
$ns_ at 2.0 "$p1 send Question?"
$ns_ at 3.0 "$p2 send Answer!"
```

Note also that the commands use for all these actions (call-setkgc, getk, getp, getq, getxP, getyP, getxPpub, getyPpub, getxcube, getycube, send some_text_message) are implemented in the command method of KGCAgent class, as discussed in the presentation of the possible messages types. The recv method of the same class deals with the possible responses to all these commands in the way discussed also in 2.3.

5 CONCLUSIONS

The presented implementation allows simulating the actual use of identity-based cryptography in ad hoc networks. The implementation is very simple, but it is also very powerful. The nodes can communicate without exchanging certificates, without the need to verify the validity of a certificate (time validity, issuer validity, revocation status) and thus without the presence of the CA (or in our case, the key generation center) (Shamir Adi, 1998). The fact that the key generation center can be and is turned off when the actual communication begins is another advantage, because in the network it would be a single point of failure. A drawback of this implementation is that it only assures message confidentiality. That is because when a node receives an encrypted message it can be sure that only itself and the expeditor know its content. But the expeditor's identity is proven only by the source IP address and this is unreliable. However, authentication can also be implemented through the presented agent by imaging a challenge-response mechanism: before node A and B will communicate they will identify each other. Node A will send to node B an encrypted message with the public key of B and will wait for B to send him back the same message, but encrypted for himself, witch means that B successfully decrypted the message, so B is who it pretends to be. Then, node B will do the same thing for node A. After these two steps, the nodes can safely communicate to one another. The man-in-the-middle attack is not possible when this authentication is used, because the node who would play this role would need the private key of one of the nodes, witch is very unlikely because it was stated that the private keys are distributed through a secure channel and with physical authentication. This is another disadvantage of this implementation: the private key issuance through a secure channel is not covered.

Still, it remains a good start that can be developed in the future by adding the authentication facility mentioned above, and also a no-repudiation mechanism.

REFERENCES

Boyen X., Martin L., 2007, *Identity-Based Cryptography Standard (IBCS) #1: Supersingular Curve Implementations of the BF and BB1 Cryptosystems*, Network Working Group, Request for Comments: 5091.

Cooks Clifford, 2001, *An Identity Based Encryption Scheme based on Quadratic Residues*, UK Crown.

Leiming Xu, 2001, *How to Add a New Protocol in NS2*.

Oliveira L.B., Aranha D., Morais E., Daguano F., Lopez J., Dahab R., 2007, *TinyTate: Identity-Based Encryption for Sensor Networks*, University of Campinas

Ros F.J., Ruiz P.M., 2004, *Implement a New Manet Unicast Routing Protocol in NS2*.

Shamir Adi, 1998, *Identity Based Cryptosystems and Signature Schemes*, Springer-Verlang.