Policy Based Management for Critical Infrastructure Protection

Gwendal Le Grand, Franck Springinsfeld, Michel Riguidel
GET/Télécom Paris, 46 rue Barrault, 75634 Paris Cedex, France
{gwendal.legrand, franck.springinsfeld, michel.riguidel}@enst.fr

This work has been realized in the context of the ACIP project [1], funded by the European Commission in the context of the "Information Society Technology" Programme.

Abstract:
Our current societies are fully dependent on large complex critical infrastructures (LCCIs). These LCCIs are large scale distributed systems that are highly interdependent, both physically and in their greater reliance on the information infrastructure, which logically introduce vulnerabilities. Failures, accidents, physical or cyber attacks can provoke major damages which can proliferate by cascading effects and then can severely affect a part or the whole society. This article aims at providing a solution for enhancing dependability and survivability of such systems by developing new models, methodologies and tools. An assessment of vulnerabilities of existing infrastructures is achieved using canonical architectures. Then, we extract abstract security policies that are implemented using a policy based management approach.
Keywords – critical infrastructure protection, policy based management.

Introduction and Background

The economy and security of Europe are increasingly dependent on a spectrum of critical infrastructures, which can be broadly grouped in the following five domains:

- Information and Communications,
- Energy (Electrical Power and Oil and Natural Gas Production and Storage),
- Transportation
- Banking and Finance
- Vital Human Services (Water and Food Supply Systems, Emergency Services, Government Services)

All the above critical infrastructures are highly interdependent, both physically and in their greater reliance on the information infrastructure. This trend has been accelerating in recent years with the explosive growth of information technology and shows no sign of abating.

"A bounded system is one in which all of the system's parts are controlled by a unified administration and can be completely characterized and controlled". An unbounded system is "characterized by distributed administrative control without central authority, limited vision beyond the boundaries of local administration, and lack of complete information by the network". "An unbounded network can be composed of bounded and unbounded systems connected together in a network". [2]
Because of their interdependencies and their increasing reliance on open systems such as the public switch telecommunications network and the Internet, all these critical infrastructures constitutes an unbounded system where faults may occur and proliferate in a severe way and where security represents a real challenge and requires new methodologies and tools.
Potential threats to the normal functioning of these infrastructures are both natural ("Murphy's Law and Mother Nature") and man-made. Individual outages can be serious enough, but this growing degree of interconnectedness can make possible a whole new scale of synergistic, nonlinear consequences.

1.1 Description of a Critical Infrastructure (CI)

A CI comprises three components that interact:

- **Administration (Adm)** that includes the human (decision-makers and workforce), economic, regulation, etc. aspects,
- **Physical (Phy)** corresponds to the material aspect of the resource supply system,
- **Information System (IS)** corresponds to the information system of the infrastructures (e.g. Supervisory Control And Data Acquisition (SCADA) …), information technology for business systems, e-commerce …

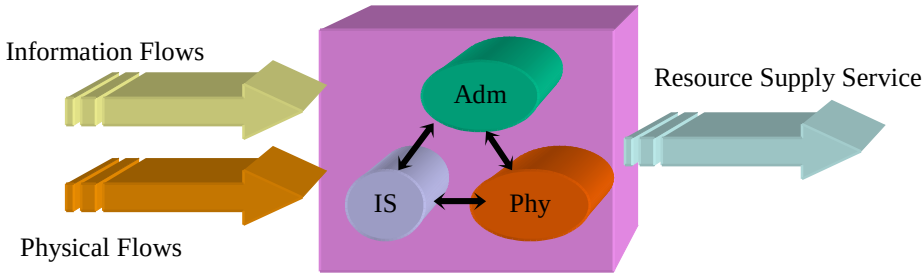


Figure 1. Description of a CI

1.2 Key Properties of CIs

There are five key properties of critical infrastructures that must be understood before one may formulate a methodology for building a CI system to support their protection.

1. CIs exhibit strong, mutual **dependencies**. For example, most critical infrastructures depend on the electric power sector and the information and communications sector.
2. Although they may provide a public and survivability service, CIs are largely **owned and operated by the private sector.** Any methodology must thus rest upon a trusted partnership wherein the public and private participants have incentives to exchange appropriate, timely levels of information.
3. CIs are becoming increasingly **IT-dependent** in order to accelerate information circulation and reduce the operating cost. For example, deregulation in the electric power sector is encouraging competition, prompting efforts to reduce costs and personnel. As a consequence, this sector is introducing greater levels of automation at multiple levels that has the undesired side effect of making these sectors more vulnerable to cyber attacks. For instance, Supervisory Control and Data Acquisition Systems (SCADAS) are part of this automation process and basically correspond to networks of computers used to monitor electricity across lines or even flows of oil in pipelines in gas & oil sector.
4. As a consequence of the increasing globalization of commerce, CIs are becoming **international.** Therefore, it is probably inadequate to develop a purely national CI system for protecting the critical infrastructures. It has to become European, if not international.
5. At the same time that the information technology revolution has led to substantially more interconnected infrastructures with generally greater centralized control, the advent of "**just-in-time**" business practices has reduced margins for error in infrastructures. In addition, the trend toward deregulation and growth of competition in key infrastructures has understandably eroded the willingness of infrastructure participants to pay for spare infrastructure capacity that could serve a useful "shock absorber" role in cushioning key infrastructures from failures elsewhere in the economy. Furthermore, the growth of mergers among infrastructure providers has led to further pressures to reduce spare infrastructure capacity as managers have sought to wring costly "excess" costs out of merged companies.

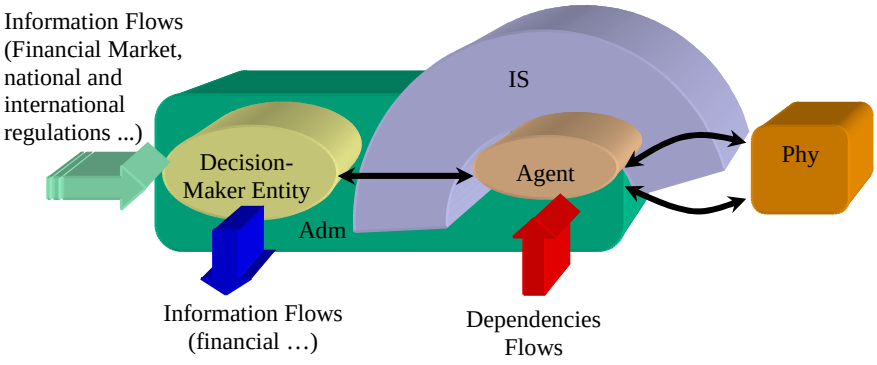


Figure 2.Description of the Administration component

1.3 CI hierarchies

CIs can be classified hierarchically. A possible classification is represented in Figure 3. Two global levels can be extracted from this classification:

- the high level corresponds to the **business level** and includes Government, Economy, Society (people, environment, life of animals, immaterial issues), and the Compound of Critical Infrastructures.
- the low level corresponds to the **technical level** and includes telecommunication, energy, transportation …

1.4 Vision and Goal

It appears that no clear and scalable security model exists for big critical infrastructures. Existing models of such systems are vague and there are no methodologies for understanding the behavior of complex systems. Especially, more work has to be done in defining a **security architecture** that takes into account cascading effects, inter and intra dependencies, distributes the intelligence in the infrastructure to increase its robustness, etc. Thus, current methodologies cannot easily be applied to CIP, may miss the right level of granularity, or cannot easily be incorporated in other models, etc. Modeling and analysis of these systems is a challenge because of their large-scale, non-linear, and time-dependent behavior. We wish to "investigate available methodologies and requirements to model and investigate criticality, vulnerability, and interdependency, and design-measures of critical infrastructures at a technical level."
Important steps have already been taken on individual infrastructures. For example, electrical networks have accurate models, but the issue of interdependent and cascading effects among infrastructures has received much less attention. This situation calls for more efforts and the development of new methods and tools.

The critical infrastructures should be **trustworthy** and **resilient** (able to provide the required level of performance under a variety of conditions). To achieve this, they should have the ability to absorb intentional or unintentional outages with minimal impact on their ability to deliver needed levels of service, both directly to consumers and to the other infrastructures that depend upon it. This would be a daunting challenge even if the technology of these critical infrastructures were static. As we know, the technology embedded throughout the western economy is undergoing a continuous and profound

transformation. Accordingly, we will seek support for the development of technologies that will counter threats and reduce vulnerabilities in those areas having potential for causing significant security, economic, and/or social impacts.
Such a robust set of critical infrastructures would be protected from hostile acts and natural outages that would significantly diminish the abilities of:

- European countries to perform essential security missions and to ensure public health and safety.
- governments to maintain order and to deliver minimum essential public services.
- businesses to ensure the orderly functioning of the economy and the delivery of essential information and communications, energy, financial, transportation, and other services.

The vision is that any interruptions or manipulations of these critical functions would be brief, predictable in impact, infrequent, manageable, geographically isolated, and minimally detrimental to the welfare of Europe.

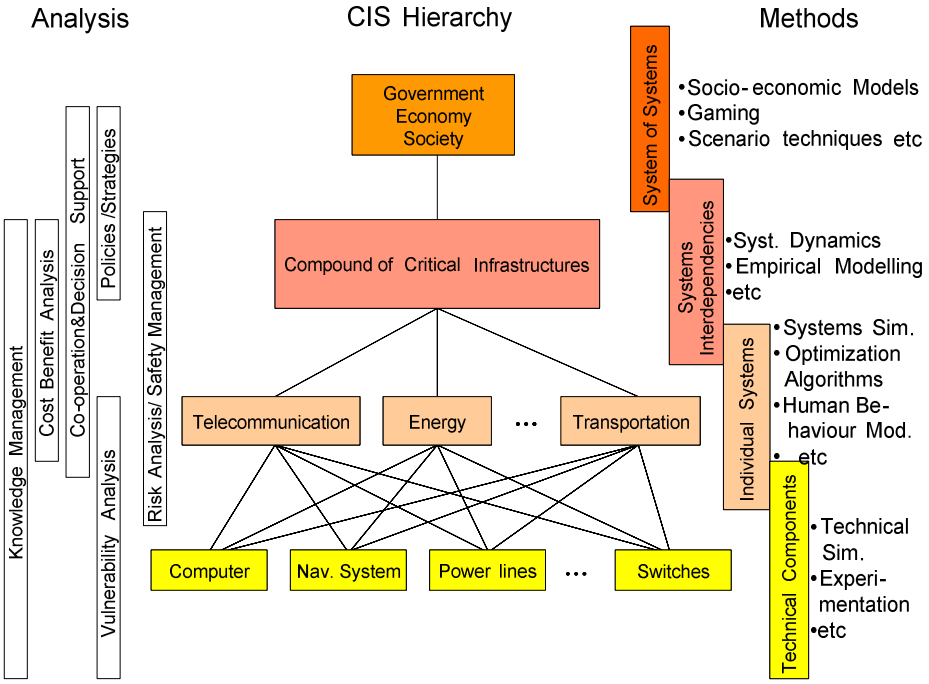


Figure 3. Hierarchy of Infrastructures

This document focuses on proposing a security model for the main CIs. We want to provide both inter and intra CI models. We wish to define a model that can define survivability, interdependencies, and dependability policies based for example on the properties of the individual entities and of the compound system (the system and interconnections between entities of the system). Moreover, the models should be exhaustive and take into account faults, intrusions, natural events and both voluntary and unintentional human behaviors.

The remainder of this paper is organized as follows: we first supply basic considerations about the critical infrastructures field. Then LCCI-related definitions are presented in section 2. It results in a collection of requirements for models to investigate vulnerability of single components of a CIS and to model impacts, disruptions, interdependencies, and cascading effects within each of the main CIS in section 3. Section 4 presents the characteristics of the security models and section 5 proposes an approach for CI protection, based on Policy Based Management.

2 Definitions

In this section, we introduce a number of definitions for CI analysis.

Dependency: “A linkage or connection between two infrastructures, through which the state of one infrastructure influences or is correlated to the state of the other.” [3]

Interdependency: “A bidirectional relationship between two infrastructures, through which the state of each infrastructure influences or is correlated to the state of the other infrastructure. More generally, two infrastructures are interdependent when each is dependent on the other.” [3]. Therefore, interdependency can be considered as the association of two different dependencies.
Four types of interdependencies can be identified [3]:

- **Physical**: Two infrastructures are physically interdependent if the state of each is dependent on the material output(s) of the other. For example, it is the case when the material output of one infrastructure is used by another

- **Cyber**: The state of an infrastructure depends upon information transmitted through the information infrastructure. e.g. use of electronic information and control systems
- **Geographic**: Infrastructures are geographically interdependent if a local environmental event can create state changes in all of them e.g. infrastructures are co-located in a common corridor
- **Logical**: Others e.g. infrastructures are linked through financial markets

An interdependency may be qualified according to several criteria such as: direct/indirect (one of the two infrastructures do not have direct influence over the other but through one or several others infrastructures), loose/tight, linear/complex, etc.

Therefore, several types of risks due to dependencies and interdependencies [2] exist:

- **Cascading Effect** : "A disruption in one infrastructure causes a disruption in a second infrastructure"
- **Escalating Effect** : "A disruption in one infrastructure exacerbates an independent disruption of a second infrastructure (e.g., the time for recovery or restoration of an infrastructure increases because another infrastructure is not available)"
- **Common Cause** : "A disruption of two or more infrastructures at the same time is the result of a common cause (e.g., natural disaster)"

Dependability

"Dependability is a property that a system might have and is usually stated as a set of requirements with which the system has to comply." [4]

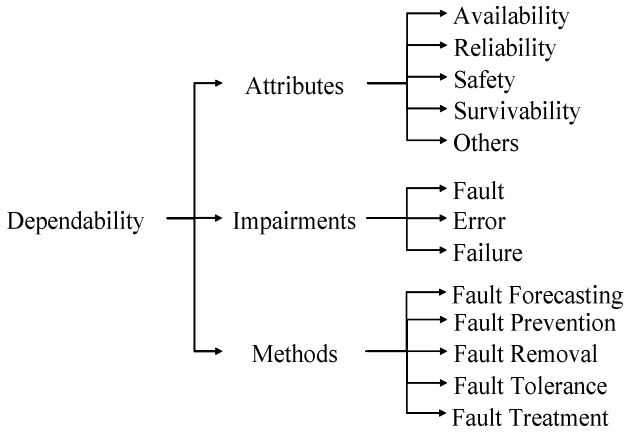


Figure 4. The Dependability Tree inspired from [5]

- **Attributes**

Dependability concerns several aspects. In a non-exhaustive way we can mention: [2]

- "[…] the reliability of a system, R(t), is defined to be the probability that the system will meet its requirements up until time *t* when operating in a prescribed environment."
- "[…] the availability and integrity of a system, A(*t*), is the probability that the system will be operating at time *t*."

- **Impairments**

The impairments sequence for a given component is: [7]

- Fault: "the adjudged (hypothesized) cause for an error"
- Error: "incorrect system state"
- Failure: "component no longer meets its specification"

Further in this article we will broaden the term "fault" to all events that can potentially provoke damages in a system.

- **Methods**
  - **Fault Forecasting:** "how to estimate the presence, creation and consequences of faults" [5]
  - **Fault Prevention:** "how to prevent the occurrence or introduction of faults" [5]
  - **Fault Removal:** "detecting and removing faults before they can cause an error" [7]
  - **Fault Avoidance:** "Fault Prevention + Fault Removal" [7]
  - **Fault Tolerance:** "how to provide a service capable of or implementing the system function despite faults"[5]
    - ♦ Error Detection
    - ♦ Error Processing: "mechanisms that remove errors from computational state" [7] (Error Recovery or Error Compensation)
  - **Fault Treatment:** "preventing faults from re-occuring" [7]
    - ♦ Fault Diagnosis: "determining cause(s) of the error(s)" [7]
    - ♦ Fault Passivation: "preventing fault(s) from being activated" [7]

Survivability
Survivability is an important notion in unbounded networked systems where traditional security measures cannot be applied.
It is defined as "the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents" [2].
Essential services are "the functions of the system that must be maintained when the environment is hostile, or when failures or accidents occur that threaten the system. […] The capability to deliver [these] essential services (and maintain the associated essential properties) must be sustained [by a critical system] even if a significant portion of [this] system is incapacitated. […] Key to the concept of survivability, then, is the identification of essential services, and the essential properties that support them, within an operational system." [2].
"To maintain their capabilities to deliver essential services, survivable systems must exhibit four key properties […]":

- Resistance to attacks
- Recognition of attacks and the extent of damage
- Recovery of full and essential services after attack
- Adaptation and evolution to reduce effectiveness of future attacks

Criticality and degree of criticality
The ***criticality*** of a system refers to several aspects of this system such as its sensitivity (i.e. its capability to move from an equilibrium state to a stressed state because of disturbances), impact of fault (impacts on the system, impacts on population …)….
Close to the notion of *criticality*, the ***degree of criticality*** is an indicator of the state of a system and depends on the degrees of criticality of the components of this system, its morphology, the time (e.g. congestion peak in certain moments in certain locations within a road transport system) and so on.

Risks
The different **risks** a system has to cope with can be categorized as follows [2]:

- **Failure**: "*Failures* are potentially damaging effects caused by deficiencies in the system or in an external element on which the system depends.[…] *failures* are typically internally generated events."
- **Accident**: "*Accidents* describe a broad range of randomly occurring and potentially damaging events such as natural disasters. *Accidents* are often externally generated events […]"
- **Attack**: "*Attacks* are potentially damaging events orchestrated by an intelligent adversary."

These risks logically cause a modification of the degree of criticality, which will be propagated by interdependencies from the location of the fault within the system.

Thus, the consequences of these faults for the system are one or several of the following: spying, performance degradation, corruption, "drying" of a flow (= the fault entails the disappearance of the flow), destruction, etc.

Therefore, interdependencies and Supply System Morphology can entail **Fault Proliferation Phenomena**.

3 Requirements of the system

The goals of a security management model are to be able to foresee the development flaws, detect anomalous behaviors to proactively manage the system in order to prevent serious problems, install prevention measures, and reactively control the system by making adjustments in response to changes (that may be sudden as when following an attack) within the system or its environment.
Even if it is almost impossible to prevent attacks 100 percent, it is really important to be able to act quickly within the system to stop a potential proliferation of the problem. Consequently, two correlated works of modeling can be distinguished: one concerning CIs and one concerning security management.

Therefore, the basic requirements of the system are motivated by the following security functional requirements: prediction and scenario simulation (development, proactive management, etc.), prevention, monitoring (global view, reactive and proactive management, real-time), distributed intelligence and autonomy.

In [1], requirements for CI modeling and security management models have been identified. They are presented hereafter.

3.1 CI Model requirements

Dependencies: the study of dependencies and interdependencies is important in many areas like attack scenarios, incident and accident analysis, and more. Current research has made progress in this field and existing initiatives can be used for further research (eg. DSOS project, PreDICT, NIRA, and more).
Hierarchical levels: Infrastructures can be represented by their hierarchical, organizational structure, starting at the system-of systems level and moving down to the levels of single infrastructures, systems, subsystems, units/components and parts. This will assist the scalability of the model.
Canonical architectures, development of formal cases: Security models should be adapted to the abstract morphology of the system that corresponds to its canonical architecture. The canonical architecture theory is described in section 4.1.1.
Model mapping: existing models should interoperate.

3.2 Security Management Model requirements

Definition of required levels for CIs and CI systems: The current research covers different aspects of vulnerability and risk analysis (eg. OCTAVE, CORAS, PreDICT, NIRA, etc.).The next steps will be to develop a practical and ready-to-use set of criteria that will define for each CI, and CI systems, what is the adequate level of security needed in order to set uniform levels of security, and minimum requirements for all EU CI systems.

Survivability: Survivability is one of the crucial aspects of today's CIs. Current models of achieving survivability in CI systems include emergent algorithms (eg. EASEL - Emergent Algorithm Simulation Environment and Language), self healing systems (Safeguard), and more.

System security assessment methodologies: Current research contains many vulnerability and assessment methods (eg. OCTAVE, CORAS), but there is little use of these methodologies by IT operations staff. The products used often include software tools that address specific IT platforms, and lack the security assessment ability. Practical and business-related methodologies (such as the End to End Security Assessment model (EESA)) that can bridge this gap are required.

Early warning models: Prevention is mostly achieved today by preventive implementation of security elements. This is usually implemented within the design stage of systems. The next required step will be to provide technical-operational tools that will be able to address the prevention stage of the incident life-cycle, with early warning abilities (within the operational stage of systems).

Incident response: Different monitoring initiatives exist (Examine, Depaude and others). Monitoring systems provide information about current status of different aspects in the monitored systems. In case of attack, there is need for models that will be able to react to different scenarios, based on policy databases or based on innovative methods and algorithms.

System management models: Current systems provide local and specific solutions for monitoring, alerting and management of security assets. Further technological research is required in order to provide 'smarter' systems that will be able to consolidate different events (network traffic, user management, application events, etc.) and build a whole security picture, thus providing useful decision support information. These can be used for audit purposes, system assessment, and incident response.

Distributed intelligence: Distributed intelligence means that the system is able to take decisions locally by its own in order to act as fast as possible. This objective can be partially reached by a hierarchical structure. Distributing decision taking in the CI increases its reactivity. However, this policy also distributes the number of entities that have control over the CI and this increases the number of places where a wrong decision might be taken. When a decision is taken locally, this decision should be reported to a central entity (at a higher level in the hierarchy) that is able to assess and pre-empt the local decision.

Distributed autonomy: in order to achieve security in the previously-described architecture, confidence models should be defined. For example, these models may include a historical component in which an entity assesses one of its children's decisions according to its past behavior. This model is derived from human behaviors. The flaw of this model resides in the fact that an attacker may try to be trusted in order to build a dangerous attack. This is why the hierarchy may always pre-empt and isolate a component of the infrastructure.

Architecture models: The IT architecture is a crucial component in the design of IT systems. Insertion of adequate security components, segmentation of networks, and other security related architecture issues, should be systematically addressed. Based on sector needs, on dependability and survivability issues, and other specific criteria, a clear set of architecture requirements should be defined for CI systems.

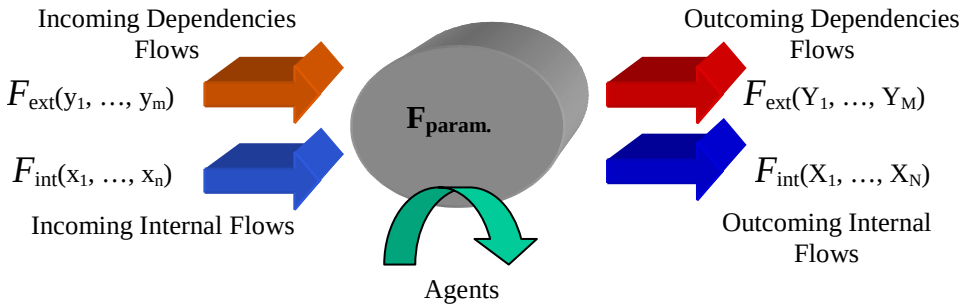
Policies/Strategies: Policies and strategies can provide a certain level of standardization in the different CIs through the EU, and provide a basis to a more uniform security management level and overall security level.

4 Security models

4.1 Technical level

An infrastructure consists of a set of functional entities (source, transformation, relay, transport, user) and a family of links. The links correspond to flows between functional entities and may be oriented links (eg. water flows) or non oriented links (e.g. information) according to the nature of the supplied resource. Therefore, an infrastructure may be modeled as a non connected graph in which the set of summits (the functional entities) is not convex.

Functional Entity



$$(F_{int}(X_1, \ldots, X_N), F_{ext}(Y_1, \ldots, Y_M)) = \mathbf{F_{param}}\ (F_{int}(x_1, \ldots, x_n), F_{ext}(y_1, \ldots, y_m))$$

Figure 5. Model of a functional entity

A functional entity may be modeled by a function F_{param}, as represented on Figure 5. This function is a specific function of flow treatment of the entity. It is parameterized by external agents or by information flows. Moreover, the function can correspond to one or several basic roles: source, transformation, routing, transport, consumption. Each entity can have internal modules. A module is an internal sub-entity that fulfills a basic role within the functional entity only for the benefit of the functional entity. In this model, interdependencies and intradependencies are modeled using two functions Fint (family of flows coming directly from another summit of the CI) and Fext (family of dependencies flows) that interact with F_{param}. Fext may comprise information flows, flows of raw materials which are not handled by the CI, functioning flows (e.g. electricity), risk flows, others.
Each flow may have the following characteristics (oriented/non oriented, material/immaterial, unicast/multicast/broadcast, etc.) and may be of one of the following types:

- staff (necessary for the correct functioning of F),
- Equipment, hardware, software, … (constituents of F),
- Information, data (supervision, monitoring, operation, …)
- Main resources (directly connected with the resource supplied by the CI)
- Secondary resources (inputs necessary for F)
- Risk flows (geographical interdependency flow, …)

4.1.1 Typology of system morphologies -- Canonical Architectures

Large infrastructures are different from classical systems. Classical systems are convex architectures that are **salient** into the environment (e.g. plants, campuses), but large infrastructures have a different morphology: they are pregnant into an environment (like nets, branches, etc.) and occupy a field. The types of flows considered vary from one infrastructure to another: in the telecom domain, traffic is examined whereas the electricity domain is concerned with electricity flows and the distribution domain with "just in time" flows. Large infrastructures can thus be represented by an abstract morphology that is characterized by three components (Figure 6): *communication* (the system coupling with the environment -- laws of physics, syntax, semantics), *connectivity* (the system's internal morphology -- for example a bus or a point to point architecture), and *infiltration* (the coupling with the system -- mimesis or camouflage). Specific security policies can then be derived from applied to **canonical architectures** extracted from an abstract morphology.

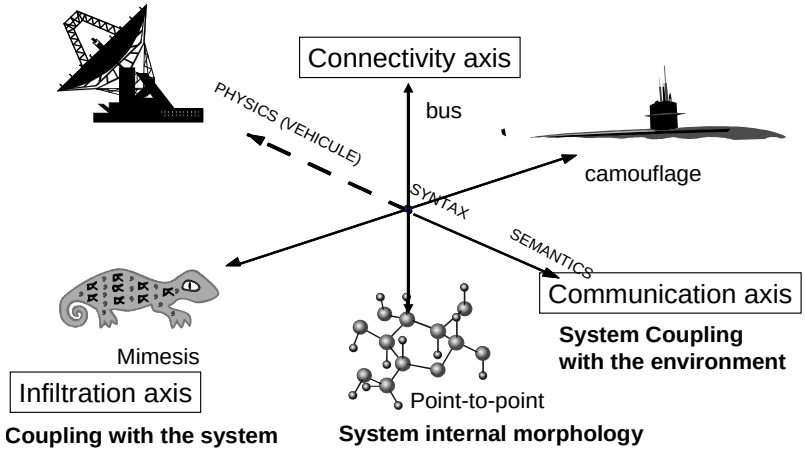


Figure 6. Attack Defense System

Whatever hierarchical level we look at - from a set of functional entities to the compound of CIs – and even in the organizational schemes of CIs, it is possible to extract a limited set of canonical architectures or patterns or morphologies which have intrinsic properties. These intrinsic properties allow to develop specific, standard and abstract security measures for each canonical architectures depending on such parameters such as the context of the fault and the nature of the flow.

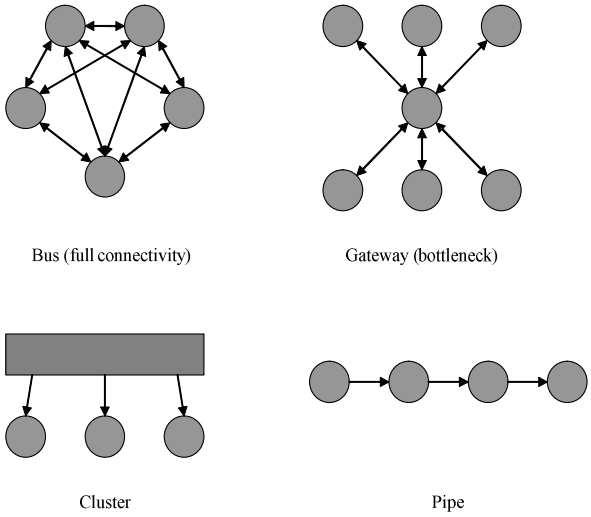


Figure 7. Examples of relevant canonical architectures

Figure 7 displays a non exhaustive set of relevant canonical architectures. Grey circles and rectangles represent physical or management entities, or compound of these entities do not necessarily belong to the same CI. Therefore the canonical architecture theory has two main characteristics:

- **Canonical architectures are flow-oriented**. Links in the figure can refer to both intra- and inter-dependencies flows.
- **Canonical architectures are generic.** Entities of a canonical architecture at a certain level can be a compound of canonical architectures at a lower level. This implies a hierarchical modeling with a relevant granularity.

By combining specific properties of canonical architectures with a certain amount of parameters such as the nature of the flow, the type and location of the fault, vulnerabilities of existing architectures can be assessed and so security solutions can be developed to design dependable and survivable architectures.

4.1.2 Temporality and cycles

The characteristic times of an entity can be classified as follows:

- Propagation time of the flows through the entity: it depends on the nature of the flow, the function F and the state of the entity,
- Self-sufficient times for the different flows: they depend on storing mechanisms, cycles of the entity (e.g. sporadic needs of the resource supplied by a flow), the pattern of the dependencies, etc.
- Development time: this is the time to build, install and operate the entity. This time depends on the geographic location, the nature of the entity, the cycles of construction materials, …
- Others

An entity may also have cycles. A cycle is a typical chain of phases for an entity (e.g. the different phases of the life of an entity -- conception, 'manufacturing', installation and deployment, operation and maintenance, obsolescence, modernization or destruction), or may express the existence of periodicities for incoming flows (some entities have one (or several) characteristic cycles over one (or several) time scale --e.g. the ring road of a city). The analysis of these cycles can reveal vulnerabilities (e.g. the existence of congestion peaks).

4.1.3 Modeling the effects of events

Events in the real physical system can be modeled locally as changes at the level of a functional entity. Inside the functional entity, this concerns the alteration of the treatment function F characteristic of the entity and the modification of the parameters of the function F. At the level of flows, it concerns the modification of the specific values of one or several flows and the activation of a critical/risk flows.

The consequences of these events can be modeled as changes at the level of the outcoming flows of the functional entity. Possible consequences may be:

- Spying: the appearance of an outcoming parasitic information flow
- Performance degradation and corruption: modification of the specific values of one or several outcoming flows
- Drying of a flow: disappearance of an outcoming flow
- Destruction: disappearance of all flows

4.1.4 Business level vs. technical level

Figure 8 shows the close relationship between the 2 global levels defined in the beginning of this document. For example, when a fault occurs at the technical level, it modifies the degree of criticality of the system and so will affect the business

level that can in turn react over the technical level according to its security policy (Dependability). Another fact is that the interdependencies are spread on the 2 levels: Logical interdependencies are usually located at the business level, Physical and Geographic interdependencies are at the technical level, and Cyber interdependencies are at both levels. Therefore, modeling aspects cannot be restricted only to the technical level without considering the relationship with the upper level.

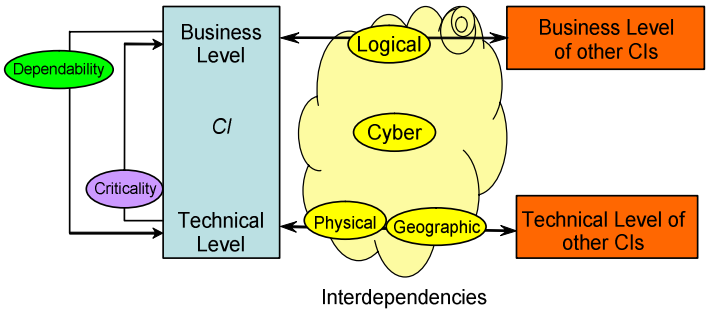


Figure 8. Business/Technical levels relationships

For modeling, a CI (and even a set of CIs with its pattern of interdependencies) could be characterized as a very large-scale network system, in which a disturbance somewhere in the system can affect everything else in the system. This network, if exposed to a non-trivial disturbance, can no longer respond linearly and either a new equilibrium may not exist or it could be reached only by control actions. Thus, there is a need for a global view of the entire network and a possibility of a quick action over the nodes of this network.

4.2 Trust Model & Crisis Management Model

Two security models can be defined:

- The Trust Model in a system with a low degree of criticality
- The Incident Response / Crisis Management Model in a system where a fault has occurred.

Figure 9 and Figure 10 define these two models according to the previously defined dependability properties and survivability properties.

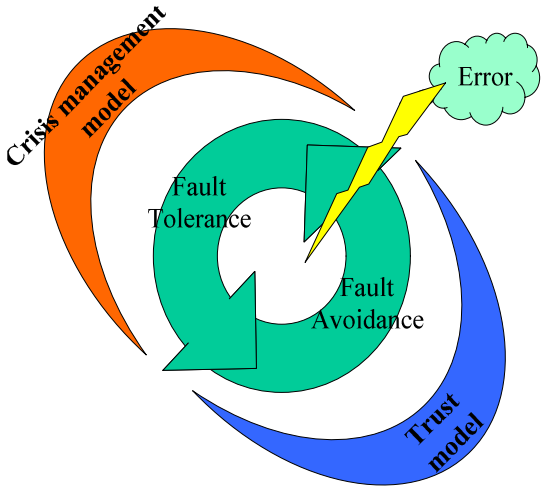


Figure 9. Crisis management and trust models (dependability view)

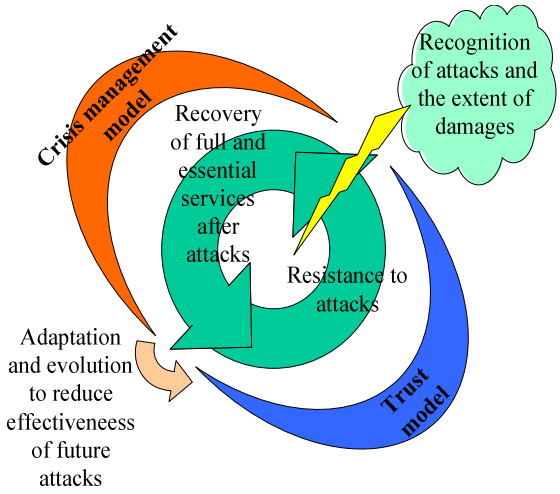


Figure 10. Crisis management and trust models (survivability view)

Trust models correspond to Fault Avoidance (Fault Forecasting, Prevention and Removal) or mechanisms to resist to attacks. Some trust model examples are listed hereafter:

- Fault Avoidance or Resistance to attacks can be achieved by **protection** (based on access control), **dissuasion** (based on reprisals), **relation** (based on negotiation), or **semblance**.
- Trust models are related to crisis management models;
- Forecasting and Detection are related to the **analysis of faults undertaken by experts, surveillance, and registration**.

Crisis management models correspond to fault tolerance or recovery of full and essential services mechanisms. Some examples may be:

- **Inaction** (« wait and see »)
- **Filtering** or switching **to a summit with redundancy of functionality** in order to mask a fault

- **Resignation** and therefore performance collapse, or stopping **critical outcoming flows (**which aims at stopping an epidemic process), when the fault cannot be masked

Trust models and crisis management models are closely related. Actually, trust models based on **information** or on a **diminution/increase in the number of dependency flows** may participate to the mitigation of a fault. Moreover, Fault Treatment entails the **adaptation and evolution to reduce the effectiveness of future attacks**.

The implementation of these models for one system is fully conditioned by the security policy which depends on the dependability of this system (safety …) and the costs (cost of redundancy …).

In the end, the goal of the security model is to:

- *Reactively* control the system by making adjustments in response to the changes within the system or its environment
- *Proactively* manage the system, for example, by detecting trends or anomalous behavior allowing action to be taken before serious problems arise

Therefore, some performance management, fault management, configuration management, security management, etc. tools are needed.

5 Policy Based Management (PBM) Approach

5.1 Definitions

This section defines the terminology we use in the PBM approach.

Policy Based Management (PBM) allows a dynamic and global network management. It is g*lobal* since a network is modeled as a state machine in which the union of all local device states gives the global network state. Dynamicity is provided through policies. A network state change provokes a reaction to the event using a bidirectional management

From [6], we define the following terms:
"An **information model** is an abstraction and a representation of the entities in a managed environment. This includes the definition of their attributes, operations and relationships. It is independent of any specific type of repository, software usage, or access protocol. An information model can be thought of as the defining document that is used to model all of the different managed objects in a managed environment."
"A **data model** is a concrete implementation of an information model in terms appropriate to a specific type of repository that uses a specific access protocol or protocols. It includes data structures, operations, and rules that define how the data is stored, accessed and manipulated. A data model can be thought of as the implementation of a subset of the data defined in the information model in a particular repository."
"A **model mapping** is a translation from one type of model to another type of model. Model mapping changes the representation and/or level of abstraction used in one model to another representation and/or level of abstraction in another model. We can use model mappings to translate between an information model and a particular data model, or between different data models."
"**Policy** is a set of rules that are used to manage and control the changing and/or maintaining of the state of one or more managed objects."
"A **policy rule** is an intelligent container. It contains data that define how it is used in a managed environment as well as a specification of behavior that dictates how the managed entities that it applies to will interact. The contained data is of four types: (1) data and metadata that define the semantics and behavior of the policy rule and the behavior that it imposes on the rest of the system, (2) a set of events that can be used to trigger the evaluation of the condition clause of a policy rule, (3) an aggregated set of **policy conditions**, and (4) an aggregated set of **policy actions**."
"**Policy-Based Management (PBM)** is defined as the usage of policy rules to manage one or more entities. […] PBM controls the state of the system and objects within the system using policies. Control is implemented using a management model such as a finite state machine. It includes installing and deleting policy rules as well as monitoring system performance to ensure that the installed policies are working correctly. PBM is concerned with the overall behavior of the system, and adjusts the policies that are in effect based on how well the system is achieving its set of policy goals."
"**Configuration management** systems are built to understand and manage all of the functions, and hence the entire configuration, of a device."
"**Provisioning** includes all actions that are necessary to activate and manage a service – everything from the time a customer orders a service to the time when that service is activated. In this sequence, configuration is one of many tasks that must be completed in order for a service to be activated."

5.2 Policy Based Network

The PBN model (represented on Figure 11) allows to automatically monitor and reconfigure large numbers of devices to conform to policy parameters. Policies are defined in a high-level language and some mechanism automatically translates them into the various low-level commands that various devices understand. In order to control the devices, policy mechanisms have to be linked to an existing repository of user and resource data. Not only can management tools use directory data to determine policies and locate resources required for the enforcement of those policies, but the tools can also publish information about themselves in the directory. The directory therefore becomes the main point of control on the network, with policy management tools acting as consoles for entering policy definitions and translating them into objects

that get published in the directory. The repository's scope integrates all infrastructure services. This helps eliminate a lot of human mistakes, which can come either from error or from not knowing the relevant policy.
PBN provides a client-server model for policy queries and responses. This scheme is designed to be extensible to all types of policies. A policy server (Policy Decision Point, or PDP) communicates with its clients (Policy Enforcement Points, or PEPs). PEPs send requests, updates, and deletes to a PDP. The PDP then returns decisions to the PEPs. These messages must be authenticated and sent over a secure channel between the PEP and the PDP given the fact that policy servers could represent a powerful means for intruders to create massive disruptions.

The mechanism is stateful. This is critical because service requests from a client PEP must be retained by the PDP until they are explicitly deleted. Without stateful communications, policies could not respond to existing conditions, which would mean that policies could not respond to environmental variables. The PDP may also push configuration information to the PEP and then remove such state information from the client when it is no longer applicable. Two Management Models exist: **Outsourcing** and **Provisioning**. Outsourcing consists in sending data to update the PDP (which in turn may update other PEPs). Provisioning concerns the installation of a policy by the PDP in a PEP.

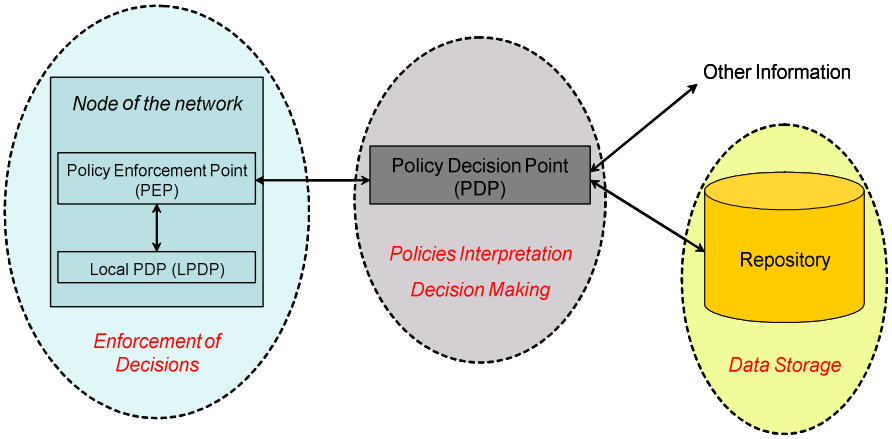


Figure 11.Policy Based Network

In addition, Local Policy Decision Points (LPDPs) let policy/state data and requests be offloaded to subsidiary policy servers closer to the PEPs they control. LPDPs, however, report all policy decision events to the central PDP, which can override them at any time.

5.3 Architecture

5.3.1 A hierarchical PBN

Figure 12 represents a hierarchical PBN fitted to the CI environment. We basically identify two levels of hierarchy in the network in which the domains we identified in the framework can be mapped. At the upper level, the Compound Managing Entity corresponds to a PDP, whereas the CI Managing Entity is a PEP that can be considered as a Compound Managing Agent. At the low level, the Compound Managing Agent is a CI Managing Agent's PDP. **The low level itself can be made up of several levels of hierarchy according to the granularity we wish to apply to the model.**

5.3.2 Information Model

In the context of Critical Infrastructure Protection it is important to use an information model because there is more than just one standard representation of data. The data defines semantics and behaviour of, and interaction between, managed entities. This system also needs to be federated and layered. An information model comprises attributes that define the basic characteristics, methods that define the access to attributes and the business and system operations, the relationships, and the behaviour of the entities.

5.3.3 Policy

Every configuration change, no matter how simple or how sophisticated, has an underlying set of business rules that govern its deployment. Therefore, policies seem to be adapted to manage such large information systems. A policy rule can be defined as a set of policy conditions and a set of policy actions. This allows the application of **Policy Based Management (PBM)** to these critical infrastructures. PBM is defined as the usage of policy rules to manage one or more entities. Therefore, one or several management entities control the state of the system and objects within the system using policies.

We can identify two different views for the application of a policy on the technical level: device-specific policies and generic policies. Actually, the second approach seems more appropriate since it is important to separate the modelling of policies from the modelling of device mechanisms (e.g. in the case of a fault in a system (Outsourcing Mode), there are standard policies (e.g. Actions = standard security measures) only depending on few parameters like the local morphology of the network (cf. existence of a set of **canonical architectures**), the type of faults, …). Therefore we suggest the utilization of device-independent policy models since this solution is more flexible. For example, in a company, a generic policy could

consist in blocking the outgoing web-based traffic, but the actual implementation of this functionality consists in deploying filtering in firewalls and other heterogeneous network equipment of the company that may use different parameters to execute the same function. The policy is thus very high level and has to be instantiated in the devices that are actually part of the infrastructure. The crucial point that has to be considered in order to achieve resides in the definition of a policy continuum and coherency.

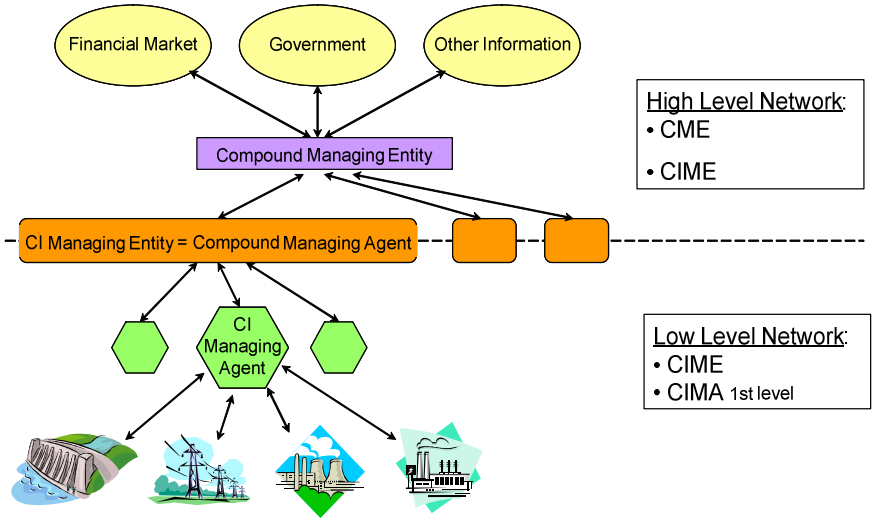


Figure 12.Architecture

5.4 Modeling approach

The models presented previously typically correspond to an Object Oriented (OO) approach. Thus, we recommend the use of an OO Information Modeling focused on describing network elements and services, and how they are related to each other. This model assumes that the network is modeled as a State Machine. A network can be represented as a set of sub-networks which are sets of functional entities (the number of levels depends on the granularity that is targeted). Each functional entity is then modeled as a state machine and so, each hierarchical upper-level entity can be modeled as a state machine. In this OO model, classes and relationships are used to model: the state of an entity, the settings to be applied to an entity that either maintain an entity's state or move the entity to a new state, and the policies that control the application of settings. This OO modeling may be done with UML. UML has three main components:

- The English description of the use case and the entity design
- UML class diagrams
- A Data dictionary that defines the syntax, semantics and use of classes, attributes and relationships

6 Conclusion and future work

We defined a modeling architecture that can fit the requirements of any CI. This architecture is based on a hierarchical PBM architecture. An assessment of vulnerabilities of existing infrastructures is achieved using canonical architectures. Then, we extract abstract security policies that are implemented using a policy based management approach. In the future, we will implement this approach in a Java and UML based application. Moreover, we will investigate in more detail the confidence relationships between the entities) of the Policy Based Management Model (at the same level of the hierarchy and at different levels of the hierarchy.

7 References

[1] ACIP project homepage, http://www.iabg.de/acip/

[2] Robert J. Ellison, David A. Fisher, Richard C. Linger, Howard F. Lipson, Thomas A. Longstaff, Nancy R. Mead. *Survivability: Protecting Your Critical Systems*. IEEE Internet Computing, November/December 1999 (Volume 3, No. 6)

[3] Rinaldi, Steven M., James P. Peerenboom, and Terence K. Kelly. *Critical infrastructure interdependencies*. IEEE Control Systems Magazine, December 2001.

[4] J. Knight and K. Sullivan. *Towards a Definition of Survivability*. Proceedings of the 3rd Information Survivability Workshop (ISW), Boston, MA, October 2000

[5] David Powell, Yves Deswarte, *On Dependability Concepts with respect to Deliberately Malicious Faults*, STCF 2001, Florianópolis/SC, 5-7 March 2001

[6] John Strassner. *A new paradigm for network management: Business Driven Device Management*. SSGRR 2002s July 29 - August 4, 2002.

[7] David E. Bakken. *Fault Tolerant System Foundations*. CptS/EE 562, Spring 2002.