ASSESSING SECURITY OF MOBILE TELECOMMUNICATION NETWORKS

A Dissertation in
Computer Science and Engineering
by
Kameswari Kotapati

© 2008 Kameswari Kotapati

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2008
The dissertation of Kameswari Kotapati was reviewed and approved* by the following:

Thomas F. LaPorta  
Distinguished Professor of Computer Science and Engineering  
Dissertation Co-Adviser  
Co-Chair of Committee

Peng Liu  
Associate Professor of Information Science and Technology  
Dissertation Co-Adviser  
Co-Chair of Committee

Trent Jaeger  
Associate Professor of Computer Science and Engineering

Carleen Maitland  
Assistant Professor of Information Science and Technology

Raj Acharya  
Professor of Computer Science and Engineering  
Head of the Department of Computer Science and Engineering

* Signatures are on file in the Graduate School.
Abstract

With the increase in popularity of mobile phones over landlines, the mobile telecommunication network has now become the primary source of communication for not only business and pleasure, but also for the many life and mission critical services such as E-911. These networks have become highly attractive targets to adversaries due to their heavy usage and their numerous vulnerabilities that may be easily exploited to cause major network outages.

Our dissertation is motivated by the numerous vulnerabilities suffered by these networks and the lack of comprehensive mobile telecommunication network specific security assessment. Our dissertation bridges this gap by assessing the current state of security of the mobile telecommunication network, and determining the network characteristics that must be changed to render the network more secure. In our assessment, we: (1) identify and classify possible threats to mobile telecommunication networks; (2) assess potential vulnerabilities that may be exploited to realize these threats; and (3) devise defense strategies to protect from such attacks.

In the first step, we conducted a comprehensive study to identify possible threats to the network, and developed a unique three dimensional taxonomy to classify these threats. This was the first detailed study of this type. Our most significant results from this study include identifying: the unique cascading effect of attacks in the mobile telecommunication network, by which an adversary can attack a network target from a remote location (for example, an adversary may prevent a police officer or an emergency health-care provider from receiving an E-911 call from a remote location); and the cross infrastructure cyber attack - a type of cascading attack where the adversary targets the mobile telecommunication network but attacks the Internet, which propagates the attack to the mobile telecommunication network. The technical impact of our work is that it has created an increased awareness of the threats faced by the mobile telecommunication network due to its numerous vulnerabilities.
In the second step, we developed the *Cellular Network Vulnerability Assessment Toolkit - CAT*, to assess potential network vulnerabilities. This toolkit allows a user to input specific pre-conditions or vulnerabilities, and provides a visual trace of the possible attacks that may be launched by exploiting the user input vulnerabilities. Visual traces allow users to trace the effect of an attack through a network and determine its side-effects and ultimate service disruption. The toolkit also allows a user to quantify the benefits of a security solution by removing attack effects based on the defenses provided. One major advantage of our toolkit is that potential solutions may be evaluated before expensive development and deployment.

Developing such a tool presented many challenges: telecommunication networks are complex systems that contain hundreds of data elements and support hundreds of services. CAT handles the complex and diverse nature the network using telecommunication specifications to find possible sources and targets of attacks. To trace the effect of an attack through the network we introduced a novel dependency and propagation model. Finally, we superimposed boolean properties on the propagation model to capture the impact of security solutions.

We used this toolkit to evaluate a newly standardized telecommunications security protocol called MAPSec. Our results show that MAPSec provides defenses for a narrow set of attacks, but does not effectively mitigate attacks with the largest potential impact. In fact, we have shown that attacks that have previously occurred would not be prevented by the current versions of MAPSec. The technical impact of this work is that it is now possible to uncover vulnerabilities in the mobile telecommunication network, determine their damage potential, and evaluate potential new solutions to fix these vulnerabilities.

In the third step, we propose a new security protocol, *End-to-End Security - EndSec* to address the most damaging attacks uncovered from the second part of our work. EndSec provides multi-hop security by inserting cryptographic checks on key parameters, allowing corruption of data or logic to be traced to its source. The technical impact of EndSec is that it brings accountability to the network previously non-existent, and enables real-time tracking of adversary activity.
To the best of our knowledge, our work is the first comprehensive security assessment of the mobile telecommunication network. Our taxonomy is the first to include cross infrastructure cyber attacks and to identify the cascading effect of attacks on telecommunication networks. Our toolkit, CAT, is the first to assess the vulnerability of the mobile telecommunication network and quantify the effectiveness of potential security solutions. Our protocol, EndSec is the first end-to-end security scheme for mobile telecommunication networks. Our results have been used as an important input to the Vulnerabilities Threat Modeling Working Group of the Next Generation Networks Task Force of the Presidents National Security Telecommunications Advisory Panel.
# Table of Contents

List of Tables .................................................... xi
List of Figures ..................................................... xii
Acknowledgments .................................................. xv

Chapter 1. Introduction .............................................. 1
  1.1 Contributions ................................................. 3
    1.1.1 Attack Taxonomy ......................................... 3
    1.1.2 Vulnerability Assessment ................................ 5
    1.1.3 Defense Strategy ...................................... 8
  1.2 Thesis Outline ............................................ 9

Chapter 2. Background .............................................. 10
  2.1 System Overview ........................................... 10
    2.1.1 Overall Architecture .................................. 11
    2.1.2 Core Network Organization ............................ 13
    2.1.3 Call Delivery Service .................................. 15
  2.2 Current State of Security .................................. 17
    2.2.1 Security in Radio Access Network ..................... 17
    2.2.2 Security in Core Network .............................. 18
    2.2.3 Security Implications of Internet Connectivity ....... 20
    2.2.4 Security Implications of PSTN Connectivity ........... 21

Chapter 3. Taxonomy of Attacks .................................... 23
  3.1 Introduction ............................................... 23
  3.2 Preliminaries .............................................. 24
    3.2.1 Cross Network Services ............................... 24
    3.2.2 Working of Call Forwarding Service .................. 25
### Chapter 5. aCAT

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1 Introduction</td>
<td>78</td>
</tr>
<tr>
<td>5.2 Architecture</td>
<td>79</td>
</tr>
<tr>
<td>5.3 The aCAT Model</td>
<td>79</td>
</tr>
<tr>
<td>5.3.1 Network Dependency Model</td>
<td>80</td>
</tr>
<tr>
<td>5.3.2 Propagation Model</td>
<td>83</td>
</tr>
<tr>
<td>5.4 aCAT knowledge base</td>
<td>83</td>
</tr>
<tr>
<td>5.5 Algorithms</td>
<td>85</td>
</tr>
<tr>
<td>5.5.1 Forward Algorithm</td>
<td>86</td>
</tr>
<tr>
<td>5.5.2 Reverse Algorithm</td>
<td>89</td>
</tr>
<tr>
<td>5.5.3 Combinatory algorithms</td>
<td>92</td>
</tr>
<tr>
<td>5.6 Interesting Attacks Discovered by aCAT</td>
<td>97</td>
</tr>
<tr>
<td>5.6.1 Alerting Attack</td>
<td>97</td>
</tr>
<tr>
<td>5.6.2 Power-Off Power-On Attack</td>
<td>99</td>
</tr>
<tr>
<td>5.6.3 Mixed Identity Attack</td>
<td>101</td>
</tr>
<tr>
<td>5.6.4 Call Redirection Attack</td>
<td>103</td>
</tr>
<tr>
<td>5.6.5 Missed Calls Attack</td>
<td>105</td>
</tr>
<tr>
<td>5.7 Impact of Expert Input</td>
<td>108</td>
</tr>
<tr>
<td>5.7.1 Experimental Results</td>
<td>108</td>
</tr>
<tr>
<td>5.7.2 Limitations of SDL</td>
<td>109</td>
</tr>
<tr>
<td>5.7.2.1 Syntax and Semantics</td>
<td>110</td>
</tr>
<tr>
<td>5.7.2.2 Technical areas</td>
<td>110</td>
</tr>
<tr>
<td>5.8 Discussion</td>
<td>112</td>
</tr>
</tbody>
</table>

### Chapter 6. eCAT

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Introduction</td>
<td>113</td>
</tr>
<tr>
<td>6.2 Preliminaries</td>
<td>115</td>
</tr>
<tr>
<td>6.2.1 MAPSec</td>
<td>115</td>
</tr>
<tr>
<td>6.2.2 eCAT Overview</td>
<td>117</td>
</tr>
<tr>
<td>6.3 Graph Marking: The Algorithm</td>
<td>118</td>
</tr>
<tr>
<td>6.3.1 Graph Marking Rationale</td>
<td>118</td>
</tr>
</tbody>
</table>
6.3.2 Graph Marking Algorithm .............................................. 119
6.4 Measuring Effective Coverage of Security Protocols ............... 124
6.5 Evaluating MAPSec by Marking Attack Graphs ....................... 127
  6.5.1 Short Message Service ............................................. 128
  6.5.2 Supplementary Service ........................................... 128
  6.5.3 Authentication Service ............................................ 130
  6.5.4 Experiments and Analysis ......................................... 132
6.6 MAPSec Utility Findings and Discussions ........................... 135
6.7 Related Work .............................................................. 136
6.8 Conclusion ................................................................. 138

  7.1 Introduction .............................................................. 139
  7.2 EndSec Overview ....................................................... 141
  7.3 Data Property Model .................................................. 142
  7.4 Network Model .......................................................... 144
  7.5 EndSec Threat Model .................................................. 145
  7.6 EndSec Assumptions .................................................... 148
  7.7 EndSec Details ........................................................... 149
    7.7.1 Structure of EndSec Messages ................................. 149
    7.7.2 EndSec Message Construction ................................. 150
    7.7.3 EndSec Corruption Check ......................................... 152
    7.7.4 EndSec Modifications .......................................... 158
  7.8 Security Analysis ...................................................... 159
  7.9 Performance Analysis ................................................ 160
    7.9.1 Time Delay ....................................................... 161
    7.9.2 Signaling Load ................................................... 166
  7.10 Related Work .......................................................... 172
  7.11 Discussion ............................................................. 174
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Agents in Circuit Switched domain</td>
<td>29</td>
</tr>
<tr>
<td>3.2</td>
<td>Agents in CF Server</td>
<td>31</td>
</tr>
<tr>
<td>3.3</td>
<td>Agents in Mail Server</td>
<td>31</td>
</tr>
<tr>
<td>3.4</td>
<td>Single Infrastructure attacks on CS domain classified by CASE I - I</td>
<td>40</td>
</tr>
<tr>
<td>3.5</td>
<td>Single Infrastructure attacks on CS domain classified by CASE I - II</td>
<td>41</td>
</tr>
<tr>
<td>3.6</td>
<td>Cross Infrastructure Cyber Attacks classified by CASE I</td>
<td>45</td>
</tr>
<tr>
<td>3.7</td>
<td>Attacks Classified by CASE II</td>
<td>46</td>
</tr>
<tr>
<td>4.1</td>
<td>Table 1-CAT Knowledge Base: Signaling Messages</td>
<td>62</td>
</tr>
<tr>
<td>4.2</td>
<td>Table 2-CAT Knowledge Base: Processes</td>
<td>63</td>
</tr>
<tr>
<td>5.1</td>
<td>Infection Propagation Rules</td>
<td>81</td>
</tr>
<tr>
<td>5.2</td>
<td>Table 1-aCAT Knowledge Base: Signaling Messages</td>
<td>83</td>
</tr>
<tr>
<td>5.3</td>
<td>Table 2-aCAT Knowledge Base: Processes</td>
<td>84</td>
</tr>
<tr>
<td>6.1</td>
<td>Total Number of Message</td>
<td>126</td>
</tr>
<tr>
<td>6.2</td>
<td>Experimental Results</td>
<td>133</td>
</tr>
<tr>
<td>7.1</td>
<td>Terminology in Time Delay Equations</td>
<td>162</td>
</tr>
<tr>
<td>7.2</td>
<td>Input Values for Time Delay Equations</td>
<td>165</td>
</tr>
<tr>
<td>7.3</td>
<td>Terminology in Signaling Load Equations</td>
<td>167</td>
</tr>
<tr>
<td>7.4</td>
<td>Call Delivery Service Values in Signaling Load Equations</td>
<td>169</td>
</tr>
<tr>
<td>A.1</td>
<td>Agents in IP Multimedia Subsystem - I</td>
<td>190</td>
</tr>
<tr>
<td>A.2</td>
<td>Agents in IP Multimedia Subsystem - II</td>
<td>191</td>
</tr>
<tr>
<td>A.3</td>
<td>Agents in LB-IM Server</td>
<td>192</td>
</tr>
<tr>
<td>A.4</td>
<td>Agents in CBS Server</td>
<td>193</td>
</tr>
<tr>
<td>B.1</td>
<td>Attacks on IMS System classified by CASE I - I</td>
<td>195</td>
</tr>
<tr>
<td>B.2</td>
<td>Attacks on IMS System classified by CASE I - II</td>
<td>196</td>
</tr>
</tbody>
</table>
# List of Figures

2.1 Overall mobile telecommunication network Architecture ........................................... 11  
2.2 Core Network Organization ...................................................................................... 14  
2.3 Call Delivery Service ............................................................................................... 16  
3.1 Signal flow for Call Forwarding Service based on Email ......................................... 26  
3.2 Schematic representation of Abstract Model elements ............................................... 27  
3.3 Abstract Model of service nodes in the CS domain ................................................. 28  
3.4 Abstract Model of service nodes in the CFS ......................................................... 30  
3.5 Abstract model in Signal flow for call delivery service .......................................... 32  
3.6 Sample Cascading attack on call delivery service ................................................. 34  
3.7 Cross Infrastructure Cyber attack using abstract model ......... 35  
4.1 Architecture of CAT ............................................................................................... 50  
4.2 Work Flow of Vulnerability Assessment .................................................................... 51  
4.3 Threat Model ......................................................................................................... 52  
4.4 Example of CEFSM and SDL ................................................................................. 53  
4.5 Speech Attack ....................................................................................................... 56  
4.6 Attack Graph of Speech Attack ............................................................................. 58  
4.7 Time Analysis: Performance Graphs ........................................................................ 74  
5.1 Architecture of aCAT ............................................................................................. 79  
5.2 Network Dependency Model .................................................................................. 80  
5.3 Looping Example .................................................................................................. 96  
5.4 Attack Graph for Attack Alerting .......................................................................... 98  
5.5 Alerting Attack ...................................................................................................... 99  
5.6 Attack Graph for Power Off Power On Attack .................................................. 100  
5.7 Power Off Power On Attack .................................................................................. 101  
5.8 Attack Graph for Mixed Identity Attack .............................................................. 102  
5.9 Mixed Identity Attack ........................................................................................... 104
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.10</td>
<td>Attack Graph for Call Redirection Attack</td>
<td>105</td>
</tr>
<tr>
<td>5.11</td>
<td>Call Redirection Attack</td>
<td>106</td>
</tr>
<tr>
<td>5.12</td>
<td>Attack Graph for Missed Calls</td>
<td>107</td>
</tr>
<tr>
<td>5.13</td>
<td>Missed Calls Attack</td>
<td>108</td>
</tr>
<tr>
<td>5.14</td>
<td>Experimental Results</td>
<td>109</td>
</tr>
<tr>
<td>6.1</td>
<td>Signal Flow for Call Delivery Service</td>
<td>116</td>
</tr>
<tr>
<td>6.2</td>
<td>Architecture of MAPsec</td>
<td>116</td>
</tr>
<tr>
<td>6.3</td>
<td>Fragment of Unmarked Attack Graph for Alerting Attack generated by aCAT</td>
<td>118</td>
</tr>
<tr>
<td>6.4</td>
<td>Fragment of Marked Attack Graph for Alerting Attack</td>
<td>119</td>
</tr>
<tr>
<td>6.5</td>
<td>Fragment of Marked Attack Graph for SM Attacks</td>
<td>127</td>
</tr>
<tr>
<td>6.6</td>
<td>Fragment of Marked Attack Graph for SS Attacks</td>
<td>129</td>
</tr>
<tr>
<td>6.7</td>
<td>Fragment of Marked Attack Graph for Authentication Material Attacks</td>
<td>131</td>
</tr>
<tr>
<td>7.1</td>
<td>Signal Flow for Call Delivery Service</td>
<td>142</td>
</tr>
<tr>
<td>7.2</td>
<td>Network Model</td>
<td>144</td>
</tr>
<tr>
<td>7.3</td>
<td>Case 1: Corruption on the link</td>
<td>146</td>
</tr>
<tr>
<td>7.4</td>
<td>Case 2: Corruption by the originator</td>
<td>147</td>
</tr>
<tr>
<td>7.5</td>
<td>Case 3: Corruption by derivators</td>
<td>147</td>
</tr>
<tr>
<td>7.6</td>
<td>Case 4: Corruption by Mediators</td>
<td>147</td>
</tr>
<tr>
<td>7.7</td>
<td>Case 5: Collusion based Mis-routing</td>
<td>148</td>
</tr>
<tr>
<td>7.8</td>
<td>Sample EndSec Message Construction</td>
<td>151</td>
</tr>
<tr>
<td>7.9</td>
<td>Originator/Derivator Check Phase</td>
<td>154</td>
</tr>
<tr>
<td>7.10</td>
<td>Node Impersonation not eliminated by Nested Path</td>
<td>156</td>
</tr>
<tr>
<td>7.11</td>
<td>Node Impersonation Eliminated by Nested Path</td>
<td>157</td>
</tr>
<tr>
<td>7.12</td>
<td>Time Delay in various schemes</td>
<td>166</td>
</tr>
<tr>
<td>7.13</td>
<td>Protocol Stacks for TCAP, ISUP, MAP</td>
<td>170</td>
</tr>
<tr>
<td>7.14</td>
<td>Signaling Load in various schemes</td>
<td>171</td>
</tr>
<tr>
<td>A.1</td>
<td>Abstract Model of service nodes in the IP Multimedia subsystem</td>
<td>189</td>
</tr>
<tr>
<td>A.2</td>
<td>Abstract Model of service nodes in the LB-IM</td>
<td>192</td>
</tr>
</tbody>
</table>
A.3 Abstract Model of service nodes in the CBS . . . . . . . . . . . . . . . . 193
Acknowledgments

It has been my privilege to work with my advisers Dr. Thomas F. LaPorta, and Dr. Peng Liu. I could never express enough gratitude for their enormous patience, guidance, and financial support. By showing through example, they taught me numerous academic and non-academic lessons that will remain with me for the rest of my life. They have particularly inspired me to be persevering and persistent.

Many thanks to my thesis committee members Dr. Trent Jaeger and Dr. Carleen Maitland, who took time of their busy schedules to give me insightful comments and valuable feedback. The memorable discussions I shared with many of my friends and colleagues have been very enlightening, and I wish to extend my thanks to them all. For the valuable life lessons I learned from my sister (Krishna Priya), I thank her.

This dissertation would not be possible if not for the complete support of my husband (Rajashekar). For his tolerance of my insanities, infinite patience, complete support, and more, I am forever indebted to him.

Most importantly, this dissertation is a dedication to my parents (Aruna & Prabhakar) absolute devotion, and unfailing faith in their children. Their support in our life’s choices means more to us than we can ever express. For this and much more, I am eternally grateful.
Chapter 1

Introduction

Mobile telecommunication networks are high speed, high capacity, voice and data communication networks with enhanced multimedia and seamless roaming capabilities for supporting mobile phones. In recent years, we have seen a tremendous growth in these networks, so much so that the number of mobile phone users are now greater than the number of land-line phone users. Mobile telecommunication networks support mobile phones not just for routine phone calls, but also for conducting business transactions such as e-trade, connecting to the Internet for email, and most importantly for emergency and mission critical services such as E-911. Today, these networks have become the lifeline of communications.

A breakdown in the mobile telecommunication network has many adverse effects ranging from huge economic losses due to financial transaction disruptions; loss of life due to loss of phone calls made to emergency workers; and communication outages during emergencies such as, September 11. Therefore, it is high priority for the mobile telecommunication network to function accurately.

It must be noted that it is not difficult for unscrupulous elements to break into the mobile telecommunication network and cause outages. The major reason being that mobile telecommunication networks were not designed with security in mind. They evolved from the old-fashioned telephone networks that were built for performance. To this day, the mobile telecommunication network has numerous well-known and unsecured vulnerabilities providing access to adversaries. Another feature of mobile telecommunication networks are network relationships (also called dependencies) that cause certain types of errors to propagate to other network locations as a result of regular network activity. Such propagation can be very disruptive to the network, and in turn affect subscribers. Finally, Internet connectivity to the mobile telecommunication network is another major contributor to the mobile telecommunication network vulnerability as it
gives Internet users direct access to mobile telecommunication network vulnerabilities from their homes.

To ensure that adversaries do not access the network and cause breakdowns, a high level of security must be maintained in the mobile telecommunication network. However, while great efforts have been made to improve the mobile telecommunication network in terms of support for new and innovative services, greater number of subscribers, higher speed, and larger bandwidth, very little has been done to update the security of the mobile telecommunication network. Accordingly, these networks have become highly attractive targets to adversaries, not only because of their lack of security, but also due to the ease with which these networks can be exploited to affect millions of subscribers.

In the following, we present some illustrating examples of attacks caused by exploiting the numerous vulnerabilities in the network infrastructure.

- Cloning attacks where un-subscribed mobile devices impersonate legitimate customers and use resources causing network overload, and denial of service to genuine customers [2].

- Rerouting calls, eavesdropping, masquerading and gaining access to unauthorized data [2] by manipulating of signaling messages and impersonating service nodes.

- Denial of service attacks on voice networks especially e-911 networks using Internet based Worms [46], text messages [23], and redirection [38].

- Hijacking of incoming or outgoing calls for industrial espionage [38].

- Cascading attacks where the adversary can attack from a remote location using subtle attack actions [38].

- Wiretapping to eavesdrop on mobile phone conversations [59].

These attacks while but a sample of possible attacks on the network illustrate the highly vulnerable nature of the network and the high degree of threat to communications. The vulnerable nature of the network has been a major motivation behind this work. The goal of this dissertation is to assess the current state of security of the mobile
telecommunication network, quantify it, and determine the network characteristics that must be changed to render the network more secure. In our assessment, we: (1) identify and classify possible threats to mobile telecommunication networks; (2) assess potential vulnerabilities that may be exploited to realize these threats; and (3) devise defense strategies to prevent attacks.

1.1 Contributions

In this section, we outline our three part contribution to security assessment of mobile telecommunication networks.

1.1.1 Attack Taxonomy

As part of our first step in security assessment, we systematically identified vulnerabilities that may be exploited to launch attacks. Such systematic identification helped us evaluate the current state of security, and extract common characteristics of attacks, and aid in better design of security protocols.

In conducting our systematic identification, we developed: (1) an abstract model to isolate vulnerable network components, and thereby detect starting point of attacks; and (2) a taxonomy to classify the attacks, and vulnerabilities, thereby extracting the common characteristics of attacks.

Our abstract model is essentially an atomic model of each network server (also called service node). Every service node performs a number of sophisticated functions. Hence, in our abstract model we breakdown the sophisticated functionality of each service node into multiple atomic parts, with each atomic part associated with a single function. From the atomic parts, we extract vulnerable areas, attack origins, and other attack characteristics. It must be noted that it is not easy to develop the abstract model as it requires a great understanding of the working of mobile telecommunication network.

Our abstract model has lead us into two insightful observations. The first being the discovery of the unique feature of attacks on mobile telecommunication networks called the cascading effect. In the cascading effect, corruption caused by an adversary, automatically propagates to remote network location due to normal network activity.
This cascading effect is made possible due to the relationships (also called dependencies) between the various components within the mobile telecommunication network. An example of the cascading effect, is an adversary preventing a police officer or an emergency health-care provider from receiving an E-911 call by exploiting a vulnerability in a remote location.

The second being discovery of a new breed of attacks called the cross infrastructure cyber attack. The cross infrastructure cyber attack is an attack in which the mobile telecommunication network may be attacked using the Internet as a launching pad. This attack is a direct result of the cascading effect described previously.

Our abstract model has also been instrumental in the design of our attack taxonomy. From our abstract model, we understood that a typical taxonomy is insufficient to classify mobile telecommunication network attacks. Hence, we developed a unique three dimensional taxonomy to classify these attacks.

In our three dimensional taxonomy, we classify attacks based on the following three criteria: (1) the level of adversary’s access to the network when the attack is launched, for e.g., access to air interface; (2) the type of attack launched, for e.g., fabrication attack; and (3) the network vulnerability exploited to launch the attack, for e.g., message or data source vulnerability. Our taxonomy is unique not only due to its three dimensional structure, but also due to provision for the cross infrastructure cyber attack in the three dimensional criteria.

Our taxonomy is the first detailed study of this type. It follows from our taxonomy that attacks launched by exploiting the same type of vulnerability exhibit common characteristics. Hence, in designing a security scheme, if its design goal is restricted to specific type of vulnerability, the security scheme will be successful in protecting the network.

The merits of this research include: (1) developing the comprehensive abstract model of network components; (2) identifying the unique cascading effect of attacks; (3) identifying the cross infrastructure cyber attack; and (4) developing the three dimensional taxonomy.
The technical impact of this work is that it has created an increased awareness of the vulnerability of the mobile telecommunication network and the possible attacks it faces. The results of this study were used as input to a working group of the Presidents National Security Telecommunications Advisory Panel. Further details of this abstract model and taxonomy may be obtained in Chapter 3.

1.1.2 Vulnerability Assessment

This part directly follows from our observations from the attack taxonomy. Regardless of how attacks are launched, the common feature of most attacks in the mobile telecommunication network is its cascading effect, i.e. the propagation of the attack to remote network areas. Hence, our interest in assessing potential vulnerabilities is to determine, not only the vulnerabilities that may be exploited to launch the attack, but also the cascading effects that result from such an attack.

To determine the vulnerabilities, and their corresponding cascading effects, we developed automated vulnerability assessment toolkits Cellular Network Vulnerability Assessment Toolkit - CAT, and Advanced Cellular Network Vulnerability Assessment Toolkit - aCAT. These toolkits allow users to input attack pre-conditions or vulnerabilities, and output a visual trace of the effects of the attack through the network in the form of an attack graph.

An attack graph is a diagrammatic representation of an attack on a real system. It shows the various ways an adversary may use to break-in to a system or cause corruption, and the various ways in which the corruption may propagate to remote parts of the network. Attack graphs are typically produced manually by red teams, and used by system administrators for protection. Our attack graph allows users to trace the effect of an attack through the network and determine its side-effects and ultimate service disruption.

As the mobile telecommunication network is at the nascent stage of development with respect to security, it is necessary to evaluate security protocols before deploying them. Hence, we extended our toolkit with security protocol evaluation capabilities. Our newly extended toolkit called Cellular Network Vulnerability Assessment Toolkit for
evaluation- eCAT allows users to quantify the benefits of security solutions by removing attack effects from attack graphs based on the defenses provided.

It must be noted that developing such a toolkits presented many challenges: (1) mobile telecommunication networks are extremely complex systems; they are comprised of several types of service nodes and control protocols, contain hundreds of data elements and support hundreds of services; hence developing such toolkits requires in-depth working knowledge of these systems; also (2) every deployment of the telecommunication network comprises of a different physical configuration; toolkits must be immune to the diversity in physical configuration; and finally (3) attacks cascade in the network due to normal network activities as a result of dependencies; toolkits must be able to track how the dependencies cause the corruption to cascade.

We overcome challenge of in-depth telecommunication knowledge by incorporating our toolkits with telecommunication specifications that are defined by the Third Generation Partnership Project (3GPP) and are available at no charge at [8].

The Third Generation Partnership Project (3GPP) is a telecommunications standards body formed to produce, maintain, and develop globally applicable ‘technical specifications and technical reports’ for a third generation mobile system based on evolved GSM core networks and the radio access technologies that they support.

These specifications detail the functional behavior and not the implementation structure of the mobile telecommunication network. They are written using simple flow-like diagrams called the Specification and Description Language (SDL) [22]. Hence we refer to these specifications as SDL specifications. Equipment and service providers use these SDL specifications as the basis of their service implementations.

We track the propagation of corruption by incorporating our toolkits with a novel dependency and propagation model to trace the propagation of data or logic corruption through the network. Finally, we superimposed boolean properties on the propagation model to capture the impact of security solutions. In the following we outline each toolkit.
Cellular Network Vulnerability Assessment Toolkit (CAT): CAT is the first version of the toolkit we developed for vulnerability assessment. CAT works by taking user input of either seeds (data item directly corrupt by the adversary and, whose cascading effect leads to a goal), or goals (data parameters that are derived incorrectly due to the direct corruption of seeds by the adversary) and uses SDL specification to uncover attacks, and their effects. CAT presents these attacks in the form of easy to read attack graphs. Further details on CAT may be found in Chapter 4.

Advanced Cellular Network Vulnerability Assessment Toolkit (aCAT): While CAT is the first step in detecting cascading attacks, CAT is dependent on the quality of SDL specification. However, SDL is limited in its expression of relationships and in-explicit in its assumptions, hence CAT identifies an incomplete set of attack effects.

We alleviate these inadequacies in SDL by adding new features to CAT. CAT with these new enhanced features is called Advanced CAT (aCAT). The new features added to aCAT include: (1) a network dependency model that explicitly specifies the exact dependencies in the network; (2) infection propagation rules that identify the reasons that cause corruption to cascade; and (3) a small amount of expert knowledge. The network dependency model and infection propagation rules may be applied to SDL specifications and help alleviate its limited expression capability. The expert knowledge helps capture the in-explicit assumptions made by SDL.

In applying these features, aCAT captures all those relationships which were previously unknown to CAT. In capturing these new relationships aCAT is able to detect a complete set of cascading effects. We also developed a new set of algorithms, the forward chaining, the reverse chaining and the combinatory algorithms. Unlike CAT, aCAT requires either seeds or goals to identify attacks.

Through extensive testing of aCAT, we found several interesting attacks and have been able to pin-point exact technical areas where SDL is lacking. Hence, aCAT produces an exhaustive and complete set of attacks for a given user input. Further details on aCAT may be found in Chapter 5.
**Cellular Network Vulnerability Assessment Toolkit for evaluation (eCAT):**

While aCAT can detect attacks, and all their cascading effects, aCAT cannot quantify the benefits of the security solutions. eCAT uses boolean probabilities in attacks graphs to detect whether a given security protocol can eliminate a certain cascading effect. Given a security protocol eCAT can measure effective coverage, identify the types of required security mechanisms to protect the network, and identify the most vulnerable network areas.

We used eCAT to evaluate a new standardized telecommunications security protocol called MAPSec. Our results show that MAPSec provides defenses for a narrow set of attacks, but does not effectively mitigate attacks with the largest potential impact. In fact, we have shown that attacks that have previously occurred would not be prevented by the current versions of MAPSec. Our analysis with eCAT and MAPSec have shown that corrupt data sources and service logic in network servers lead to the most damaging attacks. Further details may be found in Chapter 6.

The contributions of this research include: (1) toolkit to determine the possible cascading effects for a given attack; (2) toolkit to determine possible attack sources for a given cascading effect; (3) toolkit to evaluate the protection impact of a security scheme before its deployment; and (4) discovering that data sources and service logic in network servers lead to most damaging attacks.

The technical impact of this work is that it is now possible to uncover vulnerabilities in mobile telecommunication networks, to determine their potential impact, and to evaluate solutions to fix them.

**1.1.3 Defense Strategy**

This part of our research directly follows from our observation that data sources and service logic in service nodes lead to most damaging attacks. To prevent the most damaging cascading, i.e. cascading caused due to corrupt data sources and service logic, we developed a new security protocol called *End-to-End Security (EndSec)*. EndSec can not only prevent corruption from cascading but can also provide end-to-end security for signaling messages.
From our understanding of network dependencies, we designed EndSec to insert cryptographic checks on signaling message data items. This not only allows detection of corruption but also allows compromised service nodes to be traced. EndSec also defines several control messages to handle and repair corruption. EndSec is unique in that it brings accountability into the network previously non-existent, and enables real-time tracking of adversary activity. To the best of our knowledge, EndSec is the first protocol to provide end-to-end security for signaling messages.

The merits of EndSec include: (1) ability to self-diagnose and repair most corruption; (2) lightweight approach to detect corruption and compromised service nodes using the freely available SDL specifications; and (3) the provision for multi-hop security of signaling messages.

The merits of this scheme is that it provides for multi-hop security. This is the first scheme of its kind. The technical impact of EndSec is that it brings accountability to the network previously non-existent, and enables real-time tracking of adversary activity. Further details may be found in Chapter 7.

1.2 Thesis Outline

This thesis is organized as follows. In Chapter 2, we present a background on the mobile telecommunication network, and the related security issues that motivate our research. In Chapter 3, we present our abstract model and taxonomy of attacks. In Chapters 4, 5 and 6, we present our toolkit. In Chapter 7, we present our security scheme Endsec. Finally, in Chapter 8, we summarize our dissertation, and possible future work.
Chapter 2

Background

In this chapter, we present a detailed background on the mobile telecommunication network architecture and its many security issues.

The mobile telecommunication network is evolved from the early generation mobile telecommunication networks. These early generation networks were built for optimal performance, they were proprietary, and owned by reputed organizations. These networks were considered to be secure due to their proprietary ownership, and their closed nature, i.e. their control infrastructure was unconnected to any public network (such as the Internet) to which end subscribers had direct access. Security was a non-issue in the design of these networks.

Recently, connecting the Internet to the mobile telecommunication network, has not only imported the Internet vulnerabilities to the mobile telecommunication network, but also given end subscribers direct access to the control infrastructure of the cellular network thereby opening the network. Also, with the increasing demand for these networks, a large number of new network operators have come into the picture. Thus, the current mobile telecommunication environment is no longer a safe closed network, but an insecure open network with many unknown network operators having non-proprietary access to the network.

In the following, we briefly describe the architecture of the mobile telecommunication network, and its current state of security.

2.1 System Overview

In this section, we give an overview of the mobile telecommunication telecommunication system.
2.1.1 Overall Architecture

Subscribers gain access to the mobile telecommunication network via radio signals enabled by the radio access network (also called the *air interface*), as shown in Fig. 2.1. The radio access network is connected to the wireline portion of the network also called the *core network*. Core network functions include servicing subscriber requests, and routing traffic. The core network is also connected to the Public Switched Telephone Network (PSTN) and the Internet as illustrated in Fig. 2.1 [7].

The PSTN is the circuit-switched public voice telephone network that is used to deliver voice telephone calls on the fixed landline telephone network. The PSTN uses the “Signaling System No. 7 (SS7)”, a set of telephony signaling protocols defined by the International Telecommunication Union (ITU), for performing telephony functions such as call delivery, call routing and billing. The SS7 protocols provide a universal structure for telephony network signaling, messaging, interfacing, and network maintenance. PSTN connectivity to the core network enables mobile subscribers to call fixed network subscribers and vice versa. In the past, PSTN networks were also closed networks as they were unconnected to other public networks.
The core network is also connected to the Internet. Internet connectivity allows the mobile telecommunication network to provide multi-media services such as weather reports, stock reports, sports information, chat, and electronic mail. Interworking with both PSTN and Internet is possible using structures such as: protocol gateways that convert disparate traffic between the networks; federated databases that allow communication between the networks via their data sources, and the centralized multi-protocol mobility managers [48] that have protocol translation capabilities.

Interworking with the Internet has created a new generation of services - called Cross Network. Cross Network Services are multi-vendor, multi-domain and use a combination of Internet-based data and data from the mobile telecommunication network to provide a variety of services to the mobile subscriber. A sample Cross Network Service is the Email Based Call Forwarding Service (CFS) which uses Internet based Email data (in a Mail Server) to decide on the call forward number (in a Call Forward Server) and delivers the call via the cellular network.

From the functional viewpoint, the core network may also be further divided into the Circuit Switched (CS) domain, the Packet Switched (PS) domain, and the IP Multimedia Subsystem (IMS). Deployment of the IMS requires the deployment of the PS domain [7].

The CS domain offers circuit switched type of connections for user traffic and related signaling, and is used to access the PSTN. The PS domain offers packet switched type of connection for user traffic and related signaling, and is used to access the Internet. The IMS enables the network operators to offer multimedia services based on and built upon Internet applications, services and protocols. The IMS is the ultimate platform on which to provide IP Services to its subscribers. The IMS uses the Internet Protocol to transport information, and Session Initiation Protocol (SIP) and Mobile IP for session control and mobility management.

In the following section, we further discuss the internal core network organization.
2.1.2 Core Network Organization

Cellular networks are organized as collections of interconnected network areas, where each network area covers a fixed geographical region (as shown in Fig. 2.2). Every subscriber is affiliated with two networks: the home network, and the visiting network.

Every subscriber is permanently assigned to the home network from where they may roam on to other visiting networks. The home network maintains the subscriber profile, and current subscriber location. The visiting network is the network where the subscriber is currently roaming. It provides radio resources, mobility management, routing and services for roaming subscribers. The visiting network provides service capabilities to the subscribers on behalf of the home environment [2].

The core network is facilitated by network servers (also called service nodes). Service nodes are comprised of: (1) a variety of data sources (such as cached read-only, update-able, and shared data sources) to store data such as subscriber profile; and (2) service logic to perform functions such as computing data items, retrieving data items from data sources, and so on.

Service nodes may be of different types; each type assigned specific functions. The major service node types in the CS domain include the Home Location Register (HLR), the Visitor Location Register (VLR), the Mobile Switching Center (MSC), and the Gateway Mobile Switching Center (GMSC) [5].

All subscribers are permanently assigned to a fixed HLR located in the home network. The HLR stores permanent subscriber profile data and relevant temporary data such as current subscriber location (pointer to VLR) of all subscribers assigned to it. Each network area is assigned a VLR. The VLR stores temporary data of subscribers currently roaming in its assigned area; this subscriber data is received from the HLR of the subscriber. Every VLR is always associated with a MSC. The MSC acts as interfaces between the radio access network and the core network. It also handles circuit switched services for subscribers currently roaming in its area. The GMSC is in charge of routing the call to the actual location of the mobile station. Specifically, the GMSC acts as interface between the fixed PSTN network and the cellular network. The radio
access network comprises of a transmitter, receiver, and speech transcoder called the base station (BS) [20].

![Core Network Organization](image)

**Fig. 2.2. Core Network Organization**

Service nodes are geographically distributed and service the subscriber through collaborative functioning of various network components. Such collaborative functioning is possible due to the network relationships (called as dependencies). A dependency means that a network component must rely on other network components to perform a function. For example, there is a dependency between service nodes to service subscribers. Such a dependency is made possible through signaling messages containing data items. Service nodes typically request other service nodes to perform specific operations by sending them signaling messages containing data items with pre-determined values. On receiving signaling messages, service nodes realize the operations to perform based on values of data items received in signaling messages. Further dependencies may exist between data items, so that received data items may be used to derive other data items. Several application layer protocols are used for signaling messages. Examples of signaling message protocols include Mobile Application Part (MAP), ISDN User part (ISUP), and Transaction Capabilities Application Part (TCAP) protocols.
Typically in the cellular network, to provide a specific service, a pre-set group of signaling messages are exchanged between a pre-set group of service node types. The pre-set groups of signaling messages indicates the operations to be performed at the various service nodes, and is called a signal flow. In the following, we use the call delivery service [3], to illustrate a signal flow, and show how the various geographically distributed service nodes function together.

2.1.3 Call Delivery Service

The call delivery service is a basic service in the circuit switched domain. It used to deliver incoming calls to any subscriber with a mobile device regardless of their location. The signal flow of the call delivery service is illustrated in Fig. 2.3. The call delivery service signal flow is comprised of MAP messages - SRI, SRI.ACK, PRN, and PRN.ACK; ISUP message - IAM; and TCAP messages - SIFIC, Page MS, and Page.

Fig. 2.3 is illustrative of the exchange of signal messages between different network areas. It shows that when a subscriber makes a call using his mobile device, the call is sent in the form of a signaling message IAM to the nearest GMSC, which is in charge of routing calls and passing voice traffic between different networks. This signaling message IAM contains data items such as called number that denotes the mobile phone number of the subscriber receiving this call. The called number is used by the GMSC to locate the address of the HLR (home network) of the called party. The GMSC uses this address to send the signaling message SRI.

The SRI message is an intimation to the HLR, of the arrival of an incoming call to a subscriber with called number as mobile phone number. It contains data items such as the called number, and alerting pattern. The alerting pattern denotes the pattern (packet switched data, short message service, or circuit switched call) used to alert the subscriber receiving the call. The HLR uses the called number to retrieve from its database the current location (pointer to VLR) of the subscriber receiving the call. The HLR uses this subscriber location to send the VLR the message PRN. The PRN message is a request for call routing information (also called roaming number) from the VLR where the subscriber
is currently roaming. The PRN message contains the *called number*, *alerting pattern*, and other *subscriber call profile data items*.

The VLR uses the *called number* to store the *alerting pattern*, *subscriber call profile data items*, and assign the *roaming number* for routing the call. This roaming number data item is passed on to the HLR (in message PRN_ACK), which forwards it to the GMSC (in message SRI_ACK). The GMSC uses this *roaming number* to route the call (message IAM) to the MSC where the subscriber is currently roaming. On receipt of the message IAM, the MSC assigns the called number resources for the call and also requests the *subscriber call profile data items, and alerting pattern* for the *called number* (using message SIFIC) from the VLR, and receives the same in the Page MS message. The MSC uses the *alerting pattern* in the *incoming call profile* to derive the *page type* data item. The *page type* data item denotes the manner in which the subscriber mobile is to be paged to alert them of the call. It is used in the message Page. Thus subscribers receive incoming calls irrespective of their locations in the network.
If data item values are inaccurate then the network can mis-operate and subscribers are affected. Hence, accurate functioning of the network is greatly dependent on the integrity of data item values. Thus, signal flows allow the various service nodes to function together ensuring that the network services its subscribers effectively.

2.2 Current State of Security

In this section, we present the current state of the mobile telecommunication network security. As the security of the mobile telecommunication network is the security of each aspect of the network i.e., radio access network, core network, Internet connection, and PSTN connection, we detail security and possible attacks of each in detail.

2.2.1 Security in Radio Access Network

The radio access network uses radio signals to connect the subscribers mobile device with the core wireline network. Hence, it would seem that attacks on the radio access network could easily happen as anyone with a transmitter/receiver could capture these signals [6]. This was very true in the case of early generation mobile telecommunication network (first generation & second generation) where there were no guards against eavesdropping on conversations between the mobile device and BS; cloning of mobile devices to utilize the network resources without paying, such as illustrated by S. Patel [57], and shown by M. Notare et. al. [53]; and cloning BSs (as mentioned in Howard et al. [27]) to entice users to camp at the cloned BS, in an attack called false base station attacks, so that the target user provides secret information to the adversary in attacks similar to the partitioning attack shown by Rao et. al. [61].

In the current generation (third generation) mobile telecommunication network, all the above-mentioned attacks can be prevented as the network provides adequate security measures. Eavesdropping on signals between the mobile device and BS is not possible as cipher keys are used to encrypt these signals. Likewise replay attacks on radio signals are voided by the use of non-repeated random values. Usage of integrity keys on radio conversations voids the possibility of deletion and modification of conversations between mobile devices and BSs. By allowing the subscriber to authenticate the network
and vice versa, this generation voids the attacks due to cloned mobile devices and BSs. Finally as the subscribers identity is kept confidential, by only using a temporary subscriber identifier on the radio network, it is also possible to maintain subscriber location privacy [6].

However, the current generation still cannot prevent a denial of service attack from occurring if a large number of registration requests are sent via the radio access network (BS) to the visiting network (MSC). Such a denial of service attack is possible as the MSC cannot realize that the registration requests are fake until it attempts to authenticate each request and the request fails. To authenticate each registration request, the MSC must fetch the authentication challenge material from the corresponding HLR. As the MSC is busy fetching the authentication challenge material, it is kept busy and the genuine registration requests are lost [6]. Overall there is a great improvement in the radio network security in the current third generation mobile telecommunication network.

2.2.2 Security in Core Network

While the current generation network has seen many security improvements in the radio access network, the security of the core network is not as improved. Core network security is the security at the service nodes, and security on links (or wireline signaling message) between service nodes.

With respect to wireline signaling message security, of the many wireline signaling message protocols, protection is only provided for the Mobile Application Part (MAP) protocol. The MAP protocol is the cleartext application layer protocol that typically runs on the security free SS7 protocol or the IP protocol. MAP is an essential protocol and it is primarily used for message exchange involving subscriber location management, authentication, call handling, etc. The reason that protection is provided for only the MAP protocol is because it carries authentication material and other subscriber specific confidential data, and therefore, its security was considered top priority and was standardized [15], [11], [4]. While protection for other signaling message protocols was also
considered important, its was left as an improvement for the next generation network [13].

Security for the MAP protocol is provided in the form of the newly proposed protocol called Mobile Application Part Security (MAPSec) when MAP runs on the SS7 protocol stack, or Internet Protocol Security (IPSec) when MAP runs on the IP protocol.

Both MAPSec and IPSec, protect MAP messages on the link between service nodes by negotiating security associations. Security associations comprise of keys, algorithms, protection profiles, and key lifetimes used to protect the MAP message. Both MAPSec and IPSec, protect MAP messages by providing source service node authentication, and message encryption to prevent eavesdropping, MAP corruption and fabrication attacks.

It must be noted that while MAPSec/IPSec are deployed to protect individual MAP messages on the link between service nodes, signaling messages typically occur as a group in a signal flow, and hence signaling messages must be protected not only on the link, but also in the intermediate service nodes. Also, the deployment of MAPSec and IPSec is optional, and hence if any service provider chooses to omit MAPSec/IPSecs deployment, the efforts of all other providers are wasted. Therefore, to completely protect MAP messages, MAPSec/IPSec must be used by every service provider.

With respect to wireline service nodes, while MAPSec/IPSec protects links between service nodes, there is no standardized method for protecting service nodes [6]. Remote and physical access to service nodes may be subject to operators security policy and hence could be exploited (insider or outsider) if the network operator is lax with security. Accordingly, the network suffers from the possibility of node impersonation as mentioned by Lee et. al. [42], corruption of data sources, and service logic attacks. For example, unauthorized access to the HLR could deactivate customers, or activate customers not seen by the building system. Similarly unauthorized access to the MSC could cause outages for a large number of users in a given network area.

Corrupt data sources or service logic in service nodes has the added disadvantage of propagating this corruption to other service nodes in the network [38], [40], [39] via signaling messages. This fact was recently confirmed by a security evaluation of mobile
telecommunication networks [40] that showed the damage potential of a compromised service node to be much greater than the damage potential of compromised signaling messages. Therefore, it is of utmost importance to standardize a scheme for protecting service nodes in the interest of not only preventing node impersonation attacks, but also preventing the corruption from propagating to other service nodes.

In brief, the current generation core network is lacking in security for all types of signaling messages, for MAP signaling messages in service nodes, and a standardized method for protecting service nodes.

### 2.2.3 Security Implications of Internet Connectivity

Internet connectivity introduces the biggest threat to the security of the mobile telecommunication network. This is because cheap PC based equipment with Internet connectivity can now access gateways connecting to the core network. Therefore, any attack possible in the Internet can now filter into the core mobile telecommunication network via these gateways. For example, Internet connectivity was the reason for the slammer worm to filter into E-911 service in Bellevue, Washington making it completely unresponsive [46]. Other attacks that can filter into the core network from the Internet include Spamming and Phishing of short messages [23].

[23], [74], [56] expect low bandwidth denial of service attacks to be the most damaging attacks brought on by Internet connectivity. These attacks demonstrate that by sending just 240 short messages/sec it is possible to saturate the mobile telecommunication network and cause the MSC in-charge of the region to be flooded and loose legitimate short messages. Likewise, they show that it is possible to cause a specific user to loose short messages by flooding them with a large number of messages causing a buffer overflow. Such denial of service attacks are possible as the short message delivery time in the mobile telecommunication network is much greater than the short message submission time using Internet web sites [74].

Also, as both short messages and voices services use the same radio channel, contention for these limited resources may still occur, and cause a loss of voice service. To avoid loss of voice services due to contention, separation of voice and data services on
the radio network was suggested [23]. However, such separation requires major standardization and overhaul of the network, and is therefore unlikely be implemented very soon. Other minor techniques such as queue management, and resource provisioning were suggested by [74].

While such solutions could reduce the impact of short message flooding, they cannot eliminate other types of low bandwidth, denial of service attacks such as attacks on connection setup and tear down of data services. The root cause for such denial of service attacks from the Internet to the core network was identified as the difference in the design principles of these networks. While the Internet makes no assumptions on the content of traffic and simply passes it on to the next node, the mobile telecommunication network identifies the traffic content, and provides a highly tailored service involving multiple service nodes for each type of traffic [56].

Until this gap is bridged, such attacks will continue, but bridging the gap itself is a major process, as either the design of the mobile telecommunication network must be changed to match the Internet design, or vice versa, which is unlikely to happen soon. Hence, a temporary fix would be to secure the gateways connecting the Internet and core network. As a last note, Internet connectivity not only filters attacks into the core network but also into the PSTN network. Hence PSTN gateways must also be guarded.

### 2.2.4 Security Implications of PSTN Connectivity

PSTN connectivity to the mobile telecommunication network allows calls between the fixed and mobile telecommunication networks. While the PSTN was a closed network, the security free SS7 protocol stack on which it is based was of no consequence. However, by connecting the PSTN to the core network that is in-turn connected to the Internet, the largest open public network, the SS7 based PSTN network has no security left [47].

As SS7 protocols are plaintext and have no authentication features it is possible to introduce fake messages, eavesdrop, cause denial of service by traffic overload, and incorrectly route signaling messages. Such introduction of SS7 messages into the PSTN network is very easily done using cheap PC based equipment. Attacks where calls for
800, 900 numbers were re-routed to 911 servers, so that legitimate calls were lost are documented in [44]. Such attacks are more so possible due to the IP interface of the PSTN service nodes, and web-based control of these networks.

As PSTN networks are to be outdated soon, there is no interest in updating these networks. So, these networks will remain security free until their usage is stopped [47].

So far, we addressed the security and attacks on each aspect of the mobile telecommunication network. But an attack that is common to all the aspects of the mobile telecommunication network is the cascading attack. In the following chapters, we detail the cascading attack and present vulnerability assessment techniques to identify the same.

In general threats to the mobile telecommunication network are currently not well studied. There is no existing classification of these attacks or papers that propose tools to counteract the same. Hence, we are justified in our research motivation to develop taxonomy for current and future attacks, tools to assess vulnerability, and develop defense strategies.
Chapter 3

Taxonomy of Attacks

3.1 Introduction

In this chapter, we present our first step in security assessment of mobile telecom-
munication networks. We systematically identify vulnerabilities exploited to launch at-
tacks. Our aim in such systematic vulnerability identification is to extract common
attack characteristics. Such common attack characteristics aid in better design of secu-
rity protocols.

To systematically identify vulnerabilities we developed: (1) an abstract model
to further study network functionality and aid in identifying vulnerabilities; and (2) a
unique three dimensional taxonomy to classify attacks and aid in extracting common
features of attacks.

Our abstract model is designed to show the functionality of individual service
nodes in atomic form. This allows us to study interactions between the various functional
components thereby isolating vulnerabilities. Our abstract model has been instrumental
in helping us uncover: (1) cascading attacks - a type of attack where the adversary can
launch an attack from a location far from the target; (2) cross infrastructure cyber attacks
- a new breed of attack where the mobile telecommunication network may be attacked
from the Internet.

Our abstract model has also been instrumental in the design of our three dimen-
sional taxonomy. We have used the vulnerabilities isolated from the abstract model to
classify attacks. We classify attacks based on: (1) the level of the adversary’s access to
the network when the attack is launched; (2) the type of attack launched; and (3) the
network vulnerability exploited to launch the attack. Our taxonomy is unique not only
due to its three dimensional structure, but also due to the inclusion of the cross infrastructure cyber attack. From our taxonomy we have also been able to extract common attack features.

The rest of the chapter is organized as follows. In Section 3.2, we present the preliminaries required to understand our approach, in Section 3.3, we explain our abstract model. We detail our findings from the abstract model in Section 3.4, present the attack taxonomy in Section 3.5, and detail attacks by classifying them using our taxonomy in Section 3.6. We conclude by discussing our findings from the taxonomy in Section 3.7.

3.2 Preliminaries

In this section, we present some preliminary information to aid in understanding our work.

3.2.1 Cross Network Services

With the introduction of IP technologies into the traditional mobile telecommunication network, a new hybrid service called the cross network service has emerged.

The cross network service inter-works with both the mobile telecommunication network as well as the Internet. They use a combination of Internet-based data and mobile telecommunication network data to provide services to the mobile subscriber. They are to be multi-vendor, multi-domain, and cater to a wide variety of needs.

To illustrate the utility of these cross network services, we present several samples of the cross network services. We continue to refer to these services throughout this chapter. In presentation of these cross network services, we assume that these services may be accessed only through interaction with the CS domain. This interaction will provide a migration path for CS networks to IMS and will likely be used for several decades as CS domains remain in existence. In the following, we present samples of our cross network services.

Call Forwarding Service based on email: In the Call Forwarding Service (CFS), the status of the Internet based subscribers e-mail inbox is used to trigger call forwarding
in the mobile telecommunication network. For example, calls from a certain peer group that have e-mails pending in the called party’s inbox for longer than a certain period of time are delivered to the mobile device; calls from a second peer group are delivered to the voice mail. Call delivery is via the mobile telecommunication network.

**Client Billing Service:** In the Client Billing Service (CBS), the caller id of an incoming call is used to trigger a client billing system, which forwards the call and bills the client appropriately. Lawyers or insurance agents may use such a service to bill clients for the telephone advice or services offered.

**Location Based Instant Messaging Service:** In the Location Based Instant Messaging Service (LB-IM), a location track request from a buddy triggers the mobile telecommunication network to locate the subscriber. This location is revealed only if the buddy belongs in a location visibility group and a time visibility group. After the location is known the requestor can begin instant messaging.

A location visibility group comprises of groups of peers where each group is permitted to locate their LB-IM buddy only when the subscriber is in a specific geographic area. Supervisors permitted to locate their subordinates at a factory location is an example of the location visibility group.

A time visibility group comprises of groups of peers where each group is permitted to locate of their LB-IM buddy only in a specific time period. Supervisors permitted to locate their subordinates on weekdays between 9 am and 5 pm is an example of the time visibility group.

### 3.2.2 Working of Call Forwarding Service

In this section, we illustrate the working of the email based call forwarding service using Fig. 3.1. The call forwarding service is essentially a continuation (shown in Fig. 3.1) of the call delivery service, as it is activated only when a call arrives for a subscriber at the MSC service node. The call is delivered by the mobile telecommunication network using the standard call delivery procedure (mentioned in Section 2.1). When the call arrives at the MSC it triggers email based the call forwarding service in the IP
network by requesting the HLR to authenticate the call forwarding server (using message ‘Authenticate’).

Once the CF Server is authenticated, the MSC requests the CF Server to forward the call according to the preferences set by the subscriber receiving the incoming call. The CF Server checks with the Mail Server, using *Get Emails* message, to see if there are any emails waiting from the caller pending for greater than a certain period of time. Based on the received emails, and subscriber preferences the CF server decides how to forward the call, and informs the MSC of the same. The MSC uses this number to forward the call.
### 3.3 Abstract Model

The abstract model dissects functionality of the mobile telecommunication network to the basic atomic level allowing to systematically isolate and identify vulnerabilities. Such identification of vulnerabilities allows attack classification based on vulnerabilities, and isolation of network functionality aids in extraction of interactions between network components thereby revealing new vulnerabilities and attack characteristics.

As service nodes in the mobile telecommunication network are comprised of sophisticated service logic that performs numerous network functions, the abstract model logically divides the service logic into basic atomic units called *agents* (represented by elliptical shape in Fig. 3.2). Each agent performs a single function. As service nodes also manage data, the abstract model also logically divides data sources into data units specific to the agents they support. The abstract model also divides the data sources into permanent (represented by rectangular shape in Fig. 3.2) or cached (represented by triangular shape in Fig. 3.2) from other service nodes.

![Abstract Model Elements](image)

**Fig. 3.2.** Schematic representation of Abstract Model elements

We developed abstract models for the circuit switched domain, IP multimedia subsystem, and the three sample cross network services. We present the abstract model for the circuit switched domain, and the call forwarding service in this chapter. Readers interested in the abstract model for the IP multimedia subsystem, client billing service, and call forwarding service may refer to Appendix A. In the following, we describe our abstract models.
**CS Domain:** The abstract model for the CS domain is illustrated in Fig. 3.3 and described in Table 3.1. It shows agents, permanent, and cached data sources for the CS service nodes. For example, the *subscriber locator agent* in the HLR, is the agent that tracks the subscriber location information. It receives and responds to location requests during an incoming call, and stores subscriber’s location every time they move. This location information is stored in the *location data source*. Likewise the functionality of the other agents may be understood from the description Table 3.1.

Fig. 3.3. Abstract Model of service nodes in the CS domain
### Table 3.1. Agents in Circuit Switched domain

<table>
<thead>
<tr>
<th>Service Node</th>
<th>Name of Agent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC</td>
<td>Session Control</td>
<td>Manages calls, and sessions on behalf of the user. Maintains session state in the session state data source.</td>
</tr>
<tr>
<td></td>
<td>Routing</td>
<td>Provides routing information required to route the call to the user, and maintains the routing data source.</td>
</tr>
<tr>
<td></td>
<td>Channel</td>
<td>Establishes contact between the mobile device, and maintains the channel access data source.</td>
</tr>
<tr>
<td></td>
<td>Paging</td>
<td>Maintains paging lists in the paging data source, and pages the base stations to locate the user.</td>
</tr>
<tr>
<td>VLR</td>
<td>Foreign Agent Profile Manager</td>
<td>Maintains visiting subscriber’s profile (CAMEL, supplementary services data and new subscribed services data such as the CFS and LB-IM) for the duration of the subscriber’s visit.</td>
</tr>
<tr>
<td></td>
<td>Foreign Locator</td>
<td>Maintains information on all mobile devices visiting its location area. It is the point of contact for the subscriber locator agent in the HLR.</td>
</tr>
<tr>
<td></td>
<td>Foreign Registration</td>
<td>Handles registration requests from all mobile devices visiting its location area. Also, invokes the foreign authenticator agent in the VLR, which will in-turn authenticates the subscriber.</td>
</tr>
<tr>
<td></td>
<td>Foreign Authenticator</td>
<td>Fetches the authentication keys (from the registration agent in the HLR) and authenticates the subscriber. It is also in charge of the authentication data cache.</td>
</tr>
<tr>
<td>HLR</td>
<td>Registration</td>
<td>When the subscriber enters a new location area, it is in-charge of providing the subscriber data (authentication, profile and subscribed service data) to foreign agent profile manager in the new VLR and canceling this data in the foreign agent profile manager in the old VLR.</td>
</tr>
<tr>
<td></td>
<td>Authenticator</td>
<td>Retrieves keys from the authentication data source and generates authentication material for all users subscribing to its network.</td>
</tr>
<tr>
<td></td>
<td>Subscriber Locator</td>
<td>Tracks the location of all the users to which it is assigned. It stores the subscribers location information in the location data source.</td>
</tr>
<tr>
<td></td>
<td>Subscriber Profile Manager</td>
<td>Manages subscriber profile data (access policy, roaming agreements), terminal data, CAMEL, supplementary services and new subscribed services data.</td>
</tr>
</tbody>
</table>
**Call Forwarding Service:** We also present the abstract model for the CFS in Fig. 3.4. The CFS comprises of two service nodes, the CF server and the mail server. The mail server receives and stores the subscriber’s e-mails and the CF server is in-charge of deciding how the call is to be forwarded based on the incoming e-mails and subscriber preferences. Table 3.2 describes the agents used in the CF Server and Table 3.3 describes the agents of the mail server.

![Call Forwarding Service Diagram](image)

*Fig. 3.4. Abstract Model of service nodes in the CFS*
Table 3.2. Agents in CF Server

<table>
<thead>
<tr>
<th>Name of Agent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscriber Parameter Manager</td>
<td>Manages subscriber settings.</td>
</tr>
<tr>
<td>Filtering</td>
<td>Filters data retrieved by the <em>CFS mail server agent</em> based on parameters set by a subscriber and instructs the <em>CFS mail server agent</em> to retrieve more information.</td>
</tr>
<tr>
<td>Mail Server</td>
<td>Interfaces the mail server and the call forwarding server. It queries the mail server based on subscriber settings and retrieves e-mail information.</td>
</tr>
<tr>
<td>Authenticator</td>
<td>Maintains subscriber and third party authentication data used to authenticate the subscriber.</td>
</tr>
</tbody>
</table>

Table 3.3. Agents in Mail Server

<table>
<thead>
<tr>
<th>Name of Agent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscriber Parameter Manager</td>
<td>Manages subscriber settings.</td>
</tr>
<tr>
<td>Mail Startup</td>
<td>Retrieves subscriber settings and mail server data and invokes the <em>mail transfer agent</em> when a subscriber starts the mail service.</td>
</tr>
<tr>
<td>Local Delivery</td>
<td>In-charge of e-mail that is to be delivered to a local user.</td>
</tr>
<tr>
<td>Mail Transaction</td>
<td>Interfaces queries and responses to and from the incoming mail storage (post office data source).</td>
</tr>
<tr>
<td>Mail User</td>
<td>Allows the users to compose and read e-mail messages. Acts as an interface between the user and the <em>mail transfer agent</em>.</td>
</tr>
<tr>
<td>Mail Update</td>
<td>Manages the state of the incoming mail storage in the post office data source.</td>
</tr>
<tr>
<td>Mail Transfer</td>
<td>Transfers e-mail messages from one computer to another. This agent is responsible for delivering outgoing mail and receiving incoming mail.</td>
</tr>
<tr>
<td>Authenticator</td>
<td>Responsible for authenticating subscribers wanting to access the mail server and other servers that may contact the mail server with a request.</td>
</tr>
</tbody>
</table>
3.4 Abstract Model Findings

The abstract model had lead to many interesting findings. We outline them as follows.

3.4.1 Interactions

To study the network interactions, service nodes in signal flows (e.g. call delivery service) were replaced by their corresponding abstract model agents and data sources. Such an abstract model based signal flow based on the call delivery service is shown in Fig. 3.5.

In studying the abstract model signal flow, it was observed that interactions happen: (1) between agents typically using procedure calls containing data items; (2) between agents and data sources using queries containing data items; and (3) between
agents belonging to different service nodes using signaling messages containing data items.

The common behavior in all these interactions is that it typically involves data items whose values are set, or modified in agents or data source, or it involves data items passed between agents, data sources, or agents and data sources. Hence, the value of a data item can, not only be corrupt in an agent or data source, but can also be easily passed on to other agents resulting in propagation of corruption. This propagation of corruption is called as the cascading effect, and attacks that exhibit this effect as cascading attacks. In the following, we present a sample of the cascading attack.

Sample Cascading Attack: In this sample cascading attack, cascading due to corrupt data items, and their ultimately service disruption is illustrated using Fig. 3.6. Consider the call delivery service explained previously. Here the adversary may corrupt the roaming number data item (used to route the call) in the VLR. This corrupt roaming number is passed on in message PRN_ACK to the HLR, which in-turn passes this information to the GMSC. The GMSC uses the incorrect roaming number to route the call to the incorrect MSC_B, instead of the correct MSC_A. This results in the caller losing the call, or receiving a wrong number call. Thus corruption cascades and results in service disruption.

The type of corruption that can cascade is system acceptable incorrect value corruption, a type of corruption in which corrupt values taken on system acceptable values, albeit incorrect values. Such a corruption can cause the roaming number to be incorrect but a system acceptable value.

Please note that it is easy to cause such system acceptable incorrect value corruption, due to the availability of websites [72, 71] that refer to proprietary working manuals of service nodes such as the VLR. Such command insertion attacks have become highly common place, the most infamous attack being the telephone tapping of the Greek government and top-ranking civil servants [59].

Cross Infrastructure Cyber Attacks: When attacks cross into the mobile telecommunication networks from the Internet, through cross network services, it is called cross
infrastructure cyber attacks. Cross infrastructure cyber attacks can also occur when cascading attacks cross from the Internet into the mobile telecommunication network. Such a cross infrastructure cyber attack is illustrated on the CFS using Fig. 3.7.

As the CFS forwards calls based on the emails received, corruption is shown to propagate from the mail server to a Call Forward (CF) server, and finally the MSC. In the attack, using any standard mail server vulnerabilities, the adversary may compromise the mail server and corrupt the e-mail data source by deleting e-mails from whom the victim is expecting calls. The CF server receives and caches incorrect e-mail from the mail server.

When calls arrive for the subscriber, the call forwarding service is triggered, and the MSC queries the CF Server on how to forward the call. The CF Server checks its incorrect e-mail cache, and as there are no e-mails from the caller, it responds to the MSC to forward the call to the voice mail when in reality the call should have been forwarded to the mobile device. Thus the effect of the attack on the mail server propagates to the mobile telecommunication network service node. This is a classic example of a cross
infrastructure cyber attack due to cascading, where the adversary gains access to the cross network server, and attacks by modifying data in the data source of the cross network server.

Please note that it has become highly simplified to launch such attacks due to the easy accessibility to the Internet, and the subscriber preference for Internet based cross network services.

3.4.2 Isolating Vulnerabilities

From the abstract model, the major network components that are vulnerable to adversaries are: (1) data sources; (2) agents (more generally called service logic); and (3) signaling messages. By exploiting each of these vulnerabilities, data items that are crucial to the correct working of the network can be corrupted, leading to ultimate service disruption through cascading effects.
Also the effect of corrupt signaling messages is different from the effect of corrupt data sources. By corrupting data items in a data source of a service node, all the subscribers attached to this service node may be affected. However, by corrupting a signaling message, only the subscribers (e.g. caller, and called party in case of call delivery service) associated with the message are affected. Likewise, corrupting the agent in the service node can affect all subscribers using the agent in the service node. Hence, in the three dimensional taxonomy, vulnerability exploited is considered as a attack dimension, as their affect of each vulnerability is different.

Likewise, the adversarys physical access to the network also affects how the vulnerability is exploited and how the attack cascades. For example, consider the case when the subscriber has access to the air interface. The adversary can only affect messages on the air interface. Similarly, if the adversary has access to a service node, then the data sources, and service logic may be corrupted. Hence, in the three dimensional taxonomy, the physical access is considered as a category as it effects how the vulnerability is exploited, and its ultimate effect on the subscriber.

Finally, the way the adversary chooses to launch the attack ultimately affects the service in a different way. Consider a passive attack such as interception. Here the service is not effected, but may have a later effect on the subscriber such as identity theft or loss of privacy. An active attack like interruption can cause complete service disruption. Hence, in the three dimensional taxonomy, the attack means is considered as a category due its ultimate effect on service.

In the next section, we detail the mobile telecommunication network specific three dimensional taxonomy and how the above mentioned dimensions are incorporated.

In the next section, we present our taxonomy and how the above mentioned are incorporated into our taxonomy.

3.5 Attack Taxonomy

In the design of our taxonomy, we incorporate findings from our abstract model. Hence, we consider the adversary’s physical access to the network, the category of the
attack, and the vulnerability exploited. On this basis, we classify attacks into three dimensions: Dimension I: Physical Access to the Network, Dimension II: Attack Categories and Dimension III: Vulnerability Exploited.

We also differentiate between single infrastructure and cross infrastructure cyber attacks. Single infrastructure attacks are attacks on a domain using the same network domain as a launching pad. In the following, we present each dimension in detail.

**Dimension I: Physical Access to the Network:** In this dimension, attacks are classified based on the level of physical access the adversary has to the mobile telecommunication network. Dimension I may be further classified into single infrastructure attacks (Level I-III) and cross infrastructure cyber attacks (Level IV-V):

- **Level I: Access to air interface with physical device:** Here, the adversary has access to standard inexpensive off-the-shelf equipment that could be used to impersonate parts of the network [27]. The adversary may put up a false base station. Victims camping on the false base station are subject to false base station attacks. Adversary’s may also use modified mobile devices to broadcast at a high frequency, eavesdrop and cause a man in the middle attack and correspond to attacks mentioned in Chapter .

- **Level II: Access to Cables connecting Central Offices (i.e. core service nodes):** Central offices house core service nodes, which are typically accessible to authorized personnel only. If the adversary has access to cables connecting these central offices, they may cause damage by disrupting normal transmission of signaling messages and correspond to attacks mentioned in Chapter .

- **Level III: Access to mobile telecommunication core network service nodes in the Central Office:** In this case the adversary may be a disgruntled employee or a terrorist who has managed to gain access into the central office. Here the adversary can cause damage by editing the service logic or modifying data sources, such as, subscriber data (profile, security and services) stored in the service node and correspond to attacks mentioned in Chapter .
- **Level IV: Access to Links connecting the Internet and the mobile telecommunication core network:** This is a cross infrastructure cyber attack. The adversary has access to links connecting the core service nodes and the Internet based cross network servers. In this case, the adversary can cause damage by deleting normal transmission of signaling messages traversing the link, and inserting signaling messages into the link between the two networks. Level IV may be subdivided based on the interworking approaches used to connect the mobile telecommunication core network and the Internet. Section ?? details the various interworking approaches and attacks.

- **Level V: Access to Internet Servers or Cross Network Servers:** This is a Cross Infrastructure Cyber Attack. In this case, the adversary can cause damage by editing the service logic, or modifying subscriber data (profile, security and services) stored in the Cross Network Servers. This level of attack is easier to achieve Level II and Level III.

**Dimension II: Attack Categories:** In this dimension, attacks are classified based on the type of attack. The attack categories are based on Stallings [67].

- **Interception:** The adversary intercepts information, for example, reads signaling messages on a cable (Level II), but does not modify or delete them. This is a passive attack. This affects the privacy of the subscriber and the network operator. The adversary may use the data obtained from interception to analyze traffic and eliminate the competition provided by the network operator.

- **Fabrication or Replay:** In this case the adversary may insert spurious objects into the system. These objects depend on the level of the adversary’s physical access to the system. For example, in a Level II, the adversary may insert fake signaling messages. In a Level III, the adversary may insert fake service logic or fake subscriber data into this system. The effects could result in the adversary masquerading as an authority figure.
• **Modification of Resources:** The adversary causes damage by modifying system resources. For example, in a Level II, the adversary may modify signaling messages in and out of the cable and in a Level III, the adversary may modify service logic or modify subscriber data in the entity.

• **Denial Of Service:** The adversary causes an overload or a disruption in the system such that network functions in an abnormal manner. The abnormal behavior could be legitimate subscribers not receiving service, illegitimate subscribers receiving service or the entire network may disabled as a result of the attack.

• **Interruption:** The adversary causes an interruption by destroying resources. For example, in a Level II, the adversary may delete signaling messages in and out of the cable and in a Level III, the adversary may delete a subscriber data in the entity such as an HLR and the adversary may not receive service.

**Dimension III: Vulnerability Exploited:** In this dimension, attacks are classified based on the vulnerability exploited to cause the attack. The vulnerability exploited is explained as follows.

• **Data:** The adversary attacks the data stored in the system. Damage is inflicted by modifying, inserting and deleting the data stored in the system.

• **Messages:** The adversary attacks the system through the signaling messages. The adversary may insert, modify, delete, and replay signaling messages going in and out of the network.

• **Service Logic:** The adversary inflicts damage by attacking the service logic running in the various mobile telecommunication core network service nodes. For example, an interruption attack on service logic would completely delete the logic running on an entity such as the MSC.
3.6 Attack Overview

In this section, we use our taxonomy to tabulate a list of possible attacks, and detail some interesting attacks. We present attacks with respect to the CS domain and call forwarding service. Readers interested in the IMS please refer to Appendix B. We group attacks as CASE 1: Dimension I-Physical Access Vs Dimension II-Attack Categories and CASE 2: Dimension II-Attack Categories Vs Dimension III-Attack Means. Note that the Dimension I vs. Dimension III CASE can be transitively inferred from CASE 1 and CASE 2. We present tabulated attacks using CASE 1 and CASE 2.

CS Domain: We tabulate a list of possible attacks on the CS Domain. Table 3.4 and Table 3.5 shows the CASE 1 tabulation of possible Single Infrastructure attacks on the CS domain. Table 3.6 shows the CASE I tabulation of possible Cross Infrastructure Cyber attacks. Table 3.7 shows the attacks classified by CASE II. Readers interested in attacks on the IMS and other interworking approaches may refer to Chapter B.

Table 3.4. Single Infrastructure attacks on CS domain classified by CASE I - I

<table>
<thead>
<tr>
<th>Categories</th>
<th>Level I</th>
<th>Level II</th>
<th>Level III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interception</td>
<td>Observe time, rate, length, source and destination of victims locations. Eavesdrop on victims with modified mobile devices.</td>
<td>Analyze traffic patterns, gather subscriber or company data. Eavesdrop on calls and voice signaling messages. Capture authentication keys sent from HLR to VLR and use in replay attacks.</td>
<td>Analyze traffic patterns, gather subscriber or company data arriving at the compromising entity. Gather data stored in HLR and sell it to competition. At MSC, track the subscribers activities, calls and the services they access.</td>
</tr>
<tr>
<td>Fabrication/</td>
<td>When victim’s camps at false base station, calls made by the victim may be hijacked and used to make fraudulent calls, while the victim is charged.</td>
<td>Send a large number of routing requests to the MSC thereby overloading. Send <strong>registration update</strong> messages to <strong>registration agent</strong> at HLR causing incorrect call routing and mobile device shut down. Send profile change messages to the <strong>subscriber profile manager agent</strong> at HLR: Subscriber may not receive services. Request AVs from the <strong>authenticator agent</strong> from the HLR and use it to replay attacks.</td>
<td>With access to HLR adversaries may add fraudulent subscribers to the <strong>profile setting, terminal data, subscribed services and CAMEL &amp; supplementary services</strong> data sources but not the <strong>billing data store</strong>, this way the fraudulent subscribers can access the services without paying. Insert new service logic to the compromised service node so that it is disabled at any time.</td>
</tr>
<tr>
<td>Insertion</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.5. Single Infrastructure attacks on CS domain classified by CASE I - II

<table>
<thead>
<tr>
<th>Categories</th>
<th>Level I</th>
<th>Level II</th>
<th>Level III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modification Of Resources</td>
<td>With modified base station and modified mobile device, the adversary can come between the target and the network</td>
<td>Change called party number in call request messages and the call is sent to incorrect party. Change the routing numbers in routing responses, so that the MSC under attack does not receive incoming calls. Change the AVs sent to authenticate the mobile device, so that the mobile device is never authenticated.</td>
<td>Change subscriber profile, authentication data, location data at the HLR. Change the location data mapping at registration agent of the HLR. Modify the routing numbers in the routing agents of the MSC so that calls cannot be routed to the session control agents.</td>
</tr>
<tr>
<td>Denial Of Service</td>
<td>When victims camp at false base stations, they are out of reach of signals from the visiting network and can no longer receive calls and other network related services.</td>
<td>Send multiple IAM messages to the MSC; exhaust routing numbers and overload it. Send a large number of authentication requests to authenticator agent at HLR, thereby slowing down the HLR and surrounding links. Send spoofed location update messages to registration agent at the HLR causing incorrect call routing, and shutting down of the mobile device. Changing profile settings at the subscriber profile manager agent at HLR.</td>
<td>Modify the authentication keys calculating algorithm such that no mobile device is authenticated and hence do not get service.</td>
</tr>
<tr>
<td>Interruption</td>
<td>Jam victims traffic channels so the victim cannot access the channel. Broadcast at a higher intensity than allowed hogging bandwidth.</td>
<td>Delete registration or location update messages at registration agent of the HLR resulting in incorrect call routing. Delete call requests, send authentication information requests.</td>
<td>Delete subscriber preferences. At authenticator agent of HLR, replace ciphering algorithm, so that mobile devices fail to authenticate.</td>
</tr>
</tbody>
</table>

We now detail some interesting examples of cross infrastructure cyber attacks outlined in Table 3.6.

**Call forwarding Service (CFS):** We detail some **Level IV attacks** on the CF Service in the following. In the intercept operation attack, the adversary may view and capture: (1) caller-ids of calls arriving at a victim subscriber; (2) call forward numbers and trace the
location of the victim subscribers; (3) authentication messages and use it for a replay attack at a later time; and (4) e-mails fetched by the CF Server from the Mail Server and gain access to victims personal information.

In the insertion or fabrication attack, the adversary may bombard the CF server with call forward requests and authentication requests to the CF and mail servers causing a denial of service. The adversary may also request e-mail messages from the mail server with the help of the authentication messages captured earlier.

In the modification attack, the adversary may modify data parameters, contained in the signaling messages such as: (1) caller id’s so that the calls are forwarded incorrectly; (2) call forward numbers so that calls are forwarded incorrectly; (3) authentication challenge responses to an incorrect value so that the CF and mail servers are never authenticated.

In the interruption attack, the adversary may delete all messages arriving at the CF or mail servers giving an impression that these servers are not functional.

In the following we present several Level V attacks on the CF Service. In the interception attack, the adversary may steal personal information of subscribers stored in the subscriber parameter data store of the CF or mail server.

In the insertion or fabrication attack, adversary’s may insert either unauthorized subscribers into the CF subscriber parameter data sources, allowing them to use network services without paying for it; or fake e-mail data into the e-mail data cache through the CFS mail server agent, resulting in calls forwarded incorrectly.

In the modification attack, the adversary may also have the capability to modify: (1) call forward numbers for the subscriber at the subscriber parameter data store; (2) time stamps of cached e-mail data; and (3) service logic in the CFS filtering agent and the CFS mail server agent, resulting in calls forwarded incorrectly.

Denial of service may be caused by changing the CF number so that the victim does not gain access to the voice message, and sending two or three call forward numbers to the session control agent at the MSC causing confusion resulting in the call being handled improperly.
In the interruption attack, the adversary may delete (1) certain target subscriber profiles in the data sources so that they may not receive CF service; (2) emails in the post office data store at the mail server so that calls are forwarded incorrectly; and (3) service logic at service nodes such as the CFS filtering agent so that they may be unable to provide any service.

Client Billing Service (CBS): Level IV attacks on the CB service may be enumerated as follows. In interception attacks, the adversary may view not only the caller id’s of calls to the victim and sell this information, but also the authentication messages and use it for a replay attack later on.

In the insertion or fabrication attack, the adversary may bombard the CB server with both bill client requests and authentication requests possibly causing a denial of service.

In the modification attack, the adversary may modify signaling message based data items, such as caller id’s so that the calls are forwarded incorrectly, and authentication challenge response’s to incorrect values so that the CB servers are never authenticated.

In the interruption attack, the adversary may delete all messages on the link arriving to the CB Server giving an impression that these servers are not functional.

In the following we present several Level V attacks on the CB Service. In the interception attack, the adversary may steal personal information of subscribers stored in the client preferences data source in CB Server.

In the insertion or fabrication attack, subscribers not paying for the service may be inserted into the CB data source or fake charges may be inserted into a victims bill.

In the modification attack, the adversary can modify the forward numbers for subscribers, or corrupt logic in CBS timer agent and CBS bill calculator; this results in the victim receiving exorbitant bills. Other ways the price of the bills may be increased include corruption of the CBS client checker agent, or changing the bill amount or the time log of the clients consultation. Denial of service may be caused by simply removing the victims name from the client phone book.
In the interruption attack, the adversary may corrupt the phone book and delete victims from the client phone book data source, so that they may not receive CB service.

**Location Based Instant Message System (LB-IM):** Level IV attacks on the LB-IM Service may be enumerated as follows. In interception attacks, the adversary may view location track requests and the location in the location track responses violating privacy. The adversary may also view authentication messages and use them later for a replay attack.

In the insertion or fabrication attack, the adversary may bombard the LB-IM server with location track requests and authentication requests possibly causing a denial of service.

In the modification attack, the adversary may modify data parameters in signaling messages, such as: buddy names in the location track requests; buddy’s location in a location response; or the authentication responses in authentication challenge responses so that the LB-IM server is never authenticated.

In the interruption attack, the adversary may delete all messages on the link arriving to the LB-IM server giving the impression that this server is not functional.

In the following we present several Level V attacks on the LB-IM Service. In the interception attack the adversary may steal personal information of subscribers stored in the permissions and buddy list data store of the LB-IM server.

In the insertion or fabrication attack, subscribers not paying for the service may be inserted into the LB-IM permissions and buddy list data sources. At the LB-IM server, authentication information may be inserted to authenticate fake core network entities.

In the modification attack, the adversary may also have the capability to modify permissions and buddy list data stores. For example, the victim’s boss may be able to view the victims weekend locations. The service logic in the LB-IM request manager may be modified so that the requests are not properly checked. Changing the buddy list and permissions may cause legitimate members of the list to be unable to locate the subscriber thereby cause denial of service.
In the *interruption attack*, the adversary may delete certain target subscriber profiles in the data sources so that they may not receive LB-IM service. Service logic of certain agents may be completely deleted.

<table>
<thead>
<tr>
<th>Categories</th>
<th>Level IV</th>
<th>Level V</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interception</strong></td>
<td>Analyze traffic patterns, gather subscriber or company data. View details of messages between the cross network servers and the core service nodes.</td>
<td>Analyze traffic patterns, gather subscriber or company data. Steal personal information of registered subscribers.</td>
</tr>
<tr>
<td><strong>Fabrication or Insertion</strong></td>
<td>Send profile change messages to <em>subscriber profile manager agent</em> at cross network servers. Send a large number of <em>authentication requests</em> to cross network servers. Bombard the cross network servers with requests.</td>
<td>Insert unsubscribed subscribers for service into the Cross Network Servers, they receive service but are not charged for it. Insert fake data into the data stores of the cross network servers.</td>
</tr>
<tr>
<td><strong>Modification Of Resources</strong></td>
<td>Modify messages passing on the link. For example, change the challenge response to incorrect value thereby device is never authenticated; change query responses to incorrect values. Modify parameters in the signaling messages.</td>
<td>Modify service logic and data sources in the cross network servers.</td>
</tr>
<tr>
<td><strong>Denial Of Service</strong></td>
<td>Send the <em>session control agent</em> or the <em>subscribed services support agent</em> at MSC a large number of replies for a particular query or spoof it to be queries for different subscribers. Send cross network servers large number of <em>authentication requests</em> and slow them down.</td>
<td>May be caused by editing the data sources.</td>
</tr>
<tr>
<td><strong>Interruption</strong></td>
<td>Delete all messages arriving and leaving the cross network servers.</td>
<td>Delete data sources and service logic in the cross network servers.</td>
</tr>
</tbody>
</table>
Table 3.7. Attacks Classified by CASE II

<table>
<thead>
<tr>
<th>Categories</th>
<th>Data</th>
<th>Message</th>
<th>Service Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interception</strong></td>
<td>Gather customer information by reading data stored in database. At the cross network servers invoke the subscriber parameter manager agent.</td>
<td>Analyze traffic patterns, gather subscriber or company data.</td>
<td>Gather system information by observing or reading operations in the system. At the cross network servers read the service logic.</td>
</tr>
<tr>
<td>**Fabrication/</td>
<td>Send profile change messages to the subscriber profile manager agent at the cross network servers.</td>
<td>Send a large number of routing requests to the routing Agent at MSC so as to exhaust routing numbers and overload the MSC resulting in denial of service. Send registration or location update messages to the registration agent at the HLR causing incorrect call routing and shut down of the mobile devices. Send profile change messages to subscriber profile manager at HLR. Request AVs from authenticator agent at the HLR.</td>
<td>Insert unsubscribed subscribers for service into the cross network servers, they receive service but are uncharged. Insert fake data into the data stores of the cross network servers.</td>
</tr>
<tr>
<td><strong>Modification</strong></td>
<td>Modify subscriber profile Information so that the subscriber receives free services. For example, receive nation wide service when subscribed for local services. Remove subscribers name from database. For example, deny service for those already registered.</td>
<td>Modify messages passing on the link. Change the challenge response to incorrect value so that the device is never authenticated. Change query responses to incorrect values.</td>
<td>Modify service logic so that calls are forwarded to incorrect location. Change call forwarding logic. Show incorrect buddy subscriber location information. Modify service logic so as to change e-mail time stamps.</td>
</tr>
<tr>
<td><strong>Denial Of Service</strong></td>
<td>Deny service for those already registered by removing subscribers name from database. Invoke the registration agent.</td>
<td>Send multiple challenges to authenticator agent at HLR requesting authentication for multiple users. This could clog the HLR. Send the session control agent at MSC a large number of responses for a particular query or spoof it to be queries for different subscribers.</td>
<td>Modify service logic so that calls are forwarded to incorrect location. Change call forwarding logic.</td>
</tr>
<tr>
<td><strong>Interruption</strong></td>
<td>Delete subscriber data or subscriber preferences stored in the cross network serve such as mail data, phone book data, CF data, buddy list</td>
<td>Delete messages arriving and departing the cross network servers. Delete messages authentication challenge responses from the cross network servers to the HLR.</td>
<td>Delete CF rules set by the subscriber at the CFS subscriber parameter manager agent. Delete CBS service logic. Delete permissions and buddy list at the LB-IM subscriber parameter manager agent.</td>
</tr>
</tbody>
</table>
3.7 Discussion

In observing our tabulated attacks, we find a number of attack commonalities between in each dimension of attack taxonomy.

Consider Dimension I - the adversary’s physical access to the network. It is very simple to launch an attack with Level II access, due to the lack of core signaling message security, but the number of subscribers affected by a signaling message attack is very low (typically around 1). Gaining Level III access can be extremely challenging. As the adversary must have to be either an employee, or perform complex hacking to gain access into the service node. However, with Level III access, the adversary can cause many devastating attacks effecting the subscriber as well as the network. This may be the most difficult of all the physical access levels. The easiest access is Level IV, and V, however the damage is only limited to subscribers using the cross network service, and the cross network service itself.

With respect to Dimension II - attack categories, stealthy attacks can cause long term damage, and far-off disruption. Modification, and Interception attacks are the most stealthy in attack categorizes. Fabrication, Interruption, and Denial of Service attacks in the case of the telecommunication networks are easily detected due to the network structure.

In the case of Dimension III - vulnerabilities by exploiting data sources, and service logic it is possible to affect a group of subscribers, whereas signaling messages can effect only a single subscriber.

In summary, our taxonomy can not only classify currently known attacks, but can also be extended to to accommodate future technological advances. In developing this taxonomy, we have identified cascading attacks, cross infrastructure cyber attacks, key network vulnerabilities, and common attack characteristics.
Chapter 4

CAT

4.1 Introduction

In the previous chapter, we identified that regardless of how an attack is launched, if it causes a “system acceptable incorrect value corruption” then the corruption can cascade to other remote service nodes through “vehicles” such as signaling messages, cached data items, and shared databases. This propagation of corruption is called a cascading effect. We define attacks that exhibit the cascading effect as cascading attacks.

As mobile telecommunication networks are extremely complex systems comprising of thousands of service nodes of several different types, interfaced by diverse control protocols via hundreds of data elements, it is difficult to predict all the cascading effects of corruption. To be able to predict, not only all the diverse cascading effects of corruption but also detect the vulnerabilities that may be exploited to launch the attack i.e., attack origin, we developed a vulnerability assessment solution called Cellular Network Vulnerability Assessment Toolkit - CAT.

CAT is user friendly as it presents the vulnerabilities and cascading effects in the form of diagrammatic attack graphs. Attack graphs portray the vulnerabilities exploited, and remote cascading effects. Attack graphs also illustrate all the ways in which an attack may be launched exposing all vulnerabilities. This allows the network designers to determine vulnerable points in the network to protect.

It must be noted the design of CAT was challenging as it had to take into consideration the complex working of the mobile telecommunication network, as well as the diversity of the physical implementations of the network.

CAT incorporates the complex working of the network, and is robust to diversity of the physical implementations by incorporating the Third Generation Partnership Project (3GPP) defined telecommunication specification. These specifications detail the
complex working of the network using simple flow-like diagrams called the Specification and Description Language (SDL) [22], and hence we refer to them as SDL specifications. SDL specifications merely detail the functional details, and not the physical configuration aspects, and are available for no charge at [8]. These standards are used by equipment and service providers as the basis of their service implementations, systems development, engineering, and maintenance.

CAT works by taking user input of **seeds** (data parameters directly corrupt by the adversary and, whose cascading effect leads to a goal), and **goals** (data parameters that are derived incorrectly due to the direct corruption of seeds by the adversary) and uses the telecommunication specifications to identify cascading effects and construct attack graphs. The attack graphs generated by CAT show the global network view of corruption propagation and are loop free with low redundancy.

CAT is the first tool of its kind to detect cascading attacks with SDL specifications. It is practical, exhaustive, and efficient in generating output. CAT’s practicality is attributed to its ease of applicability, independence from physical deployments, and ready availability of SDL specifications [8]. It is exhaustive as it checks all relevant SDL specifications to generate an output, and is efficient in generating output as it produces attack graphs with low redundancy.

The contribution of this research include: (1) developing the first vulnerability assessment toolkit for cascading attacks; (2) usage of SDL specifications for vulnerability assessment; (3) definition of attack graphs for mobile telecommunication networks; and (4) categorization of attacks on mobile telecommunication networks.

This chapter is organized as follows. In Section 4.2, we present an overview of our toolkit. In Section 4.3 we detail the models used to develop CAT, and in 4.4 we enumerate the attack graph. In Section 4.5, we detail how the SDL specifications were used, and in Section 4.6 we present our algorithms and results. In Section 4.7, we present related work, and finally in Section 4.8, we conclude by discussing the future work.
4.2 Overview of CAT

In this section, we present an overview of CAT’s architecture, and its many features. CAT is implemented using the Java programming language. It is made up of a number of subsystems (as shown in Fig. 4.1(a)). The CAT-knowledge base contains the mobile telecommunication network knowledge required to identify where the corruption cascades. Network knowledge obtained from SDL specifications. Specifications contain simple flow chart like diagrams as shown in Fig. 4.4(b). These flow chart like diagrams are converted into data in our CAT-knowledge base (further details may be obtained in Section 4.5). The integrated data structure is similar to that of the knowledge base; it holds intermediate attack graph results.

Cascading effects typically result in what is called the chaining phenomena of cascading effects where chain is a sequence of $k$ corrupt data items (also called chain items). The first chain item is called the seed (data directly corrupt by the adversary), the second chain item is derived from the first chain item and so on. The final chain item is called the goal because it directly serves the adversary’s intent. Each chain item typically gives rise to a specific network mis-operation. To aid CAT in identifying the
cascading effects the user must provide the GUI subsystem with input in the form of seeds, and goals.

The analysis engine is incorporated with the algorithms (forward and mid-point), which explores the possibility of the seeds reaching the goal aided by the CAT-knowledge base.

Fig. 4.1(b) shows the functional architecture of the toolkit. The toolkit takes in user input (seeds and goals), and explores for the output. The initial output is a ‘maximum view’ of the attack graph which is pruned to remove redundancy to provide a clean ‘final view’. We use the term attack graph, as our graph is similar to the Internet based attack graphs [65, 9]. However, there are some major differences that were explained previously in the related work section.

Fig. 4.2. Work Flow of Vulnerability Assessment

Our attack graphs are user friendly, as they show the propagation of corruption through the network. The attack graph explains the network effects of the attack. Using these attack graphs, realistic attack scenarios may be derived. Attack scenarios explain the effect of the attack on the subscriber in a realistic setting. Each attack graph may have multiple interpretations, and give rise to multiple scenarios. Each scenario gives a different perspective on how the attack may effect the subscriber. This vulnerability assessment work flow is illustrated in Fig. 4.2. In the next section, we detail the models that were instrumental in the development of CAT.
4.3 Models

In this section, we present the models used as the basis of development of CAT.

4.3.1 Threat Model

Our work is focussed on the cascading effects that occur due to propagation of corruption to other service nodes or blocks. Hence our threat model includes any attack actions (such as various buffer overflows) that leads to cascading effects i.e., produces a seed. Readers interested in how these actions may be taken may refer to Chapter 3. Refer to Fig. 4.3, for the relationship between items that may be effected by attack.

![Fig. 4.3. Threat Model](image)

Based on their effects, we classify the relevant attack actions as those that corrupt (1) signaling messages; (2) caches; (3) database records; (4) local variables; and (5) service logic; distorted service logic will indirectly corrupt a message, a database record, or a local variable. In real life, these attack actions include (1) social engineering schemes that give adversary’s access to service nodes, and allow them to corrupt service logic; and (2) exploitation of software vulnerabilities that may overload the switch resulting in buffer overflow. Besides cascading attacks, there are a variety of other mobile telecommunication attacks, as summarized in Chapter 3; however, they are out of the scope of this work.
4.3.2 Network Model

Since CAT uses SDL specifications to derive attacks, we model mobile telecommunication networks using SDL notations.

**Block:** In our network model, we consider the network to be comprised of a set of concurrently running service nodes called *blocks* that communicate by exchanging signaling messages. Blocks represent a processing unit at a single location, and may be different types. Blocks may be of different types. There may be multiple instances of a single block type at any time. Examples of block types in the circuit switched domain of the mobile telecommunication network include the HLR, VLR, MSC, and the GMSC.

A block has two types of components: data items, and concurrently running processes. A block $B_i$ is associated with the following 4 types of data items: (1) data item contained in signaling messages to, and from the block; (2) updatable data items stored in an associated database (called a data source); (3) cached read-only data items; (4) other temporary local data items used in processing.

![Diagram](image-url)

Fig. 4.4. Example of CEFSM and SDL
**Process:** In a block, *processes* are the basic functional units. They use *service logic* to perform functions within a block. The service logic in every process includes (1) *process functionality* used to compute data items, and change the data associated with the block; (2) *database transactions* between processes and the associated database; and (3) *invocation* of other processes. Processes may be broadly classified as Mobility Management, Call Handling, Operations and Maintenance, Fault Recovery, Handover, and Subscriber Management.

A process may also be defined as a *communicating extended finite state machine* (CEFSM) which is a special case of the extended finite state machine (EFSM). The communicating extended finite state machine (CEFSM) may represented by a 6-tuple \((\Theta, \theta_0, \Xi, \delta, \Lambda, \Phi)\). The CEFSM state transition diagram of process call handling in the HLR is shown in Fig. 4.4(a). This process corresponds to the process in HLR responding to the SRI message in the call delivery service (illustrated in Fig. 2.3).

Typically a communicating extending finite state machine (CEFSM) is invoked when it receives signaling messages as input from other processes. On receiving the input, the process responds by performing the required actions, and generates an output signaling message.

In detail, the communicating extending finite state machine comprises of an initial state \((\theta_0)\) (*Idle*, such as shown in Fig. 4.4(a)). It receives an input event \((\Xi)\) (such as signaling message SRI as shown in Fig. 4.4(a)), performs certain transition actions \((\Lambda)\), generates an output event (such as signaling message PRN to VLR as shown in Fig. 4.4(a)) \((\Phi)\), and finally transitions \((\delta)\) into a final state (*Waiting for MSRN* as shown in Fig. 4.4(a)). \(\Theta\) represents the set of finite states in CEFSM. Transition actions comprise of the getting or setting certain conditions or variables or functions (such as *deriving GSM BC from ISDN compatibility*, or *setting alerting pattern* as shown in Fig. 4.4(a)).

**Signaling Message:** A signaling message \(m\) is a communication from a source block \(B_i\), to destination block \(B_j\) containing a set of \(n\) data items used to invoke processes in the destination block \(B_j\). The first \(k\) data items of the message provides origin and destination block information such as originating block address, originating process
name, destination block address, and destination process name. The remaining \( n - k \) parameters provide information regarding the processes to invoke in the receiving block.

### 4.3.3 SDL Model

SDL is a graphical object-oriented, formal language designed for the specification of event-driven, real-time, concurrent distributed systems interacting with discrete signals. In basic SDL, the system description is hierarchically structured to describe the local, and remote behavior of telecommunication systems, as Systems, Blocks, and Processes. The SDL System can be mapped to the mobile telecommunication network; the SDL Block, and Process may be mapped to the block, and process defined in the mobile telecommunication network model of the previous section, respectively.

Processes are the basic functional units of SDL systems. Specifications represent all network functions using the SDL process. Fig. 4.4(b) shows the graphical syntax of an SDL process. Fig. 4.4(c) shows the actual SDL fragment for the mobile telecommunication process state transition diagram of Fig. 4.4(a). On comparison of Fig. 4.4(a) and 4.4(c) it is obvious that the SDL process is similar in structure to the mobile telecommunication process CEFSM. Hence the mapping from the mobile telecommunication CEFSM process to SDL process is one-to-one. The SDL diagrams in Fig. 4.4(b) & 4.4(c) are representative of all the diagrams used in SDL specifications.

### 4.4 Attack and Attack Graph Overview

In this section, we detail a cascading attack detected by CAT, and also present the attack graph that is used to model these attacks.

#### 4.4.1 Sample Cascading Attack

In this section, we present a sample cascading attack called the **Speech Attack**. This attack was discovered by CAT using SDL specifications. This highlights the fact that an adversary with access to the SDL specifications could devise this attack.

The adversary targets the data item **ISDN Bearer Capability** in the IAM signaling message arriving at the GMSC. The GMSC is typically placed at the edge of the network
i.e., it is the first point of contact for a call arriving from an outside network. Hence any adversary with access to network infrastructure could easily corrupt data items in calls. The *ISDN Bearer Capability* is a data item which is used by the calling party to specify the type of ISDN channel required to carry voice communications with the receiver. The adversary may modify the *ISDN Bearer Capability* to a system acceptable value, so that it cannot be detected by error checking.

To aid in the illustration of the cascading effect of the attack, we present Fig. 4.4, and refer to Fig. 2.3. The corrupt *ISDN Bearer Capability* is passed on to the MSC in the signaling message IAM (number 6 in Fig. 2.3). The MSC uses the *ISDN Bearer Capability* in the computation of *Bearer Service* (it indicates the type of voice transmit signals between subscribers), and hence corrupts it. The corrupt *Bearer Service* is passed on to the VLR in the SIFIC message. The VLR sets up the call between the two subscribers using the corrupt *Bearer Service*. As the *Bearer Service* is incorrect, when the call is setup between subscribers the voice channel assigned is incorrect. As a result, the caller’s speech is garbled.

The attack shows that data corruption in a mobile telecommunication network has many *cascading effects* due to normal network operation. Cascading effects give rise to the *chaining* phenomena. A *chain* is a sequence of $k$ corrupt data items. The first item in the *chain* is the *seed* (data directly corrupt by the adversary), the second item is derived from the first item, and so on. The final corrupt item in the *chain* is called the
goal because it directly serves the adversary’s intent. In the case of the speech attack, the seed is the ISDN Bearer Capability, and the goal is the Bearer Service.

The goal item is usually defined by the effect of the attack on (a) the network operation; and (b) the target subscribers. Hence, a goal item has the following two properties: (1) it directly causes the mis-operation of the network; and (2) the caused mis-operation directly effects the target subscriber.

Formally, a cascading attack be defined as network state transitions caused by an adversary’s action, where the final transition results in the adversary achieving a goal. CAT deduces possible attacks on mobile telecommunication networks, and represents them in the attack graph format. In the following section, we present the CAT attack graph for mobile telecommunication networks.

4.4.2 The CAT Attack Graph

The mobile telecommunication network specific attack graph may be defined as a network state transition showing the paths through a system starting with the conditions of the attack, followed by attack action, and ending with its cascading effects. The mobile telecommunication network state may be defined as the collective state of all its components. Fig. 4.6 shows the attack graph output produced by CAT. This attack graph output is in telecommunication terminology, and corresponds to the previously described speech attack. CAT builds this graph bottom-up with the user input ISDN Bearer Capability as a seed, and Bearer Service as goal.

This attack graph has been divided into layers for description purposes where the conditions of the attack occurs at the lowest layer, and the final cascading effect at the highest layer. Each node has been given a node label, followed by an alphabet for description purposes.

Nodes represent states in the network with respect to the attack, and may be broadly classified as conditions, actions, and goal(s).

Condition Nodes: Nodes at the lowest layer typically correspond to the conditions that must exist for the attack to occur. These condition nodes directly follow from the
Fig. 4.6. Attack Graph of Speech Attack

taxonomy of the previous chapter. They are adversary’s physical access, target, and vulnerability exploit, and are detailed as follows.

Adversary’s Physical Access: The adversary may have any of the following levels of physical access to the network.

- Level I: Access to air interface.
- Level II: Access to links connecting core service nodes.
- Level III: Access to core service nodes.
- Level IV: Access to links connecting the Internet service nodes (cross network servers), and mobile telecommunication core service node.
- Level V: Access to cross network servers.
In the case of the speech attack, the adversary has access to links connecting to the GMSC service node i.e. Level II physical access which is represented by Node A in the attack graph.

Adversary’s Target: A service node is the high level description of the adversary’s target. In the case of the speech attack, the adversary corrupts data item ISDN Bearer Capability in the IAM message arriving at the GMSC. Hence the target of the attack is the GMSC represented by Node B.

Vulnerability Exploited: The adversary may take advantage of vulnerabilities such as detailed in the threat model, and explained briefly as follows.

- **Signaling Message**: Adversary exploits vulnerabilities in messages by modifying or corrupting data items in signaling messages.

- **Service Logic**: Adversary may exploit service logic vulnerabilities and replace the service logic in processes so that the process behaves abnormally.

- **Data Source**: Adversary’s may exploit vulnerabilities in data sources, and corrupt the data.

In the case of the speech attack, the adversary exploits vulnerabilities in a message (IAM) represented by Node D in the attack graph.

Our attack graphs show all the possible conditions for an attack to happen, i.e., we not only see the speech attack due to corruption of a message IAM arriving at the GMSC, but also other possibilities such as the corruption of Bearer Service in signaling message SIFIC represented by Node M.

**Action Nodes:** Nodes at higher layers are actions that typically correspond to effects of the attack propagating through the network. Effects typically include propagation of corruption between service nodes, such as from MSC to VLR (Node N), propagation of corruption within service nodes, such as ISDN Bearer Capability corrupting Bearer Service (Node L), and so on. Actions may further be classified as adversary actions, normal network operations or normal subscriber activities. Adversary actions include the insertion, corruption, or deletion of data, signaling messages, or service logic represented
by Node E. Normal network operations include sending (Node N), and receiving signaling messages (Node E). Subscriber activity may include updating personal data or initiating service.

**Goal Nodes:** Goal nodes typically occur at the highest layer of the attack graph. They indicate corruption of the goal items due to the direct corruption of seeds by the adversary (Node A).

**Edges:** In our graph, edges represent network transitions due to both normal network actions, and adversary actions. Edges help show the global network view of adversary action. This is the uniqueness of our attack graph. Transitions due to adversary action are indicated by an edge marked by the letter ‘A’ (edges connecting layer 0, and layer 1). *By inclusion of normal network transitions in addition to the transitions caused by the adversary, our attack graph not only shows the adversary’s activity but also the global network view of adversary’s action. This is a unique feature of the attack graph.*

**Trees:** In the graph, trees are distinguished by the tree numbers assigned to its nodes. For example, all the nodes marked with number 2 belong to Tree 2 of the graph. Some nodes in the graph belong to multiple trees. Tree numbers are used to distinguish between AND, and OR nodes in the graph. Nodes at a particular layer with the same tree number(s) are AND nodes. For example, at Layer 4, Nodes H, I, J, and K are AND nodes; they all must occur for Node M at Layer 5 to occur. A node with multiple tree numbers are called OR nodes. The OR node may be arrived at using alternate ways. For example, Node O at Layer 6 is an OR node, the network state indicated by Node O may be arrived at from Node M or Node N.

Each attack tree shows the attack effects due to corruption of a seed at a specific network location (such as signaling message or process in a block). For example, Tree 1 shows the attack due to the corruption of the seed Bearer Service at the VLR. Tree 2 shows the propagation of the seed ISDN Bearer Capability in the signaling message IAM. These trees show that the vulnerability of the mobile telecommunication network is not
limited to one place, but can be realized due to the corruption of data in many network locations.

In constructing the attack graph, CAT assumes that an adversary has all the necessary conditions for launching the attack. Our attack graph format is well-suited for telecommunication networks because data propagates through the network in various forms during the normal operation of a network; thus, an attack that corrupts a data item manifests itself as the corruption of a different data item in a different part of the network after some network operations take place.

### 4.4.3 Attack Scenario Derivation

This section explains the principles involved in the derivation of realistic attack scenarios from attack graphs in telecommunication semantics.

**Step 1: End User Effect:** Goal node(s) are used to infer the end effect of the attack on the subscriber. From the attack graph illustrated in Fig. 4.6 i.e., the speech attack, the goal nodes are Node O at Layer 6. According to the goal node, the SIFIC message to the VLR has incorrect goal item *Bearer Service*. The SIFIC message is used to inform the VLR, the calling party’s preferences such as voice channel requirements, and request the VLR to setup the call based on the calling party, and receiving party preferences. If the calling party’s preferences (such as *Bearer Service*) are incorrect, then the call setup by the VLR is incompatible with the calling party, and the communication is ineffective (garbled speech). From the goal node, it may be inferred that Alice, the receiver of the call is unable to communicate effectively with Bob, the caller, as Alice can only hear garbled speech from Bob’s side.

**Step 2: Origin of Attack:** Nodes at layer 0 indicate the origin of the attack, and hence the location of the attack may be inferred. The speech attack may originate at the following locations: signaling messages IAM, and the service nodes VLR.

**Step 3: Attack Propagation and Side effects:** Nodes at all other layers show the propagation of corruption across the various service nodes in the network. In the speech attack, from the other layers it may be inferred that the seed is the ISDN Bearer
Capability, and the attack spreads from the MSC to the VLR. We found that some non-goal nodes (that indicate the propagation of corruption) also indicate side effects on the user in addition to the goal node that shows the final effect on the user.

**Attack Scenario:** Using the above guidelines the following attack scenario may be derived as follows. Trudy, the adversary corrupts the ISDN Bearer Capability of Bob, the victim, at the IAM message arriving at the GMSC. The GMSC propagates this corruption to the MSC, which computes, and hence corrupts the Bearer Service. The corrupt Bearer Service is passed on to the VLR, which sets up the call between Bob, the caller, and Alice, the receiver. Bob and Alice cannot communicate effectively as Alice is unable to understand Bob.

### 4.5 Usage of SDL Specifications

In this section, we detail the usage of the SDL flow like diagrams from the specifications in our knowledge base. We also present some issues that must be addressed for effective usage of these specifications.

#### 4.5.1 CAT Knowledge Base

The CAT knowledge base contains the network knowledge of the mobile telecommunication service whose vulnerability is subject to analysis. We populated the knowledge base using the SDL specification for call delivery [3] as we are interested in analyzing the vulnerability of the call delivery service.

<table>
<thead>
<tr>
<th>Message Name</th>
<th>Data Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRI</td>
<td>Mobile Station International ISDN Number (MSISDN), Alerting Pattern, CUG interlock, CUG outgoing message, ISDN BC, ISDN LLC, ISDN HLC, GMSC Pre-paging support.</td>
</tr>
</tbody>
</table>
CAT knowledge base stores the network information in two database tables. Table 4.1 shows the structure of Table 1 in the CAT knowledge base, and some sample data stored in the table. This table stores the signaling message information information, and data parameters contained by these signaling messages. It presents a sample of signaling message data by showing the data contained in the SRI signaling message. This table aims to captures all the data items that may be used to propagate corruption between blocks.

Table 4.2 shows the structure of Table 2 in the CAT knowledge base, and some sample data stored in the table. This table stores the information of the basic functional unit of SDL systems, i.e a process. It captures all the aspects of a SDL Process, i.e. the initial, and final states, input, and output messages, and actions. On close observation, it can be seen that this table is a direct translation of the SDL fragment shown in Fig. 4.4(c). This table aims to captures all the data items that may be used to propagate corruption within a block, i.e. one data item corrupting another.

### Table 4.2. Table 2-CAT Knowledge Base: Processes

<table>
<thead>
<tr>
<th>Process Name</th>
<th>Block Name</th>
<th>Initial State</th>
<th>Input Message</th>
<th>From Block</th>
<th>Action</th>
<th>Output Message</th>
<th>To Block</th>
<th>Final State</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRI_HLR</td>
<td>HLR</td>
<td>Idle</td>
<td>SRI</td>
<td>GMSC</td>
<td>Derive GSM BC from ISDN compatibility; Derive basic service from GSM BC; Set Pre-paging support;</td>
<td>PRN</td>
<td>VLR</td>
<td>Wait for MSRN</td>
</tr>
</tbody>
</table>

Unlike other methods, CAT does not require as input network topology, network configuration, adversary profile information or attacks in atomic form. CAT works for any implementation which follows telecommunication standards. CAT assumes that the adversary has the necessary tools to penetrate the network at the five different levels of physical access, and three different vulnerabilities as described in Section 4.4.
4.5.2 Issues with SDL

CAT traces the flow of data through the network. Corruption of a data item may lead to the corruption of other data items which may or may not lead to the goal. When converting SDL specification to data in the CAT knowledge base there are a number of issues that must be taken into consideration for effective usage of the specification.

Message and Message-ACK Pairs: Issues arise with message, and the corresponding message-ack pairs, such as the message SRI and SRI-ACK, PRN and PRN-ACK. The message typically comprises of a subscriber key which the receiving block uses to assign a value. The message-ack contains this value but not the subscriber key. The absence of the subscriber key in the message-ack (PRN-ACK) results in CAT missing some attacks, because of loss of dependencies. The loss of the subscriber key must be compensated during the design of the knowledge base, and algorithms.

Data Dependencies in Action Attributes: The Action column in Table 2 of the CAT knowledge base contains procedures such as ‘Derive GSM BC from ISDN compatibility’. SDL specifications may not specify details of the subroutine such as: (1) input and output data items in the subroutine; and (2) relationship between these input and output data items. Without these details CAT may miss some cascading effects due to loss of data relationships.

In many cases SDL specifications does not provide this information. Uncovering multiple seed attacks requires knowledge of the input-output data items, and their relationships. If this information is present in the SDL specifications (Action attribute of Table 2) then the attacks are discovered. If they are not present in the the SDL specifications, these attacks may not be discovered.

Thus CAT is limited by SDL’s inadequacies. In the case of the message and message-ack pairs, we can compensate the inadequacy during the creation of the knowledge base. However, in the case of data dependencies where SDL is unhelpful, CAT discovers an incomplete set of attacks.
4.6 Algorithms

In this section, we present our algorithms - *Forward Exploration Algorithm, Heuristic Based Mid-Point Algorithm*. We detail rules to identify corruption propagation, attack types detected, and analysis of performance of these algorithms.

4.6.1 Introduction

CAT automatically extracts attack graphs out of SDL specifications stored in our CAT knowledge base.

CAT takes seeds and goal as input, and generates an attack graph that traces the progression of the seeds to a goal. With more seeds, the algorithm will produce a more detailed attack graph showing the many ways to reach a goal. Note that by exhaustively trying every possible goal, and every possible set of seeds, CAT can be easily extended to automatically identify all the possible attacks associated with a certain service such as the call delivery service.

Cascading Effect Detection Rules: We have developed rules to aid in the detection of cascading effects. We have incorporated these rules into our algorithms.

CE1: Corruption propagates from a message to its destination block if the seed is contained in the incoming message, and the same seed occurs in the action taken by a process in the block.

CE2: Corruption propagates from seed to seed (or goal) in a block if the seed occurs in the action column of a process in the block, and another seed or the goal itself occurs in the same action column.

CE3: Corruption propagates from a block to an outgoing message in a block if seed or goal occurs in the action column of a process in the block, and the same seed or goal occurs in the outgoing message from the block.

Using these detection rules, CAT can detect the attack types detailed in the next section.
4.6.2 Attack types

The following are the attack types that may be detected by CAT.

1. **1-Indirect attacks:** In this attack, corruption of Seed₁ leads to corruption of the goal. Seed₁ → Goal.

2. **N-Indirect attacks:** In this attack, given any 'k' seeds ∈ {S}, Seed₁, Seed₂, ..., Seedₖ, and Goal, corruption of Seed₁ leads to corruption of some seed Seedᵢ (where Seedᵢ ∈ {S}), which in-turn leads to corruption of some seed Seedₖ (where Seedₖ ∈ {S}), and so on until the Goal is corrupt. Seed₁ → ... Seedᵢ → Seedₖ → ... → Seedₙ → Goal.

3. **Collaborative attack:** In this attack, a single seed cannot reach the Goal but the corruption of multiple seeds allows for reaching of the Goal i.e. Seed₁ & Seed₂ → Goal.

4.6.3 Exhaustive Forward Exploration Algorithm

The Forward Exploration algorithm starts ‘bottom-up’ from the seed, and works towards finding the goal. It is an exhaustive method, and checks every row in the CAT Knowledge database. The Forward exploration algorithm is detailed in Algorithm 1, and

2. The algorithm works in two phases.

| Input: | \{Seed\} : Set of Input Seeds; \{Goal\} : Goal |
| Procedure Build-Tree(sᵢ) |
| 1: for ∀ sᵢ ∈ \{Seed\} do |
| 2: for ∀ row ∈ (Table-1) .or. ∀ row ∈ (Table-2) do |
| 3: if sᵢ in Data Item of Signaling Message(Table-1) .or. sᵢ in Action of Process(Table-2) then |
| 4: Create-Tree |
| 5: Add-Node(PA,Tgt,Vul) to Tree ... \{Initialize the Tree\} |
| 6: Detect-Cascading-Effect(sᵢ) |
| 7: Add-Tree to Graph |
| 8: end if |
| 9: end for |
| 10: end for |
| 11: Prune Graph |

End Procedure

**Algorithm 1:** Forward Exploration: Bottom Up Approach - I
Phase 1: Building Trees: In this phase attacks are detected, and trees are built based on seed values at network location, and cascading effect detection rules described previously. The first part of phase 1, is shown in Algorithm 1. Algorithm 1 shows the build tree procedure, which creates a tree, when a seed occurs in a network location (signaling message or process in a block). The tree is initialized by creating condition nodes. The rest of the tree is constructed in the Detect – Cascading – Effect procedure shown in Algorithm 2.

Cascading effects are detected in Detect – Cascading – Effect procedure shown in Algorithm 2 aided by the cascading effect detection rules. The procedure attempts to connect the seed to the goal. Our attack graph is succinct because when the seed does not lead to the goal, the tree is rolled back (Line 23, 36), and not added to the graph.

As the tree is built, garbage is collected to remove extraneous nodes, and assign layers to aid in pruning the graph. Garbage collection is performed by rolling back graph nodes when it is found that the goal is unreachable. Rollback is performed by resetting the current graph node pointer to point to the initial graph node.
Procedure Detect-Cascading-Effect($s_i$)
1: while true do
2:   if $s_i$ in Action of Process (Table-2) then
3:     Add-Node(Corrupt $s_i$ in Action of Process in Block) to Tree
4:     if goal in Action of Process (Table-2) then
5:       Add-Node(Corrupt $s_i$ corrupts goal in Action) to Tree {CE2}
6:       Break {Seed Corrupts Goal ... Goal Reached}
7:   else
8:     if $s_k \in \{Seed\}$ in Action of Process (Table-2) then
9:       if Collaborative then
10:      Add-Node(Corrupt $s_i$ & $s_k$ in Action) to Tree
11:    else
12:      Add-Node(Corrupt $s_i$ corrupts $s_k$ in Action) to Tree {Indirect Attacks}
13:    end if
14:  else
15:  {Check if $s_i$ corrupts other Seeds}
16:  if $s_k \in \{Seed\}$ in Action of Process (Table-2) then
17:  {CE2}
18:    if Collaborative then
19:      Add-Node(Corrupt $s_i$ & $s_k$ in Action) to Tree
20:    else
21:      Add-Node(Corrupt $s_i$ corrupts $s_k$ in Action) to Tree {Indirect Attacks}
22:    end if
23:  else
24:    {Check if $s_i$ corrupts other Seeds}
25:  {Check if $s_i$ corrupts other Seeds or the goal}
26:  if $s_i$ in Output Message of Process (Table-2) then
27:    {Corrupt seed $s_i$ does not corrupt other seeds or propagate to other blocks}
28:      Add-Node(Corrupt $s_i$ in Output Message) to Tree {CE3}
29:    else
30:      Rollback Tree
31:      Break
32:    end if
33:  end if
34:  else
35:    {Corrupt seed $s_i$ does not corrupt other seeds or propagate to other blocks}
36:      Add-Node($s_i$ in Output Message) to Tree {CE1}
37:    else
38:      Rollback Tree
39:      Break
40:    end if
41:  end if
42: end while
End Procedure

Algorithm 2: Forward Exploration: Bottom Up Approach - II
This allows for reuse of graph nodes. Layers are assigned to the nodes in the graph, starting with the condition nodes at layer 0, and nodes ‘x’ edges away from the goal at layer ‘x’. Fig. 4.6 clearly illustrates the concept of layers. These layers are shown in the attack graph output Fig. 4.6, but are useful while pruning the graph. Once all the trees are built, they are merged into a graph. The graph is pruned to remove extraneous nodes as shown in Line 11 of Algorithm 1.

**Termination Conditions:** This phase is exhaustive and stops when (i) the seed reaches the goal, or (ii) the seed has not reached the goal, but stops propagating, or (iii) all messages up to the terminating message are explored.

**Phase 2: Integrating and Pruning trees:** This phase integrates and prunes all the trees built in the previous section. Low redundancy attack graphs are constructed by collapsing similar nodes and paths at the same layer into a single node and path. Fig. 4.6 shows the attack graphs with low redundancy.

The Forward exploration algorithm checks row in the CAT knowledge base, while exploring to find the goal (exhibited by Line 2 of Algorithm 1). Hence this method is exhaustive, it produces a graph that addresses all possible attacks for a given seed, and goal. All common paths are eliminated and redundancy is reduced by pruning the graph hence this method is also succinct (exhibited by Line 11 of Algorithm 1).

The Forward exploration is a good approach if the seed actually reaches the goal. It detects the four different types of attacks detailed previously, and produces attack graphs with low redundancy, and low execution time. Issues arise when the seeds fail to reach the goal.

### 4.6.4 Heuristics Based Mid-Point Algorithm

This algorithm works ‘bottom-up’ from the seed, and ‘top-down’ from the goal, and terminates when the seed nodes from the bottom meet the goal nodes from the top. When seeds do not meet the goal the given heuristics is used as a terminating condition (Line 3-5) to make the algorithm efficient. Algorithm 3, and 4 shows the Heuristics Based Mid-Point method.
*Heuristics for Algorithm Termination:* As the network model, and the CAT-knowledge base is already known, it is possible to assign heuristics as a terminating condition. The following are the heuristics that may be assigned.

- **Limit Layers Heuristics:** This heuristic states that if the number of layers in the graph exceeds a set limit number \( \text{layer} - \text{limit} \) (e.g., 10 Layers) the algorithm can terminate. This approach is not practical because it takes execution time, and creates extraneous \( \text{layer} - \text{limit} \) layers of nodes. Also, this approach may not detect attacks that take up more than \( \text{layer} - \text{limit} \) layers in the attack graph.

- **Limit Node Heuristics:** This heuristic states that if the number of nodes in a graph exceeds a set number of nodes \( \text{node} - \text{limit} \) (e.g., 20 Nodes) then the algorithm can terminate. This approach is also not practical because it takes execution time, and creates extraneous \( \text{node} - \text{limit} \) number of nodes. Also, this approach may not detect attacks that take up more than \( \text{node} - \text{limit} \) layers in the attack graph.

- **1-Indirect Attack Checking Heuristic:** This heuristic checks for 1-Indirect Attacks. If the goal and the seed appear together in the action column of the process, and the output signaling message has the goal data parameter, it is clear that the seed causes 1-Indirect Attack. This heuristic is efficient in eliminating seeds that do not cause 1-Indirect attack using a single database query for a single seed. \( N \) seeds would require \( N \) database queries. However, this heuristic is not sufficient to eliminate seeds that do not cause \( N \)-Indirect Attacks.

- **\( N \)-Indirect Attack Checking Heuristic:** This heuristic checks for \( N \)-Indirect Attacks by considering \( N^2 \) 1-Indirect Attacks for \( N \) seeds. This heuristic checks each seed with every other seed, and the goal for 1-Indirect Attacks. A seed may be eliminated from \( N \)-Indirect attacks if any of the other seeds it affects, fails to reach the goal. This approach is exhaustive, and requires \( N^2 \) database queries for \( N \) seeds, and hence completely defeats the purpose of the heuristic method.

- **Parameter Based Heuristic:** In this approach, lists containing sets of related, and unrelated data parameters are created. The list of related parameters contains
pairs of associated data parameters. For example, if data X is used to derive data Z, and data Y is not involved in computing the Z, the pair ‘X-Z’ belongs in the related parameters list, and the pair ‘Y-Z’ belongs in the unrelated parameters list. Creation of such lists is practical and feasible due to the knowledge of the network model, and the CAT knowledge base. If the seed-goal pair occurs in a ‘related parameters’ list, the algorithm is executed, else the algorithm is not executed. The advantage of this approach is that it does not require any computation, and reduces execution time. This approach can, on average, produce good results, but may not satisfy all possible cases, and in some cases produce incorrect results.

**Issues with Parameter Based Heuristics:** It is not possible to have the perfect heuristic. There is always the problem of mis-detection resulting in Missed seeds, and Over-Seeking seeds. Due to inadequate heuristics, seeds that actually reach the goal are assumed to fail to reach the goal resulting in Missed seeds. This may result in attacks not being detected, and happens because it is not possible to have a complete and exhaustive list of ‘related’ parameters, and ‘un-related’ parameters. This may be minimized by picking a heuristic that only checks for ‘un-related’ Parameters. Over-Seeking seeds are those that fail to reach the goal but are assumed to be able to reach the goal carry the algorithm works towards matching the seed nodes, and the goal nodes. This may result in unnecessary seeking of the goal with a large number of extraneous nodes, and increase in execution time.

In the next section, the results generated by the two algorithms are discussed.
**Input:** \{Seed\} : Set of Input Seeds; \{G\} : Goal

**Output:** Graph

**Procedure Mid-Point**

1. for every \(s_i \in \{\text{Seed}\}\) do
2. \(\ldots\) \{Check if the tree may be built\}
3. if Check-Heuristic\((s_i) = \text{FALSE}\) then
4. \(\text{continue}\)
5. \(\text{else}\)
6. \(\ldots\) \{Initializing the Tree\}
7. if \((s_i \in \text{Message}) \text{ or } (s_i \in \text{Action})\) then
8. \(\quad\text{Add-NodePA,Tgt,Vul)} \text{ to Tree}\)
9. \(\text{else}\)
10. \(\quad\text{Add-Node(Goal)} \text{ to Tree}\)
11. \(\text{end if}\)
12. seed-element \(\leftarrow s_i\)
13. goal-element \(\leftarrow \text{Goal}\)
14. goal-reached \(\leftarrow \text{compare(seed-element, goal-element)}\)
15. \(\ldots\) \{Building the Tree\}
16. while goal-reached=FALSE do
17. \(\quad\text{seed-element} \leftarrow \text{find-next-seeds(seed-element)}\)
18. \(\quad\text{build-tree-upwards(seed-element)}\)
19. \(\quad\text{goal-reached} \leftarrow \text{compare(seed-element, goal-element)}\)
20. \(\ldots\) \{Tracks the seed upward\}
21. if goal-reached=FALSE then
22. \(\quad\text{goal-element} \leftarrow \text{find-prev-seed(goal-element)}\)
23. \(\quad\text{build-tree-downwards(goal-element)}\)
24. \(\quad\text{goal-reached} \leftarrow \text{compare(seed-element, goal-element)}\)
25. \(\text{end if}\)
26. end while
27. end if
28. end for
29. \text{End Procedure}

**Algorithm 3:** Heuristic Based Mid Point Method - I

**Procedure:** Compare\((s_e, g_e)\)

1. if \(s_e = g_e\) then
2. \(\quad\text{Add (Corrupt } s_e \text{ corrupts } g_e) \text{ to Tree}\)
3. \(\quad\text{return true}\)
4. else
5. \(\quad\text{return false}\)
6. end if

**End Procedure**

**Algorithm 4:** Heuristic Based Mid Point Method - II
4.6.5 Results

The two algorithms studied above have the ability to detect 1-Indirect, N-Indirect, and Collaborative attacks, examples of some representative results produced by CAT are explained below.

1-Indirect attacks: Seed$_1$ → Goal. Seed$_1$: ISDN BC → Goal:Bearer Capability. Corrupting the ISDN BC in the IAM message leads to incorrect calculation of the Bearer Capability.

N-Indirect attacks: Seed$_1$ → Seed$_2$ . . . Seed$_i$ → Seed$_j$ → . . . → Seed$_n$ → Goal. Seed$_1$: MSISDN → Seed$_2$: Pre-paging support → Goal:IMSI. Corrupting the MSIDN in the IAM signaling message leads to the retrieval of incorrect IMSI at the HLR, which leads to incorrect retrieval of TMSI and hence call is sent incorrectly. Incorrect subscriber receives calls.

4.6.6 Performance Analysis

We evaluated the performance of our algorithms based on execution time, by varying the Seed Failure Ratio. Seed Failure Ratio may be defined as the ratio of number of seeds failing to reach the goal to the total number of seeds, input to the algorithm (shown in Formula 4.1).

\[
\text{Seed Failure Ratio} = \frac{\text{Number of Seeds failing to reach the goal}}{\text{Total number of seeds input}} \quad (4.1)
\]

The results of the experiments are shown in Fig. 4.7. We conducted the experiment with a goal, 8 seeds and heuristics. Seeds and heuristics are varied to match the Seed Failure Ratio. When the Seed Failure Ratio is 0 (i.e., all the seeds reach the goal) we find that the performance of both the algorithms is similar. The difference in performance is evident when the Seed Failure Ratio increases i.e, increasing the number of seeds failing to reach the goal.
With 0% mis-detection (exact heuristic matching), the mid-point algorithm performs best and can cause up to 68% decrease in time, in comparison to the forward exploration algorithm. With the 50% mis-detection rate, the mid-point algorithm causes up to 11% decrease in time, in comparison with the forward exploration algorithm.

4.7 Related Work

Two sets of work are relevant to CAT, one on vulnerability assessment in telecommunication networks, and the other on attack graphs.

Mobile telecommunication network vulnerability assessment: The vulnerabilities of telecommunication networks are well addressed in the literature. Lorenz et al. [44], Moore et al. [47], Kluepfel et al. [36] identify threats, vulnerabilities, and attack
scenarios on the SS7 domain. Lorenz et al. [44] defines an attack taxonomy for SS7 network entities. Their work classifies attacks based on the SS7 entity (Service Switching Points, Signal Transfer Point, and Service Control Point) under attack.

Telecommunication specifications [2], [6], [1] specify mobile telecommunication security, and identify certain security threats. They are eavesdropping, masquerading, traffic analysis, browsing, compromising authentication vectors, manipulation of messages, disturbing or misusing network services, denial of service, resource exhaustion, misuse of privileges, and abuse of services. Howard et al. [27], El-Fishway et al. [21], Lo et al. [43], Welch et al. [76], Clissmann et al. [16] have identified single infrastructure threats or attack scenarios on the mobile telecommunication network while trying to prove the inadequacy of current security schemes, or present new architectures or guidelines for mobile telecommunication network security. The attack scenarios identified in the papers mentioned include masquerade, unauthorized notification of resources, line-tap attacks, eavesdropping, man in the middle, guessing attack, replay attacks, and interleaving attacks.

Brookson et al. [14] motivates the need for security. Mitchell et al. [45], Boman et al. [13], and Bharghavan et al. [12] discuss the security features available in current mobile telecommunication networks. Kotapati et al. [41] presents a taxonomy of cyber attacks in mobile telecommunication networks.

However, they neither use specifications to devise cascading attacks nor perform attack-graph based vulnerability analysis.

CAT is the first attempt to detect cascading attacks using the available SDL specifications. CAT captures the unique semantics of telecommunication infrastructures by constructing attack graphs. Mobile telecommunication network administrators typically use conventional tools to assess the physical configuration vulnerabilities introduced due to the implementation of the network. In the literature there are no known techniques for vulnerability assessment in mobile telecommunication networks.

**Attack-graph technologies:** Schneier [64] was one of the first to suggest the usage of a tree structure called the attack graph, to represent attacks. Since then, attack
graph generation techniques have been extensively studied by [70, 58, 69, 62, 65, 9, 55, 34, 35, 66, 73, 17, 77, 18, 60, 49, 50, 51]. However these technologies are not designed to handle telecommunication semantics i.e. dependencies that cause corruption to propagate.

The earlier work in this area includes Netkuang [77], a network configuration vulnerability checker that performs a goal based breath first search. Swiler and Philips [70, 58, 69] generate their attack graphs by backward exploration from the goal given atomic attacks as input. In contrast, our exhaustive forward exploration algorithm generates attack graphs by forward exploration from seeds to goals.

Model checking is a major technique proposed for automatic attack graph generation, and has been used by: Ritchey and Amman [62] for vulnerability analysis of a network; Sheyner et al. [65] for automatic generation of attack graphs; and Ramakrishnan et. al. [60] for identifying configuration vulnerabilities. However, all the above techniques have the disadvantage of not being scalable. Ammann et al. proposed a fix to this problem by exploiting a monotonic assumption to achieve scalability [9]. Ou et al. [55] proposed a logic-programming method for scalable network vulnerability analysis. Jha et al. [34, 35, 66] have analyzed attack graphs in terms of properties such as survivability, reliability, etc. They also present a minimization technique that allows analysts to decide on the minimal set of security measures to guarantee the safety of the system.

All the aforementioned attack graph research focuses on Internet vulnerability assessment. The related work surveyed reveals that CAT is the first to use SDL specifications to detect possible mobile telecommunication network attacks. The semantics of attack graphs generated by CAT are fairly different from Internet attack graphs. For example, in Internet attack graphs only exploits can cause state transitions, while in attack graphs generated by CAT, many state transitions are caused by legitimate actions. Nevertheless, by viewing the corruption propagation actions of CAT as accidental “unintentional exploits”, the attack graphs generated by CAT could be conceptually deduced to an Internet attack graph [65, 9]. Accordingly, existing attack graph analysis techniques (e.g., [34, 35, 66]) could be applied to analyze the graphs generated by CAT.
In this sense, existing attack graph analysis techniques are complementary to CAT. Nevertheless, it should be noticed that the focus of our work is to generate, and interpret mobile telecommunication specific attack graphs, although analyzing them is part of our future work.

From the viewpoint of attack graph generation, please note that besides the new cascading attacks identified, the main contribution of our work is identification of cascading effect detection rules, which are necessary to identify cascading attacks. We incorporate the cascading effect detection rules into our attack graph generation algorithms that are light-weight, and ad-hoc attack. We developed our own graph generation algorithms, instead of adopting an existing methods such as model checking, and [9] because it is difficult to pre-determine completely or precisely the attacker’s goals and both model checking, and [9] can analyze vulnerabilities only when the goals are provided as input.

In the sense of exploiting certain network dependencies to “chain” attack actions together, our work is relevant to alert correlation research (i.e., [73, 18, 17, 49, 50, 51]). Nevertheless, alert correlation is an intrusion detection activity that needs alerts to be raised in prior to correlating them while CAT does not need any alerts.

4.8 Discussion

In conclusion, CAT has been developed to detect cascading attacks. While CAT can detect a complete set of 1-Indirect and N-Indirect attacks, it is limited by SDL’s inadequacies, and hence cannot completely detect collaborative attacks. To alleviate limitations in SDL and ensure that CAT can detect a complete set of attacks for a given user input and SDL specifications, we plan to extend CAT. We call this extension of CAT as advanced CAT (aCAT). aCAT will support a number of features which will allow for detection of a complete set of attacks detailed in the next chapter.
Chapter 5

aCAT

5.1 Introduction

In this chapter, we present Advanced CAT - (aCAT) an extension of CAT. CAT is limited in its capabilities due to the limitations in SDL. aCAT is enhanced with features to compensate CAT’s limitations by incorporating: (1) minor amount of expert knowledge to the SDL specifications so as to fill the lacking in SDL; (2) unique network dependency model and infection propagation rules for accurate chain detection; and (3) forward, reverse, and combinatory algorithms to the analysis engine.

aCAT is also improved in terms of its user input requirements. It requires as input either seeds (data items that are directly corrupted by the adversary) or goals (the ultimate data item(s) corrupted due to cascading effect in a remote location), whereas CAT requires both seeds, and goals.

We used aCAT to identify cascading attacks in several of the key services offered by the mobile telecommunication network, and found that aCAT can identify a better set of cascading effects in comparison to CAT. aCAT has also detected several interesting, and unforeseen cascading attacks that are subtle, and difficult to identify by other means. These newly identified cascading attacks include the Alerting Attack, Power-off Power-on Attack, Mixed Identity Attack, Call Redirection Attack, and Missed Calls Attack.

This chapter is organized as follows. In Section 5.2, we detail the improved architecture of aCAT. In Section 5.3, we detail the network dependency model, and the infection propagation rules used to develop aCAT. In Section 5.4, we present the aCAT knowledge base, and in Section 5.5, we present our algorithms. In Section 5.6, we present some interesting attacks detected by our algorithms, and in Section 5.7, we present the impact of expert knowledge. Finally, in Section 5.8, we conclude by discussing the future work.
5.2 Architecture

Fig. 5.1. Architecture of aCAT

Although the overall architecture of aCAT (illustrated in Fig. 5.1) is similar to the architecture of CAT there are many significant differences in the internal aCAT architecture detailed as follows:

- aCAT knowledge base contains expert knowledge in addition to SDL specifications.
- Data in aCAT knowledge base is formatted using the network dependency knowledge.
- The analysis engine is incorporated with the forward, reverse, and combinatorial algorithms that require either seeds or goals as input.
- The aCAT algorithms are incorporated with infection propagation rules.

5.3 The aCAT Model

In this section, we present the network dependency model, and the infection propagation rules incorporated into aCAT.
5.3.1 Network Dependency Model

In principle, cascading attacks are the result of propagation of corruption between network components, (such as, signaling messages, caches, local variables, and service logic) due to relationships (also called dependencies) that exist between these network components. Hence to uncover these attacks, we define a network dependency model, and a set of infection propagation (IP) rules (Table 5.1) that captures the propagation of corruption.

In our first IP rule we define corruption in the three basic mobile telecommunication components, i.e. data items, signaling messages, and service logic. Subsequently we categorize dependencies as within a block (intra-block) or as between blocks (inter-block). In particular, our second IP rule defines the corruption due to inter-block dependencies, and it is exhibited by signaling messages $M_1$, $M_2$, and $M_3$ in Fig. 5.2. Message $M_1$
Table 5.1. Infection Propagation Rules

<table>
<thead>
<tr>
<th>No.</th>
<th>Corruption</th>
<th>Infection Propagation Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Network Components</td>
<td>A data item $d_a$ of correct value $u$ at any instant of time $t$ is said to be corrupt iff there exists a value $v$, of $d_a$ at the same instant of time $t$ where $u \neq v$.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A signaling message is said to be corrupt iff there exists a data item contained in the message where the data item has been corrupted directly by an adversary or indirectly due to cascading effects.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The service logic of a process is corrupt iff there exists a set of input data items with correct original value, and the output item computed with the service logic is corrupt.</td>
</tr>
<tr>
<td>2.</td>
<td>Inter-Block Dependency</td>
<td>Correlation spreads between blocks iff there exists a corrupt signaling message propagating between the respective blocks.</td>
</tr>
<tr>
<td>3.</td>
<td>Derivative</td>
<td>The output data item in an AND derivative dependency is corrupt iff all ‘n’ input data items are corrupt.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The output data item in a OR derivative dependency is corrupt iff any one of the ‘n’ input data items are corrupt.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The output data item in a KEY derivative dependency is incorrect iff the key (input item) used to retrieve the output item is corrupt.</td>
</tr>
<tr>
<td>4.</td>
<td>Messages</td>
<td>The output message generated by the process is corrupt iff either one of the following is true:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. The service logic of the process is corrupt.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. The service logic of the process is not corrupt but either one of the following is true:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A. Input signaling message is corrupt.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B. Data source owned by the block is corrupt.</td>
</tr>
<tr>
<td>5.</td>
<td>Chaining</td>
<td>Chaining in cascading effects occur iff one of the following are true:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Input data items are corrupt, and the AND, OR or KEY dependency results in corrupt output data item(s).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Service logic of the process is corrupt resulting in corrupt output data items.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Service logic of the process is corrupt resulting in incorrectly written data sources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4. Service logic of the process is corrupt, and invokes incorrect processes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5. Service logic of the process is corrupt, and produces corrupt signaling messages.</td>
</tr>
</tbody>
</table>

contains corrupt data item $d_A$ indicated by $d_A^*$, and hence spreads corruption from block $B_i$ to $B_j$.

We further classify *intra-block dependency* as either (1) *derivative dependency* that defines relationship between data items in a block; or (2) *process to data source dependency* that defines relationship between a process, and the data source within
a block; or (3) *inter-process dependency* that defines relationships between processes within a block. Intra-block dependency is typically invoked in response to an inter-block dependency, i.e. the arrival of a signaling message, and the use of data items in these messages as input to the intra-block dependency (e.g. $P_1$).

*Derivative* dependencies typically cause the spread of corruption between data items within a block due to the *process functionality used in computing these data items*. The process functionality uses ‘n’ data items (received in messages or owned by the block) as input ($\text{fn}^{\text{Input}}$) with derivative dependency operators ($\text{Dep}\_\text{Operator}$) to derive output data items ($\text{fn}^{\text{Output}}$). The derivative dependency operators are AND, OR or KEY. Our IP rule 3 details the spread of corruption due to each derivative dependency operator. In the following, we also use examples to explain the semantics of each derivative operator.

Corruption propagation is exhibited due to *AND dependency operator* in process $P_1$ of Fig. 5.2 where, $d_A$ and $d_F$ are both corrupt and used as input to an AND derivative operator to compute, and thereby corrupt $d_G$. Process $P_2$ exhibits corruption propagation due to the *OR dependency operator* when data items $d_A$, and $d_C$ are used as input to compute output item $d_F$. $d_F$ is corrupt as a result of corrupt $d_A$ alone. Fig. 5.2 shows $P_1$ using data items $d_A$, and $d_B$ as the input to the *KEY operator* to retrieve $d_H$. $d_H$ is incorrect as the retrieval key $d_A$ is corrupt.

*Process to data source* dependency exists between processes, and the data sources within a block. It is due to the process reading $\text{Read}()$ data items from, and writing $\text{Write}()$ data items to data sources. Data items used in transactions with the data sources may be received in a message or owned by the block itself. In Fig. 5.2, process $P_2$ reads corrupt item $d_A$ and $d_C$ from, and writes corrupt items $d_F$ and $d_G$ the data sources. *Inter-Process* dependency occurs between processes in a block due to process invocation $\text{Invoke}()$ of other processes, or using data items as input in the invocation. In Fig. 5.2, process $P_1$ invokes process $P_2$ using incorrect data $d_H$ within block B$_j$.

Our IP rule 4 takes the various classes of intra-block dependency into consideration in defining the reasons for corruption of an output signaling message. In our final IP rule, we consolidate the various categories of dependencies, and IP rules to define reasons for chaining to occur.
5.3.2 Propagation Model

With the knowledge of the derivative dependency we can classify attacks based on the type of chaining. A *chain* is a result of the cascading effect, and it is comprised of a sequence of corrupt items, where each item is derived from the previous one. Chains may be classified as *linear* or *branching*. In the *linear chain*, a single corrupt data item is sufficient to corrupt the next item in the chain. For example, a single corrupt data item $d_1$ leads to corruption of data item $d_i$, and so on ($d_1 \rightarrow d_i \rightarrow \ldots \rightarrow d_n$). The *branching* chain is specifically caused by the AND derivative dependency; here all input items in the AND dependency must be corrupt to corrupt the next chain item. For example, both the data items $d_1$, $d_2$ in the AND input lead to corruption of $d_3$, the next chain item ($d_1 \& d_2 \rightarrow d_3$; $d_3 \& d_4 \& d_5 \rightarrow \ldots \rightarrow d_n$).

5.4 aCAT knowledge base

In this section, we present our aCAT knowledge base with the aim of illustrating its superiority in capturing dependencies in comparison to the CAT knowledge base. As we plan to use CAT to detect vulnerabilities in call origination, call delivery, and mobility management services, we have populated our database using the specifications for call handling [3], and for mobility [4]. We have also added a minor amount of expert input, in those areas where SDL is ambiguous, and lacking.

<table>
<thead>
<tr>
<th>Message Name</th>
<th>Data Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRI</td>
<td>Mobile Station International ISDN Number (MSISDN), Alerting Pattern, CUG interlock, CUG outgoing message, ISDN BC, ISDN LLC, ISDN HLC, GMSC Pre-paging support.</td>
</tr>
</tbody>
</table>

The tables in the aCAT knowledge base are shown in Table 5.2, and Table 5.3. Although the overall structure of the aCAT knowledge base is similar to the CAT knowledge base. On observation, we can see a major difference in the action column of
the process table. Subroutine’s without any relationships (dependencies) have been replaced with intra-block dependencies. For example, the statement Derive GSM BC from ISDN compatibility information is replaced by an OR derivative dependency \( f(ISDNBC \lor ISDN LLC \lor ISDN HLC) = \text{GSM Bearer Capability} \) (first part); the statement Derive basic service from GSM BC is replaced by an KEY derivative dependency \( (\text{GSM Bearer Capability} \sim) = \text{Basic Service} \) (second part); and the statement Set Pre-paging support is replaced by an AND derivative dependency \( (\text{HLR Pre-paging support} \land \text{GMSC Pre-paging support} \land) = \text{Pre-paging support} \) (fourth part). These statements are a direct translation from the SDL.

Table 5.3. Table 2-aCAT Knowledge Base: Processes

<table>
<thead>
<tr>
<th>Process Name</th>
<th>Block Name</th>
<th>Initial State</th>
<th>Input Message</th>
<th>From Block</th>
<th>Action</th>
<th>Output Message</th>
<th>To Block</th>
<th>Final State</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRI_HLR</td>
<td>HLR</td>
<td>Idle</td>
<td>SRI</td>
<td>GMSC</td>
<td>(1) (ISDN BC</td>
<td></td>
<td>ISDN LLC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(2) (GSM Bearer Capability \sim) = Basic Service</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(3) (MSISDN \sim) = IMSI; (MSISDN \sim) = HLR Pre-paging support; (MSISDN \sim) = MSC#, VLR#, LMSI</td>
<td>← Expert Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(4) (HLR Pre-paging support \land \text{GMSC Pre-paging support} \land) = \text{Pre-paging support}</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We also added some expert input to the table (third part). For example, the MSISDN is used to retrieve the IMSI sent in the output message ‘PRN’, hence we added the following KEY derivative dependency \( (\text{MSISDN} \sim) = \text{IMSI} \). We also identified that the MSISDN retrieves the HLR pre-paging support used to compute pre-paging support, hence we added the following KEY derivative dependency \( (\text{MSISDN} \sim) = \text{HLR Pre-paging support} \); As the location information needed to route the output message ‘PRN’, we added the following KEY derivative dependency \( (\text{MSISDN} \sim) = \text{MSC#}, VLR#, LMSI \).
In addition to the derivative dependencies presented in Table 2 of the aCAT knowledge base, the action column of Table 2 may also contain the following: (a) process and data source dependencies represented by reads \texttt{Read(d_x)}, and writes to the database \texttt{Write(d_x)}; and (b) inter-process dependencies represented by invocation to other processes \texttt{Invoke(d_x)}.

Thus using Table 1 in the aCAT knowledge base, we can identify the inter-block dependencies, and by using Table 2 we can identify all the intra-block dependencies in the network dependency model. This a major difference from the CAT knowledge base. Although the CAT knowledge base allowed accurate identification of the inter-block dependencies, the intra-block dependencies are often vague, and in-explicit. Hence cascading effects were sometimes mis-identified or missed. aCAT is devoid of these problems as it the CAT knowledge base accurately contains all the dependencies.

5.5 Algorithms

In this section we describe the forward chaining, reverse chaining, and combinatorial algorithms in detail. All of the algorithms use the aCAT knowledge base to detect attacks, and their cascading effects. The algorithms are followed by a discussion on loop elimination, and pruning used by the three algorithms.

aCAT works by taking either seeds or goals as input. When the input is a seed, the forward chaining algorithm is used. When multiple seeds are the input, attack graphs are built using the forward combinatory algorithm. Forward algorithms build the attack graph bottom-up. The algorithms start by identifying the conditions of the attack, followed by the deduction of the chain.

When the input is a goal, the reverse chaining algorithm is used. When multiple goals are the input, attack graphs are built using the reverse combinatory algorithm. The reverse algorithms search for the data item (seed) whose corruption of which leads to a chain in which the last item is the goal input by the user. Both the forward, and the reverse algorithms are depth-first search algorithms.

aCAT algorithms are exhaustive as they check every row in process table of the aCAT knowledge base (indicated by line 2 in Algorithm 4, 5). The graphs generated by
aCAT are succinct as only those states from which the adversary reaches the goal are added to the attack graph.

5.5.1 Forward Algorithm

The forward algorithm takes in seed(s) as input, and determines the chaining, in other words the cascading effects due to corrupt seed(s). The forward algorithm locates the seed in the target vulnerabilities specified in the threat model. The algorithm assumes that the adversary satisfied the necessary conditions to launch the attack, and that the seed can be corrupted. The seed is added to the chain. The block at which the seed occurs or the destination block of the message at which the seed occurs, is the check-point-block $B_{chk}$. Chaining is tracked by performing the forward dependency check at each ‘check point’ block. The forward dependency check for the single seed case is presented in Algorithm 4. The multiple seed case is explained in combinatorial algorithm is presented in Algorithm 6.

The forward dependency check consists of the two parts. Part 1 detects the next item in the chain (lines 3-11 in Algorithm 4), and Part 2 detects if the corrupt current chain items are propagating to other blocks (lines 12-17 in Algorithm 4).

Part 1 detects the next item in the chain by using [IP rule 5]. The input to this part comprises of user input seeds. The user input and the next item(s) in the chain (detected by Part 1) are the items currently corrupt in the check point block $B_{chk}$, and are stored in local corrupt items $l$.

If the input seed(s) occur as an input data item in a ‘derivative dependency’ and the dependency operator determines that the output is corrupt if the input is corrupt. Then by IP rule 1, the output item is added to the chain, and local corrupt items. IP rule 2 may be applied to ‘derivative dependencies’ assuming that output items are corrupt if the process service logic is corrupt. Lines 4-5 in Algorithm 4 indicating 1, 2 in [IP rule 5] is called the forward dependency rule F1.

If the seed(s) occurs in the following ordered sequence of dependencies as $\text{Write()}$, $\text{Read()}$, and input in a ‘derivative dependency’ where the output item is dependent on the input, then by 3 in [IP rule 5], the output item is added to the chain, and local
corrupt items (indicated by lines 6-7 also called the forward dependency rule F2). This rule assumes that the corrupt item is added to the common source, which is then read by other processes, and used to incorrectly derive other items.

If the seed(s) occurs in the following ordered sequence of dependencies as a new process invocation \texttt{Invoke()}, where the called process uses the seed as input in a ‘derivative dependency’ and the output item is dependent on the input, then by 4 in [IP rule 5], the output item is added to the chain, and local corrupt items (indicated by lines 8-9 also called the forward dependency rule F3). This rule assumes that corrupt items are passed on to other processes which use them to incorrectly derive other items.

Propagation of corruption from the current check point block to the other blocks is detected when any item in the local corrupt items occurs in an outgoing message from the check point block. The forward dependency check is conducted at the ‘destination block’ of the message. The ‘destination block’ is now the new ‘check point’ block, and the local corrupt items for the new ‘check point’ block are the local corrupt items occurring in the message to the new ‘check point’ block next. This part also enables detection of chaining by rule 5 indicated by lines 12-17 (called the forward dependency rule F4). 5 of [IP rule 5] holds if 1 in [IP rule 5] holds, and part 2 detects outgoing messages with output items of the dependencies.

The ‘forward dependency check’ is repeated until one of the following two terminating conditions are reached: (1) there are no outgoing messages from the checkpoint block with corrupt items i.e., next = \(\emptyset\); or (2) there are no more messages to explore in the aCAT knowledge-base.
**Input**: Seed; **Output**: Chain $C = \{c_0, c_1, c_2 \ldots c_n\}$, finite sequence of corrupt items.

**Bchk**: Check Point Block: Block containing chain item, or the destination block of the message carrying the chain item.

Let $l = \{l_0, l_1 \ldots l_k\}$ indicate the set of corrupt items in $B_{chk}$.

Let next indicate the set of local corrupt items in message propagating from $B_{chk}$ to next block.

Items in $C$ and $l$ are corrupt. $C \Rightarrow corrupt(C)$ and $l \Rightarrow corrupt(l)$

**Procedure Forward dependency check(seed):**

**Require**: $C[0] \leftarrow seed$; chain_index $\leftarrow 0$; Add seed to $l$

1. repeat
2. for $(\forall l_i \in l)$ and $(\forall Process in B_{chk})$ do
3. \{...... Start Part 1\}
4. if $(l_i \in fn^{Input}$ and Dep Operator $) \Rightarrow corrupt(fn^{Output})$ then
5. $C[++chain\_index] \leftarrow fn^{Output}$; Add $fn^{Output}$ to $l$ \{F1 \ldots 1,2 of IP rule 5\}
6. else if $(l_i \in \{Write()\), Read(), fn^{Input}\}$ and Dep Operator $) \Rightarrow corrupt(fn^{Output})$ then
7. $C[++chain\_index] \leftarrow fn^{Output}$; Add $fn^{Output}$ to $l$ \{F2 \ldots \ldots 3 of IP rule 5\}
8. else if $(l_i \in \{Invoke(P_x)\), fn^{Input}p_x\}$ and Dep Operator $) \Rightarrow corrupt(fn^{Output})$ then
9. $C[++chain\_index] \leftarrow fn^{Output}$; Add $fn^{Output}$ to $l$ \{F3 \ldots \ldots 4 of IP rule 5\}
10. end if
11. \{...... End Part 1\}
12. \{...... Start Part 2\}
13. if $\exists i \in message(B_{chk}, B_j, containsData\{d_a\})$ then
14. next $= \{l_i \in message(B_{chk}, B_j, containsData\{d_a\})\}$
15. $l = next$; $B_{chk} = B_j$ \{F4 \ldots \ldots 5 of IP rule 5\}
16. end if
17. \{...... End Part 2\}
18. end for
19. until next $\neq \emptyset$

**End-Procedure**

**Algorithm 5**: Forward Dependency Check
5.5.2 Reverse Algorithm

The reverse algorithm takes a goal(s) as input, and works backwards in a depth first search to locate the set of all seeds, whose corruption of which leads to the goal. The algorithm locates the goal data item in a block or a message under the assumption that the corruption of the goal is the cascading effect of corrupt seed(s). This algorithm works backwards to find the other chain elements \( \{c_n, c_{n-1} \ldots c_1\} \) leading to the seed \( c_0 \). The algorithm starts by adding the goal to the chain. The block at which the goal occurs or the origin block of the message at which the goal occurs, is the check-point-block \( B_{chk} \). Chaining is tracked by performing the reverse dependency check at each ‘check point’ block. The reverse dependency check is explained for the single goal case in Algorithm 5.

The reverse dependency check consists of the two parts. Part 1 detects the previous chain items (lines 3-10 in Algorithm 5), and Part 2 detects if the these previous chain items have propagated to \( B_{chk} \) (lines 11-14 in Algorithm 5). Part 1 detects the previous item in the chain by using rules specified in [IP rule 5]. The user input (goal), and previous item(s) in the chain (detected by part 1) are the items currently corrupt in the check point block \( B_{chk} \), and are stored in local corrupt items \( l \).

If the user input goal(s) occur as an output data item in a ‘derivative dependency’, the dependency operator determines if the input may be corrupt if the output is corrupt. Then by 1, 2 in [IP rule 5], the input items are added to the chain, and local corrupt items. Lines 3-4 in Algorithm 5 indicating 1, 2 in [IP rule 5], are called the reverse dependency rule R1. The reverse dependency check aims to detect the seed of the cascading effect, and hence the source of the input items in the ‘derivative dependency’ must be traced.
Input: Goal; Output: Chain $C = \{c_n, c_{n-1}, c_{n-2} \ldots c_1, c_0\}$, finite sequence of corrupt items.
Let $l = \{l_0, l_1 \ldots l_n\}$ indicate the set of corrupt items in $B_{chk}$.
Let $next$ indicate the set of local corrupt items in message arriving at $B_{chk}$ from previous block.
Items in $C$ and $l$ are corrupt. $C \Rightarrow corrupt(C)$ and $l \Rightarrow corrupt(l)$

Procedure Reverse dependency check($goal$):
Require: $C[n] \leftarrow goal$; chain_index $\leftarrow n$; Add goal to $l$

1: repeat
2: \hspace{0.5cm} for $(\forall l_i \in l)$ and $(\forall$ Process in $B_{chk}$) do
3: \hspace{1cm} if $(l_i \in fn_{Input}^{P_x})$ and $fn_{Dep\_Operator}^{P_x} \Rightarrow corrupt(fn_{Input}^{P_x})$ then
4: \hspace{1cm} C[chain_index] $\leftarrow fn_{Input}^{P_x}$
5: \hspace{1cm} Add $fn_{Input}^{P_x}$ to $l$ \hspace{1cm} \{R1 \ldots \ldots \text{1,2 of IP rule 5}\}
6: \hspace{1cm} if $(fn_{Input}^{P_x} \in \{fn_{Output}^{P_y}, write_{P_y}() , read_{P_x}()\})$ and $(fn_{Dep\_Operator}^{P_y} \Rightarrow corrupt(fn_{Input}^{P_y}))$ then
7: \hspace{1cm} C[chain_index] $\leftarrow fn_{Input}^{P_y}$
8: \hspace{1cm} Add $fn_{Input}^{P_y}$ to $l$ \hspace{1cm} \{R2 \ldots \ldots \text{3 of IP rule 5}\}
9: \hspace{1cm} else if $(fn_{Input}^{P_x} \in \{fn_{Output}^{P_y}, invoke_{P_y}(P_x)\})$ and $(fn_{Dep\_Operator}^{P_y} \Rightarrow corrupt(fn_{Input}^{P_y}))$ then
10: \hspace{1cm} C[chain_index] $\leftarrow fn_{Input}^{P_y}$
11: \hspace{1cm} Add $fn_{Input}^{P_y}$ to $l$ \hspace{1cm} \{R3 \ldots \ldots \text{4 of IP rule 5}\}
12: \hspace{1cm} end if
13: \hspace{1cm} end if
14: \hspace{1cm} if $\exists l_i \in$ message($B_j , B_{chk} , containsData\{d_a\}$) then
15: \hspace{1cm} next $= \{l_i \in$ message($B_j , B_{chk} , containsData\{d_a\})\}$
16: \hspace{1cm} $l = next$
17: \hspace{1cm} $B_{chk} = B_j$ \hspace{1cm} \{R4 \ldots \ldots \text{5 of IP rule 5}\}
18: \hspace{1cm} end if
19: \hspace{1cm} end for
20: until $n \neq 0$
End-Procedure

**Algorithm 6**: Reverse Dependency Check
The source of the input items in the ‘derivative dependency’ may be one of the following:

(1) Computed by another process \( y \) which may have written this value to the database, and read by the current process. Hence based on dependency operator of process \( y \), it is possible for the input items of ‘derivative dependency’ in process \( y \) to be corrupt, and hence added to the chain, and local corrupt items (indicated by lines 5-6 also called the reverse dependency rule R2). This rule is an application of 3 in [IP rule 5], and may be detected if the input items occurs in the following ordered sequence of dependencies as \( \{ \text{fn}_{P_y}^{\text{Output}}, \text{Write}_{P_y}(), \text{Read}_{P_x}() \} \).

(2) Computed by another process \( y \) which may have invoked the current process with the input items of ‘derivative dependency’ as parameters. Hence based on dependency operator of process \( y \), it is possible for the input items of ‘derivative dependency’ in process \( y \) to be corrupt, and hence added to the chain, and local corrupt items (indicated by lines 7-8 also called the reverse dependency rule R3). This rule is an application of 4 in [IP rule 5], and may be detected if the input items occurs in the following ordered sequence of dependencies as \( \{ \text{Invoke}(P_x), \text{fn}_{P_x}^{\text{Input}} \} \).

(3) Arrived on incoming messages into the block (indicated by lines 11-14 also called the reverse dependency rule R4). The reverse dependency rule 4 is the application of 5 in [IP rule 5], and forms part 2 of the reverse dependency check. The reverse dependency check is conducted at the ‘origin block’ of the message. The ‘origin block’ is now the new ‘check point’ block, and the local corrupt items for the new ‘check point’ block are the local corrupt items occurring in the message from the new ‘check point’ block next.

The ‘reverse dependency check’ is repeated until one of the following two terminating conditions are reached: (1) there are no incoming messages into the checkpoint block with corrupt items i.e., next = \( \emptyset \); or (2) there are no more messages to explore in the aCAT knowledge-base. At this point the last chain element detected is the seed. The algorithm locates the seed in the target vulnerabilities specified in the threat model, and assumes that the adversary has the necessary conditions for the attack, and that the seed can be corrupted.
The complexity of the forward and reverse algorithms is similar, and their worst case performance is \( O(\# \text{ of } B_{chk} \times \# \text{ of processes in } B_{chk}) \). It may be approximated as \( O(p) \) where \( p \) is \( \# \text{ of process CEFSM in aCAT knowledge base} \), and \( p \simeq (\# \text{ of } B_{chk} \times \# \text{ of processes in } B_{chk}) \).

### 5.5.3 Combinatory algorithms

The Combinatory algorithms take multiple seeds or goals as input, and detect multiple chains of corrupt items as output. These algorithms detect individual cascading effects of a single input item as well as the combined cascading effect of multiple input items. To detect the combinatory effect, a global array is maintained for each individual block (denoted by \( G_{HLR}[], \ldots G_{GMSC}[] \)). This array contains items corrupt at a block due to the various input items.

The forward combinatory algorithm (shown in Algorithm 6) works by taking in multiple seeds as input, and determines the chain items. The algorithm starts by detecting the chain for \( \text{seed}_1 \) (using the forward dependency check algorithm shown in Algorithm 4), and storing the corrupt chain items in the global array for the respective block (Lines 1-2 in Algorithm 6). The \( \text{seed}_1 \) chain stored in the global array is used to check for the combined effect of the input seeds. For example, if \( \text{seed}_1 \) and \( \text{seed}_2 \) occur in a block \( X \), then the effect of corrupt \( \text{seed}_1 \) and \( \text{seed}_2 \) is checked based on the forward dependency rules.

A new chain is started for each input seed and the check-point-block \( (B_{chk}) \) (block at which the seed occurs or the destination block of the message at which the seed occurs) is detected for the input seed. The other corrupt items in this check-point-block are fetched from the global array (line 7). The other corrupt items, and the input seed are added to the local corrupt items \( l \) (line 8). They are then used to detect the next item in the chain based on forward dependency rules 1-4 (lines 10-20). This procedure is repeated until the rest of the items, and hence the last chain item, the goal is detected. This chain contains the individual effect of \( \text{seed}_2 \) and the combined effect of \( \text{seed}_1 \), and \( \text{seed}_2 \). The new chain items are stored in the global array for the respective blocks, and are used to check for the combined effect of the input seeds. Addition of corrupt items
from global array to \( l \), the local corrupt items, ensures detection of the combined effect of the input seeds. Thus chains are detected, and the chain for a \( \text{seed}_j \) contains the individual effect of \( \text{seed}_j \), and the combined effect of \( \text{seed}_1 \ldots \text{seed}_j \).

The reverse combinatory algorithm is very similar to the forward combinatory algorithm except that the forward dependency rules are replaced by the reverse dependency rules i.e, lines 10-20 of forward combinatory algorithm in Algorithm 6 are replaced by lines 3-12 of the reverse dependency check in Algorithm 5.

The problem with the combinatory algorithm is the explosion in the number of cases to be considered during chain detection. Consider 3 data items each of which occurs in 3 different network locations. In the algorithm, if the order of corruption is considered i.e., corruption of each seed or goal item at each of its network locations relative to every other item at their locations, the total number of cases to be explored is \( (3C_1 \times 3) \times (3C_1 \times 3) \times (3C_1 \times 3) = 729 \). Most of these 729 cases are redundant. To eliminate redundant cases, and efficiently detect chain items without losing completeness the combinatory algorithms apply the following optimization techniques.

- **Order of Corruption**: The order of corruption considered in the algorithm is the order of input provided by the user i.e., the combinatory algorithm considers the first user input as the first corrupt item, the second user input as the second corrupt item, and so on. For example, user input is the following order 1.\( \text{seed}_x \), 2.\( \text{seed}_y \), and 3.\( \text{seed}_z \). The combined effect chain considers the corruption of \( \text{seed}_x \) first, the corruption of \( \text{seed}_y \) second, and the corruption of \( \text{seed}_z \) third.

- **Reducing start points**: This optimization applies to the forward algorithm. Forward algorithms start by detecting the check-point-block (\( B_{\text{chk}} \)), the block at which the seed occurs. In considering the following rules the number of starting check-point-blocks may be reduced.
Input: \{Seed_1, Seed_2 \ldots, Seed_m\}

Output: Chain \( C = \{c_0, c_1, c_2 \ldots c_n\} \) \( N \) multiple chains of corrupt items; 
\( N \) is the total number of chains generated.

Let \( C^i \) denote the \( i \) th chain and \( l = \{l_0, l_1 \ldots l_k\} \) indicate the set of corrupt items in \( B_{chk} \).
Let next indicate the set of local corrupt items in message propagating from \( B_{chk} \) to next block.

Items in \( C^i \) and \( l \) are corrupt i.e. \( C^i \Rightarrow corrupt\) (\( C^i \)) and \( l \Rightarrow corrupt\) (\( l \))

Global Array of corrupt block items: contains data items corrupt at a block due to the input seeds.
E.g. \( G_{HLR}, G_{VLR}, G_{MSC}, G_{GMSC} \)

Let \( g_i \) denote set of corrupt items in global array of block \( B_{chk} \)

Procedure Forward Combine check(\{Seed_1,Seed_2 \ldots,Seed_m\});

1: Call Forward dependency check(Seed_1)
2: Call Store Global Array(\( C_0 \)) \{Chain items from Seed_1 are stored in global arrays\}
3: for (\( \forall s_i \in \{\text{Seed}_2 \ldots \text{Seed}_m\} \)) and (\( \forall j \text{ such that } 1 \leq j \leq N \)) do
4: \( C_j \leftarrow s_i; \) \( l \leftarrow \emptyset; \) \( l \leftarrow s_i \ldots \ldots \{\text{Starting a new chain}\}\)
5: Detect \( B_{chk} \) for seed \( s_i \)
6: for (\( \forall l_i \in l \)) and (\( \forall \text{Process in } B_{chk} \)) do
7: \( g_i = \text{Fetch}_\text{Global_Array}(B_{chk}) \)
8: Add \( g_i \) to \( C_j \) and \( l \ldots \ldots \{\text{Ensures combined effect of previously corrupt seeds}\}\)
9: \ldots \ldots \{\text{Detecting next chain item in block } B_{chk}\}\)
10: if (\( l_i \in \text{fn}\) Input \( \text{ and Dep}_{\text{Operator}} \Rightarrow corrupt(\text{fn Output})\) then
11: Add \( \text{fn}\) Output to \( C_j, l \) \{F1 \ldots \ldots 1,2 \text{ of IP rule 5}\}
12: else if (\( l_i \in \{\text{Write()}, \text{Read()}, \text{fn}\) Input \( \} \) and \( \text{Dep}_{\text{Operator}} \Rightarrow corrupt(\text{fn}\) Output\) then
13: Add \( \text{fn}\) Output to \( C_j, l \) \{F2 \ldots \ldots \text{Rule 3 of IP rule 5}\}
14: else if (\( l_i \in \{\text{Invoke}(P_x), \text{fn}_{P_x}\} \) and \( \text{Dep}_{\text{Operator}} \Rightarrow corrupt(\text{fn}\) Output\) then
15: Add \( \text{fn}\) Output to \( C_j, l \) \{F3 \ldots \ldots \text{Rule 4 of IP rule 5}\}
16: end if
17: if \( \exists l_i \in \text{message}(B_{chk}, B_j, \text{containsData}{d_a})\) then
18: \( \text{next} = [l_i \in \text{message}(B_{chk}, B_j, \text{containsData}{d_a})]\)
19: \( l = \text{next; } B_{chk} = B_j \) \{F4 \ldots \ldots 5 \text{ of IP rule 5}\}
20: end if
21: end for
22: \( j++\)
23: end for

End-Procedure

Algorithm 7: Forward Combinatory algorithm
• If a data item is found both in ‘message’, and ‘destination block’, the two locations may be reduced to a single case by considering the effect of corruption at the incoming message alone. This is because corrupting the data item at the ‘incoming message’ will corrupt the data item stored in the destination block. Consider Fig. 5.2; seed $d_A$ occurs in message $M_1$ and in block $B_j$. The corruption of seed $d_A$ in message $M_1$ eventually spreads to block $B_j$, and the effect of corruption of $d_A$ is seen in block $B_j$.

• If a data item is found both in a message, and in the ‘origin block’ but not in the ‘destination block’, the two locations may be reduced to a single case by considering the effect of corruption at the ‘origin block’ alone. This is because corrupting the data item at the ‘origin block’ will corrupt the data item in the outgoing message. Consider Fig. 5.2; seed $d_G$ occurs in outgoing message $M_3$, block $B_j$, but not in block $B_k$. The corruption of seed $d_G$ in block $B_j$ eventually spreads to block $B_k$ in message $M_3$, and the effect of corruption of $d_G$ is seen at block $B_j$.

• Consolidating the check-point-blocks such that all other cases are considered.

Consider Fig. 5.2 with the following seeds, and their locations.

- Seed $d_A$ : Location: $M_1$, $B_j$, $M_2$, $M_3$,
- Seed $d_B$ : Location: $M_1$, $B_j$, $M_2$.

In this case it is sufficient to consider of seed $d_A$: Location: $M_1$, and Seed $d_B$: Location: $M_1$, because this case captures all of the other cases. Message $M_1$ with corrupt seeds $d_A$ and $d_B$ eventually propagates to block $B_j$, messages $M_2$ and $M_3$, and to block $B_k$.

By applying the above optimization rules the total number of scenarios considered may be reduced to the maximum of optimized number of of data item occurrences $\max[\text{data item occurrences}_{1,...,j}] \times i$; where $i = \text{total number of data items input}$. In the case discussed above, if four is the maximum of optimized number of of data item occurrences this results in $4 \times 3 = 12$ cases, a large reduction from the original 729 cases.

Consider the following random scenario from the call delivery service.

• Data 1: MSISDN : 13 points of attack.
• Data 2: IMSI : 5 points of attack.
• Data 3: MSRN: 5 points of attack.

If these data items are used as seeds, and each data item at each of its network locations is considered relative to every other item at their locations, the total number of cases to be explored is 
\[13C_1 \times 3 \times 5C_1 \times 3 \times 5C_1 \times 3 = 8775.\]
Using the above optimization rules the number of starting point scenarios for attacks that consider these parameters may be reduced as follows.

• Data 1: MSISDN : 5 points of attack
• Data 2: IMSI : 3 points of attack.
• Data 3: MSRN: 2 points of attack.

The total number of scenarios to be considered may be reduced to 
\[\text{max} \left[\text{data item occurances}_{1, i}\right] \times i, \text{ i.e. } 5 \times 3\]
Therefore the numbers of cases considered is reduced from 8,755 to 15.

The complexity of the forward, and reverse combinatory algorithms is similar, and their worst case performance is \(O(Np)\) of where \(N\) is the total number of chains generated, and \(p\) is the # of rows in the process table of the aCAT knowledge base.

![Looping Example](image)

**Fig. 5.3.** Looping Example

All of the above algorithms use the following loop elimination conditions, and pruning methods.
**Loop Elimination:** Loops frequently arise in building attack graphs due to telecommunication semantics. For example, consider the seed ‘MSISDN’. This data item appears in messages IAM, SRI, PRN, PRN, ACK, and SRI, ACK. Tracking the corruption of seed ‘MSISDN’ from signaling messages IAM leads the algorithm to follow the corruption from message IAM to the GMSC, ‘SRI message’, the HLR, ‘PRN message’, the VLR, and back to the GMSC as shown in Fig. 5.3. This can be eliminated by checking looping conditions, and ensuring that the same path is not traversed twice for a seed.

**Pruning:** Trees are pruned, and merged into a graph. Trees are built bottom up. Every tree is assigned a tree number. Nodes in the tree are divided into layers. All nodes at a layer that are common are merged into a single node with the single node assuming all the tree numbers.

### 5.6 Interesting Attacks Discovered by aCAT

Our algorithms, detect the following interesting attacks. aCAT constructs attack graphs assuming the worst case pre-conditions to be true.

The alerting attack, and power-off power-on attack are discovered using only SDL knowledge. The call redirection attack is discovered with both expert & SDL knowledge.

#### 5.6.1 Alerting Attack

The attack graph for the alerting attack is shown in Fig. 5.6. With the help of the guidelines defined in Section 4.4.3 the following may be derived.

**Step 1: End User Effect:** Goal node(s) are used to infer the end effect of the attack on the subscriber. In the alerting attack, the goal nodes are Node A at layer 5, and Node C at layer 4. According to the goal node, the Page message to the BSS has incorrect goal item ‘page type’. The Page message is used to inform subscribers of the arrival of incoming calls, and ‘page type’ indicates the type of call. ‘Page type’ must be compatible with the subscriber’s mobile station or else the subscriber is not alerted. From the goal node it may be inferred that Alice, a subscriber of the system is not alerted on the arrival of an incoming call, and hence does not receive incoming calls.
Step 2: Origin of Attack: As nodes at layer 0 indicate the origin of the attack, we can infer that the alerting attack may originate at the following locations: signaling messages SRI, PRN, the service nodes VLR, or the HLR.

Step 3: Attack Propagation and Side effects: Nodes at all other layers show the propagation of corruption across the various service nodes in the network. In the alerting attack, from the other layers it may be inferred that the seed is the alerting pattern and the attack spreads from the HLR to the VLR, and from the VLR to the MSC. We found that some non-goal nodes (that indicate the propagation of corruption) also indicate side
effects on the user in addition to the goal node that shows the final effect on the user. Examples of these side user effects are provided in the next section.

![Diagram of Alerting Attack]

**Fig. 5.5. Alerting Attack**

*Attack Scenario:* Using the above guidelines the following attack scenario may be derived. Trudy, the adversary corrupts the alerting pattern of Alice, the victim, at the SRI message arriving at the HLR. When there is an incoming call for Alice, her profile is downloaded from the HLR to the VLR, and from the VLR to the MSC. The call proceeds as usual but Alice is not paged as required. Hence Alice’s mobile station cannot detect the incoming call, and the call is missed. This attack is subtle to detect because network administrators find that the network processes the incoming call correctly, and that the subscriber is alerted correctly. They may not find that this alerting pattern is incompatible with the mobile station itself. This attack is illustrated in Fig. 5.5.

### 5.6.2 Power-Off Power-On Attack

The attack graph for the power-off, and power-on attack is shown in Fig. 5.6. With the help of the guidelines defined in Section 4.4.3 the following may be derived.
Fig. 5.6. Attack Graph for Power Off Power On Attack

**Step 1: End User Effect:** Goal nodes (Nodes A, B and C) identify end subscriber effects. Node A identifies that subscribers are disabled when they move to a new location as the previous location cancellation does not occur. Nodes B and C identify that incoming calls are not received by the subscriber as they may be sent to incorrect locations (Node B) or because subscriber page messages are sent to incorrect locations, and hence not answered (Node C).

**Step 2: Origin of Attack:** Nodes at layer 0 in Fig. 5.6 indicate the target of the attack is the HLR.

**Step 3: Attack Propagation and Side effects:** Nodes at other layers indicate the seeds used in the attack as the MSC number, and VLR number. The MSC number and VLR number together indicate the current location of the subscriber. Hence corrupting the MSC number and VLR number corrupts the subscriber location. If the subscriber location is corrupt the network cannot locate the subscriber when a call arrives, resulting in resetting the location of the subscriber. The figure also illustrates the \textit{AND derivative dependency} (Nodes G, H, and D) and the \textit{branching chain} i.e., both the data items ‘MSC number’ and ‘VLR number’ must be corrupt to corrupt data ‘subscriber location’
(\(d_{MSC\ number} \& d_{VLR\ number} \rightarrow d_{subscriber\ location}\)). Using these guidelines and general knowledge of the network, the following attack scenario may be constructed.

**Attack Scenario:** This is an attack on the home network targeting the subscriber. This attack is illustrated in Fig. 5.7. Trudy, the adversary, corrupts the ‘MSC number’ and ‘VLR number’ of Alice, the victim, to an unresolved value in the HLR. The HLR uses these parameters to locate a subscriber when a call request arrives. As the parameters are corrupt, the HLR cannot locate Alice when a call arrives for her. The HLR thus resets Alice’s ‘MSC number’, and ‘VLR number’ effectively de-registering her. After this, Alice cannot receive calls (Node 10). Alice will only begin to receive calls again if she power-cycles her phone, or if she changes location triggering a location update, thus refreshing the HLR with her correct MSC, and VLR numbers. Note, that Trudy does not need to know valid MSC, and VLR numbers for this attack to be effective.

### 5.6.3 Mixed Identity Attack

The attack graph for the mixed identity attack is shown in Fig. 5.8. Using the guidelines in Section 4.4.3 attack scenario(s) may be derived as follows.

**Step 1: End User Effect:** Goal node(s) of Tree 1 and 3 (Nodes 23 and 21) identify the end effect as the inability of the subscriber to receive incoming calls due to incorrect
Fig. 5.8. Attack Graph for Mixed Identity Attack

paging. Goal node(s) of Trees 2 and 4 (Nodes 34 and 35) identify another effect of the attack as the inability of subscribers to authenticate themselves with the network.

Step 2: Origin of Attack: Nodes at layer 0 indicate the origin of the attack. From Fig. 5.8 it can be inferred that the two possible targets of the attack are the HLR or the VLR.

Step 3: Attack Propagation and Side effects: From other layers it can be inferred the seed used in the attack is the IMSI (Nodes 9, 12, and 13). The IMSI is the unique identity of a subscriber in the network. Corrupting the IMSI has a number of side effects as the IMSI is used as the key to retrieve incorrect parameters (Nodes 16, 17, 19, and 22).
Corrupting the IMSI at the HLR or in signaling messages for incoming calls (Trees 1 and 3), leads to subscribers being paged incorrectly leading to loss of incoming calls. When a subscriber moves to a new location, a corrupt IMSI at the HLR leads to registration cancelation of the incorrect subscriber in the old location (Tree 2, Node 20) and failed authentication of the subscriber at the new location (Tree 2, Node 34). The linear chain is demonstrated by Tree 2. Using these results and general knowledge of the network, the following attack scenarios may be constructed.

**Attack Scenario 1**: This is an attack on the home network targeting the network itself. In this attack, the identities of a group of victims are mapped to a designated victim's identity. The designated victim is charged for all the calls. This attack is illustrated in Fig. 5.8. Trudy maps the IMSI of all victims to that of Alice at the HLR. The VLR of location area B has uncorrupt data, and hence all the mobile stations will be enabled. When Alice moves to a new location, she requests the VLR in the new location area A to register her. The VLR in the new location area A registers Alice and requests the HLR to do the same. The HLR registers Alice, and cancels Alice’s previous location i.e., the current location of rest of the victim group. This results in cancellation of all non-traveler victims. Alice, the designated victim is charged for all victim group’s calls before the cancellation; after the cancellation, none of the non-traveller victims can receive calls.

**Attack Scenario 2**: Trudy replaces the IMSI of organization Alice Inc. (victim) in every incoming call, with IMSI of rival organization (Bob Inc.) (Tree 3, Node 12). Every call arriving at the network for Alice Inc. is received by Bob Inc.

### 5.6.4 Call Redirection Attack

The attack graph for the call redirection attack is shown in Fig. 5.10. Using the guidelines in Section 4.4.3 attack scenario(s) may be derived as follows.

**Step 1: End User Effect**: Goal nodes (Nodes 21 and 20) identify the end result of the attack as redirection of incoming calls to other subscribers. This is because the ‘Page message’ is routed to the incorrect subscriber (due to incorrect goal items TMSI and LAI).
Step 2: Origin of Attack: Nodes at layer 0 in Fig. 5.10 indicate that the target of the attack is the VLR.

Step 3: Attack Propagation and Side effects: Nodes at the other layers indicate the seed used in the attack is the MSRN (roaming number). The MSRN is used to route incoming calls to the designated subscriber. Each incoming call is assigned a MSRN. By switching the MSRNs (Nodes 10 and 11) it is possible to switch the destination of the incoming call (Nodes 12 and 13), and send the call to the incorrect subscriber (Nodes 20 and 21). Tree 1 and 2 are examples of the line cascading effect.

Using the above guidelines, and general knowledge of the network, the following attack scenario may be constructed.

Attack Scenario 1: This is an attack on the visiting network targeting the network itself. In this attack, the roaming number of subscriber Bob is corrupt (possibly by incrementing or decrementing the number). This results in the redirection of the Bob’s calls to another subscriber Alice (to whom the corrupt roaming number has been assigned). Hence Alice is alerted of the incoming call instead of Bob. Another side effect of corrupting the MSRN is that if the MSRN is not assigned to any other subscriber the call is dropped. For every incoming call arriving at the MSC for Bob, the adversary changes the roaming number to an unassigned value, in the incoming call. Hence Bob is
not alerted of the incoming call. Bob can make calls, and update his profile but can never receive his calls at a particular location and is unaware of this problem. This problem may be rectified if Bob travels to a new location. This attack is illustrated in Fig. 5.11.

5.6.5 Missed Calls Attack

The attack graph for the missed calls attack is shown in Fig. 5.12. With the help of the guidelines defined in Section 4.4.3 attack scenarios may be derived as follows.
**Step 1: End User Effect:** The goal node (Node 13) identifies that incoming calls are not received by the subscriber as Page messages are sent to incorrect location (incorrect LAI). Goal node (Node 17) identifies that subscribers are disabled when they move to a new location because updates due to location changes (Update location message) contain incorrect subscriber identity (incorrect IMSI) resulting in cancelation of the update.

**Step 2: Origin of Attack:** Nodes at layer 0 in Fig. 5.12 indicate that the target of the attack is the VLR.

**Step 3: Attack Propagation and Side effects:** Nodes at the other layers indicate the seed used in the attack is the LAI (Location Area Identifier). The LAI identifies the location the subscriber is currently visiting. The combination of the TMSI and LAI uniquely identifies a subscriber and their current location i.e., at the VLR. Corruption of either the TMSI or the LAI results in corruption of subscribers identity (Node 14). Node 14 is an illustration of an OR node.

Using the above guidelines and general knowledge of the network, the following attack scenarios may be constructed.
Attack Scenario 1: This is an attack on the visiting network targeting the subscriber. In this attack, the location area (LAI) of the victim is corrupt during the incoming call profile download. The corrupt LAI causes the subscriber to be paged at the incorrect location for incoming calls, and results in the subscriber not receiving the call. Subscribers can make calls, and update their profile but can never receive incoming calls at this particular location. Receiving calls if moved to another location is possible. This attack is illustrated in Fig. 5.13.

Attack Scenario 2: This is another attack scenario derived from the Tree 2 in the attack graph shown in Fig. 5.13. This is an attack on the visiting network targeting the network itself. The victim in this attack is Wireless XYZ Inc. (the company providing
service in a location area A). Subscribers currently visiting this area A, cannot register with the network. This because the subscriber’s LAI in the update location message is corrupt, resulting in retrieval of incorrect IMSI, and authentication material (Node 14). The subscriber responds incorrectly to the incorrect authentication material, and as a result cannot register with the network. An adversary may capture all update location messages on the air interface, and corrupt the LAI’s thereby preventing subscriber registration.

5.7 Impact of Expert Input

5.7.1 Experimental Results

We conducted experiments to find the percentage of attacks detected using SDL, and expert knowledge. We tested for single and double seed combinations in the forward (shown in Fig. 5.14(a)), and reverse chaining algorithms (shown in Fig. 5.14(b)). When the user input corresponds to areas where SDL is adequate, we found that expert knowledge does not show any improvement in number of attacks detected. On an average, goals detected by SDL knowledge cover 70% of all the possible goals that are detected with expert input. The seeds detected by SDL knowledge cover 77% of all of the possible seeds that are identified with expert input.
The results of these experiments indicate the usefulness of SDL knowledge for both aCAT, and adversaries. For aCAT, SDL provides a significant portion of the knowledge base. For adversaries, SDL provides a rich set of possible attacks. In fact, it provides information that may be used to launch the large majority of possible attacks. Furthermore, the remaining attacks are still possible even without the direct targeting of an adversary.

### 5.7.2 Limitations of SDL

SDL specifications are free, easy to obtain, and contribute to 70% of the total network knowledge in aCAT. We identify the inadequacies in SDL, and detail how aCAT alleviates these problems. In doing so, we prove that aCAT can be an important tool in uncovering sophisticated cascading attacks in advance.
5.7.2.1 Syntax and Semantics

Due to limitations in syntax and semantics SDL is inadequate in defining process functions in service logic of processes. The limitations are as follows.

- SDL does not distinguish between cached data received from other blocks/processes, and data owned by the block.
- SDL does not present explicit specification of subroutines in processes.
- SDL does not explicitly specify the input, and output data items in process functions in all the cases.
- SDL does not explicitly specify the relationship between input, and output data items in process functions in all the cases.

Without this information chaining and cascading effects are not accurately detected. If the chaining and cascading effects are inaccurate, in-exhaustive attack graphs with false positives and negatives are produced. By applying the network dependency model on the SDL it can be ensured that chaining, and cascading effects are accurately detected, and exhaustive attack graphs are produced.

5.7.2.2 Technical areas

SDL makes many assumptions in specifying certain technical areas. These assumptions are not obvious, and hence if not taken into consideration would result in incomplete SDL knowledge. Incomplete SDL knowledge results in reduced identification of attacks. By recognizing the following inadequate areas, and compensating it with expert input coverage of attacks may be increased.

- **Message and Message-ACK Pairs**: SDL does not consider the data dependency that arises due to message-ack pairs e.g., message ‘SRI’ and ‘SRLACK’, ‘PRN’ and ‘PRN_ACK’. When a block receives the message (of the message, and message-ACK pair), it retrieves a data item (called key) from the message, and assigns a value to this key item. The block responds with this value in the message-ACK.
SDL does not capture the fact that this value has been assigned for the key data item (retrieved from the Message). This is because when the originating block sends out the Message it opens a connection with the destination block, and receives the Message-ACK (with value) on the same connection. SDL implicitly assumes that the value has been assigned for the key, and does not represent it. Expert input takes this into consideration resulting in detection of greater number of attacks.

For example, the VLR receives the ‘PRN’ message it assigns a ‘MSRN’ for the given ‘IMSI’ (International Mobile Subscriber Identity). The VLR sends this ‘MSRN’ back to the HLR in the ‘PRN_ACK’ message. SDL does not capture the fact that the ‘MSRN’ has been assigned for the ‘IMSI’. SDL implicitly assumes that the ‘MSRN’ has been assigned for the ‘IMSI’, and does not represent it.

- **Authentication and Ciphering**: With respect to authentication and ciphering based messages, and process, SDL generally does not provide the data items involved, and data dependencies. It provides the name of the function such as ‘Obtain Authentication Sets at VLR’, and does not give any other information. Identifying data items and dependencies with respect to authentication and ciphering based processes identifies possible attacks, and effects on authentication processes.

- **Location items**: Location items such ‘VLR number’, ‘MSC number’, and ‘MSRN’ indicate the location where the subscriber is currently visiting. These items are used to route signaling messages to the subscriber’s visiting location. If these items are corrupt the signaling messages are routed to the incorrect location. This type of attack is not detected because SDL does not take into consideration the items used to route signaling messages.

By hardcoding the above cases into algorithms the coverage of SDL knowledge vulnerability assessment may be increased. Adversaries may also alleviate the inadequacies in SDL, and devise much more devastating attacks, by careful in-depth study of the specifications or by gaining unauthorized access to specific requirements documents. We would also like to state that attacks can still be devised even without this extra
knowledge, even if some of the sophisticated cascading effects are unanticipated by the adversary.

5.8 Discussion

In conclusion, we have successfully developed a complete, and efficient solution for vulnerability assessment in mobile telecommunication networks. Our solution is unique due to the incorporation of the network dependency model, IP rules, and expert input. Our solution is also efficient which is evident in the interesting attacks detected. We propose to extend aCAT to analyze our attack graphs, and support cross network service vulnerability assessment.
Chapter 6

eCAT

6.1 Introduction

The mobile telecommunication network is at the early stages of security development. Hence, a major effort is underway to secure these networks. As a result, many new security protocols are being proposed.

While the design goals and threat model of these security protocols may be common knowledge, it would be beneficial to find: (1) the exact protection coverage of these security protocols, in terms of percentage of attacks prevented; (2) the other kinds of security mechanisms required to tackle the attacks that can evade the security protocol under observation; and (3) the most vulnerable network areas and check if the security protocol under observation can protect them. Such information gives a good evaluation of the security protocol before it can be deployed.

eCAT uses graph technology, boolean probabilities, and Coverage Measurement Formulas (CMF) to: (1) Measure effective coverage; (2) Identify the types of required security mechanisms to protect the network; and (3) Identify the most vulnerable network areas (also called hot-spots).

eCAT uses attack graphs to capture the effective coverage. Although attack graphs for mobile telecommunication networks have been generated by approaches such as aCAT (Chapter 5), these approaches are limited in performing quantitative security protocol coverage analysis. In particular, they are defense agnostic and cannot measure the coverage of security protocols; and they cannot detect network areas most in need of protection. eCAT extends these approaches with a set of new capabilities. It uses a boolean probability based graph marking algorithm to measure the effective coverage, and quantifies the same with CMF’s.
Our graph based approach is motivated by a key observation that attack graphs can not only capture the effects of the potential attacks against a mobile telecommunication network, but can also capture the consequences of the security protocol in terms of its protection coverage. Such consequences may be used to evaluate the effectiveness of a security protocols, and more so, measure the protection coverage offered by the security protocol. Hence, attack graphs generated by eCAT provide a practical way to evaluate the effectiveness of security protocols. The merits of eCAT are as follows:

1. eCAT is *automatic* and *visual*, i.e., it automatically outputs (i) attacks enabled by corruption of data items (data poisoning) in messages, data sources, and service logic; (ii) protection coverage offered by the security protocol; and (iii) network hot-spots.

2. eCAT is *interactive* and allows users to indicate invalid pre-conditions necessary for attacks when the security protocol is applied, thereby allowing for evaluating the effectiveness of security protocols.

3. eCAT is *practical* as it provides attack graph results that may be directly used in our CMF formulas, to quantify protection coverage of security protocols.

We validate the utility of our toolkit by evaluating MAPSec (Mobile Application Part Security), a key new security protocol introduced to combat mobile telecommunication network vulnerabilities. MAPSec is a good protocol to evaluate because (1) it is a relatively new protocol, and our results may help network operators in deciding on MAPSec; and most importantly (2) our initial study reveals that while the mobile telecommunication network suffers from numerous vulnerabilities [41], MAPSec is designed to only protect signaling messages. It follows that MAPSec can only protect a minor portion of total network vulnerabilities, and hence MAPSec alone is insufficient for protecting mobile telecommunication networks. Results from our evaluation can provide a quantified value to MAPSec’s effectiveness.

Results obtained from eCAT’s evaluation of MAPSec provide insights into the not only effectiveness of MAPSec but also the current state of mobile telecommunication
network security. Please note that this is the first comprehensive analysis of potential effectiveness of MAPSec.

The rest of the chapter is organized as follows. In Section 6.2, we present the preliminaries, in Section 6.3 we present our graph marking algorithm. In Section 6.5, we present a detailed evaluation of eCAT, and in Section 6.5.4, we detail our experiments and analysis. In Section 6.6, we detail our findings from evaluation. In Section 6.7 we present the related work, and finally in Section 6.8 we conclude with a discussion.

6.2 Preliminaries

In this section, we present the background on MAPSec and outline our approach.

6.2.1 MAPSec

MAPSec is designed to securely transport a crucial application layer protocol called Mobile Application Part (MAP). The MAP protocol (implemented on the ‘security free’ Signaling System No. 7 family of protocols) is used by the various service nodes to exchange signaling messages. MAP is primarily used for message exchange involving subscriber location management, authentication, call handling, handovers (managing calls while moving), supplementary services (SS), short message (SM) services, etc. MAP messages are sent in the clear and may be used to carry a variety of data items some of which are confidential. To protect such confidential data items in MAP messages, MAPsec was introduced [15, 11, 4, 10]. The main thrust of MAPSec is to provide message data and origin authentication to MAP messages.

We illustrate the MAP message protocol using the call delivery service presented in Chapter 2. As mentioned earlier, the call delivery service is a basic service used to deliver incoming calls to any subscriber with a mobile device regardless of their location. In the call delivery service, the signaling messages SRI, SRI_ACK, PRN, and PRN_ACK are MAP messages (refer Fig. 6.1). In the following, we present the threat model of MAPSec.
Threat Model of MAPSEC: Consider the MAP messages presented in the call delivery service. If an adversary gains access to cables connecting to the VLR, using readily available probes or protocol analyzers, they may capture the MAP message PRN (which is sent in the clear) and modify the data item alerting pattern as mentioned previously and output it again with the corrupt alerting pattern. If MAP messages are authorized or encrypted, occurrence of such message based attacks may be eliminated.
MAPSec is used exactly for this purpose. It provides MAP message data authentication and origin authentication thereby preventing MAP message corruption and fabrication attacks. It provides a framework that allows ciphering, signing and authorization of MAP messages.

The MAPSec architecture is illustrated in Fig. 6.2. It requires a Key Administrative Center (KAC) to be maintained by every service provider. The function of the KAC is to negotiate MAPSec security associations on a network basis. The security association comprises of keys, algorithms, protection profiles and key lifetimes used to protect the MAP messages between the various service nodes. KAC’s use the Internet key exchange (IKE) protocol and Key Management Protocol (ISAKMP) framework to negotiate the security associations. The security associations are used to protect the MAP messages between the various service nodes by creating secure and ciphered MAP communication between the various service nodes. In the following we present an overview of eCAT.

6.2.2 eCAT Overview

Advanced Cellular Network Vulnerability Assessment Toolkit (aCAT) was developed to capture potential data corrupting attack actions and their potential effects in mobile telecommunication networks. aCAT can detect attack actions and their cascading effects, and presents in them in the form of attack graphs. We define attack graphs produced by aCAT as 'unmarked attack graphs' since they are produced under the assumption that all attacks can happen and can cascade. Unmarked attack graphs do not indicate special conditions or areas such as hot-spots. A fragment of such an unmarked attack graph generated by aCAT for the alerting attack discussed in Chapter 5 is shown in Fig. 6.3.

In this chapter, we extend aCAT to eCAT, with security protocol evaluation capabilities. eCAT extracts from the unmarked attack graphs (provided by aCAT), attacks eliminated by the security protocol under consideration, network hot-spots, and so on. This process of extracting information from unmarked attack graphs is called as attack graph marking.
In marked attack graphs, shading is used to denote eliminated attacks, and hence unshaded nodes denote attacks evaded by the security protocol. Such graph marking is possible by enabling the user to input the attack pre-conditions that the security protocol can eliminate. Our graph marking algorithm then computes the attack effects that are eliminated by eliminating pre-conditions. Our graph marking scheme, detailed next 6.3, is efficient in computing these attack effects in large graphs. We can then use the CMF formulas to quantify protection coverage of the security protocol. We further detail the protection coverage quantification in 6.4. In the following we detail our graph marking algorithm and use MAPSec to illustrate the same.

**Fig. 6.3.** Fragment of Unmarked Attack Graph for Alerting Attack generated by aCAT

### 6.3 Graph Marking: The Algorithm

In this section, we present the rationale behind our approach, our graph marking algorithm, and CMF.

#### 6.3.1 Graph Marking Rationale

The aim of the marking graphs is to (1) extract security protocol coverage from an unmarked attack graphs; (2) locate most vulnerable network areas or hot-spots; and
identify the other required security protocols in addition to MAPSec. This process is called *attack graph marking*.

In producing unmarked attack graphs we assume that all attack pre-conditions and their effects are possible. As deployment of some security protocols can eliminate some of the attack pre-conditions, we use *boolean probabilities* (1 and 0) to extract attack effects reduction. Using boolean probabilities enables eCAT to distinguish between existing (true or 1) and eliminated (false or 0) states (nodes in a attack graph) in an attack. By computing boolean probabilities for each node in the attack graph, eCAT can extract the attack effects that may be eliminated by deploying the security protocol, and estimate most vulnerable network areas.

**6.3.2 Graph Marking Algorithm**

In this section, we present our *graph marking algorithm* incorporated into eCAT (shown in Algorithm 1). We use our algorithm to illustrate how an unmarked graph produced by aCAT (a fragment of this unmarked attack graph for alerting attack is shown in Fig. 6.3), is marked extract MAPSec’s coverage. The corresponding marked attack graph generated by eCAT is shown in Fig. 6.4. Our algorithm is efficient in that
it only requires as input (1) the unmarked attack graph; and (2) boolean probabilities for layer 0 nodes in the attack graph.

**Input:** Users of eCAT need only to set the boolean probability for layer 0 nodes based on the security protocols (e.g. MAPSec) under evaluation. Layer 0 nodes are the pre-conditions of the attack. Pre-conditions indicate the level of adversary’s access to the network, vulnerability exploited by the adversary, and service node targeted. By indicating to eCAT if MAPSec can or cannot prevent the adversary from exploiting a certain vulnerability or gaining a certain level of network access, eCAT can compute if the attack effect can or cannot be prevented. This indication can be provided by setting the values of the pre-conditions to either 0 or 1. The number 0 indicates that the security protocols can prevent that pre-condition from existing.

For example, consider Node 5 in attack graph for the alerting attack shown in Fig. 6.4. This node occurs at layer 0 and is hence a pre-condition. This pre-condition states that the adversary can attack using level 2 access i.e. access to cables connecting service nodes. We know that MAPSec provides authentication of messages and source address checking, hence the usage of MAPSec eliminates the possibility of attack using level 2 access. Therefore, we set the boolean probability of this node to 0.

Accordingly, the number 1 indicates that the security protocols cannot prevent that pre-condition from existing. For example, consider Node 3 at layer 0 in the attack graph shown in Fig. 6.4. This pre-condition states that the adversary can attack using level 3 access i.e. direct access to service nodes. We know that MAPSec does not offer protection to data sources and service logic in the service node. Hence we set boolean probability of this node to 1.

**Working:** Our Graph Marking algorithm makes use of the boolean probabilities provided at layer 0 to compute probabilities and mark nodes at higher layers. This algorithm is particularly efficient when the unmarked attack graph is large as it is easy to visualize the coverage of the security protocol under evaluation using color coding invalid nodes. Our algorithm starts by computing probabilities for every node (indicated by line 1 in Algorithm 1) starting from layer 1 onwards up-to the maximum layer (indicated by line 2) in the attack graph.
**Input**: $G$: Unmarked Attack Graph; $P_0$: Set of probabilities for layer 0 nodes;

**Output**: $\bar{G}$: Marked Attack Graph i.e. attack graph with invalid node marking;

Let $N_G$ be the set of nodes in the unmarked attack graph $G$;

Let max_layer indicate the maximum number of layers in a attack graph;

**Procedure Graph Mark($G$):**

1. `for (∀ node$_i$ ∈ $N_G$) do`
2. `for (ℓ ← 1 to max_layer) do`
3. `if node$_i$.layer == ℓ then`
4. `Set boolean result ← 1, temp ← 0;`
5. `if node$_i$.tree_no == 1 then`
6. `for (∀ $p_j$ ∈ Parent(node$_i$)) do`
7. `result ← result $\times$ $p_j$.probability;`
8. `end for`
9. `node$_i$.probability ← result;`
10. `else`
11. `for (∀ $t_k$ ∈ node$_i$.tree_no) do`
12. `for (∀ $p_j$ ∈ Parent(node$_i$)) do`
13. `if $p_j$.tree_no == $t_k$ then`
14. `if $p_j$.probability == 0 then`
15. `result ← 0; break`
16. `else`
17. `result ← 1;`
18. `end if`
19. `end for`
20. `temp ← temp + result;`
21. `end if`
22. `end for`
23. `node$_i$.probability ← temp;`
24. `end if`
25. `end if`
26. `end for`
27. `Marking Nodes`
28. `if node$_i$.probability == 0 then`
29. `node$_i$.mark ← invalid;`
30. `else`
31. `node$_i$.mark ← valid;`
32. `end if`
33. `end if`
34. `end for`
35. `Return $\bar{G}$`

**Algorithm 8: Graph Marking**

Our graph marking algorithm starts by considering every node in layer 1. If the node belongs to a single tree (indicated by line 5) (e.g., Nodes 14, 15, 16) it means that this attack state exists due to corruption at a single network location. For example, Tree 2 (Node 14) shows attack effects due to corruption of a message ‘MAP, SRI’. Hence the
probability of single tree nodes is the boolean product of probabilities of its parent nodes (indicated by line 8 in Algorithm 1 and shown in (6.1)).

\[
Pr[\text{Single Tree Node}_i] = \prod Pr[\text{Parents}(\text{Single Tree Node}_i)] \quad (6.1)
\]

For example, parents of Node 14 are nodes at layer 0 with tree number 2, i.e. Nodes 2, 8, 5, 11. Hence the probability of Node 14 is the product of probabilities of Nodes 2, 8, 5, 11 (shown in (6.2)).

\[
Pr[\text{Node}_{14}] = Pr[\text{Node}_2] \times Pr[\text{Node}_6] \times Pr[\text{Node}_7] \times Pr[\text{Node}_{11}] = 1 \times 1 \times 0 \times 1 = 0 \quad (6.2)
\]

Accordingly, if a node belongs to multiple trees (indicated by line 12) (e.g., Node 13), it means that this attack state can exist due to corruption at multiple network locations. Hence the probability of multiple tree nodes (such as Node 13) is the boolean sum of probabilities of parent nodes in different trees (indicated by lines 13-23 in Algorithm 1 and shown in (6.3)).

\[
Pr[\text{Multi Tree Node}_i] = \sum_i \text{Parents}_{i,i} \text{[Multi Tree Node]} \quad (6.3)
\]

For example, Node 13, belongs to trees 1, 3, 4, and 7. Hence probability of Node 13, is the sum of probabilities of parent nodes in tree 1, probabilities of parent nodes in tree 3, probabilities of parent nodes in tree 4, and probabilities of parent nodes in tree 7 (shown in (6.4)).
\[ Pr[Node_{13}] = Parent_{t1} + Parent_{t3} + Parent_{t4} + Parent_{t7} \]
\[ = (Pr[Node_1] \times Pr[Node_2] \times Pr[Node_3] \times Pr[Node_4]) \]
\[ + (Pr[Node_2] \times Pr[Node_3] \times Pr[Node_4] \times Pr[Node_5]) \]
\[ + (Pr[Node_3] \times Pr[Node_4] \times Pr[Node_5] \times Pr[Node_6]) \]
\[ + (Pr[Node_4] \times Pr[Node_5] \times Pr[Node_6] \times Pr[Node_7]) \]
\[ = (1 \times 1 \times 1 \times 1) + (1 \times 1 \times 1 \times 1) + (1 \times 1 \times 1 \times 1) + (1 \times 1 \times 1 \times 1) = 1 \quad (6.4) \]

Thus this algorithm computes probabilities of all nodes layer by layer in an iterative manner. With the probabilities computed, the algorithm marks all nodes with 0 probability as invalid and other nodes as valid (indicated by lines 30-34 in Algorithm 1). The algorithm also handles cases where nodes effect two or more trees.

\textbf{Output:} The attack graph shown in Fig. 6.4 is part of the output obtained from marking the attack graph for the alerting attack. All invalid attack effects (nodes) are shown shaded whereas all valid nodes are shown unshaded. The attack graph shows 7 trees i.e., 7 effects of corruption. By using MAPSec, we can see that 3 of the 7 possible attacks can be eliminated.

\textbf{Hot-spots:} Our algorithm also marks the network \textit{hot-spots} in the attack graph. With respect to the attack graph, we define \textit{hot-spots} as pre-condition nodes i.e. layer 0 nodes, with the highest tree number count. A high tree number count in a pre-condition node indicates an increased attractiveness of the network location to adversaries. This is because by breaking into the network location indicated by the pre-condition node, the adversary has a higher likelihood to succeed, and can cause the highest amount of damage. Our marked attack graphs also show if MAPSec offers any protection to these hotspot nodes.

The number of pre-condition nodes chosen as hot-spots can be set as per user requirements. In our case we set this value to 4. In the case of the alerting attack, the pre-condition nodes indicate the network hot-spots are Nodes 3, and 9. Together
these nodes indicate the most attractive physical access, to adversary is at level 3 (from Node 1), i.e. direct access to service nodes by exploitation of data source or service logic vulnerabilities (from Node 3) particularly the MSC service node (from Node 4). By protecting these hot-spot network locations, it is possible to prevent a large number of attacks or have a huge reduction effect on network-wide vulnerabilities. For example, by protecting hot-spots in Fig. 6.4, attacks presented in trees 1, 3, 4, and 7 can be prevented. By observing the hot-spot pre-condition nodes it is evident that MAPSec does not offer protection to the hot-spot nodes.

Marking of network hot-spots is particularly useful in extremely large attack graphs because users are immediately aware of most vulnerable network parts which when protected can prevent a large number of attacks. We now present our coverage measurement formulas which uses the results obtained from the attack graph to quantify the impact of security solutions.

6.4 Measuring Effective Coverage of Security Protocols

The CMF comprises of a set of three formulas to capture the coverage of security protocols. Our triplet formula comprises of (i) Effective Coverage, to capture the average effective number of attacks eliminated by the security protocol; (ii) Deployment coverage, to capture the coverage of protocol deployments; (iii) Attack Coverage, to capture the attack coverage provided by the security protocol. In the following we detail each formula.

1. The Effective Coverage is computed as a ratio of number of attacks eliminated by the security protocol to the number of protocol deployments (shown in (6.5)). A high value of Effective Coverage indicates that the security protocol offers a higher protection for a security deployment.

\[
\text{Effective Coverage} = \frac{\# \text{ of Attacks Eliminated}}{\# \text{ of Protocol Deployments}} \tag{6.5}
\]
In the case when MAPSec is applied to the alerting attack (from Fig. 6.4), the number of attacks prevented is 3 (Trees 2, 5, 6), and the number MAPSec deployment, i.e. MAP messages protected, is 3 (MAP_SRI, MAP_PRN, Page MS). Hence, Effective Coverage when MAPSec is used to protect from alerting attack is \( \frac{3}{3} \) (shown in (6.6)).

\[
\text{Effective Coverage} = \frac{3}{3} = 1 \quad \text{[MAPSec protect alerting attack]} \quad (6.6)
\]

2. The Deployment coverage is computed as a ratio of number of protocol deployments to the number of required deployments (shown in (6.7)).

\[
\text{Deployment Coverage} = \frac{\text{# of Protocol Deployments}}{\text{Total # of Required Protocol Deployments}} \quad (6.7)
\]

In the case when MAPSec is applied to the alerting attack (from Fig. 6.4), the number of protocol deployments i.e. MAP messages protected is 3, and the total number of required deployments i.e. MAP messages requiring protection is 4 (from Table 6.1, which shows the total number of messages in each MAP service). Hence, Deployment Coverage when MAPSec is used to protect from alerting attack is \( \frac{3}{4} \) (shown in (6.8)).

\[
\text{Deployment Coverage} = \frac{3}{4} = 0.75 \quad \text{[MAPSec protect alerting attack]} \quad (6.8)
\]

3. The Attack Coverage is computed as a ratio of number of attacks eliminated to the total number of attacks in the attack graph (shown in (6.9)). The higher this value the greater is the security solution’s efficacy in eliminating a large number of attacks on the network.
Table 6.1. Total Number of Message

<table>
<thead>
<tr>
<th>MAP Service Name</th>
<th>Total # of MAP Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location Update Service</td>
<td>16</td>
</tr>
<tr>
<td>Short Message Service</td>
<td>14</td>
</tr>
<tr>
<td>Handover Service</td>
<td>8</td>
</tr>
<tr>
<td>Supplementary Service</td>
<td>14</td>
</tr>
<tr>
<td>Call Handling Service</td>
<td>4</td>
</tr>
</tbody>
</table>

\[
\text{Attack Coverage} = \frac{\text{# of Attacks Eliminated}}{\text{Total # of Attacks in the Attack Graph}} \tag{6.9}
\]

In the case when MAPSec is applied to the alerting attack (from Fig. 6.4), the number of attacks prevented is 3, and total number of attacks in the alerting attack scenario is 7. Hence, Attack Coverage when MAPSec is used to protect from alerting attack is 3/7 (shown in (6.10)).

\[
\text{Attack Coverage} = \frac{3}{7} = 0.42 \quad \text{[MAPSec protect alerting attack]} \tag{6.10}
\]

*Interpretation:* Typically a good measure of CMF is a low value of deployment coverage, a high value of effective coverage, and a high value of attack coverage. In the case when MAPSec is applied to the alerting attack, a deployment coverage of 75% offers an attack coverage of 42%, and effective coverage of 1. From this we can infer that a single deployment of MAPSec protects against a single attack. Also, a 75% deployment offers an attack coverage of 42% which illustrates the limited effectiveness of MAPSec.

As our formulas consider the effective coverage as a percentage of the network-wide vulnerabilities, they provide a value that reflects the protection coverage of the security protocol on the network-wide vulnerabilities. As our results are generated from
attack graphs, which are a product of specifications, we can intuitively argue that our CMF formula is accurate, and gives users an idea of network-wide vulnerabilities.

As MAPSec’s performance in the case of the alerting attack did not seem impressive especially with respect to coverage of attacks, hot-spots and their protection, we further evaluate MAPSec with respect to other MAP services before we conclude MAPSec’s evaluation.

6.5 Evaluating MAPSec by Marking Attack Graphs

In this section, we evaluate MAPSec extensively in conjunction with all the major MAP services. We detail our evaluation with respect to the Short Message (SM), Supplementary (SS) Service, and authentication services. In doing so, we detail how eCAT may be applied to real life scenarios to obtain insightful results.
6.5.1 Short Message Service

The SM service is used to store and forward short text messages to the subscriber. Subscribers may send short messages (called mobile originated or MO in telecom terminology), or receive short messages (called mobile terminated or MT in telecom terminology). An attack on the SM service may occur as follows. The adversary may corrupt the service center address or ‘SC_Address’ in the MAP message. As a result the short message may be sent to a rogue service center. Adversaries may benefit by capturing these text messages.

On setting the boolean probability of Node 5 to 0 (because usage of MAPSec can eliminate level 2 access), eCAT produces the resulting marked attack graph shown in Fig. 6.5. Hot-spots are identified as Nodes 1, 3, and 4. Together these nodes indicate that highest vulnerability of the network exists at level 3 access (from Node 1), i.e. direct access to service nodes by exploitation of data source or service logic vulnerabilities (from Node 3), particularly the MSC service node (from Node 4). By protecting these hot-spot network locations, attacks presented in trees 1, 2, 3, 4, 7, 8, and 9 can be eliminated. Note that MAPSec does not offer protection to these hot-spots.

**CMF for SM Attacks:** From Fig. 6.5, it can be observed that the number of attacks eliminated is 4 (Trees 5, 6, 10, 11), the number of MAPSec deployments, i.e. MAP messages protected is 3 (MAP_MO_Fwd_SM, MAP_MT_Fwd_SM, MAP_MT_Fwd_SM_ACK), the total number of attacks in the SM attack scenario is 13, and total number of required deployments i.e., MAP messages requiring protection, is 14 (from Table 6.1). Hence, Effective Coverage when MAPSec is used to protect from SM attack is 1.33, Deployment Coverage is 0.21, and Attack Coverage is 0.31. While MAPSec’s coverage is better here than the alerting attack, MAPSec neither has a greater attack coverage nor has a higher impact factor.

6.5.2 Supplementary Service

The Supplementary Service (SS) comprises of additional services that augment the basic service. These include call forwarding, call holding, barring of incoming calls,
Fig. 6.6. Fragment of Marked Attack Graph for SS Attacks

barring of outgoing calls, call line identification, and so on. The MAP protocol is extensively used by the SS to register, erase, activate, deactivate and respond to subscriber’s queries concerning these services. The SS is denoted by a data item called supplementary service code (‘SS_code’) which denotes the type of supplementary service being registered, erased, activated, or deactivated. For example, barring of incoming calls is denoted by ‘SS_code’ of value 351.

By corrupting this ‘SS_code’, it is possible to register a subscriber for a different service i.e., if a subscriber wants to register for call forwarding, they could be registered for barring incoming calls. Using eCAT to detect all possible effects of ‘SS_code’ corruption at various network locations, we have constructed an attack graph (a fragment of which is shown in Fig. 6.6). This attack graph shows effects of corruption of ‘SS_code’ at possible locations.

On setting the boolean probability of Node 14 to 0 (because usage of MAPSec can eliminate level 2 access), eCAT produces the marked attack graph shown in Fig. 6.6. Hot-spots are identified as Nodes 7, 8, and 9. Together these nodes indicate that highest vulnerability of the network exists at level 3 access (from Node 9), i.e. direct access to service nodes by exploitation of data source or service logic vulnerabilities (from Node 7), particularly the MSC service node (from Node 8). By protecting these hot-spot network
locations, attacks presented in trees 1, 2, 3, 5, 6, 7, 8, 9 and 10 can be prevented. Once again, note that MAPSec does not offer protection to these hot-spots.

**CMF for SS Attacks:** From Fig. 6.6, it can be observed that MAPSec protects 3 MAP messages ‘MAP_deactivate_SS’ arriving at VLR, ‘MAP_deactivate_SS’ arriving at HLR, and ‘MAP_Register_SS’. Also, a typical supplementary service comprises of 14 messages (obtained from Table 6.1). Hence, Deployment Coverage when MAPSec is used to protect supplementary service is 0.21. We can also observe from Fig. 6.7, that MAPSec prevents 3 out of 14 attacks. Therefore, Attack Coverage is 0.21, and Impact factor is 1. Hence for a deployment coverage of 21%, MAPSec has an attack reduction of 21% and an impact factor of 1.

### 6.5.3 Authentication Service

The MAP protocol carries authentication material as clear text in its messages. This material is used to authenticate subscribers and allows them to access location update, handover, short message services, and so on. The authentication material is stored by the HLR in the form of a triplet array of random values (RAND), signature of random value (SRES), and ciphering keys (K_c). When a subscriber registers or moves to a new location, the HLR provides the corresponding VLR with this triplet array using MAP messages. The VLR picks a random index for RAND and passes this RAND value to the subscriber’s mobile device. The mobile device then uses the RAND value to compute the SRES and K_c. The index value of the cipher key K_c is called Ciphering Key Sequence Number or ‘CKSN’.

If the RAND value is corrupt, the subscriber’s mobile will compute an incorrect SRES which it uses to authenticate itself to the VLR. The VLR finds the SRES to be incorrect and refuses service to the subscriber. The subscriber and the network communicate over the air interface by encrypting their communications using one of the cipher keys K_c. Also, during lengthy network connections the network may re-negotiate the keys. To ensure that the subscriber and the network are using the same cipher keys K_c, the CKSN is sent using MAP messages. If the CKSN is incorrect, the subscriber
cannot be understood and hence the communication is ineffective. Thus, attacks on authentication material can effect all the services.

Fig. 6.7. Fragment of Marked Attack Graph for Authentication Material Attacks

We use eCAT to detect possible attacks due to corruption of ‘RAND’ and ‘CKSN’ corruption at various network locations. A fragment of the attack graph shown in Fig. 6.7. On setting the boolean probability of Node 7 to 0 (because usage of MAPSec can eliminate level 2 access), eCAT produces the resulting marked attack graph shown in Fig. 6.7. Hot-spots are identified as Nodes 4, 5, and 6. Together these nodes indicate that highest vulnerability of the network exists at level 3 access (from Node 4), i.e. direct access to service nodes by exploitation of data source or service logic vulnerabilities (from Node 6), particularly the VLR service node (from Node 5). By protecting these hot-spot network locations, attacks presented in trees 1, 2, 3, 5, 6 and 7 can be prevented. Note, once again that MAPSec does not offer protection to these hot-spots.

**CMF for Authentication Material Attacks:** From Fig. 6.7, we observe that MAPSec protects MAP messages ‘MAP_Send_Id_Response’, ‘MAP_Authenticate’, and ‘MAP_Prepard_HO’. Also, the total of all MAP messages considered in this scenario is 38 (Table 6.1), including the location update, handover and short message services. Hence, *Deployment Coverage* when MAPSec is used to protect authentication attacks
is 0.07 As MAPSec prevents 3 out of 10 attacks, therefore, Attack Coverage is 0.3, and Effective coverage is 1. Hence, although the impact factor seems to be low, a low deployment coverage of MAPSec can reduce attacks by 30%. Hence deploying MAPSec in this case may be adequate if not extremely effective.

6.5.4 Experiments and Analysis

To further confirm the coverage provided by MAPSec, we have conducted extensive experiments and analysis. We evaluated MAPSec’s performance in all the major services that could use MAPSec including - location update, short message, handover, supplementary and call handling services.

In our testing, we considered each individual MAP message in each service and found the number of attacks that may be prevented by protecting that individual MAP message using MAPSec. Our extensive test results are documented in Table 6.2A- 6.2E.

In each table, the first column is the name of the MAP message protected by MAPSec. The second column is the number of attacks that may be eliminated by protecting the corresponding MAP message with MAPSec. The third column is the total number of attacks that can occur without MAPSec’s protection; this typically corresponds to all the attack trees in the attack graph. These include attacks due to data source and service logic corruption, as well as attacks due to corruption of data items in MAP messages. This column gives us the network-wide attacks. The final three columns correspond to values of effective coverage, deployment coverage, and attack coverage for protecting the corresponding MAP message using MAPSec. The final row corresponds to the values for protecting all the MAP messages in the service.

The following are the observations from our tables.

- A major observation from our tables is that protecting a single MAP message or a selected set of MAP messages is better than protecting all the MAP messages in a service. For example, consider the supplementary service. Protecting a single message ‘MAP_DEACTIVATE_SS_ACK’ gives better values of effective coverage, deployment coverage, and attack coverage (3, 0.07, 0.375) over protecting all the MAP messages (2.85, 1, 0.29) in the supplementary service.
Table 6.2. Experimental Results

### A. Location Update Service

<table>
<thead>
<tr>
<th>MAP_Message_Name</th>
<th># Attacks Prevented</th>
<th>Total of Possible Attacks</th>
<th># Effective Coverage</th>
<th>Deployment Coverage</th>
<th>Attack Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP_UPDATE_LOCATION_AREA</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>0.06</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_UPDATE_LOCATION_AREA-ACK</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0.06</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_SEND_IDENTIFICATION</td>
<td>3</td>
<td>12</td>
<td>3</td>
<td>0.06</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_SEND_IDENTIFICATION-ACK</td>
<td>3</td>
<td>13</td>
<td>3</td>
<td>0.06</td>
<td>0.23</td>
</tr>
<tr>
<td>MAP_SET_CIPHERING_MODE</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>0.06</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_SET_CIPHERING_MODE-ACK</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.06</td>
<td>0</td>
</tr>
<tr>
<td>MAP_FORWARD_NEW_TMSI</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0.06</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_FORWARD_NEW_TMSI-ACK</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.06</td>
<td>0</td>
</tr>
<tr>
<td>MAP_UPDATE_LOCATION</td>
<td>8</td>
<td>30</td>
<td>8</td>
<td>0.06</td>
<td>0.27</td>
</tr>
<tr>
<td>MAP_UPDATE_LOCATION-ACK</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>0.06</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_CANCEL_LOCATION</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>0.06</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_CANCEL_LOCATION-ACK</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.06</td>
<td>0</td>
</tr>
<tr>
<td>MAP_INSERT_SUBSCRIBER_DATA</td>
<td>26</td>
<td>40</td>
<td>26</td>
<td>0.06</td>
<td>0.65</td>
</tr>
<tr>
<td>MAP_INSERT_SUBSCRIBER_DATA-ACK</td>
<td>6</td>
<td>18</td>
<td>6</td>
<td>0.06</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_AUTHENTICATE</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0.06</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_AUTHENTICATE-ACK</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0.06</td>
<td>0.33</td>
</tr>
<tr>
<td><strong>All Messages</strong></td>
<td><strong>60</strong></td>
<td><strong>163</strong></td>
<td><strong>3.75</strong></td>
<td><strong>1</strong></td>
<td><strong>0.37</strong></td>
</tr>
</tbody>
</table>

### B. Short Message Service

<table>
<thead>
<tr>
<th>MAP_Message_Name</th>
<th># Attacks Prevented</th>
<th>Total of Possible Attacks</th>
<th># Effective Coverage</th>
<th>Deployment Coverage</th>
<th>Attack Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP_SEND_ROUTING_INFO_FOR_SM</td>
<td>6</td>
<td>20</td>
<td>6</td>
<td>0.07</td>
<td>0.3</td>
</tr>
<tr>
<td>MAP_SEND_ROUTING_INFO_FOR_SM-ACK</td>
<td>5</td>
<td>14</td>
<td>5</td>
<td>0.07</td>
<td>0.36</td>
</tr>
<tr>
<td>MAP_MT_FORWARD_SHORT_MESSAGE</td>
<td>4</td>
<td>12</td>
<td>4</td>
<td>0.07</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_MT_FORWARD_SHORT_MESSAGE-ACK</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0.07</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_SEND_INFO_FOR_MT_SMS</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0.07</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_SEND_INFO_FOR_MT_SMS-ACK</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0.07</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_PROCESS_ACCESS_REQUEST</td>
<td>8</td>
<td>25</td>
<td>8</td>
<td>0.07</td>
<td>0.32</td>
</tr>
<tr>
<td>MAP_SEARCH_FOR_MS</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.07</td>
<td>0.5</td>
</tr>
<tr>
<td>MAP_AUTHENTICATE</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0.07</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_AUTHENTICATE_ACK</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0.07</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_SEND_INFO_FOR_MO_SMS</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.07</td>
<td>0.5</td>
</tr>
<tr>
<td>MAP_SEND_INFO_FOR_MO_SMS-ACK</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.07</td>
<td>0.5</td>
</tr>
<tr>
<td>MAP_MO_FORWARD_SHORT_MESSAGE</td>
<td>4</td>
<td>11</td>
<td>4</td>
<td>0.07</td>
<td>0.36</td>
</tr>
<tr>
<td>MAP_MO_FORWARD_SHORT_MESSAGE-ACK</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.07</td>
<td>0.5</td>
</tr>
<tr>
<td><strong>All Messages</strong></td>
<td><strong>37</strong></td>
<td><strong>111</strong></td>
<td><strong>2.64</strong></td>
<td><strong>1</strong></td>
<td><strong>0.33</strong></td>
</tr>
</tbody>
</table>
### C. Supplementary Service

<table>
<thead>
<tr>
<th>MAP_Message_Name</th>
<th># Attacks Prevented</th>
<th>Total # of Possible Attacks</th>
<th># Effective Coverage</th>
<th>Deployment Coverage</th>
<th>Attack Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP_REGISTER_SS</td>
<td>8</td>
<td>25</td>
<td>8</td>
<td>0.07</td>
<td>0.32</td>
</tr>
<tr>
<td>MAP_REGISTER_SS_ACK</td>
<td>3</td>
<td>11</td>
<td>3</td>
<td>0.07</td>
<td>0.27</td>
</tr>
<tr>
<td>MAP_ERASE_SS</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0.07</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_ERASE_SS_ACK</td>
<td>2</td>
<td>7</td>
<td>2</td>
<td>0.07</td>
<td>0.285</td>
</tr>
<tr>
<td>MAP_ACTIVATE_SS</td>
<td>3</td>
<td>10</td>
<td>3</td>
<td>0.07</td>
<td>0.3</td>
</tr>
<tr>
<td>MAP_ACTIVATE_SS_ACK</td>
<td>4</td>
<td>14</td>
<td>4</td>
<td>0.07</td>
<td>0.285</td>
</tr>
<tr>
<td>MAP_DEACTIVATE_SS</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0.07</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_DEACTIVATE_SS_ACK</td>
<td>3</td>
<td>8</td>
<td>3</td>
<td>0.07</td>
<td>0.375</td>
</tr>
<tr>
<td>MAP_INTERROGATE_SS</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>0.07</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_INTERROGATE_SS_ACK</td>
<td>7</td>
<td>20</td>
<td>7</td>
<td>0.07</td>
<td>0.35</td>
</tr>
<tr>
<td>MAP_INVOKE_SS</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>0.07</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_INVOKE_SS_ACK</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.07</td>
<td>0</td>
</tr>
<tr>
<td>MAP_GET_PASSWORD_SS</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.07</td>
<td>0</td>
</tr>
<tr>
<td>MAP_GET_PASSWORD_SS_ACK</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0.07</td>
<td>0.33</td>
</tr>
<tr>
<td><strong>All Messages</strong></td>
<td><strong>40</strong></td>
<td><strong>135</strong></td>
<td><strong>2.85</strong></td>
<td><strong>1</strong></td>
<td><strong>0.29</strong></td>
</tr>
</tbody>
</table>

### D. Call Handling Service

<table>
<thead>
<tr>
<th>MAP_Message_Name</th>
<th># Attacks Prevented</th>
<th>Total # of Possible Attacks</th>
<th># Effective Coverage</th>
<th>Deployment Coverage</th>
<th>Attack Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP_SEND_ROUTING_INFORMATION</td>
<td>23</td>
<td>67</td>
<td>23</td>
<td>0.25</td>
<td>0.34</td>
</tr>
<tr>
<td>MAP_SEND_ROUTING_INFORMATION_ACK</td>
<td>16</td>
<td>49</td>
<td>16</td>
<td>0.25</td>
<td>0.36</td>
</tr>
<tr>
<td>MAP_PROVIDE_ROAMING_NUMBER</td>
<td>18</td>
<td>50</td>
<td>18</td>
<td>0.25</td>
<td>0.36</td>
</tr>
<tr>
<td>MAP_PROVIDE_ROAMING_NUMBER_ACK</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td><strong>All Messages</strong></td>
<td><strong>59</strong></td>
<td><strong>174</strong></td>
<td><strong>14.75</strong></td>
<td><strong>1</strong></td>
<td><strong>0.34</strong></td>
</tr>
</tbody>
</table>

### E. Handover Service

<table>
<thead>
<tr>
<th>MAP_Message_Name</th>
<th># Attacks Prevented</th>
<th>Total # of Possible Attacks</th>
<th># Effective Coverage</th>
<th>Deployment Coverage</th>
<th>Attack Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP_PREPARE_HANDOVER-REQ</td>
<td>9</td>
<td>36</td>
<td>9</td>
<td>0.125</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_PREPARE_HANDOVER-RES</td>
<td>6</td>
<td>22</td>
<td>6</td>
<td>0.125</td>
<td>0.27</td>
</tr>
<tr>
<td>MAP_PROCESS_ACCESS-SIGNALLING</td>
<td>5</td>
<td>20</td>
<td>5</td>
<td>0.125</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_SEND_END_SIGNAL-REQ</td>
<td>4</td>
<td>16</td>
<td>4</td>
<td>0.125</td>
<td>0.25</td>
</tr>
<tr>
<td>MAP_SEND_END_SIGNAL-RES</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>0.125</td>
<td>0.33</td>
</tr>
<tr>
<td>MAP_SEND_HANDOVER-REPORT-REQ</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>0.125</td>
<td>0.35</td>
</tr>
<tr>
<td>MAP_SEND_HANDOVER-REPORT-RES</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td>MAP_ALLOCATE_HANDOVER-NUMBER</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td><strong>All Messages</strong></td>
<td><strong>29</strong></td>
<td><strong>111</strong></td>
<td><strong>3.62</strong></td>
<td><strong>1</strong></td>
<td><strong>0.26</strong></td>
</tr>
</tbody>
</table>
• Other observations include: (i) MAPSec has an average network-wide attack coverage of 33%, with a maximum attack coverage of 65%, and a minimum attack coverage of 0%; and (ii) the effective coverage from deploying MAPSec can be a high of 26 to a low of 0.

• Most vulnerable network areas, i.e. hot-spots corresponds to data sources and service logic, are not protected by MAPSec. Hence the other security protocols required are for protecting data sources and service logic.

In the next section, we detail further observations from our tables.

6.6 MAPSec Utility Findings and Discussions

In this section, we qualitatively discuss our findings from evaluating MAPSec. Our findings may be categorized as with respect to the network in general and with respect to MAPSec.

1. Data Sources and Service Logic: Data Sources and Service Logic are the most vulnerable network areas. This is because a corrupt data source or service logic may be used by many different services and hence, cause many varied cascading effects spawning a large number of attacks (indicated by multiple trees in attack graphs). Thus attacks that occur due to exploiting data source vulnerabilities constitute a major portion of the network-wide vulnerabilities and hence a major problem. In other words by exploiting service logic and data sources the likelihood of attack success is very high. Hence data source and service logic protection mechanisms must be deployed.

It must be noted that MAPSec protects neither service logic nor data sources, rather it protects MAP messages (level 2). Message corruption has a low spawning effect. Typically a single message corruption causes a single attack, as messages are typically used by a single service. As a result, MAPSec is a solution to a minor portion of the security problem, and only protects up to 33% (from our experiments) of attacks.

2. Lack of Protection from Cascading Effects: Another observation is that, although MAPSec can prevent MAP message attacks from occurring, MAPSec cannot
prevent a successfully launched attack from cascading. When an incoming MAP message (unprotected by MAPSec) is corrupt, the service node assumes it is correct and, hence securely transports it, thereby aiding cascading effects. For MAPSec to be truly successful every leg of the MAP message transport must be secured using MAPSec.

3. Risk Level of MAP message attacks: Finally, the risk level of attacks without MAPSec is very high. MAP messages can be very easily corrupted when they traverse third party networks enroute to their destinations. Rogue third parties can very easily target messages from a certain source, or to certain destinations. While the risk level of service logic or data source corruption may be low because service providers typically house service nodes in central offices where they have some measure of control over direct access to physical equipment, this may not be possible when messages traverse through third party networks.

In conclusion, MAPSec is 100% true to its threat model and can prevent the occurrence of all MAP message related attacks which are extremely high risk. As MAPSec’s deployment is optional, if any service provider chooses to omit its deployment, the efforts of all other providers is wasted. Hence to completely protect MAP messages, MAPSec’s deployments must be used by every service provider. However the overhead for deploying MAPSec can be high, both in terms of processing load and monetary investment.

Finally, as MAPSec can only protect 33% of attacks, it alone is insufficient to protect the network. A complete protection scheme for the network must comprise of data source and service logic protection in addition to MAPSec.

6.7 Related Work

In this section, we review the literature on various attack graph technologies and illustrate that our method is the first to conduct security protocol coverage analysis for mobile telecommunication networks. Attack graph technologies have been extensively studied by [70, 58, 69, 62, 65, 9, 55, 34, 35, 66, 73, 17, 77, 18, 60, 49, 50, 51, 37, 38]. Each of these techniques show how attacks can occur, and progress by exploiting vulnerabilities. While a lot of research has been conducted in uncovering attacks using graphs, few have
used graphs for practice oriented analysis. In the following, we summarize major attack
graph technologies that have used graphs for further analysis.

Amman et. al [62, 9], and Swiler and Phillips [70, 58, 69] have developed schemes
to search their attack graphs to find the shortest and the most likely to succeed attack
paths. Our scheme is similar to theirs, as we have also developed a scheme to finds
attacks, from our attack graph, that can be prevented using MAPSec. However, the
focus of our work is entirely different, in the sense that we want to evaluate a security
scheme by quantifying its coverage, and its effectiveness.

Jha et. al. [35, 34, 65] used model checking to generate attack graphs. They
analyzed their graphs for survivability, reliability, and minimization. In each case they
attempt to solve the ‘minimum cost and maximum benefit’ problem, for example, finding
the minimum critical set of atomic attacks that must be prevented to guarantee that
the adversary cannot reach the goal. They approach the problem by considering attack
likelihoods to be markov decision processes (MDP) and used the value iteration algorithm
to find the optimal selection policy. Although our work is similar in aim, their solution
cannot be applied to our case, as the markov process assumption is inapplicable to
mobile telecommunication networks. In detail, markov processes tend to approximate the
network to a single state, but the the mobile telecommunication network is a concurrent
distributed system, whose current state is the collective state of all it service nodes and
messages.

Noel et. al. [52] use dependency graphs to compute a set of minimum cost network
hardening measures to guarantee the safety of the system. The solution is a set of initial
conditions that can be independently disabled at minimum cost. Although the focus of
our work is similar i.e. get maximum protection, nevertheless our work is more practice
oriented in the sense that : (1) Our approach is focused on measuring the effectiveness of
security protocols in terms of the attacks it can protect; (2) We have provided a formula
to quantify the the same; and (3) We evaluate the performance of a MAPSec against
real network.
To the best of our knowledge our work is the first of its kind to analyze deployment impact of security solutions on mobile telecommunication networks, provide a CMF formulas, and most importantly evaluate MAPSec and provide insights into its utility.

6.8 Conclusion

We have evaluated MAPSec using a highly effective graph based scheme - eCAT. MAPSec’s evaluation has shown that while MAPSec provides protection from a narrow set of attacks, it does not effectively mitigate attacks with the largest potential impact, i.e. attacks caused by corrupt data sources, and service logic. In fact, we have shown that attacks that have previously occurred would not be prevented by the current versions of MAPSec. Also, for signaling messages in the mobile telecommunication network to be truly protected, all signaling messages not just MAP messages must be protected. It would be truly beneficial if MAPSec is extended to support all other signaling message protocols. In the next chapter, we present a solution to protect all types of signaling message protocols as well as detect data source, and service logic corruption.
Chapter 7

EndSec: An End-to-End Message Security Protocol

7.1 Introduction

In this chapter, we present a new security protocol called End-to-End Security (EndSec) for protecting all types of signaling messages, and detecting data source and service logic corruption.

EndSec is motivated by the results obtained from evaluating MAPSec. Our evaluation of MAPSec shows that compromised service nodes (due to corrupt data sources and service logic) have the potential to cause maximum damage in terms of the number of attack effects, and the ultimate service disruptions, in comparison to signaling messages. We also found that just protecting the MAP protocol alone is insufficient to provide signaling message security. To truly provide signaling message security all types of signaling messages must be protected.

We propose EndSec to overcome the limitations of MAPSec. The design goals of EndSec include: (1) protecting all types of wireline signaling messages; (2) providing end-to-end security for wireline signaling messages, thereby preventing compromised service nodes from corrupting signaling messages; (3) providing multi-hop security to each data item in the signaling message; and (4) detecting intermediate service nodes causing corruption due to corrupt service logic or data sources. We also analyze EndSec’s security properties, and its performance.

EndSec works by inserting cryptographic checks on data items in signaling messages. This requirement allows protection to be extended to all types of signaling messages, provides end-to-end security for signaling messages, and multi-node security to data items. It allows compromised service nodes to be traced bringing accountability previously non-existent, and enables real-time tracking of adversary activity in some cases.
It must be noted it is not easy to devise an end-to-end security protocol because:
(1) It requires great understanding of the network dependency model and propagation
model in the mobile telecommunication network; (2) The security protocol must be
generic enough to be applicable to various types of signaling message protocols; (3) To
provide end-to-end message security, security must be provided not only on the link
but also in the service nodes en-route to the destination; (4) Provision for multi-hop
data item security must provide security for each data item in the multiple intermediate
service nodes that handle these data items and must consider the multiple destinations
for each data item; and finally (5) the protocol must be able to trace corruption to its
source.

To the best of our knowledge, EndSec is the first protocol to provide end-to-end
security for signaling messages. The merits of EndSec include: (1) identifying data item
properties and network model unique to the mobile telecommunication network; (2)
provision for multi-hop security for data items in signaling messages; and (3) ability to
detect the service node causing the corruption using network knowledge provided by the
Third Generation Partnership Project (3GPP) [8] specifications.

Please note that while EndSec is motivated by the evaluation of MAPSec and illus-
trated in this chapter on the circuit switched network, it is a general solution. Hence,
similar conclusions on EndSec’s utility and performance may be drawn on the IP multi-
media subsystem.

The rest of the chapter is organized as follows. In Section 7.2, we present an
overview of EndSec, and detail the data property model of in Section 7.3. In Section
7.4, we detail our network model, and in Section 7.5 we outline the threat model used to
design EndSec. In Section 7.6, we present the assumptions made in the design of EndSec,
and in Section 7.7 we present our scheme EndSec in detail. In Section 7.8 we analyze
the security properties of our scheme, and in Section 7.9, we present a comparative
performance evaluation of EndSec. In Section 7.10, we present related work, and finally
in Section 7.11, we conclude with a discussion.
7.2 EndSec Overview

In this section, we present a brief overview of EndSec. EndSec is a protocol to provide end-to-end security for signaling messages in mobile telecommunication networks. As mentioned earlier, security of signaling messages essentially depends on security of the data items carried by the signaling message. If the integrity of the data item is compromised, the network malfunctions resulting in many detrimental effects.

Hence, the primary design goal of EndSec is to protect data item integrity and detect when a service node is compromised, and corrupts data item values. The specific design goals of EndSec may be specified as follows:

1. Protect all types of wireline signaling message protocols.

2. Protect the integrity of a data item from its source service node to its destination service node(s).

3. Detect when an intermediate service node is compromised and corrupting data items.

EndSec works by requiring every data item to be signed by their source service nodes using public key encryption. This brings accountability to every data item by allowing the source of corruption to be traced, thereby discouraging corruption, and hence the ultimate service disruption.

EndSec also requires every service node in the signal flow path to embed the PATH taken by the signal flow in each message. PATH allows to trace the instances and types of service nodes that have processed the signal flow thereby aiding in detection of collusion based mis-routing, and node impersonation attacks.

On receiving an EndSec message, an ‘EndSec Check’ is performed to detect corruption. To handle and correct corruption, we also introduce several control messages, such as, EndSec-Repair-Data, EndSec-Repair-PATH, EndSec-Alert, EndSec-Audit-Request, and EndSec-Audit-Response. In the following sections, we detail the various models that motivated the design of EndSec.
7.3 Data Property Model

In this section, we illustrate the properties of data items using the call delivery service illustrated Fig. 6.1, and detailed in Chapter 2. We include the figure here, as Fig. 7.1 for ease of reading.

Several application layer protocols are used for signaling. In the call delivery service, we focus on Mobile Application Part (MAP), ISDN User part (ISUP), and Transaction Capabilities Application Part (TCAP) protocols. The MAP messages are SRI, SRI_ACK, PRN, and PRN_ACK; the ISUP message is the IAM; and TCAP messages are SIFIC, Page MS, and Page.

When a call is placed to a mobile subscriber, the call (signaling message IAM) is sent to the nearest GMSC which is in charge of routing calls, and passing voice traffic between disparate networks (refer Fig. 7.1). As mentioned earlier, signaling messages contain data items used to indicate functions to be performed at the receiving service node. Consider the IAM signaling message shown in Fig. 7.1; it contains the ‘called number’ data item. The ‘called number’ data item denotes the mobile phone number of the subscriber receiving this call. The ‘called number’ is used by the GMSC to locate the
address of the HLR (home network) of the called party. The GMSC uses this address to send the HLR the MAP signaling message \textbf{SRI}.

The \textbf{SRI} MAP message is an intimation to the HLR of the arrival of an incoming call to a subscriber with ‘called number’ as mobile phone number. The \textbf{SRI} MAP message contains data items such as the ‘called number’, and the \textit{alerting pattern}. The \textit{alerting pattern} denotes the pattern (packet switched data, short message service or circuit switched call) that is used to alert the called mobile subscriber. The HLR uses the ‘called number’ to retrieve from its database the current location of the subscriber receiving the call. The HLR uses this subscriber location to send the VLR the MAP message \textbf{PRN}. The \textbf{PRN} MAP is a request for call routing information (called roaming number) from the VLR that is in-charge of the location where the subscriber is currently roaming. The \textbf{PRN} MAP message contains the ‘called number’, \textit{alerting pattern}, and other subscriber call profile data items.

The VLR uses the ‘called number’ to store the \textit{alerting pattern}, subscriber call profile data items, and assign the roaming number data item for routing the call. This roaming number data item is passed on to the HLR (in MAP message \textbf{PRN\_ACK}) which forwards it to the GMSC (in MAP message \textbf{SRI\_ACK}). The GMSC uses this roaming number to route the call (message \textbf{IAM}) to the MSC where the subscriber is currently roaming. On receipt of the message \textbf{IAM}, the MSC assigns the ‘called number’ resources for the call and also requests the subscriber call profile data items, and \textit{alerting pattern} for the ‘called number’ (using message \textbf{SIFIC}) from the VLR, and receives the same in the \textbf{Page MS} message. The MSC uses the \textit{alerting pattern} in the incoming call profile to determine the \textit{page type} data item. The \textit{page type} data item denotes the manner in which to alert the mobile station. This \textit{page type} data item is used to page the mobile subscriber (using message \textbf{Page}).

In summary, data items in signaling messages typically exhibit the following properties:

1. They may be used to derive other data items (via \textit{derivative dependencies}); for example \textit{alerting pattern} is used to derive \textit{page type}. 
2. Different data items have different destination service nodes; for example, the destination service node of *subscriber call profile data items* is the VLR.

3. Some data items have multiple destination service nodes; for example, the multiple destination service nodes of the ‘called number’ include the GMSC, HLR, VLR, and MSC.

### 7.4 Network Model

Our network model is comprised of a number of service nodes (as shown in Fig. 7.2) of different types that are geographically distributed, communicating using signaling messages. In Fig. 7.2, we denote the various service node types by $SN_a$, $SN_b$, $SN_c$, and $SN_d$, and signaling messages by $M_1$, $M_j$, and $M_k$. The network typically triggers a signal flow for delivering a service. We further categorize the behavior of service nodes with respect to data items in the signal flow as *originators*, *derivators*, and *mediators*; and present them in detail as follows.

![Network Model Diagram](image)

1. **Originators**: Service nodes that create data items unaided by other service nodes are defined as *originators*. For example, service node $SN_a$ is the originator with respect to data items $d_{a1}$, $d_{a2}$, $d_{a3}$, as they are created by $SN_a$ unaided by other service nodes.

2. **Derivators**: Service nodes that derive data items aided by other service nodes are defined as *derivators*. Derivators use data items that are originated or derived by other service nodes as sources in *derivation operations* (denoted by ‘Derive’ in Fig. 7.2)
to derive the value of other data items. For example, the service node SN<sub>b</sub> is the derivator with respect to data item d<sub>b1</sub>, as SN<sub>b</sub> uses the data item d<sub>a2</sub> that is originated by SN<sub>a</sub>, and received in signaling message M<sub>i</sub>, as a source to derive the data item d<sub>b1</sub>.

- Derivation operations that use data items as a source to derive other data items include database queries using the source data item as the key. For further information on these operations please refer to [38].

- Derived data items may be further passed on to other service nodes, and may be used as sources in future derivations. For example, the derived data items d<sub>b1</sub> is passed on to SN<sub>d</sub> where it is used as the source to derive d<sub>d2</sub>.

3. **Mediators**: Service nodes that just route data items (that are originated or derived elsewhere) to other service nodes using signaling messages may be defined as mediators. For example, service nodes SN<sub>b</sub> and SN<sub>c</sub> are mediators with respect to the data item d<sub>a1</sub>. Data item d<sub>a1</sub> is created by service node SN<sub>a</sub>, and routed to the service node SN<sub>d</sub> via mediator service nodes SN<sub>b</sub> and SN<sub>c</sub> using signaling messages M<sub>i</sub>, M<sub>j</sub>, and M<sub>k</sub>.

Please note that a single service node may be an originator, derivator, and mediator with respect to different data items. For example, the service node SN<sub>b</sub> is the creator with respect to d<sub>b2</sub>, derivator with respect to d<sub>b1</sub> (as d<sub>a2</sub> is used to derive d<sub>b1</sub>), and mediator with respect to d<sub>a1</sub>.

7.5 **EndSec Threat Model**

The EndSec threat model includes any active attack actions such as buffer overflow, and command injection that can cause data item corruption in (1) signaling messages; and (2) service nodes. With respect to corruption in service nodes: (i) data items may be corrupted in caches, database records, and local variables; (ii) data items may also be corrupted due to compromised service logic that computes data items or generates signaling messages containing data items. Readers interested in how these actions may be taken may refer to [41]. In our threat model, we only consider the ‘system
acceptable incorrect value corruption' where corrupt values take on system acceptable, albeit incorrect values.

We organize the EndSec threat model into various cases; each case indicating locations where data item corruption may occur. In detail, we consider corruption of data items contained in signaling messages in the (a) links between service nodes; and (b) service nodes. With respect to service nodes, we consider the cases of corruption in (i) originators; (ii) derivators; and (iii) mediators. In the following, we discuss each case in detail.

Case 1: Corruption on the link: Corruption of data items on the link can occur when an adversary compromises a link, and corrupts data item values contained in signaling messages traversing the link. The data item value that is received at the destination is not the same as the data item value sent on the link. For example, Fig. 7.3 shows that the service node SN_a sends signaling messages M_i containing data item d_{a1} into the link, to service node SN_b. However, the data item value d'_{a1} received by SN_b is not the same as the value of d_{a1} sent by SN_a.

Case 2: Corruption by originators: When a service node is compromised, and it is the originator of data item values, it may create corrupted data item values. For example, Fig. 7.4 shows a compromised originator service node SN_a, that creates corrupted data item values d_{a1}, and d_{a2}.

Case 3: Corruption by derivators: In this case the compromised derivator service node may derive corrupt data items although they use correct source data items values. For example, Fig. 7.5 shows the compromised derivator service node SN_b, which uses
correct value of $d_{a1}$ it receives from $SN_a$ to derive inaccurately, and thereby corrupt the value of data item $d_{b1}$.

**Case 4: Corruption by mediators:** A compromised mediator service node may corrupt and route data item values. For example, Fig. 7.6 shows a compromised mediator service node $SN_b$, which corrupts data item $d_{a1}$ and routes it to service node $SN_c$.

![Fig. 7.6. Case 4: Corruption by Mediators]
We also consider corruption caused as a result of collusion based mis-routing, and node impersonation.

**Case 5: Collusion based Mis-routing:** In collusion based mis-routing, a compromised service node in the signal flow path in collusion with another service node, mis-routes the message to the colluding service node. The colluding service node may also cause corruption. For example, in Fig. 7.7, we see service node $SN_b$ colluding with $SN_d$, and hence mis-routes signaling messages $M_j$ to service node $SN_d$ instead of service node $SN_c$. The corrupt service node $SN_d$ corrupts data item values $d_{a1}$.

**Case 6: Node Impersonation:** In node impersonation attacks, a compromised service node has keys stolen from another service node. The compromised node uses these stolen keys to impersonate the other service node.

Our scheme EndSec can detect corruption due to the above cases.

### 7.6 EndSec Assumptions

In this section, we detail the assumptions made in deploying EndSec. Our assumptions are based on the specifications provided by the Third Generation Partnership Project (3GPP) [8]. Specifications pre-set the working of the network, and are used by
equipment, and service operators as the basis for service implementation. The EndSec assumptions are detailed as follows.

1. The originator, derivator, and mediator service node types for each data item is known by every service node.

2. A signal flow is pre-set with respect to the order of the service node types that must process the signal flow.

3. The identity of a service node is its unique point code (24 bit value) as decided by the network.

4. While the signal flow path is fixed in advance with regard to the service node type, the exact service node instance processing the signal flow can only be computed by the previous service node instance.

5. Every service node shares different symmetric keys with every service node one hop away.

6. Every service node is aware of the public key and instances of every other service nodes. This may be obtained from a Distribution Center (DC).

7. Every service provider logs every incoming and outgoing signaling message for a given period of time. This is typical for service nodes [59].

7.7 EndSec Details

In this section, we present the detail the working of EndSec.

7.7.1 Structure of EndSec Messages

The EndSec signaling message is comprised of two major parts; the first part deals with data items, and the second part deals with the path taken by the signal flow. In the following we present each part in detail.

For every data item, the originators and derivators must generate data item signatures by creating a hash (denoted by $H$) of the data item, and signing the hash using
private keys (denoted by $K^r$) in asymmetric key cryptography. The combination of the data item and its signature created by private keys ensures data item integrity, and allows to detect link corruption and compromised service nodes.

EndSec also records the PATH taken by the signal flow, and its nested signature in the message. By PATH, we mean the sequence of service nodes through which the signal flow has passed. PATH is recorded to: (1) Allow retrieval of public keys and service node types from the DC using identities of service nodes in PATH; Public keys are used during data signature verification, and service node type is required during path verification; (2) Prevent malicious compromised service nodes from mis-routing the signal flow, and node impersonation; (3) Enable real-time auditing of the signal flow.

Every service node in the signal flow records the identity of the next service node in the path by appending it to the path recorded so far. We call this ‘Nesting’, and hence the name Nested path. Nesting the path prevents a compromised service node from causing data item corruption and omitting its identity from the path. As the current service node records the identity of the next service node, even if the current service node is compromised and incorrectly records the identity of the next service node, the next service node can detect this error, and demand a correction of the path.

### 7.7.2 EndSec Message Construction

In this section, we present the various mechanics involved in the creation of the EndSec message using the sample signal flow shown in Fig. 7.8. We assume that the Service Node SN$_a$, shown in Fig. 7.8, is the first service node in the signal flow path. Service Node SN$_a$ initiates the signal flow by sending message $M_i$ to SN$_b$. SN$_a$ creates data items $d_{a1}$ and $d_{a2}$, and uses its private keys ($K^r_A$) to create data item signatures ($K^r_A(H(d_{a1}))$, and $K^r_A(H(d_{a2}))$). These signatures aid in detecting corrupt data items.

Service node SN$_a$ also starts a record of the path by assigning a unique identifier (denoted as ‘Id’ in Fig. 7.8) for the signal flow. This unique signal flow identifier aids in tracking the signal flow, repairing, and auditing during corruption checking. Service node SN$_a$ appends its own identity (denoted as ‘A’ in Fig. 7.8) to the assigned signal flow identifier, and also appends the identity of the next service node (denoted
Fig. 7.8. Sample EndSec Message Construction

as ‘B’ in Fig. 7.8) in the signal flow path. Service node SN_a also creates a signature of this triplet of signal flow identifier, its own identity, and next service node identity (K^{r_A} \{ H(Id \| A \| B) \}).

Service node SN_a also uses symmetric keys (denoted as K_{AB} in Fig. 7.8) shared with SN_b, the next service node in signal flow path to encrypt the quartet of data items, data item signatures, PATH, and PATH signatures. Usage of symmetric keys provides protection to the message on the link. The Endsec message message M_i is now ready to be sent to service node SN_b.

On receiving message M_i, service node SN_b performs the EndSec Corruption Check (detailed later) followed by which it processes the message by processing data items and PATH. Data items are processed by copying data items and corresponding signatures, for which the service node is a mediator, into the outgoing signaling message. For example, service node SN_b is the mediator to data item d_{a1}, and hence copies d_{a1}, and its signature to signaling message M_j (next message in the signal flow). Service node SN_b also creates some new data items (e.g. d_{b1}), derives others (e.g. d_{b2}), signs, and adds them to the outgoing signaling message M_j as shown in Fig. 7.8.

PATH is updated by appending the identity of the next service node in the PATH to the received PATH. Hence, service node SN_b updates the PATH by adding SN_c’s identifier to the PATH. The new updated path is (Id \| A \| B \| C). PATH signatures are also updated by
appending a hash of the the next service node identity to the received PATH signature, and signing the updated PATH signature with its own private keys. Service node SN_b also updates the PATH signatures, by appending a hash of the the SN_c’s identity to the received path signatures and signing the updated PATH with its private key $K^r_b$, so the new updated path signature is $K^r_B (K^r_A \ [ H(Id | A | B) ]\ | H(C))$. This way the updated PATH and signature ‘nests’ the received PATH and signature. Signaling message $M_j$ is ready to be sent when the quartet of message components data item, PATH, and their respective signatures are encrypted using symmetric keys.

In summary, every service node in the signal flow path handles the data items and the appends identity of the next service node to the PATH in a similar manner. In the next section, we detail how corruption is detected during the EndSec Corruption Check.

### 7.7.3 EndSec Corruption Check

The *EndSec corruption check* is a test that is performed on a received message to check for corruption. This check is performed when a message is received, and before it is processed. It is used to detect corruption on the link, corruption caused by a mediator, derivator, and originator, collusion based mis-routing, and node impersonation attacks. In the following we outline the various phases in the *EndSec corruption check* followed by which we detail each phase in detail. The *EndSec corruption check* involves the following five phases:

1. In the *signature verify phase*, corruption on the link is detected by verifying data items, PATH, and their respective signatures.

2. In the *originator and derivator check phase*, corruption by the mediator is detected by checking the service node type, and position of originators and derivators in the signal flow.

3. In the *path check phase*, collusion based mis-routing, and node impersonation attacks are detected by checking the PATH.

4. In the *auto-audit phase*, we attempt to detect corruption by the derivator, and originator by automatic backtracking.
(5) In the offline audit phase, an audit is triggered manually to detect originator and derivator corruption.

1. **Signature Verify Phase:** In this phase, signatures of data items and PATH are verified. If the verification of either data items or PATH fails, this implies that corruption has occurred as a result of a compromised link. To rectify the situation and obtain the correct values, an EndSec-Repair message is sent to the appropriate originator or derivator service nodes with the respective signal flow identifier (Id).

2. **Originator and Derivator Check Phase:** In this phase, we detect mediator corruption by checking if the type and type position of originator and derivator service nodes corresponds to the 3GPP [8] set type and type position values. Service node type position is recorded in the PATH, and service node types may be retrieved from the DC using the identities found in the PATH. The comparison with 3GPP values is done using a pre-set table such as presented in Fig. 7.9.

   Fig. 7.9 shows a service node of type A, SN ax, as originator of data items d ai, d aj, sent in message M i to SN b, with their respective signatures. The second service node in the signal flow is the compromised SN b. SN b is mediator to data item d ai, however it is compromised, and hence corrupts the data item d ai, replaces its signature, and passes this in message M j to service node SN ay. SN ay is mediator to d aj. It is also of type A and is the second of type A in the signal flow. Hence SN ay’s type position is ‘2’. SN ay is also compromised, it corrupts the data item d aj, replaces its signature, and passes this in message M k to service node SN c.

   On receiving message M k, SN c performs the originator and derivator check. The 3GPP specified originator types and type position for both d ai and d aj are ‘A’ and ‘1’ respectively. The types and type position of d ai, received in message M k is ‘B’, and ‘1’. This implies that d ai has been corrupted by the mediator node SN b. Similarly, the types and type position of d aj, received in message M k is ‘A’, and ‘2’. This implies that d aj has been corrupted by the mediator node of type ‘A’ and second in signal flow, i.e., SN ay. To rectify the situation and obtain the correct values, SN c requests the originator service node SN ax, to send the correct value of d ai, d aj using the EndSec-Repair-Data message. SN c also informs the next service node in the signal flow, of B’s and Ay’s compromised
1. \( M_i : K_{Ax, B} \{ d_{ai}, d_{aj} \} \)
2. \( d_{ai} = d_{ai}' \)
3. \( M_j : K_{B, Ay} \{ d_{ai}', d_{aj} \} \)
4. \( d_{aj} = d_{aj}' \)
5. \( M_k : K_{Ay, C} \{ d_{ai}', d_{aj}' \} \)

6. Originator/ Derivator Check

Data Items received in Message \( M_k \)

<table>
<thead>
<tr>
<th>Data Item Name</th>
<th>Originator/ Derivator Type</th>
<th>Position</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>A</td>
<td>2</td>
</tr>
</tbody>
</table>

3GPP Pre-set Originator/Derivator types for data items in Message \( M_k \)

<table>
<thead>
<tr>
<th>Data Item Name</th>
<th>Originator/ Derivator Type</th>
<th>Position</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
<td>1</td>
</tr>
</tbody>
</table>

Mis-match: Mediator Corruption

7. EndSec-Repair-Data (Id, \( d_{ai}, d_{aj} \))
8. EndSec-Alert (Id, B, Ay)

Fig. 7.9. Originator/Derivator Check Phase

status by sending the EndSec-Alert message. The EndSec-Alert message is forwarded to every service node on the signal flow path.

3. Path Check Phase: In this phase, the PATH is checked to detect collusion based mis-routing, and node impersonation attacks. We check the PATH to ensure that: (1) the service node type sequence in PATH matches the signal flow sequence specified by the 3GPP to detect mis-routing due to collusions; and (2) data items in the message are only signed only by service nodes in the PATH to detect collusion attacks where mis-routing is not reflected in PATH.

The first path check detects mis-routing by matching the service node type sequence through which the signal flow has passed with the service node type sequence specified by the 3GPP. If there is a mis-match in the service node type sequence, it means that one of the service nodes has been compromised and might be in collusion.
with the mis-routed service node. In such a case, the compromised status of the service
node and its colluding service node is informed to the next service node in the signal
flow by sending the EndSec-Alert message. The EndSec-Alert message is forwarded to
every next service node the signal flow path till the last service node in the signal flow
path.

Consider the sample case, a service $S$, whose route with respect to the service
node type has been set by the 3GPP to be $ABCADE$. Suppose a message in service $S$,
arrives at a service node of type E, with $ABFADE$ as the sequence of service node types
through which the signal flow has passed. When the sequence of service node types in the
$PATH$ ($ABFADE$) is matched with the 3GPP set value ($ABCADE$) there is a mis-match.
The mis-match shows that B has caused a mis-routing. Instead of routing the message
to service node of type C, the message has been sent to service node of type F. Hence,
it means that the service node B is compromised and it is possible that B is colluding
with F. B’s compromised status and F’s occurrence is informed to every other service
node in signal flow path using the EndSec-Alert message.

The second path check detects collusion based mis-routing, when the mis-routing
is not reflected on the $PATH$. $PATH$ is accurate, and corresponds to the specified 3GPP.
In such a case, data items may be signed by service nodes whose identity is not present
in the $PATH$. Such data items are discarded, and requests are sent to the appropriate
service nodes to provide the right value. The EndSec-Repair-Data message is used for
this purpose. It contains the unique signal flow identifier and the name of the data item
that is to be replaced.

The third possible case of $PATH$ corruption is node impersonation where the $PATH$ is
corrupt such that the $PATH$ appears accurate, signatures correspond to the $PATH$, but one
of the service nodes in the $PATH$ is compromised and has stolen keys. These keys are used
to create appropriate signatures for data items and $PATH$ corrupt by the service node. It
must be noted that the usage of nested $PATH$ signatures makes the successful execution of
node impersonation attacks very difficult. In the following, we present illustrative cases
of node impersonation attacks, and why the attacks are difficult to execute when nested
$PATH$ signatures are used.
In Fig. 7.10, we present an illustrative example of node impersonation. Fig. 7.10 shows compromised service node SN\textsubscript{b}, second in the signal flow path, with keys stolen from a service node of same type as the first service node SN\textsubscript{a} in the signal flow path (i.e. of type A but of name A*). Service node SN\textsubscript{b} cleverly corrupts the data item d\textsubscript{a1} to d\textsubscript{a1}', and replaces data and path signatures of SN\textsubscript{a} using keys stolen from A*. As A* conforms to the expected type A, PATH, and signatures appear accurate and corruption is not detected.

\begin{tabular}{|c|c|c|}
\hline
1. M\textsubscript{i} : & \{d\textsubscript{a1}, \text{K} \text{r}_A[H(d\textsubscript{a1})], \text{Id A B}, \text{K} \text{r}_A[H(\text{Id A B})]\} \text{KAB} & \\
\hline
2. Replace A with A* in Signatures & & \\
\hline
3. M\textsubscript{j} : & \{d\textsubscript{a1}', d\textsubscript{b1}, \text{K} \text{r}_A[H(d\textsubscript{a1}')], \text{K} \text{r}_B[H(d\textsubscript{b1})], \text{Id A* B C}, \text{K} \text{r}_B[H(\text{Id A* B C})]\} \text{KBC} & \\
\hline
\end{tabular}

Fig. 7.10. Node Impersonation not eliminated by Nested Path

While node impersonation by service node second in the signal flow path cannot be eliminated by nested PATH signatures, node impersonation by service nodes third or higher in signal flow path can be very difficult. We illustrate such a case of node impersonation by service nodes third in signal flow path in Fig. 7.11.

Consider the service node SN\textsubscript{c} shown in Fig. 7.11. It is the third in the signal flow path, and contains keys stolen from a service node that is of the same type as the second service node in the signal flow path (i.e. of type B but of name B*). To replace service node SN\textsubscript{b}'s data item signatures, and identity in nested PATH signatures, SN\textsubscript{c} must have keys from the first service node (i.e. A) or any other service node of the same type. This is because SN\textsubscript{a} has signed B’s identity in the nested PATH signatures, and to completely replace SN\textsubscript{b}'s identity, keys of A or type A are required. Hence, for node impersonation by third service node in the signal flow path to be truly successful two different key types are required.
Thus, for node impersonation by \( n \)th service node in the signal flow path to be truly successful, it requires \((n-1)\) stolen keys of ‘k’ number of different types corresponding to signal flow; where \( k \leq (n-1)\). Therefore, further along the the service node is in the signal flow path, the more number of keys need to be stolen and node impersonation becomes more complicated.

4. **Auto-audit phase:** This phase is triggered automatically due to unexpected errors that result in incomplete service. When an unexpected error occurs, and results in incomplete service, the service node receiving such an error may automatically trigger an audit. In this audit, a EndSec-Audit-Request message with signal flow identity as parameter is sent to every service node in the signal flow path. When a service node receives a EndSec-Audit-Request message, it must look into its log of messages, and retrieve the incoming and outgoing messages corresponding to the requested signal flow identity. The resultant messages are then be sent to the audit requester service node using EndSec-Audit-Response message. When all the messages corresponding to the signal flow identity are received, they may be sent to the network expert for identification of the compromised originator and derivator.

However, it must be noted that auto-audit may trigger some false positives, and may also not detect some originator and derivator attacks. Some originator and derivator corruption can cause the network to work without errors, albeit incorrectly from the
network point of view. For example, a subscriber’s phone calls may be forwarded to a E-911 center. Such errors cannot be detected by auto-audit. Likewise due to malfunctioning of the network, some errors may occur unexpectedly, and such errors may trigger an auto-audit.

5. **Offline audit phase:** The offline audit phase is developed to detect originator and derivator corruption that may not be detected by auto-audit. Such an audit may be triggered by network operators in response to subscriber complaints, or other reasons. This phase works similar to the auto-audit phase, and only differs in the triggering mechanism.

### 7.7.4 EndSec Modifications

In the interest of maintaining the privacy of service node identities we present two modifications of our EndSec scheme called the *Gateway EndSec*, and the *Hybrid EndSec*.

**Gateway Method:** In this scheme, we assume that messages within the boundaries of a network are safe (due to adequate supervision by network operator or other mechanisms) whereas signaling messages that cross network boundaries are vulnerable. In order to protect messages outside their network, a service provider will require securing messages that cross network boundaries. Hence, only messages that cross network boundaries are secured by EndSec i.e., encrypting data items, signatures, and *PATH*. In this case, only gateway service nodes at the edge of the network share public keys with other network gateway nodes.

**Hybrid Method:** In this scheme, EndSec is deployed at two levels, a level for inside network messages, and another for cross network messages. Within network boundaries, EndSec is deployed by using keys from service nodes inside the network boundaries, while crossing network signaling messages are protected by EndSec using keys from gateway nodes. From the security point of view, this scheme is very similar to the basic scheme. However from the privacy point of view the identities of service nodes within the network are not revealed to the outside network.

In the following section, we analyze the performance of EndSec and its modifications.
7.8 Security Analysis

In this section, we justify that the security goals of EndSec have been achieved.

**Goal 1: Protect all types of wireline signaling messages**

Typically, if a security protocol must be applicable to a wide variety of wireline signaling message protocols, it might be so generic that it would require major overhaul of the protocol stack at both the lower layers, as well as, the application layer. The advantage of using EndSec is that it does not require any changes to the protocol stack at the lower layers. EndSec only requires minor changes at the application layer. With Endsec, the application layer changes are at the payload to accommodate data item signatures, path, and path signatures. Likewise, to accommodate the symmetric key based encryption to prevent link corruption, minor changes must be made to the packet header.

**Goal 2: Provide end-to-end security for all wireline signaling messages**

To cause a cascading attack with unsecured signaling messages, the adversary must corrupt the data items contained in these signaling messages. Without end-to-end security of these signaling messages, adversaries can corrupt these data items either on the link between service nodes, or in the intermediate service nodes. With Endsec, corruption on the link is more so difficult due to the usage of the symmetric keys. Even if corruption was to occur, it would be detected by during the ‘signature verify phase’ of the EndSec Corruption’ Check where signatures and data items are verified.

Also, while corruption in intermediate service nodes can occur with EndSec, it can also be detected in the Originator and Derivator Check Phase of the EndSec Corruption Check due to the usage of data items signatures, PATH, and PATH signatures. The originator/derivator service node type is known from the 3GPP, and the exact originator/derivator service node instance and type may be obtained from the PATH. By comparing these values it is possible to detect if a data item has been corrupt by an intermediate service nodes. Thus EndSec provides security and detects corruption not only on the link, but also on intermediate service nodes.
**Goal 3: Provision for multi-hop security for every data item in signaling messages**

When a data item is used in a signaling message, it could be passed on to other service nodes in signaling messages or used to derive other data items. Without multi-hop security for every data item in the signaling message, it is possible to corrupt data items in links or intermediate service nodes and pass them on to other service nodes thereby cascading the corruption.

By using EndSec, it is possible to detect corruption both on the link and in intermediate service nodes (as justified in Goal 2). EndSec also prevents corruption from cascading by defining several control messages that aid in rectifying corruption. Thus providing multi-hop security for individual data items.

**Goal 4: Identify the intermediate service node causing corruption**

When a service node creates data items, it also creates signatures for these data items. When an intermediate service node corrupts data items, it must also replace the corresponding data item signature or the corruption is obvious. In signing the corrupt data item its identity is automatically recorded. When corruption is detected in the EndSec Corruption Check, this identity recorded in the corrupt data item signature automatically points to the service node that caused the corruption.

If the compromised service node has stolen keys, it would use these stolen keys to sign data items and corrupt the PATH signatures. Depending on its position in the path, the compromised service node may or may not be able to do it. If it is unable to corrupt the PATH signatures, the corrupt data item is automatically discarded thereby making it unable to cascade. If it is able to corrupt the PATH signatures, such a case cannot be handled by EndSec.

### 7.9 Performance Analysis

In this section, we analyze and compare time delay and signaling load due to EndSec with other schemes.
7.9.1 Time Delay

In this section, we analyze the time delay in processing messages when the basic version of EndSec, Gateway EndSec, Hybrid EndSec, and MAPSec schemes are implemented. We describe the methodology used to derive the general equations for time delay. We also conduct a comparative time delay analysis due to each of the schemes when used in the call delivery service.

**General Equations:** In Table 7.1, we summarize the terminology used to derive the general time equations.

We start with the equation for the time taken to apply MAPSec ($\Delta T_{MAPSEC}$) (shown in (7.1), (7.2)). The time taken to apply MAPSec is time taken create each MAP message at the source service node, and and the time taken to verify each MAP message at the destination service node, in the signal flow path.

We consider protection mode 2 of MAPSec, which ensures confidentiality, integrity and authenticity. Protection mode 2 of MAPSec works by encrypting payload using symmetric keys and computing the MAC on the header and encrypted payload of original message. It follows that the time taken to create the message at the source is the time taken to encrypt with symmetric keys ($t_{Es}$), and time to create the MAC on the message ($t_{MAC}$). The time taken to verify the MAPSec message at the receiver is the time to compute the MAC on the message ($t_{MAC}$), and the time to decrypt the message with symmetric keys ($t_{Ds}$).

\[
\Delta T_{MAPSEC} = \sum_n (t_{Es} + t_{MAC}) + (t_{MAC} + t_{Ds}) \quad (7.1)
\]

\[
\Delta T_{MAPSEC} = \sum_n t_{Es} + 2t_{MAC} + t_{Ds} \quad (7.2)
\]

We show the general equation for the time taken to apply the basic version of EndSec ($\Delta T_{ENDSEC-B}$) to all messages in (7.3), (7.4). $\Delta T_{ENDSEC-B}$ is the sum of time taken to create the message at the source, and the time taken to verify the message at the receiver. The time taken to create the message at the source (shown in (7.3) is the
Table 7.1. Terminology in Time Delay Equations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$</td>
<td>Time taken at service nodes to process the signal flow without any security</td>
</tr>
<tr>
<td>$\Delta T_{MAPSEC}$</td>
<td>Time taken at service nodes to apply MAPSEC to all MAP messages in signal</td>
</tr>
<tr>
<td>$T_{MAPSEC}$</td>
<td>Time taken at service nodes to process the signal flow by MAPSEC. ($T +$</td>
</tr>
<tr>
<td>$\Delta T_{ENDSEC-B}$</td>
<td>Time taken at service nodes to apply basic version of ENDSEC to all messages</td>
</tr>
<tr>
<td>$T_{ENDSEC-B}$</td>
<td>Time taken at service nodes to process the signal flow by Basic ENDSEC. ($T$</td>
</tr>
<tr>
<td>$\Delta T_{ENDSEC-G}$</td>
<td>Time taken at service nodes to apply Gateway ENDSEC to all messages in</td>
</tr>
<tr>
<td>$T_{ENDSEC-G}$</td>
<td>Time taken at service nodes to process the signal flow by Gateway ENDSEC. ($T$</td>
</tr>
<tr>
<td>$\Delta T_{ENDSEC-H}$</td>
<td>Time taken at service nodes to apply Hybrid ENDSEC to all messages in</td>
</tr>
<tr>
<td>$T_{ENDSEC-H}$</td>
<td>Time taken at service nodes to process the signal flow by Hybrid ENDSEC. ($T$</td>
</tr>
<tr>
<td>$t_{E_s}$</td>
<td>Time taken to encrypt with symmetric keys.</td>
</tr>
<tr>
<td>$t_{S_a}$</td>
<td>Time taken to sign with asymmetric key cryptography.</td>
</tr>
<tr>
<td>$t_{D_s}$</td>
<td>Time taken to decrypt with symmetric keys.</td>
</tr>
<tr>
<td>$t_{V_a}$</td>
<td>Time taken to verify with asymmetric key cryptography.</td>
</tr>
<tr>
<td>$t_{MAC}$</td>
<td>Time taken to create MAC digest of the message.</td>
</tr>
<tr>
<td>$t_H$</td>
<td>Time taken to create hash of the data item.</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Position of service node in signal flow path.</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Position of boundary crossings in the signal flow path. $\beta \leq \eta$.</td>
</tr>
<tr>
<td>$m$</td>
<td>Number of MAP messages in signal flow path.</td>
</tr>
<tr>
<td>$n$</td>
<td>Number of signaling messages in signal flow path.</td>
</tr>
<tr>
<td>$F_{cross}$</td>
<td>Factor of boundary crossing (1 indicates message crosses network boundary).</td>
</tr>
</tbody>
</table>

sum of: (1) time to encrypt the message with symmetric keys ($t_{E_s}$); (2) time to secure data items, that is: (i) time to create hash of data items ($t_H$); and (ii) the time taken to sign data items with private keys in asymmetric key cryptography ($t_{S_a}$); (3) time to sign the nested PATH, that is: (i) time to create hash of path ($t_H$); and (ii) the time taken to sign path with asymmetric key cryptography ($t_{S_a}$).
\[
\Delta T_{\text{ENDSEC-B}} = \sum_n t_{E_s} + (t_H + t_{S_a}) + (t_H + t_{S_a}) + t_{D_s} + (t_H + t_{V_a}) + (\eta - 1)(t_H + t_{V_a})
\]  
(7.3)

\[
\Delta T_{\text{ENDSEC-B}} = \sum_n t_{E_s} + (\eta + 2)t_H + 2t_{S_a} + \eta t_{V_a} + t_{D_s}
\]  
(7.4)

The time to verify the basic version of EndSec message at the receiver (shown in (7.3)) is the sum of: (1) the time to decrypt the message with symmetric keys \(t_{D_s}\); (2) the time to verify data items, that is: (i) time to create the hash of encrypted data items \(t_H\); and (ii) time to verify data items with private keys in asymmetric key cryptography \(t_{V_a}\); and (3) time to verify the nested PATH signature depends on the position \(\eta\) of the service node in the signal flow path as the \(\eta^{th}\) node in the signal flow path needs to verify ‘\(\eta - 1\)’ nesting in the PATH. It follows that the time to verify PATH is: (i) time to create multiple hash’s for each nesting of PATH \(((\eta - 1)t_H)\); and (ii) time to verify the nested PATH which is the time to decrypt each nesting of the path using private keys in asymmetric key cryptography \(((\eta - 1)t_{V_a})\).

The general equation for the time taken to apply Hybrid EndSec \((\Delta T_{\text{ENDSEC-H}})\) to all messages is shown in (7.5). Delay due to Hybrid EndSec is similar to the basic version of EndSec with respect to signing and verifying data items, and nested PATH signatures. However, delay due to Hybrid EndSec differs with respect to the nested PATH verification, as nesting levels differ for cross network and inner network messages. Consider the case of cross network messages indicated by \(F_{\text{cross}}\) (the probability that the signaling message crosses network boundaries) in (7.5), the PATH is contains identities of gateway nodes through which the message has passed. Hence, the nested PATH signature verification depends on the position \(\beta\) of the gateway node in the signal flow path, and is indicated by \((\beta - 1)(t_H + t_{V_a})\). Likewise when the signaling message is within the network boundary, the nested PATH signature verification depends on the position \(\beta\) of the inner network service node in the signal flow path indicated by \((\eta - 1)(t_H + t_{V_a})\).
\[
\Delta T_{ENDSEC-H} = \sum_n t_E + t_D + 3t_H + 2t_S + t_V + F_{cross} (\beta - 1)(t_H + t_V) + (1 - F_{cross})(\eta - 1)(t_H + t_V)
\]  
\[\text{(7.5)}\]

The Gateway EndSec scheme is only deployed when a signaling message crosses network boundaries. Signaling messages cross network boundaries based on subscriber profile, location of caller, and so on. Hence time taken to apply Gateway EndSec depends on the probability that the signaling message crosses network boundaries \(F_{cross}\) and time taken to apply Basic EndSec (shown in (7.6)).

\[
\Delta T_{ENDSEC-G} = F_{cross} (\Delta T_{ENDSEC-B})
\]  
\[\text{(7.6)}\]

**Comparative Time Delay Analysis in the Call delivery Service:** We use the general equations for the delay presented previously to conduct a comparative analysis of time delay in the call delivery service. We found values for each of the time parameters in the literature (as shown in Table 7.2), and compare time delay performance of each scheme. While readers may argue about the accuracy of time delay parameters, we would like to point out that these values are not used for computing absolute values but to illustrate the relative different in each of the schemes.

We compare the time delay due to all the schemes by studying the effect of signaling messages crossing network boundaries. To study the effect of signaling messages crossing network boundaries we consider three boundary crossing scenarios in the call delivery service. We present each case as follows.

**Case A:** In this case, signaling messages do not cross network boundaries. Here the caller and called subscriber belong to the same network, and the called subscriber roams in its home network. Hence, signaling messages do not cross network boundaries.
Table 7.2. Input Values for Time Delay Equations

A. Time

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>(T)</td>
<td>3600ms</td>
<td>[26]</td>
</tr>
<tr>
<td>(t_{E_s})</td>
<td>0.48ms</td>
<td>Using Rijndael symmetric block cipher of 128 bit key and 128 bit block [24]</td>
</tr>
<tr>
<td>(t_{D_s})</td>
<td>0.58ms</td>
<td>[24]</td>
</tr>
<tr>
<td>(t_{S_a})</td>
<td>0.07ms</td>
<td>Consider 1024 bit RSA, time unaffected by input size. [54]</td>
</tr>
<tr>
<td>(t_{V_a})</td>
<td>1.52ms</td>
<td>Consider 1024 bit RSA, time unaffected by input size. [54]</td>
</tr>
<tr>
<td>(t_{MAC})</td>
<td>1.14 µsec</td>
<td>Consider CBC-MAC Rijndael with 128 bit key per 128 bit block. [19]</td>
</tr>
</tbody>
</table>

B. Time to compute SHA-1 hash of various lengths [63]

<table>
<thead>
<tr>
<th>Input Size (Bytes)</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0153</td>
</tr>
<tr>
<td>16</td>
<td>0.0156</td>
</tr>
<tr>
<td>32</td>
<td>0.0158</td>
</tr>
<tr>
<td>48</td>
<td>0.0160</td>
</tr>
<tr>
<td>64</td>
<td>0.0253</td>
</tr>
<tr>
<td>80</td>
<td>0.0263</td>
</tr>
<tr>
<td>128</td>
<td>0.0360</td>
</tr>
<tr>
<td>256</td>
<td>0.0573</td>
</tr>
<tr>
<td>512</td>
<td>0.100</td>
</tr>
<tr>
<td>1024</td>
<td>0.186</td>
</tr>
<tr>
<td>2048</td>
<td>0.357</td>
</tr>
<tr>
<td>16384</td>
<td>2.7600</td>
</tr>
<tr>
<td>65536</td>
<td>11.040</td>
</tr>
</tbody>
</table>

Case B: In this case, some signaling messages cross network boundaries. Here the caller and called subscriber belong to different networks, and the called subscriber roams in its home network. Hence, some signaling messages (such as, IAM, SRI, and SRI-ACK) cross network boundaries.

Case C: In this case, most of signaling messages cross network boundaries. Here the caller and called subscriber belong to different networks, and the called subscriber roams in its home network. Hence, all signaling messages in call delivery service, except SIFIC, and Page MS cross network boundaries.

We computed the values for time delay in each of the schemes, and present our results in Fig. 7.12. Our results show that there is a very minor difference in delay between the no security, and MAPSec schemes. Also, there is a big difference in time delay between EndSec schemes, and MAPSec and no security schemes. This big difference
is mainly due to the verification of nested PATH signatures in the EndSec scheme. For better performance in time by EndSec, the gateway EndSec may be deployed. Gateway EndSec fares better than EndSec in the three boundary crossing cases as it is deployed only between network boundaries.

7.9.2 Signaling Load

In this section, we analyze and compare the signaling load due to MAPSec, EndSec and its enhancements when used in the call delivery service. In the following, we describe the methodology used to derive the general equations for signaling load and signaling load in the call delivery service.

**General Equations:** In Table 7.3, we summarize the terminology used to derive the general signaling load equations. The signaling load generated by processing a signal flow without any security protocols ($L$) (shown in (7.7)) is the length of all messages in signal flow. Every message is comprised of overhead (to indicate message type, and so on) and data items. Hence, the length of all messages in signal flow is the sum of overheads ($M_o(t)$) for all the messages and their respective data items ($\sum_i s(d_i)$).
Table 7.3. Terminology in Signaling Load Equations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>Signaling load to processing a service without any security protocols.</td>
</tr>
<tr>
<td>$L_{MAPSEC}$</td>
<td>Signaling load due to processing a service using MAPSec.</td>
</tr>
<tr>
<td>$L_{ENDSEC-B}$</td>
<td>Signaling load due processing a service using the basic version ENDSEC.</td>
</tr>
<tr>
<td>$L_{ENDSEC-G}$</td>
<td>Signaling load due processing a service using Gateway ENDSEC.</td>
</tr>
<tr>
<td>$L_{ENDSEC-H}$</td>
<td>Signaling load due processing a service using Hybrid ENDSEC.</td>
</tr>
<tr>
<td>t</td>
<td>Types of signaling messages in the signal flow.</td>
</tr>
<tr>
<td>$M_o(t)$</td>
<td>Message overhead is a function of signaling message types (t).</td>
</tr>
<tr>
<td>$s(d_i)$</td>
<td>Size of data item $i$ in bytes.</td>
</tr>
<tr>
<td>i</td>
<td>Number of data items in a signaling message</td>
</tr>
<tr>
<td>m</td>
<td>Number of MAP messages in signal flow path.</td>
</tr>
<tr>
<td>n</td>
<td>Number of signaling messages in signal flow path.</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>Key length of asymmetric key cryptography scheme</td>
</tr>
<tr>
<td>h</td>
<td>Size of hash output.</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Position of service node in signal flow path.</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Position of boundary crossings in the signal flow path. $\beta \leq \eta$</td>
</tr>
<tr>
<td>$F_{cross}$</td>
<td>Factor of boundary crossing (1 indicated message crosses network boundary).</td>
</tr>
</tbody>
</table>

$$L = \sum_n M_o(t) + \sum_i s(d_i) \quad (7.7)$$

Signaling load generated by processing a signal flow using MAPSec ($L_{MAPSEC}$) is the lengths of all messages in signal flow. Lengths of messages in this case, is the sum of (1) Overhead due to MAPSec i.e., ($M_o(t = MAPSEC)$) which is the protection header length (22 bytes for protection mode 2), and MAP message overhead ($M_o(t = MAP)$); (2) size of encrypted MAP payload which is the size of data items $s(d_i)$, and size of MAC of the message which is 4 bytes; and (3) length due to all other types of signaling messages, this is similar to the no security case as MAPSec only processes MAP messages (shown in (7.8)).
\[ L_{MAPSEC} = \sum_n M_o(t = MAPSEC) + \sum_i s(d_i) + 4 + \sum_{n-h} M_o(t) + \sum_i s(d_i) \quad (7.8) \]

Signaling load generated by using Basic EndSec \((L_{ENDSEC-B})\) is the lengths of all messages in signal flow. The lengths of all messages in signal flow, as shown in (7.9 and 7.10), is the sum of: (1) message overhead \((M_o(t))\); (2) size of all its data items; (3) size of the data item signature, this is data item hash rounded to the nearest multiple of public key size \((\kappa)\), hence computed as a ceiling of public key size \([h/\kappa])\); (4) path size is 9 bytes for the starting service node, and 3 bytes for each subsequent node \((9 + 3\eta)\); (5) size of nested path signature, as every service node in the \(PATH\) encrypts a portion of the \(PATH\), this is path hash usually rounded to the nearest multiple of public key size \((\eta[h/\kappa])\). In each of the EndSec schemes, we do not consider the extra length due to the private key cryptography of the scheme because the private key cryptography algorithm \((f6)\) we consider generates output with same size as input.

\[ L_{ENDSEC-B} = \sum_n (M_o(t) + \sum_i s(d_i) + i[h/\kappa] + (9 + 3\eta) + \eta[h/\kappa]) \quad (7.9) \]

\[ L_{ENDSEC-B} = \sum_n (M_o(t) + \sum_i s(d_i) + [h/\kappa](\eta + i) + (9 + 3\eta) \quad (7.10) \]

Signaling load generated by using Gateway EndSec \((L_{ENDSEC-G})\), as shown in (7.11), is similar to Basic EndSec when the message crosses network boundaries. Thus the terms are multiplied by the factor of boundary crossing when the signaling messages cross network boundaries \((F_{cross})\). When signaling messages crosses network boundaries, the \(PATH\) is encrypted by gateway nodes, and hence nested \(PATH\) size depends on the position of boundary crossings in the signal flow \(PATH\) \((\beta, \text{where } \beta \leq \eta)\). Hence, \(PATH\) and \(PATH\) signatures are multiples by position of boundary crossings \((\beta)\).

\[ L_{ENDSEC-G} = \sum_n M_o(t) + \sum_i s(d_i) + F_{cross}\{i[h/\kappa] + (9 + 3\beta) + \beta[h/\kappa]\} \quad (7.11) \]
In the Hybrid EndSec scheme, signaling load is similar to other EndSec schemes with respect to data items, as shown in (7.12). However, PATH and PATH signature size depends on the number of boundary crossings ($\beta$) when the message crosses the boundary, and when the message is within the network boundaries it depends on number of service nodes ($\eta$) is signal flow path.

$$\mathbb{L}_{ENDSEC-H} = \sum_n M_0(t) + \sum_i s(d_i) + i[h/\kappa] + F_{cross}\{(9 + 3\beta) + \beta[h/\kappa]\}$$

$$+(1 - F_{cross})\{(9 + 3\eta) + \eta[h/\kappa]\} \quad (7.12)$$

**Comparative Signaling Load Analysis in the Call delivery Service:** We use the general equations for the signaling load presented previously to conduct a comparative analysis of signaling load in the call delivery service. We found values for each of the load parameters in the specifications (as shown in Table 7.4), and compare load for each scheme. We compare the load due to all the schemes by studying the effect of signaling messages crossing network boundaries by considering the three boundary crossing scenarios in the call delivery service presented previously.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$</td>
<td>1458 per sec</td>
</tr>
<tr>
<td>$t$</td>
<td>ISUP, TCAP, MAP</td>
</tr>
<tr>
<td>$M_0(t=TCAP)$</td>
<td>32 + 2i (or) 24 + 2i</td>
</tr>
<tr>
<td>$M_0(t=ISUP)$</td>
<td>16 + 2i + 2i</td>
</tr>
<tr>
<td>$M_0(t=MAP)$</td>
<td>34 + 2i (or) 26 + 2i</td>
</tr>
<tr>
<td>$M_0(t=MAPSec)$</td>
<td>22 + $M_0(t=MAP)$</td>
</tr>
<tr>
<td>$s(d_i)$</td>
<td>Refer Tables in Appendix</td>
</tr>
<tr>
<td>$i$</td>
<td>Refer Tables in Appendix</td>
</tr>
<tr>
<td>$h$</td>
<td>20 byte SHA-1</td>
</tr>
<tr>
<td>$m$</td>
<td>4</td>
</tr>
<tr>
<td>$n$</td>
<td>8</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>1024 bit keys (Considering RSA)</td>
</tr>
</tbody>
</table>
The call delivery service, contains three types of signaling messages: Mobile Application Part (MAP), ISDN User part (ISUP), and Transaction Capabilities Application Part (TCAP). The protocol stacks for each of these signaling messages along with the overhead in terms of bytes are presented in Fig. 7.13 [28, 29, 33, 31, 32, 30]. From this we derive the message overhead ($M_o(t)$) for each type of signaling message. The overhead for a TCAP invoke is 20 bytes, at the overhead due to a TCAP response is 12 bytes with 2 additional bytes per data item.

The message overhead for ISUP, includes the byte overhead at the three lower layers, 2 additional bytes per data item, and 1 additional byte per data item. The message overhead for MAP message is 2 bytes added to the message overhead for TCAP.

There are 8 signaling messages in the call delivery service, of which 4 are MAP messages. We do not count the messages on the air interface as EndSec is only a wireline network protocol. The number of data items in the call delivery service signaling messages and size of individual data items may be obtained from [32, 3], and are detailed in Appendix C. Considering the public key algorithm to be the 1024 bit RSA, we derived signaling load values for each of the schemes, and present our results in Fig. 7.14.

Our results show that the load due to MAPSec is only a few bytes more than the ‘No Security Scheme’, this is because MAPSec’s overhead is only 22 bytes, and MAPSec is only applied to 4 of the 8 messages in the call delivery service. Our results also show that EndSec requires a much higher signaling load (of the order of 20,000 bytes) in
Fig. 7.14. Signaling Load in various schemes

comparison to the No Security Scheme. The reason being the encrypted data item, and nested PATH signatures, as the size of public key encryption is rounded to the nearest multiple of public key size which is 128 bytes in this case. In the call delivery service the number of data items is 116, and it follows that the size of data item signatures is 116 × 128.

In boundary crossing case A, none of the signaling messages cross boundaries and when Gateway EndSec is used none of the messages are protected by EndSec. Accordingly, the Gateway EndSec scheme in case A, performs exactly like the ‘No Security Scheme’. In boundary crossing case B, three messages (IAM, SRI, and SRI-ACK) cross boundaries and EndSec is only applied to these three messages when Gateway EndSec is used. As the number of data items in these three messages is 96, in-turn causing the high signaling load due to data item signatures. This accounts for the high value of signaling load due to Gateway EndSec in boundary crossing case B. Similarly in boundary crossing
case C, all but two signaling messages (SIFIC, Page MS) cross boundaries, accounting for the high signaling load.

When Hybrid EndSec is used in boundary crossing case A, the signaling load is exactly the same the base EndSec scheme. This is because none of the signaling messages cross boundaries and there is no need to deploy EndSec in two levels. In boundary crossing case B, as three messages cross boundaries EndSec is deployed in two levels, it follows that the nesting levels in the nested PATH signature is greatly reduced thereby slightly reducing in the signaling load. Likewise in boundary crossing case C, as Hybrid EndSec is deployed in 2 levels with the first 6 messages crossing boundaries and the last two signaling messages within network, the nesting levels in the nested PATH signature is only slightly reduced, therefore the signaling load is slightly less than the boundary crossing case A, yet greater than he boundary crossing case B.

In summary, EndSec is good protocol to ensure network security. From observing the graphs, we find that from the processing standpoint EndSec overhead is not very high, however from the signaling load standpoint EndSec’s overhead is very high due to the nested PATH signatures. An alternative to using the Basic EndSec is the Gateway EndSec schemes which has lower signaling overhead.

7.10 Related Work

In this section, we review the literature on (1) current state of security in mobile telecommunication networks; and (2) existing schemes that could be used in EndSec, and illustrate that our method is the first comprehensive scheme for complete end-to-end message security for mobile telecommunication networks.

The current state of security for mobile telecommunication network is that traffic on the air interface is protected to ensure integrity, confidentiality and authentication. However, the wireline network is lacking in protecting integrity, confidentiality, and authentication of all types of traffic [13, 42, 75].

The major standardized scheme for signaling message security is MAPSec [4]. MAPSec is an application layer protocol that is used to protect Mobile Application Part (MAP) protocol messages traveling between two service nodes. The MAP protocol is
used to handle the services with respect to mobility, such as, subscriber location management, authentication, call handling (explained previously), and so on. MAPSec provides message data authentication and origin authentication thereby preventing MAP message corruption and fabrication attacks. It provides a framework that allows ciphering, signing and authorization of MAP messages.

Internet Protocol Security (IPSec) [68] is also another major protocol that may be used to protect MAP messages that run on the Internet Protocol.

In summary, both MAPSec and IPSec can be only used to provide security to MAP messages between two end points. In order to protect all communications in mobile telecommunication networks it is clearly not enough to protect only the MAP protocol on the links between service nodes [13]. A complete traffic protection scheme for mobile telecommunication networks must provide end-to-end security to individual data items in a signal flow with the ability to track the source service node of the data item. Such protection is completely lacking in current protection schemes for mobile telecommunication networks.

Other prevalent schemes for providing message security include onion routing [25]. In onion routing, the path to be taken by the message is determined first, and for each portion of the path, a layer of encryption is added. This encryption is peeled off when the message exits the corresponding part of the path. Applying such a scheme for our network model is very tedious as (1) every service node must share symmetric keys with every other service node in the network, making the number of keys used in this case exponential and the scheme impractical; (2) the path to be taken by the signal flow must be computed in advance of every signal flow and keys must be distributed securely adding additional computation and signaling load; (3) encryption must be added in layers for each individual data item, taking into consideration the multiple destination service nodes of some data item, making the scheme computationally expensive.

While the use of symmetric key cryptography to sign individual data items may seem a better choice than public key cryptography, it must be noted the use of symmetric key cryptography in EndSec suffers from a number of drawbacks explained as follows: (1) the number of keys required to implement this scheme is exponential, as the keys must
be shared by every pair of service nodes in the network; (2) as the destination service nodes of individual data items cannot be predicted in advance, using symmetric keys would require advance pre-processing; and (3) as most data items have multiple destinations, multiple copies of data items must be created and signed by the corresponding keys. Overall symmetric key cryptography for signing data items is highly impractical as its implementation requires high overhead in terms of keys, data items, and added processing.

In the group key scheme, a group of users share a common key which may be used to secure communications among a group of users. While the usage of group keys to sign individual data items does not suffer from the key explosion problem, and creating multiple signed data items for every destination service node, it also suffers from some serious drawbacks detailed as follows: (1) in advance of any signal flow, the signal flow path must be computed and keys distributed; and (2) the most serious flaw in this scheme is that group keys do not allow tracking of the originators or derivators of data items as the group keys used to sign data items are common to all service nodes in the signal flow path.

To the best of our knowledge our work is the first of its kind to propose a unique scheme for end-to-end security of data items. Our scheme guarantees security for data items as well as allows tracking of compromised service nodes.

7.11 Discussion

In conclusion, the mobile telecommunication network is highly vulnerable and may be subjected to very damaging attacks due to corrupt data sources and service logic. EndSec is a good solution for securing data items in signaling messages, and preventing corruption from propagating. EndSec is a good scheme if corruption must be contained to the corrupt service node. While EndSec may incur higher than usual signaling load, it is best scheme so far for providing security to all signaling messages at a low level. Alternatives to EndSec such as the gateway and probabilistic schemes are good solutions for reducing load but maintaining security.
Chapter 8

Conclusion

In this chapter, we summarize the dissertation, and present possible extensions to this research.

8.1 Summary of Thesis

In this dissertation, we have assessed the security of mobile telecommunication networks. Our security assessment covers attack identification and classification, vulnerability assessment, and network protection.

In attack identification and classification, we conducted a detailed study of network functionality through our proposed abstract model, and also conducted a detailed study of attacks on the network. Our study has given us insights into the unique cascading effect of attacks on mobile telecommunication networks. We also realized the possible occurrence of cross infrastructure cyber attacks, where attacks may be launched on the mobile telecommunication network from the Internet. Our three dimensional attack taxonomy classifies attacks based on three different categories. With our taxonomy it is possible to extract common characteristics of attacks. Another unique feature of this taxonomy is that it can be expanded to accommodate future expansions of technology.

In our vulnerability assessment, we developed toolkits to assess potential cascading effects of attacks, and quantify the benefits of a security solution using visual attack graphs. Our toolkits are unique in that they can handle the variety and diversity of the physical network configuration. Such handling is made possible due to the usage of SDL specifications, and incorporation of novel dependency and propagation models.

Our toolkits may be used by network operators to detect sources of unusual network events and to protect the network from outage due to cascading attacks, during critical events such as 9/11. It can also be used to check incoming traffic from certain
rogue networks (hence trace corruption). Also, our toolkits can be used to evaluate security protocols before their actual deployment saving network operators from financial losses.

Results from our vulnerability assessment show that corrupt data sources, and service logic in service nodes have the ability to cause most damage in terms of disrupting the maximum number of subscribers affected. Any mechanism that detects such attacks can also prevent them from occurring.

Our work on network protection directly follows from this observation. As a solution to detecting data sources, and service logic corruption we propose the *End-to-End Security - EndSec* protocol. EndSec brings accountability to the network security, and detects data source, and service logic corruption by requiring each data item to be signed by the creator. While the utility of EndSec is illustrated on the mobile telecommunication network similar conclusions may also be drawn for any other application requiring multi-hop security.

The major finding from our research is that the most damaging attacks may be launched from the: (1) Internet; (2) by exploiting insecure core network signaling messages; or (3) by corrupting data sources or service logic in service nodes. However, any attack that can lead to cascading has the potential to cause greatest damage. Cascading attacks are stealthy and unexpected as their ability to cause damage depends greatly on normal network operations, and their ability to remain undetectable.

Our dissertation is the first comprehensive security assessment of the mobile telecommunication network. Our work is of significance as we have exposed the many vulnerabilities of the network, provided a unique taxonomy, presented toolkits to trace attacks through the network, evaluated security protocols, and proposed a unique multi-hop security protocol. Our research has brought focus to the state of security of the mobile telecommunication network, so much so that a portion of this dissertation has been used as an important input to the *Vulnerabilities Threat Modeling Working Group* of the *Next Generation Networks Task Force* of the *Presidents National Security Telecommunications Advisory Panel*. 
8.2 Future Work

In this section, we detail possible future work in the area of vulnerability assessment, and security schemes for mobile telecommunication networks.

8.2.1 Vulnerability Assessment

Our toolkits assesses potential network vulnerabilities by using our Knowledge Base that is populated with SDL specifications. Toolkits use the SDL specifications to find the dependencies in the network that cause corruption to propagate. While we have populated the Knowledge Base with SDL specifications of major services in the circuit switched domain (such as call delivery, short message, handover, mobility management, and supplementary services) we have overlooked the minor services in the circuit switched domain (such as CAMEL), and all the services in the Packet Switched/IP multimedia subsystem.

In the future, the Knowledge Base may be populated with the minor services in the circuit switched domain in addition to the existing major services. In doing so, the Knowledge Base will contain a comprehensive list of services in the circuit switched network. Such a comprehensive list of services for the circuit switched domain may be used to discover all cascading attacks on the circuit switched domain.

As the IP multimedia subsystem is the future of the telecommunication network, any work to enhance the Knowledge Base is incomplete if the IP multimedia subsystem services is not added to Knowledge Base. However, adding the IP multimedia subsystem to the Knowledge Base is a challenging task as not all the IP multimedia subsystem specifications come with the SDL specifications. Creators of the Knowledge Base will have to have a deep understanding of network operations to read the text detailing operations and code the network functions in a form similar to the SDL specifications that is suitable to the Knowledge Base.

With such a comprehensive Knowledge Base containing both the circuit switched domain and IP multimedia subsystem, our toolkit can be a valuable addition to network
operators, aiding in detecting the origin of cascading attacks, and its many stealthy effects.

In addition to the mobile telecommunication network, our toolkits can also be extended to any wireless network that is based on specifications if the propagation and dependency models are suitably changed, and Knowledge Base is also modified accordingly.

8.2.2 Network Security Schemes

As part of securing the network, we proposed EndSec. EndSec protects signaling messages at the data item level. EndSec prevents corruption from propagating by allowing corruption to be detected, and contained.

While EndSec can detect mediator service node and signaling message corruption EndSec cannot detect originator and derivator corruption that does not cause unexpected errors. Typically originator and derivator corruption is caused due to data source or service logic corruption. A data item corrupted as a result of a corrupt data source cannot be detected by another service node, as this data item may indicate a subscriber preference that is proprietal and known only to the originator or derivator service node. Again, a data item corrupted due to corrupt service logic cannot be detected by another service node, as this data item may indicate a system value that can only be derived using the functionality available in the derivator service node.

Hence, the next required security scheme must be able to detect corruption caused by originator or derivator service nodes with corrupt data sources or service logic. The new scheme must detect invalid changes to the data source or to the service logic that generates data items. A good way to do this is to use a SDL specification based model checker to verify the integrity of the suggested change, thereby preventing invalid changes.

Another method to prevent invalid changes to data sources or to the service logic, is to require author authentication, or code signatures. A scheme must be designed that requires special signatures before a data source or service logic may be changed.

Such a scheme would provide more security to the network, and bring accountability ensuring that the attacks do not cascade.
8.3 Final Remarks

Next to the Internet, the mobile telecommunication network is the most highly used communication network. It is also the most vulnerable network with inadequate security measures, making it a very attractive target to adversaries wishing to cause complete communication outages during emergencies. As the mobile telecommunication network is moving in the direction of the Internet i.e., becoming an amalgamation of several types of diverse networks, more attention must be paid towards securing these networks. A push from government agencies requiring mandatory security standards for operating mobile telecommunication networks would be just the momentum needed towards securing these networks.

Of all the many possible attacks on the mobile telecommunication network, cascading attacks have the most potential to stealthily cause major network mis-operation. At the present stage, there is no standardized scheme to protect from such attacks. EndSec is a good solution for protecting from cascading attacks, as it requires every data item to be signed by the source service node. As service nodes are unlikely to corrupt data items they are to be accounted for by their signatures, the possibility of cascading attacks is greatly reduced. EndSec has the added advantage of providing end-to-end security for all types of signaling messages. Hence, standardizing EndSec and mandating its deployment would be another good step towards securing the network.

Until security solutions such as EndSec are finalized, our toolkit aCAT would be an excellent aid to network operators to detect cascading attacks. aCAT could be used for real-time detection of sources of unusual network events thereby protecting the network from unusual outage due to cascading attacks during critical events such as 9/11. Likewise, aCAT could also be used to check (hence trace corruption) of incoming traffic from certain rogue networks.

Finally, both the Internet and PSTN are open gateways that adversaries can use to gain access and attack the network. As PSTNs security is unlikely to be improved, its gateways to the core network must at least be secured. Likewise, as neither the Internets design nor security is unlikely to be changed to suit the mobile telecommunication network, at least its gateways to the core network must be adequately secured.
Finally, as the mobile telecommunication network is an amalgamation of many diverse networks, it has too many vulnerable points. Hence, the future design of the network must be planned to reduce the number of vulnerable networks points, and reduce the number of service nodes that participate in servicing the subscriber, thereby reducing the number of points from where the adversary may attack.
Bibliography


[71] Switch. 5ESS Switch. In http://www.alleged.com/telephone/5ESS/.


Appendix A

Abstract Model

**IMS Domain:** The abstract model for the IMS domain is illustrated in Fig. A.1 and described in Tables A.1 and A.2. The IMS abstract model in Fig. A.1 shows a user agent. This user agent is a part of the mobile device. It contains the functionality and data needed by the mobile device to interact with the core network and to provide service to the subscriber.

![Diagram of service nodes in the IP Multimedia subsystem](image)

Fig. A.1. Abstract Model of service nodes in the IP Multimedia subsystem
<table>
<thead>
<tr>
<th>Service node</th>
<th>Name of Agent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Agent</td>
<td>User Authenticator</td>
<td>Manages authentication data (secret key, algorithm etc) stored in the mobile device.</td>
</tr>
<tr>
<td></td>
<td>Resource Reservation</td>
<td>Decides the resources needed for the session based on the incoming or outgoing session descriptors and its own terminal capabilities.</td>
</tr>
<tr>
<td></td>
<td>Signaling Channel</td>
<td>Establishes contact between the mobile station and the network.</td>
</tr>
<tr>
<td></td>
<td>Media Flows</td>
<td>Starts or Initiates media flows for a session based on terminal capabilities.</td>
</tr>
<tr>
<td></td>
<td>User Session</td>
<td>Initiates or handles incoming sessions on behalf of the user.</td>
</tr>
<tr>
<td></td>
<td>User Registration</td>
<td>It is invoked when a mobile device powers up, or enters, or exits a new area. This invokes the foreign registration agent at the P-CSCF.</td>
</tr>
<tr>
<td>HSS</td>
<td>Authenticator</td>
<td>In-charge of retrieving keys from the authentication data source and generating authentication material for all users subscribing to its network. It may be invoked to authenticate the subscriber.</td>
</tr>
<tr>
<td></td>
<td>Subscriber Registra-</td>
<td>Handles all registration messages, checks if the user is already registered, or is allowed to register in the visiting network. Contacts the authenticator agent in the user agent.</td>
</tr>
<tr>
<td></td>
<td>Session Control Agent Data Manager</td>
<td>Manages data of all the S-CSCF's session control agent in the network.</td>
</tr>
<tr>
<td></td>
<td>Update Tracker</td>
<td>Informs the assigned session control agent at the S-CSCF of the user profile settings and services.</td>
</tr>
<tr>
<td>I-CSCF</td>
<td>Home Registration</td>
<td>Receives registration messages at the entry point into the home network.</td>
</tr>
<tr>
<td></td>
<td>Address Resolver</td>
<td>Resolves URIs to IP addresses.</td>
</tr>
<tr>
<td></td>
<td>Home Session</td>
<td>Handles sessions for a user and forward messages between session control agent and proxy session control agent.</td>
</tr>
<tr>
<td></td>
<td>S-SCSF Session Agent Selector</td>
<td>Selects session control agent for the registered subscriber based on subscribers requirements and session control agent capabilities.</td>
</tr>
</tbody>
</table>
Table A.2. Agents in IP Multimedia Subsystem - II

<table>
<thead>
<tr>
<th>Service node</th>
<th>Name of Agent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-CSCF</td>
<td>SA Negotiator</td>
<td>Negotiates security association with the user agent.</td>
</tr>
<tr>
<td></td>
<td>Address Resolver</td>
<td>Resolves URIs to IP Addresses.</td>
</tr>
<tr>
<td></td>
<td>QOS Resource Authorizer</td>
<td>Approves and authorizes the QOS resource reservation requests based on current network load and radio link capacity.</td>
</tr>
<tr>
<td></td>
<td>Proxy Session Control Agent</td>
<td>Handles incoming and outgoing sessions for a user currently registered with the visiting network.</td>
</tr>
<tr>
<td></td>
<td>Subscriber Home Domain Data Manager</td>
<td>Saves addresses of visiting subscriber’s session control agent in the S-CSCF during registration.</td>
</tr>
<tr>
<td></td>
<td>Foreign Registration</td>
<td>It is the registration agent in the visiting network and is invoked by the user registration agent in the user agent.</td>
</tr>
<tr>
<td>S-CSCF</td>
<td>Serving Authenticator</td>
<td>Manages authentication data cached from the home network and performs the 3-way AKA.</td>
</tr>
<tr>
<td></td>
<td>Session Registration</td>
<td>Handles subscriber registration by caching user profile data, services data from home domain and stores home domain address locally. It also provides the home network the address of the assigned session control agent in the S-CSCF.</td>
</tr>
<tr>
<td></td>
<td>Session Control</td>
<td>Handles sessions for the user in the home network. Performs service control based on subscribers profile &amp; subscribed services data.</td>
</tr>
</tbody>
</table>
**LB-IM Service:** We present the abstract model for the LB-IM in Fig. A.2, and describe the abstract model in Table A.3.

![Abstract Model of service nodes in the LB-IM](image)

**Fig. A.2.** Abstract Model of service nodes in the LB-IM

**Table A.3.** Agents in LB-IM Server

<table>
<thead>
<tr>
<th>Name of Agent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscriber Parameter Manager</td>
<td>Manages subscriber preferences.</td>
</tr>
<tr>
<td>Request Manager</td>
<td>Locate subscribers and respond to subscriber location requests.</td>
</tr>
<tr>
<td>IM Manager</td>
<td>In-charge of sending location track requests to members on the buddy list.</td>
</tr>
<tr>
<td>Location Tracking</td>
<td>Responsible for tracking the location of the user by querying the network and maintaining current location data of the subscriber.</td>
</tr>
<tr>
<td>Authenticator</td>
<td>Responsible for authenticating subscribers and other servers accessing the LB-IM Server.</td>
</tr>
</tbody>
</table>
**CBS Service:** We present the abstract model for the CBS in Fig. A.3, and describe the abstract model in Table A.4.

![Abstract Model of service nodes in the CBS](image)

**Fig. A.3.** Abstract Model of service nodes in the CBS

**Table A.4.** Agents in CBS Server

<table>
<thead>
<tr>
<th>Name of Agent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscriber Parameter Manager</td>
<td>Manages all of the parameters set by the subscriber.</td>
</tr>
<tr>
<td>Time Agent</td>
<td>Times the call and logs it.</td>
</tr>
<tr>
<td>Forwarding</td>
<td>Forward the call to the appropriate party.</td>
</tr>
<tr>
<td>Bill Calculator</td>
<td>Calculates the bill, based on call time and other constraints.</td>
</tr>
<tr>
<td>Client Checker</td>
<td>Interfaces the mail server with call forwarding server and queries the mail server based on subscriber parameters and retrieve information.</td>
</tr>
<tr>
<td>Authenticator</td>
<td>Maintains subscriber authentication data and the address of the third parties used to authenticate the subscriber.</td>
</tr>
</tbody>
</table>
Appendix B

Attack Overview in IP Multimedia Subsystem

We tabulate a list of possible attacks on the IMS Domain. Tables B.1 and B.2 shows attacks on IMS domain classified by CASE 1. As Level I of the IMS Subsystem, is similar to Level I of the CS domain we do not consider it in Table B.1.

Also, in the IMS Dimension I-Level III may be further divided into Level III a and Level III b.

- **Level IIIa: Remote Access to IMS service nodes**: The adversary has remote access to IMS service nodes.

- **Level IIIb: Direct access to IMS service nodes**: The adversary has direct access to IMS service nodes. This level is similar to the Level III defined in the original taxonomy.

We use the above classification to tabulate our attacks. The cross infrastructure cyber attacks in the IMS case is very similar to the CS domain hence we do not consider them here.
<table>
<thead>
<tr>
<th>Categories</th>
<th>Level II: Links connecting IMS service nodes</th>
<th>Level IIIa: Remote connection to IMS service nodes</th>
<th>Level IIIb: Direct connection to IMS service nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interception</strong></td>
<td>Analyze traffic patterns. Eavesdrop on calls and voice messages. Capture authentication vectors sent from authenticator agent (HSS) to serving authenticator agent in S-CSCF and use it in replay attacks.</td>
<td>Analyze traffic patterns, gather data arriving at the compromising entity.</td>
<td>Gather subscriber data stored in service nodes violate subscriber privacy. Track a victim subscriber's activities at the session control agent data manager in the HSS.</td>
</tr>
<tr>
<td><strong>Fabrication/Insertion</strong></td>
<td>Send repeated INVITE messages to session control agent in S-CSCF thereby overload home network, and unable to service valid incoming requests and generate outgoing responses. Send registration and location update messages to subscriber registration agent in (HSS) causing incorrect call routing, and shutting down the mobile device. Send profile change messages to session control agent data manager in HSS. Request authentication vectors (AV) from authenticator agent (HSS).</td>
<td>Insert new service logic to the compromised service node so that it directs all traffic to the adversary's service location.</td>
<td>Insert new service logic to the compromised service node so that it is disabled at a particular time. With access to HSS, add new subscribers to the profile setting, subscribed data stores and not the billing data store, allowing fraudulent subscribers can access the system without paying.</td>
</tr>
<tr>
<td><strong>Modification Of Resources</strong></td>
<td>Change the SIP:URI in the registration signaling message so that the subscriber cannot be registered. Change the destination SIP:URI address in the outgoing INVITE signaling messages and the request is sent to incorrect party. Change the SIP:URI addresses in the incoming session invite messages; message does not reach the subscriber. Change the AVs sent to authenticate the User Agent, so that the User Agent is never authenticated.</td>
<td>Cause buffer overflows, to execute malicious code and hence modify service logic.</td>
<td>Modify destination of session invite signaling messages. Modify session descriptors, Change subscriber profile, AV, location mapping at S-CSCF and HSS data sources. Modify the session handling capabilities at the session control agent and proxy session control agent. Modify service logic in subscriber home domain manager so that user session control agent mapping is corrupt resulting in denial of service to the users.</td>
</tr>
<tr>
<td>Categories</td>
<td>Level II: Links connecting IMS service nodes</td>
<td>Level IIIa: Remote connection to IMS service nodes</td>
<td>Level IIIb: Direct connection to IMS service nodes</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Denial Of Service</td>
<td>Send repeated INVITE messages to session control agent in S-CSCF and overload Home Network. Send large number of authentication request to authenticator agent in (HSS) slowing down the authenticator agent and surrounding links. Send repeated registration or location update messages to subscriber registration agent in (HSS) causing incorrect call routing, and shutting down of mobile device. Send profile change messages to HSS (Session Control Agent Data Manager).</td>
<td>Buffer overflow could also cause a denial of service at the IMS service nodes.</td>
<td>Modify a parameter in the Authentication Vector Algorithm so that none of the mobile nodes may be authenticated and hence do not get service.</td>
</tr>
<tr>
<td>Interruption</td>
<td>Delete registration or location update messages to subscriber registration agent in HSS resulting in incorrect call routing. Delete call invite requests. Delete AVs sent from authenticator agent (HSS) to serving authenticator agent in S-CSCF.</td>
<td>Delete data sources in the service nodes.</td>
<td>Delete subscriber preferences. At authenticator agent (HSS), replace ciphering algorithm with fraudulent algorithms failing to authenticate all mobile devices. Delete subscriber preferences</td>
</tr>
</tbody>
</table>
Appendix C

Data items in Messages

In this chapter, we present messages in the call delivery service, and the data items they carry.

A. Initial Address Message (IAM)

<table>
<thead>
<tr>
<th>Data Parameters</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message type</td>
<td>1</td>
</tr>
<tr>
<td>Nature of connection indicators</td>
<td>1</td>
</tr>
<tr>
<td>Forward call indicators</td>
<td>2</td>
</tr>
<tr>
<td>Calling party’s category</td>
<td>1</td>
</tr>
<tr>
<td>Transmission medium requirement</td>
<td>1</td>
</tr>
<tr>
<td>Called party number</td>
<td>4</td>
</tr>
<tr>
<td>Transit network selection</td>
<td>4</td>
</tr>
<tr>
<td>Call reference</td>
<td>7</td>
</tr>
<tr>
<td>Calling party number</td>
<td>4</td>
</tr>
<tr>
<td>Optional forward call indicators</td>
<td>3</td>
</tr>
<tr>
<td>Redirecting number</td>
<td>3</td>
</tr>
<tr>
<td>Closed user group interlock code</td>
<td>6</td>
</tr>
<tr>
<td>Connection request</td>
<td>9</td>
</tr>
<tr>
<td>Original called number</td>
<td>4</td>
</tr>
<tr>
<td>User-to-user information</td>
<td>131</td>
</tr>
<tr>
<td>Access transport</td>
<td>3</td>
</tr>
<tr>
<td>User service information</td>
<td>13</td>
</tr>
<tr>
<td>User-to-user indicators</td>
<td>3</td>
</tr>
<tr>
<td>Generic number</td>
<td>5</td>
</tr>
<tr>
<td>Propagation delay counter</td>
<td>4</td>
</tr>
<tr>
<td>User service information prime</td>
<td>13</td>
</tr>
<tr>
<td>Network specific facility</td>
<td>4</td>
</tr>
<tr>
<td>Generic digits</td>
<td>4</td>
</tr>
<tr>
<td>Origination ISC point code</td>
<td>4</td>
</tr>
<tr>
<td>User teleservice information</td>
<td>5</td>
</tr>
<tr>
<td>Remote operations</td>
<td>8</td>
</tr>
<tr>
<td>Parameter compatibility information</td>
<td>4</td>
</tr>
<tr>
<td>Generic notification indicator</td>
<td>3</td>
</tr>
<tr>
<td>Service activation</td>
<td>3</td>
</tr>
<tr>
<td>Generic reference</td>
<td>5</td>
</tr>
<tr>
<td>MLPP precedence</td>
<td>8</td>
</tr>
<tr>
<td>Transmission medium requirement prime</td>
<td>3</td>
</tr>
</tbody>
</table>

A. Initial Address Message (IAM) continued

<table>
<thead>
<tr>
<th>Data Parameters</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location number</td>
<td>4</td>
</tr>
<tr>
<td>Forward GVNS</td>
<td>26</td>
</tr>
<tr>
<td>CCSS</td>
<td>3</td>
</tr>
<tr>
<td>Network management controls</td>
<td>3</td>
</tr>
<tr>
<td>Circuit assignment map</td>
<td>7</td>
</tr>
<tr>
<td>Correlation id</td>
<td>3</td>
</tr>
<tr>
<td>Call diversion treatment indicators</td>
<td>3</td>
</tr>
<tr>
<td>Called IN number</td>
<td>4</td>
</tr>
<tr>
<td>Call offering treatment indicators</td>
<td>3</td>
</tr>
<tr>
<td>Conference treatment indicators</td>
<td>3</td>
</tr>
<tr>
<td>SCF id</td>
<td>3</td>
</tr>
<tr>
<td>UID capability indicators</td>
<td>3</td>
</tr>
<tr>
<td>Echo control information</td>
<td>3</td>
</tr>
<tr>
<td>Hop counter</td>
<td>3</td>
</tr>
<tr>
<td>Collect call request</td>
<td>3</td>
</tr>
<tr>
<td>Application transport parameter</td>
<td>5</td>
</tr>
<tr>
<td>Pivot capability</td>
<td>3</td>
</tr>
<tr>
<td>Called directory number</td>
<td>5</td>
</tr>
<tr>
<td>Original called IN number</td>
<td>4</td>
</tr>
<tr>
<td>Calling geodetic location</td>
<td>3</td>
</tr>
<tr>
<td>Network routing number</td>
<td>4</td>
</tr>
<tr>
<td>QoR capability</td>
<td>3</td>
</tr>
<tr>
<td>Pivot counter</td>
<td>3</td>
</tr>
<tr>
<td>Pivot routing forward information</td>
<td>3</td>
</tr>
<tr>
<td>Redirect capability</td>
<td>3</td>
</tr>
<tr>
<td>Redirect counter</td>
<td>3</td>
</tr>
<tr>
<td>Redirect status</td>
<td>3</td>
</tr>
<tr>
<td>Redirect forward information</td>
<td>3</td>
</tr>
<tr>
<td>Number portability forward information</td>
<td>3</td>
</tr>
</tbody>
</table>

| Number of Data Items                     | 62      |
| Message Overhead                         | 173     |
| Sum of Data Items                        | 399     |
| Message Size                             | 572     |
### B. MAP SRI

<table>
<thead>
<tr>
<th>Data Parameters</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke Id</td>
<td>4</td>
</tr>
<tr>
<td>Interrogation Type</td>
<td>4</td>
</tr>
<tr>
<td>GMSC Address</td>
<td>9</td>
</tr>
<tr>
<td>MSISDN</td>
<td>9</td>
</tr>
<tr>
<td>OR Capability</td>
<td>4</td>
</tr>
<tr>
<td>CUG Interlock</td>
<td>4</td>
</tr>
<tr>
<td>Number of Forwarding</td>
<td>5</td>
</tr>
<tr>
<td>ISDN BC</td>
<td>4</td>
</tr>
<tr>
<td>ISDN LLC</td>
<td>4</td>
</tr>
<tr>
<td>ISDN HLC</td>
<td>4</td>
</tr>
<tr>
<td>Network Signal Info</td>
<td>200</td>
</tr>
<tr>
<td>Supported CAMEL Phases</td>
<td>2</td>
</tr>
<tr>
<td>Call Reference Number</td>
<td>8</td>
</tr>
<tr>
<td>Forwarding Reason</td>
<td>4</td>
</tr>
<tr>
<td>Basic Service Group</td>
<td>5</td>
</tr>
<tr>
<td>Alerting Pattern</td>
<td>1</td>
</tr>
<tr>
<td>Supported CCBS Phase</td>
<td>4</td>
</tr>
<tr>
<td>IST Support Indicator</td>
<td>4</td>
</tr>
<tr>
<td>Call Diversion Treatment Indicator</td>
<td>1</td>
</tr>
</tbody>
</table>

| Number of Data Items     | 19      |
| Message Overhead         | 72      |
| Sum of Data Items        | 280     |
| Message Size             | 352     |

### C. MAP SRI ACK

<table>
<thead>
<tr>
<th>Data Parameters</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke Id</td>
<td>4</td>
</tr>
<tr>
<td>MSISDN</td>
<td>9</td>
</tr>
<tr>
<td>CUG Interlock</td>
<td>4</td>
</tr>
<tr>
<td>IMSI</td>
<td>8</td>
</tr>
<tr>
<td>MSRN</td>
<td>9</td>
</tr>
<tr>
<td>Forwarding Data</td>
<td>41</td>
</tr>
<tr>
<td>VMSC address</td>
<td>9</td>
</tr>
<tr>
<td>GMSC Camel Subscription Info</td>
<td>21</td>
</tr>
<tr>
<td>Location Information</td>
<td>70</td>
</tr>
<tr>
<td>Subscriber State</td>
<td>14</td>
</tr>
<tr>
<td>Basic Service Code</td>
<td>5</td>
</tr>
<tr>
<td>North American Equal Access pre-13</td>
<td>13</td>
</tr>
<tr>
<td>SS-List</td>
<td>30</td>
</tr>
<tr>
<td>numberPortabilityStatus</td>
<td>4</td>
</tr>
<tr>
<td>IST Alert Timer</td>
<td>4</td>
</tr>
</tbody>
</table>

| Number of Data Items     | 15      |
| Message Overhead         | 56      |
| Sum of Data Items        | 245     |
| Message Size             | 301     |

### D. MAP PRN

<table>
<thead>
<tr>
<th>Data Parameters</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke Id</td>
<td>4</td>
</tr>
<tr>
<td>IMSI</td>
<td>8</td>
</tr>
<tr>
<td>MSC Number</td>
<td>9</td>
</tr>
<tr>
<td>MSISDN</td>
<td>9</td>
</tr>
<tr>
<td>LMSI</td>
<td>4</td>
</tr>
<tr>
<td>GSM Bearer Capability</td>
<td>1</td>
</tr>
<tr>
<td>Call Reference Number</td>
<td>8</td>
</tr>
<tr>
<td>GMSC Address</td>
<td>9</td>
</tr>
<tr>
<td>Alerting Pattern</td>
<td>1</td>
</tr>
<tr>
<td>Network signal Info</td>
<td>200</td>
</tr>
<tr>
<td>Supported CAMEL Phases in GMSC</td>
<td>2</td>
</tr>
<tr>
<td>Additional Signal Info</td>
<td>1</td>
</tr>
</tbody>
</table>

| Number of Data Items     | 12      |
| Message Overhead         | 58      |
| Sum of Data Items        | 456     |
| Message Size             | 514     |

### E. MAP PRN ACK

<table>
<thead>
<tr>
<th>Data Parameters</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke Id</td>
<td>4</td>
</tr>
<tr>
<td>Roaming Number</td>
<td>9</td>
</tr>
</tbody>
</table>

| Number of Data Items     | 2       |
| Message Overhead         | 26      |
| Sum of Data Items        | 13      |
| Message Size             | 39      |

### F. SIFIC

<table>
<thead>
<tr>
<th>Data Parameters</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSRN</td>
<td>9</td>
</tr>
<tr>
<td>Bearer service</td>
<td>4</td>
</tr>
<tr>
<td>Teleservice</td>
<td>5</td>
</tr>
<tr>
<td>Dialed number</td>
<td>9</td>
</tr>
<tr>
<td>Number of forwarding</td>
<td>5</td>
</tr>
<tr>
<td>CUG Interlock</td>
<td>4</td>
</tr>
</tbody>
</table>

| Number of Data Items     | 6       |
| Message Overhead         | 44      |
| Sum of Data Items        | 36      |
| Message Size             | 80      |

### G. PAGE MS

<table>
<thead>
<tr>
<th>Data Parameters</th>
<th># Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMSI</td>
<td>8</td>
</tr>
<tr>
<td>eMLPP priority</td>
<td>4</td>
</tr>
<tr>
<td>TMSI</td>
<td>4</td>
</tr>
<tr>
<td>Location area identity</td>
<td>5</td>
</tr>
<tr>
<td>Channel type</td>
<td>4</td>
</tr>
</tbody>
</table>

| Number of Data Items     | 5       |
| Message Overhead         | 42      |
| Sum of Data Items        | 25      |
| Message Size             | 67      |
Vita

Kameswari Kotapati

Kameswari Kotapati was born in Chennai (previously Madras), the fourth largest metropolitan city in India. She received her Bachelor of Engineering degree from the University of Madras at Chennai, India where she topped her graduating class. She received her M. S. degree from University of Massachusetts, Dartmouth. She is a Ph.D candidate in the department of Computer Science and Engineering at Pennsylvania State University. Her doctoral dissertation in the area of security in emerging wireless telecommunication networks, was advised by Dr. Thomas F. LaPorta and Dr. Peng Liu. She was one of the first to perform extensive studies in this area and has authored her work in a number of journals and conferences. The highlight of her dissertation, was that it was used as a major input by the Vulnerabilities Threat Modeling Working Group of the Next Generation Networks Task Force of the Presidents National Security Telecommunications Advisory Panel. She is a member of the Networking and Security Research Center and Cyber Security Lab at Penn State, and the IEEE. Her research interest include mobile telecommunication network security, services, & protocols, location based security, services, & protocols, and seamless capabilities.