A SECURITY MODEL
FOR MOBILE AD HOC NETWORKS

RICARDO STACIARINI PUTTINI

SUPERVISOR: RAFAEL TIMÓTEO DE SOUSA JÚNIOR

DOCTORAL THESIS IN ELECTRICAL ENGINEERING

PUBLICATION 004/2004

BRASILIA / DF: OCTOBER 2004
UNIVERSITY OF BRASILIA
FACULTY OF TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING

A SECURITY MODEL
FOR MOBILE AD HOC NETWORKS

RICARDO STACIARINI PUTTINI

DOCTORAL THESIS SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
OF THE FACULTY OF TECHNOLOGY OF THE UNIVERSITY OF BRASILIA, AS PART OF
THE REQUIREMENTS FOR OBTAINING THE TITLE OF DOCTOR

APPROVED BY:

Dr RAFAEL TIMÓTEO DE SOUSA JÚNIOR, University of Brasilia – DF – Brazil (UnB)
(SUPERVISOR)

Dr LUDOVIC MÉ, École Supérieure d’Électricité – France (Supélec)
(EXTERNAL EXAMINER)

Dr WILLIAM FERREIRA GIOZZA, University of Salvador – BA – Brazil (UniFacS)
(EXTERNAL EXAMINER)

Dr ANTÔNIO JOSÉ MARTINS SOARES, University of Brasilia– DF – Brazil (UnB)
(INTERNAL EXAMINER)

Dr CLÁUDIA JACY BARENCO ABBAS, University of Brasilia– DF – Brazil (UnB)
(INTERNAL EXAMINER)

Dr PAULO HENRIQUE PORTELA DE CARVALHO, University of Brasilia – DF – Brazil (UnB)
(DEPUTY)

PUTTINI, RICARDO STACIARINI
A Security Model for Mobile ad hoc Networks [Distrito Federal] 2004, 191 pp., 297 mm

Doctoral thesis – University of Brasilia, Faculty of Technology, Department of Electrical Engineering

1. Mobile ad hoc Networks 2. Distributed Digital Certification 3. Authentication in Mobile ad hoc Networks
4. Intruder Detection in Mobile ad hoc Networks

I. ENE/FT/UnB. II. A Security Model for Mobile ad hoc Networks

BIBLIOGRAPHICAL REFERENCE


TERMS OF COPYRIGHT

NAME OF AUTHOR: Ricardo Staciarini Puttini
TITLE OF DISSERTATION: A Security Model for Mobile ad hoc Networks.
LEVEL/YEAR: Doctor/2004

The University of Brasilia is granted permission to reproduce copies of this Doctoral Thesis and to loan or sell those copies solely for academic or scientific purposes. The author reserves all other publication rights and no part of this dissertation may be reproduced without his written permission.

Ricardo Staciarini Puttini
SQN 303 Bloco H Apto. 118 – Asa Norte
CEP 70735-080 – Brasilia – DF – Brazil
To my daughter Luiza
ACKNOWLEDGEMENTS

To my supervisor Prof. Dr. Rafael Timóteo de Sousa Júnior, for his constant support, encouragement, dedication and friendship, which were essential for the execution of this work and my development as a researcher.

To Prof. Ludovic Mé, of the École Supérieure d’Électricité (Supélec), the true coordinator of this work, for his wisdom and inestimable guidance and for his welcome and friendship during my stay in France. Merci beaucoup, Ludo!

To Prof. Paulo Henrique Portela de Carvalho and Prof. Antônio José Martins Soares, for their enriching discussions, cooperation and friendship.

To my family, especially my parents, Cláudio and Katea, my siblings Simone, Marcelo, Mônica and Solange, my darling Silvia and my beloved daughter, Luiza, for their constant love and affection.

To all at the UnB’s Department of Electrical Engineering and my colleagues at LabRedes, my heartfelt thanks.

This work was carried out with the support of CAPES, an agency of the Brazilian Government for encouraging professional and scientific training; the author was supported by a grant from the agency during the external research phase of this work, between September 2001 and December 2002.

To the Fundação de Empreendimentos Científicos e Tecnológicos (Foundation for Scientific and Technological Enterprise – FINATEC) for its support for my taking part in international conferences.
ABSTRACT

This thesis presents a model for securing mobile ad hoc networks (Manet). A combination of preventive and corrective security services allows the model to be used in a broad range of Manet application scenarios, including those where the probability of node compromising should not be neglected. Our model has three basic security services that interact with one another. First, a digital certification service allows a specific security policy to be applied to provide an effective means of identifying nodes and discrimination between trusted and untrusted network nodes. An authentication service ensures that messages originating from an untrusted node are treated in accordance with the security policy, enabling them to be simply discarded or processed with restrictions. Finally, an intrusion detection service deals with identification and elimination of compromised nodes. The proposal has been applied for securing routing and autoconfiguration protocols. An actual proof-of-concept implementation shows the effectiveness of the proposed services.
CONTENTS

1. INTRODUCTION .................................................................................................................. 1

1.1. MANET APPLICATIONS ................................................................................................. 3

1.2. VULNERABILITIES OF AD HOC NETWORKS ............................................................. 4

1.3. REQUIREMENTS FOR SECURITY SOLUTIONS IN AD HOC NETWORKS .............. 5

1.4. A SECURITY MODEL FOR MOBILE AD HOC NETWORKS ...................................... 7

1.5. ORGANISATION OF WORK .......................................................................................... 11

2. SECURITY IN MOBILE AD HOC NETWORKS STATE OF THE ART ...................... 12

2.1. MANET TRUST MODELS AND CERTIFICATION SERVICES ............................... 12

2.2. SECURITY OF ROUTING PROTOCOLS ........................................................................ 16

2.2.1. Routing protocols ....................................................................................................... 17

2.2.1.1. Ad hoc on-Demand Distance Vector Routing (AODV) ........................................ 21

2.2.1.2. Optimized Link State Routing Protocol (OLSR) .................................................. 21

2.2.1.3. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF) .............. 22

2.2.1.4. Dynamic Source Routing Protocol (DSR) .......................................................... 22

2.2.2. Security of routing protocols ....................................................................................... 23

2.3. SECURITY OF AUTOCONFIGURATION PROTOCOLS ................................................ 27

2.3.1. Autoconfiguration protocols ...................................................................................... 27

2.3.2. Allocation with duplicate-address detection ............................................................. 30

2.3.3. Best-effort allocation ................................................................................................ 31
2.3.4. Conflict-free allocation ................................................................. 32
2.3.5. Security of autoconfiguration protocols ........................................ 33
2.4. MANET INTRUSION DETECTION ...................................................... 34
3. MANET SECURITY MODEL ............................................................... 41
  3.1. VULNERABILITY AND ADVERSARY MODELS ................................ 41
    3.1.1. Vulnerabilities model .............................................................. 42
    3.1.2. Adversary model .................................................................. 43
    3.1.3. Security requirements ............................................................ 44
  3.2. SECURITY MODEL ...................................................................... 45
    3.2.1. Self-organising distributed services model .................................. 45
    3.2.2. Distributed trust model ............................................................ 46
      3.2.2.1. K-of-N trust model ............................................................. 48
      3.2.2.2. Identification of Manet nodes .......................................... 50
      3.2.2.3. Security policy considerations ........................................ 51
    3.2.3. Manet authentication extension (MAE) .................................... 53
    3.2.4. Detection of and response to intrusions in Manet ...................... 53
    3.2.5. Integrated security services .................................................... 56
4. CERTIFICATION AND AUTHENTICATION IN MANET ......................... 60
  4.1. CERTIFICATION SERVICE IN MANET ........................................... 60
    4.1.1. Basic Certification Services ...................................................... 62
4.3.2.3. MAE for the AODV Protocol .............................................................. 86
4.3.2.4. MAE for the DSR Protocol ................................................................. 86
4.3.2.5. Evaluation of MAE Protection ............................................................... 87
4.4. AUTHENTICATION OF AUTOCONFIGURATION PROTOCOL ...................... 88
4.4.1. Vulnerabilities of the DCDP Protocol ...................................................... 88
4.4.2. MAE for the DCDP Protocol ................................................................. 89
4.5. SECURITY POLICY AND CONFIGURATIONS FOR CERTIFICATION AND AUTHENTICATION SERVICES ................................................................. 90
5. INTRUSION DETECTION AND RESPONSE IN MANET ......................... 93
5.1. COMPLETELY DISTRIBUTED IDS ............................................................. 93
5.2. LIDS MODULAR ARCHITECTURE ............................................................. 95
5.2.1. Intrusion Detection Framework .............................................................. 96
5.2.2. Manet Environment Constraints ............................................................ 97
5.2.3. Messages generated by the LIDS ............................................................ 98
5.3. MISUSE INTRUSION DETECTION ............................................................ 99
5.3.1. Attacks and Attacks Signatures against the Routing Protocol ................. 101
5.3.1.1. Fabrication + Impersonation of HELLO Messages: ............................. 103
5.3.1.2. Fabrication of HELLO Messages: ...................................................... 105
5.3.2. Attack and Attack Signature against Distributed Applications ............. 107
5.3.3. Intrusion Response ............................................................................. 109
5.4. BEHAVIOUR INTRUSION DETECTION .................................................... 109
5.4.1. Mixed Distribution Models for Statistic Characterisation of Behaviour ..........112

5.4.1.1. EM Algorithm ..........................................................................................114

5.4.1.2. Principal problems in applying the EM algorithm and proposed solutions....117

5.4.1.3. Automatic estimation of the optimum order of the model.......................118

5.4.1.4. Detection algorithm ..................................................................................119

5.4.1.5. Detection algorithm for real time operation with GMM .........................120

5.4.1.6. Recursive updating of model adjusted parameters ...................................122

5.4.2. Characterization of Normal Traffic in a Manet and Creation of a Normal Behavior Model ........................................................................................................123

5.4.3. Detection of DDoS and Port Scanner Attacks .............................................125

5.4.4. Response to Intrusions ................................................................................127

6. EXPERIMENTS AND RESULTS ....................................................................128

6.1. EXPERIMENTAL PLATFORM .....................................................................128

6.2. NETWORK TOPOLOGY AND MOBILITY .................................................130

6.3. EXPERIMENTS WITH OLSR AND DCDP PROTOCOL VULNERABILITY ......131

6.4. MAE AND L-CERT.......................................................................................134

6.4.1. Experiment Parameters ..............................................................................135

6.4.2. Evaluation of Communication Overhead ..................................................137

6.4.3. Evaluation of Computational Overhead ......................................................142

6.4.4. L-Cert Performance Evaluation .................................................................146

6.4.5. Local CRL and Valid Certificate Cache Evaluation .................................147
6.5. L-IDS: Detection Due to Incorrect Use .................................................. 147

6.5.1. Performance Considerations ............................................................ 149

6.6. Security Evaluation .................................................................................. 150

6.7. L-IDS: Detection Per Behavior Modeling ............................................... 154

7. Conclusions ............................................................................................... 160

BIBLIOGRAPHICAL REFERENCES .................................................................. 165

Attachment I – IEEE 802.11b or Wi-FIAO AD-HOC Mode Wireless Network Technologies .................................................. 172

Attachment II – Threshold Encryption ......................................................... 174

Attachment III – Certification Protocol Message Syntax .............................. 176

Attachment IV – MANET Authentication Extension (MAE) Syntax .............. 177

Attachment V – XML (DTD) Specification for All L-IDS Messages ............. 178

Attachment VI – Experimental MIB for the OLSR Protocol in the ASN-1 Format .................................................................................. 180
INDEX OF TABLES

TABLE 2-1 – QUALITATIVE METRICS OF A MANET ROUTING PROTOCOL..........................18
TABLE 2-2 – QUANTITATIVE METRICS OF A MANET ROUTING PROTOCOL ......................19
TABLE 2-3 – COMPARISON BETWEEN THE SOLUTIONS FOR SECURING ROUTING PROTOCOLS ....26
TABLE 2-4 – QUALITATIVE METRICS FOR A MANET AUTOCONFIGURATION PROTOCOL.........29
TABLE 2-5 – QUANTITATIVE METRICS FOR A MANET AUTOCONFIGURATION PROTOCOL........29
TABLE 2-6 – MAIN PROPOSALS FOR DISTRIBUTED IDS................................................37
TABLE 2-7 – IDS DESIGNS USING MOBILE AGENTS ....................................................38
TABLE 4-1 – VULNERABILITIES OF THE OLSR PROTOCOL...........................................81
TABLE 4-2 - MAE OF MANET ROUTING PROTOCOLS..................................................87
TABLE 4-3 – TYPES OF AUTHENTICATION POLICY.......................................................91
TABLE 5-1 - ATTACK SIGNATURES AGAINST THE OLSR PROTOCOL ................................102
TABLE 5-2 – CHARACTERIZATION OF TRAFFIC GENERATED BY DDoS ATTACKS ............126
TABLE 5-3 – CHARACTERIZATION OF TRAFFIC GENERATED BY PORT SCANNER ............126
TABLE 5-4 – BEHAVIOR MODELS AND MONITORED VARIANTS.....................................127
TABLE 6-1 – ATTACKS IMPLEMENTED IN THE ATTACK PROGRAM ................................133
TABLE 6-2 – SIZE OF MAE ELEMENTS (OLSR AND DCDP)............................................137
TABLE 6-3 – AVERAGE NUMBER OF ADDRESSES ANNOUNCED IN HELLO MESSAGES
  (NEIGHBORS) AND TC (MS)..........................................................................................138
TABLE 6-4 – AVERAGE SIZE, IN BYTES, OF HELLO MESSAGE (NO MAE).......................138
<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-5</td>
<td>Average size (bytes) of HELLO messages – IPv4</td>
</tr>
<tr>
<td>6-6</td>
<td>Average size (bytes) of HELLO messages – IPv6</td>
</tr>
<tr>
<td>6-7</td>
<td>Average size (bytes) of HELLO messages – IPv4</td>
</tr>
<tr>
<td>6-8</td>
<td>Average size (bytes) of HELLO messages – IPv6</td>
</tr>
<tr>
<td>6-9</td>
<td>L-Cert communication overhead</td>
</tr>
<tr>
<td>6-10</td>
<td>Average number of sent and received messages (HELLO and TC)</td>
</tr>
<tr>
<td>6-11</td>
<td>Average time (ms) for RSA signature generation</td>
</tr>
<tr>
<td>6-12</td>
<td>Average time (ms) for RSA signature verification</td>
</tr>
<tr>
<td>6-13</td>
<td>Total processing overhead in relation to HELLO interval</td>
</tr>
<tr>
<td>6-14</td>
<td>Total processing overhead in relation to HELLO interval</td>
</tr>
<tr>
<td>6-15</td>
<td>Total processing overhead in relation to HELLO interval</td>
</tr>
<tr>
<td>6-16</td>
<td>Basic operations and L-Cert computational complexity</td>
</tr>
<tr>
<td>6-17</td>
<td>Average calculation time (ms) of L-Cert basic operations</td>
</tr>
<tr>
<td>6-18</td>
<td>L-Cert performance</td>
</tr>
<tr>
<td>6-19</td>
<td>Size of Cache (Kbytes)</td>
</tr>
<tr>
<td>6-20</td>
<td>Abstraction rules</td>
</tr>
<tr>
<td>6-21</td>
<td>Detection of Attacks against OLSR Protocol</td>
</tr>
<tr>
<td>6-22</td>
<td>Average time for Intrusion Response Time Start</td>
</tr>
</tbody>
</table>
INDEX OF FIGURES

FIGURE 1-1 – SECURITY MODEL APPLIED TO ROUTING AND AUTOCONFIGURATION SERVICES .....9

FIGURE 2-1 – CLASSIFICATION OF MANET ROUTING PROTOCOLS .................................20

FIGURE 2-2 – DCDP – ASSIGNING A NEW IP ADDRESS TO A NEW NODE .......................33

FIGURE 2-3 – IDWG INTRUSION DETECTION FRAMEWORK ......................................35

FIGURE 2-4 – TAXONOMY OF INTRUSION DETECTION SYSTEMS ..............................36

FIGURE 3-1 – MODEL OF DISTRIBUTED, COLLABORATIVE AND SELF-ORGANISING SERVICES......46

FIGURE 3-2 – COLLABORATION PROTOCOL FOR DISTRIBUTED CERTIFICATION SERVICES .......49

FIGURE 3-3 – COLLABORATION PROTOCOL FOR INTRUSION RESPONSE ..........................55

FIGURE 3-4 - IMPLEMENTATION VIEW OF THE SECURITY MODEL ....................................57

FIGURE 3-5 – PROTOCOL ARCHITECTURE OF THE SECURITY MODEL ............................57

FIGURE 3-6 - IMPLEMENTATION VIEW OF THE EXTENDED SECURITY MODEL .........................59

FIGURE 4-1 – ISSUE OF PARTS OF THE PRIVATE KEY ..................................................71

FIGURE 4-2 – FABRICATION OF HELLO MESSAGES ....................................................81

FIGURE 4-3 – FABRICATION + IMPERSONATION OF HELLO MESSAGES ..........................82

FIGURE 4-4 – FABRICATION OF TC MESSAGES ..........................................................82

FIGURE 4-5 – MODIFICATION + IMPERSONATION OF TC MESSAGES ..............................83

FIGURE 4-6 - PARAMETERS FOR THE AUTHENTICATION AND CERTIFICATION POLICY ........92

FIGURE 5-1 – MODULAR ARCHITECTURE OF THE LIDS .............................................96

FIGURE 5-2 - ATTACK SIGNATURE: FABRICATION + IMPERSONATION OF HELLO MESSAGES: ..........................................................105
Figure 5-3 - Attack Signature: Fabrication of HELLO messages: .................... 107

Figure 5-4 - Attack Signature: Stepping Stone for Telnet Connections ............ 109

Figure 5-5 - \( \Pi \) for a cluster with unidimensional Gaussian distribution .......... 121

Figure 5-6 - \( \Pi \) for a cluster with bi-variant Gaussian distribution and diagonal covariance matrix ................................................................. 121

Figure 6-1 – Implemented L-IDS Modular Architecture ................................. 149

Figure 6-2 – Example of Experimental MANET Topology ............................ 151

Figure 6-3 – Behavior model with 03 clusters and recognition of new datum reflecting normal behavior ................................................................. 155

Figure 6-4 – Behavior model with 03 clusters and recognition of new datum reflecting abnormal behavior ............................................................ 155

Figure 6-5 – Simulation generation process .................................................. 157
# LIST OF ACRONYMS

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAFID</td>
<td>Autonomous Agents For Intrusion Detection</td>
</tr>
<tr>
<td>CA</td>
<td>Certification Authority</td>
</tr>
<tr>
<td>DCA</td>
<td>Distributed Certification Authority</td>
</tr>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>AODV</td>
<td>Ad hoc On-Demand Distance Vector</td>
</tr>
<tr>
<td>ARAN</td>
<td>Authenticated Routing for Ad hoc Networks</td>
</tr>
<tr>
<td>AREQ</td>
<td>Address Request</td>
</tr>
<tr>
<td>AREP</td>
<td>Address Reply</td>
</tr>
<tr>
<td>CBAP</td>
<td>Certificate Revocation List</td>
</tr>
<tr>
<td>DAD</td>
<td>Duplicated Address Detection</td>
</tr>
<tr>
<td>DDoS</td>
<td>Distributed Denial of Service</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Configuration Protocol</td>
</tr>
<tr>
<td>DS</td>
<td>Digital Signature</td>
</tr>
<tr>
<td>DSDV</td>
<td>Destination-sequenced Distance-vector</td>
</tr>
<tr>
<td>DCDP</td>
<td>Dynamic Configuration Distribution Protocol</td>
</tr>
<tr>
<td>FSD</td>
<td>Finite State Diagram</td>
</tr>
<tr>
<td>DSR</td>
<td>Dynamic Source Routing Protocol</td>
</tr>
<tr>
<td>MSS</td>
<td>Mobility Support Station</td>
</tr>
<tr>
<td>p.d.f.</td>
<td>Probability density function</td>
</tr>
<tr>
<td>c.d.f.</td>
<td>Cumulative density function</td>
</tr>
<tr>
<td>FSR</td>
<td>Fisheye State Routing</td>
</tr>
<tr>
<td>GMM</td>
<td>Gaussian Mixture Model</td>
</tr>
<tr>
<td>HC</td>
<td>Hash Chain</td>
</tr>
<tr>
<td>HNA</td>
<td>Host and Network Association</td>
</tr>
<tr>
<td>IDA</td>
<td>Intrusion Detection Agent</td>
</tr>
<tr>
<td>IDS</td>
<td>Intrusion Detection System</td>
</tr>
<tr>
<td>IDWG</td>
<td>Intrusion Detection Working Group</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers.</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force.</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPSec</td>
<td>IP Security</td>
</tr>
<tr>
<td>IREQ</td>
<td>Initiator Request</td>
</tr>
<tr>
<td>K_{CA}</td>
<td>CA’s private key</td>
</tr>
<tr>
<td>K_{UC}</td>
<td>CA’s public key</td>
</tr>
<tr>
<td>L-Cert</td>
<td>Local Certification Service</td>
</tr>
<tr>
<td>L-IDS</td>
<td>Local Intrusion Detection Service</td>
</tr>
<tr>
<td>L-SPM</td>
<td>Local Security Policy Management Service</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>MAC (1)</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>MAC (2)</td>
<td>Message Authentication Code</td>
</tr>
<tr>
<td>MAE</td>
<td>Manet Authentication Extension</td>
</tr>
<tr>
<td>MAIDS</td>
<td>Mobile Agent Intrusion Detection System</td>
</tr>
<tr>
<td>Manet</td>
<td>Mobile Ad hoc Networks</td>
</tr>
<tr>
<td>MIB</td>
<td>Management Information Base</td>
</tr>
<tr>
<td>MID</td>
<td>Multiple Interface Declaration</td>
</tr>
</tbody>
</table>
MPR – Multipoint Relay
MS – MPR-Selector Set
NIST – National Institute of Standards and Technology
OLSR – Optimized Link State Routing Protocol
PAN – Personal Area Network
PCA – Principal Component Analysis
PDA – Personal Digital Assistant.
PGP – Pretty Good Privacy
PKIX – Public Key Infrastructure (PKI) based on X.509
RF – Radio Frequency
RFC – Request for Comments
RREQ – Route Request
RREP – Route Reply
RREP-ACK – Route Reply Acknowledgment
RSA – Rivest-Shamir-Adelman
SAODV – Secure AODV
SPARTA – Security Policy Adaptation Reinforced Through Agents
SEAD – Secure Efficient Ad hoc Distance vector
SNMP – Simple Network Management Protocol
SPR – Secure Routing Protocol
SSL – Secure Sockets Layer
TBRPF – Topology Dissemination Based on Reverse-Path Forwarding
TC – Topology Control
TCP – Transmission Control Protocol
TLS – Transport Layer Security
TTL – Time To Live
UDP – User Datagram Protocol
ZRP – Zone Routing Protocol
1. INTRODUCTION

Mobile ad hoc networks (Manets) are wireless networks whose mobile nodes exchange information without the help of a predefined network infrastructure [26]. In these networks, also known as spontaneous networks, the nodes communicate directly with one another, in a peer-to-peer communications architecture. Since there is no supporting infrastructure, the routing services are established cooperatively and each node taking part in the network acts as a possible router for the others. Thus, when a node needs to communicate with another that is not within reach of its link, it relays its packets via a neighbouring node that is closer to the destination node, which in turn relays the packet onward. Therefore, Manets are multi-hop mobile networks in which the connectivity between nodes is ensured by collaborative routing [24,100].

A Manet essentially consists of mobile nodes with one or more wireless network interfaces. In general, wireless links will continue to have a considerably lower capacity than wired ones. This is due not only to the differences and limitations in terms of the nominal throughput of an interface (i.e. the maximum transfer rate of the radio link is generally lower than the nominal rate of wired links), but also to other factors peculiar to networks using wireless transmission, such as: multiple-access effects, fading, noise and interference from electromagnetic sources outside the system, etc. Furthermore, mobile nodes have to be powered by portable power supplies (batteries) which run out over time. Thus, important criteria for optimising the resources and services planned for Manet are efficient use of the available bandwidth and power.

In a Manet the nodes may, continually and at any time, appear, disappear or move around within the network. As a result, the nodes of the Manet are constructed dynamically and the network topology is subject to frequent and unforeseeable changes. This characteristic linked to the mobility of the nodes, allied to the limited trust and bandwidth of wireless links, means that the availability of a specific node cannot be ensured. In this way, the services in a Manet cannot be concentrated in centralised entities. Conversely, and like routing services, the services in a Manet must be provided in a distributed and self-organising manner, by collaboration between network nodes. This collaboration normally makes use of the natural

1. Mobile Ad hoc NETwork.
redundancies resulting from the communication model which, to a certain extent, compensates for the lack of certainty regarding the availability of individual nodes.

The Manet presented above is characterised by two basic services necessary to form these networks: routing [25,58,83,88] and autoconfiguration [80,81,87]. The routing service is related to the multi-hop nature of Manets. Thus, the routing protocol must be designed to take account of the constant changes in the network topology depending on the mobility of the nodes. Whereas the autoconfiguration service is related to the association of the nodes to the network, allowing rapid set-up with little or no user intervention.

This work presents a new security model designed to meet the specific requirements of ad hoc network environments. Its aim is to define and develop a set of integrated security services that are fully distributed and can be provided through collaboration between the nodes of a Manet. Besides this, this set of services must be flexible to the point of allowing various levels of security to be defined for a variety of Manet application contexts. The proposed model is directly applied and developed to provide security for the services essential in a Manet, namely routing and autoconfiguration.

Security in ad hoc networks is still a recent subject in the specialist technical literature, although many works have been published on the subject in recent years. Most of these initiatives consist of isolated proposals dealing with each security issue piecemeal, focusing on the provision of security alternatives applying to a specific protocol (such as a routing protocol) [28,41,48,49,84] or a particular context of use [7,36,91]. This work differs from those initiatives by proposing a security solution applicable to the ad hoc context as a whole, rather than being aimed at providing security for a specific protocol, and so as to provide levels of security that are adaptable to the requirements of the Manet application, enabling it to be used in contexts with differing security policies.
1.1. MANET APPLICATIONS

There are various demands for current and future ad hoc network technology [26]. The emerging field of mobile and nomadic computing, currently with emphasis on operation by Mobile IP\(^2\), is gradually evolving and is beginning to require highly adaptable network technologies that can manage multi-hop clusters of ad hoc networks that operate automatically or connected to the internet at one or more points. In particular, the use of Manet technologies is related to spontaneous network formation. Indeed, the concept of self-organisation makes ad hoc networks a flexible alternative for network formation, allowing mobile networks to be set up quickly without the need to install infrastructure beforehand. Within this context, there are many scenarios for using Manets in commercial, industrial, academic, governmental or military applications, including:

- Inter-group communication and cooperative work: dynamic formation of collaborative work groups, in business, academic and commercial environments, among others.
- Personal Area Networks (PAN): establishment of network communication for small-scale environments via peer-to-peer communication, eliminating or reducing the need to install network connection and interconnection devices.
- Working at sites with no infrastructure or where the infrastructure has been destroyed: application in scenarios where communications have to be established rapidly over dynamic networks and are critical for survival, such as rescue operations at accident or sabotage sites, fires, collapse, maintenance at remote sites, etc.
- Sensor networks: networking of various sensors, which may be moving, for exchange and processing of information relating to the measurements being made.
- Moving networks: networks consisting of moving systems, such as aircraft, road vehicles or troops in the battlefield.

Another area that lends itself to Manet technologies is pervasive communication and the establishment of networks that can be accessed from anywhere. Thus, link-based mobile ad hoc networks can be operated as a robust and inexpensive alternative or supplement to cellular mobile networks.

---

\(^2\) The charter of the IETF’s MobileIP working group can be found at: http://www.ietf.org/html.charters/mobileip-charter.html
1.2. VULNERABILITIES OF AD HOC NETWORKS

Many of the vulnerabilities suffered by conventional network architectures also apply to Manets. Meanwhile, some of the special characteristics of Manets emphasise these vulnerabilities, since they offer new ways in which they can be exploited. Apart from this, Manets have their own peculiar vulnerabilities which do not affect other network architectures [66,119]. The following stand out among the special characteristics of Manets that emphasise known vulnerabilities in conventional networks or which involve new vulnerabilities specific to the ad hoc context:

- the wireless nature of the link service – the nodes can monitor the use of the network through adjacent nodes that are within range of their receiver\(^3\);
- the decentralised peer-to-peer communication model – the nodes can communicate directly with one another;
- mobility – the network topology changes dynamically;
- the collaborative communication model – the nodes depend upon one another to establish and maintain network connectivity; and
- frequent use of power sources that discharge in use – mobile nodes use portable power sources.

These characteristics make Manets more vulnerable than wired networks to a broad spectrum of attacks, such as passive listening, spoofing (where one entity assumes the identity of another) and denial of service; an adversary can therefore exploit these to:

- listen promiscuously to transmissions from nearby nodes;
- communicate directly with any node that is within its transmission range;
- move to collect information on the activity of other more distant nodes or to escape monitoring by nearby nodes;
- practise non-cooperation (e.g. to save its own battery power or to provoke dysfunctions in relaying of network packets); and

---

3 This work uses the definition of Manet given in RFC 2501 [26]. Thus, access to the data link service is assumed for each node in the Manet. We do not elaborate on the security of the data-link level, either in the analysis of vulnerabilities or in the definition of the security services. In this respect, if the link service uses some kind of authentication or other cryptographic protection (e.g. WEP protocol in IEEE 802.11 networks), it is assumed that a compromised or malicious node is capable of authenticating itself and encrypting/decrypting data link tables and hence also able to monitor its neighbours’ communications.
• provoke unnecessary activity so as to discharge other nodes’ power sources more quickly.

Additionally, in conventional networks, services such as routing and autoconfiguration are delegated to entities designed to be secure (e.g. routers and autoconfiguration servers). These entities perform a controlled set of functions and have a special position in the network topology that affords them effective protection. Thus, these entities present a reduced set of vulnerabilities, since they do not have generic functions, unnecessary functions can be deactivated and they can activate protection, logical or physical, relating to their position within the network architecture, generally at points of concentration or centralisation within controlled parts of the network. In Manets, on the other hand, the basic services, as well as the other network services, are provided in decentralised form and with potential participation by all the network nodes. These nodes are often implemented in computer equipment using generic hardware and software that is subject to a series of vulnerabilities related to their operating system, software bugs, backdoors, viruses, etc. Furthermore, a Manet node that does not have proper physical protection may be captured [71].

Thus it is not uncommon to have malfunctioning or compromised entities in the network. Regarding the existence of incorrect entities, a node that is making attacks on the Manet can even move, either to attack other parts of the network or to escape monitoring by its neighbours. This characteristic makes it difficult to detect attacks and to distinguish the incorrect node(s) from the other correct nodes in the network.

1.3. REQUIREMENTS FOR SECURITY SOLUTIONS IN AD HOC NETWORKS

The security solution considered in this work involves the definition of an integrated set of security services for Manets, providing protection for the other network services. In this context, two types of requirements are considered in this section. First, we present the requirements to be satisfied, in general, by all the network services of a Manet, including for the security services to be defined for the operation of the network. Next, we present the actual security requirements, so enabling us to identify and design the various security services considered in our model.

An analysis of the characteristics peculiar to Manets enables a set of basic requirements to be established for consideration in the design of any service for this type of
network. Next, we present the main features of Manets and the requirements for networks services deriving from them:

- No concentration points and no guarantee of the availability of individual nodes: Manet services must follow a distributed approach.
- Mobility and dynamic network topology: the distribution must be self-organising and cooperative, so as to avoid interruptions in the services when node connectivity changes and to take advantage of redundancies in the communication model to optimise the availability of the service.
- Bandwidth and power-supply limitations: the services must not generate excessive overheads in the network, so that services can be provided locally wherever possible, avoiding relaying and retransmission of messages.

In general, the above requirements must also be satisfied when designing security services. Note that, with the exception of some security services that execute in isolation in the local host, most security techniques and protections used in traditional networks are not suited to ad hoc networks [52,66,119]. In conventional architectures, services with access control, authentication, authorisation, monitoring and security management are associated with clearly defined devices, such as authentication servers or firewall systems. These components cannot exist as individual devices in Manets. Therefore, the security services in these networks have to follow a distributed approach through cooperation and self-organisation. Moreover, wherever possible this cooperation must be completely local, so restricting the communication and processing overheads to the area around the nodes involved.

Various approaches are presented for the definition of security requirements in Manets [7,18,28,36,41,48,49,52,66,84,91,119]. In this work, we consider the combination of two basic requirements: distinguishing trusted and untrusted Manet nodes, and the identification and subsequent isolation of compromised or misbehaving nodes.

Nodes are distinguished by the definition and application of a trust model that lays down conditions for nodes to join the network. Joining the network, defined as a relationship of mutual trust between nodes, is required of nodes as a first line of defence for cooperative services, requiring the nodes to be first members of the network and restricting cooperation to nodes which are network members. In this scenario, control and packet-relaying information is exchanged only among the set of nodes that have established mutual trust. The prior requirement for explicit joining of the network is justified, in particular, by the promiscuous nature of wireless communications, whereby any device configured with a wireless interface
can communicate over the network. This ease of access characteristic of wireless networks is even more critical in Manet scenarios, owing to their spontaneous nature. Once membership has been established, a node must be able to the other members that it has joined, just as it must be possible for the other network nodes to check that its membership is valid.

Another fundamental requirement relates to the existence of malfunctioning or compromised entities in a Manet. Since the occurrence of such entities cannot be neglected, the security services must be designed to remain robust even in the presence of compromised or misbehaving nodes. In this respect, we can only allow degraded performance caused by the presence of nodes that are incorrectly integrated with the network to be temporary, and incorrect nodes must be identified and isolated from the cooperative services before the robustness of the service is compromised.

Finally, as we have said above, there are a number of contexts for applying Manets. Different contexts will certainly require different levels of security. Thus, a Manet set up to enable cooperative working in a classroom requires different levels of security from a Manet set up to provide communication and information services for a rescue operation in a disaster area. Similarly, the security levels would be even stricter in a Manet set up between troops manoeuvring on a battlefield. In general, these requirements can be expressed in terms of a security policy that specifies the levels of security required in each case. The aim of this work is therefore to ensure that security services can be quickly adapted to the security policy defined for each specific application context.

1.4. A SECURITY MODEL FOR MOBILE AD HOC NETWORKS

This work consists of the definition of a security model able to reduce or eliminate the vulnerabilities of a Manet, whether they be common to other network types or peculiar to the Manet context. To this end, the proposed security model defines a set of integrated security services, provided in accordance with the requirements imposed by the Manet architecture, discussed in the previous section. Thus, the first contribution of this work is a proposal for a cooperation model that is compatible with peer-to-peer architecture, retaining distributed self-organisation as fundamental characteristics of the design and development of services in Manets. Therefore, all the security services discussed are designed in accordance with this model, with absolutely no need for centralised entities, even at the stage of bootstrapping the network.
The second contribution of this work is the analysis of vulnerabilities leading to the identification of security requirements for the Manet context and the definition of a set of security services to be developed, along with the interaction between those services. In particular, in the security model presented in this work, we propose mechanisms for interaction between preventive and corrective security services. While preventive security services prevent attacks from occurring or attempted attacks from being successful, corrective security services cancel out the undesirable effects of attacks by eliminating previously trusted nodes which have become compromised or which begin to misbehave.

An authentication service based on a distributed certification service [93] provides a basic preventive security solution, allowing trusted nodes to be distinguished from untrusted ones, whereas a distributed intrusion detection system (IDS) [95] provides a corrective security solution, by feeding information on misbehaving nodes back to the certification system, allowing those nodes to be isolated as the main counter-measure. This interaction between preventive security services (e.g. certification) and corrective services (e.g. IDS) is the most important characteristic of this work [96,97], since most of the related published works [28,41,48,49,52,66,84,119] have set themselves the objective of proposing a specific security service, usually preventive, based on cryptographic authentication techniques.

Moreover, the proposed security model is directly applied to provide security for routing [25,58,83,88] and autoconfiguration [80,81,87] protocols, as shown in Figure 1-1. The definition of a Manet Authentication Extension (MAE) enables the messages of the routing and autoconfiguration protocols to be used unchanged. This is very important, since the Manet task force of the Internet Engineering Task Force (IETF) recently concluded the definition phase for the Internet experimental standards for Manet routing protocols, and the security aspects were not directly considered in these initial versions. Local certification services (L-Cert) and intrusion detection services (L-IDS) are provided collaboratively in all Manet nodes. In order for collaboration to take place, it is sufficient for there to be communication between the nodes. Self-organisation is a direct consequence of the possibility of initiating immediate collaboration at any time and with any nodes. Finally, to maintain a

---

4 Intrusion Detection System.
5 Manet Authentication Extension.
6 In 2003, after nearly four years of work on the subject, the IETF’s Manet task force defined protocols AODV [88], OLSR [25], DSR [58] and TBRPF [83] as experimental Internet standards for Manet routing protocols. For more details, the official charter of the Manet task force can be accessed at http://www.ietf.org/html.charters/manet-charter.html.
robust solution, there is a requirement for a minimum (threshold) number of nodes to agree (concept of security policy) and cooperate (concept of coordinated communication) for a new node to be admitted to the network (certification service) or for an accusation against a compromised node to initiate a response (intrusion detection system).

The final contribution of this work is the design and analysis of these security services (i.e. the distributed certification and authentication service as preventive mechanisms, and the distributed intrusion detection system as a corrective mechanism) and their interaction in the context of securing the routing and autoconfiguration protocols.

As regards the design of the certification service, the certification functions, typically performed by certification authorities (CAs) in conventional architectures, are distributed between the members of a Manet [66,119]. A minimum (threshold) number of the nodes have to cooperate to provide collaborative certification services. These services are established using threshold cryptography, originally proposed by Shamir [102]. This technique was used originally to define a distributed certification scheme adapted for the Manet context by L. Zhou et al. [119,120], which was later supplemented by Kong et al. [66]. In this work we extend and correct [66], with the following contributions [93]:

- proposal for a mechanism for bootstrapping the service, obviating the need for a centralised node (dealer) when setting up the network described in [66,119]. Our mechanism was developed simultaneously with [62] but has different design and operating principles.
- approach based on a security policy, with a definition of objective criteria for operation of the certification service, correcting Sybil’s vulnerabilities to attack
[32], and enabling the service to be adapted to contexts with different security requirements.

- complete design of certification protocols, including definition of message syntax, sequence and semantics, specification of valid certificate cache mechanisms, and local certificate revocation list (CRL\(^7\)), plus the full implementation of the validation protocol.

The authentication service is provided by the Manet Authentication Extension (MAE) [93], which is designed to provide authentication in message-oriented services\(^8\), such as routing and autoconfiguration protocols. The contribution of this work consists of defining that extension independently of the routing or autoconfiguration protocol adopted. This is an alternative approach to recent work that proposes security protocols AODV [28,41] and DSR [48,49] by modifying the syntax or even the mechanisms of the protocols in question.

Finally, as regards the intrusion detection system, we propose a new IDS architecture adapted to the requirements of the Manet context, with the following characteristics [94,95]:

- fully distributed intrusion detection process, allowing collaborative detection at any phase of detection (data collection, data analysis or alert management);
- self-organisation, through the use of an agents platform;
- modular architecture, allowing simultaneous use of intrusion detection techniques for misuse and for behaviour modelling\(^9\), and the use of various auditing data sources during the detection process.

Even though intrusion detection is a topical issue in the literature on network security, few works have in actually been published about intrusion detection in the Manet context, for example [42,51,73,79,111,117].

The IDS designed was implemented and assessed for detection of attacks at network level (routing protocol) and application level (attack through chain of telnet sessions or stepping-stone attack), using information from the Management Information Base (MIB) and analysis of selected network packets.

\(^7\) Certificate Revocation List.

\(^8\) For connection-oriented services, the SSL/TLS protocols can be used.

\(^9\) There are two classic aspects to intrusion detection [29]: detection by misuse, where attacks are identified by recognising patterns previously related to attacks and formally defined in the form of a signature, and detection by behaviour, where operating conditions that differ significantly from previously modelled normal operating conditions.
1.5. ORGANISATION OF WORK

This work is organised as follows: Chapter 2 is devoted to a presentation of the state of the art in Manet security, focusing mainly on the contributions relating to certification, intrusion detection and security of routing and autoconfiguration protocols. In Chapter 3, we present the proposed security model, together with a vulnerability analysis and the definition of security requirements underlying the concept of the model. Chapter 4 contains a detailed description of the preventive security services, namely the Manet certification and authentication services, the latter with an analysis of its application to the four Manet routing protocols defined as experimental internet standards (i.e. AODV, OLSR, TBRPF and DSR) and the Dynamic Configuration Distribution Protocol (DCDP), studied in the context of this work. Chapter 5 deals with the presentation of the intrusion detection system and the design of the signature of attacks against the nodes of a Manet, with special emphasis on attacks carried out against the routing protocol. This Chapter also presents the intrusion response mechanism, providing the corrective security service. Chapter 6 sets out the experiments carried out and results achieved with the aim of validating the proposed model, comprising both experiments conducted on the basis of real implementations of security services, and experiments under simulation used essentially to validate the effects of node mobility on the performance of the planned services. Lastly, Chapter 7 concludes the thesis with final considerations and future work.
2. SECURITY IN MOBILE AD HOC NETWORKS STATE OF THE ART

Manet security is an issue that has been broadly discussed in the recent specialised literature, since the context and problems are new, with many points still open. In general, the published works and ongoing efforts to define security techniques for these networks can be divided into two groups: definition of trust models [7,17,18,36,52,62,66,71,91,119], allowing trusted network nodes to be distinguished from untrusted ones, and security of protocols related to the basic services of this kind of network, with special emphasis on security of routing protocols [28,41,48,49,84]. On the other hand, there are few publications to date on corrective security techniques, such as intrusion detection systems specifically designed for Manet, despite the recent appearance of a few works on this important topic [42, 51,73,79,111,117].

In this chapter, we discuss the main results presented in the literature on Manet security, covering the topics relating to the definition of trust models, with routing and autoconfiguration security protocols, and with the design of intrusion detection systems in these network environments. At the same time, this chapter aims to sketch out the state of the art in mobile ad hoc networks and situate the contributions of this work in relation to other related work.

2.1. MANET TRUST MODELS AND CERTIFICATION SERVICES

In general, the main security proposals for protocols and services in mobile ad hoc networks use the concept of logical separation of the network into trusted and untrusted nodes. In this way, the network nodes must generally establish a relationship of trust between one another before using security services and protocols. Moreover, once trust has been established, i.e. when the nodes agree to trust one another, this has to be established formally and in a verifiable way. This can be done by distributing tokens [116], for establishing security membership (shared cryptographic keys) [48,49,84], for the use of digital certificates in schemes similar to a public key infrastructure (PKI) [28,41], or any other form of verifiable expression of the relationship established.

Conventional certification architectures include Kerberos [65], standard X.509 [46] and PKIK [4]. In these standards, two entities authenticate each other via a certification...
authority. This type of architecture only works properly, however, in networks with a defined infrastructure. In *ad hoc* networks, it does not work satisfactorily owing to the following:

- The high cost of maintaining centralised servers in a large-scale *ad hoc* network;
- The CA servers in an *ad hoc* network are vulnerable to attack;
- The mobility of the nodes requires constant authentications, which creates problems of scalability and congestion at the CA servers;
- Multi-hop communication over an error-prone wireless channel exposes data transmissions to high error rates.

Some variations, such as hierarchical CAs and delegated CAs [71], can reduce but not solve the problem of the robustness of the system and the availability of services within the network.

The design of Manet security services must be based on a distributed approach, with self-organisation and localisation. Two initiatives stand out as alternatives for this type of service. J. Hubaux *et al.* [17,18,52] present a proposal for a model in which relationships of trust are established between pairs of nodes (peer-to-peer), in a scheme similar to the PGP (Pretty Good Privacy) system [121]. Each node generates locally a public/private key pair. The private key is then used to sign certificates for other trusted nodes, while the public key is used to verify those certificates. A node must have different certificates linking its public key to its identity, each one being signed by another node of the Manet that trusts it. The certification services is distributed through the use of locally maintained repositories of certificates, which store the certificates of the nodes located nearby. The proposal also presents the algorithms for building and updating these repositories. The strong point of this approach is the adoption of a clearly peer-to-peer trust model, which does not require the use of any external entity, even to bootstrap the service. The scheme is therefore naturally self-organising.

The certificates are validated by establishing multiple certification chains (i.e. the certification route from one public key to another) from the public key of the node carrying out the validation for the public key that is being validated. To quantitatively assess the trustworthiness of the validation process, authentication metrics are designed and used, assigned to each chain of certificates.

This approach has two important vulnerabilities, however, that hinder its use in more generic scenarios with more restrictive security requirements. First, the use of authentication metrics is useful for dealing with misbehaving nodes issuing false certificates. This technique is not very successful, however, for exposing valid certificates from compromised nodes,
since the certification metrics are designed to combat incorrect issue of certificates, but not correctly issued certificates. Finally, this technique is not resistant to Sybil attacks [32], where one node forges multiple identities to build fictitious certification chains and distributes these certificates to nearby nodes, so increasing the standard values for the authentication metrics in the validation process. Thus, by winning the trust of only one trusted node or even compromising a single legitimate network node, a node can win the trust of the whole network.

The alternative proposed by P. Hubaux consists of defining a distributed certification authority (DCA) [66,71,119]. In this approach, the CA’s private key ($K_{CA}$) is used to sign certificates for all the nodes in the Manet. A certificate signed with $K_{CA}$ can be quickly verified by using the CA’s public key ($K_{UCA}$), known to all the network nodes. The facilities and services provided by the CA are distributed by sharing the private key between the network nodes by using threshold cryptography techniques [102]. Each node of the Manet ($v_i$) maintains a part of the private key ($K_{CA,i}$) and any coalition of $K$ (a system constant, usually defined in terms of the average number of neighbours in the network) holders of parts of the private key can collectively perform the CA function. The $K_{CA}$ cannot be recovered by any of these nodes, however. Certificates are revoked by the issue of counter-certificates, which also have to be signed with the $K_{CA}$. Thus, a list of revoked certificates can be maintained locally by each Manet node, caching all the counter-certificates issued. Obviously, $K$ holders of parts of the private key need to agree among themselves before a counter-certificate can be issued against any node. This duality balances the issue and revocation of certificates, both processes being conditioned by the size of the coalition ($K$).

Self organisation is achieved for the definition of a protocol for the dynamic establishment of coalitions of $K$ holders of parts of the private key. These coalitions are formed for the provision of three basic services: (1) signing of certificates, used in the issue, renewal and revocation of certificates; (2) issue of new parts of the private key, used to initiate nodes joining the network; and (3) in the updating of the parts of the private key, which must occur periodically to prevent an adversary being able to gradually compromise various Manet nodes until the system breaks down once the $K$th node is compromised. This last service is directly related to the concept of tolerance to intrusion, in that the security of the whole system depends on an intrusion detection mechanism capable of tracing and eliminating misbehaving network nodes (by revoking certificates), before a single adversary or a group of cooperating adversaries, succeed in compromising $K$ different nodes.
Such a certification service based on a DCA was originally proposed by L. Zhou and Z. J. Haas [119]. In this original proposal, the holders of the parts of the private key are restricted to a set of “special” nodes which have to be initiated previously. This means that the system is not completely self-organising, as a centralised dealer has to distribute the parts of the private key off-line to the holder nodes. A scheme for updating the parts of the private key was also proposed. In a later contribution, J. Kong et al. [66] propose the generalisation of this original model, enabling any node holding a valid certificate to take part in the DCA services. In this scenario, a node that does not have part of a private key may receive one from the other $K$ network nodes which already hold parts of the private key. This procedure reduces the network bootstrap requirements, since the centralised distribution of the parts of the private key is required only for initiating the first $K$ nodes of the Manet. Finally, reference [71] presents an improved version of the procedure for updating the parts of the private key.

Another difference between these works relates to the certificate issuing and revocation policy. In reference [119], new certificates have to be issued via an out-of-band procedure, normally performed by a centralised CA that is also responsible for distributing the parts of the private key. Although this centralised CA is not needed for renewing or revoking certificates, it must always be present to issue certificates for nodes that do not have valid ones. In references [66,71], new certificates are issued by the DCA (i.e. any coalition of $K$ holders of parts of the private key). Whereas in [66] no mention is made of the policy for the issue of certificates to nodes without valid certificates, in [71] a request for the issue of certificates is always signed, unless there is some explicit restriction on the requesting node (e.g. a counter-certificate). This policy makes the system vulnerable to Sybil attacks [32] since, by forging multiple identities, a node can easily obtain $K$ valid certificates and, therefore, $K$ different parts of the private key, thereby definitively breaking the system.

Regarding certification services, the proposal of this work [93] is based on [66,71], with the following contributions:

- The issue and renewal of certificates in [66,71] follows some pre-established rule. In this work, rather than proposing any inflexible rule, we adopt an approach to the issue and renewal of certificates in accordance with the security policy, which can be translated into a set of configurable parameters and options, so making the issue and renewal of certificates, and the issue and updating of parts of the private key, flexible and adaptable to various scenarios of Manet use. This approach to the definition of a security policy also allows us to define appropriate conditions for bootstrapping new networks, correcting the vulnerabilities relating to Sybil attacks.
The proposals in [66,71] do not specify how valid certificates are acquired and stored in each node. Furthermore, the local CRL is built up gradually, with counter-certificates accumulating and being signed and immediately flooded to the whole network, while the synchronisation of new nodes joining the network, which need to initiate their empty CRLs, is not mentioned. In this work, we propose a set of mechanisms for localised distribution and maintenance of the local databases containing the certificates which are valid (certificate cache) or revoked (CRL). These mechanisms also have a number of configurable options which enable the database synchronisation services to be adapted to the other policies for routing and distributing information over the network. These options include the choice of proactive or on-demand methods for building and maintaining the database of valid certificates and the CRL, the definition of timers for maintaining valid certificates without using a cache, and maximum sizes for the cache of valid certificates.

References [66,71] consider the use of a single DCA. In this work, we extend that model to support multiple DCAs, with the respective mechanisms for building certification paths. This characteristic is fundamental for scenarios where two or more independently bootstrapped Manets can be merged.

Regarding the design of the protocols used in certification services, in [66,71] only the sequence of messages in each procedure is specified. The protocol design is supplemented in this work with a proposal for precise message syntax and semantics.

Finally, we adopt a trust model implemented in the form of a distributed certification service [93]. As will be discussed in more detail later in this work, one of the grounds for choosing this model is the possibility of building a trust model that supports the revocation of trust for compromised and/or misbehaving nodes. It therefore becomes necessary to use asymmetrical cryptography primitives.

### 2.2. SECURITY OF ROUTING PROTOCOLS

As mentioned in the previous chapter, routing and autoconfiguration services are fundamental to the design of Manet technology. In this section, we review the main works on the security of routing and autoconfiguration protocols.
2.2.1. Routing protocols

Routing in ad hoc networks is rather different from the routing used in wired networks. Routing in ad hoc networks depends on many factors including topology, router selection and the specific characteristics which may be heuristic when finding the best or fastest path for delivering data. The main characteristics peculiar to Manets that can have a direct impact on routing protocol design are listed below:

- Scarce resources, such as bandwidth and power. The routing algorithms must make efficient use of the available bandwidth, allowing energy saving wherever possible.
- Symmetrical and asymmetrical links. A link is known as symmetrical when two nodes are within one another’s transmission area, with the same routing characteristics in both directions. When this is not the case, the link is called asymmetrical and routing becomes a difficult task. Most routing protocols on ad hoc networks are based on symmetrical links since asymmetrical links should be avoided.
- Mobility patterns. Some nodes are highly mobile while others can be fixed or move slowly. It is difficult to predict the movement pattern of nodes, besides, the number of nodes in a network can be very high and the task of finding a route to the destination will result in frequent exchanges of control information between nodes. In this way, the nodes’ high mobility can lead to an overload in maintaining the routes by the routing protocols, so that there is no bandwidth left for transmitting data packets.
- Scalability. A routing protocol must continue to function effectively from small networks, with a few dozen nodes, up to large-scale networks with hundreds of nodes and multi-hop topology.

Given the particular features of the ad hoc network environment, in 1997 the IETF set up the Manet working group with the task of researching and developing specifications for Manet routing and introducing them as internet standards, making peer-to-peer routing feasible in a purely mobile and wireless environment.

To assess the merit and performance of a routing protocol, the Manet working group lists a number of metrics that the protocols must follow [26]. These metrics have been divided into qualitative and quantitative types and must be evaluated independently of any routing protocol. Table 2-1 describes the qualitative metrics of a Manet routing protocol. Table 2-2
shows the quantitative points [26] to be observed for analysing the performance of a Manet routing protocol.

It is not a simple task to determine the efficiency of an ad hoc network routing protocol. A number of factors have to be considered, such as the network size (number of nodes), the number of neighbours each node has, the speed at which the topology changes, the frequency at which the nodes enter and leave the inactive state, among others. Taking account of these factors, the efficiency of an ad hoc network routing protocol can be measured with the following ratios [26]:

- Data bits transmitted / data bits delivered: this measure represents the efficiency of the data bits delivered within the network. Indirectly, this measure also provides the average number of hops made by the data packets.

<table>
<thead>
<tr>
<th>Table 2-1 – Qualitative metrics of a Manet routing protocol.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>Distributed operation</td>
</tr>
<tr>
<td>Loop free</td>
</tr>
<tr>
<td>Operation on demand</td>
</tr>
<tr>
<td>Proactive operation</td>
</tr>
<tr>
<td>Security</td>
</tr>
<tr>
<td>Operation in a period of inactivity</td>
</tr>
</tbody>
</table>
Table 2.2 – Quantitative metrics of a Manet routing protocol.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay and end-to-end data performance</td>
<td>Statistical data such as variance, mean and distribution are very important in assessing the efficiency of a routing protocol.</td>
</tr>
<tr>
<td>Route discovery time</td>
<td>A special way of measuring the end-to-end packet delay for on-demand routing algorithms is the time required to establish routes when requested.</td>
</tr>
<tr>
<td>Percentage of packets delivered out of order</td>
<td>External measurement to assess the routing performance of transport-layer protocols such as TCP, which deliver packets in the correct order.</td>
</tr>
<tr>
<td>Efficiency</td>
<td>While routing effectiveness is an external measure of performance assessment, efficiency is an internal measure of its performance. If data-packet and control traffic have to share the same medium, and the capacity of the media is limited, then excessive control packet traffic will have an impact on routing performance.</td>
</tr>
</tbody>
</table>

- Control bits transmitted / data bits delivered: this measure represents the efficiency of the protocol in use in terms of control packets versus data packets delivered. Note that this measure should include not only the bits of the routing control packets, but also the bits in the data packet header. In other words, everything that is not a data packet is a control packet and must be counted in the algorithm.

- Control packets and data packets transmitted / data packets delivered: rather than measuring the efficiency of the routing protocol in terms of bits, this measure attempts to capture the efficiency of access to the protocol channel.

The design of Manet routing protocols has been a fairly active research topic in recent years. A number of proposals for the construction of routing protocols have been presented. Figure 2.1 presents a classification of the main routing protocols proposed. This classification takes account of the following criteria:

- Regarding route discovery policy:
  - Reactive protocols: These determine the routes to be used only on demand, i.e. only when a route is requested does the protocol initiate the route discovery process. The main advantage of the reactive algorithm is that it helps to save bandwidth and power. The delay in determining a route may be significantly high, however.
  - Proactive protocols: These periodically and continuously broadcast routing information, keeping an up-to-date knowledge of all routes, so that, when a packet needs relaying, the route is already known and can be used immediately. They
have the advantage of minimum delay when the route is requested, since it is immediately selected from the routing table. On the other hand, in order to keep the consistency and topology up to date, the network is in continuous use for exchanging routing packets and information. These updates are initiated by timing mechanisms and even with a balanced network there is a constant exchange of information.

- Hybrid protocols: These have a hierarchical structure, in that part of the routing information is updated proactively, and the process is supplemented by (reactive) route discovery on demand.

Regarding the routing algorithm:

- Distance vector algorithm: Routes are constructed on the basis of distance information (e.g. number of hops) between the origin and the destination, held by each node/router.
- Link status algorithms: The routes consider all the links in the network topology to calculate the best routes between origin and destination.
- Origin routing algorithms: Routes are established for an origin-destination pair and are available at the origin of the packets transmitted.

**Figure 2-1 – Classification of Manet routing protocols**

No existing protocol has optimum characteristics for all scenarios [26,100]. Thus, the works coordinated by the Manet working group recently identified a set of Manet routing protocols that are due to form a core of protocols providing a comprehensive and flexible routing service in the various Manet application scenarios. These protocols are: Ad hoc on-Demand Distance Vector Routing (AODV) [88], Optimized Link State Routing Protocol (OLSR) [25], Topology Dissemination Based on Reverse-Path Forwarding (TBRPF) [83] e Dynamic Source Routing (DSR) [58]. A brief description of these protocols follows.
2.2.1.1. **Ad hoc on-Demand Distance Vector Routing (AODV)**

AODV is a reactive routing protocol that uses a distance vector type routing algorithm. As a general rule, AODV attempts to eliminate the need to broadcast routing messages, which limits its scalability. Another important feature of the AODV protocol is that it attempts to minimise latency when new routes are requested. AODV provides an intermediate solution between reactive and proactive routing. In the first latency is high, since it is necessary to wait for a response to a routing request. In the second, the volume of data exchanged may be very high for an ad hoc network with high mobility. In comparison with conventional routing algorithms, such as distance vector and link status, AODV substantially reduces the number of routing messages broadcast onto the network. This is due to its reactive approach. AODV operates similarly to conventional algorithms, which can help where an ad hoc network is interconnected with a fixed network. Although it works similarly to conventional algorithms, AODV can support multicast and unicast traffic. However, the protocol offers a single route to each destination, which cannot be a good characteristic.

2.2.1.2. **Optimized Link State Routing Protocol (OLSR)**

OLSR is a proactive routing protocol that uses a link-status type routing algorithm. The key feature of this protocol is the use of multipoint relays (MPRs), which are nodes selected to relay the broadcast messages in the routing protocol flooding process. Only nodes selected as MPRs broadcast information onto the network in this way. MPRs combined with the local elimination of duplication are used to minimise the number of control packets sent onto the network. OLSR is designed for use in large-scale networks, where traffic between a specific set of nodes is random and sporadic. As a proactive protocol, OLSR is also suited to scenarios where pairs of communicating nodes are constantly changing.

The nodes executing OLSR use HELLO messages, exchanged between one-hop neighbours, to detect and update their neighbour set. Each node periodically broadcasts these messages giving information about heard neighbour interfaces and their link status. The link status may either be “symmetric” (link has been verified to be symmetrical), “asymmetric” (communication has been verified in one direction only), “MPR” (neighbouring node has been selected as one of the broadcaster’s MPRs, in which case the link must also be symmetric) or “lost” (neighbour has moved away and is no longer being heard). HELLO messages are not relayed to other nodes.
Each node independently selects its own MPR set from among its “symmetric” neighbourhood. The MS must be computed in such a way that, through the neighbouring nodes in that set, it can reach all two-hop neighbours.

For provision of routes to nodes more than two hops away, each node maintains topological information about the network. This information is acquired by means of topology control (TC) messages. Nodes that have been selected as MPRs by other nodes periodically generate TC messages, which contain the list of all selector nodes (MS). TC messages are flooded to the whole network by the MPR nodes. A message sequence number (SN) field is used to avoid duplicated message processing. This field is generated as a sequence of integers, incremented monotonically each time a message is generated.

2.2.1.3. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)

TBRPF is a proactive protocol that provides routing step-by-step along the shortest paths to each destination. It uses a link-state type routing algorithm in which each node generates an origin tree based on the topological information, using a modified Dijkstra algorithm. To minimise processing over the network, each node reports only a part of its origin tree to neighbouring nodes. The protocol uses a combination of different periodic updates to keep all neighbours informed of its dispatch by the tree or origin. Each node also has the option of sending additional topology information (full tree) to offer robustness in highly mobile environments. TBRPF discovers neighbours through differentiated “Hello” messages, which only count changes of neighbours’ status. This modification results in many fewer Hello messages than those used by other link-state protocols such as OSPF. Finally, the protocol is composed of main modules. The first is for discovering neighbours and the second for routing, which discovers the network topology and computes the routes to each destination.

2.2.1.4. Dynamic Source Routing Protocol (DSR)

DSR is a reactive (on-demand) routing protocol, composed of two main mechanisms: “route discovery” and “route maintenance” which work together. It uses a routing algorithm at the origin that allows multiple routes to a given destination and enables the sender to select and monitor the routes used for sending its packets. It provides loop-free routing, supports unidirectional links and rapid convergence when the network topology is altered. It was
developed for medium-sized ad hoc mobile networks and designed to support high rates of mobility.

2.2.2. Security of routing protocols

The process of standardising routing protocols is still ongoing and many security aspects of these protocols have yet to be tackled. In practice, none of these protocols offers a satisfactory response to security aspects. However, most of the works presented on Manet security relate to the protection of routing protocols [28,41,48,49,84,116].

An analysis of the vulnerabilities of the possible AODV protections is presented in references [28,41,116]. Security aspects of the DSR protocol are analysed in [49,84]. In this last reference, a secure routing protocol (SPR) is proposed, directly applied as an extension of the DSR and ZRP protocols. In [48], the authors discuss the security analysis of proactive distance-vector routing protocols, focusing on the DSDV protocol.

Most of the security mechanisms proposed are based on some type of authentication extension specified for the routing protocol. Dahill et al. [28] propose the Authenticated Routing for Ad hoc Networks (ARAN) protocol, a modified version of the AODV protocol, using specially defined digital signatures to authenticate routing protocol messages. Authentication is achieved by including digital signature(s) in each message of the protocol. The certificates are specified to provide a link between a node’s IP address and its public key, used to validate the digital signatures included in the messages. The certificates are provided by a centralised server trusted by all the Manet nodes. Since the protocol messages contain information that is modified during relaying between origin and destination, the messages are signed only by the sending node. On the other hand, any node relaying such messages containing changeable information (e.g. route discovery and route reply messages) must also sign the message. Therefore, this solution leads to high consumption of computing resources, and a significant increase in the size of messages at each hop. In a similar approach, M. Zapata and N. Asokan propose the SAODV (Secure AODV) protocol [41]. SAODV is a security extension that is sent together with each AODV message. The original specification of the AODV messages is preserved. In contrast to the ARAN protocol, SAODV requires the message to be signed only by the sender. To provide protection for mutable fields

---

10 The use of digital signatures to bind a node’s IP address to its public key seems inadequate, since addressing in ad hoc networks is bound to follow the current trend towards dynamic address allocation and autoconfiguration [41].
(e.g. “hop count”), hash chains are used (summary) [50]. The proposal explores the likely nature of the mutable fields, which are incremented monotonically. It also includes a brief discussion of the security extension to secure other Manet routing protocols, especially DSR. Both the approaches described in [28,41] depend on the existence of certification services, which are assumed as a part of network and node bootstrapping [28] or merely discussed en passant [41].

An alternative to using asymmetric cryptography, as in the works described above, is to establish security associations between nodes, allowing the use of symmetrical cryptography primitives. These associations can be derived from node synchronisation, as in [89] or directly from mobility, enabling local security associations only [19]. The approaches in [48,49,84] also make use of message authentication extensions, but using symmetrical cryptography. In [84], P. Papadimitratos and Z. Haas present the SRP, designed to secure reactive routing protocols, focusing on DSR and IERP protocols. In the SRP, for each route discovery, both origin and destination must have established a security association between them (i.e. sharing a secret key). Apart from this, SRP does not provide any protection for route error messages, so this protocol is vulnerable to attacks using these messages. In the ARIADNE [49] and SEAD (Secure Efficient Ad hoc Distance vector) [48] protocols proposed by Y. Hu et al., the authentication keys are extracted from a broadcast authentication protocol, known as TESLA [89]. This protocol, however, requires a level of clock synchronisation between nodes in the ad hoc network, which is hardly a realistic prospect for Manet. ARIADNE is designed to provide security in reactive routing protocols, focusing in particular on DSR and IERP. SEAD proposes security mechanisms to be applied in distance-vector type routing protocols, with a detailed analysis of the DSDV protocol, combining symmetric cryptographic authentication with the use of hash chains11 for authenticating mutable fields that are incremented monotonically (in this case, “hop count” and “sequence number”). The security strategies for the design of the ARIADNE and SEAD protocols are generalised by Y. Hu et al. in [50].

A different approach is proposed by H. Yang et al. [116]. The protocol designed is AODV-S, a modified version of the AODV protocol (“next hop” is added to the RREQ messages and the RREP messages are flooded onto the network, rather than being sent in unicast to a requesting node). There is no authentication for AODV-S messages. Alternatively, a token is continually transmitted and renewed for each member node, and

11 Hash chains are used here in a similar way to SADOV [41].
these tokens are used for nodes misbehaving or compromised in the exchange of routing protocol messages. Owing to the broadcast nature of the wireless communication channel, all the nodes executing AODV-S promiscuously monitor the protocol messages, in an attempt to detect attacks against the routing protocol. Unfortunately, the design of this protocol is based on an incorrect hypothesis, namely that an adversary cannot adopt the identity of another node, if this is expressed by the access address to the medium (MAC).

A review of the works discussed above leads to the conclusion that a relationship of trust between nodes must be established before secure protocols are used. In fact, many works assume that this phase takes place as part of the network bootstrap or ignore it, assuming prior existence of relationships of trust between nodes [28,41,84].

Table 2-3 compares the security solutions presented above, emphasising the main techniques used.

The contribution of this work regarding securing of routing protocols is the design for a Manet authentication extension (MAE), incorporating and generalising various techniques used in other proposals. This solution offers the following advantages [93]:

- Independence of the routing protocol: The specification for the authentication extension is supplied in terms of syntax and semantics for use with any routing protocol, without altering the protocol’s original specification. We discuss the use of our MAE with the four routing protocols being standardised by the IETF (i.e. AODV, OLSR, TBRPF and DSR) and we define precisely the authentication objects needed to securely authenticate each of the messages used. The aim is to propose a flexible and extendable solution, to allow for extensions meeting the specific authentication needs of each protocol.

---

12 The ad hoc network interface works in promiscuous mode and each heard message is copied and analysed by the monitoring module.

13 The relationship of trust can be established by a model of digital certificate with distribution of asymmetric cryptographic keys, as in [28,41], or by using security associations, with shared secret keys and symmetric cryptography, as in [48,49,84].
Table 2-3 – Comparison between the solutions for securing routing protocols

<table>
<thead>
<tr>
<th>System</th>
<th>Protocol(s) analysed</th>
<th>Alterations to original routing protocol</th>
<th>Trust model / key generation</th>
<th>Authentication system</th>
<th>Other techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARAN [28]</td>
<td>AODV, DSR</td>
<td>YES</td>
<td>presumes certificate distribution</td>
<td>multiple digital signatures, using certification</td>
<td>-</td>
</tr>
<tr>
<td>SAODV [41]</td>
<td>AODV</td>
<td>NO</td>
<td>presumes certificate distribution</td>
<td>digital signature of transmitter, using certification</td>
<td>hash chains for authenticating mutable fields</td>
</tr>
<tr>
<td>SRP [84]</td>
<td>DSR, IERP</td>
<td>NO</td>
<td>assumes security association</td>
<td>authentication with symmetric cryptography</td>
<td>-</td>
</tr>
<tr>
<td>ARIADNE [49]</td>
<td>DSR</td>
<td>YES</td>
<td>cryptographic keys derived from TESLA</td>
<td>authentication with symmetric cryptography</td>
<td>-</td>
</tr>
<tr>
<td>SEAD [48]</td>
<td>DSDV</td>
<td>NO</td>
<td>cryptographic keys derived from TESLA</td>
<td>authentication with symmetric cryptography</td>
<td>hash chains for authenticating mutable fields</td>
</tr>
<tr>
<td>AODV-S [116]</td>
<td>AODV</td>
<td>YES</td>
<td>association tokens with node identification</td>
<td>-</td>
<td>proactive monitoring of messages and elimination of misbehaving nodes</td>
</tr>
</tbody>
</table>

- Adaptation to the trust model: The MAE is fully adapted to the security model adopted, with self-organising distribution of certificates. The authentication system provides for the use of digital signatures and the encapsulation of certificates from entities signing the message\(^{14}\). As an alternative, however, the MAE design allows the use of symmetric cryptography primitives, making them compatible for use with authentication systems such as TESLA [89] or authentication derived from mobility, [19] among others. The choice of authentication mechanism adopted is configurable, allowing the solution to be adapted to various scenarios of Manet use and different security policies.

- Security of mutable fields: In common with other similar works [50], the use of hash chains for securing mutable fields is supported.

- Finally, we discuss and evaluate the use of the MAE for providing the authentication service for routing link-state type proactive protocols, like TBRPF and OLSR. In the specific case of OLSR, we also present some optimisations deriving from direct interactions between the routing protocol and the certification service protocol (for instance the OLSR flooding mechanism can be used for...

---

\(^{14}\) In general, only the sender of a message has to sign it. However, if additional signatures are necessary, it is possible to load digital signatures and certificates of more than one entity.
flooding certification service messages, taking advantage of the optimisations introduced by using MPRs).

2.3. SECURITY OF AUTOCONFIGURATION PROTOCOLS

Although considerable effort has gone into designing and standardising Manet routing protocols, autoconfiguration protocol design is still in its infancy. As a result, proposals for improving security in Manet routing protocols are appearing rapidly, whereas the literature on secure protocols for Manet autoconfiguration is still rare. In this section we present some proposals for resolving the Manet autoconfiguration issue.

2.3.1. Autoconfiguration protocols

The conventional alternatives for autoconfiguration in TCP/IP networks involve dynamic distribution of addresses using the DHCP protocol [33] and, more recently, autoconfiguration by random address allocation [23], as proposed by the IETF’s zeroconf working group\(^\text{15}\). However, the use of DHCP requires a central server for distributing information such as the IP address, network mask, standard gateway and other additional network information. As we have seen, the use of central servers is a problem in ad hoc networks. Furthermore, the protocols of the zeroconf working group are still at the preliminary stage.

Thus, the mechanisms adopted in conventional networks are not suited for use in ad hoc environments. Accordingly, a set of new proposals for autoconfiguration solutions specially designed and suited for Manets is starting to appear.

The autoconfiguration protocol must allow automatic allocation of IP addresses\(^\text{16}\) (and of other network parameters, such as DNS server addresses). In order to develop an autoconfiguration protocol suited to the various Manet application scenarios, we identified the following qualitative characteristics [118], shown in Table 2-4.

H. Zhou et al. [118] define some metrics that can be used to analyse the performance of an autoconfiguration protocol in ad hoc networks, shown in Table 2-5 below.

Autoconfiguration protocols can be classified in two ways. One relates to the autoconfiguration process and the second relates to the mechanisms used to detect duplicate

\(^{15}\) http://www.ietf.org/html.charters/zeroconf-charter.html

\(^{16}\) In contrast to infrastructure-based networks, in Manets the subnet mask and standard gateway configurations are not generally necessary, since routing takes place node-to-node and a routing protocol must always be active.
addresses, taking account of how and when these duplicate addresses are detected. The two classifications are presented below:

Regarding the autoconfiguration process:

- **Stateless**: Allows the node to construct its own IP address, based on either the hardware identifier or a random number. This process does not depend on a second entity to carry out autoconfiguration. After constructing the IP address, a duplicate-address detection mechanism is needed to ensure that the generated address is unique.

- **Stateful**: This requires each network node to maintain a set of IP addresses. This implies the need for a second entity to take part in the process of associating a new IP address. Apart from this, maintaining a common structure distributed between all the network nodes consumes bandwidth mainly when there is frequent merging and splitting of ad hoc networks.

As for the duplicate address detection (DAD\textsuperscript{17}) process:

- **Allocation with conflict detection**: Conflict detection works on the trial-and-error principle. The node attempts to select an IP address and requests the approval of all the nodes in the ad hoc network. If any network node give a negative response, this means that this IP address is already in use.

\textsuperscript{17} Duplicated address detection.
### Table 2-4 – Qualitative metrics for a Manet autoconfiguration protocol

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed operation</td>
<td>An essential property of <em>ad hoc</em> networks, since centralisation of information is unfeasible in this context.</td>
</tr>
<tr>
<td>Unique IP addresses</td>
<td>Ensure that two or more nodes do not take the same IP address.</td>
</tr>
<tr>
<td>Nodes leaving the network</td>
<td>An IP address is associated with a node only for the time it stays in the network. When a node leaves the network, its IP address must remain available to be assigned to other nodes.</td>
</tr>
<tr>
<td>Loss of messages</td>
<td>If a node fails or messages are lost, the protocol must act fast enough to ensure that two or more nodes do not take the same IP address.</td>
</tr>
<tr>
<td>Multi-hop operation</td>
<td>The only reason a node will fail to be configured with an IP address is if there are no available IP addresses on the whole network. In this case, if any network node has a free IP address, that address must be assigned to the requesting node, even if it is two or more hops away.</td>
</tr>
<tr>
<td>Support for splitting and merging of networks</td>
<td>The protocol must be capable of adjusting the distribution of addresses when two or more separate <em>ad hoc</em> networks merge, and when a network splits into smaller ones.</td>
</tr>
<tr>
<td>Security</td>
<td>If the network and link layers do not guarantee security, the protocols will be vulnerable to many forms of attack; additional mechanisms are required to inhibit changes to the operation of the protocols.</td>
</tr>
<tr>
<td>Operation in a period of inactivity (sleeping mode)</td>
<td>During the period of inactivity, the network may regard the addresses allocated to the sleeping nodes as free. For this purpose, the nodes must be able to notify the network that they are about to enter sleeping mode.</td>
</tr>
</tbody>
</table>

### Table 2-5 – Quantitative metrics for a Manet autoconfiguration protocol

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to detect duplicate addresses</td>
<td>Two or more nodes may not have the same IP address. If conflict occur, these must be detected in the shortest possible time.</td>
</tr>
<tr>
<td>Complexity</td>
<td>Taking account of the limited memory and low computing power of mobile nodes, the solution must be as simple as possible.</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Broadcast-based protocols consume excessive bandwidth and should be avoided where possible. Communication between neighbouring nodes only (localisation) is preferable.</td>
</tr>
<tr>
<td>Uniformity</td>
<td>If the protocol distributes the addresses uniformly, the probability of conflicts is low, leading to less processing over the network.</td>
</tr>
<tr>
<td>Latency</td>
<td>Latency is the time between the start and end of the autoconfiguration process, when an IP address is assigned to a new node.</td>
</tr>
</tbody>
</table>
Best-effort allocation: The network nodes are responsible for assigning IP addresses to new nodes, attempting to assign a free address that is not in use by any network node. All the network nodes maintain a table of IP addresses in use or free on the network. Thus, when a new node arrives on the network, its nearest neighbour will select a free IP address to assign to it. The problem is that two or more nodes may arrive at the same time and the nodes can offer the same IP address. The advantage of this protocol is that it works very well with proactive routing protocols, since the nodes frequently broadcast details of the network addresses already in use.

Conflict-free allocation: This uses the concept of binary division, meaning that each node has different sets of IP addresses. Each node can assign an IP address without the need to consult other nodes for approval. In this way, all the network nodes are responsible for the process of assigning an IP address. This mechanism offers the advantage of not having to broadcast to assign an IP address.

Some authors classify the mechanisms for detecting duplicate addresses in another way. In [114] protocols are classified as active or passive. Active protocols are those that broadcast additional information onto the network, with a need for additional control packets to ensure that the protocol works properly. Whereas passive protocols are those that detect duplicate addresses, but without the need to broadcast additional control packets onto the network. It is all done just be monitoring the routing protocol traffic. However, with this latter method there is a period of time in which packets can be delivered to the wrong destination, known as the vulnerability period.

Manet autoconfiguration is a fairly new topic in the academic world and embryonic in the market for data communications networks. Searches on this topic reveal little completed work. No standard has been adopted and all the sources made available on the internet are still in draft form, or are articles published in conferences specialising in the field. We go on to present some proposals for autoconfiguration of addresses in ad hoc networks.

2.3.2. Allocation with duplicate-address detection

Perkins et al. propose [87] a protocol based on the principle of trial-and-error. A node randomly selects an IP address and makes a duplicate address detection request (Address Request message – AREQ), broadcast to all network nodes. The requester then waits for a set period of time for positive responses (Address Reply message – AREP), indicating that the
selected address is already allocated. For this request and response process, the requesting node uses a temporary address selected from a limited block of addresses, reserved solely for the autoconfiguration process. This allows the response message to be relayed directly to the requester by unicast. There is no duplicate address processing for this temporary address, since the address is allocated for a short period of time only. This protocol does not support splitting or merging of networks and uses a flooding process for obtaining the approval of all the network nodes when selecting addresses. The solution is not readily scaleable and can lead to long latencies during autoconfiguration, or even excessive network overload.

The protocol proposed by J. Jeong et al. [57] works with two duplicate address detection mechanisms. The first is called strong duplicate address detection (Strong DAD), featuring limited times for message exchanges. The second is called weak duplicate address detection (Weak DAD) and uses routing protocol messages to check that addresses are unique. DAD checking takes place during allocation of the address to joining nodes (Strong DAD) and during network merging (Weak DAD). This protocol is similar to the one used in [87]. Splitting and merging of networks are considered, however. It needs to be integrated with a proactive routing protocol.

Finally, in the proposal by K. Weniger [114], duplicate address detection is of the passive kind, performed merely for monitoring routing protocol traffic. Based on proactive routing protocols, three different types are used: One uses the sequence number employed for the routing protocols. The second is based on the localisation principle, since in proactive routing protocols, the nodes move at limited speed. The third type exploits the neighbourhood since a node knows all of its neighbourhood and also the neighbourhood of the origin of the routing packet. The protocol needs additional mechanisms for handling merging and splitting of ad hoc networks.

2.3.3. Best-effort allocation

The proposal by S. Nesargi and R. Prakash [81] presents an alternative to the use of temporary IP addresses [87]. Each configured node maintains a data structure containing the allocated IP addresses and those which are still pending since they are involved in autoconfiguration. A node requesting an IP address sends a (broadcast) message to its neighbours already on the network. One of these neighbours responds to the request and acts as “trustee” of the requesting node. The trustee selects an address that is not in the allocated list and broadcasts an authorisation request flooding all the network nodes (Initiator Request message – IREQ). All the nodes must give a positive response, authorising the allocation of
the new address. If any response is negative, the trustee node must then select another address and repeat the DAD process, obtaining approval from all the other network nodes to conclude the assignment of the selected address. The protocol caters for merging and splitting of ad hoc networks.

Similarly, Boleng [10] proposes the use of trustees (here known as attachment agents) for selecting the address to be assigned to a new node and for performing the DAD. Trustees select addresses using sequential numbering, requiring only the last allocated address to be stored. However, to allow addressed allocated to nodes that have left the network to be re-used, the configured nodes also maintain a cache of available addresses, built from exit notifications broadcast by departing nodes. The salient feature of this proposal is the adoption of a variable-size address space (e.g. variable-length addresses) with the aim of minimising data-transmission overheads. The address space grows and shrinks as required (as nodes join and leave). However, this approach rules out the possibility of using other software that explicitly use the IP addressing format. It also caters for merging and splitting of networks.

2.3.4. Conflict-free allocation

Conflict-free allocation follows a completely different approach to the solutions discussed previously. In this autoconfiguration protocol, each node has a set of IP addresses that are used to configure new nodes joining the network, without having to consult any other node already configured on the network. These sets of addresses are different within the same ad hoc network.

The DCDP (Dynamic Configuration Distribution Protocol) proposed by A. Misra et al. [78] is an example of this type of protocol and uses a buddy system model [102] to supply different sets of IP addresses to the network nodes. This model adopts a mechanism for binary division of the address block. A node wishing to join a network (a client node) makes a (broadcast) request for an IP address (Address Request message). Configured neighbouring nodes respond to the request (Address Reply message), notifying the size of their block of available addresses (free_ip_block). The client node then selects the neighbour with the largest free_ip_block (Server Pool message). The selected neighbour (server node) divides its set of IP addresses into two halves and sends one half to the client node (IP_Assigned message), keeping the other half for itself to handle future requests. When the client receives the set of addresses, it assigns the first one to itself and keeps the rest as a set of available addresses. The client node then sends a message confirming the success of the operation (IP_Assignment_OK message). The process is illustrated in Figure 2-2.
In recent work, F. Buiati [14] presents a complete specification for the DCDP protocol, based on the work done in [78] with the improvements proposed in [80]. This proposal works with merging and splitting of ad hoc networks to maintain distinct sets of IP addresses in the configured nodes of the network.

2.3.5. Security of autoconfiguration protocols

Securing the autoconfiguration process in ad hoc networks is not a common subject, and was not even discussed in the technical literature until recently. In [13] a first approach to the subject is presented, and it will also be discussed further on in this work. The approach consists of adopting the trust model as soon as a new node joins the network. To do so, a node first has to gain the trust of the network (i.e. by getting a certificate), using the distributed certification services. Only once this stage is complete does the node perform the autoconfiguration process. The autoconfiguration service messages must all be authenticated with the Manet authentication extension (MAE) [93]. All communication, including autoconfiguration of addresses, takes place between neighbours one hop apart and does not need a previously configured IP addresses\(^\text{18}\). The proposal for secure autoconfiguration in this

\(^{18}\) Alternatively, the newcomer node may be allowed to select an address from a block reserved for the autoconfiguration process, as in [87].
work is based on the DCDP protocol, and details the vulnerabilities of that protocol and the protection model applied to it. Nevertheless, the solution is a generic one that can readily be used with other autoconfiguration protocols, since new nodes are certified before the autoconfiguration process is executed. To do so, it is sufficient for the autoconfiguration protocol messages to append an MAE containing the appropriate information for authenticating them (in the case of DCDP, a digital signature suffices, since there are no mutable fields in the messages).

Simultaneous work, similar in certain respects to this one, is presented by A. Cavalli and J. Orset [20]. However, in that proposal the authors restrict themselves to defining an authentication extension for the autoconfiguration protocol messages, assuming prior distribution of certificates, without specifying how that is done.

2.4. MANET INTRUSION DETECTION

Most of the current research on Manet security is devoted to the provision of preventive protection of the basic protocols (e.g. routing), using a mechanism similar to ours [28,41,84]. As a general rule, these solutions are not tolerant, in isolation, to the presence of compromised nodes in the network. These security mechanisms can be reinforced by proactive security services, such as intrusion detection systems.

Intrusion detection systems are designed to detect attacks against computer systems and networks, or against information systems in general. It is quite difficult, if not impossible, to set up demonstrably secure information systems and keep them secure throughout their working life. Therefore intrusion detection systems are designed to monitor the use of these systems to detect the appearance of insecure states.

Generally speaking, current interest in intrusion detection can be divided into three basic processes: data collection, design of the detection algorithm (analysis) and alert management. The IETF’s intrusion detection working group (IDWG) 19 defines the components that perform these tasks [115], as shown in Figure 2-3. The sensor collects raw data about the operation of the system being monitored (e.g. audit traces, network packets). These data are pre-processed to give events that are sent to the analyser, where the events generated are assessed in terms of an intrusion detection mechanism. If these events are sensitive, the analyser manages alerts, which are fed back to the manager. Finally, this

19 The Group’s official site is at http://www.ietf.org/html.charters/idwg-charter.html
component, in addition to correlating and classifying the alerts in order to refine previous analyses, provides the information needed to respond to any attacks detected.

Figure 2-3 – IDWG intrusion detection framework

H. Debar et al. [29] propose a taxonomy for current intrusion detection systems, shown in Figure 2-4. These classification criteria are discussed below:

**Detection method:** It describes the characteristics of the analyser. When the IDS uses information on the normal behaviour of the monitored system, trying to detect variations from this normal state, the IDS is said to have a behaviour base. If the IDS uses information on attacks that may be detected (attack signatures), it is known as a misuse IDS.

**Behaviour on detection:** This describes the IDS’s response to attacks detected. When the system reacts actively to an attack by taking corrective action (closing breaches) or proactively (recording possible attackers, closing down services), the system is classed as active. If the system merely generates and sends alerts (including pager, etc.), it is said to be passive.

**Data source:** This discriminates the IDSs on the basis of the type of input data they analyse. This information may be audit trails (e.g. system logs) in a computer, network packets, an application log or even alerts generated by other intrusion detection systems.

**Detection paradigm:** This describes the detection mechanism used by the IDS. These systems can assess states (secure/insecure) or transitions (from secure to insecure).

**Frequency of use:** Some IDSs are used to monitor the target system continuously in real time, while others are executed periodically.

Note that IDSs have specific requirements in the Manet context which are not compatible with conventional approaches to intrusion detection. Namely, like the other security services, an IDS must be distributed, self-organising and – if possible – operate in a
localised way. Since IDS design for Manets is a fairly recent affair, we present below a summary of the main initiatives for conceiving and designing intrusion detection systems that satisfy these requirements, even if the systems were not specifically designed for our target environment. This is important to know as it helps to identify key concepts in the conception of an IDS designed specifically for Manet environments.

As a first analysis, it can be said that much has been done a studied recently on distributed intrusion detection systems. Table 2-6 sets out the main proposals for distributed intrusion detection systems that are already sufficiently mature in their development to enable the principles and directions considered in each design to be validated.

Figure 2-4 – Taxonomy of intrusion detection systems
Table 2-6 – Main proposals for distributed IDS

<table>
<thead>
<tr>
<th>IDS</th>
<th>Data source</th>
<th>Detection method</th>
<th>Distributed pre-processing</th>
<th>Centralised detection</th>
<th>Real-time analysis</th>
<th>Response type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAFID [103]</td>
<td>System</td>
<td>Misuse</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Passive</td>
</tr>
<tr>
<td>DIDS [106]</td>
<td>System/Netw</td>
<td>Hybrid</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Passive</td>
</tr>
<tr>
<td>Grids [104]</td>
<td>System/Netw</td>
<td>Hybrid</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Passive</td>
</tr>
<tr>
<td>CSM [112]</td>
<td>System</td>
<td>Anomaly</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Active</td>
</tr>
<tr>
<td>JiNao [37]</td>
<td>MIB/Network</td>
<td>Hybrid</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Passive</td>
</tr>
<tr>
<td>EMERALD [90]</td>
<td>System/Netw</td>
<td>Hybrid</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Active</td>
</tr>
<tr>
<td>IDA [5]</td>
<td>System</td>
<td>Misuse</td>
<td>Mobile agents</td>
<td>Yes</td>
<td>Yes</td>
<td>Passive</td>
</tr>
<tr>
<td>SPARTA [67]</td>
<td>System/Netw</td>
<td>Misuse</td>
<td>Mobile agents</td>
<td>No</td>
<td>Yes</td>
<td>Passive</td>
</tr>
</tbody>
</table>

All the proposals presented in Table 2-6, except CSM, EMERALD and SPARTA, are organised hierarchically around a central node. This central node is the core of IDS and it uses information collected in distributed form to detect intrusions. In these architectures, only the data collection process is distributed and they are therefore not suited to the Manet context.

CMS architecture is fully distributed. A local IDS is installed in each cooperating node for collaborative identification of the originator of network connections. The EMERALD architecture was actually specially designed to accommodate the scalability requirements of large networks. This IDS is made of communicating generic nodes, called monitors, which are installed in each system. Of the architectures in Table 2-6, only SPARTA was specifically designed for wireless network environments. However, that system is designed to detect attacks against distributed applications and it does not take account of attacks against the network layer.

As regards the self-organisation criterion, the use of platforms of agents offers an alternative to the client-server distribution model. In particular, one can consider the use of mobile agents, which are agents that move from one node to another loading data and executable code. With careful design, this type of agent enables the amount of data exchanged on the network to be reduced considerably, which makes the IDS architecture especially interesting in Manet environments, where bandwidth is limited and links are not very reliable. The use of mobile agents, in contrast to conventional approaches using distributed applications where the data are transported to the computer system that processes them,
enables the code even to move data. Moreover, a node that dispatches a mobile agent does not need to wait its turn before continuing its normal processing, since such agents can be dispatched again or destroyed in other nodes, without having to move back to the node that created them.

The use of mobile agents in IDS design has become quite a hot topic in recent years. Table 2-7 summarises the main initiatives in the field.

W. Jansen [55] of the Mobile Agent Security project at the NIST (National Institute of Standards and Technology, USA) provides an enthusiastic analysis of the possible benefits and drawbacks of the use of mobile agents in intrusion detection, whereas P. Mell et al. [77] define an IDS architecture using mobile agents, resistant to denial-of-service attacks. However, this architecture is closely tied to the network infrastructure, which is not found in Manets. A similar approach can be found in [21].

<table>
<thead>
<tr>
<th>IDS</th>
<th>Data source</th>
<th>Detection method</th>
<th>Data collection and pre-processing</th>
<th>Analysis (detection)</th>
<th>Correlation of alerts</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAFID [103]</td>
<td>System (host)</td>
<td>Misuse</td>
<td>Mobile agents</td>
<td>Centralised</td>
<td>No</td>
</tr>
<tr>
<td>IDA [5]</td>
<td>System (host)</td>
<td>Misuse</td>
<td>Mobile agents</td>
<td>Centralised</td>
<td>No</td>
</tr>
<tr>
<td>MAIDS [43]</td>
<td>System/Network</td>
<td>Hybrid</td>
<td>Mobile agents</td>
<td>Centralised</td>
<td>Yes</td>
</tr>
<tr>
<td>SPARTA [67]</td>
<td>System/Network</td>
<td>Misuse</td>
<td>Mobile agents</td>
<td>Mobile agents</td>
<td>No</td>
</tr>
</tbody>
</table>

E. Spafford et al. [103] (Purdue University, USA) designed the AAFID system (Autonomous Agents For Intrusion Detection) and M. Asaka et al. [5] (Information-Technology Promotion Agency – IPA, Japan) proposed the IDA system (Intrusion Detection Agent). Both systems have a hierarchical architecture organised around a central node. This node is the core of IDS and uses information collected in a distributed way. Similarly, G. Helmer et al. [43] (Iowa State University, USA) propose MAIDS (Mobile Agent Intrusion Detection System), involving an IDS based on intelligent agents. Mobile agents are used to allow various types of specialised agents, known as low-level agents, to travel between data-collection points and implement simple detection of suspect activities. In each instance of MAIDS, static collaborative agents (high-level agents), specialising in a specific category of intrusion, collaborate with one another locally to make inferences about intrusion detection. The concept of cooperation between MAIDS instances is defined in terms of a centralised data warehouse system. Other approaches having central entities that analyse the data and
make inferences on intrusion can be found in [8,9]. In all these architectures, only the data
collection process is distributed and they are therefore not suited to the Manet context.

Once again, we return to the SPARTA system (Security Policy Adaptation Reinforced
Through Agents), proposed by Krügel et al. [67] (Technical University – Vienna, Austria). In
this proposal, developed concurrently with the proposal presented in this work, mobile agents
are used to trigger events and queries, both being formally specified in a typified language.
Each node in an autonomous local IDS is formed of local sensors and a platform of mobile
agents. Unfortunately, the SPARTA architecture has a single central node, known as the
management console. Although this central entity is not directly involved in the intrusion
detection process, it plays an important role in the system architecture, since each node
joining the collaborative detection service has to register with the management console. This
latter characteristic makes SPARTA incompatible with the Manet context.

Finally, a completely different approach for the use of mobile agents in intrusion
detection is proposed in independent works by S. Hofmeyr et al. [45] (University of New
Mexico, USA) and S. Fenet et al.[ 35] (Université Claude Bernard Lyon, France). In [45], the
IDS architecture emulates the biological immune system by using specialised mobile agents
(each intrusion pattern is mapped by a different agent) roaming through the monitored system
looking for traces of intrusions. In [35] the mobile agents imitate the behaviour of social
insects. Unfortunately, the mobility pattern is poorly described in both systems, so it is
difficult to assess the usability of these proposals in Manet contexts.

IDS design for Manet is not a completely new issue and it has been tackled recently
[3,51,73,79,111,117]. Y Zhang and W. Lee [117] introduce the basic requirements for this
special type of IDS. In a previous work, we introduced the preliminary architecture concepts
[3,85,94].

In [42], S. Gwalani et al. propose an IDS for Manet that is essentially designed to
reinforce the security of the routing protocol. However, this IDS has a centralised
architecture.

V. Mittal and G. Vigna [79] present an IDS made up of various sensors for detecting
attacks against the routing protocol, which promiscuously monitor the network links. This
IDS has collaboration concepts yet the detection mechanism assumes that information on the
overall topology is available. In Manet, it would be most appropriate to use localised topology
information, since the topology is dynamic and the global topology information may not be
completely up to date.
Y. Huang et al. [51] and C.-Y. Tseng et al. [111] present IDS designs for Manet based on a strategy of anomaly detection. The drawback of these works lies in the absence of cooperation between nodes, as each node acts in isolation to detect attacks.

A strategy for intrusion detection and response to combat uncooperative nodes in ad hoc networks is presented by S. Marti et al. in [73]. However, this approach does not include any notion of collaborative security services. The work in [116] presents a security solution based on a modified version of AODV, using an intrusion detection mechanism combined with a system of tokens used to ensure that the nodes have access to the routing services. However, this solution does not incorporate any preventive protection (authentication). On the contrary, only a simple neighbourhood verification system is used. Unfortunately, as mentioned above, this mechanism is based on the incorrect hypothesis that MAC addresses cannot be spoofed. Moreover, the intrusion detection mechanism is restricted to flooding RREP messages only, not generalising to combat all the attacks described in terms of generating, modifying and spoofing other routing protocol messages.

This work present a fully distributed intrusion detection system [95], in the sense that the data collection, analysis (detection) and alert-management processes work in distributed mode. No central entity is required. In this design, a local IDS (LIDS) is placed in each Manet node. The LIDS intercommunicate using a mechanism that takes account of the restrictions on bandwidth and connectivity. A platform of mobile agents provides self-organisation. Moreover, the collaborative detection process basically takes place in the neighbourhood or in a restricted number of nodes involved in an attack in multiple phases, so restricting the communication and processing overhead to the local level or between selected network nodes. Finally, contrary to the majority of works discussed above, the IDS presented in this work allows an effective strategy of active responses to detected attacks, as it is fully integrated with the other security services.
3. MANET SECURITY MODEL

This chapter presents the security model proposed for Manets. To define the properties of the model, we first discuss vulnerability and adversary models. This is important because it defines the scope of the protection that the model offers; no security solution offers full coverage and unlimited scope. Having characterised the vulnerabilities and adversaries, we can establish the network security requirements that are taken into account when designing the model.

The security model itself comprises a combination of preventive security services (certification/authentication) and corrective services (intrusion detection and response), which are also designed in the context of this study, with a specific design relating to Manet environments. The concept of the model also includes the integration of existing security services offered in the main operating environments (e.g. packet filtering, cryptographic tunnelling between applications – SSL/TSL – and across networks or nodes – IPSec). Finally, we discuss the extension of the model to include other security services yet to be designed (e.g. distributed firewall, distributed security policy management).

3.1. VULNERABILITY AND ADVERSARY MODELS

There is a relatively broad spectrum of potential security breaches and weaknesses in computer systems, especially when networked together, resulting in long lists of vulnerabilities. A number of authors have discussed the question of the vulnerabilities of computer systems, such as Landwehr et al. [68], presenting a summary of the potential vulnerabilities of computer systems and proposing a taxonomy of these weaknesses. Although it does not claim to be a complete summary of the security vulnerabilities in computer systems, it is worth mentioning the initiative of the Computer Emergency Response Team (CERT)\(^2\) which maintains an active repository of the main vulnerabilities observed and reported in information security incidents.

Like the work of Landwehr et al. [68] and the CERT initiative, most summaries of vulnerabilities analyse systems in conventional network environments. Clearly, many vulnerabilities that exist in these environments are also possible in ad hoc networks. However, owing to the particular characteristics of the ad hoc network environment, it has

\(^2\) http://www.cert.org
vulnerabilities peculiar to it that are not found in other types of networks. Apart from this, many vulnerabilities that exist in conventional networks can be exploited in particular ways in ad hoc network environments, thereby becoming new vulnerabilities exclusive to such networks. In this work, the discussion will focus on those vulnerabilities that arise out of the ad hoc environment. In particular, we analyse the various security weaknesses related to the basic services of those networks, i.e. the routing and autoconfiguration protocols.

3.1.1. Vulnerabilities model

To characterise the vulnerabilities specific to ad hoc networks, it is important to identify which characteristics of these networks require the main security aspects to be treated differently from conventional networks. Those aspects are:

- **Ad hoc** networks basically use wireless communications. The absence of a physical link between the network node and an infrastructure enables a number of actions to take place:
  - Listening: the wireless communication channel can be listened to by a computer; to do this, the listening device only has to be within range of the wireless transmitter(s).
  - Direct communication: in an ad hoc network, it is sufficient for two nodes to be close enough (wireless transmission range) for them to communicate with one another.
  - Mobility: nodes can move freely (at limited speed) in an ad hoc network, leaving the direct communication range of some nodes and entering the range of others.

- The nodes cooperate with one another to establish and maintain network connectivity (i.e. routing), as they do for the other essential services, since the network has no centralised entity. In this way, in contrast to conventional networks, where the essential services are located at specific points in the network (such as gateways and autoconfiguration servers) that have appropriate protection and controlled access, in Manets these services are distributed between the various nodes which have to collaborate to ensure that the network operates properly. Thus, non-cooperation on the part of one or more nodes, whether owing to malfunction or selfishness (e.g. a node refuses to collaborate to save its battery power), can compromise the functionality of the network as a whole.
Nodes use portable power supplies (batteries). This type of power source discharges with use. Therefore, a range of services and protocols are designed to operate efficiently as regards power consumption. It is possible, however, to cause incorrect behaviour in one or more nodes of a Manet, whereupon they start consuming their battery power more rapidly, so reducing their useful operating time on the network.

Finally, another aspect fundamental to the concept of the vulnerabilities of ad hoc networks is the possible existence of compromised nodes. A compromised node is one which has gained the trust of the network and then begun to misbehave at some later stage. This improper behaviour could be cause by an individual fault, intentional or otherwise, or even compromising action by agents external to the network. Thus, the vulnerabilities that cannot normally be exploited by untrusted nodes, can nevertheless constitute potential breaches to be exploited by compromised nodes that are trusted by the network.

Our security model takes account of the following vulnerabilities, which are explored in more detail in the case of routing and autoconfiguration services.

- **Modification:** messages are modified incorrectly while being relayed through intermediate nodes between the sender and the recipient.
- **Spoofing:** in this case, any node uses the identity of another to generate network messages.
- **Fabrication:** generation of false messages.
- **Non cooperation:** a node agrees to collaborate with others on the network, but does not carry out its task when its collaboration is required.\(^{21}\)

### 3.1.2. **Adversary model**

An adversary is any agent that performs actions with the aim of corrupting the Manet’s services (attacks). The adversary may be either internal (a trusted node that acts improperly) or external (a node compromised by an external agent). In particular, we acknowledge that a legitimate (trusted) node may be compromised by external agents through exploitation of the vulnerabilities of the operating system or of services normally executed in a node, or even by the (physical) capture of a node that is not adequately protected.

\(^{21}\) It is essential to distinguish between non-cooperation and non-participation. In the latter case, a node is physically present, but it does not specifically agree to take part in the Manet’s collaborative services. In other words, the node does not execute the routing or autoconfiguration protocol, for example.
We acknowledge that if a node is compromised by an external agent, all the secret information held in the node (such as secret keys, private keys, passwords, etc.) may be disclosed. The adversary may then launch attacks from the compromised node or simply spoof it.

Considering the specific characteristics of the ad hoc environment and the vulnerabilities defined in the scope of this work, we can identify the main actions that an adversary can carry out to compromise network functionality:

- Listen promiscuously to the wireless communication channel and obtain information from the traffic generated by and sent to its neighbours.
- Communicate directly with any node that is within its transmission range.
- Move, at limited speed, to collect information on more distant nodes, communicate directly with other nodes that are not in its neighbourhood or even escape monitoring by nearby nodes.
- Fail to collaborate with nearby nodes, even after agreeing to do so (e.g. execution of the routing or autoconfiguration protocol).

Finally, multiple adversaries may coexist in the network and may coordinate their action to compromise the security mechanisms in operation. From the point of view of the security services we make no distinction between internal and external adversaries, since even if trusted nodes can be distinguished from untrusted ones, the Manet environment is not guaranteed to be free of nodes compromised by external agents.

3.1.3. Security requirements

An analysis of the vulnerability and adversary models set out in the previous section allows us to define a set of requirements to be satisfied by the planned Manet security model. We discuss those requirements in this section:

- Trusted versus untrusted nodes: There must be an effective way of immediately verifying whether or not a message comes from a trusted node. Messages from untrusted nodes must be processed in accordance with a security policy, enabling them to be quickly and quietly discarded.
- Protection against modification: A message may not be modified incorrectly on the network by untrusted nodes. It must also be possible to detect incorrect modification of messages by trusted nodes.
• Protection against spoofing: A trusted node must have a unique identity. This identity may not be usurped by untrusted nodes or even by other trusted ones.

• Protection against fabrication: Fabricated messages may not be injected onto the network by untrusted nodes. Similarly, it must be possible to detect fabrication of false messages by trusted nodes.

• Protection against non-cooperation: Since non-cooperation cannot be effectively prevented, it must be possible to detect it.

• Elimination of compromised nodes: Trusted nodes that become compromised must be eliminated from the collaborative network services, by revoking the trust granted to them.

3.2. SECURITY MODEL

This section presents the security model proposed for ad hoc networks. The model considers the generic requirements for Manets to define security services that operate in accordance with a model of self-organising distributed services. We then present a distributed trust model that can satisfy the requirement to distinguish between trusted and untrusted nodes in a Manet environment. The model is based on a distributed certification service, that also provides effective unique identification of nodes. Finally, we propose to integrate a set of security services, focusing on an authentication service and an intrusion-detection service that interact to provide the other security requirements noted.

3.2.1. Self-organising distributed services model

A general architecture for distributed, self-organising and collaborative services in Manets is shown in Figure 3-1. Each Manet node hosts an autonomous and active instance of the service. These instances are generically known as L-Service (Local Service). An L-Service collaborates with L-Services in nearby nodes by means of a collaboration protocol. Collaboration may begin at any time, and may be established between any other L-Services available at the time collaboration begins. This is an essential characteristic in that the availability of individual nodes cannot be guaranteed, since a node may simply move out of the network’s communication range. This concept of self-organisation is exactly the same as the one used in the routing service concept, the L-Service being represented by the daemon.

22 The use of IP or MAC addresses for this purpose is inappropriate, since it has been shown that this information can be spoofed.
which is executed automatically in each Manet node, and the cooperation protocol being represented by the routing protocol. Thus, each of the security services defined in our model, together with other Manet services, including the routing and autoconfiguration services discussed in this work, fit into this general architecture. Note that no centralising entity is required in the design of the services shown in Figure 3-1.

In the collaboration illustrated in Figure 3-1, the Manet nodes exchange information using the cooperation protocols defined for each service. Moreover, this collaboration must extend to the packet relaying service, since the nodes depend on one another to maintain network connectivity.

Finally, the cooperation protocol must be carefully designed so as to maintain localised cooperation and interaction restricted to the nodes’ neighbourhood (one hop) or to a limited number of selected nodes. This limits the communication overhead between L- Services. This requirement relates to the limited bandwidth and power supplies available in these network environments.

### 3.2.2. Distributed trust model

A fundamental issue regarding the securing of Manet collaborative mechanisms is the proper specification of the nodes’ membership of the network, allowing trusted nodes to be distinguished from untrusted ones. This definition of membership (i.e. related to the concept of mutual trust between nodes) may be imposed on nodes as a first line of defence for collaborative services, requiring nodes to be members of the network before they may cooperate with other members. In this scenario, information is exchanged using the cooperation protocols defined for each service and for packet routing in the environment of a set of mutually trusting nodes.
There are three important aspects relating to the definition and imposition of Manet membership referred to above. First, the actual process of establishing trust (e.g. for network membership) has to be considered. An alternative is to define a relationship of trust between each of the nodes that are interacting (peer-to-peer model). While this solution can be effective in some specific cases, it is not scalable to situations where one Manet is formed of several dozen or several hundred nodes. Thus, in general, this process must also follow a collaborative approach. It must be possible to revoke Manet membership if a compromised node is detected on the network. Secondly, membership is a requirement of the collaboration model. Once membership is established, a node must be able to prove to the other members that it is a network member, just as it must be able to check and verify other nodes’ claims of membership. In order for this to be possible a basic service for authentication at origin must be available in all the collaboration protocols. Finally, the third aspect to consider relates to the actual concept of trust, which varies for each particular Manet. A variety of scenarios may be considered, varying from completely open situations (all nodes are trusted) to extremely restrictive ones (none are trusted).

In our work, we consider that this trust model operates via a distributed certification service. Network membership is then explicit in the form of digital certificates issued to the Manet nodes that are regarded as trusted. As will be discussed in more detail in the next chapter, trust is established collaboratively, by the formation of a distributed certification authority where the secret certification code (the certification authority’s private key) is shared between the participating nodes.

The trust model implemented by this distributed certification service has the following advantages:

- Relationships of trust in a Manet are collaboratively established and maintained, in scalable form. The relationship of trust can be cancelled in the event of compromised nodes simply by revoking the node’s certificate.
- The use of digital certificates allows each node to digitally sign the messages it generates, making it possible to provide an authentication-at-origin service with non-repudiation. The actual digital certificates are used to uniquely identify the nodes and as protection against spoofing by other untrusted nodes.

\(23\) When considering the security requirements associated with eliminating compromised nodes, this authentication must not be subject to repudiation, as the detection of misbehaviour by neighbouring nodes is regarded as grounds for issuing an accusation against those nodes.
• The use of the model in various Manet application scenarios is easily adjusted in accordance with a security policy, by defining operating parameters that reflect the various security requirements of each application.

3.2.2.1. K-of-N trust model

The certification authority’s private key is shared between participant nodes by means of threshold cryptography [102]. With this technique, the secret certification code may be split into as many parts as necessary (e.g. N parts). Thus, if the security policy permits, potentially all the network nodes can hold a part of the private certification key. However, only a determined number of nodes (K) is required to provide the certification services.

This number K (the threshold) is an important system constant, representing a compromise between the security level required and the scalability of the system. The higher the value of K, the more nodes must take part in collaborative certification (i.e. confirming their trust) in order to make the certification services available. A system with a large K is therefore fairly robust in the presence of compromised nodes, since at least K compromised and collaborating nodes are required to breach the security of the certification system, whereas K/2 compromised nodes can collaborate to cause Byzantine faults in the service [62]. In the extreme case, where K = N, all the network nodes holding parts of the private key must give their approval in order for certification services to be complete. On the other hand, large values of K can compromise the availability and scalability of the system. Thus, the lower the value of K, the less communication overhead is required to provide the services. In the extreme case where K = 1, the system is potentially insecure, since a single node can break the system on its own (i.e. all holders know the private certification key). An interesting approach is suggested in [66], consisting of choosing a value of K close to the size of the neighbourhood (one hop), allowing certification services to be localised and communications to take place between neighbours of the nodes needing certification services.

This “K-to-N” type trust model, where K is the number of nodes that must trust another node in order to admit it to the network and N is the (non-fixed) number of nodes that hold a part of the private certification key. Each node holding part of the DCA private key may therefore issue partial certificates to the nodes it trusts. These nodes, in turn, collect any K partial certificates issued to them and can combine them to obtain the complete certificate. The distribution of parts of the private certification key and the revocation of certificates are also performed collaboratively.
Certification services are provided via local certification services (L-CERT) collaborating together. These L-CERTS dynamically form coalitions of K nodes. The protocol for the basic distributed certification services (issue and revocation of certificates and distribution of parts of the private certification key) is illustrated in Figure 3-2 below. Steps 1 to 3 consist of establishing a dynamic coalition of K nodes, beginning with a service request (step 1), followed by the collection of responses from nodes that agree to service the request (step 2) and the notification by the requester to all the participants that the coalition has been formed (step 3). Finally, the service is complete when K responses (partial results) have been collected from all members of the coalition, duly signed with the parts of the private key of each of these nodes (step 4).

Figure 3-2 – Collaboration protocol for distributed certification services
Finally, we discuss the revocation of certificates, used to eliminate compromised nodes from the network. Trusted nodes that detect malicious activity by other nodes may generate accusations against them. In this way, K nodes generating accusations against one node can collaborate to revoke its certificate, thereby eliminating it from the network. The detection and accusation mechanisms also have to be executed collaboratively, so preventing a single compromised node from generating accusations against correct nodes and making network services unavailable to them. Certificates are revoked by issuing counter-certificates, signed by K nodes which are collaboratively accusing a node of being compromised. Counter-certificates are stored locally, in a local list of revoked certificates (L-CRL).

3.2.2.2. Identification of Manet nodes

Digital certificates contain a link between an identity and a public key, which is used in asymmetric cryptography schemes. The identification expressed in a certificate can be a name of a network entity (host or service), a user identifier (e.g. e-mail address), etc. The public key, on the other hand, is associated with a private key that is known only to the owner of the certificate. This private key can be used to authenticate messages exchanged over the network, using digital signatures.

In this work, the digital signatures themselves are used to identify trusted network nodes. In this way, the certificates are the only identification considered from the viewpoint of the security services. This choice is based on the fact that there is no identification – for nodes or users – that cannot be spoofed. As regards IP addresses, the usual form of identification in network services, apart from being easy to spoof, a node tends to have a dynamically configurable IP address, so that the address may change frequently. The association of node identifiers with MAC addresses is more permanent, in that these addresses are configured at the factory. Even so, a node can change its MAC address if it changes its network interface hardware (for instance, a notebook can change its PCMCIA wireless networking card). Moreover, MAC address can generally be spoofed since these addresses can be changed manually, and it is also possible to generate tables formed in a particular way (e.g. rawsocket API) where MAC addresses are defined in software. In this case, only some manufacturers reinforce the use of the hardware MAC address, so preventing these tables being sent onto the network with an adulterated MAC address. Finally, typical identifiers for

24 MAC addresses can be administered locally. In this case, the unique address assigned to the interface in the factory is not used.
users and nodes are names attributed by the users themselves or system administrators and offer no guarantee of uniqueness or protection against spoofing.

From the security point of view, the use of digital certificates for identification offers cryptographic protection to the identifier, and therefore allows the identity to be verified during the process of authenticating messages by origin. With the use of digital signatures, this authentication cannot even be repudiated. We make no distinction between names of network entities (hosts) or users (e-mail addresses) in the identifier contained in the digital certificate used. This allows a particular node to be used by different users at different times. However, to enable a node to be used by more than one user, it is necessary to define which certificate will be used to authenticate the basic Manet services (e.g. routing and autoconfiguration).

### 3.2.2.3. Security policy considerations

One of the most critical points - that is still open - regarding establishment of trust in Manets is how to translate the security policy adopted in the network application environment into objective criteria that allow this policy to be imposed on users via the security mechanisms adopted. This is obvious when analysing the various proposals for Manet trust models using digital certificates [62, 66, 119]. In many of these proposals, the solution is structured by first accepting the security policy, mainly as regards the issue of new certificates.

In [119] certificates have to be issued first by a centralised certification authority and certificates can only be renewed collaboratively. This is not much of a security policy, since it is left to the discretion of the CA. In [66] the centralise CA only issues the certificates for the first K nodes. The certificates for the other nodes are issued collaboratively, based on a policy that has to involve some kind of off-line verification, with presentation of physical evidence of the requester’s identity. In [71, 116], by the same authors as [66], a request for a certificate is always served by all the nodes, unless the requester is already identified as a misbehaving node. This approach does not prevent an attacker, if discovered, from changing identity and staying on the network, or even forging multiple identities and obtaining the secret certification key after obtaining K different partial private keys. In reference [52] certificates are issued on a peer-to-peer basis, with no uniform criterion for the policy of issuing certificates to the whole network.
As regards the renewal of certificates, the approaches in [66,71,116] allow renewal only if the requester is a “well-behaved node, i.e. if there are no accusations against it. In this case, there are no mechanisms to ensure that the distributed certification service continues to implement this policy for the issue of certificates. The problem is even more critical in the case of the issue of parts of the private certification key.

In any case, the minimum requirements of the security policy of the distributed certification services are as follows:

- The issue of certificates requires the requester to be uniquely identified, otherwise the solution is subject to breaches by an adversary forging multiple identities (Sybil attacks). With this condition it is difficult for nodes not having a prior certificate; the security policy must establish the criteria to be used to verify the requester’s identity.

- The issue of parts of the private certification key and the renewal of certificates must be allowed only for nodes already having a valid certificate. This certificate must authenticate requests for these services, as a minimum guarantee for the provision of the service to the other nodes.

These minimum requirements may be sufficient for Manet application scenarios with a low probability of compromised nodes, such as a group of students in a classroom. In situations where compromised nodes are more likely, however, or even where the impact of a security incident is more critical, the security policy may require more complete verification of the requester’s identity before providing any of these services, since the presentation of a valid certificate may be insufficient to guarantee that the certification policy is being obeyed at all times, since the requester may be a compromised node.

In our proposal, the various parameters defining the security and performance requirements of the certification system can be configured in accordance with the security policy adopted, always observing the minimum requirements defined above. Examples of these configurable parameters are the issue and revocation of certificates, rules for distribution and storage of counter-certificates (i.e. revoked certificates), period of validity of an issued/validated certificate, etc. This approach allows the security solution to be matched to the security policy, rather than restricting the scope of use of the model in environments with pre-defined policies.
3.2.3. Manet authentication extension (MAE)

The distributed certification service presented in the previous section provides a robust and efficient solution for distribution of trust. However, this trust has to be imposed in all the network transactions to be secured. This applies to routing-protocol and autoconfiguration messages, for example. For this to be possible, it is necessary to append an authentication extension to these messages, containing the object(s) authenticating the messages. There are standardised protocols that are efficient for peer-to-peer message exchanges, where a relationship of trust already exists between the peers. This is the case of the IPSec protocol [109], for secure communication between two networks or two hosts, and SSL/TLS [31], for establishing a secure communication session between two applications. However, these solutions are adequate only for cases where there is end-to-end authentication between two entities that already have routing between them. IPSec and SSL/TLS are therefore unsuited to the security needs of Manet routing and autoconfiguration protocols. Apart from this, some routing protocol considered have fields that are altered as the messages pass through the network. These fields also have to be authenticated, requiring the design of special authenticating objects and the use of multiple authentication objects in these cases.

In this context, we are planning a Manet authentication extension (MAE) that allows our trust model to be used to authenticate message-orientated applications having requirements exceeding the capabilities of IPSec. Our MAE, discussed in more detail in the next chapter, allows multiple authentication objects to be incorporated in a single authenticated message and is fully adapted to operate in conjunction with the certificates of the trust model.

3.2.4. Detection of and response to intrusions in Manet

While the certification and authentication services provide basic preventive security conditions, preventing untrusted nodes from attacking the Manet, a corrective security service is also needed to handle the requirements of detecting attacks by compromised nodes or adversaries spoofing those nodes, and to eliminate those nodes from the network’s collaborative services.

Two basic approaches to corrective mechanisms are being studied and developed. Firstly, when adversary nodes can be explicitly identified, notably when they carry out attacks involving actions such as fabrication, modification and spoofing of autoconfiguration and routing protocol messages, or non-cooperation attacks. The corrective counter-measure that is
designed to re-establish normal operation of the network and its associated services is the elimination of the adversary nodes in the network, by revoking their certificates. These nodes are then no longer able to take part in or corrupt the Manet’s collaborative services, as their messages can never be authenticated. Alternatively, where the origin of the attacks cannot be precisely identified, one can attempt to mitigate the effect of attacks by avoiding the use of network paths that have presented problems or by applying packet filters (at network level) or message filters (at application level) for anomalous information flows. While in the first case we aim to positively identify the attack and the adversary, in the second we detect anomalous or degraded operating conditions.

These two scenarios can readily be related to the two aspects of intrusion detection techniques currently under discussion. In detection of misuse-related intrusion, an attack is identified by recognising its characteristic traces, after being identified and formally expressed in an attack signature. In this case, the identification of the attack is positive and, if carried out accurately, it satisfies the conditions for accusation and elimination of adversaries. In detection of behaviour-related intrusion, operating conditions are detected that deviate from the normal conditions previously identified and formally expressed in a model of the system’s behaviour. Attacks are not explicitly detected, but degraded operating conditions can be mitigated by avoiding network paths or nodes that present problems or failures, by applying selective filters in routing messages, or by avoiding relaying and/or processing of packets exhibiting an anomalous use profile, using packet filters (firewalls). These counter-measures are effective in the event of packet flooding attacks (e.g. DDoS) and use of network scanners.

An intrusion detection and response service (IDS) specially designed for Manets provides the corrective security mechanism that completes our security solution, satisfying all the requirements discussed in section 3.1.3. This IDS is fully distributed in operation. A local IDS (L-IDS) is located in each network node, and these collaborate with the L-IDS in neighbouring nodes to collect information, execute the detection algorithm and coordinate the response to intrusion. The IDS has a modular architecture and can easily be extended to execute various types of data collection, detection strategies and response mechanisms.

The basic protocol for a collaborative response to intrusion consists of three stages, illustrated in Figure 3-3. Initially, the L-IDSs collaborate to detect an attack, using mobile agents. Whenever an attack is detected, an IDS generates a signed accusation (alert) against the adversary and broadcasts it to the whole network (step 1). This process is repeated by all L-IDSs detecting the same attack (step 2). When the alert correlation mechanism in an L-IDS identifies K accusations from different L-IDSs against the same node, a process is initiated to
revoke the certificate of the accused node. Then, a request to form a dynamic coalition is issued by all the L-IDSs identified as accusers of the compromised node (step 3). These nodes sign a partial counter-certificate against the compromised node and send it to the requester (step 4). The requester reconstitutes the complete counter-certificate from the K partial counter-certificates received and broadcasts it to the whole network, so completing the revocation of the certificate (step 5). All nodes then update their local CRL.

Figure 3–3 –Collaboration protocol for intrusion response
3.2.5. Integrated security services

The security solution presented in this work consists of an integrated set of services that together provide a distributed and collaborative security solution that satisfies the general requirements for Manets and the security requirements discussed in section 3.1.3. In particular, the certification (L-CERT) and authentication (MAE) mechanisms provide preventive security, while the intrusion detection and response mechanisms (L-IDS) provide corrective security. This combination, illustrated in Figure 3-4, is a salient feature of the proposed security model.

Besides the services explicitly mentioned in the previous paragraph, the security model can be extended to promote interaction with other security services already designed, such as:

- local firewall service, which provides a network-level packet filter available locally for configuration and reconfiguration in accordance with the security policy and intrusion-response strategy, and the establishment of cryptographic tunnelling using the IPSec protocol to make a private connection between two networks over a virtual private network (VPN);
- SSL/TLS service, using L-CERT to create secure (peer-to-peer) sessions between applications.

Figure 3-4 shows a view of the implementation of the proposed security model. This view can be supplemented by the protocol architecture model for the proposed solution, as shown in Figure 3-5.
Figure 3-4 - Implementation view of the security model

Figure 3-5 – Protocol architecture of the security model
Finally, although these services are beyond the scope of this work, the design and development of other distributed security services can be considered, such as:

- distributed firewall service (L-Firewall), which provides a collaborative response for reconfiguring the packet filters in various different nodes in order to mitigate attacks involving packet traffic passing through more than one network node (e.g. DDoS);
- distributed security policy management service (L-SPM), designed mainly to establish and distribute, in a secure and cooperative manner, updates of the security policy and of the security policy configuration bases (e.g. parameters of the distributed certification and authentication services, attack-signature database, firewall rules, etc.

Figure 3-6 shows a view of this extended security model, including L-Firewall and L-SPM services.
Figure 3-6 - Implementation view of the extended security model
4. CERTIFICATION AND AUTHENTICATION IN MANET

This chapter presents details of the proposed certification and authentication protocols and algorithms adopted in the proposed security model. These mechanisms are then applied in the preventive security of routing and autoconfiguration protocols. In both cases, modification, fabrication and/or impersonation attacks on protocol messages are discussed and details are given of preventive protection against such attacks.

4.1. CERTIFICATION SERVICE IN MANET

The certification service presented in this section is an adaptation of [66,71], which provides scalable algorithms and protocols for the secure distribution of certification services between nodes on a Manet. The service is available to any network node providing it is possible to locate a coalition with a minimum number of nodes (K). The basic objectives of this certification service for Manets are:

- Distribution and localisation. While distribution is linked to the absence of centralised entities in Manets, localisation is linked to performance and scalability requirements.
- Bootstrapping and self-starting: The need for a centralised entity (dealer) for the start up of new nodes must be minimised, or even eliminated. If a dealer is required, it must only be during the network’s bootstrap phase.
- Proactive updating of the certification secret: Since a set of K nodes can carry out all certification services, the compromising of this number of nodes by the same attacker (or by cooperating attackers) will break the system’s security. By periodically updating the parts of the key held by the nodes the solution shows a relative degree of tolerance to intrusion between each update cycle.

The distribution of the functionalities of a certification authority is based on the RSA cryptographic protocol [98] and achieved by sharing the private key of the certification authority among all nodes participating in the network using the threshold cryptography technique [102] (for further details, see Annex II). Consider an ad hoc network in which each node \( v_i \) possesses a pair of RSA keys \( \{K_D, K_U\} \), where \( K_D = <d, n> \) is the private key and \( K_U = <e, n> \) is the public key, with \( n \) as the RSA computational model.
Equally, the system’s DCA must possess a pair of RSA keys, \( \{ K_{IAC}, K_{UAC} \} \), in which \( K_{IAC} = <d_{AC}, n_{AC}> \) is the private certification key, used to sign all certificates of the network nodes. Any one of these certificates can be verified by the system’s public key (\( K_{UAC} \)), which is known to all.

According to the collaborative trust model K-by-N, introduced in the previous chapter, \( d_{AC} \) is shared by the network nodes using threshold cryptography. In addition to its pair of keys, each \( v_i \) node possesses a part of \( d_{AC} (K_{IAC}) \) which must be kept private, with the same level of security as for its own private key. Any group of \( K \) nodes, within the \( N \) holders of the parts of \( d_{AC} \), can form a certification coalition and function as a certification authority. Equally, it is not possible for any single node to know \( K_{IAC} \). Even a coalition of \( K \) nodes is incapable of recovering the private key from the system, unless the nodes exchange their individual parts of \( K_{IAC} \).

The threshold \( K \) represents a compromise between service availability and intrusion tolerance, so that a group of adversaries needs to destroy \( (N - K + 1) \) holders of parts of the private key to disable the service (since this would prevent self-starting) and compromise at least \( K \) parts of the private key to steal \( K_{IAC} \). When the system is being constructed, a careful analysis must be made to choose \( K \). The lower the value of \( K \), the easier it is to break \( K_{IAC} \). However, as \( K \) increases the system’s security increases but tolerance to faults decreases. With the appropriate choice of \( K \), certification coalitions are dynamically established in the 1-hop neighbourhood to provide the certification service, maintaining the localisation characteristic of the solution.

The certificates generated are aimed at certifying the public keys of each network node, as in a common cryptographic system. Thus in the security mechanism used, each network node possesses a certificate signed by the secret key \( K_{IAC} \) in the format

\[
\text{CERTIFICATE}_{vi} = <v_i, K_{U_i}, T_{sign}, T_{expire}, CERT_i>,
\]

where:

- \( v_i \) is the identifier of the node;
- \( K_{U_i} \) is its public key;
- \( T_{sign} \) is a timestamp with the date and time the certificate becomes valid; and
- \( T_{expire} \) is a timestamp with the date and time the certificate expires.
- \( CERT_i \) is the certificate’s signature with \( K_{IAC} \).

61
Two methods are used to verify the validity of the certificate:

- Implicit revocation of the certificate: Each entity must renew its certificate at each interval of $T_{\text{renew}}$, before expiry ($T_{\text{expire}} \leq T_{\text{sign}} + T_{\text{renew}}$).

- Explicit revocation of the certificate: Counter-certificates are signed collaboratively by the DCA to revoke certificates of compromised nodes independent of their expiry date. Counter-certificates possess the format $\text{COUNTER\_CERTIFICATE}_i = < \perp v_i, KU_i, T_{\text{sign}}^{+1}, T_{\text{expire}}, \text{CERT}^{+1} >$, where $\perp$ denotes counter-certification and $T_{\text{sign}}^{+1}$ is the time when the counter-certificate was created. The counter-certificate is also signed with $K_{\text{DC}}$ ($\text{CERT}^{+1}$). Counter-certificates are kept locally by all of the Manet’s nodes in certificate revocation lists (CRL). Only counter-certificates for revoked certificates that have not expired still need to be in the CRL.

4.1.1. Basic Certification Services

4.1.1.1. Issue and Renewal of Certificates

A node without a certificate or that needs to renew its certificate must request a new certificate from other nodes that provide the certification process. The certification policy used must specify how the nodes that receive certificate requests must meet such requests. Different policies can be specified for the issue and renewal of certificates, such as:

- Issue, according to a specific identification policy from the requester;
- Process manually, asking the user of the node that receives the request to decide if the certificate should be issued;
- Deny the service explicitly, generating a return message for the requesting node;
- Deny the service silently, discarding the request without sending any notification to the requesting node;
- Others.

Technically, there is no difference between the issue and renewal processes for a certificate. However, the policies for such services can be specified separately. After all, criteria for signing a certificate from a new entity must be much more stringent than for signing a certificate from an entity that already belongs to the network and that has no record of misbehaviour.
As discussed in the previous chapter, the service is conducted in four stages (Figure 3-2): (1) and (2) formation of the coalition, (3) service request, and (4) collection and processing of replies. These stages are discussed below.

(1) When a node \( v_i \) needs to receive a certificate, it sends out a coalition request message (COALITION_REQ) containing REF_N, where REF_N is the transaction reference number which is generated randomly and must be included in all other messages in the process.

(2) Any node that possesses a part of \( K_{IC} \) must reply to the request by sending an authenticated coalition acknowledgement message (COALITION_ACK) to the requester, containing their identity \( v_j \) and REF_N.

(3) The requesting node collects replies (COALITION_ACK) until it is possible to form a coalition of \( K \) nodes\(^25\). The requesting party itself can be part of the coalition if it possesses a part of \( K_{IC} \). The set of identities of the nodes in the coalition is given by \( \beta = \{ v_j / v_j \in \text{coalizão} \} \). Then the node distributes a certificate request message (CERT_REQ), containing the request for the certificate, \( < v_i, KU_i, T_{sign}, T_{expire}, CERT_REQ_i > \), where \( CERT_REQ_i \) is the signature of the certificate request with \( KU_i \), together with \( \beta \) and REF_N. Where a certificate is to be issued to a node without a valid certificate, the information regarding the identity of the node required by the certification policy is added to the request. Alternatively, in a certificate renewal, the certificate that is still valid (i.e. not expired) is added to the request.

(4) Upon receipt of the CERT_REQ, the nodes belonging to the coalition identify their own identity in the coalition. If a valid certificate is sent together with the message, the request is treated as a certificate renewal. Otherwise, it is considered a certificate request for an uncertified node. The appropriate certification policy is applied and each node in the coalition decides whether to agree to the request. In the case of a positive response, the partial certificate \( CERT_{ij} \) is calculated and sent to the requesting party, in a PARTIAL_CERT message.

\( CERT_{ij} \) is calculated as follows. Each node \( v_j \) that decides to meet the request of \( v_i \) calculates its additional key \( K_{ij}^{\beta} \) according to Eq. 4-1:

\(^{25}\) The requesting node waits for COALITION_ACK messages during a period \( T_{coalition} \). If it is not possible to form a coalition with the replies received during this time, a new COALITION_REQ is made and an exponential-backoff process is adopted for the waiting period \( T_{coalition} \).
\[ K_{I_{j,\beta}} = K_{I_{c_{k_j},L_{j,\beta}}}(0) = K_{I_{c_{k_j}}} \prod_{r \in [0,\pi_j]} \frac{v_r - v_j}{v_r - v_j} \mod n_{n_{AC}} \] \tag{4.1}

where: \( L_{j,\beta}(v_i) \) are Lagrange coefficients for the interpolation of the generator polynomial \( f(x) \), given by (Eq. 4.2):

\[ L_{j,\beta}(v_i) = \prod_{r \neq i} \frac{v_i - v_j}{v_r - v_j} \] \tag{4.2}

\( K_{I_{j,\beta}} \) is called the additive key, since the Lagrange interpolation gives (Eq. 4.3):

\[ \sum_{j \neq 0} K_{I_{j,\beta}} = \sum_{j \neq 0} K_{I_{c_{k_j},L_{j,\beta}}}(0) = d_{n_{AC}} \mod n_{n_{AC}} = t \cdot n_{n_{AC}} + d_{n_{AC}} \] \tag{4.3}

where: \( 0 \leq t < k \).

Next, \( v_i \) calculates the partial certificate of \( v_i \) (\( CERT_{i,\beta} \)), signing the hash of the new certificate (\( cert \)) with the additive key \( K_{I_{j,\beta}} \) (Eq. 4.4):

\[ CERT_{i,j} = (cert)^{K_{I_{j,\beta}}} \mod n_{n_{AC}} \] \tag{4.4}

Finally, after receiving the \( K \) partial certificates, \( v_i \) combines them to generate a candidate for the signing of the certificate (\( CERT_i \)), according to Eq. 4.5:

\[ CERT_i = \prod_{j \neq 0} CERT_{i,j} \mod n_{n_{AC}} = (cert)^{\sum_{j \neq 0} K_{I_{j,\beta}}} \mod n_{n_{AC}} = (cert)^{t \cdot n_{n_{AC}} + d_{n_{AC}}} \mod n_{n_{AC}} \] \tag{4.5}

Noting that \( CERT_i = (cert)^{t \cdot n_{n_{AC}}} \mod n_{n_{AC}} \), it can be seen that this candidate \( CERT_i \) differs from \( CERT_i \) by a constant. This polarisation can be removed by the following algorithm:
Algorithm 1 – Calculation of $CERT_i$

**Inputs:** $CERT_i'$ (candidate to sign the certificate) and $cert_i$ (hash of the certificate to be signed).

**Output:** $CERT_i$ (signature of the certificate).

1: $Z := (cert_i)^{+\infty} \mod n_{\mathcal{C}}$
2: $r := 0, Y := CERT_i'$
3: while $j < K$ do
4: $Y := Y \cdot Z \mod n_{\mathcal{C}}; \quad j := j + 1$
5: if $(cert_i = Y^{+\infty} \mod n_{\mathcal{C}})$ then
6: break while
7: end if
8: end while
9: saida : $Y = CERT_i$

If any fault occurs in any of the nodes in the coalition and one of the partial certificates is not received, the partial certificates from the other nodes become useless, and the whole process must start again. In order for this process to be executed with 1-hop communications alone, the requesting node must have at least $K$ neighbouring nodes.

[71] presents an alternative procedure that would eliminate the need to establish the coalition explicitly. In this case, instead of using encryption key $KI_{\mathcal{C}j}$ in Eq. 4-4, $KI_{\mathcal{C}j}$ is used directly. This avoids steps (1) and (2). However, a failure in this approach was noted while developing this work, since there are no mathematic guarantees that the algorithm for calculating $CERT_i$ will give a correct value in a maximum of $K$ iterations. Indeed, in order to obtain the correct value for $CERT_i$ in this case, the number of iterations of the aforementioned algorithm is potentially very high, which makes using it computationally unfeasible.

### 4.1.1.2. Certificate Revocation

If, according to the network’s security policy, the certificate of a node $v_i$ is deemed to be compromised, it must be revoked. The revocation of a certificate occurs with the signature of a counter-certificate for this node. When a counter-certificate is signed ($CERT_i^-$), it must be flooded throughout the network and added to the local CRL of all nodes. This counter-certificate is held in the CRL until the time $T_{\text{expire}}$. It will depend on the policy for the reissue
of certificates to define if a node for which the certificate has been revoked will have its certificate reissued after time $T_{rev}$.

The revocation of certificates occurs in the same way as the issue of new certificates. Initially it is necessary to form a coalition of $K$ nodes (COALITION_REQ and COALITION-ACK messages). Next, a request is generated for the revocation of the certificate (CERT_REVOKE_REQ) containing the certificate to be revoked together with the coalition data ($\beta$). Finally, the nodes in the coalition sign the partial counter-certificate ($CERT_{+i,j}^j_i$) with its additive key $K_{i,j}$, as shown in equation Eq. 4.4. The partial counter-certificates are returned to the requester (CERT_REVOKE_ACK), which recovers the requested counter-certificate using the same certificate recovery process shown in Eq. 4-5 and in Algorithm 1, and floods it throughout the network.

The revocation of certificates can be requested in two cases: self-revocation and revocation due to the detection of an intrusion. Self-revocation occurs when a node decides to revoke its own certificate (e.g. because it believes its private key has become unsafe). In this case the node itself signs the request for the revocation of its certificate with its private key, and this is immediately accepted by the nodes in the coalition, which are able to verify the signature of the request. In the case of revocation following the detection of an intrusion, as discussed in the previous chapter, one of the nodes of the Manet (requester) must collect at least $K$ accusations (alerts) signed by different nodes against the compromised node. This node forms a coalition with other nodes that generate accusations. After verifying the accusations, the coalition members sign the counter-certificate and send it back to the requester.

### 4.1.1.3. Certificate Validation

Certificates must be validated whenever a message is received that is signed by a certificate seen for the first time by one of the nodes in the Manet. In general, the local verification of the validity of a certificate involves:

1. verification of the dates of issue ($T_{sign}$) and expiry ($T_{expire}$);
2. query of the local CRL to check if the certificate has not been explicitly revoked; and
3. verification of the signature of the issuer.

In order to verify the signature of the issuer (step 3), the certification path must be constructed between the certificate that is being validated and the certificate of the issuer that
signs it (usually the network DCA). The certificates of the trusted CAs are kept in a cache of valid certificates. If the issuer is a trusted CA, the signature is verified by extracting the issuer’s public key from its certificate (usually $KI_{AC}$). If the certificate that is being validated has been signed by a CA that is not directly trusted by the validing node, the certification path is constructed repeatedly until a certificate is found on the certification path from a trusted CA.

Since the validation process involves significant computational cost, it is useful for certificates that have recently been validated to be kept in a local cache of valid certificates for later consultation. Obviously, revoked or expired certificates must be removed from the local cache of valid certificates.

### 4.1.2. Bootstrapping the system with a Dealer

For certification services to be available, there must be at least $K$ nodes holding parts of $KI_{AC}$ in the Manet. The conventional way to start these first nodes involves a centralised entity at the time the network is formed, called a dealer. The dealer generates the pair of RSA keys $\{KI, KU\}$ and a random polynomial $f(x)$ of the level $K-1$ (Eq. 4-6):

$$f(x) = d_{AC} + a_1x + ... + a_{K-1}x^{K-1} \quad \text{Eq. 4-6}$$

where:

$f(0) = d_{AC}$; and the coefficients $a_1, ..., a_{K-1}$ are generated at random ($a_1, ..., a_{K-1} < n_{AC}$).

The partial keys of the entities $v_i$ are given by (Eq. 4-7):

$$KI_{AC,i} = (f(v_i) \mod n_{AC}) \quad \text{Eq. 4-7}$$

Each of the initial $K$ nodes of the network receives a partial key $KI_{AC,i}$ and this is sufficient for the network to function. The dealer also distributes a set of values used in verifying the partial certificates, called witness polynomials $f(x)$, i.e. $\{g^d_{AC}, g^n_{AC}, ..., g^{n_{AC}}\}$, where $g$ is known by the entire network, as well as the public key $KU_{AC}$. After the network has been initialised, the dealer node and the polynomial are no longer necessary for the system to function and they are discarded. From this point on, the nodes that have already been initialised are responsible for issuing certificates and parts of $KI_{AC}$ to new nodes.
4.1.3. Issue and Proactive Updating of Parts of the Private Key

4.1.3.1. Issue of Parts of the Private Key

Every node $v_i$ that possesses a valid certificate on the network can acquire a part of $KI_{AC}$ (e.g. $KI_{AC,i}$). For the security and robustness of the system, the dealer node used solely for initiating the network does not remain connected when the network starts functioning. Therefore, there must be a mechanism for the self-organised issue of parts of the private certification key. The idea is that entities that have already been initialised (e.g. that possess part of the private key) can themselves initialise new entities, which in turn can also participate in the process of initialising new nodes. This eliminates the need for a central node in the system. This process is presented in this section.

To generate a new part of the private key ($KI_{AC,i}$) the node $v_i$ must choose a coalition of $K$ nodes $\beta = \{v_j / v_j \in \text{coalizão}\}$ and send them a request for part of the private key (SECRET_SHARE_REQ), together with $\beta$ and its valid certificate. When $v_j$ receives the request, it checks the certificate of $v_i$ and the list of CRL revocations. If it decides to grant the request, it calculates its partial key $P_{j,\beta}$ for $v_i$ (Eq. 4-8).

$$P_{j,\beta} = KI_{AC,i} L_{j,\beta}(v_i) \mod n_{AC}$$  \hspace{1cm} \text{Eq. 4-8}

Using the partial keys from each of the $K$ nodes, $v_i$ can construct $KI_{AC,i}$ by Lagrange interpolation (Eq. 4-9):

$$KI_{AC,i} = f(v_i) = \sum_{j=1}^{K} KI_{AC,j} L_{r,j}(v_i) = \sum_{j=1}^{K} P_{j,\beta} \mod n_{AC}$$  \hspace{1cm} \text{Eq. 4-9}

However, the nodes will not necessarily send these partial keys directly, since $L_{j,\beta}(v_i)$ only depends on the IDs of the coalition and node $v_i$ can easily discover the part of the private key of $v_j (KI_{AC,j})$ from the partial key $P_{j,\beta}$. If $v_i$ receives $K$ of any partial keys (e.g. $P_{1,\beta}, ..., P_{K,\beta}$) it can discover the partial keys of the coalition and recover the system’s private key ($KI_{AC}$).

To resolve this problem, $v_i$ must only receive the sum of the keys $\{P_1, P_2, ..., P_i\}$. For this, a scrambling scheme is used, with the following steps:
(1) To generate a new part of the private key, $K_{AC,i}$, the node $v_i$ chooses a coalition of $K$ nodes, sending out a coalition request message (COALITION_REQ) containing REF_N together with its valid certificate.

(2) Any node that possesses a part of $K_{AC}$ can answer the request. The certification policy is applied and each node in the coalition decides whether or not to meet the request. In the case of a positive response, an authenticated message of coalition acknowledgement (COALITION_ACK) is sent to the requester, containing its identity $v_j$ and REF_N.

(3) The requesting node collects the replies (COALITION_ACK) until it is possible to form a coalition $\beta = \{v_j / v_j \in \text{coalizão}\}$ of $K$ nodes. Then the node sends out a request message for part of the private key (SECRET_SHARE_REQ1), containing $\beta$ and REF_N.

(4) Upon receipt of SECRET_SHARE_REQ1, the nodes belonging to the coalition identify their own identity in the coalition. Each node $v_j$ generates a random factor $d_{r,j}$ for each member $v_r$ of $\beta$ with an ID smaller than its own. Each factor $d_{r,j}$ is encrypted with the key $KU_r$ of the member that will receive it and sent to the requesting node ($v_i$) in SECRET_SHARE_ACK messages.

(5) The requesting node sends the encrypted factors $d_{r,j}$ to their recipients (SECRET_SHARE_REQ2).

(6) After deciphering all the factors received, each $v_j$ calculates a scrambled partial key ($\overline{P_{j,\beta}}$), subtracting all factors received from its partial key $P_{j,\beta}$ (i.e. sent by nodes with an ID greater than their own) and adds to this same key the factors it has sent to the nodes with an ID smaller than its own. Finally, it sends the result back to $v_i$ (PARTIAL_SECRET_SHARE).

The calculation of $\overline{P_{j,\beta}}$ is shown in Eq. 4-10.

$$\overline{P_{j,\beta}} = P_{j,\beta} + \sum_{r < j} \text{sign}(v_r - v_j)d_{r,j}$$  \hspace{1cm} \text{Eq. 4-10}

where: $\text{sign}(v_r - v_j) = 1$ if $(v_r - v_j) > 0$ and $\text{sign}(v_r - v_j) = -1$ if $(v_r - v_j) < 0$.  

69
After receiving the scrambled partial keys from the entire coalition, the node $v_i$ calculates its secret key $K_{AC,i}$ as shown in equation Eq. 4-11.

$$K_{AC,i} = \sum_{\alpha \in \alpha} p_{j,\beta} = \sum_{\alpha} \left( p_{j,\beta} + \sum_{\beta \neq i, j} \text{sign}(v_i - v_j) d_{r,j} \right)$$

Eq. 4-11

$$= \sum_{\alpha} p_{j,\beta} + \sum_{\beta \neq i} \text{sign}(v_i - v_j) d_{r,j}$$

$$= \sum_{\alpha} p_{j,\beta}$$
The issue of parts of the private key is shown in Figure 4-1.

4.1.3.2. Proactive updating of Parts of the Private Key

There are two ways of protecting the system from having its secret broken over time: changing the system keys \( \{ K_{IC}, K_{UC} \} \) from time to time or updating all parts of the private key periodically. The first solution can be carried out by restarting the system periodically, although this is not the best option since the certificates and partial keys of the entire network would have to be updated at the same time.
In the second solution, all parts of the private key are updated periodically. Thus the system tolerates up to $K - 1$ compromised nodes between these updates, since $KI_{AC}$ can only be obtained with $K$ parts of the private key. Two alternatives are proposed in [71] to achieve an efficient and scalable update of the parts of the private key. The first is a simple sequential process based on the issue of parts of the private key presented in the previous item. Initially, a coalition of $K$ nodes updates their own parts of the private key by applying the protocols proposed in [44]. Then the issue protocol of the parts of the private key updates the other nodes in the network. The second alternative involves the parallel updating of the partial keys of the entire network by rapid convergence.

In this second case, the time is split into two periods ($T_{update}$) and each of these is composed of a phase for updating the partial keys and an operational phase. During the operational phase the nodes renew their certificates periodically.

At the start of the update phase, a coalition of $K$ nodes from the system generates an updating polynomial $f(x) = f_1x + ... + f_{K-1}x^{K-1}$ where $f_0(0) = 0$. $f_u$ is then encrypted with $KU_{AC}$ and signed by the coalition with its parts of $KI_{AC}$. In this way, an attacker is prevented from discovering $f_u$ and from simulating a coalition to conduct a false update of the partial keys.

The encrypted polynomial, together with its signature, is sent out across the whole network. As soon as it is received by a node, the node verifies its signature with $KU_{AC}$ and requests the update service of partial keys to $K$ nodes, which do not need to have their keys updated to provide the service. The $K$ nodes calculate $U_i = f_i(v_i)$ and send it to $v_i$, which simply adds this value to its own part of the current private key $KI_{AC,i}$ to generate a new key $KI_{AC,i}'$ (Eq. 4-12). The part of the old private key will be kept during the transition and will only be eliminated after the updating phase.

$$KI_{AC,i}' = KI_{AC,i} + U_i = f(v_i) + f_i(v_i) = f_{new}(v_i)$$  \hspace{1cm} \text{Eq. 4-12}$$

Note that, by defining $f_{new} = f + f_u$, we have $f_{new}(0) = f(0) + f_u(0) = d_{AC}$. With this, the polynomial is updated and $KI_{AC}$ remains unchanged.
4.1.4. Verification of Keys and Certificates

When a node $v_i$ is initialised, whether by the dealer or by the network, it verifies the validity of its part of the private key $KL_{AC}$ by testing the equality shown in equation Eq. 4.13:

$$g^{KL_{AC}} = g^{d_{AC}} \cdot (g^n)^{v_i} \cdot (g^{\alpha_1})^{v_1} \cdots (g^{\alpha_N})^{v_N}$$

Eq. 4.13

where: $\{g^{d_{AC}}, g^n, \ldots, g^{\alpha_N}\}$ are the witnesses of polynomial $f(x) = d_{AC} + a_1 x + \ldots + a_k x^{k-1}$ and $g$ is a value known by the entire network.

In the case of a collaborative issue of parts of the private key, if the equality of Eq. 4.13 is not verified, $v_i$ will know that the part calculated is not valid. This can be the result of a corrupted node controlled by an attacker or of a node that has made a mistake. However, $v_i$ has no way of identifying which of the partial keys $P_{ij}$ is corrupted. To solve the problem, when the coalition changes the scrambled factors $d_{i,j}$, they also send to $v_i$ a witness $g^{d_{i,j}}$.

For each scrambled $\overline{P_{i,j}}$, $v_i$ verifies the equality of the equation Eq. 4.14:

$$g^{KL_{AC,i}} = \prod_{\sigma_{i,j}} (g^{d_{i,j}})^{\text{wgn}(v_i)}$$

Eq. 4.14

where: $g^{KL_{AC,i}} = g^{d_{i,j}} \cdot (g^n)^{v_1} \cdot (g^{\alpha_1})^{v_1} \cdots (g^{\alpha_N})^{v_N}$ is the witness of part of the private key $v_j$.

If Eq. 4.14 is not verified, $\overline{P_{i,j}}$ is defective. Then $v_i$ marks $v_j$ as being compromised and sends the accusation and all the proof against $v_j$ to the network.

With certificate verification, $v_j$ prepares $\{A_1, A_2, r\}$ as proof for a node $v_i$ of the validity of its partial certificate $CERT_{ij}$. For this, $v_j$ generates a random value $u$ and calculates $A_1 = g^u$ e $A_2 = (cert)^u$. Then, $v_j$ calculates $c = \text{hash}(g^{KL_{AC}} | CERT_{ij} | A_1 | A_2)$ and $r = u - cKL_{AC}$. Next $\{A_1, A_2, r\}$ is signed by $v_j$ and presented to $v_i$, which in turn, as the recipient of $\{A_1, A_2, r\}$ or only a neighbouring node monitoring the behaviour of $v_j$, calculates $c$ and verifies if $A_1 = g^c \cdot (g^{KL_{AC}})^c$ and $A_2 = (cert)^c \cdot (CERT_{ij})^c$. If all the answers correspond, the certificate is valid.
4.1.5. Local Base for Certification Data

Some certification data must be kept permanently in each node $v_i$ that takes part in the certification process (L-Cert). This data includes: the private key of the node ($K_i$) and its own certificate ($CERTIFICATE_{vi}$), valid certificates for each of the trusted CAs, its own part of the private certification key ($K_{AC,i}$) with the current version of the key share, an encrypted polynomial with $K_{UAC}$ and the corresponding version of the update of the share.

Additionally, the L-Certs dynamically maintain a table with the recently validated certificates (cache of valid certificates) and a table with the revoked certificates (local CRL). Records in these tables must be removed automatically when they expire (i.e. current time = $T_{expire}$). The records in the table of valid certificates are kept during a maximum period of time ($T_{out}$), which can be configured in accordance with the certification policy. When a certificate in this table is consulted, the time for the deletion of the record from the table is altered to $min(current\ time + T_{out}, T_{expire})$. This table also has a maximum size ($N_{CACHE}$), which can also be configured according to the certification policy. When this maximum number is reached, the records with the closest deletion time are removed from the table to free up space for new records to be included.

In this work, the construction, management and synchronisation of the cache of valid certificates and the local CRL are configured depending on the type of routing protocol used. The options for the management of the bases of certificates are the proactive or on-demand construction of the tables.

In the proactive distribution of valid certificates, the sender must attach its certificate to the messages sent. If the certificates are sent in all the messages, this provides the maximum availability of the certificates. However, this approach implies a considerable network overhead. A possible optimisation involves only attaching the certificate to some of the messages sent. Alternatively, nodes can explicitly request the certificate from other nodes when they are required and are unavailable in the cache of valid certificates. As occurs with on-demand routing protocols, the owner of the requested certificate is not the only party that can grant this request. Other nodes that possess the requested certificate in their cache can respond to the request.

There needs to be a synchronisation strategy for the local CRL, allowing roaming or recently-arived nodes to have the local CRL updated. Once again, proactive and reactive approaches are possible. In the proactive approach, each node periodically changes (e.g. at each $T_{CRL\_SYNC}$) the information in its CRL with its neighbours so they can all possess the...
complete list of revoked certificates on the network. In the reactive approach, a node explicitly requests an updated copy of the CRL from its neighbours.

### 4.1.6. Initiation of a New Node prior to Autoconfiguration and Routing

For the issue and renewal of certificates, at least K nodes must be available and must function collaboratively as a certification authority. All nodes without certificates must undergo this process to obtain a valid certificate and thus benefit from the basic network services (e.g. autoconfiguration and routing). However, in this process it is assumed that nodes already have an IP address to communicate with the other nodes on the network and thereby obtain a certificate. This is not the case if the node requires the autoconfiguration service to obtain its IP address, since this service is only available after authentication. There are two proposed solutions to solve this problem.

Given that the messages needed for the issue of the new node’s certificate are only transmitted in the (1-hop) neighbourhood of the requesting node, a simple solution consists of the requesting node randomly choosing a temporary IP address and conducting the certificate request procedure using this address. Nodes that grant the request send replies to the requesting node with the TTL of the IP package configured with the value 1 (one), ensuring the package containing the reply is not sent further than the neighbouring nodes. The requesting node collects all replies from the neighbouring nodes even if they do not contain its MAC address. This is possible because the chosen address may have been duplicated. After recovering the certificate, the requesting node starts the autoconfiguration process. The temporary IP address is then discarded and no longer used by this node.

It is possible for duplicated temporary IP addresses to be allocated since the choice is made at random. However, since nodes that have not been initialised only send certification service requests to neighbouring nodes, the problem only occurs if the duplicated IP addresses are neighbours of the requesting node. This can be avoided by adopting some kind of routing protocol using HELLO messages. The requesting node monitors these messages and identifies all the IP addresses of the neighbouring nodes. Another solution is the use of a special range of IP addresses for temporary allocation to the nodes, reducing the probability of duplicated allocation of IP addresses when more than one uninitalised node requests the certification service at the same time, using the same IP address.

If there are not enough neighbouring nodes (i.e. at least K nodes) the certification service cannot be conducted exclusively with communications in the 1-hop neighbourhood. In this case, the requesting node can progressively increase the TTL of the messages from the
certification service and use flooding instead of broadcasting to send out requests. However, since the certification service replies are sent to the requesting party in unicast, the requesting party can only receive them if it has a configured IP address and is participating in the routing service, since it is at a distance of more than 1-hop from the granting nodes.

Nodes that have recently joined the Manet will not have access to autoconfiguration and routing services before obtaining a valid certificate. Moreover, the strategy of choosing a random IP address is not sufficient, since a node without a certificate will still be unable to access the routing service. A simple solution is for nodes that forward requests from the requesting node to act as proxies for that node when the replies are transmitted in their neighbourhood. Obviously, this requirement is only applicable to messages for the issue of a new certificate and if the certification policy allows these messages to be sent further than the 1-hop neighbourhood.

4.1.7. Use with Multiple DCAs

When two or more Manets that have been initialised by different processes join together to form a single network, there is a distributed certification authority – with its respective private certification key – for each of them. Thus certificates of nodes originating in different Manets cannot verify each other since they are signed with different private keys. This means the nodes in the new network have different certificates from different DCAs. A simplistic solution to this problem consists of requiring that all nodes from one of these networks acquire a certificate issued by the other’s DCA. However, this approach implies an excessive communications overhead since all nodes wishing to receive the new certificate must make a certification request. An alternative solution is to establish a new relationship of trust between the networks that have joined together rather than between individual nodes. To this end we propose the establishment of a cross trust network between the distributed certification authorities.

Without loss of generality, the following shows the formation of a cross trust relationship between nodes from two different DCAs (e.g. DCA₁ and DCA₂). The numbers of nodes in the dynamic coalition of DCA₁ and DCA₂ are given as \( K₁ \) and \( K₂ \) respectively. One of the nodes from DCA₁ must ask the nodes from DCA₂, to sign the certificate of DCA₁. At least \( K₂ \) nodes from DCA₂ must agree to trust the network of DCA₁, forming a dynamic coalition to sign the requested DCA certificate. The node responsible for requesting the cross trust certificate receives the partial certificates of DCA₁, signed with parts of the private key of DCA₂, and reconstitutes the new certificate of DCA₁, signed by DCA₂. Finally, the
requesting party must circulate the new certificate and the counter-certificates contained in its local CRL throughout the network, thereby completing the formation of cross trust. In the same way, $K_1$ nodes from DCA$_1$ must sign a certificate for DCA$_2$ using the private certification key of DCA$_1$.

Note that these cross trust certificates have different properties and objectives to certificates signed for individual nodes, since the trust established is reflected among all nodes from the same DCA. Thus the creation of this relationship must also be controlled by specific certification policies.

4.2. MANET AUTHENTICATION EXTENSION (MAE)

Manet Authentication Extension (MAE), discussed in this section, consists of a protocol for the authentication of message-based applications, where communications do not follow a classic client-server model. This applies to routing and autoconfiguration protocols for Manets in particular. MAE is also used to authenticate messages from security services, such as L-Certs and LIDS.

MAE is attached to the messages of authenticated protocols. This MAE must contain all information necessary to ensure the correct authentication and integration of these messages, guaranteeing protection against fabrication, modification and impersonation attacks. The objective is to design the MAE in a flexible and adaptable way so it can be used in securing different Manet protocols and applications. In particular we discuss its use in securing the AODV, OLSR, TBRPF and DSR routing protocols and the DCDP autoconfiguration protocol. With regard to routing protocols in particular, which are already in a mature phase of standardisation, the idea is to allow MAE to be used without the need to alter the messages of the protocol that are to be protected, i.e. preserving the syntax and sequence of the messages without modifications. This approach differs from other related works [e.g.28,41].

Authentication Objects: The design of the MAE follows the modern trend of creating extensive protocols by the successive addition of objects with specific purposes in messages, as occurs in IPv6, among others. Thus MAE is made up of a set of authentication objects, which provide the authentication services adapted to the needs of the different protocols that are to be protected. Since the basic aim is authentication, a MAE always contains a mandatory authentication object, which carries either a digital signature (DS$^{26}$) or a message

---

26 Digital signature.
The choice of the type of mandatory authentication object depends on the authentication policy chosen for the Manet.

Digital signatures are used together with digital certificates to provide authentication using asymmetric cryptography. This authentication system is completely integrated with the model of trust and the certification services proposed in this work. A digital signature is computed simply as a function of the hash of the protected message with the private key from the message originator. In addition to authenticating the origin and controlling the integrity of messages, this alternative provides the non-repudiation service – fundamental for corrective protection of the proposed security model.

On the other hand, the use of MAC makes it possible to apply the MAE in environments where authentication methods are used with shared cryptographic keys and symmetric cryptographic primitives. A MAC is calculated as the hash of the protected message connected with the secret authentication key. The simplest case consists of the use of a single secret key (group key), shared by all nodes in the Manet. Note that this system is not resistant to intruders since the secret used by all nodes is exposed if only one node is compromised. In the same way, even if it is possible to detect attacks in this scenario, since there is no non-repudiation, it is not viable to remove all compromised nodes from the system by revoking the cryptographic key. However, the advantages offered by the use of a symmetric cryptographic key may compensate for the use of such an authentication system, as is the case in the studies in [48,49]. Finally, there are alternatives to the distribution of cryptographic keys that have been specially adapted for the Manet context, such as the use of the TESLA protocol and the use of information on the mobility of the nodes for the derivation of the authentication keys, which in these scenarios makes the use of symmetric cryptography even more promising. However, these mechanisms do not allow the non-repudiation of messages, making it harder to eliminate compromised nodes from the system. For this reason, this authentication system can only be applied in scenarios which do not require the proactive elimination of compromised nodes.

The mandatory certification object (i.e. DS or MAC) is used to authenticate all non-mutable fields of the message, including the fields of the MAE itself. As a rule, mutable data in messages is discounted when calculating the mandatory authentication object.

Other certification objects are used to provide additional services. The options that are currently defined are[93]:

27 Message authentication code.
Certificates (CERT): Certificate objects are used to carry, together with a DS object, the certificate of the message signatory, with the option of carrying the other certificates necessary to establish the certification path for this certificate.

Hash Chains (HC): A common challenge in authenticating routing protocols is found in the existence of mutable fields in messages from protocols, i.e. fields that are altered by intermediate nodes between origin and destination [28,41]. This is the case with DSR and AODV protocols, which have messages that are altered progressively as they are forwarded by intermediate nodes [88,58]. Some methods to protect the typical mutable fields of these protocols (e.g. hop count and IP address traces) using hash chains are presented in [50]. This strategy is used in this work, through the inclusion of authentication objects such as HASH_CHAINS in the MAE that authenticate messages containing these kinds of mutable fields.

Anti-Repetition Protection (SEQNUM): A typical attack in environments where messages are authenticated consists of repeating messages with valid authentication after the time they are legitimately sent (reply attack). Most protocols that spread their messages beyond the 1-hop neighbourhood possess protection against repeated message processing. However, in general this protection consists of a sequence number that is increased consecutively and which is vulnerable to certain types of reply attack [84]. MAE supports optional protection against this type of attack based on random sequence numbers that are authenticated together with the non.mutable fields and included in the authentication information as SEQNUM objects.
4.3. ROUTING PROTOCOL AUTHENTICATION

4.3.1. Routing Protocol Vulnerabilities

Attacks against routing protocols are usually related to the insertion of incorrect routing information, in the attempt to disturb the protocol’s normal operations. This occurs with modification attacks (incorrect alteration of protocol routing message during forwarding), fabrication (generation of false routing messages) and impersonation (seizure of identity of another network entity). It is equally possible to combine these basic operations, providing a broader range of attacks. The vulnerabilities described in this section are attacks against routing protocols for Manets that are based on fabrication, modification and/or impersonation of protocol messages.

4.3.1.1. OLSR Protocol Vulnerabilities

As stated above, the OLSR protocol possesses two fundamental types of messages: HELLO and TC. While HELLO messages are restricted to the 1-hop neighbourhood (i.e. cannot be forwarded by nodes), TC messages are spread throughout the network. Thus modification attacks do not apply to HELLO messages, but only to TC messages. Moreover, TC messages do not possess mutable fields, i.e. the message is ready when it leaves the originating node and passes through the entire network without modifications. All modification attacks are therefore also impersonation attacks, since the alterations are interpreted as information supplied by the originating node. Fabrication attacks are possible with both HELLO and TC messages, and can be combined with impersonation attacks. The vulnerabilities shown here consider the original specification of the OLSR protocol [25], without protection from a MAE. These attacks can be mitigated easily with the use of a suitable MAE. Table 4-1 shows the attacks described in this section.
Table 4-1 – Vulnerabilities of the OLSR Protocol

<table>
<thead>
<tr>
<th>Attack</th>
<th>OLSR Message</th>
<th>Fabricated Routing Information</th>
<th>Information of Origin in the Corrupted Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fabrication</td>
<td>HELLO</td>
<td>Neighbour List</td>
<td>Any*</td>
</tr>
<tr>
<td>Fabrication + Impersonation</td>
<td>HELLO</td>
<td>Link-status</td>
<td>IP Address of the impersonated node</td>
</tr>
<tr>
<td>Fabrication</td>
<td>TC</td>
<td>MS list</td>
<td>Any*</td>
</tr>
<tr>
<td>Modification + Impersonation</td>
<td>TC</td>
<td>Sequence Number</td>
<td>IP Address of the Originator</td>
</tr>
</tbody>
</table>

*In general the adversary will try to direct network traffic to itself, using its IP address in the identification of the message originator. Alternatively, the attacker may direct traffic to a non-existent node, in order to provoke denial-of-service. In this case, it uses an unavailable IP address as the message originator. Finally, the adversary can impersonate another node to force all traffic to be directed to it.

**Fabrication of HELLO Messages:** In this attack, shown in Figure 4-2, the adversary fabricates a HELLO message by advertising all the nodes previously divulged in any HELLO message it has already received, together with an additional unused address, with symmetric link status. On receiving this message, all of the attacker’s neighbours choose it as sole MPR. Thus all traffic moving towards nodes that are not direct neighbours of one of these nodes is then forwarded to the attacker.

Before the attack: A (target) has B (target) as MPR, which forwards traffic from A to C. C is at a distance of 02 hops from A and from the attacker.

1 – HELLO Message from B: attacker identifies A and C as neighbours of B.
2 – HELLO Message from A: attacker identifies B as a neighbour of A.
3 – After receiving a new HELLO message from B, the attacker fabricates a HELLO message announcing A, B, C and X (an additional unavailable address) with symmetric link status. This means that A and B choose the attacker as MPR, preventing any traffic from A to C.

Figure 4-2 – Fabrication of HELLO messages
Fabrication + Impersonation of HELLO Messages: Figure 4-3 shows this attack, where an attacker generates a spoof HELLO message after receiving a legitimate HELLO message, advertising all nodes included in the correct message with “lost” link status. The node being impersonated is the same node that originated the correct HELLO message. When its neighbours receive the spoofed HELLO message, all the nodes that had their links erroneously announced with the “lost” status change the status of their link with the spoofed node to “heard”. Thus no traffic is forwarded to the node impersonated by these links.

Before the attack: A (target) and B (target) established a symmetric link between each other by means of previous HELLO messages.
1 – HELLO Message from A: Attacker identifies B as a neighbour of A.
2 – After receiving the HELLO message from A, the attacker fabricates a HELLO message impersonating A, advertising B with lost link status. This message makes B alter its link status with A to asymmetric, thereby avoiding any traffic being forwarded via this link.

Fabrication of TC Messages: In this attack (Figure 4-4), the attacker fabricates a TC message advertising faraway nodes (2 hops or more) as part of its MS set. This means that its neighbours choose to route traffic to these advertised nodes via the attacker.

Before the attack: A (target) has a route to D, at a cost of 03 hops, via B (target).
1 – TC Message from C: attacker identifies D at a distance of 03 hops.
2 – After receiving the TC message, the attacker fabricates another TC message, advertising D as part of its MS (direct neighbours). This makes A stop routing traffic to D via B and start routing it via the attacker.
Modification + Impersonation of TC Messages: In this attack (Figure 4-5), the attacker alters the message sequence number field of a TC message before sending it on. The message sequence number field is increased by an integer. With this attack, the processing and forwarding of TC messages from the node advertised as the originator of the modified message is interrupted throughout the network.

1 – TC Message from A (target): message sequence number = x
2 – Attacker forwards the TC message with the message sequence number field modified to x+i (an arbitrary integer). This paralyses the processing and forwarding of TC messages from A to C and D since these messages have a message sequence number field < x+i.

![Diagram of Modification + Impersonation of TC Messages](image)

4.3.1.2. Vulnerabilities of the TBRPF Protocol

The TBRPF protocol is significantly different from the other Manet routing protocols since messages are only sent to neighbours. As a result no messages are forwarded, which makes it impossible to apply modification attacks.

With TBRPF it is possible to fabricate, with or without impersonation, HELLO messages or topology update messages. In the case of the fabrication of HELLO messages, the effect is similar to fabrication and fabrication + impersonation attacks for OLSR messages: alterations are made to the one hop and two hop neighbour sets learned by participating nodes or the link statuses learned between two neighbours. In the fabrication of topology update (TU) messages, an attacker can alter the calculation of the sub-trees of the topology kept by the neighbouring nodes. These changes will be passed on in TU advertisements of neighbours, spreading the false routing information throughout the network.

4.3.1.3. Vulnerabilities of AODV and DSR Protocols

AODV and DSR protocols are reactive routing protocols, meaning information on routes is obtained on demand, by means of explicit requests. The two principal message types

---

28 The fight back against this type of attack is not defined for OLSR, in spite of the fact this is a common practice in other link-state routing protocols (e.g. OSPF).
for this type of routing protocol are: route request (RREQ) and route reply (RREP), indicating respectively the request for a route and the reply to this request. These protocols also use route error (RERR) and route reply acknowledgement (RREP-ACK) messages as auxiliary messages to notify routing errors (e.g. unavailability of routes) and to confirm receipt of RREP replies, respectively. The basic difference between AODV and DSR lies in the fact that the first uses a distance vector routing algorithm, while the second uses routing at source. This means the content of RREP messages is different in each case. For AODV, there are only the originating and destination address, as well as a hop count which is altered each time the message is forwarded. In DSR, however, the IP addresses of all intermediate nodes are added to the message as the network nodes forward it, from the origin until the destination. There now follows a summary of the principal vulnerabilities of the AODV and DSR protocols involving message fabrication, modification and impersonation [28].

**Modification Attacks**

- Modification of the destination sequence number of RREP messages (AODV protocol): The routing sequence numbers are increased consecutively to prevent duplicated/old messages (routes) from being processed. An attacker can modify this number, increasing it by a large integer, which will mean the most recent updates will be considered older. Thus real messages will no longer be processed. This is similar to the modification attack for sequence numbers in TC messages (OLSR).
- Modification of the hop count (AODV protocol): by modifying (increasing or decreasing) the hop count of RREQ or RREP messages, this alters the calculation of the routing table of all nodes that use this message.
- Modification of routes in RREQ and RREP messages (DSR protocol): by excluding, including or modifying addresses in the list of addresses that make up the routes between the origin and destination, this alters the path that will be executed by the message in the routing back to the source.

**Impersonation + Fabrication Attacks**

- Fabrication of RREP messages, impersonation of network nodes: An attacker can listen to RREP messages exchanged between nodes and fabricate RREP messages altering the routes between two nodes.
- Fabrication of RERR Messages: RERR messages can be fabricated, impersonating network nodes that lie on the path that is to be broken.
Fabrication Attacks

- Fabrication of RREP messages to “poison” the route cache (DSR protocol): since RREQ and RREP messages contain the list of IP addresses in a route between two nodes, an optimisation to avoid requests for routes consists of promiscuously listening to messages exchanged on the network and storing the advertised routes in a cache of routes that is consulted before sending route requests. Thus by fabricating RREP and RREQ messages by announcing false routes, these routes can be learned by nodes listening promiscuously, and that will subsequently execute the routing contained in the false messages.

4.3.2. MAE for Routing Protocols

4.3.2.1. MAE for the OLSR Protocol

The OLSR protocol allows multiple messages to be grouped in a single OLSR packet for transmission in each hop. Normally, some OLSR messages within the package are only processed locally, while others are also forwarded to other nodes as part of the flooding mechanism. In addition, not all messages in a packet come from the same originator. This means protection by the MAE is defined to be used at the level of messages and not at the level of packages. In other words, every OLSR message must be authenticated by a MAE. Since the package fields are not effectively used by the routing algorithm and are only used for the encapsulation of multiple messages in 1-hop transmissions, these fields do not need to be protected by a MAE.

None of the OLSR messages (e.g. HELLO, TC, MID, HNA and FRR) have any mutable fields. However, the generic header of the messages possesses hop count and time to live fields that are mutable. HELLO and FRR messages are only broadcast to the 1-hop neighbourhood, while TC, MID and HNA messages are flooded across the entire network. Given that these mutable fields are not used in calculating the routing table, but only by the flooding algorithm (which is already robust in itself due to the natural redundancies of the network topology), no additional protection is required for the authentication of these fields. In this way the MAE for the authentication of the OLSR simply consists of a single authentication object containing a digital signature or a message authentication code. All message fields are authenticated, except the hop count and time to live fields, which must be set to zero for the calculation of DS or MAC. A certification object can be added to each
message if the authentication system is DS. In this case, a basic implementation of a distribution mechanism of proactive certificates can be defined, requiring that the signatory’s certificate be included in HELLO and TC messages. A further optimisation can be made if the certificate is only included in HELLO messages when a new neighbour is detected.

4.3.2.2. MAE for the TBRPF Protocol

TBRPF is a proactive routing protocol that uses a link-state algorithm. This protocol possesses no mutable fields in its messages that are used by the routing algorithm. In particular, the protocol’s messages are only forwarded among neighbours. The MAE to protect the TBRPF protocol is simply constructed using a single authentication object containing a digital signature or a message authentication code. An efficient mechanism for the proactive distribution of certificates consists of requiring that the signatories’ certificates are only included in HELLO messages.

4.3.2.3. MAE for the AODV Protocol

The AODV has mutable fields in route request (RREQ) and route reply (RREP) messages. These fields contain hop count routing metrics that are altered each time the message is processed and forwarded by the nodes between the message source and destination. A hash chain (HC) object [41] is included and updated each time these fields are altered (i.e. each time the message is forwarded). This protection prevents attackers from decreasing the hop count. Route error messages (RERR) are only authenticated by the node that forwards them. Route reply acknowledgement messages (RREP-ACK) do not possess mutable fields and are only authenticated by the sender. Certificates can be distributed efficiently, on demand, in RREQ and RREP messages.

4.3.2.4. MAE for the DSR Protocol

The security of DSR is significantly more complex, and a complete solution requires all intermediate nodes that forward messages to authenticate them. Limited security can be obtained by combining the mandatory authentication object with an HC object, implementing a hop-by-hop summary of the trace of IP addresses in RREQ messages. This prevents an attacker from falsifying an initiating node or removing correct IP addresses from the routing list [49]. RREP messages could simply be authenticated by the destination of the route
discovery (i.e. the node generating the RREP message). In a similar way to AODV, certificates can be distributed efficiently on demand in RREQ and RREP messages.

### 4.3.2.5. Evaluation of MAE Protection

Table 4-2 shows the principal characteristics of the routing protocols for Manet and the MAE requirements for each of them.

<table>
<thead>
<tr>
<th>Table 4-2: MAE of Manet Routing Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Routing Protocol</strong></td>
</tr>
<tr>
<td>DSR</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>AODV</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>OLSR</td>
</tr>
<tr>
<td>TBRPF</td>
</tr>
</tbody>
</table>
4.4. AUTHENTICATION OF AUTOCONFIGURATION PROTOCOL

Unlike routing protocols, the standardisation process for autoconfiguration solutions in ad hoc networks is still in the initial stages. This makes it difficult to analyse the vulnerabilities of the current alternatives since they are only preliminary proposals aimed at showing key concepts and the applicability of autoconfiguration methods. Thus these proposals do not show sufficient specification detail and maturity for them to be treated as serious candidates for a standard protocol in their current state. However, F. Buiati [14] has carried out important work in specifying and implementing the DCDP protocol. In the recent work [13], we proposed security mechanisms that can be applied to this protocol using the same concepts of the trust model via L-Cert and authentication with MAE presented in this work. We have therefore restricted our discussion of the vulnerabilities and protection of the autoconfiguration protocol to the scope of the DCDP protocol. However, since the different autoconfiguration protocols have similar scaling to the DCDP protocol (i.e. basically the same types of messages are used – e.g. service request and service response) we believe that this analysis can easily be extended to other protocols.

4.4.1. Vulnerabilities of the DCDP Protocol

Autoconfiguration protocols have two basic types of mechanism: the autoconfiguration process itself and the mechanisms for maintaining and updating the autoconfiguration databases (synchronisation). We have focussed our analysis of vulnerabilities on the autoconfiguration process, since the synchronisation mechanisms are still in early stages of development.

The autoconfiguration process involves communications between a requester (client) and a server, which meets these requests. We have classified fabrication attacks against the autoconfiguration protocol as request attacks, where the attacker adopts the role of the client, and server attacks, where the attacker responds maliciously to the requests from network clients.

Communications of the DCDP autoconfiguration process evolve essentially in the 1-hop neighbourhood. Thus attacks involving the modification of protocol messages do not apply in this stage (although such attacks may make sense in the synchronisation phase). With regard to client attacks, these can range from simple fabrication attacks (e.g. a node requests a block of IP addresses and thereby makes the block allocated unavailable for allocation) to
combined fabrication and impersonation attacks (e.g. a node requests the liberation of a block of addresses that are still in use, thereby potentially causing duplicated allocations). On the other hand, in the case of server attacks, impersonation attacks must be accompanied by a denial-of-service attack on the impersonated node. Otherwise, even if the impersonated messages are delivered, the correct messages will also be delivered, which limits the efficiency of this kind of attack. Thus, without loss of generality, in this case we have basically considered message fabrication attacks. These messages may result from a combined process of fabrication and impersonation, although without producing more significant effects than those resulting from simple fabrication attacks.

**Request attacks:** In this type of attack, attackers act as clients and fabricate messages requesting the autoconfiguration service. For example, an attacker may request the allocation of an IP address, thereby making the assigned block of IP addresses unavailable to other nodes (fabrication attack). Another example is the possibility for an attacker to request the liberation of a block of IP addresses, thereby making it possible for these IP addresses to be used to meet other requests, even if the IP addresses are still being used by network nodes (fabrication + impersonation attack).

**Server attacks:** In this type of attack, attackers act as network servers and meet requests by responding with false messages. For example, an attacker can respond to an address request message and provide IP addresses that are already being used by other nodes in the ad hoc network, resulting in a conflict of addresses. Alternatively, the attacker can respond to a request for the liberation of an IP address by a node wishing to leave the network, confirming the node’s departure but not liberating the IP address to meet new requests.

### 4.4.2. MAE for the DCDP Protocol

The MAE for the DCDP protocol is essentially made up of a mandatory authentication object and a CERT object, if the authentication system is DS. The distribution of certificates can be handled by the routing protocol, if a proactive protocol is being used. Otherwise, CERT objects must also be included in messages for synchronising the autoconfiguration databases, since they operate proactively. If two networks are joined that were initialised with different DCAs, the cross trust relationship must be established prior to the detection and resolution of address conflicts. Certificates should be included in autoconfiguration protocol messages in such cases, since routing between nodes from different networks will not be effective until the process for resolving address conflicts has been completed.
4.5. SECURITY POLICY AND CONFIGURATIONS FOR CERTIFICATION AND AUTHENTICATION SERVICES

Manets can be used in different contexts, each with different security requirements. For example, an open network for sharing data in a classroom, in which users can enter or leave freely, has different security requirements from a Manet set up for a recovery mission in an area affected by a natural disaster or a military Manet on a battlefield. While affiliation to the network is not pre-determined in the first case, in the other two cases the nodes may have been initialised in advance (with certificates and parts of the private certification key) to provide identification and access control suited to these scenarios.

A set of configuration options means that the certification and authentication service provided in our security solution can be quickly adapted to suit different security policies. The objective is to make it possible for security requirements to be mapped in specific configurations, determining the behaviour of the different security mechanisms. Thus in the “open network” scenario, as exemplified by the classroom, the authentication system can be configured simply for DS, and the certification policy for new nodes can be “grant provided requesting parties identify themselves”, for example by showing their student card to nearby users that are their neighbours on the network. Certification restrictions can be related to previous inappropriate behaviour. In recovery scenarios, a well-defined group of nodes is cooperating on the Manet and the probability of an attacker intentionally interfering with the network is low. Equally, users cooperating on the network know and trust each another, obviating the need for a mechanism to identify compromised nodes. Thus all nodes can be initialised with a shared secret key and the authentication system configured for MAC. New nodes are not admitted to the network, except those already configured with the “group key”. Finally, the military Manet scenario involves a network with a high probability of nodes being compromised, including by means of physical capture. Access to the network must be tightly controlled, and the certification policy must be “always reject silently”. However, certificates can be renewed for nodes that have no history of misbehaviour (possibly compromised) and which can prove their identity. Table 4-3 shows these security policy concepts applied to the configuration parameters of the solution, for the scenarios mentioned.
## Table 4-3 – Types of Authentication Policy

<table>
<thead>
<tr>
<th>Type of Policy</th>
<th>Example Scenario</th>
<th>Authentication System</th>
<th>Certification Policy</th>
<th>Certificate Renewal Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>open network</td>
<td>classroom</td>
<td>DS</td>
<td>dependent on verification of identity</td>
<td>always grant</td>
</tr>
<tr>
<td>managed network</td>
<td>recovery</td>
<td>MAC with group key</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>network managed in</td>
<td>battlefield</td>
<td>DS</td>
<td>reject silently</td>
<td>dependent on verification of identity</td>
</tr>
<tr>
<td>hostile environment</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4-6 shows the configuration parameters for the authentication and certification policy considered in our work [93].
Figure 4-6 - Parameters for the Authentication and Certification Policy
5. INTRUSION DETECTION AND RESPONSE IN MANET

The intrusion detection system is fundamental in the security model proposed in this work, providing proactive monitoring of the Manet security status. This monitoring is aimed at identifying violations of the security policy (attacks). Moreover, by means of an equally proactive response to the detection of attacks, the IDS interacts with other security services (e.g. authentication, certification, access control) to eliminate the causes of an attack (e.g. revoking the certificate of nodes that violate the security policy) or mitigating its effects (e.g. reconfiguring packet filtering to avoid the forwarding of packets that violate the security policy).

This chapter presents the specification of an intrusion detection system designed for Manet. It sets out a new architecture for IDS, resulting from the specific requirements of the Manet environment. This means the proposed IDS has distribution, self-organisation and localisation characteristics. Use is made of the characteristics of mobility and autonomy associated with the technology of mobile agents [3,85], which are incorporated in the IDS to provide an efficient and flexible solution for the constraints of bandwidth and connectivity on Manets. In addition, the design of the IDS considers a complete distribution of the intrusion detection process, instead of restricting distribution to data collection tasks, which is the most common approach used by the majority of IDS projects based on mobile agents. Finally, the modularisation of the architecture of the distributed IDS is underlined, allowing the immediate extension of the system’s capacities in terms of the derivation and incorporation of new IDS modules.

5.1. COMPLETELY DISTRIBUTED IDS

The intrusion detection process consists of the collection and analysis of network audit data from the system or from applications. Intrusions must be reported to security management as soon as they are detected. Similarly, an automatic response may be triggered upon detection, aimed at eliminating the causes or effects of the intrusion. Given the lack of centralisation, the mobility of the nodes and the wireless nature of link connections in the Manet environment, some (if not all) tasks required for the intrusion detection process described above must be executed in a distributed and cooperative way [117]. In our design, a local IDS (LIDS) is placed on each node of the Manet. The LIDSs communicate with each
other using a mechanism that takes into account the restrictions of the Manet context, i.e. limited bandwidth or poor connectivity. Such architecture was initially defined in [117] as a basic requirement for IDS in Manet environments. We have proposed the use of mobile agents to provide flexible and autonomous means of interaction between the LIDSs [3].

If an LIDS fails to cooperate during a given period of time (e.g. the node has moved away, failed or is compromised), the intrusion detection service must not degrade. The inherent redundancies in Manets compensate for the nodes that are not cooperating in the detection process, since it is possible for more than one node to monitor and detect the same attack.

Mobile agents are an alternative to the client-server distribution model [55]. The use of mobile agents, as opposed to traditional approaches where data is transported towards the computation points, allows the code to move towards the data. Carefully designed agents can reduce the quantity of data exchanged through the network while providing a flexible means of distribution. In addition, a node dispatching an agent does not need to wait for it to return to resume normal processing, since the agents can be dispatched or even destroyed by other nodes without having to return to the originator node.

One LIDS cooperates with others by dispatching mobile agents to other nodes and by processing the tasks embedded in the incoming agents. In our design, cooperation can take place in the different stages of intrusion detection. During data collection, nodes exchange information on events to construct evidence of intrusions. In executing the detection algorithm, the LIDSs exchange information on the current state of the structures that implement the detection algorithm. Finally, alert correlation is also possible when a node uses alerts from others to reinforce evidence of suspicious activity that has been detected locally. In any case, cooperation is executed about by means of mobile agents that roam from one system to another. Mobile agents can also provide a first element of response to the dynamic nature of Manet topology and membership. Indeed, when a node joins the network, it does so with a running LIDS containing a mobile agent platform. It can therefore take part immediately in the collaborative intrusion detection process. In the same way, when a node leaves the network, other neighbouring nodes can collaboratively provide the information required for the execution of the execution process.

Maintaining collaboration within a restricted number of nodes relates to bandwidth usage and scalability requirements. The reason for executing certain detection tasks exclusively in the local neighbourhood is doubly justified by the nature of Manets. A node on
these networks must always collect and maintain information about its neighbours, independent of the routing protocol being used. Moreover, any information going to or coming from a node must be routed through one of its neighbours, at which point its forwarding can be promiscuously monitored due to the broadcast nature of wireless links [119]. Thus neighbours of the node that is being attacked are naturally eligible as primary sources of information about the status of the node suffering the effects of an attack. Neighbouring nodes are also eligible collaborating peers to collaborate with the aim of discovering new information related to the ongoing intrusion that is unavailable locally.

5.2. LIDS MODULAR ARCHITECTURE

In this section we present the proposed modular architecture of the LIDS, as shown in Figure 5-1. This architecture consists of an adaptation of the intrusion detection framework proposed by the work group on intrusion detection of IETF [115], made up of sensors, analysers and managers. The adaptations aim to extend this framework to meet the requirements of the Manet context.

In each LIDS the sensor module collects data from audit data sources and synthesises it, the analyser module processes the synthesised data to detect situations that could constitute violations of the security policy, while the manager module carries out the management interface for the whole process, in addition to executing tasks for alert correlation and automatic intrusion response activation. The limitations of the Manet environment are taken into consideration with the distribution manager and agent platform modules.
5.2.1. Intrusion Detection Framework

The audit data used by the LIDS can be collected from different sources, such as a network packet capture interface (network level), a log system (system level) or even a MIB (network, system or application level). This data is usually raw and has poor semantics. In addition, the data is available in a format that depends on the data source. Thus some preprocessing is applied to the data, transforming it into semantically richer messages used by the detection algorithm, which are called events. This transformation of raw data is referred to as event abstraction. Event abstraction can use many different techniques, such as pattern matching [53], data mining [70] or statistical correlation [117].

In this project, the sensor module is decomposed into two modules: Event Extractor and Data Collector. These modules separate the tasks of data retrieval and data abstraction in two different entities. The idea is to enable multiple implementation for the data collector module, which can operate simultaneously collecting data from different audit sources. This therefore permits the event abstraction process to have abstraction rules that use raw information from multiple sources. Similarly, the event extractor is capable of multiple implementations, allowing the simultaneous use of different abstraction techniques.

The analyser processes events according to a pre-defined detection strategy. At least two intrusion detection methodologies are currently being discussed: detection for misuse and
for anomalies [29]. In general, these methodologies can be said to be complementary. It is our objective to have a hybrid intrusion detection strategy, combining techniques for misuse and anomaly detection. Thus each implementation of a specific detection algorithm in LIDS architecture is encapsulated in an IDS nucleus module. It is also possible to have multiple instances of this module, each implementing a specific detection algorithm.

A concise representation of the current status of the detection algorithm is represented in a detection status message. Such information can relate to the detection status of a specific attack, in misuse detection or to the status of the behaviour model, in anomaly detection. The analyser can also carry out a query of a local or remote sensor, to establish the occurrence of specific events in a detection path that is being examined on demand. Upon detection of any type of activity considered intrusive by the detection algorithm, an alert is sent to the manager module.

The manager module is divided into two modules: alert manager and communication framework. While the alert manager interprets alerts, eliminating false positives and false negatives, correlating alerts and intrusion response, the communication framework provides a communication interface between the IDS and other security services, including those for interaction with other intrusion detection systems. This communication can be conducted using standardised IDS message formats, such as IDMEF [69].

5.2.2. Manet Environment Constraints

In most distributed IDS architectures, distribution is restricted to data collection. This is mandatory in IDS for ad hoc networks since the remote collection of significant volumes of raw data is prohibitive due to limited wireless bandwidth. In addition to local data collection, the IDS proposed in this work permits a complete distribution of IDS tasks, making it possible for both the execution of the detection algorithm and the management of alerts to be distributed equally. Thus data collection in this IDS is always kept local. Data exchanged between the LIDSs consists exclusively of concise information (e.g. events) resulting from local pre-processing of raw data. Cooperation in executing the detection algorithm is carried out by exchanging detection status messages, while collaboration in alert management is carried out by exchanging alert distribution.

In any case, distribution and cooperation are conducted through agents that generate high level IDS messages (e.g. events, queries, detection status and alerts). The distribution manager module manages the distribution of these messages to local and remote modules. Agents are in turn dispatched and managed in the agent platform module.
5.2.3. Messages generated by the LIDS

Messages generated by the LIDS can be specified in terms of the following clauses:

- Messages refer to a set of system entities, possibly empty, which is formed by attacker↔target pairs. An attacker consists of a set, possibly empty, of identifiers of network entities which it is deemed may be responsible for an attack. Similarly, a target consists of a set, possibly empty, of identifiers of network entities that it is deemed may be affected by the attack.

- Event, query, alert and detection status messages possess the following attributes: identifier, originating entity identifier and a set, possibly empty, of attacker↔target pairs.

- A periodic query message is a special type of query message, which is automatically generated on a periodic basis. This message possesses an additional attribute: period, indicating the intervals between the generation of such messages.

- Alert and detection status messages possess an additional attribute: attack identifier, identifying what attack is being monitored at the time the messages are generated.

- A detection status message possesses an additional Boolean attribute: isClone, indicating if the message’s attacker↔target pairs should be eliminated in the originating IDS nucleus (isClone = false) or if they should be cloned in the message (isClone = true) and kept active in the originating LIDS.

- Event, query, periodic query, alert and detection status messages possess a further Boolean attribute – isLocal, indicating whether the message should be consumed locally (isLocal = true) or whether it should be dispatched to remote LIDSs (isLocal = false). In the first case, event and detection status messages are consumed by the analysing module; query and periodic query messages are consumed by the sensor module; and alert messages are consumed by the management module. Alternatively (isLocal = false), messages must be dispatched to remote LIDSs using the distribution manager module.

- In order to allow messages to be dispatched by the distribution manager module (isLocal = false), they must also possess: a set, possibly empty, of destination entity identifiers, a Boolean attribute isFlooded and an entire positive TTL attribute. The message must always be dispatched to LIDSs for which the identifiers are in the set of identifiers of the destination entities. If isFlooded =
false, the set of identifiers of the destination entities cannot be zero. Otherwise, the message will be flooded across the network in a radius measured in terms of the TTL. If TTL = 1, the message will be spread within the 1-hop neighbourhood of the originating message. If TTL = 2, 1-hop and 2-hop neighbours will receive the message, and so on.

5.3. MISUSE INTRUSION DETECTION

For misuse intrusion detection, a pattern matching mechanism is used whereby the signatures of the attacks are described as finite state diagrams (FSD) with a timeout, similar to the scheme proposed by [53]. In this scheme, the IDS nucleus model is composed of a finite state machine, which processes received events to identify transitions between the FSD states that define the signatures of monitored attacks.

An FSD possesses an initial state, a final state and intermediate states. A set of attacker↔target pairs is maintained in each state of the FSD, except for initial and final states. Therefore every time an event triggers a transition from the initial state to another state in the FSD, it receives the attacker↔target pair(s) contained in the event. This makes it possible for the same attack to be tracked simultaneously from different sources. In the same way, an attacker↔target pair is removed from the FSD whenever a transition is triggered to the final state, containing this attacker↔target pair.

A time limit may be imposed on an attacker↔target pair remaining in a non-initial state, triggering a transition to another state or auto-transitions, if no transitions are triggered by events for this attacker↔target pair during a per-determined time period. Thus a transition is always triggered for an attacker↔target pair, whether this is due to the occurrence of an event or the exceeding of a time limit.

Events and queries can be originated locally or remotely, allowing collaboration in data collection. With regard to collaboration in executing the detection algorithm, detection status messages correspond to information on the current state of an FSD, which represents an attack signature. Thus, by distributing this information, a node can ensure the detection process continues in another node, starting from the last locally-detected transition. In each transition in one of the FSDs, one of the following actions can be executed: a query, the distribution of an execution status or the generation of an alert, when the transitions indicate a positive recognition of an attack pattern. These messages can be consumed locally or dispatched in agents to other nodes.

99
The IDS Kernel for this type of detection can be specified in terms of the following clauses:

- An **attack signature** must be specified in terms of an FSD with a timeout. The nucleus module of the IDS must keep a set of FSDs.
- An **FSD** is defined as an automaton formed of an initial state, a final state, a finite set of intermediate states and a finite set of transitions, forming an oriented graph where the states are the nodes and the transitions are the links. An FSD contains: an attack identifier, a set of states and a set of transitions.
- A **state** is associated with a specific situation or status (possibly abstract) assumed by the monitored system at a specific moment in time, referring to a given set, possibly empty, of system entities formed of attacker↔target pairs. Initial and final states always have an empty set of attacker↔target pairs.
- A **transition** consists of an alteration of one state to another, being defined in terms of the original state (current), the destination state (new state), of one (single) factor that triggers the transition and a set, possibly empty, of actions that must be executed. A transition that has the same original and destination state is known as an auto-transition. The factor that triggers a transition can be an event or a timeout. The factor that triggers a transition and the transition itself are specified for the same, possibly empty set of attacker↔target pairs.
- Each state possesses a **timeout** that monitors the maximum time \((\text{Time}_{\text{max}})\) that an attacker↔target pair can remain in a given state without undergoing a transition (even if this is an auto-transition). This time is measured from the entry of an attacker↔target pair in the set of system entities of a state. Each time an attacker↔target pair remains in a state for \((\text{Time}_{\text{max}})\), a timeout transition is triggered for this attacker↔target pair. \(\text{Time}_{\text{max}}\) can be equal to 0 (zero), indicating that a timeout transition must be triggered immediately. Alternatively, \(\text{Time}_{\text{max}}\) can take on a negative value, indicating that the timeout never expires.
- Each time a timeout is triggered for an attacker↔target pair, the pair must be removed from the set of attacker↔target pairs for the origin state and added to the set of attacker↔target pairs for the destination state, with the timeout reset, except in the case of transitions to the final state, when the attacker↔target pair must simply be removed.
The execution of an action consists of the generation of a new message which can be: a query, a detection status message or an alert. This message is always generated for the same set of attacker↔target pairs as the transition that triggered the execution of the action. Alert and detection status messages receive the same attack identifier as the FSD’s identifier and the same identifier as the identifier for the destination state of the transition.

- An **event** is consumed by the IDS nucleus module by verifying whether any transition has been triggered by it in all states of all the FSDs.
- A **detection status** message is consumed by the IDS nucleus module by adding all attacker↔target pairs of the message to the set of attacker↔target pairs of the state with the same identifier as the execution status message, in an FSD with the same attack identifier as the message attack identifier.

Examples are then presented of attacks against the routing protocol and against distributed applications that may be detected using an IDS nucleus which uses a misuse detection approach, as defined above. Three examples of complete development of attack signatures are included, showing the possibilities for collaboration in the alert management processes (Fabrication + Impersonation of HELLO message of the OLSR routing protocol Figure 4-3), execution of the detection algorithm (Fabrication of HELLO messages of the OLSR routing protocol Figure 4-2) and data collection (telnet connection chains).

### 5.3.1. Attacks and Attacks Signatures against the Routing Protocol

This section discusses the detection of attacks against the OLSR routing protocol. These attacks were presented in section 4.3.1.1. Figure 5-1 shows the type of attack signature developed for each attack in terms of FSD.
Table 5-1 - Attack Signatures against the OLSR Protocol

<table>
<thead>
<tr>
<th>Attack</th>
<th>OLSR Message</th>
<th>Falsified Routing Information</th>
<th>Corrupted Message Source Information</th>
<th>Attack Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fabrication</td>
<td>HELLO</td>
<td>Neighbour List</td>
<td>Any</td>
<td>Inconsistencies in routing information between different HELLO messages.</td>
</tr>
<tr>
<td>Fabrication + Impersonation</td>
<td>HELLO</td>
<td>Link-status</td>
<td>IP Address of impersonated node</td>
<td>Anomalies in protocol scaling</td>
</tr>
<tr>
<td>Fabrication</td>
<td>TC</td>
<td>MS list</td>
<td>Any</td>
<td>Inconsistencies in routing information between different HELLO messages.</td>
</tr>
<tr>
<td>Modification + Impersonation</td>
<td>TC</td>
<td>Sequence Number</td>
<td>IP Address of originator</td>
<td>Anomalies in protocol scaling</td>
</tr>
</tbody>
</table>

To detect the first three attacks shown in Table 5-1, the information that must be collected (sensor module) is the neighbour list with the respective link-statuses (symmetric, asymmetric), the 2-hop neighbour list with the respective MPRs and the MPR list. These lists are kept by the OLSR routing daemon and must be collected each time there is a change in content (e.g. change in the link-status, addition of a new neighbour, removal of a neighbour, etc.). To detect the fourth type of attack a list can be drawn up of message sequence numbers, also kept by the daemon. An alternative to collecting this information would be to use the experimental MIB SNMP defined for the OLSR [94]29. An alternative would be to use a network capture interface that could capture HELLO and TC messages promiscuously and process them to obtain the desired information. The problem posed by this approach lies in the repetition of processing already carried out by the OLSR routing daemon in the LIDS sensor module.

As a rule, the signatures for each of the attacks shown in Table 5-1 can be identified considering the following:

- **Fabrication of HELLO Messages:** This attack can be identified by verifying inconsistencies between the OLSR neighbour lists of neighbouring nodes. Indeed, any node that can listen to HELLO messages from the attacker and from any other node that is not heard by the attacker is capable of detecting inconsistencies, since the attacker advertises the node it cannot hear as if it were its neighbour, although

---

29 An experimental MIB for the OLSR was proposed by the author in a previous work [94]. However, this MIB is not standardised, since the IETF has not yet begun standardisation work of MIB for ad hoc routing protocols. However, judging by the MIB-II standard for other routing protocols (e.g. OSPF) [75], it is fairly likely that the neighbour list is in the MIB to be standardised for the OLSR or even for all other ad hoc routing protocols.
this is not the case. Candidates for attack detection are the old MPRs of nodes that selected the attacker as a new MPR.

- Fabrication + Impersonation of HELLO Messages: This attack can be identified by scaling anomalies in the routing protocol, which occur due to the appearance of more than one HELLO message in the same HELLO_INTERVAL period, each with the same originator and advertising the link-status of the same neighbour. In this case, the link-status of the neighbour changes between asymmetric and symmetric or MPR within a single HELLO_INTERVAL period.

- Fabrication of TC Messages: This attack can be characterised by the presence of inconsistencies advertised simultaneously by different nodes. Some nodes advertised in the MS of the originator (i.e. the attacker) in the false message are not neighbours of this node. As the fabricated TC message must be flooded throughout the network, this message will eventually reach nodes that do not have the attacker in their neighbourhood. These nodes can identify the attack.

- Modification of TC Messages: This attack can be identified by scaling anomalies in routing protocol messages, given that both the correct message and the modified message are sent out in the neighbourhood of the node that originated the correct message in a single TC_INTERVAL period. Thus both the real message originator and its neighbours can detect the attack.

The complete specification process of the attack signatures, in terms of event abstraction and FSD definition, is shown below for the first two attacks of Table 5-1.

5.3.1.1. Fabrication + Impersonation of HELLO Messages:

This attack is shown in Figure 4-3 in the previous chapter, where an attacker fabricates an impersonated HELLO message after receiving a legitimate HELLO message, advertising all nodes that were included in the correct message as having a lost link status. The node being impersonated is the same node that originated the correct HELLO message. This attack is similar to the attacks identified above against the OSPF routing protocol [113]. This attack is called NHOP, since it affects nodes at N = {1, 2, 3, ...} hops from the attacker.
A) Events

During a HELLO_INTERVAL period, one of the following options can occur: (1) No information regarding a specific neighbour is received in (normal) HELLO messages, (2) an update of the link status of a specific neighbour is received during a (normal) HELLO_MESSAGE interval, or (3) two or more updates of the link status of a specific neighbour are received in HELLO messages, with one of them including a link status transition to “asymmetric” and the other to “symmetric”, in the same HELLO_INTERVAL period. The following events can therefore be defined:

- **NHOP_E1**: The link status was altered from “asymmetric” to “symmetric” (i.e. an update was received informing the “MPR” or “symmetric” status of a specific neighbour). The attacker is the node that had its link status altered. The target is an empty set. In other words, the attacker↔target pair is given by: {identifier of the node that had its link status altered} ↔ { }.

- **NHOP_E2**: The link status was altered from “symmetric” to “asymmetric” (i.e. an update was received informing the “lost” status of a specific neighbour). The attacker↔target pair is given by: {identifier of the node that had its link status altered} ↔ { }.

B) Event abstraction

Event abstraction is fairly simple: each time an update of a node’s link status is received, this is compared with the previous link status of the same node.

C) Specification of the attack signature (FSD)

**Figure 5-2** shows the FSD for this attack signature. Upon receiving an NHOP_E1 event, a transition is triggered from the initial state to the NHOP_S1 state. This state possesses a timeout with TIME\_max equal to the HELLO_INTERVAL. A transition to the final state is triggered if no new events are received indicating changes in the link status of the node identified as the attacker (i.e. event NHOP_E2). If an attack is carried out, an NHOP_E2 event will be generated within the HELLO_INTERVAL period and another transition to the final state is generated, this time causing the generation of an NHOP_A1 alert indicating the positive detection of an attack.
**D) Cooperation in alert management: correlation of alerts**

This attack can be detected by any LIDS that is neighbour both of the attacker and the impersonated node (compromised node). Thus if the alert management module generates a remote alert that is spread throughout the network, for example, with a TTL = 2, all nodes that detect the attack locally also receive the remote alert referring to the same attack and to the same attacker↔target pair. These alerts can be easily correlated, since they concern the same type of attack detected by different nodes, with proof against the same attacker. Thus these alerts can be combined into a single group alert, which identifies all nodes that collect proof against the same attacker, enabling a collaborative response to the intrusion, as discussed below in section 5.3.3.

5.3.1.2. **Fabrication of HELLO Messages:**

This attack is shown in Figure 4-2 in the previous chapter, where an attacker fabricates a HELLO message by advertising all nodes previously included in any HELLO message it has received, together with an additional unavailable address, with symmetric link status. On receiving this message, all of the attacker’s neighbours choose it as sole MPR. The detection of this attack shows cooperation in executing the detection algorithm. This attack is called N+1HOP, since it affects nodes at N + 1 = {1, 2, 3, ...} hops from the attacker.

**A) Events and Detection Status**

Detection of this attack begins in nodes that have their MPR set altered. These nodes generate a local event (isLocal = true) N+1HOP_E1, indicating the alteration of the MPR set, and indicate the MPR node as attacker and all nodes at two hops’ distance with which the MPR node has a symmetric link as target. This event has the following attacker↔target pair: (identifier of MPR node)↔(identifiers of 2-hop neighbours with symmetric link with the MPR node). This event is consumed locally and generates a transition to detection status N+1HOP_S1 which is dispatched in the node’s 1-hop neighbourhood (isLocal = false, isFlooded = true and TTL = 1, isClone = false). Detection is interrupted in this node and continues in destination nodes of N+1HOP_S1.
N+1HOP_S1 possesses a timeout with $\text{TIME}_{\text{max}} = 0$, indicating that a transition will be triggered as soon as it is activated in the destination nodes. On arriving at the destination nodes, N+1HOP_S1 triggers a transition to N+1HOP_S2 (isLocal = true) and the execution of a local query N+1HOP_C1, with the same attacker↔target pair as N+1HOP_E1. This query can generate three types of local events (isLocal = true):

- **N+1HOP_E2**: the local node (which executes the query) is on the target list of this query, but the MPR node is not among its 1-hop neighbours. It has the following attacker↔target pair: (identifier of MPR node)↔(identifiers of 2-hop neighbours with symmetric link with the MPR node).

- **N+1HOP_E3**: the local node (which executes the query) is in the target list of this query and the MPR node is among its 1-hop neighbours. This event has empty sets for attacker and target. It has the following attacker↔target pair: (identifier of MPR node)↔(identifiers of 2-hop neighbours with symmetric link with the MPR node).

- **N+1HOP_E4**: the local node is not on the target list of this query. It has the following attacker↔target pair: (identifier of MPR node)↔(identifiers of 2-hop neighbours with symmetric link with the MPR node).

**B) Event abstraction**

Each time an alteration occurs in the MPR list (insertion or exclusion), an event N+1HOP_E1 is generated identifying the 2-hop neighbours that have the new MPR as a 1-hop neighbour (2-hop neighbour list).

The abstraction of events N+1HOP_E2, N+1HOP_E3 and N+1HOP_E4 consists of verifying if the local node is listed in targets in the N+1HOP_C1 query. If the answer is negative, event N+1HOP_E4 is generated. If the answer is positive, the local 1-hop neighbour list is checked in search of the identifier of the new MPR. If it is found in the list the event N+1HOP_E3 is generated. Otherwise, the event N+1HOP_E2 is generated.

**C) Specification of the attack signature (FSD)**

Figure 5-3 shows the FSD for the signature of this attack. Upon receiving event N+1HOP_E1, a transition is triggered from the initial state to the N+1HOP_S1 state. This state is dispatched to the neighbourhood and detection continues in remote nodes that receive N+1HOP_S1. On receiving N+1HOP_S1, the remote node executes an immediate transition to N+1HOP_S2, making query N+1HOP_C1 (timeout transition with $\text{TIME}_{\text{max}} = 0$ for
If an N+1HOP_E3 or N+1HOP_E4 event is received, a transition is executed to the final state. Upon receiving an N+1HOP_E2, an auto-transition is triggered and the alert N+1HOP_A1 is triggered, indicating the positive detection of an attack.

**5.3.2. Attack and Attack Signature against Distributed Applications**

This section discusses the misuse detection of stepping stone attacks. This attack involves establishing a chain of telnet connections. A root node (attacker) establishes a connection with another node. A new connection is established with another node from this initial connection, and so on. This type of attack precedes more invasive attacks [105] (application level). All nodes in the chain collaborate for the detection of the attack in the data collection phase.

**A) Events**

The detection of telnet connection chain attacks is usually divided into two stages. Firstly, when a node receives a request to open a telnet connection (local event STEPSTONE_E1). This node then makes a remote query (STEPSTONE_C1) to the originating node of the connection (isLocal = false, isFlooded = false, destination = address of node that opens the telnet connection), with the attacker as the originating node of the local incoming telnet connection and the target as its own address. The STEPSTONE_C1 query seeks to identify if there are incoming telnet connections on the node that executes them. If the calling node does not have incoming telnet connections, a remote event (STEPSTONE_E2) is generated and sent to all target nodes of STEPSTONE_C1. If there is an incoming connection, a remote event (STEPSTONE_E3) is dispatched to all target nodes of STEPSTONE_C1. The attacker in this event is the originating node of the incoming local
The detection process described above only states that there is a chain of incoming and outgoing connections from a root node (attacker) to the last node in the chain (which did not start telnet connections with other nodes). However, there are no guarantees that the connections in this supposed chain are interrelated. Thus the second stage in detecting this attack consists of evaluating the existence of correlations between the connections in the chain. Here we describe only the attack signature for execution of the first stage of detection. The second stage can be conducted using any statistical correlator such as in [105].

B) Event abstraction

The abstraction of events for this attack can be conducted by executing periodic STEPSTONE_PC1 queries (isLocal=true, period = T_{PC1}) to identify the existence of active incoming telnet connections. For this it is necessary to recover the “tcpConnTable” table from the node’s local MIB, since this table is a standardised MIB variable [75]. Alternatively, it is possible to monitor the TCP packets that arrive at a node with SYN bit set and with the originating port as the known port of the telnet protocol. The procedure for executing STEPSTONE_C1 is exactly the same, with the only difference that this query is conducted on a remote node.

C) Specification of the attack signature (FSD)

Figure 5-4 shows the FSD for the attack signature. On receiving a STEPSTONE_E1 event, a transition from the initial state to the STEPSTONE_S1 state is triggered and a STEPSTONE_C1 query is sent to the attacker in event STEPSTONE_E1. On receiving a STEPSTONE_E2 event, this causes a transition to the final state. If a STEPSTONE_E3 event occurs, a transition is made to the STEPSTONE_S2 state. In this state, if a STEPSTONE_E2 event is received, an attack is declared by generating a STEPSTONE_A1 alert on making the transition to the final state. If the event received is STEPSTONE_E3, an auto-transition is executed, without any alerts being generated, but generating a new STEPSTONE_C1 query.
5.3.3. Intrusion Response

In the three attacks for which an attack signature has been completely described in the previous section, more than one node can detect the attack generated by the same attacker. Thus a correlation of simple alerts can be defined for the implementation of alerts in a management module, making it possible to reduce the probability of any false positives [116]. This process consists of storing all local and/or remote alerts referring to the same attack and the same attacker for a configurable time period. When \( K \) alerts are collected by the same node, this can generate a request for the formation of a coalition containing the identifiers of all nodes that generated alerts, thereby putting the intrusion response process into action by revoking the attacker’s certificate (Figure 3-3).

5.4. BEHAVIOUR INTRUSION DETECTION

Intrusion detection techniques based on behaviour assume that an intrusion can be detected by observing deviations from the normal or expected behaviour of a system or user. Normal or valid behaviour is extracted from prior reference information on the monitored system. The IDS compares the reference behaviour model with current activity and generates alarms each time a deviation from the original model is observed. This means that any behaviour observed that cannot be reconciled with the previously stipulated reference model is considered an anomaly and therefore potentially indicative of an attack.

Many systems based on behavioural modelling have been proposed and tested, and although they possess different characteristics and architectures, their design and development can as a rule be described in terms of three phases [29]:

![Figure 5-4 - Attack Signature: Stepping Stone for telnet connections](image-url)
• Construction of a normal or valid behavioural model: This stage consists of modelling the reference behaviour of the system. During this stage, the majority of hypotheses concern the sources of information to be used and how this information can be processed to construct a model that describes the system’s operations in a consistent and complete way. In a systematic way, but possibly a specific way, this stage can be divided into:

  - Identification of what type of audit data is to be used to describe the system’s normal behaviour. In general, the same type of information used during this modelling stage must be used (as an input) in the detection stage. It is important to note moreover that some pre-processing of the information collected is often necessary.
  
  - Construction of the behavioural model. Various types of models can be used to describe the normal behaviour of a system. Many systems have been developed using statistical modelling [56], neural networks or genetic algorithms, among other techniques. In general, these models have a pre-determined architecture with various parameters that must be adjusted automatically using some kind of learning or optimisation algorithm. In some cases, the architecture and the learning algorithm are strongly connected by means of an iterative and adaptive process for adjusting the parameters of the model progressively. In this type of model there is also the possibility of updating previously adjusted parameters to follow changes that take place in the normal behaviour of the system. Other approaches based on specialist systems are also possible, but the automatic updating of the model due to changes in behaviour can be more difficult.
  
  - Obtaining prior reference information (training). Even after defining the type of information to be used and the model’s architecture, obtaining a good set of initial reference information is not straightforward. Indeed, it is usually difficult to satisfy the requirements that the data contain no types of anomalous use and is representative of all normal behaviour in the system.

• Detection: This phase consists of making inferences regarding the operational state of the system, comparing information acquired from current system use with the behavioural model configured in the previous stage. The new data concerning previous system use is presented to the IDS. The design of the detection algorithm can vary depending on the type of information used or the system architecture, but
must also consider other criteria such as performance and robustness if it is a system involved with real time detection. Independent of the type of model used, the algorithm must allow a clear definition for the deviation to be evaluated. The deviation can be defined in a binary way, so that the IDS will define all new information presented as either normal or abnormal. Alternatively, the deviation can take the form of a significance test, i.e. observed behaviour can be evaluated as valid/invalid with a given probability. In many IDSs that operate with anomaly detection techniques, alerts generated during this phase are post-processed in order to eliminate false positives. This occurs essentially because in behavioural intrusion detection, unlike misuse intrusion detection, there is no positive recognition of an attack, but only the indication of activities that were not observed during the modelling phase.

- Updating the behavioural model in use. As system use behaviour changes, the behavioural model must be updated to avoid the erroneous indication (alerts) of anomalies by the IDS. This can be a continuous update process, but periodic updates can be tolerated even in a system that operates in real time. Updates of the behavioural model usually occur gradually and in the long term, thereby avoiding the occurrence of distorted adaptations due to erroneous use for a short period of time. Thus, if a large change in system use behaviour is about to take place, the system must be restarted, constructing the system’s behaviour model again using new reference information that reflects the new behaviour, or even altering aspects of the architecture of the system’s behaviour model to adapt it to the new circumstances. It is important to emphasise that this gradual updating of the system gives an attacker the opportunity to induce the system progressively into erroneous behaviour that will be learned as acceptable behaviour by the updating mechanism of the valid behaviour model. This is one of the most significant disadvantages of the behavioural intrusion detection approach.

In this work we have sought to design a behavioural IDS that complements the intrusion detection and response system described in the previous section. We have used a statistical modelling approach to construct the behavioural model. In this type of approach it is usually necessary to map available audit events for collection and analysis into random variables, i.e. in numeric domains, even if some audit events can already be observed in this form. Initially we plan to observe numeric values that reflect the traffic and bandwidth conditions for specific types of applications and protocols in a Manet. It should be
emphasised that statistical modelling is usually complex, since different applications and protocols possess very different statistical rules. Thus it was decided that a mixed distribution model should be created which should be adjusted to the data set to be used to characterise normal traffic in a Manet.

Obviously, the characterisation of what would be a normal traffic profile for a Manet is still an unresolved problem and there is little consensus in this respect. Our modelling must therefore be adjusted to a traffic profile considered normal for a specific Manet with clearly defined operating conditions. This means that this work is of a preliminary nature. Nevertheless, and following the example of similar work conducted in this area [16], preliminary results enable the successful discrimination of attacks characterised by significant changes in standard traffic, such as DDoS and network scanner attacks. This is therefore the principal objective of the behavioural IDS described in this section.

5.4.1. Mixed Distribution Models for Statistic Characterisation of Behaviour

A mixed distribution model is normally used to model the probability density function (p.d.f.) of a random d-dimensional variable $y$ for which the realisations are extracted from the audit information area and can be formally defined as follows:

Let $Y = [y_1, y_2, ..., y_d]$ be an observable realisation vector of $y$. A mixed distribution model for this data is defined expressing the p.d.f. of the data as the linear combination of basic kernel functions as shown in Eq. 5-1:

$$p(y) = \sum_{k} w_k g_k(y, \hat{\Theta}_k)$$

**Eq. 5-1**

where: $g$ represents each kernel function, $w_k$ are the ponderation factors of each kernel function and $\hat{\Theta}_k$ are the parameters of the kernel functions. The vector $\Theta = [w_1, w_2, ..., w_k, \hat{\Theta}_1, \hat{\Theta}_2, ..., \hat{\Theta}_k]$ represents all the unknown parameters of the mixed model, which are to be adjusted to fit $\Theta$ to $Y$. The number $K$ is the order of the system and is generally fixed.

This finite mixed model has been used to model distributions of different supposedly random phenomena[76]. An iterative algorithm for optimisation, by a criteria of maximum

---

30 A similar form can be defined considering a mixed distribution model for the joint p.d.f. of random one-dimensional "d" variables.
likelihood (ML), was presented in [30] and is called the Expectation Maximisation (EM) algorithm.

As the set of reference data must contain information on different valid behaviour, it is normally useful for this data to be clustered. The use of mixed models with automatic data clustering is immediate, by adopting a parametric mixed model [22,99]. This model is defined by assuming that each kernel function individually represents the p.d.f. of each cluster in the data set. Thus a mixed model in the order of K is directly applicable in situations where Y can be identified as originating from a populational mix of K groups. In such cases, the coefficients \( w_k \) are equivalent to the probability of each cluster \( p(k) \). Equally, the further probabilities of each realisation \( p(y_i | k) \) can be obtained, given the values of each kernel function in \( \mathcal{F} \). Given that \( p(y_i) \) can be directly estimated from Eq. 5-1 by Bayes theorem, it is possible to obtain an estimate for the further probabilities in the form \( p(k | y_i) = \frac{p(y_i | k)p(k)}{p(y_i)} \). Prior knowledge of the order of the model may not be available and it is useful for it to be automatically inferable. We have developed an algorithm for automatically determining \( K \), based on an optimisation by a criterion of entropy maximisation. This algorithm, adapted from [99] for parametric models, is described in the following section.

For multivariable data, the special case of multivariable Gaussian kernel functions makes a model known as the Gaussian Mixture Model (GMM). This model in particular can be easily adjusted iteratively by the EM algorithm, since there are closed forms for the computation conducted in each iteration. In addition, the algorithm has strong convergence properties, given a correct estimate of \( K \). Thus this work considers the case of a GMM. Therefore the description of the EM algorithm presented here refers specifically to this case. [30,76] provides a more generic description of the EM algorithm. This approach may appear a little restrictive, but some points regarding the GMM must be emphasised. In cluster analysis, the application of parametric GMM is largely adopted, since the clusters assume an elliptic format. However, for a set of data containing a group or groups of observations resulting from a number of normal populations greater than the order of the system or observations that do not possess normal characteristics, more general models need to be used. It is easy to produce more generic parametric models, using uniform, Gaussian, Gaussian with displacement and scaling distributions, as well as t distributions, since the EM algorithm has already defined for these cases [76,22,2]. Another important type of kernel function for which it would be interesting to produce the EM algorithm is Pareto distributions, which are broadly used for modelling intermittent traffic. Another possibility consists of adopting semi-parametric mixed
models [99], where a mixed model of a greater order is adjusted to data and different mixes of kernel functions adjusted by the EM algorithm are optimised to describe the p.d.f. of each cluster (i.e. the p.d.f. of each cluster is formed by different mixed models of a high order, allowing the order of the model to be higher, and therefore more generic than the number of clusters in the data.)

For the case of GMM, $g_k$ in Eq. 5-1 is substituted by $\phi(y_i,i_k,R_k)$, which denotes a normal multivariable p.d.f. with average $\mu_k$ covariance matrix $R_k$. Eq. 5-1 can be rewritten as Eq. 5-2:

$$p(y_i) = \sum_{k=1}^{K} w_k \phi(y_i,i_k,R_k)$$

where: $\Theta = [w_1, w_2, ..., w_K, i_1, i_2, ..., i_K, R_1, R_2, ..., R_K]$

5.4.1.1. EM Algorithm

The maximum likelihood estimate consists of finding an estimate $\Theta^*$ para $\Theta$ that maximises the likelihood of $y$ for a set of observations $Y = [y_1, y_2, ..., y_n]^T$. Assuming that $y_1, y_2, ..., y_n$ are independent realisations of characteristic vector $Y$, the function of logarithmic likelihood as a function of $\Theta$ is given by Eq. 5-3:

$$\log L(\Theta) = \sum_{j=1}^{n} \log \left( \sum_{k=1}^{K} w_k \phi(y_j,i_k,R_k) \right)$$

An estimate of $\Theta$ with maximum likelihood is given by the roots of Eq. 5-3, which corresponds to a local maximum of Eq. 5-4:

$$\frac{\partial \log L(\Theta)}{\partial \Theta} = 0$$

Given that it is difficult to optimise $\Theta$ directly, occult (unobserved) variables $z_{jk}$ are introduced, where $z_{jk}$ is defined as 1 or 0, independent of whether or not $y_j$ comes from the $k$-th component of the mixed model ($j = 1, ..., n; k = 1, ..., K$). The complete vector of data (unobserved) $X$ is formed by $X = [x_1, x_2, ..., x_n]^T$, where $z_j = [z_{j1}, z_{j2}, ..., z_{jK}]^T$ are variable

---

51 A Stochastic Model is more realistic in some cases (e.g. Markov Model), but this model complicates computation considerably and is not being considered at this stage of development of the project.
occult vectors for a realisation \( y_j \) with \( z_1, z_2, \ldots, z_n \) being independent realisations of a multinomial distribution consisting of an experiment with \( K \) categories, with respective probabilities \( w_1, \ldots, w_K \). Realisations \( x_i = (y_{i1}^T, z_{i1}^T), \ldots, x_k = (y_{ik}^T, z_{ik}^T) \) are considered independent and identically distributed.

For this specification, the function of logarithmic likelihood for the complete vector \( X \) is given by:

\[
\log L_c(\mathbf{O}) = \sum_{i=1}^{n} \sum_{j=1}^{K} z_{ij} \log \{ w_i \phi(y_{ij}, i, \mathbf{R}_i) \} \quad \text{Eq. 5-5}
\]

The EM algorithm [30] is effective when the process of maximising the likelihood of the complete data vector (\( X \)) is simpler than maximising the likelihood of incomplete data (Eq. 5-3). The EM algorithm is executed iteratively and consists of two steps in each iteration: step E (expectation) and step M (maximisation). Considering \( \mathbf{O}' \) as an expectation of \( \mathbf{O} \) in the \( i \)-th iteration, step E requires the calculation of Eq. 5-6:

\[
Q(\mathbf{O}; \mathbf{O}') = E_{\mathbf{O}'} \left( \log L_c(\mathbf{O}) \mid Y \right) \quad \text{Eq. 5-6}
\]

where: \( Q(\mathbf{O}; \mathbf{O}') \) is the expected conditional value of \( \log L_c(\mathbf{O}) \), given observed data \( Y \) and the current adjustment \( \mathbf{O}' \) to \( \mathbf{O} \).

Given that \( \log L_c(\mathbf{O}) \) is a linear function of occult variables \( z_{ij} \), step E is executed simply by substituting \( z_{ij} \) with its expected conditional value, given \( y_{ij} \), using \( \mathbf{O}' \) for \( \mathbf{O} \). In other words, \( z_{ij} \) is substituted in Eq. 5-6 by Eq. 5-7:

\[
\tau_i (y_j; \mathbf{O}') = E_{\mathbf{O}'} (z_{ij} \mid y_j) = \frac{w_i \phi(y_{ij}, i, \mathbf{R}_i)}{\sum_p w_p \phi(y_{ip}, i, \mathbf{R}_p)} \quad \text{Eq. 5-7}
\]

\( \tau_i (y_j; \mathbf{O}') \) can be recognised in Eq. 5-7 as the current estimate of further probability of the \( j \)-th realisation \( (y_j) \) having come from the \( k \)-th group, i.e. \( p(k \mid y_j) \). Eq. 5-7 can therefore be rewritten as follows:

\[
p(k \mid y_j) = \frac{w_i \phi(y_{ij}, i, \mathbf{R}_i)}{\sum_p w_p \phi(y_{ip}, i, \mathbf{R}_p)} \frac{p(k)p(y_j \mid k)}{p(y_j)} \quad \text{Eq. 5-8}
\]
This equation is an expression of Bayes theorem, recognising that the estimate of the a priori probability of each cluster \(\rho(k)\) is given by the current estimate of the ponderation factor \(w^*_k\).

Substituting Eq. 5-8 for Eq. 5-7, the expression is obtained for step E:

\[
Q(\Theta; \Theta') = \left( \sum_{k} \sum_{i} w'_k \phi(y_i, \theta_i, \mathbf{R}^k_i) \right) \exp \left( \frac{\sum \log(p(y_i))}{n} \right) \tag{5-9}
\]

In step M in the \((i + 1)\)-th iteration, the objective is to choose \(\Theta^{i+1}\) that maximises \(Q(\Theta; \Theta')\). Thus the current adjustment for the proportions of the mixture \((w^*_k)^{i+1}\), the components of the average \((\mathbf{i}^{i+1}_k)\) and the covariance matrices \((\mathbf{R}^{i+1}_k)\) are given explicitly by Eq. 5-10:

\[
w^*_k = \frac{1}{n} \sum p(k | y_i)
\]

\[
\mathbf{i}^{i+1}_k = \frac{\sum p(k | y_i) y_i}{\sum p(k | y_i)}
\]

\[
\mathbf{R}^{i+1}_k = \frac{\sum p(k | y_i)(y_i - \mathbf{i}^{i+1}_k)(y_i - \mathbf{i}^{i+1}_k)^T}{\sum p(k | y_i)}
\]

An interesting characteristic of the EM algorithm is that the likelihood of mixture \(L(\Theta)\) can never decrease after an EM sequence. Thus, \(L(\Theta^{i+1}) \geq L(\Theta')\), implying the convergence of \(L(\Theta)\) for a certain value \(L^*\), if the sequence of values for the likelihood is limited. Steps E and M are alternated repeatedly until the likelihood or the expectations for the parameters change from an arbitrarily small value, indicating the convergence of the algorithm.

The EM algorithm can be summarised as follows:

**Algorithm 2 – EM**

1: Starts \(\Theta^0\) with random values for \(w^*_1, w^*_2, ..., w^*_K, \mathbf{i}^{0}_1, \mathbf{i}^{0}_2, ..., \mathbf{i}^{0}_K, \mathbf{R}^{0}_1, \mathbf{R}^{0}_2, ..., \mathbf{R}^{0}_K\)
2: For \(i = 0\), calculate \(L^i\) according to Eq. 5-9.
3: For \(i = i+1\), calculate \(\Theta^{i+1}\) (Eq. 5-10) and \(L^{i+1}\) (Eq. 5-9)
4: If \(L^{i+1} - L^i > \delta\) (convergence constant), repeat 3
5: Update the real values of the parameters \(\Theta^* = \Theta'\)

---

Embrapa 19/11/02 15:25
Mise en forme : Puces et numéros
5.4.1.2. Principal problems in applying the EM algorithm and proposed solutions

The first problem in applying the EM algorithm, as described in the previous section, is related to the fact that in general the likelihood function has multiple local maxima. Thus different initiations can lead to different adjusted models, corresponding to maxima that are different from the function. This is especially critical in the case of random initiations, since any local maximum can be reached, resulting in sub-optimum or even inadequate adjustments. Various initiation procedures have been proposed to deal with this problem [76,39,22]. Among them, an immediate solution consists of making a fixed number ($C_{max}$) of random initiations for each application of the EM algorithm, and using whichever results in a maximum value for step E, after convergence. Pre-clustering [76,39] can also be used (which provides estimated values for the a priori probabilities of each cluster).

Moreover, if components of covariance matrices have no restrictions (i.e. consider that this matrix is a full matrix and not a diagonal matrix, for example) the likelihood function is not limited, given that each point of data implies a singularity on the perpendicular of the parameter space [76]. In addition, special attention must be taken in cases where an adjusted component (cluster) possesses a generalised variance (i.e. the determinant of the matrix of covariances) that is very small (not zero), implying relatively large values for the local maximum. This component corresponds to a cluster containing few points, which are relatively close to each other. It is therefore necessary to monitor the relative size of the proportions of the adjusted mixtures, the components of the generalised variances in order to eliminate these false local maxima. It is also necessary to monitor the Euclidian distances between the averages of adjusted components, in order to verify whether the implied clusters represent a real separation between the averages or if one or more clusters have almost fallen into a sub-space of the original characteristic vector [76].

Given the adaptations and precautions mentioned in the previous paragraphs, a modified version of the EM algorithm is summarised as follows:

<table>
<thead>
<tr>
<th>Algorithm 3 – Modified EM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Start with a counter $c = 0$</td>
</tr>
<tr>
<td>2: $c = c + 1$</td>
</tr>
<tr>
<td>3: Start $\Omega$ with random values for $w^0_0, w^0_1, ..., w^0_k, i^0_0, i^0_1, ..., i^0_k, R^0_0, R^0_1, ..., R^0_k$</td>
</tr>
<tr>
<td>4: For $i = 0$, calculate whether $L^i$ corresponds to Eq. 5-9.</td>
</tr>
<tr>
<td>5: For $i = i + 1$, calculate whether $\Omega^{i+1}$ (Eq. 5-10) and $L^{i+1}$ (Eq. 5-9)</td>
</tr>
<tr>
<td>6: If $L^{i+1} - L^i &gt; \delta$ (convergence constant), repeat 5</td>
</tr>
</tbody>
</table>
7: If the determinant of any of the covariance matrices $< \epsilon$ (a small constant), repeat 2
8: If ($c = 1$) or ($L_i > L_{opt}$) then make $L_{opt} = L_i$ and $\Theta_{opt} = \Theta'$
9: If $c <= C_{max}$, repeat 2
10: Update the real values of the parameters $\Theta^* = \Theta_{opt}$.

5.4.1.3. Automatic estimation of the optimum order of the model

For the purposes of the EM algorithm, the order of the model ($K$) must be assumed a priori. Considering, however, that in many cases the number of partitions is not known a priori, it is useful for there to be a mechanism to discover the most probable number of partitions for a given model. The objective here consists of constructing an estimate for $K$ that implies an “ideal partition”, i.e. $p(k | y_i)$ is close to the unit for a value $k$ and close to zero for all other values, for each realisation $y_i$. As described in [99], this ideal partition must be obtained by minimising the Shannon entropy given observed data

$$H_k = -\sum_{i=1}^{K} p(k | y_i) \log(p(k | y_i))$$

Eq. 5-11

The expected value of this entropy is found by taking the mean of $H_k$ on all observed data

$$E^* (H_k) = -\sum_{k=1}^{K} \sum_{i=1}^{n} p(k | y_i) \log(p(k | y_i)) / n$$

Eq. 5-12

where: $E^*$ denotes an expectation estimator.

Then adjust $K_{max}$ models with different orders ($K = 1, 2, ..., K_{max}$) and evaluate the expected entropy (Eq. 5-12) for each of them. The model that results in a minimum average is considered the optimum model.

The EM algorithm with automatic estimation of the optimum order can be summarised as follows:

<table>
<thead>
<tr>
<th>Algorithm 4 – EM with Estimation of the Optimum Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: $K = 0$, $H_{opt} = 0$, $K_{opt} = 1$</td>
</tr>
<tr>
<td>2: $K = K + 1$</td>
</tr>
<tr>
<td>3: Adjust the model of the order $K$ to data $Y$ using the modified EM algorithm (algoritmo 3)</td>
</tr>
<tr>
<td>4: Calculate expected value of $H_k$ (Eq. 5-12)</td>
</tr>
</tbody>
</table>

This argument can be easily verified by the simple inspection of the expression for the entropy. A formal treatment can be found in [9].
5.4.1.4. Detection algorithm

During the detection phase, the behaviour model has already been computed and is available for making inferences about new data presented to the system. The objective is to define some penalty $\lambda$ (e.g. $0 \leq \lambda \leq 1$), indicating the level of normality concerning this realisation from certainly anomalous ($\lambda = 0$) to a certainly normal ($\lambda = 1$).

Many different approaches for defining such criteria for the statistical behaviour model represented by Eq. 5-1 are possible. In this work we have defined a detection procedure composed of two stages: a (Bayesian) classification stage and a cluster relevance inference.

Classification is straightforward for parametric mixed models and consists of the evaluation of the posterior cluster probabilities of each cluster conditioned to the new data $y^*$, i.e. $p(k | y^*)$ (Eq. 5-8) for $k = (1, 2, ..., K)$.

Cluster relevance inference is a little more complex. As all kernel distribution functions used in our model have a continuous nature, there is no practical reason to consider the posterior probability of the new data, conditioned to the probability of cluster $p(y^* | k)$ by the simple evaluation of the p.d.f. of the new point. A more realistic approach consists of evaluating the probability of the new data being contained in some relevance interval $(\Pi_i)$, defined as a function of the new observation $y^*$ and of the cluster distribution parameters (e.g. $\mu_k$ and $R_k$), which can be expressed formally as Eq. 5-13:

$$p(y^* \in \Pi_i | k) = \int_{\Pi_i} g_i(y, \hat{e}_i) d\Pi_i$$  \hspace{1cm} \text{Eq. 5-13}$$

Indeed, the probability defined in Eq. 5-13 resembles a cumulative distribution function (c.d.f.), if we define $\Pi_i$ as shown in Eq. 5-14 below [59]:

$$\Pi_i = \left\{ y \in \mathbb{R}^d | \frac{||y - \hat{e}_i||}{||R_i||} \geq y^* \right\}$$  \hspace{1cm} \text{Eq. 5-14}$$

where: $|| \cdot ||$ and $\| \|\| \|$ denote some kind of norm operators and $\gamma$ is a constant that depends on $y^*$. 

5: If (K = 1) or (H_k < H_{opt}) then $H_{opt} = H_k$; $K_{opt} = K$; and $\Theta_{opt} = \Theta^*$
6: If $K < K_{max}$ (a fixed constant), then repeat 2
7: Update the real model order with the optimal value: $K = K_{opt}$
8: Update the real values of the parameters $\Theta^* = \Theta_{opt}$.  

119
Finally, the function of the detection penalty can be defined according to Eq. 5-15:

\[ \lambda(y^*) = \sum_{k} p(k \mid y^*) p(y \in \Pi_k \mid k) \]  

**Eq. 5-15**

### 5.4.1.5. Detection algorithm for real time operation with GMM

The procedure for creating the reference behavior model is usually executed *off-line*. Restrictions related to computational complexity are not severe at this point. However, it is desirable that phases of detection and updating of the behavior model may be carried out continuously. Thus, algorithms for model detection and updating must be designed for real time operation. This section shows how the detection process can be computed in real time.

The Eq. 5-13 cannot always be evaluated in an analytic manner. A general solution would be to evaluate this numerically integer equation, but that can be not feasible as the numeric evaluation is computationally intensive even in unidimensional or bidimensional cases, making the real time execution difficult or impossible [40]. In fact, Eq. 5-13 may be difficult for arbitrary \(g_0\), core functions with one algorithm computationally effective for evaluation of this integer being established in the special case of Gaussian distributions. Thus, when a GMM is used, the Eq. 5-13 evaluation may be done choosing conveniently the non-defined elements of this equation, that is, the standard operator and \(\gamma\). Therefore, \(\Pi_k\) is defined as the complementary space (concave) of the equal density ellipsoid (em \(\mathbb{R}^d\)), whose boundary contains \(y^*\) and has as gravity centre \(\mu_k\). That means \(\Pi_k\) is internally limited by a ellipsoidoid surface with d dimensions, formed by all points having the same density value as \(y^*\) (i.e. \(\phi(y, i, R_\beta) = \phi(y^*, i, R_\beta)\)). Thus, rewriting Eq. 5-14, we have Eq. 5-16:

\[ \Pi_k = \left\{ y \in \mathbb{R}^d \mid \sum_{\alpha} (y^r_\alpha - \mu_\alpha) [R^{-1}_\alpha]_{\beta \rho} (y^r_\rho - \mu_\rho) \geq \gamma^2 \right\} \]  

**Eq. 5-16**

where: \(y = (y_1, y_2, ..., y_d)\); \(i = (\mu_1, \mu_2, ..., \mu_d)\); \([R^{-1}_\alpha]_{\beta \rho}\) is the element of the \(\alpha\)-th line and the \(\beta\)-th column of the reverse co-variance matrix, and \(\gamma\) is given by Eq. 5-17:

\[ \gamma^2 = \sum_{\rho} (y^r_\rho - \mu_\rho) [R^{-1}_\rho]_{\beta \rho} (y^r_\beta - \mu_\beta) \]  

**Eq. 5-17**
This strategy can be illustrated for the uni and bidimensional spaces, as shown in Figure 5-5 and Figure 5-6, respectively. This last one was designed for a bi-variant Gaussian distribution, with diagonal co-variance matrix (non co-related).

Figure 5-5 - II for a cluster with unidimensional Gaussian distribution

![Figure 5-5 - II for a cluster with unidimensional Gaussian distribution](image)

Figure 5-6 - II for a cluster with bi-variant Gaussian distribution and diagonal co-variance matrix

![Figure 5-6 - II for a cluster with bi-variant Gaussian distribution and diagonal co-variance matrix](image)

This procedure may be used also for cases of multivaried Gaussian distributions, with non-restricted co-variance matrix, as it is always possible to find a linear transformation that map a given multivaried Gaussian distribution in a non-co-related Gaussian distribution (diagonal co-variance matrix) with the same value of $\gamma$.\footnote{This procedure is called analysis of main component (PCA) \cite{60} and is equivalent to the rotation of coordinate axis in the main directions. The rotation matrix is formed by the original co-variance matrix self-vectors, whose self-values correspond to the variations of new distribution. The translation used to position the main axis origin on the average point is implicit in Eq. 5-17.}
As the observed data may belong to a multidimensional space \((\mathbb{R}^d)\), a generic distance \(\gamma'\) is defined in Eq. 5-18. That enables the normalization of probabilities expressed in Eq. 5-13 for data belonging to different dimensional spaces, allowing the reduction of computation to unidimensional space, what can be carried out through a simple table lookup procedure.

\[
\gamma' = \frac{\gamma}{\sqrt{d}} \quad \text{Eq. 5-18}
\]

5.4.1.6. Recursive updating of model adjusted parameters

As the behavior in the usage of information systems changes frequently, the reference behavior model must also be updated to prevent false positives. The updating must be considered as an adaptation of the original model to accommodate slight variations in system behavior, considering the model may become invalid or incomplete in case of significant changes.

The approach proposed herein takes into account the possibility of updating the behavior model through the recursive and continuous updating of model parameters. Thus, the updating happens in the clusters \((w_i)\) probabilities and in core distributions parameters. The usual estimates are used for the continuous estimates of the statistics of model [110]. It is important that both the logarithm similarity and the entropy may be equally estimated and compared with pre-established values (i.e. values obtained after training), as such values may offer an idea of “how good” the new model is, when compared with the reference model. The estimate for recursive and continuous updating, \textit{a priori}, of clusters probabilities and of first and second order distribution snapshots are shown in Eq. 5-19.

\[
w_i^{new} = w_i^{old}(1-\eta_1) + \eta_1 p(z_i | y_i) \quad \text{Eq. 5-19}
\]

\[
i_i^{new} = i_i^{old}(1-\eta_2 p(z_i)^{old}) + \eta_2 p(z_i)^{new} y_i
\]

\[
R_i^{new} = R_i^{old}(1-\eta_3 p(z_i)^{old}) + \eta_3 p(z_i)^{new} (y_i - i_i^{old})(y_i - i_i^{old})^T
\]

The estimates are applicable only in cases where behavior changes happen in the long term and the system, application and/or users remain stable. Constant \(\eta_1, \eta_2, \ldots, \eta_3\) must be carefully chosen to prevent non-stability ((1/n can be a first choice for \(\eta_1\) and values even lower must be used for \(\eta_2\) and \(\eta_3\), as changes in the distribution snapshots have more energy than changes \textit{a priori} in clusters probabilities).
5.4.2. Characterization of Normal Traffic in a Manet and Creation of a Normal Behavior Model

The intention is to create a behavior model to characterize normal traffic conditions in a Manet. There is no consensus over what would be a traffic profile that could be considered as typical in a Manet. In fact, except for some control protocols and network signaling (i.e. routing) which are present in almost all Manets, it is likely that each network of this type has a traffic profile dependent on the application for which the network was designed. Therefore, the characterization of normal traffic for a Manet has to be done individually for each case, adjusting the normal behavior model to a specific situation, referring to a defined network application.

Another important aspect is the fact that it is difficult to obtain real samples of the traffic of an operating Manet, which are proven to be free of possible intrusion traces. An alternative widely used in previous works on Manets is simulations. One advantage of this approach is that it is possible to simulate different factors like mobility and network use, in a repeated and controlled way. However, the validity of simulations can always be questioned when the intention is to model real environments, whose many factors may not have been properly considered in the simulation. As it is difficult to install a real Manet, this work uses a characterization of the normal traffic profile obtained in a simulation. It is important to mention that the training process of the behavior model and the intrusion detection process are exactly the same cases where real data on network traffic are available. Thus the intention is to validate an intrusion detection process per behavior, using simulated data and, in later works, this process will be applied to more real situations, in which real traffic data for training and intrusion detection are available.

As it is a simulation, three aspects are defined for the characterization of the traffic in the Manet: control traffic, applications executed in the nodes and the node mobility model:

- Control traffic: consists basically of the traffic generated by the routing protocol (UDP), and the ARP traffic. The traffic generated by the self-configuration protocol is not considered, as the simulated network has a fixed number of nodes, new additions or network initializations not being considered. The DNS traffic is also not considered, as that is an issue still open in terms of Manet.
- Applications executed in the nodes: for the scenario to be sufficiently representative, four types of traffic generated by different applications in all nodes of the simulated network are used. They are: simple remote session (i.e. telnet), blast data transference (i.e. FTP), continuous data transference with constant bit
rate (CBR) (i.e. videoconference or audio-conference) and simple application of asynchronous question and answer (i.e. ping). For each traffic type some more conditions are defined for traffic distribution throughout the network. These parameters are adjusted so that average occupation of wireless links is around de 20% of its capacity.

- Simple remote session (telnet)
  - uses TCP;
  - generated traffic is bidirectional;
  - interval between messages: Poisson process;
  - multiple sessions between different origins/destination, the origin and destination nodes (evenly distributed), the start time (Poisson process) and the session duration (normally distributed) randomly defined.

- Blast data transference (FTP)
  - uses TCP;
  - random “file” size (normally distributed);
  - multiple transfers between different origins/destinations, including origin and destination nodes (evenly distributed) and the start time (Poisson process).

- cbr data transference (videoconference)
  - uses UDP;
  - 128kbps fixed cbr rate;
  - multiple transfers between different origins/destinations, being the origin, destination nodes (evenly distributed) and the start time (Poisson process) and the session duration (normally distributed) randomly defined.

- Simple application of asynchronous question and answer (ping)
  - uses ICMP;
  - always sends 4 requests, at 1 second intervals;
  - always sends answers;
  - multiple transfers between different origins/destinations, being the origin and destination nodes (evenly distributed) and the start time (Poisson process) randomly defined.
• Mobility model: uses *random waypoint algorithm* model developed by CMU and which enables an even distribution of nodes in a pre-defined area, usually rectangular. Uses, for simulation purposes, a 50-node Manet in a 250m x 250m area with 50m transmission reach, resulting in an average neighboring of 6.28 nodes.

• Simulation time is 1,000 seconds.

A Gaussian mixed model may be adjusted to the traffic conditions created as per the definitions above, only by defining the variants that will be monitors. Examples of variants that can be monitored are defined above, considering the following monitored variants: incoming connection/session rate, session duration, number of packages received with errors, etc. It can be noted that such variants are co-related characteristically for the considered applications.

### 5.4.3. Detection of DDoS and Port Scanner Attacks

Many DDoS attack types have been created recently. As a rule, for such an attack, an opponent must have compromised a certain number of goals, in which the attack generation tools are installed. The real attack happens at a second phase, when the tools installed in all available goals generate excessive traffic against a new particular goal, flooding it with spurious traffic packages, causing it to become unavailable due to network processing capacity overload. It is the purpose of this work to detect only the second phase of the attack, that is, when many nodes are generating spurious traffic to one node goal to be made unavailable.

A collection of the main tools and respective known\(^\text{34}\) DDoS attacks may be viewed in Table 5-2.

In the case of port scanner attacks, there is also the generation of excessive illegitimate traffic against the goal being scanned, the difference being that this traffic is not necessarily originated in multiple nodes, as in the case of DDoS. However, it is possible to have a distributed port scanner attack, where information on the scanned node is gathered from different nodes. The characterization of traffic generated by port scanner attacks is shown in Table 5-3.

\(^\text{34}\) An updated list of tools for DDoS attacks, technical analysis and useful links may be viewed in [http://staff.washington.edu/dittrich/](http://staff.washington.edu/dittrich/).
Table 5-2 – Characterization of Traffic Generated by DDoS Attacks

<table>
<thead>
<tr>
<th>DDoS Attack</th>
<th>Type of Generated Traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>smurf</td>
<td>flooding of ICMP echo-reply packages</td>
</tr>
<tr>
<td>trinoo</td>
<td>flooding of UDP datagrams in random ports</td>
</tr>
<tr>
<td>TFN2K</td>
<td>flooding of ICMP, UDM and TCP syn (flag syn pre-set) packages; wrong packages; smurf</td>
</tr>
<tr>
<td>TFN2K (ping flood)</td>
<td>flooding of ICMP and smurf packages</td>
</tr>
<tr>
<td>TGN2K Targa 3</td>
<td>invalid IP packages</td>
</tr>
<tr>
<td>stacheldraht v.2.666</td>
<td>flooding of ICMP, UDP, TCP syn (flag syn pre-set) packages, TCP null (no pre-set flag), TCP ack (flag ack pre-set) and smurf</td>
</tr>
<tr>
<td>shaft</td>
<td>flooding of ICMP, UDP, TCP syn (flag syn pre-set) packages</td>
</tr>
<tr>
<td>mstream</td>
<td>flooding of TCP ack (flag ack pre-set) packages</td>
</tr>
</tbody>
</table>

Table 5-3 – Characterization of Traffic Generated by Port Scanner

<table>
<thead>
<tr>
<th>Port Scanner</th>
<th>Type of Generated Traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP port scanner</td>
<td>successive requests for TCP connection opening, in different ports</td>
</tr>
<tr>
<td>UDP port scanner</td>
<td>successive UDP datagrams, in different ports</td>
</tr>
</tbody>
</table>

To detect such attacks using the detection per behavior proposed in previous sections, separate behavior models (and detection) for TCP, UDP, ICMP and IP traffic area created. As shown in Table 5-4, for each model a set of related variants is monitored. As for the variants considered for monitoring, standard variants of MIB II [16] are used, making the gathering of such information easy, as this information may be easily available in Manet nodes when standardized SNMP agents are installed. Also a sensor module similar to the one defined for intrusion detection per incorrect usage is used.

Table 5-4 shows what attacks are to be detected using a system normal behavior model, of the GMM type, having as reference data the simulated traffic, generated according to the definitions in the previous section. The purpose of the proposed detection per behavior mechanism is to differentiate the attack from the network normal traffic.

A collaborative monitoring can also be used, defining a sensor to maliciously and wirelessly hear and synthetize information on the neighbors traffic. However, it is important to mention that this approach is not very effective in the DDoS case, as the node that monitors the neighborhood of the goal node would become equally unavailable. For the case of scanner attacks, this method may be effective, allowing neighbors to detect attacks against a goal node.
## Table 5.4 – Behavior Models and Monitored Variants

<table>
<thead>
<tr>
<th>Behavior Model</th>
<th>Variants to be monitored</th>
<th>Attacks possibly detected</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>- connection or incomes number/rate</td>
<td>- TFn and TFn2K</td>
</tr>
<tr>
<td></td>
<td>- duration of a connection</td>
<td>- stacheldraht</td>
</tr>
<tr>
<td></td>
<td>- tcpInErrs*</td>
<td>- shaft</td>
</tr>
<tr>
<td></td>
<td>- tcpNoPorts*</td>
<td>- mstream</td>
</tr>
<tr>
<td></td>
<td>- TCP scanner</td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>- udpInDatagrams</td>
<td>- trinoo</td>
</tr>
<tr>
<td></td>
<td>- udpInErrs*</td>
<td>- TFn and TFn2K</td>
</tr>
<tr>
<td></td>
<td>- udpNoPorts*</td>
<td>- stacheldraht</td>
</tr>
<tr>
<td></td>
<td>- shaft</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- UDP scanner</td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td>- icmpInEchos</td>
<td>- smurf</td>
</tr>
<tr>
<td></td>
<td>- icmpOutEchos</td>
<td>- TFn (ping flood)</td>
</tr>
<tr>
<td></td>
<td>- icmpInErrs*</td>
<td>- stacheldraht</td>
</tr>
<tr>
<td></td>
<td>- shaft</td>
<td></td>
</tr>
<tr>
<td>IP</td>
<td>- ipReasmFails*</td>
<td>- TFn2K (Targa 3)</td>
</tr>
</tbody>
</table>

* these variants are observed with zero average and variance in the reference model creation data, as they are error conditions not observed in simulated traffic. Therefore, for this case, usage of such variants results in singularities in the maximization function of EM algorithm, and must be avoided. In real networks, however, these variants have non-zero values which reflect occasional failures of the monitored network/system.

### 5.4.4. Response to Intrusions

In the case of DDoS attacks, the attack origin (opponent) cannot be clearly identified in the generated spurious traffic packages, as they contain erroneous information in most cases. The defense alternative for this type of attack is to avoid the forwarding of spurious traffic, requiring the collaboration of all network entities that forward traffic from their origin points (i.e. nodes that have been victims and which have the attack tools installed in their systems) to the final goal (the node that suffers the DDoS attack). In the case of Manets, this collaboration already exists, as a principle, as a result of the presence of a L-IDS in each node. A continuation of this work will investigate the possibility of relating alarms generated by different nodes on the path of spurious traffic packages so as to identify the path of such packages, which might then be automatically filtered, for the duration of the attack.

This method of intrusion response, which was not developed in the scope of this work due to time constraints, shall be even more effective in the case of port scanner attacks, as in such cases, the origin of packages is true and may be identified. That would enable another type of answer – based on the revoking of attack origin certificate. Identification of attack origin would not happen through the proposed intrusion detection per behavior, but through the alarm relating mechanism that would allow the identification the path of malicious packages from origin to destination (goal node).
6. EXPERIMENTS AND RESULTS

For validating the proposed security model, the security services are implemented and tested in an experimental Manet, initially formed by 10 mobile nodes. An attack generator is equally developed and allows the simulation of an opponent in the trial network. The security mechanisms are applied to the protection of routing protocols and self-configuration. The OLSR routing protocol and the DCDP self-configuration protocol, with improvements proposed in [14] are used. A simulation model for the validation of security services in distribution environments and more generic topology is also proposed. This model is used to evaluate the IDS per behavior, described in section 0.

6.1. EXPERIMENTAL PLATFORM

To create an experimental Manet, 10 computers are used, each with a IEEE802.11b (WiFi) (Attachment I) network interface configured for operation in ad hoc mode. These hosts have operational system with kernel GNU/Linux (version 2.4.7), distribution Red Hat (version 9.0) installed. WEP (wired equivalent privacy) (Attachment I), a mechanism to protect the link level, is not used in the experiments, but that does not affect the validation of the security mechanisms being considered, as such mechanisms operate in higher layers of the network. Notwithstanding, in real networks, this data link level security mechanism must be used.

The OLSR protocol implementation developed by the University of Oslo (Unik), Norway [108], version 0.4.7 (uolsrd) is used. This implementation, programming language C, has a BSD license and may be freely used and modified. This version may be compiled for environments GNU/Linux and Windows.

Implementations for DCDP self-configuration protocol, available for download and usage, were not found, as the definition of a self-configuration protocol for Manet is at initial phases. Therefore, an implementation was developed [14]. It contains only the network incoming and outgoing nodes. Mechanisms for joining and partition have not yet been implemented. DCDP implementation is integrated to the routing daemon uolsd (language C), allowing the usage of improvements using MPRs for the floods required by the protocol.

For the information on routing and self-configuration services to be available to the intrusion detection system, or to any other services, an SNMP (OLSRAgent) agent is developed to implement the experimental MIB for the OLSR, proposed herein. The agent is developed using package Agent API[^36] developed in Java2. The package has a GNU (version 2) public license, and may be freely used and have its source code modified. For communication between the SNMP agent and the routing-self-configuration daemon (it is the same package) the SMUX protocol, proposed by W3C as a experimental multiplexing protocol for session management, is used. This protocol offers a “light” communication channel to the communication layer on top of a TCP connection. Specification of implemented OLSR experimental MIB is on Attachment VI. As the SNMP agent is created in Java2, it runs in any platform having a Java virtual machine compatible with Sun API Java, version 1.4.0 or later.

Agent SNMP of NetSNMP[^37] project (snmpd) is also used, to gather information from the MIB-II. This agent, created in the C language, may be compiled for many platforms, including Unix/Linux and Windows.

The L-IDS is (LIDS) implemented separately, in Java2, using mobile agents platform aglet developed by IBM [^63].

A package for the generation of attacks against routing and self-configuration protocols is also implemented. This software must be capable of maliciously listen to the environment, to gather information on the neighboring nodes that can be used in attacks and generate messages in the network, received or forwarded, with modifications (modification attack) or new messages inserted in the network (production attack). Any IP or MAC[^38] address may be used as origin address. Messages may be transmitted in unicast or broadcast, depending on the specified IP destination.

The package capture library pcap[^39], developed as part of the tcpdump project, is used for the implementation of the attack generator. For the generation of packages in the network the standard Unix/Linux socket interface is used, upon creation of a raw socket which allows the creation and sending of data link frames (MAC). This module is created in language C and is available only for Unix/Linux environments.

[^38]: Some network interfaces do not allow a frame to be transmitted with a MAC address different from its built in address (e.g. Lucent). In such cases, the impersonation is limited to the IP address.
6.2. NETWORK TOPOLOGY AND MOBILITY

The simulation of topologic and mobility standards in the experimental Manet described in the previous section is fairly limited, and the repetition of an experiment under the same conditions is difficult. In fact, a Manet topology may vary in unpredictable manners. The versatility of the electromagnetic wave propagation in the presence of obstacles, of signal drop with distance and of mobility makes modeling difficult to the point of making the issue an impossible one. Especially, it is important to have in mind that mobility does not refer only to the mobility of Manet nodes, but also to the dynamics of propagation space. Therefore, when a door opens in an indoor environment or when a vehicle passes between two nodes, the link distribution changes.

For the evaluation of topology and mobility aspects of the proposed services, link distribution simulation models (propagation) for mobile networks are used. Such models are created from simplifications of real scenarios. Two types of simplified models for the definition of network topology, considering mobility, are used in the experiments [12]. The first, called random graph model, consists of a model applicable to indoor environments. In this model, the link between any two nodes exists, at a given time, with probability $p_1$. Mobility is simulated through a simple process which defines transitions from link state between existing and non-existing, and vice-versa. If it exists, the link may no longer exist, at a given time, with probability $p_2$. Equally, a non-existing link may become existing, at a given time, with probability $p_3$ (usually, $p_2 = p_1$ keeps the number of existing links away from zero, when $p_2 > p_3$, or from the total possible number of links, when $p_3 > p_2$). Probabilities $p_1$, $p_2$ and $p_3$ are always the same for all the nodes, independently of the positions in which they are. This model assumes the existence of the link depends much more on the propagation conditions and obstacles between two nodes (i.e. walls, furniture, etc.) than on its position in the space.

The other model, called random unit graph model, is applicable to outdoor or indoor environments without many obstacles. In this model, the link between two nodes exists only if distance between them ($d$) is smaller than a given value, representing the reach of the wireless interfaces link. Mobility may be simulated in this model in many different ways; this work uses the model of random point path [12]. As per this model, upon coming to a given destination point, a node remains at it for a constant time ($t_{stop}$) and then a new destination point is randomly chosen (evenly distributed throughout the simulated space). Movement speed is randomly chosen, having an even distribution between a maximum and minimum
value ($V_{\text{max}}$ and $V_{\text{min}}$). If initial distribution of nodes in space is initially even, it will retain such characteristic stationarily. In this case, node density, stationarily, may be easily calculated dividing the total node quantity in the simulation per area (2D) or volume (3D) of space region being simulated.

The used simulation environment is the Network Simulator 2 (NS-2)\(^{40}\). This environment was compiled with extensions for ad hoc mobile networks developed in the context of the Monarch CMU (Carnegie Mellon University – USA)\(^{41}\) project, and extensions for OLSR simulation developed by IRINA – France\(^{42}\).

Two modules for conversion of NS-2 trace files to the format of network analyzers (ns2tcpdump) and for identification of MIB-II important variant values from packages incoming and outgoing nodes (tcpdump2mib) were also developed. These modules enable the usage of simulated data to verify the operation of implemented security services. The modules were created in the C language and use parts of libpcap and ucd-snmp (NetSNMP) library codes.

6.3. EXPERIMENTS WITH OLSR AND DCDP PROTOCOL VULNERABILITY

The uolsr daemon, which implements routing and self-configuration protocols, may be used with or without the authentication service (i.e. MAE). To run the routing daemon the command\(^ {43}\) is:

```
./uolsr [-i <interface> -cert [<node certificate>] -share [<part of private key>] -autoconf]
```

where:

- `<interface>`: identifier of network interfaces where daemon will be run (e.g. eth0), and may be a list of interfaces;
- `<node certificate>`: digital certificate used to sign generated messages;
- `<part of private key>`: part of ACD private key to be used by the node.

---

\(^{40}\) Available in http://www.isi.edu/nsnam/ns/ (viewed in August 2004).


\(^{42}\) Available in http://hipercom.inria.fr/OOLSR/ (viewed in August 2004).

\(^{43}\) Only command options sensitive to the part added to the program are shown. The uoldr has many other command line options which are not shown here.
Option -cert controls usage of MAE. In case it is not present, self-configuration and routing services run but generated messages will not have a MAE. When the option –cert is specified, all generated messages are signed. This option may or may not have parameters. In case it is specified without parameters, the node must obtain a certificate through the collaborative process (i.e. via L-Certs) before generating and receiving messages from the routing and self-configuration protocols. Otherwise, the option receives as parameter a PEM file containing, in addition to the node/user certificate, the certificate of the certificate authority signing it (equally used to verify MAE of received messages) and the private key associated to the certificate, for signature of messages. In this case, only the certificate renewal will be carried out collaboratively.

The option –share controls the node action on the collaborative certification services. In case it is not present, the node cannot be part of coalitions for rendering collaborative certification services, that is, the node will not have a part of the ACD private key. When option –share is specified, the node runs, along with uolsr, an L-Cert instance. This option may or may not have parameters. In case it is specified without parameters, the node must obtain its part of the ACD private key through the collaborative process (i.e. via L-Certs) before entering coalitions for rendering certification services. Otherwise, the option receives as parameter a PEM file containing its part of the private key. In this case, only the updating of the private key part will be done collaboratively.

Finally the option –autoconf indicates the self-configuration process must be carried out by this node. In case this option is not specified, only the routing protocol is run and the node cannot be configured or comply with requests using the self-configuration protocol. The specification of –autoconf option indicates the self-configuration service must be run for the listed interfaces, the self-configuration of IP address of such interfaces starting soon after the uolsr service initialization is complete. This initialization includes obtaining a certificate, if required and not specified as an attribute of option –cert.

Attacks against OLSR protocol, defined in section 4.3.1.1, and customer and server attacks against DCDP protocol, defined in a preceding section, are implemented in the attack generator and may be run from any nodes of the experimental network. To carry out attacks the command is:

```
./attack [-i <interface> ]-a <type of attack> -ip <IP address of the target to be impersonated > -mac <MAC address of target to be impersonated > -cert <digital certificate of the message issuer > -v -p]
```
where:

- `<interface>`: identifier of network interface where attack will be generated (i.e. eth0);
- `<type of attack>`: string identifying the attack type, as shown in Table 5-1;
- `<IP address of the target to be impersonated>`: IP address to be impersonated;
- `<MAC address of target to be impersonated>`: MAC address to be impersonated;
- `<digital certificate of message issuer>`: digital certificate used for signing generated/forwarded messages, must be the same certificate used by the node having the IP (and MAC) address to be impersonated, in case option “-IP and -MAC” are specified.

### Table 6-1 – Attacks implemented in the `attack` program

<table>
<thead>
<tr>
<th>identification string</th>
<th>Attack</th>
<th>attacked protocol</th>
<th>options to be specified</th>
</tr>
</thead>
<tbody>
<tr>
<td>nhop</td>
<td>HELLO Production + Impersonation</td>
<td>OLSR</td>
<td>-a, -IP [-MAC]</td>
</tr>
<tr>
<td>nhop+1</td>
<td>HELLO Production</td>
<td>OLSR</td>
<td>a</td>
</tr>
<tr>
<td>nhop+2</td>
<td>TC Production</td>
<td>OLSR</td>
<td>a</td>
</tr>
<tr>
<td>tcsseqnum</td>
<td>TC Modification + Impersonation</td>
<td>OLSR</td>
<td>-a, -IP [-MAC]</td>
</tr>
<tr>
<td>dcdpclient</td>
<td>Customer attack</td>
<td>DCDP</td>
<td>a</td>
</tr>
<tr>
<td>dcdpserver</td>
<td>Server attack</td>
<td>DCDP</td>
<td>a</td>
</tr>
</tbody>
</table>

The program also works as a protocol analyzer for OLSR and DCDP protocols, showing messages received and sent by the software in a text format understandable to humans. Option `-v` enables the verbal exhibition of messages sent and received by the program. Option `-p` prevents the malicious hearing of messages.

With option `-i` only one network interface shall be specified, that is, the simultaneously generate attacks in more than one interface, more than one program instance must be created.

Attack messages may be forwarded/generated with or without a MAE. Option `-cert` controls this process. In case this option is not present, attacks are generated without the MAE. When option `-cert` is specified, all generated messages are signed. This option receives as parameter a PEM file containing, in addition to the sender certificate, the certificate of the certification agency signing it (also used to check the MAE of received messages) and the private key associated to the certificate, for message signature. No command options are mandatory and if command is activated without option specifications, it will work solely as network analyzer, showing received and sent messages, in real time, on the screen.

The first experiment carried out consisted of running OLSR and DCDP protocols as specified, that is, without MAE protection. In this case the `uolsrd` must be used without options `-cert` and `-share`. All attacks shown in Table 5-1 were carried out in this scenario and the results show the malfunctioning of protocols caused by the attacks. The effects are basically DoS ones, but other effects can be noticed as previously exposed. This experiment offers the opportunity to check, in practical terms, the presence of inferred vulnerabilities. As
For the OLSR protocol, the used implementation (uolsrd) has a graphic interface which enables the easy verification of routing malfunctions caused by the attacks (i.e. link symmetry breaking, changes in MPR sets). With regards to DCDP, for both client and server attacks, the effect of attacks when the nodes try to self-configure in the vicinity is that the opponent cannot complete the process.

6.4. MAE AND L-CERT

For implementing encryption and digital certification manipulation functions in the MAE implementation, the crypto library of OpenSSL\textsuperscript{44} project is used. MAE is implemented in a pre-compiled library (mae) having functions to verify and generate digital signatures (DS objects) and to manipulate digital certificates (i.e. obtaining of public keys, signature verification, etc.). This library contains also the specific functions for generation and verification of MAE for OLSR protocols (mae-olsr) and DCDP (mae-dcdp). Extension of MAE to other services may be done adding new separately compiled modules, then linked to the mae library. Thus MAE implementation happens as an API. This API is used to insert the generation and verification of received messages in the OLSR and DCDP protocol implementations, that is, in the uolsrd program.

The L-Cert also uses the crypto library of project OpenSSL and, as the self-configuration service, is integrated to the routing daemon (uolsrd). This integration is justified by the intention to distribute certificates throughout the network in integration with the routing protocol, which will distribute the routes to all nodes. Also, improvements to the routing protocols for the carrying out of flooding (i.e. MPRs in the OLSR case), are also used by the certification service.

The authentication and certification policy (Figure 4-6) to be used must be specified in a daemon uolsrd configuration file called "policy.conf". This file must be in the same folder from where the command to start olsrd is run. Alternatively, the option -policy <configuration file> informing as parameter the path and name of file to be used.

As node (and certificate) identifiers an MD5 summary of the digital signature of certificate, done by the issuer, is used. This identifier is used in the processes of coalition formation and as index in the cache of valid certificates and in CRL. It is also used as the single identifier of nodes, from the security services point of view.

\textsuperscript{44} Available in http://openssl.org (Last visited August 2004).
To start the distributed certification service, a negotiating software (dealer), in C language, is used. This program receives as parameters the size of the coalition \( K \) and the certificate requests for the first nodes (files with ".csr" extension). After the ACD private key is generated, the dealer signs all certificate requests and generates, for each new certificate signed, a part of the private key. The results of this operation are recorded in PEM files, having the same names of the certificate request files, but with extensions ".pem" and ".share.pem" for the new certificate and their part of the ACD private key, respectively.

Command for activating the dealer is.

```bash
./dealer <coalition size> <.csr files>
```

### 6.4.1. Experiment Parameters

For carrying out tests in the experimental Manet, the following parameters are considered (policy):

- **Size of certification key (ACD):** This parameter influences the size of certificates, the computation required by certification services and the verification of validity of CERT (certified) objects present in a MAE, when they are not present in the cache of valid certificates. On the other hand, it represents the certification security of all the system. The experiments consider \( KIA_{CD} \) with 1024, 2048 or 4096 bits. The use of RSA keys with 512 bits or less is not considered safe. For usage over long periods of time (i.e. 30 days or more), the use of keys with 1024 bits is not recommended.

- **Size of node keys:** This parameter influences the size of certificates, the size of DS objects (digital signature) present in the MAE and the computerization required to generate digital signatures. On the other hand, this parameter represents the security of digital signatures that authenticate messages. For networks that are gone and are dismantled in a short time (i.e. 24 hours), usage of RSA keys with 512 bits may be tolerated. In all other cases bigger keys must be used. The experiments were carried out with keys from 512 to 4096 bits.

- **Size of Coalition \( K \):** This parameter represents the commitment between service availability and system security. It influences the computational (and network) cost of distributed certification services. As a rule, this parameter must be comparable to the average quantity of neighbors of a network hop. Considering that, the experiments consider \( K = 3 \).

- **Information on the certificates:** To keep it simple and standardized, certificates
complying with standard X.509v3 are used. To reduce certificate size, as they are large in MAE of all messages in case a proactive certificate distribution scheme is used, it carries minimal information. The certificates contain: name (distinguished name) of certificate owner, name (distinguished name) of issuer (i.e. the AC), public key owner, issuing timestamp, expiration timestamp and issuer digital signature. Certificates are transmitted over the MAE and over messages of protocol L-Cert encoded in ASN.1, standard X.509.

- **Cache policy and local certificates:** As the OLSR protocol is a proactive routing protocol, a proactive distribution of certificates policy is adopted, and they are included in all messages. That is, all the MAE has, in addition to the mandatory object (DS), a CERT object. The cache size is not limited and valid certificates are only removed from the cache in case of certificate expiration or revoking.

- **CRL policy:** A reactive CRL synchronization strategy is used, in which nodes coming to the network must explicitly request the current CRL version to its neighbors. In case a new certificate is issued, the node with higher ID in the coalition automatically sends the updating, when a partial certificate is issued in favor of the new certified node.

- **Certificate issuing policy:** Certificates are issued according to a policy of manual verification of issuer identity. Therefore, when a request for the issuing of a new certificate is received, it is asked to the user whether he wishes or not to comply with it. The user must then check the requester identity, using a procedure established in the security policy. Certificates are always valid for 3600 seconds (01 hour).

- **Certificate renewal policy:** Certificates are renewed automatically. A request for certificate renewal whose previous certificate (still valid) has not been revoked and against which no malicious activity has been locally detected (i.e. via IDS), is automatically signed. In other cases, the request is treated as a request for a new certificate. Certificates are always renewed for additional 3600 seconds (01 hour).

- **Policy for issuing parts of ACD private key:** A manual service policy is used, similar to the policy for certificate issuing.

- **Policy for renewing parts of ACD private key:** A 24-hour period is adopted for the proactive renewal of the private keys.

As for the network topology and mobility, a random unit graph type simulation model is used, in a Manet formed by a given number of nodes (10, 50, 100 and 1000), which are evenly distributed in a bi-dimensional 250m x 250m area. Maximum movement speed is
established as 5m/s, to consider movements resulting from humans walking or vehicles running at low speeds.

6.4.2. Evaluation of Communication Overhead

The overhead in the network caused by the addition of MAE to the OLSR routing protocol messages may be compared to the OLSR messages control overhead without any authentication. To compare the two situations, it is necessary to identify the quantity of bytes of the MAE for different configurations of the certification policy, as the MAE has a fixed size for each message, as well as the quantity of bytes contained in OLSR messages for Manets in different sizes. The ratio between the two values is the additional overhead, relative to the control overhead of a network with OLSR routing.

Table 6-2 shows the sizes, in bytes, for the elements that form a MAE having a DS object (digital signature) and a CERT object (certificate of the message issuer). The table also shows, in the last but one column, the size of a MAE containing only one DS object. The objective of this column is to evidence that the largest contribution to the MAE size is clearly due to the CERT object. This, in turn, has its size influenced both by the size of the certification key, as it now carries the ACD signature, and the size of the certificate key, considering it carries a public key.

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>CA Key (bits)</th>
<th>Header MAE (bytes)</th>
<th>Header DS Obj. (bytes)</th>
<th>DS Object (bytes)</th>
<th>Header CERT Obj. (bytes)</th>
<th>CERT Object (bytes)</th>
<th>MAE (w/oCERT) (bytes)</th>
<th>MAE (w/CERT) (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>4096</td>
<td>4</td>
<td>4</td>
<td>64</td>
<td>4</td>
<td>724</td>
<td>72</td>
<td>800</td>
</tr>
<tr>
<td>1024</td>
<td>4096</td>
<td>4</td>
<td>4</td>
<td>128</td>
<td>4</td>
<td>788</td>
<td>136</td>
<td>928</td>
</tr>
<tr>
<td>2048</td>
<td>4096</td>
<td>4</td>
<td>4</td>
<td>256</td>
<td>4</td>
<td>916</td>
<td>264</td>
<td>1184</td>
</tr>
<tr>
<td>512</td>
<td>2048</td>
<td>4</td>
<td>4</td>
<td>64</td>
<td>4</td>
<td>468</td>
<td>72</td>
<td>544</td>
</tr>
<tr>
<td>1024</td>
<td>2048</td>
<td>4</td>
<td>4</td>
<td>128</td>
<td>4</td>
<td>532</td>
<td>136</td>
<td>672</td>
</tr>
<tr>
<td>2048</td>
<td>2048</td>
<td>4</td>
<td>4</td>
<td>256</td>
<td>4</td>
<td>660</td>
<td>264</td>
<td>928</td>
</tr>
<tr>
<td>512</td>
<td>1024</td>
<td>4</td>
<td>4</td>
<td>64</td>
<td>4</td>
<td>340</td>
<td>72</td>
<td>416</td>
</tr>
<tr>
<td>1024</td>
<td>1024</td>
<td>4</td>
<td>4</td>
<td>128</td>
<td>4</td>
<td>404</td>
<td>136</td>
<td>544</td>
</tr>
<tr>
<td>2048</td>
<td>1024</td>
<td>4</td>
<td>4</td>
<td>256</td>
<td>4</td>
<td>532</td>
<td>264</td>
<td>800</td>
</tr>
</tbody>
</table>

Evaluation of OLSR protocol control overhead was performed based on the simulation results. Table 6-3 indicates the average quantity of IP addresses announced in HELLO and TC messages for different node densities. In the case of HELLO messages, the value corresponds to the average number of neighbors that the nodes have at the time of message generation. In the case of TC messages, the value refers to the MS number.
Table 6-3 – Average number of addresses announced in HELLO messages (neighbors) and TC (MS)

<table>
<thead>
<tr>
<th>Number of Nodes in the Network</th>
<th>X (m)</th>
<th>Y (m)</th>
<th>Reach of Wireless Link (m)</th>
<th>HELLO</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>250</td>
<td>250</td>
<td>50</td>
<td>1.26</td>
<td>4.81</td>
</tr>
<tr>
<td>50</td>
<td>250</td>
<td>250</td>
<td>50</td>
<td>6.28</td>
<td>8.23</td>
</tr>
<tr>
<td>100</td>
<td>250</td>
<td>250</td>
<td>50</td>
<td>12.57</td>
<td>10.37</td>
</tr>
<tr>
<td>1000</td>
<td>250</td>
<td>250</td>
<td>50</td>
<td>125.7</td>
<td>22.35</td>
</tr>
</tbody>
</table>

Message size varies according to size of IP address used. For IPv4, this size is equal to 04 bytes and for IPv6 the size is 16 bytes per address. Thus, Table 6-4 shows the average size (theoretical) of HELLO messages in each node density in the considered space.

Table 6-4 – Average size, in bytes, of HELLO message (no MAE)

<table>
<thead>
<tr>
<th>Number of nodes in the network</th>
<th>10</th>
<th>50</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neighborhood average size</td>
<td>1.26</td>
<td>6.28</td>
<td>12.57</td>
<td>125.66</td>
</tr>
<tr>
<td>Message average size (IPv4)</td>
<td>33.03</td>
<td>53.13</td>
<td>78.27</td>
<td>530.65</td>
</tr>
<tr>
<td>Message average size (IPv6)</td>
<td>48.11</td>
<td>128.53</td>
<td>229.06</td>
<td>2038.62</td>
</tr>
</tbody>
</table>

Overhead caused in the network by the addition of an MAE to a OLSR message corresponds to the size of the MAE, shown in the last column of Table 6-2. This overhead represents an increase in message size that varies between 400 and 1200 bytes. The average size of OLSR messages, without MAE, however, varies between 30 and 500 bytes, in the case of IPv4, and between 58 and 2058 bytes, in the case of IPv6 (Table 6-4). That represents an increase of up to 40 times in message size. However, it must be considered that before transmitting a message in a wireless network in an ad hoc manner, the arbitration of the shared media is necessary. This process is not fully efficient, so the band available in a link does not correspond to the nominal band. For example, in the case of IEEE 802.11b technology, used in the experiments described herein, the CSMA/CA protocol is used (see protocol description in Attachment I), and its efficiency may be below 50%, for links in a neighborhood with more than five nodes running the protocol in ad hoc mode. Therefore, the overhead represented by the increase in a message size, which does not cause breaking of the message in more than a package, does not represent big losses in the occupation of the transmission environment. Breaking of a message (HELLO or TC) must happen whenever the message is over the MTU (maximum transfer unit) of the network. As an example, let’s consider once more the case of IEEE 802.11b, ad hoc manner, where a data link frame may

---

45 TC messages are smaller than HELLO messages, as they announce only part of the neighborhood of a node, that is, the neighbors choosing this node as MPR.
carry up to 2,304 bytes. In IPv4, not considering IP headers (20 bytes), UDP (8 bytes) and OLSR package (4 bytes) the network MTU is 2272 bytes. In the case of IPv6, the IP header size depends on the present options, but, usually, it is not over 48 bytes. Therefore, the network MTU in IPv6 is around 2260 bytes. To each HELLO message, a MAE must be added, with CERT objects (in the case of proactive distribution of certificates). Table 6-5 and Table 6-6 show the average sizes of HELLO messages for IPv4 and IPv6 addressing, respectively, containing a MAE with CERT object per message, for each node density considered, depending on the sizes of ACD and certificate keys.

### Table 6-5 – Average size (bytes) of HELLO messages – IPv4 (MAE with CERT object)

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>CA Key (bits)</th>
<th>Number of nodes in the network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>512</td>
<td>4096</td>
<td>833</td>
</tr>
<tr>
<td>1024</td>
<td>4096</td>
<td>961</td>
</tr>
<tr>
<td>2048</td>
<td>4096</td>
<td>1217</td>
</tr>
<tr>
<td>512</td>
<td>2048</td>
<td>577</td>
</tr>
<tr>
<td>1024</td>
<td>2048</td>
<td>705</td>
</tr>
<tr>
<td>2048</td>
<td>2048</td>
<td>961</td>
</tr>
<tr>
<td>512</td>
<td>1024</td>
<td>449</td>
</tr>
<tr>
<td>1024</td>
<td>1024</td>
<td>577</td>
</tr>
<tr>
<td>2048</td>
<td>1024</td>
<td>833</td>
</tr>
</tbody>
</table>

### Table 6-6 – Average size (bytes) of HELLO messages – IPv6 (MAE with CERT object)

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>CA Key (bits)</th>
<th>Number of nodes in the network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>512</td>
<td>4096</td>
<td>848</td>
</tr>
<tr>
<td>1024</td>
<td>4096</td>
<td>976</td>
</tr>
<tr>
<td>2048</td>
<td>4096</td>
<td>1232</td>
</tr>
<tr>
<td>512</td>
<td>2048</td>
<td>592</td>
</tr>
<tr>
<td>1024</td>
<td>2048</td>
<td>720</td>
</tr>
<tr>
<td>2048</td>
<td>2048</td>
<td>976</td>
</tr>
<tr>
<td>512</td>
<td>1024</td>
<td>464</td>
</tr>
<tr>
<td>1024</td>
<td>1024</td>
<td>592</td>
</tr>
<tr>
<td>2048</td>
<td>1024</td>
<td>848</td>
</tr>
</tbody>
</table>

In the case of IPv4 addressing (Table 6-5), the size of HELLO messages added from a MAE usually occurs only in the case of high node densities (1000 nodes in a 250m x 250m area) and for an ADC key of 4096 bits and a certificate key of 2048 bits. Thus, increase in message size is tolerable in this case. In case of IPv6 (Table 6-6), the same analysis remains valid even in networks with moderate node density (100 nodes in a 250m x 250m area). In the case of networks with high node density, the HELLO message, without MAE, is already over
the network MTU limit, and must be broken. In such cases, there are two possible solutions: add a MAE for each part of the message or use a single MAE for all messages. In the first option, each part of the message may be processed as they are received. However, the number of frames needed to send all HELLO message parts is 04, for most considered cases, and that may be impractical. In the second option the problem is loosing one part of the message and having to throw out the whole of it, as signature verification of incomplete messages is not possible.

In any case, improvements to the certificate distribution mechanism may be considered, as the certificates are valid for long periods in relation to the HELLO interval. Thus, the local cache of certificates may be used to prevent the distribution of a certificate in each sent message. A possible improvement consists of loading CERT objects to the MAE only when new nodes are detected in the neighborhood (i.e. a node whose certificate is not in the local certificate cache). However, this alternative causes a lack of certificate for the first HELLO message coming from a node not having the certificate in the cache of its neighbors.

Table 6-7 and Table 6-8 show the average sizes of messages to IPv4 and IPv6, respectively, containing a MAE without CERT object. Evaluating the data of this table, we see that the addition of this MAE does not cause the generation of additional frames in the network, regardless of the network configuration considered.

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>CA Key (bits)</th>
<th>Number of nodes in the network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>512</td>
<td>4096</td>
<td>105</td>
</tr>
<tr>
<td>1024</td>
<td>4096</td>
<td>169</td>
</tr>
<tr>
<td>2048</td>
<td>4096</td>
<td>297</td>
</tr>
<tr>
<td>512</td>
<td>2048</td>
<td>105</td>
</tr>
<tr>
<td>1024</td>
<td>2048</td>
<td>169</td>
</tr>
<tr>
<td>2048</td>
<td>2048</td>
<td>297</td>
</tr>
<tr>
<td>512</td>
<td>1024</td>
<td>105</td>
</tr>
<tr>
<td>1024</td>
<td>1024</td>
<td>169</td>
</tr>
<tr>
<td>2048</td>
<td>1024</td>
<td>297</td>
</tr>
</tbody>
</table>
Table 6-8 – Average size (bytes) of HELLO messages – IPv6 (MAE without CERT object)

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>CA Key (bits)</th>
<th>Number of nodes in the network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>512</td>
<td>4096</td>
<td>120</td>
</tr>
<tr>
<td>1024</td>
<td>4096</td>
<td>184</td>
</tr>
<tr>
<td>2048</td>
<td>4096</td>
<td>312</td>
</tr>
<tr>
<td>512</td>
<td>2048</td>
<td>120</td>
</tr>
<tr>
<td>1024</td>
<td>2048</td>
<td>184</td>
</tr>
<tr>
<td>2048</td>
<td>1024</td>
<td>312</td>
</tr>
<tr>
<td>512</td>
<td>1024</td>
<td>120</td>
</tr>
<tr>
<td>1024</td>
<td>1024</td>
<td>184</td>
</tr>
<tr>
<td>2048</td>
<td>1024</td>
<td>312</td>
</tr>
</tbody>
</table>

Regarding the self-configuration protocol, this service generates control traffic only at certain times, when a new node comes to the network. Also, messages of the considered protocol (DCDP) are not over 70 bytes (without MAE). Therefore, inclusion of MAE in messages of this protocol is fully tolerable in terms of network overhead.

As for the communication overhead generated by distributed certification services, Table 6-9 shows the number of necessary broadcast and unicast communications for each of the basic certification services. An analysis of Table 6-9 indicates the communication overhead for obtaining a certification service increases with \( K \) in a linear fashion. However, as with the self-configuration protocol, the services generate traffic when a new node comes to the network and requests a certificate and a part of the private key, or through the periodic processes of certificate renewal and updating of parts of the ACD private key. The frequency of such processes is considerably lower than the frequency of generation of routing protocol control messages. Therefore, the communication overhead generated by the distributed certification protocol is much lower than the control overhead generated by OLSR, even in cases where no authentication is used. Also, with a convenient \( K \) choice, the overhead will be located in the neighborhood of the node requesting the authentication services, giving scalability to the projected services.

Table 6-9 – L-Cert Communication Overhead

<table>
<thead>
<tr>
<th>Certification Service</th>
<th>Number of broadcasts or floodings</th>
<th>Number of unicastss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certificate issuing or renewing</td>
<td>2</td>
<td>2 * ( K )</td>
</tr>
<tr>
<td>Issuing of parts of private key</td>
<td>3</td>
<td>3 * ( K )</td>
</tr>
<tr>
<td>Certificate revoking*</td>
<td>1</td>
<td>2 * ( K )</td>
</tr>
</tbody>
</table>

* does not include the alarm co-relation, as this is the L-IDS overhead.
6.4.3. Evaluation of Computational Overhead

The first measurement for computational overhead evaluation caused by MAE is the time required for creation and verification of the MAE of all generated and received messages, respectively, during a HELLO interval (i.e. average time between sending of two HELLO messages). Obviously, computational platforms with different performances show different absolute values for this measurement. Also, the average number of TC and HELLO messages received during this interval varies according to the network topology dynamics. As a matter of fact, the number of HELLO messages received for processing increases and the neighborhood increases (i.e. arrival of a new neighbor). Therefore, the computing time required for processing generated and received MAE is indirectly estimated. As the generation of an RSA signature and the checking of an RSA signature are clearly the most computer-intensive operations in the process of generating a new message and checking the MAE of received messages, the number of such operations in message issuing and receiving, in typical Manet scenarios, is estimated.

Table 6-10 shows the results of the simulation, indicating average quantity of HELLO and TC messages sent by a node, during a HELLO interval, considering TC interval equal to three times the HELLO interval. In the same way, the table shows the average number of HELLO and TC messages received in the same interval. It is important to have in mind that HELLO messages are generated only in the neighborhood of a hop and TC messages are broadcast in the Manet by flooding.

<table>
<thead>
<tr>
<th>Number of nodes in the network</th>
<th>10</th>
<th>50</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average number of sent messages (HELLO and TC)</td>
<td>1.16</td>
<td>1.05</td>
<td>1.03</td>
<td>1.01</td>
</tr>
<tr>
<td>Average number of sent messages (HELLO and TC)</td>
<td>6.07</td>
<td>14.52</td>
<td>22.94</td>
<td>148.01</td>
</tr>
</tbody>
</table>

Considering each MAE must have a DS object (digital signature) and CERT object (certificate), the following operations are considered computing overhead:

- 01 generation of RSA digital signature with the certificate public key, for each sent message;
- 01 verification of RSA digital signature with certificate public key for each received message.

The verification of validity of message author’s certificate, involving 01 verification of RSA digital signature with ACD public key, may be disconsidered as, supposing the
existence of one cache of valid certificates, all nodes learn each other’s certificates stationarily.

Table 6-11 and Table 6-12 have overhead average time for generation and verification of RSA signatures, respectively, for different sizes of certificate key. The values were evaluated in three platforms with different computational performance: Platform 1 – laptop computer with Pentium IV 1.7GHz processor, 256Mbytes RAM; Platform 2 – laptop computer with Pentium III 50MHz, 256Mbytes RAM processor; Platform 3 – PalmTop Compaq iPAQ with Intel StrongARM 206MHz processor, 64Mbytes RAM.

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>Platform 1</th>
<th>Platform 2</th>
<th>Platform 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.91</td>
<td>2.31</td>
<td>32.8</td>
</tr>
<tr>
<td>1024</td>
<td>4.36</td>
<td>9.2</td>
<td>150</td>
</tr>
<tr>
<td>2048</td>
<td>20.2</td>
<td>68.1</td>
<td>850</td>
</tr>
</tbody>
</table>

To calculate total overhead (processing time), for each platform the time spent in signature generation is multiplied by the average number of generated messages and the result is added to the product of time spent in signature verification and the average time of certificate verification. The results can be normalized within the HELLO interval. Table 6-13, Table 6-14 and Table 6-15 show the total overhead for a 2s HELLO interval.

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>Platform 1</th>
<th>Platform 2</th>
<th>Platform 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.165</td>
<td>0.293</td>
<td>3.51</td>
</tr>
<tr>
<td>1024</td>
<td>0.328</td>
<td>0.701</td>
<td>11.5</td>
</tr>
<tr>
<td>2048</td>
<td>0.928</td>
<td>2.01</td>
<td>29.90</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>Platform 1</th>
<th>Platform 2</th>
<th>Platform 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.103%</td>
<td>0.168%</td>
<td>0.236%</td>
</tr>
<tr>
<td>1024</td>
<td>0.352%</td>
<td>0.467%</td>
<td>0.601%</td>
</tr>
<tr>
<td>2048</td>
<td>1.801%</td>
<td>2.049%</td>
<td>2.414%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>Platform 1</th>
<th>Platform 2</th>
<th>Platform 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.223%</td>
<td>0.334%</td>
<td>0.455%</td>
</tr>
<tr>
<td>1024</td>
<td>0.746%</td>
<td>0.992%</td>
<td>1.278%</td>
</tr>
<tr>
<td>2048</td>
<td>4.560%</td>
<td>5.035%</td>
<td>5.813%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>Platform 1</th>
<th>Platform 2</th>
<th>Platform 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.223%</td>
<td>0.334%</td>
<td>0.455%</td>
</tr>
<tr>
<td>1024</td>
<td>0.746%</td>
<td>0.992%</td>
<td>1.278%</td>
</tr>
<tr>
<td>2048</td>
<td>4.560%</td>
<td>5.035%</td>
<td>5.813%</td>
</tr>
</tbody>
</table>
sent + received (HELLO and TC) messages – Platform 3

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>10</th>
<th>50</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>2.968%</td>
<td>4.229%</td>
<td>5.715%</td>
<td>27.632%</td>
</tr>
<tr>
<td>1024</td>
<td>12.190%</td>
<td>16.224%</td>
<td>20.916%</td>
<td>92.681%</td>
</tr>
<tr>
<td>2048</td>
<td>58.375%</td>
<td>66.332%</td>
<td>78.070%</td>
<td>&gt; 100%</td>
</tr>
</tbody>
</table>

An analysis of values in the tables shows the high computerization required by the use of asymmetric encryption. Even so, in the case of platform 1 (Pentium IV 1.8GHz, Table 6-13), the total computerization overhead is not over 2.6% for almost all situations evaluated, except for the extreme case of a high node density and usage of a 2048 bits certificate key, when the computational capacity is a little higher than 8%. However, in platforms with lower computing capacity, as in the case of platform 3 (Table 6-15), usage of proposed mechanisms is only feasible for 512 bits certificate key and in networks with moderate node density. Thus, as suggested in Table 6-13, with the fast increase of computational capacity of processors designed for mobile computing platforms, even the use of asymmetric encryption may be tolerated. But, as the computing cost is still high, it is necessary to find alternatives to the RSA encryption system (i.e. elliptic curve encryption) or adopt a symmetric encryption system, giving up the possibility of an authentication with non-refusal – limiting the possibilities of response to model intrusion.

The computational overhead for MAE calculation and verification in the case of the self-configuration protocol is minimum, compared to the overhead resulting from the routing protocol, once the DCDP generates less control traffic than the OLSR.

For evaluation of the computational cost of basic distributed certification services, the process complexity in the requesting node and in the coalition nodes is evaluated, as shown in Table 6-16. It is important to note that the “bsic” operation is different in each case. Each of the operations requires a different time for the calculation, as shown in Table 6-17.
Table 6-16 – Basic operations and L-Cert computational complexity

<table>
<thead>
<tr>
<th>Service</th>
<th>Basic operation</th>
<th>Complexity</th>
<th>Requesting Node</th>
<th>Coalition Node*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certificate verification</td>
<td>RSA signature verification (ACD public key)</td>
<td>O(1)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Certificate issuing, renewal and revoking</td>
<td>RSA signature verification (ACD public key)</td>
<td>O(K)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RSA signature generation (part of the ACD public key)**</td>
<td>O(1)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Multiplication in the Lagrange interpolation (Eq. 4-5)</td>
<td>O(K)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Issuing of parts of private key and updating of parts of private key</td>
<td>Generation of RSA signature (parte da chave privada da ACD)**</td>
<td>-</td>
<td>O(1)</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RSA signature verification (public key certificate)</td>
<td>-</td>
<td>O(K-1)***</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RSA signature generation (private key certificate)</td>
<td>-</td>
<td>O(K-1)***</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>sum in the computerization of part of private key (Eq. 4-10)</td>
<td>O(K)</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

*Computational cost per node. To evaluate the total computational cost in all nodes, the value must be multiplied by K. However, as the computational cost is measured in processing time, this time in the coalition nodes is considered only once, as this computerization is performed in parallel in K nodes.

**Computational cost of an RSA signature with a part of the private key or the encryption with public RSA key is higher than the computational cost of a standard RSA signature, as in the last case it is possible to use optimizations of the computerization algorithm due to knowledge of module factoring into its prime factors.

***Only one of the coalition nodes performs K-1 operations. Each of the other coalition nodes performs a lower number of operations.

Table 6-17 – Average calculation time (ms) of L-Cert basic operations

<table>
<thead>
<tr>
<th>Basic operation</th>
<th>Size of key</th>
<th>Platform 1</th>
<th>Platform 2</th>
<th>Platform 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA signature verification (ACD public key)</td>
<td>1024</td>
<td>0.328</td>
<td>0.701</td>
<td>11.5</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>0.928</td>
<td>2.01</td>
<td>29.90</td>
</tr>
<tr>
<td></td>
<td>4096</td>
<td>2.91</td>
<td>7.00</td>
<td>92.2</td>
</tr>
<tr>
<td>RSA signature generation (part of ACD private key)</td>
<td>1024</td>
<td>4.36</td>
<td>9.2</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>26.2</td>
<td>68.1</td>
<td>850</td>
</tr>
<tr>
<td></td>
<td>4096</td>
<td>81.5</td>
<td>197</td>
<td>2670</td>
</tr>
<tr>
<td>RSA signature verification (certificate public key)</td>
<td>512</td>
<td>0.165</td>
<td>0.293</td>
<td>3.51</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>0.328</td>
<td>0.701</td>
<td>11.5</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>0.928</td>
<td>2.01</td>
<td>29.90</td>
</tr>
<tr>
<td>RSA signature generation (certificate private key)</td>
<td>512</td>
<td>0.91</td>
<td>2.31</td>
<td>32.8</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>4.36</td>
<td>9.2</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>26.2</td>
<td>68.1</td>
<td>850</td>
</tr>
<tr>
<td>Multiplication in the Lagrange interpolation</td>
<td>1024</td>
<td>1.65</td>
<td>3.26</td>
<td>67.1</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>2.02</td>
<td>5.01</td>
<td>91.1</td>
</tr>
<tr>
<td></td>
<td>4096</td>
<td>6.5</td>
<td>14.7</td>
<td>109.9</td>
</tr>
<tr>
<td>Sum in the computerization of part of private key</td>
<td>1024</td>
<td>0.000023</td>
<td>0.000061</td>
<td>0.00189</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>0.000022</td>
<td>0.000063</td>
<td>0.00208</td>
</tr>
<tr>
<td></td>
<td>4096</td>
<td>0.000029</td>
<td>0.000081</td>
<td>0.00305</td>
</tr>
</tbody>
</table>
6.4.4. L-Cert Performance Evaluation

Total computerization time for each of the L-Cert basic services are shown in Table 6-18. The data are collected in the experimental Manet (ACD key with 4096 bits and certificate key with 1024) and refer to a K = 3 value, running the algorithms in nodes with platform 1 and 2 type hardware. The times are measured between sending of message that confirms coalition formation (step 3, Figure 3-2) and conclusion of computerization of new certificate or part of private key. The time for coalition formation is not considered as, during this phase, the certification policy is applied, leading to an undetermined time. As certification services are performed only upon new additions to the network or when nodes must renew the certificates, which happens over much longer periods than the ones in Table 6-18, these periods are considered acceptable.

<table>
<thead>
<tr>
<th>Service</th>
<th>Average time for obtaining service (s)</th>
<th>Number of broadcast messages</th>
<th>Number of unique messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certificate issuing, renewal and revoking</td>
<td>0.91</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>Issuing of parts of private key and updating of parts of private key</td>
<td>1.23</td>
<td>3</td>
<td>9</td>
</tr>
</tbody>
</table>

Considering the times used in the calculation of operations of each of these services, it is possible to see that time for complete process is considerably higher than the sum of times required for all necessary switchings. Therefore, a significant part of the time required to obtain service is due to communications.

One last aspect to be analyzed is the effect of mobility in the availability of distributed certification services. In general, the nodes chosen to be part of the coalition must remain in the neighborhood of the requesting node, at least until the receiving of the message (broadcast) confirming the coalition formation. In case one of the nodes moves before receiving this message, it will not know it was chosen as coalition member, and will not respond to the request, causing all the calculation performed by the other server nodes to be lost. To prevent that, the requesting node may wait for responses of the coalition members for a pre-established period of time (T_timeout) after which it may decide to send a unicast message to each node which is part of the coalition, but have not responded to the message informing their formation. That enables such nodes to respond to the request, even if they are more than one hop away from requesting node. However, in case one of the coalition nodes fails or becomes unavailable, the process must be restarted with the formation of a new coalition.
6.4.5. Local CRL and Valid Certificate Cache Evaluation

As discussed in the previous sections, the use of certificate cache is essential for the operation of preventive security services (MAE and L-Cert). In terms of the valid certificate cache, it is interesting to check the quantity of memory required to store the certificate collected by the adopted distribution process. Table 6-19 shows the quantity of memory required when certificates of all Manet nodes (simulation) are loaded in the cache.

<table>
<thead>
<tr>
<th>Certificate Key (bits)</th>
<th>CA Key (bits)</th>
<th>Number of nodes in the network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>512</td>
<td>4096</td>
<td>7.1</td>
</tr>
<tr>
<td>1024</td>
<td>4096</td>
<td>7.7</td>
</tr>
<tr>
<td>2048</td>
<td>4096</td>
<td>9.0</td>
</tr>
<tr>
<td>512</td>
<td>2048</td>
<td>7.1</td>
</tr>
<tr>
<td>1024</td>
<td>2048</td>
<td>7.7</td>
</tr>
<tr>
<td>2048</td>
<td>2048</td>
<td>9.0</td>
</tr>
<tr>
<td>512</td>
<td>1024</td>
<td>7.1</td>
</tr>
<tr>
<td>1024</td>
<td>1024</td>
<td>7.7</td>
</tr>
<tr>
<td>2048</td>
<td>1024</td>
<td>9.0</td>
</tr>
</tbody>
</table>

As for the local CRL, it is reduced in size, considering it may contain at a given time, at most, $K-1$ counter-certificates. Otherwise, the system has been broken by the existence of, at least, $K$ affected nodes in the network.

6.5. L-IDS: DETECTION DUE TO INCORRECT USE

In the developed implementation, intrusion detection is based on information collected from the MIB, kept locally in each Manet node per SNMP agents (OLSRAgent and ucd-snmp). Usage of this type of information source is twice justified. First, there is an extensive set of standardized MIBs, covering a wide variety of network applications and services, such as in the cases of MIB-II and MIB RMON. These standardized MIBs are implemented in SNMP agents developed for different platforms, giving the L-IDS a satisfactory portability. Second, the MIB information is formed by data describing entities from different system architecture levels. Thus, it is possible to monitor, at the same time, the network and system levels, and even some applications.

The data gathering module consists of software capable of performing SNMP consultations (i.e. SNMP GET, SNMP GETNEXT, etc.) and receiving traps (SNMP TRAP). The resulting data correspond to the MIB variant values at the time of consultation or sending of trap.
The rules for data abstraction are run in specialized classes of the `eventAbstractorRule` (event abstraction rule), and must implement the `ruleProcessing` interface (rule processing), containing a `processRule()` function that receives as parameter the data gathered/collected by the `data collecting` Module. Thus, abstraction rules may be written with all flexibility offered by language Java2, without breaking the software modular structure. These specialized classes may also be compiled in execution time, upon request. To detect the attacks described in 5.3.1, the following abstraction rules are implemented, shown in Table 6-20:

The **IDS core** for implementation of intrusion detection per incorrect usage consists of a machine of finite states capable of running and keeping multiple DEF instances, one for each attack being monitored.

<table>
<thead>
<tr>
<th>Abstraction rule</th>
<th>Raw Data</th>
<th>Consultation</th>
<th>Generated Events</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>trapOLSRNeighborLinkStatusChanged</code></td>
<td><code>olsrNeighborEntry</code></td>
<td>-</td>
<td><code>NHOP_E1, NHOP_E2</code></td>
<td>Implements processing of messages indicating change of state of link with a neighbor.</td>
</tr>
<tr>
<td><code>trapOLSRNeighborMPRSetChanged</code></td>
<td><code>olsrNeighborEntry</code>, <code>olsr2hopNeighborTable</code></td>
<td>-</td>
<td><code>N+1HOP_E1</code></td>
<td>Implements processing of messages indicating change in MPRs set.</td>
</tr>
<tr>
<td><code>queryOLSRNeighborTable</code></td>
<td><code>olsrNeighborTable</code></td>
<td><code>N+1HOP_C1</code></td>
<td><code>N+1HOP_E2, N+1HOP_E3, N+1HOP_E4</code></td>
<td>Checks whether a node is announcing the local node as its neighbor without that being the case.</td>
</tr>
<tr>
<td><code>queryTCPConnTable</code></td>
<td><code>tcpConnTable</code></td>
<td><code>STEPSTONE_PC1, STEPSTONE_C1</code></td>
<td><code>STEPSTONE_E1, STEPSTONE_E2</code></td>
<td>Checks the existence of telnet connections in chain, at a given node.</td>
</tr>
</tbody>
</table>

The **alarm managers** implement a simple co-relation, checking whether the different alarms received were generated by different nodes. This process also stores all accusations (alarms) signed locally or received from other nodes. The **communicator** module implements the communication protocol (SMUX) for sending information to the L-Cert (uolsrd). Through this module group alarms are sent with the addresses of all the nodes that have detected an attack coming from a single opponent. This module also serves for the L-Cert to consult the existence of accusations generated in the L-IDS against a node for which a counter-certificate signature is being requested.

The **agent platform** module has an `aglet` server that may generate and receive mobile agents, destroy them, or forward them to other nodes. The creation of mobile agents is always
requested by the distribution manager module, whenever a consultation, event, detection status or alarm must be run in a remote node. Similarly, when the agent platform receives an agent coming from another node, the message to be run locally, which is passed to the distribution manager for local processing.

Figure 5-1 illustrates the implemented L-IDS.

![Figure 6-1 – Implemented L-IDS Modular Architecture](image)

### 6.5.1. Performance Considerations

Compared to the routing protocol authentication service, the L-IDS takes up much less memory and processing time. While the uolsrd (daemon OLSR, daemon DCDP and L-Cert) process CPU usage varies between 0.1% and 2% in a computer with platform 1 type hardware (Pentium 4 1.8GHz, 256M RAM), the L-IDS process always takes up less than 0.1% of CPU time. As for memory, while average memory allocation for the uolsrd process is 5.8Mbytes, memory used by the L-IDS process is always lower than 2Mbytes. Network overhead is also low, as the intrusion detection process always happens in the neighborhood of nodes and a collaborative investigation (forwarding of mobile agents to remote nodes) is only triggered when some event that may be associated to an attack is detected locally. Therefore, the L-IDS seems to have good scalability, as communications generated by L-IDS do not depend on the interaction with nodes more than two hops away in the Manet.
The higher overhead in the network caused by the L-IDS consists of the flooding of alarm messages, when an intrusion is detected. L-IDSs must share this type of information to generate a collaborative response to intrusion. However, these messages happen only when an opponent is detected and only until this opponent certificate is revoked.

6.6. SECURITY EVALUATION

For the validation of implemented security services, an experimental Manet with 10 nodes is used, and it is described in 6.1. Two of these nodes are used as properly behaved nodes, i.e. they correctly run OLSR and DCDP and L-Cert and L-IDS services. The two remaining nodes act as opponents, generating the previously described attacks.

The evaluation process is divided into three phases. First, the network is formed with routing and self-configuration protocols being run together with the intrusion detection service only. The objective is to evaluate the efficiency of detection of the IDS created. The opponent nodes carry out the attacks defined against the protocols and the attack effects happen in a way similar to what happens when there is no protection mechanism (section 6.3). Effects of the attack can be observed even when the attacks are detected, as the corrective protection mechanism (interaction between L-IDS and L-Cert) is not active. To illustrate the detection process, Figure 6-2 shows a particular topology assumed by the Manet during the experiment. In the figure, properly behaved nodes are A, B, C, D, E, F, G and H, while opponent nodes are X and Y.

![Figure 6-2 – Example of Experimental Manet Topology](image-url)
Each of the four attacks defined in 4.3.1.1 are perpetrated simultaneously by opponent nodes X and Y. As a result, it is observed that at least 03 of the L-IDSes running in correct nodes are capable of detecting attacks generated by each of the opponents, without any false negative. Table 6-21 shows the effects of attacks generated by X, observed by the correct node A.

Table 6-21 – Detection of Attacks against OLSR Protocol

<table>
<thead>
<tr>
<th>Attack</th>
<th>Produced/modified message</th>
<th>MPR Set before the attack</th>
<th>MPR Set during attack</th>
<th>Nodes detecting the attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>N+1HOP</td>
<td>HELLO, with B,C,D,E,F,H,Z = “sym”</td>
<td>D,E</td>
<td>X</td>
<td>C,D,G</td>
</tr>
<tr>
<td>NHOP</td>
<td>HELLO, with B,C,D,E = “lost”, impersonating E</td>
<td>D,E</td>
<td>D</td>
<td>A,B,F</td>
</tr>
<tr>
<td>TC_MS</td>
<td>TC, with F,G,H</td>
<td>D,E</td>
<td>X</td>
<td>A,B,F,G,H</td>
</tr>
<tr>
<td>TC_SEQNUM</td>
<td>TC de E, with modified sequence number</td>
<td>D,E</td>
<td>D,X</td>
<td>A,B,C,D,E,F,G,H</td>
</tr>
</tbody>
</table>

As the experimental topology does not undergo sudden variations, remaining nearly static throughout the experiment, there are no false positives for defined attack signatures. This result shows the usage of an intrusion detection system per incorrect use, with attack signatures precisely specified (low rate of false positives) is a coherent alternative to start automatic response to intrusions, through the revoking of certificates, as this work proposes. However, even if this first result is successful, it is not yet conclusive, as the evaluation of behavior in the presence of mobility was not yet properly characterized.

Another important aspect is that the precise specification of attack signatures is a complex task which can only be performed for previously known attacks. However, signatures may be generalized to identify non-acceptable states in the running of protocols being monitored, as in the case of NHOP and N+1HOP attack signatures, in which malfunctions in the scaling protocol and in network local topology trigger alarms. These signatures may be completed to model a more complete set of attacks, even those which are not yet explicitly developed, which are then identified by the appearance of non-acceptable running conditions. This approach, usually called intrusion detection per specification [64,111] is another field to be further investigated in future works.

The second phase of the validation process consists of running OLSR and DCDP with preventive protection (MAE), along with L-Cert. Thus, at this phase, only the L-IDS is not starting. The eight properly behaved nodes have their certificates and parts of the private key distributed off-line by the dealer. As shown in Figure 6-2, any correct node in the trial
topology has at least three neighbors equally correct. The coalition sizes as $K = 3$ is then defined. Obviously, attacks generated from opponent nodes with no MAE authentication are no longer effective, since such messages are discarded in the processing by the OLSR and DCDP daemons. However, as the IDS service is still not available, attacks generated by certified nodes (affected) can still disturb the network. To simulate the existence of affected nodes, the two opponent nodes are allowed a certificate, using the distributed certification process (L-Cert). Then it is possible to check the proper operation of certification and self-configuration with certification services: the opponent nodes first request a certificate, assigning a temporary IP address, and then request and receive an IPs address block through the correct usage of DCDP. Having the certificates and an IP address, the nodes begin to generate attacks which are again successful in disturbing the routing and self-configuration services.

In the third and last phase of the validation process, all implemented security services are activated. Then, when attacks against the OLSR protocol are generated, the L-IDS begin detecting the presence of affected nodes and revoke their certificates ($K = 3$). With the revoking of the opponents’ certificates, the attack effects are fully offset. In the experiments, the maximum time for certificate revoking of both opponents happened in the case of TC_MS attacks (production of TC messages), whose process took 12 seconds to be completed (including certification revocation).

Table 6-22 shows the average time for generation of group alarm for each of the considered attacks. This time represents the time the opponent has to “escape” the modification of neighboring nodes. However, accusations against a node are stored in the alarm manager modules until the affected nodes certificates are revoked, moving then to a detection “log” (in a file). Thus, if a node manages to escape the monitoring of its neighbors, but again acts incorrectly, its certificate can still be revoked based on the stored information. Also, nodes generating accusations against opponents will refuse certificate renewal to those nodes.
Another important aspect is the choice of $K$ parameter. Clearly, there is a relationship between security and service performance/availability in this choice. If $K$ is chosen with value lower than the neighborhood size, all services may be provided in a localized manner. Also, for the intrusion response process to be effective, it is necessary that there are at least $K$ nodes detecting attacks generated by a given opponent. For $K$ values higher than the neighborhood size, it is possible that the number of neighbors who can detect disturbing activities in the affected node is not enough to cause the opponent’s certificate revoking, allowing it to continue the attack generation locally without having its certificate revoked. In such cases, the nodes detecting the attacks must be able to “ask the help” of other correct nodes, so that they can come close to the neighborhood and detect the attack to complete the required number of nodes in the coalition that revokes the certificate.

It is also possible that there are affected nodes in a neighborhood which do not cooperate in the detection and intrusion response process. Thus, even if $K$ is lower than the neighborhood, situations in which the nodes of a neighborhood cooperating in the intrusion detection process are not enough are possible. A fairly simple heuristics may be used to define the value of $K$. Let’s use $ns$ as the average size of neighborhood. The maximum number of affected nodes in a network is equal to $K - 1$. Then, assuming $K = ns - (K - 1)$, we have $K = (ns + 1)/2$. This is the exact case of the Figure 6-2 topology, as considering X and Y nodes, the smaller number of nodes in a neighborhood (including the node itself) is 5. Therefore, $K = 3$ meets the proposed heuristics, in this case.

<table>
<thead>
<tr>
<th>Attack</th>
<th>Average time for generation of group alarm (s)*</th>
</tr>
</thead>
<tbody>
<tr>
<td>N+1HOP</td>
<td>2.2</td>
</tr>
<tr>
<td>NHOP</td>
<td>3.1</td>
</tr>
<tr>
<td>TC_MS</td>
<td>3.5</td>
</tr>
<tr>
<td>TC_SEQNUM</td>
<td>1.4</td>
</tr>
</tbody>
</table>

*Including the generation of coalition forming message, but not certificate revoking.
6.7. L-IDS: DETECTION PER BEHAVIOR MODELING

Many EM algorithm implementations are available from on-line scientific libraries like *statlib* and other on-line specialized sources. They include widely used implementations, like the *autoclass* program [22] (originally in Fortran, recently converted to C), EEMIX [76] (Fortran) and MCLUST [39] (environment S-Plus). No Java implementation for the EM algorithm for multivaried v.a. was found. As the L-IDS is fully written in Java (i.e. sensor, alarm manager, etc.), it is necessary to implement the IDS core per behavior modeling also in Java. In [2], there is an implementation of the EM algorithm for the generalized bi-dimensional case applied to Gaussian distributions. Although it is poorly documented, this implementation was completely rewritten for the n-dimensional case.

Figure 6-3 and Figure 6-4 illustrate a behavior modeling algorithm and intrusion detection simulation for a new sample considered normal and another one considered not normal, respectively. Reference data were created from 03 gaussian distributions, independent and clearly separated, and mixed in equal proportions. The EM algorithm, along with the algorithm of automatic estimation of parametric GMM order were successful in identifying the distribution parameter and the optimum model order, a posteriori, as the figures show. Values calculated for the entropy of parametric GMM adjusted to first, second, third and fourth orders EM algorithm are also shown. The figures illustrate the clear discrimination of the model order.

To detect intrusion, an event is considered normal when the $\lambda$ (Eq. 5-15) value is equal to or higher than 0.012, as that is the lowest $\lambda$ value of all realizations observed during the training phase. Therefore, for the case of Figure 6-3, the new event (red point) that corresponds to $\lambda = 0.3279$ is considered normal. On the other hand, the Figure 6-4 event, for which $\lambda = 0.000157$, is considered as a sign of intrusion.

---

Figure 6-3 – Behavior model with 03 clusters and recognition of new datum reflecting normal behavior.

Figure 6-4 – Behavior model with 03 clusters and recognition of new datum reflecting abnormal behavior.
The data present in the previous figures are useful only as demonstrations, as they were created artificially from distributions having exactly the same distribution that forms the nuclear functions of the mix model being used. Therefore, it is quite natural that the reference data perfectly match the model adjusted to them. In real life, however, such matching will not always occur.

Figure 6-5 illustrates the generation model and the simulation data processing for verifying the applicability of techniques for the intrusion detection per behavior, presented in section 5.4, to the detection of attacks against Manet networks. Firstly, the script trafficgen allows the simulation models to be adjusted (i.e. Manet with 50 nodes, 250m x 250m area, 50m transmission span, etc.). The ns-2 is used as a simulation tool and creates a trace file containing all generated packages, forwarded and received in all network nodes (out traffic). However, MIB variants must be kept and monitored for each node. The file is then split into many other files, one per network node, using program ns2tcpdump. Only packages generated, received or forwarded through this node are put in the same file. Therefore, such file corresponds to a dump of packages captured by a network analyzer, with capturing interface in non malicious mode. This file also transforms abstract packages created by the ns-2 into packages similar to those captured by a network analyzer: all fields of layer 3 and 4 protocols are filled out (including 4 bytes IPv4 addresses) and an absolute timestamp is assigned, compatible with the relative time measurement used by ns-2, for each package. The result of ns2tcpdump program is *.pcap files, in a format compatible with the dump format of raw packages of the libpcap library. As this format is supported by many network analyzers, like ethereal47, the *.pcap files may be viewed and analyzed with such tools. Then, each of such files is processed by the tcpdump2mib program, which produces (*.mib files) a list of samples for the MIB variants value, sampled within a time interval that can be defined per parameter pass upon command. Finally, a module of modified L-IDS MIB extractor allows the feeding of such information into the L-IDS. This data collector, designed for off-line data processing, carries out periodic consultations which show, for the time interval of the consultation, the values assumed by MIB variants stored in the *.mib file. It is important to remember that the sampling time informed to tcpdump2mib program (i.e. for generation of *.mib file) does not have to be the same time of periodic consultations used by the L-IDS MIB extractor module. In real life, consultation time is longer than time used by tcpdump2mib.

Two traffic models are closely analyzed: TCP and UDP. Separate usage of these models creates an implicit discrimination of all UDP and TCP generated traffic. Therefore the UDP behavior model will model only the videoconference and routing protocol applications. In the TCP case, the traffic generated by Telnet and FTP applications is modeled.

![Diagram of simulation generation process]

Figure 6-5 – Simulation generation process

In the case of UDP model, only the udpInDatagrams (UDP datagrams UDP which enter a node are used) and ipForwDatagrams (IP datagrams IP forwarded by the node) variants. As such variants are single-mode ascending, the result of event abstraction is a learning event (realization), whose value (udpIn ; ipForw) is obtained subtracting from (udpInDatagrams ; ipForwDatagrams) (current periodic consultation) current value the preceding value (previous periodic consultation). The consultation period was adjusted to be equal to the OLSR TC interval (equal to three times the HELLO interval). To do the training and model adjustment, all training events generated in all nodes are consumed in the same L-IDS, and then the GMM is adjusted to the reference data (events). The result is the parameters of the mix models, which are informed in a detection status message and distributed to all network L-IDS. At this time (receiving of detection status message) the intrusion detection process can be started.

As for the GMM adjustment to the data resulting from the simulation to UDP model, the formation of two well defined clusters can be observed: the first, with average on (52.3 ; 93.9) datagrams and standard deviation of (10.2 ; 39.7) datagrams. Certainly, this cluster indicates the traffic conditions of a node that is not receiving any package from the videoconference application, and may or may not be forwarding videoconference datagrams (high standard deviation in ipForwDatagrams variants). Another cluster, with average on (203...
158
datagrams and with standard deviation equal to \((21.1 ; 47.1)\) datagrams results from the videoconference traffic modeling (source CBR at 128kbps). Obviously, there is a contribution of OLSR protocol traffic in the value of the average and the deviation of this cluster. The ratio between variants is positive but small (36,7 datagrams).

For the DDoS attack generation, a UDP CBR (2Mbps) traffic generation is simulated in four origin nodes chosen randomly, toward a single destination node. Applying the detection model, non-normal situations are detected in all nodes forwarding traffic from the origin to the destination. This result, quite interesting in the DDoS detection perspective, is made possible by the joint analysis of two udpInDatagrams and ipForwDatagrams variants.

Another analysis of this DDoS attack is necessary: obviously the node receiving all generated traffic (from all its neighbors) will soon be unavailable (the ns-2 indicates the generation of many package forwarding and disposal errors in the area adjoining the destination node). However, distant nodes, although generating/forwarding a significant data quantity, are not necessarily broken by the attack. As the intrusion detection system identifies malfunctions in all nodes along the path, it is suggested that, in case of interaction between these intermediate nodes, the forwarding of packages coming from that origin be blocked. The forwarding must be blocked based on the link addresses and not on the destination addresses of IP datagram, as they can be easily faked and, in more advanced DDoS attacks, they are continuously modified (at every package).

In the case of TCP model, MIB tcpPassiveOpens (number of connections opened passively in the node) and tcpInSegs (number of segments received, including ones with errors and for connection opening) are used as variants. As with UDP, a consultation with period equal to \(5 \times (TC \text{ interval})\) is defined, where event abstraction calculates \((tcpPO ; tcpIN)\) as the difference between \((tcpPassiveOpens ; tcpInSegs)\) value in current and previous consultation. To prevent singularities (i.e. formation of a cluster with zero average and small variation for tcpPassiveOpens), events in which tcpPassiveOpens is equal to zero are disposed of as normal with regards to both the learning process and the intrusion detection process. In terms of adjustment, formation of two clusters with averages on \((1.11 ; 38.41)\) and on \((1.05 ; 97.11)\) happen, modeling respectively the telnet and the FTP.

Events generated in all nodes are consumed in a single L-IDS and the detection status message containing the model adjusted parameters is distributed. To generate a scanner attack, an origin-destination pair is chosen randomly and the origin then sends requests for TCP connections to the destination, at a rate of 10 requests per second. A drain is installed on the destination and at every 30 connection requests one is accepted (i.e. indicating a match
with a port that responds). As soon as MIB variant values begin to reflect the additional traffic, the attack is detected by the destination.

The data obtained in the simulations encourage the continuation of research. Many improvements of the statistic model and data gathering are possible. Also, the results must be validated with data from real networks.
7. CONCLUSIONS

This work presents the conception and implementation of a security model for ad hoc mobile networks, completely distributed and self-organized, having in the combination of preventive and corrective security services as its most distinctive feature. Prevention happens through the establishment of a reliable model that is used as the key element in an authentication service. The model is materialized in the form of a distributed and self-organized certification service. Correction happens through an intrusion detection system, equally distributed and self-organized. The main contribution to the model effectiveness is the definition of precise interaction mechanisms between preventive and corrective services in order to offer a single automatic and balanced response to intrusions.

As for the conception of the certification service and definition of authentication mechanisms, this work uses the work of other authors as reference, adding important contributions, like the definition of CRL maintenance and formation mechanisms and cache of valid certifications; the restructuring of parameters that define the type of certification security policy to be used, allowing the adaptation of certification mechanisms to different Manets application contexts – with special focus on the correction of vulnerabilities related to Sybil attacks; definition of trust relationships between different ACDs – enabling the merging of Manet networks started at different times and circumstances; and the proposal of an authentication extension for Manet (MAE) which incorporates security mechanisms capable of securing the many routing and self-configuration protocols defined for this type of network.

The secure model and the usage of authentication technique, both based on digital certification, is an important feature of the model as it offers a non-refusable identification of entities carrying out certain actions in the network. It is on this feature that the interaction between preventive and corrective services are based, enabling the devising of a response to the intrusions based on the revoking of certificates. The counterpoint for the model is the compulsory use of primitive asymmetric encryption. Such operations represent to this day a high computational cost which may be intolerable in some environments. However, the computer technology advancement and the increase of processing and memory capacity, together with a coherent choice of service parameters (e.g. switch and coalition sizes, and others), the use of asymmetric encryption shall not prevent the application of the security model in different scenarios of Manet application. Also, the results achieved encourage the
search for other asymmetric encryption techniques (i.e. elliptic curve encryption), alternatives to the classic RSA, which result in a lower computer and network overhead.

As for the conception and implementation of the intrusion detection system, two approaches to the intrusion detection process are used together, enabling the detection of many attack types in different points of the system architecture. The incorrect use approach allows the definition of precise attack signatures against routing protocols, generating reliable information on the existence of malicious or error activity in the network, with positive identification of the origin. Thus, the trust collaboratively assigned to some nodes/users may be removed, in case they are affected or behave incorrectly. On the other hand, the behavior modeling approach to intrusion detection not only enables the identification of complex attacks, like DDoS, but also offers insights on how such attack effects may be minimized using the collaborative mechanisms already defined for the security model as a whole.

From the real complete implementation of the security model for OLSR routing protocols and DCDP self-configuration, important lessons on the proposed model can be learned:

- Choice of parameters for the implementation of certification and authentication services and definition of more precise security policies for the services result in important trade-offs between performance criteria and computer and network overhead, and requirements referring to the strength of security of the self-organized and collaborative mechanisms.

- The combined use of intrusion detection certification reinforces the security offered by the first, allowing the correction of later distortions through the elimination of poorly behaving entities from the network. Taking advantage of the redundancy and point-to-point connectivity of Manets, the interaction and collaboration mechanisms may be designed so as to enable the collaboration, and the successful completion, locally. This way, both the IDS and the certification are fully scalable even for large networks and networks with high node density.

- Choice of MIB as information source for the intrusion detection service allows the monitoring of different aspects of system and network operation, and also of applications. This is a mechanism that may be readily and easily extended to cases in which the necessary information is not available as part of standardized MIB (i.e. OLSR experimental MIB), and it also offers a significant amount of information about the system to be monitored, at a low cost.

- Experiments in real environments do not enable evaluation of all aspects required for
the complete validation of the mechanisms, considering the difficulties in reproducing different conditions of traffic mobility, broadcasting and profiles. Therefore, network simulators, which are fairly advanced in terms of resources for the simulation of mobile networks, offer complementary visions to the visions provided by exercises with real Manets. Especially, simulation exercises must allow the evaluation of the effects of the network topology dynamics (i.e. mobility and propagation) in the security services projected.

Regarding the anomaly approach to intrusion detection, a new model for statistic modelling of network behavior is proposed, using a parametric model of gaussian mixes, with detection of anomalies per use of Bayesian classification criteria (a posteriori probability calculation). The objective of this model is to allow the simultaneous modeling of different event types (i.e. applications) which reflect on the same group of variants available for monitoring. Preliminary results indicate this type of model may be adequate, provided choice of variants to be modeled and monitored is careful. Notwithstanding, the model, in addition to requiring further validation with reference data coming from real networks, has some important limitations that need to be investigated and flexibilized, including the difficulty to model non-numeric data or numeric data with special characteristics (i.e. modular data like time of day or day of the week, etc.). Also the parametric Gaussian mixed model is not adequate for modeling more complex data which do not have the statistic characteristic of normality.

Finally, security issues in ad hoc mobile networks are in their initial phase of conception and development, and there is still a lot to be researched about this important and current topic. Possible future works directly arising from the results of this thesis include:

- Definition (formal) and implementation of a model for the self-organized initializing of certification services. The results of this work use the figure of a dealer to make the initial distribution of certificates and parts of the ACD private key between the first (K) network nodes. There are mechanisms for the collaborative generation of a RSA private key by entities not having any complete knowledge of the generated key [38], and for the distribution of this shared key, in the form of key parts similar to the ones used in the projected certification services. It is necessary to adapt such mechanisms to the requirements of generation and distribution of the reliability model proposed herein. That may eliminate the need for a dealer at the network initial phase, making the certification mechanisms, in fact, completely self-organized.
• Implementation of the operation modes with symmetric encryption. Security services are detailed, designed and implemented only for use with asymmetric encryption. Results herein prove this technique still represents a high computational cost which may be an impediment in environments with lower processing capacity nodes. Implementation of security services may be, therefore, completed with authentication mechanisms using symmetric encryption, through authentication protocols especially designed and adapted for spontaneous environments, like in the case of the TESLA protocol.

• Definition of security policies and application techniques and enforcement of such policies for distributed certification processes. Even if the model developed herein has made the adoption of different distributed certification policies more flexible, the methods for verification and enforcement of policies remain undefined. This important issue, still not tackled in specialist technical literature, is worth of research.

• Extension of security mechanisms to other Manet routing protocols. This work offers an analysis of vulnerabilities that takes into account the four main Manet routing protocols (AODV, OLSR, TBRPF and DSR), as well as an extension of authentication for Manet (MAE), defined jointly with a distributed certification service, which allows the preventive security of such protocols. However, the corrective protection mechanisms of the security model are developed only for the OLSR protocol. To extend the intrusion detection and response service to other routing protocols, or even to the DCDP self-configuration protocol, considered herein, one may: (1) review the vulnerability analysis for identification and specification of concrete attacks which may be implemented in the attack generator; (2) identify and specify such attacks signatures, in terms of the existing detection mechanisms or incorporating new resources to the existing ones; (3) define and implement local data collection agents (i.e. MIB and communication daemon ↔ snmp agent); and (4) test and validate the process.

• Simulations to verify the effect of mobility on the security services, and especially on the IDS performance and effectiveness. Validation of intrusion detection service in the context of this work is limited to nearly static topologies, with reduced mobility. However, Manets are mobile networks! Thus, results herein need additional validation if effects of mobility on processes and mechanism effectiveness are to be studied. Particularly, it is still necessary to assess the IDS effectiveness in terms of false positives and false negatives resulting from mobility. The response to intrusion
processes must undergo the same type of analysis, to determine whether there are no circumstances in which opponents may move to escape certificate revocation.

- **Access control to collaborative network level (L-Firewall).** Results of intrusion detection per behavior modeling show at least one type of attack (DDoS) where collaboration between network nodes to filter flows of undesirable packages may be an effective response to the detection of this type of attack, minimizing its effects. As the flow of information on the network origin address (IP) is not reliable, nodes must cooperate to identify the paths used by these flows with hop-to-hop information, many times derived from the link layer address. That is one of the functions that may be developed through collaborative L-Firewall services.

- **Security policy management service (L-SPM).** Attack signature bases, package filtering rules, access lists, and other important information for the security services, must have an updating routine compatible with the nature of Manets. A self-organized service of security policy updating and its impact on the parameters of policy control mechanisms is still challenging current research.

- **Intrusion detection based on behavior modeling:** The proposed model of intrusion detection based on behavior malfunctions is still in the initial development phases, and was used only with artificial data which do not necessarily represent a real behavior network. The model must be validated in experiments using real data. Also, improvements and flexibilization of the requirements in the model concept are possible, such as the use of different types of core functions, the use of semi-parametric mix models, the adoption of stochastic models (i.e. Markov) for the elimination of the requirement for statistic independence between realizations (events), and others.

  This work was validated in presentations in important international congresses and with the publication in internationally acknowledged media [3, 13, 85, 92, 93, 94, 95, 96, 97]. It is our intention to release, with a GPL or FreeBSD license, the produced programs and program improvements in the web address http://www.manet.redes.unb.br.
BIBLIOGRAPHICAL REFERENCES

[96] R. Puttini; L. Me; R. de Sousa - Preventive and Corrective Protection for


ATTACHMENT I – IEEE 802.11B OR WI-FIAO AD-HOC MODE\textsuperscript{48} WIRELESS NETWORK TECHNOLOGIES

This is by far the most widely used wireless network (WLAN) technology today. It is a very cost-effective technology and speeds of up to 11Mbps. Its operating frequency is 2.4GHz, with 11 available communications channels, of which only three are channels which, if operated in parallel, do not present a significant inter-channel interference: channels 1, 6, and 11. In practice, these channels, if used simultaneously in the same region, do not generate signal degradation that affects the link's end speed. The IEEE 802.11b technology was endorsed by a representative group of manufacturers, and a convention was adopted to facilitate identification of interoperable products in this technology. This certification was called Wi-Fi, which means \textit{Wireless Fidelity}.

The range achieved with IEEE 802.11b is very much dependent on the type of antenna being used, which may vary in terms of gain and technology, but the rule for a 1dBi standard antenna is a range of 100m for indoor areas, and up to 300m for open or outdoor areas. If the antenna being used is changed, distances measured in kilometers may be achieved, and that is why today there are already links operating at those distances.

In the IEEE 802.11b protocol, the key media access mechanism is called \textit{distributed coordination function} (DCF). This mechanism is based in a random access scheme that uses carrier detection to avoid collisions. Therefore, this protocol is configured as a CSMA/CA (\textit{carrier sense multiple access with collision avoidance}) access mode. In this protocol, whenever a station has a packet to transmit, it monitors the channel activity. If the channel is idle for a period greater than the time between frames being distributed (\textit{distributed interframe space} – DIFS), the station transmits the packet. Otherwise, it monitors the channel until it is idle for a period equal to DIFS, and, then, initiates a random duration counter (\textit{backoff}) before initiating the transmission, to minimize the probability of a new collision. Additionally, to prevent a single station from monopolizing the channel, it must reset the counter whenever it transmits two or more adjoining packets. Since a station cannot detect whether there has been a collision, an acknowledgement (ACK) is transmitted by the destination station after a short period of time called \textit{short interframe space} (SIFS), whenever a packet is received without errors. If, after a period of time equal to ACKtimeout no ACK

\textsuperscript{48} Adapted from http://www.brasilmobile.com
has been received, the transmitting station assumes an error has occurred (e.g. collision), and re-schedules the transmission according to the size of the backoff window.

The security mechanism at link level on IEEE 802.11b networks is known as *wired equivalent privacy* (WEP). WEP is responsible for encoding the data transmitted through the network. There are two WEP standards, 64 and 128 bits. The 64-bit standard is supported by any access point or interface following the Wi-Fi standard, which includes all products presently sold. The 128-bit standard, on the other hand, is not supported by all products. To enable it, it is necessary that all components used in the network support the standard, otherwise nodes supporting the 64-bit standard only will remain out of the network.

Actually, WEP is comprised of two distinct keys, 40 and 24 bits on the 64-bit standard, and 104 and 24 bits in the 128-bit standard. Therefore, the encryption complexity used in both standards is not the same as in true 64- or 128-bit encryption standards. In addition to the bit number detail in the encryption keys, WEP has other vulnerabilities. Some programs already largely available are capable of breaking the encoding keys if monitoring the network traffic for some hours is possible, and the tendency is for these tools to become more sophisticated with time.

WEP is being phased out on most access points, but can be easily activated through the configuration utility. The most complicated is that it is necessary to manually define an encryption key (an alphanumeric or hexadecimal value, depending on the utility) that should be the same to all access points and network stations. At the stations, the key, as well as the ESSID address and other network configurations, can be defined by using another utility, supplied by the board's manufacturer.
ATTACHMENT II – THRESHOLD ENCRYPTION

In 1979, A. Shamir [102] proposed a model that allows dividing a piece of information D in N parts, so that D can be rebuilt from K parts, but the knowledge of K – 1 parts reveals no information about D. This technique was proposed for the implementation of an effective key management encrypted system.

This model was called the (K,N) threshold scheme, where N is the total number of parts from D and K is the minimum number of parts needed to reconstruct D. According to Shamir, an efficient threshold scheme can be very useful in managing encryption keys, since it allows for protecting an encryption key.

The scheme is based in polynomial interpolation: given K points in a two-dimensional plane \((x_i, y_i)\), with different values of \(x_i\), there is one and only one polynomial \(q(x)\), of K – 1th degree, with \(q(x_i) = y_i\) for every \(i\). To divide D into N parts as in \(D_i\), a random polynomial of the K – 1th degree is generated, as shown in equation 1.

\[
q(x) = a_0 + a_1x + ... + a_kx^{k-1}
\]  

(1)

where: \(a_0 = D\).

The, \(D_i\) is calculated (equation 2):

\[
D_i = q(x_i) \mod p
\]  

(2)

where: \(p\) is a prime number, \(x_i\) are called identifiers for each part of D and the modulus provides a greater precision by assuring a field where interpolation is possible.

This prime number is greater than D and N. Coefficients \(a_1, ..., a_{k-1}\) are randomly chosen from a uniform distribution of integers in \([0, p]\).

Given any subset of K values of \(D_i\), with their respective identifiers \((x_i)\), the coefficients \(q(x)\) can be calculated by interpolating and subsequent calculation of \(D = q(0)\). However, knowledge of only K – 1 values is not enough to calculate D.

Shamir defined some properties for the (K,N) threshold scheme. The size of the \(D_i\) part should not be greater than the size of the original datum D. When a value for K is fixed, the \(D_i\) parts may be dynamically added or eliminated without affecting the other parts.
Additionally, it is easy to change the $D_i$ without changing $D$, provided a new polynomial with the same $D$ is created.

Making this exchange frequently enhances the system's security, since the broken parts cannot be accumulated, unless all are originated from the same polynomial $q(x)$.

Lastly, a hierarchical scheme can be created where the number of parts to determine $D$ depends on their importance. For example, if the president of a company receives three values for $D$, the vice-president receives two, and the executives receive only one value each, the $(3,N)$ scheme is defined. With that, three executives are necessary to access the system, or one executive and the vice-president, or the president alone.
ATTACHMENT III – CERTIFICATION PROTOCOL MESSAGE SYNTAX

Message syntax for the Manet certification service and for the enveloped certification objects is shown in Figure A3.1 and Figure A3.2 below, respectively.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>MSG_TYPE</td>
<td>MSG_TTL</td>
<td>MSG_LENGTH</td>
<td></td>
</tr>
<tr>
<td>MSG_ORIGINATOR</td>
<td>MSG_SEQ_NUMBER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CERT_OBJECTS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(variable size, completed for 32-bit word alignment)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure A3.1 – L-Cert Message Syntax

Where:
- MSG_TYPE: field used to differentiate the message from other types of enveloped messages;
- MSG_TTL: time-to-live, used to define the range, in terms of hops, of the message dissemination (flooding); (e.g. 0 if within one hop and 255 if the entire network);
- MSG_LENGTH: message size, in bytes;
- MSG_ORIGINATOR: message originator (e.g. IP address);
- MSG_SEQ_NUMBER: message sequential number, to prevent the processing of duplicate or old messages (each node starts a pseudo-random number generator that generates sequential numbers for its messages);
- CERT_OBJECTS: one or more certification objects.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>CERT_TYPE</td>
<td>Reserved</td>
<td>NEXT_CERT</td>
<td></td>
</tr>
<tr>
<td>CERTIFICATE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(variable size, completed for 32-bit word alignment)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure A3.2 – Syntax of each CERT_OBJECT

Where:
- CERT_TYPE: object type identifier;
- CERT_LENGTH: object size, in bytes;
- CERT: certification object, according to the CERT_TYPE field.
ATTACHMENT IV – MANET AUTHENTICATION EXTENSION
(MAE) SYNTAX

Syntaxes proposed for the MAE and its authentication objects are shown in Figure A4.1 and Figure A4.2, respectively.

![Diagram of MAE Syntax](image)

<table>
<thead>
<tr>
<th>MSG_TYPE</th>
<th>Reserved</th>
<th>MSG_LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

AUTH_OBJECTS
(variable size, completed for 32-bit word alignment)

Figure A4.1 – MAE Syntax

Where:
- **MSG_TYPE**: field used to differentiate the message from other types of enveloped messages;
- **MSG_LENGTH**: message size, in bytes;
- **AUTH_OBJECTS**: one or more authentication objects.

![Diagram of Authentication Object Syntax](image)

<table>
<thead>
<tr>
<th>AUTH_TYPE</th>
<th>Reserved</th>
<th>NEXT_AUTH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

AUTHENTICATOR
(variable size, completed for 32-bit word alignment)

Figure A4.2 – Syntax of each Authentication Object

Where:
- **AUTH_TYPE**: type of authentication object;
- **NEXT_AUTH**: object size, in bytes;
- **AUTHENTICATOR**: object of the authentication (e.g. DS, MAC, HC, etc.).
ATTACHMENT V –XML (DTD) SPECIFICATION FOR ALL L-IDS MESSAGES

<!-- DTD for L-IDS -->

<!ELEMENT MESSAGE (ID,ID_ENT_ORIG,PAR_ATAC_ALVO*,TIPO_MSG,IS_LOCAL,ID_ENT_DEST,IS_FLOODED,TTL)>

<!ELEMENT ID (#PCDATA)>
<!ELEMENT ID_ENT_ORIG (#PCDATA)>

<!ELEMENT PAR_ATAC_ALVO (ATACANTE,ALVO)?>
<!ELEMENT ATACANTE (ID_ENT_REDE)?>
<!ELEMENT ID_ENT_REDE (#PCDATA)>
<!ELEMENT ALVO (ID_ENT_REDE)?>
<!ELEMENT ID_ENT_REDE (#PCDATA)>

<!ELEMENT TIPO_MSG (MSG_EVENT,MSG_QUERY,MSG_QUERY_PERIODIC,MSG_ALERT,MSG_STATE_DETECT)>

<!ELEMENT MSG_EVENT (#PCDATA)>
<!ELEMENT MSG_QUERY (#PCDATA)>

<!ELEMENT MSG_QUERY_PERIODIC (PERIOD)>
<!ELEMENT PERIOD (#PCDATA)>

<!ELEMENT MSG_ALERT (ID_ATAQUE)>
<!ELEMENT ID_ATAQUE (#PCDATA)>

<!ELEMENT MSG_STATE_DETECT (ID_ATAQUE,IS_CLONE)>
<!ELEMENT ID_ATAQUE (#PCDATA)>
<!ELEMENT IS_CLONE (#PCDATA)>

<!ELEMENT ID_ENT_ORIG (#PCDATA)>
<!ELEMENT IS_LOCAL (#PCDATA)>
<!ELEMENT IS_FLOODED (#PCDATA)>

<!ELEMENT TTL (#PCDATA)>

<!ATTLIST IS_CLONE valor (TRUE|FALSE) #REQUIRED>
<!ATTLIST IS_LOCAL valor (TRUE|FALSE) #REQUIRED>
boolean (TRUE|FALSE) #REQUIRED>
<!ATTLIST IS_FLOODED
boolean (TRUE|FALSE) #REQUIRED>

ATTACHMENT VI – EXPERIMENTAL MIB FOR THE OLSR PROTOCOL IN THE ASN-1 FORMAT

RAHMS-OLSR-MIB DEFINITIONS ::= BEGIN

IMPORTS
   MODULE-IDENTITY, OBJECT-TYPE
   experimental, IpAddress FROM SNMPv2-SMI ;

rahmsOlsrMIB MODULE-IDENTITY
LAST-UPDATED "0207051145Z"
ORGANIZATION "ESEO"
CONTACT-INFO "rahms@eseo.fr"
DESCRIPTION "The MIB module for RAHMS networks"
::= { experimental 6060 }

rahms OBJECT IDENTIFIER ::= { experimental 6262 }

-- the olsr group
olsr OBJECT IDENTIFIER ::= { rahms 1 }

-- OLSR Neighbor Table
-- The OLSR Neighbor Table contains information concerning this entity's
-- existing neighbors and the status of the link between this host and
-- each of its neighbors

olsrNeighborTable OBJECT-TYPE
SYNTAX SEQUENCE OF olsrNeighborEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "A table containing OLSR neighbor information."
::= { olsr 1 }

olsrNeighborEntry OBJECT-TYPE
SYNTAX olsrNeighborEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "A conceptual row of the olsrNeighborTable containing
   Information about the connection towards a particular
   OLSR neighbor. Each row of this table is transient, in
   that it ceases to exist when (or soon after) the
   connection with a neighbor is lost."
INDEX { olsrNeighborAddress }
::= { olsrNeighborTable 1 }
olsrNeighborEntry ::= SEQUENCE {
    olsrNeighborState INTEGER,
    olsrPreviousNeighborState INTEGER,
    olsrNeighborAddress IpAddress
}

olsrNeighborState OBJECT-TYPE
SYNTAX INTEGER { ASYM(1),SYM(2),MPR(3),LOST(4) }
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The state of this OLSR neighbor connection."
::= { olsrNeighborEntry 1 }

olsrPreviousNeighborState OBJECT-TYPE
SYNTAX INTEGER { ASYM(1),SYM(2),MPR(3),LOST(4) }
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The previous state of this OLSR neighbor connection."
::= { olsrNeighborEntry 2 }

olsrNeighborAddress OBJECT-TYPE
SYNTAX IpAddress
MAX-ACCESS read-only
STATUS current
DESCRIPTION "This neighbor IP address"
::= { olsrNeighborEntry 3 }
END