MODELLING ATTACK SCENARIOS AGAINST SOFTWARE INTENSIVE CRITICAL INFRASTRUCTURES

Claudio Balducelli

Agency for New Technologies, Energy and Environment (ENEA)[1]

Keywords: Intrusion detection, attack trees, attack patterns, critical infrastructures, attack simulation

Abstract

Recent increases in physical and cyber terrorism have placed large complex critical infrastructures, such as information intensive networked systems, increasingly under threat. The fault tolerance of these networks are decreasing by the increasing dependency of large complex critical infrastructures on each other and on the Internet. There has already been a great deal of research done into ways of increasing the reliability of the hardware components of large complex critical infrastructures.

This paper proposes a means for modelling and documenting information-security attacks in a structured and reusable form in such a way that security analysts will be able to use the structures described to identify commonly occurring attack patterns derived from real attack data.

In fact, a very important requirement to evaluate and test the possibility to safeguard information intensive critical infrastructures is the availability of a sets of realistic types of attacks and intrusions.

The paper proposes a sort of *reference language* to model and implement intrusions and faults scenarios. The proposal is to formalize the propagation paths of attackers or a system faults through the definition of attack *trees*. The root of an attach tree represent an event that could significantly harm the infrastructure's mission. Every path in the attack tree represents a unique type of attack (or a unique type of fault propagation) for the infrastructure. Different types of nodes and links can be utilized during the design phase of an attack tree: such a model allows also the insertion of a certain *degree of certainty* inside the different paths.

The utilization of such attack trees realizes also a *formal representation* of the attacks, that will support the development of an *attack simulator* to be utilized as a test-bed to evaluate the robustness of information networked systems.

Two specific test beds are considered. The first one , that is more deeply illustrated in the paper, implements a safeguard system against attacks and intrusions aimed to harm the continuity of services furnished by an electrical power transmission network. The second one tries to monitor and reduce the disturbances generated by swarms, against a national telecommunication network.

[1] *Via Anguillarese 301, 00060 Rome*

The problem

To protect the software intensive infrastructures, a lot of work has been done on the explicit detection of faults, viruses and attacks, based on their *signature*. However this approach cannot identify unknown dangers and frequent updates are necessary. More recently, research has been done into how *normality* can be defined for a system so that an alarm can be raised whenever there is a significant deviation from normality [1]. This can create problems with false alarms, but it has the advantage that it can detect and respond to new faults, accidents and threats. Within the deregulated electricity industry, information about the *normal* state of the network could also be extremely useful to network operators who may not have wide experience and need to cope with new technology, regulations and patterns of demand.

At present, most intrusion detection systems are based around the so-called *fortress* architecture. The system is surrounded by barriers with strict control policies and behind these barriers the whole network lies open.

The problem with this approach is that it offers no protection against malicious insiders. One of the Safeguard[2] objectives is to increase the survivability of Large Complex Critical Infrastructures (LCCI) by creating a protection system that will work even when an attacker is inside. This will not be an extension of the fortress model with better tools, but an alternative technology, utilising an agent-based approach, that will build Safeguard defences into the system from the bottom up. Every application and process within the management network will be watched for abnormality and it will be very difficult for a hostile entity to damage the system without causing significant deviations at this level. This low-level monitoring will be capable of operating independently of the higher levels to further enhance the robustness of the system.

Although the Safeguard multi-agent system [2] will be tested on both electricity and telecommunications networks, this paper will use the electricity management network to illustrate the problem of modelling the most dangerous types of attacks and faults with the intention of generating accident scenarios that will be adopted for testing and validating the developed Safeguard Agents.

The adopted strategy

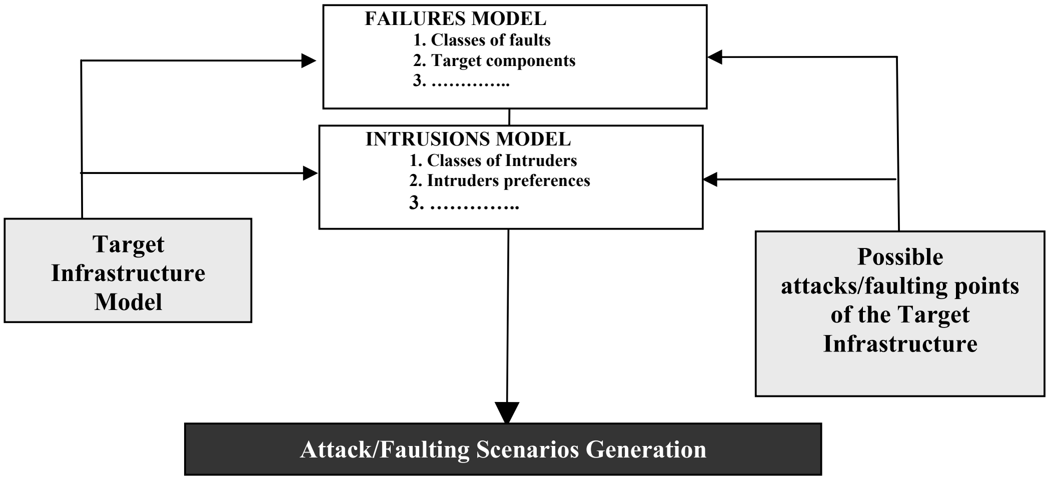
The whole infrastructure is formed by more components that could be linked together. These components-based descriptions represent what in fig. 1 are named as *Target Infrastructure Model*.

The *possible attack/faulting points* of the target infrastructure are also analysed and modelled. To establish specific points of attack and the ways to conduct the attacking activity, a generic sets of vulnerable functions were also considered.

Then the principal types of infrastructure intruders or failures will be examined. Here is a long list of types of intrusions can be considered: intrusions can be unintentional or malicious, they may be done by single persons or by communities and organisations. Single intruders may have very *poor resources* to conduct the attack activity. Organisations, and especially *terrorist organisations*, may have a lot of resources to execute the attacks. They could be generally able to conduct *distributed attacks* composed of sets of efficient sequences of single attacks in various points of the layered infrastructures.

[2] Safeguard is a multi-agent system developed inside an European project, whose Website can be found at www.ist-safeguard.org

Fig. 1 – General strategy to generate specific attack/faulting scenarios



The study and the analysis of the *intruders' behaviour*, in relation to the intrusions typologies and the types of knowledge and preferences of the involved actors, is necessary to support the generation of attack scenarios which are more realistic and have higher degrees of occurrence.

Anyway, not only intrusions can be considered. In some cases the same consequences could be generated also by accidental faults. The intruders' goal is, in many cases, the generation of faults and may be not so easy to distinguish an attack from a simple failure.

Also the *metrics* adopted to monitor the normal/abnormal system state may be not able to make a clear distinction between faults or attacks symptoms.

For the above reasons a same modelling framework was adopted to act as a container of two type of knowledge: knowledge about attacks and knowledge about faults. Attacks and faults patterns could be generated starting from *attack trees* models [3] [4] where the knowledge about the sequences of the events that contribute to the final attack/fault goal is contained**.**

The methodology

Attack trees have existed in various forms, and under various names, for many years, but have been most recently described as a systematic method to characterize system security based on varying attacks [5]. They refine information about attacks by identifying the compromise of enterprise security or survivability as the root of the tree. The ways that an attacker can cause this compromise iteratively and incrementally are represented as lower level nodes of the tree.

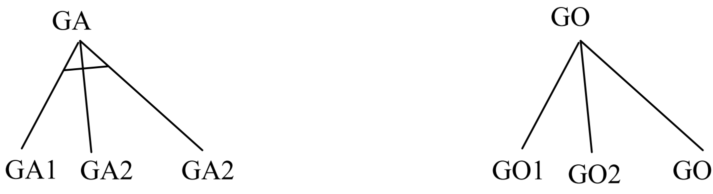
Using attack trees to model intrusions and failures

A very important requirement for Safeguard system testing environment is the availability of a sets of realistic types of attacks and intrusions.

It seems to be useful to have a *reference language* to develop and implement the model of the intrusions or the faults. The proposal is to formalize the ways that an attacker or a system fault can cause this compromise by means of *attack trees*.

Regarding attacks, a critical infrastructure has a potential sets of attack trees. The root of an attach tree represents an *event that could significantly harm the infrastructure's mission*. Every path in the attack tree represents a unique type of attack (or a unique type of fault propagation) for the infrastructure. From the point of view of the attackers, the initial nodes of the attack trees represent the final objectives of the attacks; every node could be decomposed inside lower level nodes using the <AND> and <OR> decomposition types:

Fig. 2 – Attack trees

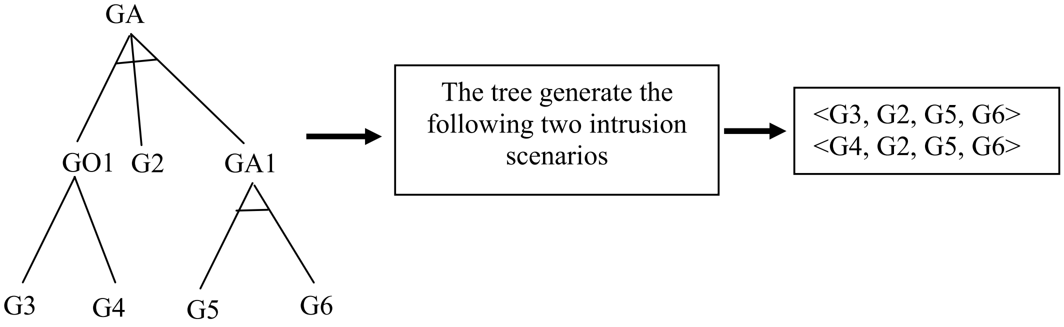


The attack trees could be visualized graphically (see fig 2 above) or in the following textual form:

Goal	GA		***Goal***	GO	
	AND	GA1		***OR***	GO1
		GA2			GO2
		GA3			GO3

An attack trees is composed by any combination of AND/OR type of nodes. The terminal leafs of the tree represent the actions to execute for reaching the high level goals. An attack tree generates *intrusion scenarios,* composed by *sequences of actions,* as in fig 3.

Fig. 3 – Generation of intrusion scenarios from attack trees



The tree generate the following two intrusion scenarios

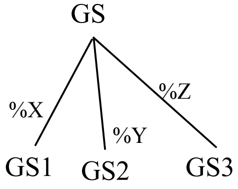
<G3, G2, G5, G6>
<G4, G2, G5, G6>

To an attack tree could be associated a *precondition,* including some assumptions (as skill, resources, knowledge etc.) about the attacker or the attacked system that are necessary for an attack or a fault to succeed and a *post-condition* including the obtained knowledge, results etc. In such a way the above attack tree can be defined in a *text form* as in the following:

Precondition		Pstart		
Goal	GA			
	AND	GO1		
			OR	G3
				G4
		G2		
		GA1		
			AND	G5
				G6
Post-condition:		Presult		

A possible extension of this reference model, as suggested in, consists in another type of node (in addition to the OR and AND type), the SCORE type of node as visualized in fig 4.

Fig. 4 – SCORE type of node



Where 0 < %X < 100 ,   0 < %Y < 100 , 0 < %Z < 100

SCORE node is a special type of <OR> node where the goal GS is not reached with 100% of certainty if almost one of the three sub-goal are reached. When one of the sub-goal, as GS2, is reached the Goal GS is reached with %Y of certainty.

This type of goal could by used to insert *more or less degree of certainty* to the result of an attack, as normally happens in the real domains.

Considering the node G01 in the attack tree of fig. 3 as a *score* node, the tree may be written in textual form as follow:

```
Goal          GA
Precondition  Pstart

              AND   GO1
                          SCORE        (%60)G3
                                       (%40)G4
                    G2
                    GA1
                          AND   G5
                                G6
```

Post-condition: Presult

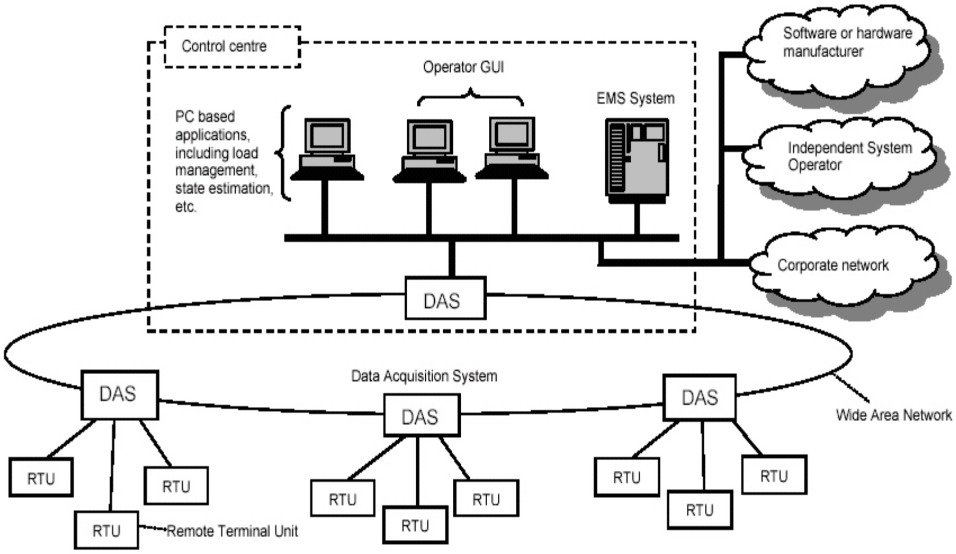
In this case the above attack tree generates the following intrusion scenarios:

<G3, G2, G5, G6> with 60% of Presult certainty
<G4, G2, G5, G6> with 40% of Presult certainty

The electricity network test case

A typical electricity management system consists of a number of interconnected computers running server, database, firewall and monitoring and control software. Figure 5 shows a typical example, with the different functionalities that are being performed and the connections with the corporate network and the Independent System Operator. A small electricity network could be controlled from one information network of this type, housed within a single *Control Centre* connected through a Wide Area Network to more RTUs (*Remote Terminal Units*) that acquire data locally from the various electrical substations. Larger electricity systems require several Control Centres in operation.

Fig. 5 - The electricity grid management network


Control centre
Operator GUI
PC based applications, including load management, state estimation, etc.
EMS System
Software or hardware manufacturer
Independent System Operator
Corporate network
DAS
Data Acquisition System
Wide Area Network
RTU
Remote Terminal Unit

Although this system contains firewalls, virus scanners, diagnostic software and intrusion detection systems it remains vulnerable to unknown attacks, failures and accidents.

Furthermore, the increased interdependency of large complex critical infrastructures puts them increasingly in danger of cascading failures that could knock out several services at once. A solution is needed which can detect problems in these networks and respond to them in real time. Safeguard project is working to create this solution, and the next section will describe the architecture of the testing environment that has been developed so far.

The testing environment

The complete test-environment of Safeguard project will be composed by two different systems:

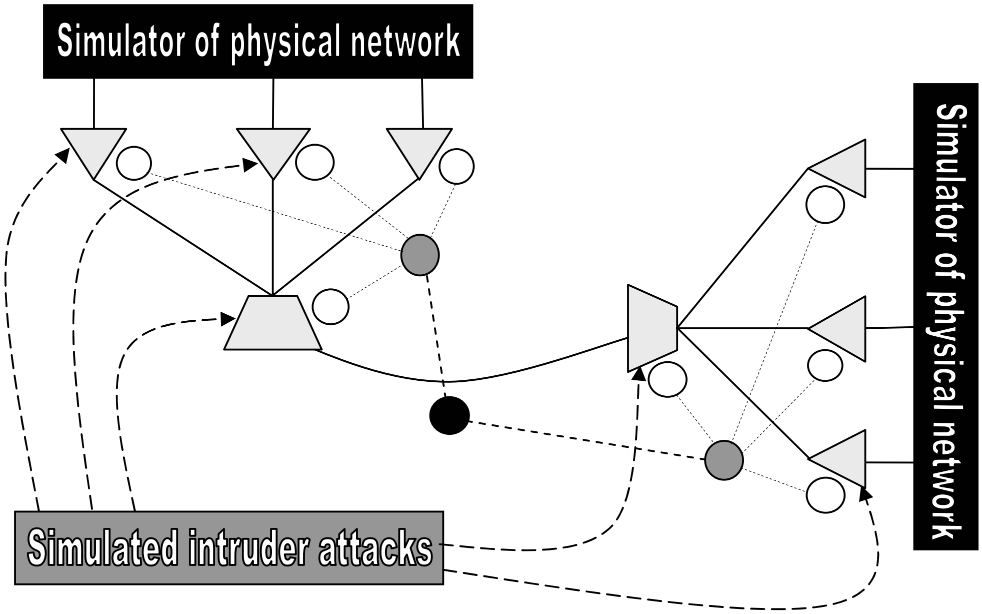
- The first one is the "critical infrastructure" to be safeguarded.
- The second one is the Safeguard system itself that will be implemented as a multi-agent software structure including diagnosing and self-healing algorithms.

As it was illustrated in [2], the first system is composed by three logical layers (physical, control and organizational). The most important issues are relative to faults arising from the control layer, described above for the electricity domain, producing main consequences on the physical layer. Faults arising from organizational layer will not considered.

The control layer represents the reference domain to be safeguarded against faults, intrusions, attacks and communication problems. In the electrical infrastructure, when a faults on the control layer arises, there are not immediate consequences on the physical electrical network. Anyway the fault must be detected in a short time and the problem solved as fast as possible. In fact, if the system works without the availability of adequate control and supervisory functions, the possibility to reach unrecoverable states will increase during time.

Two types of architectures must be developed to produce the complete testing environment: the architecture of the safeguarded system (the first one) and the architecture of the safeguarding system (the second one). The present work is more focused on the description of the first system, as visualized in fig 6, and on the failure scenarios that will be considered here.

Fig. 6 - The electrical infrastructure testing environment


Simulator of physical network
Simulator of physical network
Simulated intruder attacks

Controlled components	Controlling components
▽ RTU	Level 1 ○ **Monitoring, filtering, local diagnosis**
	Level 2 ◍ **Faults classifiers, CBR/Neural recognitions, self-healing, global diagnosis**
⏢ Control Centre	Level 3 ● **Network administration, rescheduling**

The physical layer of this system represents the electrical network. It will be simulated by an *electrical power-flow simulator* producing the relative sources of data variable in time. These data represent tele-measures of the electrical power-flow values and tele-signals relative to the breakers statuses acquired and managed in real time by the control layer. The simulator accepts also tele-commands coming from the control centers operators.

The control layers have to be simulated too. The architecture will include a set of tasks, communicating together and running inside Control Centers and remote (RTUs) units.

The organizational layer will not be simulated thought a software system as the previous one, but some of the most important operator actions (as tele-commands activations and some supervisory functions execution) will be considered and processed by the simulated system.

As visualized in figure 6, inside testing environments it is also available a simulator of intruder attacks with the role of modifying or compromising the most important critical functions. Attacks scenarios can be edited using the attack trees model described above and all the possible combination of attack patterns are simulated.

In the figure the circles represent the safeguard agents (from lower level agents to higher ones), whose functionalities are described in [2]. Using the above architecture, the attacks or failures consequences could be tested *with* and *without* the presence of the safeguard agents with the role of controlling components.

Modelling attack scenarios

Attack scenarios are described in a textual form or using a formalism as the attack trees model described above. The utilization of such model is more useful to obtain computerized attack scenarios and to produce all the combinations of attack patterns that we need to test the vulnerabilities of the considered infrastructure.

An attack scenario description

Here follow, in a textual form, the description of a particular attack scenarios against some functionalities of the controlled components of the electrical grid infrastructure.

Precondition:

An intruder enters inside the WAN network on which RTUs and CCs are connected. He may be a society employer (internal intruder), utilising local computer machines, or an external intruder able to find the IP ports connected to some RTUs.

Attack condition:

The intruder builds a tele-command message and sends it toward an RTU. He can obtain knowledge about the structure of tele-command messages because he was a worker of the electrical society and/or utilising some messages sniffer applications. In both cases he is able to send opening/closing breakers request to an RTU .

Scenario:

The intruder sends the tele-command and, after a certain time, the control centre Operator receives an answering tele-signal indicating a breaker is changing its (closed/opened) status. As it is not the results of any planned operation, the operator supposes that some protection has fired on the electrical grid or some fault took place at physical level.

Possible subsequent failures:

In such scenario the operator may have not the right picture of the electrical network state. For example he could see a power-flow out of limit condition in a transmission line, he may be afraid about subsequent cascade failures and may execute some actions with the objective to recover the situation, but producing instead an opposite effect.

An attack scenario described by fault tree

Here follows the description of the same attack scenario using the attack/fault trees model.

Goal: Producing false statuses in the network with subsequent cascade failures

Precondition: Knowledge how discovering tele-command messages structure/layout

OR **1**. The intruder is employed by the electrical society

AND **1**. Enable access to WAN network through a PC installed at LAN on a Control Centre

2. Build RTU a false Tele-command message to open/close a breaker

SCORE **1**. (80%) Discover Tele-command structure using the society CC manuals.

2. (80%) Discover Tele-command structure accessing source CC software.

3. Build and run a task sending the Tele-command

SCORE **1**. (60%) Build the task inside a portable PC that will be connected to the LAN

2. (70%) Utilise a PC already installed on the LAN network.

2. The intruder is external to the society

AND **1**. Enable access to electrical WAN network through a LAN of another society

2. Build RTU a false Tele-command message to open/close a breaker
***SCORE* 1**. (50%) Discover Tele-command structure using sniffer applications
3. Build and run a task sending the Tele-command
SCORE **1**. (40%) From an external PC.
2. (70%) From portable PC that will be connected to the LAN electricity network.

Postcondition: Tele-signals relative to opening/closing of breakers will arrive a Control Centre out of the operator control

From the above attack tree the following attack patterns list could be elicited:

Attack patterns:
<1.1, 1.2.1,1.3.1> with 48% of probability of success
<1.1, 1.2.2,1.3.1> with 48% of probability of success
<1.1, 1.2.1,1.3.2> with 56% of probability of success
<1.1, 1.2.2,1.3.2> with 56% of probability of success
<2.1, 2.2.1,2.3.1> with 20% of probability of success
<2.1, 2.2.1,2.3.2> with 35% of probability of success

Conclusions

A significant set of attack/fault trees are actually under development for the electricity power transmission system and for the public telecommunication infrastructure. The generated attack patterns will be used inside two testing environments, one for each infrastructure type, with the goal of analysing the main types of vulnerabilities and experimenting the most appropriate detection and self-healing strategies and algorithms implemented inside a distributed multi-agent safeguarding system. A reduction in the number of dangerous attacks and faults, and a mitigation of the consequences are the principal objectives to be achieved.

Acknowledgements

I would like to acknowledge all the members of the Safeguard project team for their help in relation with the ideas and suggestions about the contents of this paper. These include David Gamez, John Bigham and Wes Carter from Queen Mary University of London, Stefan Burschka from Swisscom, Simin Nadjm-Tehrani from Linkoping University, Carlos López Ullod from AIA in Barcelona, Giordano Vicoli and Sandro Bologna from ENEA.

Biography

Claudio Balducelli, degree in physic, University of Rome (1974), is a senior scientist employed at ENEA since 1983 in the field of Information Technologies applied to operator support systems for nuclear control room applications. His interests include operator models and knowledge formalisation, task planning, computerised procedures, diagnostic algorithms, case-base reasoning and learning, fuzzy logic. In the past years he took part, inside EU and Italian national projects, to the definition of user and functional specifications of emergency management training and support systems. He co-ordinated also the prototypical implementation of various site applications like a co-ordination training system for the Genova Oil Port managers (MUSTER project) and an emergency operator support system for major Oil Deposits and Pipelines in Italy. Actually he is team leader inside SAFEGUARD FP5 project (Safeguarding Critical Infrastructures).

References

Thompson B.B., Marks II R.J., Choi J.J., El-Sharkawi M.A., Huang M.Y., Bunje C. (2002), Implicit Learning in Auto-encoder Novelty Assessment, in *proceedings of the 2002 International Joint Conference on Neural Networks, 2002 IEEE World Congress on Computational Intelligence*, May 12-17, 2002, Honolulu, pp. 2878-2883.

Balducelli C., Bologna S. (2001), Agent Based Architectures to Improve Survivability of Large Complex Critical Infrastructures, in *proceedings of the TIEMS2002 Conference,* Waterloo (TORONTO), Canada – May 14-17, 2002

Schneier, B. (1999), Attack Trees: Modelling Security Threats, *Dr. Dobb's Journal*, December 1999, ISSN 1044-789X.

Andrew P. Moore, Robert J. Ellison, Richard C. Linger, Attack Modelling for Information Security and Survivability, Mar 2001, www.cert.org/archive/pdf/01tn001.pdf

Schneier, B. (2000), Secrets and Lies: Digital Security in a Networked World, John Wiley & Sons Eds., August 2000, ISBN: 0471253111