# Dependency Analysis for Critical Infrastructure Security Modelling: A Case Study within the Grid'5000 Project

Thomas Schaberreiter[1,2,3], Sébastien Varrette[1], Pascal Bouvry[1], Juha Röning[2], and Djamel Khadraoui[3]

[1] University of Luxembourg, Computer Science and Communications Research Unit, 6 rue Richard Coudenhove-Kalergi, L-1359 Luxembourg
{firstname.lastname}@uni.lu

[2] University of Oulu, Department of Electrical and Information Engineering, P.O.Box 4500, FIN-90014 University of Oulu, Finland
ouspg@ee.oulu.fi

[3] CRP Henri Tudor, Service Science & Innovation (SSI), 29, avenue John F. Kennedy, L-1855 Luxembourg, Luxembourg
{firstname.lastname}@tudor.lu

**Abstract.** Critical infrastructure (CI) services (like electricity, telecommunication or transport) are constantly consumed by the society and are not expected to fail. A common definition states that CIs are so vital to our society that a disruption would have a severe impact on both the society and the economy. CI security modelling was introduced in previous work to enable on-line risk monitoring in CIs that depend on each other by exchanging on-line risk alerts expressed in terms of a breach of Confidentiality, a breach of Integrity and degrading Availability (CIA). One important aspect for the accuracy of the model is the decomposition of CIs into CI security modelling elements (CI services, base measurements and dependencies). To assist in CI decomposition and provide more accurate results a methodology based on dependency analysis was presented in previous work.

In this work a proof-of-concept validation of the CI decomposition methodology is presented. We conduct a case study in the context of the Grid'5000 project, an academic computing grid with clusters distributed at several locations in France and Luxembourg. We show how a CI security model can be established by following the proposed CI decomposition methodology and we provide a discussion of the resulting model as well as experiences during the case study.

**Keywords:** Critical infrastructures, dependency analysis, risk monitoring, security modelling.

## 1 Introduction

Critical infrastructures (CIs) are complex interacting systems on a nationwide or even international level that provide services to society and economy. More importantly, those infrastructures are so vital to society and economy that a failure

A. Cuzzocrea et al. (Eds.): CD-ARES 2013 Workshops, LNCS 8128, pp. 269–287, 2013.

in service delivery or a degradation of quality-of-service would have severe consequences.

CIs face a multitude of risks due to for example technical faults, human error or deliberate attacks. Furthermore, dependencies to services provided by other CIs can cascade risks from one CI to another which makes risk monitoring in CI services even more complex. In recent years, research on CI security modelling tried to evaluate on-line risks (breach of Confidentiality, breach of Integrity and degrading Availability - CIA) in complex systems like CIs on the CI service level. The basic idea of CI security modelling is to estimate the current risk of a CI service by observing the system state of the components that are used to provide the service as well as by observing the risk experienced by dependencies. The main components of the model are *CI services* (services provided to customers or internal services utilized to provide services to customers), *base measurements* (system measurements defining the state of a service) and *dependencies* among CI services. For each CI service, risk is derived from the observations using a Bayesian network based approach.

A crucial part for accuracy of risk estimates is a well performed analysis of a CI and its dependencies to be able to represent it as a CI security model. In complex systems like CIs, possibly operated on a national or even international level, this can be problematic. To assist with the decomposition necessary for the CI security model, a methodology based on dependency analysis was proposed in [1]. The main idea of this approach is to use a *socio-technological* approach of analysing a complex system, meaning that all possible information sources, reaching from written technical documentation or contracts to interviews with employees on all organizational levels of a CI are utilized to get a holistic view of the complex systems and its complex dependencies. Another important aspect of this methodology is a graphical representation of the findings to get a visual feedback of the analysis.

In this work we present a proof-of-concept validation of the dependency analysis methodology based on a case study conducted within the framework of the Grid'5000 platform. We show that we can decompose parts of the Grid'5000 platform into CI services, base measurements and dependencies and thus build a CI security model. To simplify our work, we only take availability risk into account. The Grid'5000 project maintains an academic computing grid with clusters distributed at several locations in France and Luxembourg to perform large-scale scientific experiments that require substantial processing power and/or storage. The sites of Grid'5000 are geographically distributed and connected via a dedicated high-speed network connection operated by an independent provider, which makes this case study especially interesting for validation of the CI security model since being able to model dependencies to external providers is a major goal of the CI security model. While Grid'5000 is not a CI in the narrow sense, because it lacks its importance to society and economy, it is similar to CIs of the computing or telecommunication sector in technical as well as organizational realization. Therefore the Grid'5000 project is an ideal candidate for validating research related to CIs having in mind that it is hard to get access

to data coming from actual CIs. Furthermore, as a second contribution of this work, we provide an analysis of our case study experiences and we detail positive as well as negative aspects.

The remainder of the paper is organized as follows: Section 2 discusses related work. Section 3 introduces the previously proposed dependency analysis methodology as well as the CI security model. Section 4 presents the dependency analysis case study and Section 5 discusses case study results. Finally, Section 6 concludes the paper and gives an outlook on future work.

## 2 Related Work

Dependency analysis in CIs is a complex task. In [2] Rinaldi et al. provide an excellent introduction to the problem of CI (inter)dependency and its associated problems. Some authors propose CI models that focus on the interdependencies between CIs (e.g. [3], [4]), the main goal of those methods is to integrate CI sector-specific behaviour models with each other, whereas in CI security modelling we try to establish a CI model that uses model parameters (CI services and CIA risk) that are valid for each CI sector and thus allows to establish a cross-sector model. The first step of establishing such a model is to analyse the CIs and their dependencies to achieve an abstract representation of CIs as a modelling basis. In this work we use PROTOS-MATINE method for dependency analysis developed by OUSPG (Oulu University Secure Programming Group). The method was originally developed for identifying protocol dependencies [5] and is based on analysis of all available information sources to get a holistic picture of dependencies. The method has proven to be especially useful in combination with the semantic tool Graphingwiki [6] which allows to graphically represent the dependency model and refine the model whenever new information arrives. The PROTOS-MATINE method has furthermore been used to analyse critical dependencies in the context of antivirus software [7] and the socio-technical dependencies of a VoIP infrastructure [8]. In [1] the PROTOS-MATINE method was adapted to be used in the context of the CI security model [9], [10], [11].

The goal of CI security modelling is to propose a risk-based cross-sector CI model to be used for on-line risk monitoring. The model parameters are *CI services*, *dependencies* between CI services and *base measurements* (observable measures that define a CI service state). The model output is *CI service risk*, where risk is seen as a breach of Confidentiality, a breach of Integrity or degrading Availability (CIA). In [12] a Bayesian network based risk estimation methodology was introduced to the CI security model which provides a more sophisticated way of estimating risk as well as some advanced features like risk prediction. Some advantages of the CI security model compared to other CI models (e.g. [13], [14]) or CI risk models [15], [16] is the flexibility of the approach that allows to model CI systems to the desired level of detail, the comparability of different CI sectors due to abstraction to common, risk related indicators, and the ability to include dependent service risk in CI risk estimation.

## 3 CI Dependency Analysis

The dependency analysis method presented in [1] is used to find the modelling entities of the CI security model, the CI services, base measurements and dependencies. The core idea of the approach is to find a suitable model by combining information gathered from all available information sources at different organizational levels in a CI (preparedness and business continuity level, process level and technical level). The combination of internal (e.g. contracts, policies, expert interviews, manuals) and external (e.g. news, vulnerability feeds, laws/regulations) information sources provides a holistic view on the analysed system.

After gathering the information sources, the dependency analysis is carried out in an iterative manner. The first step is to generate a high level model of the system, finding the high level services and dependencies that define the system. In the next steps, the model is refined and detailed low level services/dependencies and base measurements are identified. During the process of better understanding the system by adding more detail, it is expected that the high level system model will be refined as well. It is expected that the high level system model will be mainly composed of services and dependencies, since the base measurements (e.g. sensor outputs), are expected to be rather available on the process level/ technical level than on the preparedness and business continuity level.

One important aspect of the proposed CI dependency analysis method is the graphical representation of the CI model. In order to facilitate discussion and expert input, the model has to be presented in a clear and intuitive way so that errors and wrong assumptions in the model are easier to identify. In this work the model is represented as a graph, where the CI services and base measurements represent nodes (CI services are visualized as elliptic shapes and base measurements as rectangle shapes) and the dependencies among services as well as among services and base measurements are represented as edges.

### 3.1 Bayesian Network Based CI Security Model

Since the publication of the CI dependency analysis method in [1], the CI security model was refined by adding a Bayesian network based component for risk estimation in [12]. The core concept of CI dependency analysis (identification of services, dependencies and base measurements) does not change compared to the original proposal, but the concept of how service risk is derived from dependent service risk and base measurement observations did change, therefore the integration of the Bayesian network based CI security model with the CI dependency analysis methodology is presented in this Section.

In the original proposal of the CI security model in [9], [10], [11], the service risk is calculated using a weighted sum method for which each dependency of a service (another CI service or a base measurement) is weighted by an expert based on the assumed importance to the service. In the Bayesian network based CI security model, service risk is estimated based on the state of dependencies, which is represented by the conditional probability table (CPT) of a node.

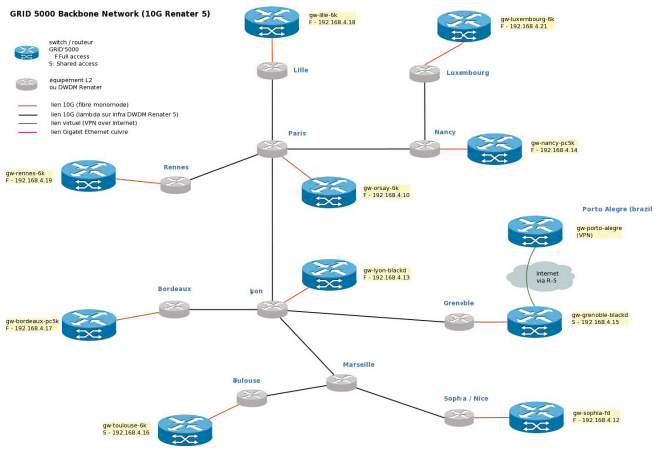


**Fig. 1.** Network backbone interconnection between the sites of the Grid'5000 platform

For each state combination of the dependencies of a service, the most probable service risk is estimated. Those probabilities can be learned from data records as well as estimated by experts where learning is not possible.

During the CI dependency analysis process, the Bayesian network based security model requires identification of following parameters:

- Normalization values for base measurements based on classification of normal operation/ allowed deviation from normal operation. To avoid state explosion, the CI security model requires that all base measurements are normalized to 5 discrete values, where 1 characterizes a base measurement value during normal operation and 5 the maximum allowed deviation. The classification of the boundaries is assumed to be done using technical specifications and expert estimation.
- For each service, the service risk has to be estimated for incidents that happened during the time period that is used to learn the risk probabilities from data. This is necessary since service risk is an abstract concept that can not be directly measured. Estimating service risk for past incidents will allow to learn the most probable service risk for each dependency state combination.
- In situations where service risk probabilities can not be learned from data due to missing data records or insufficient data, expert estimation is required to estimate the service risk probabilities for each missing dependency state combination.

In this case study we will, besides identifying the CI services, base measurements and dependencies, also describe the normalization process for each identified base measurement.

## 4 Grid'5000 Case Study

In this Section, the CI dependency analysis methodology is validated using the Grid'5000 platform[1] which provides the users with a fully customizable testbed to perform advanced experiments in all areas of computer science related to parallel, large-scale or distributed computing and networking. Grid'5000 is a distributed computing platform featuring clusters located in nine geographical sites – eight in France (Bordeaux, Grenoble, Lille, Lyon, Nancy, Reims, Rennes, Sophia, Toulouse) and one in Luxembourg. The interconnect backbone between all sites is illustrated in Figure 1. In addition, international connections to Brazil, Japan and the Netherlands operated via the site of Grenoble are available.

### 4.1 High-level Grid'5000 Security Model

The resulting CI security model of the highest decomposition level of Grid'5000 can be seen in Figure 2. The infrastructure is divided into two providers, the *Grid'5000* provider and the *Network provider*, since those infrastructures are operated by separate companies and therefore the organizational structure and day-to-day operation of the infrastructures differ greatly from each other.

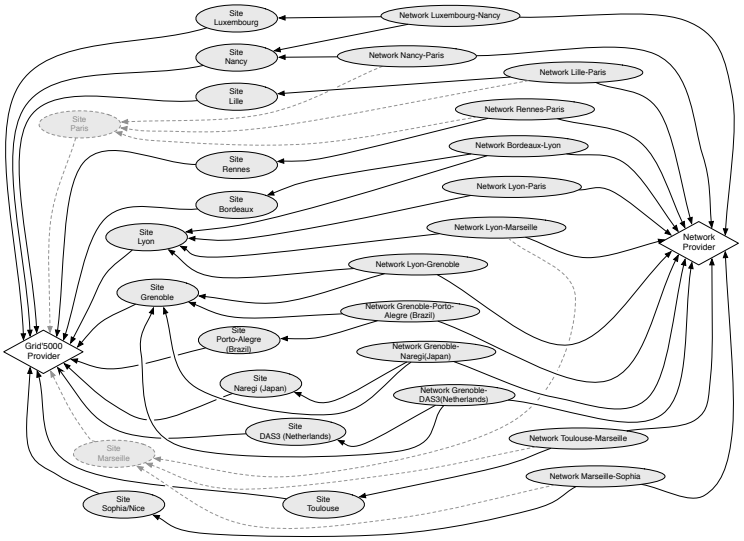


**Fig. 2.** Grid'5000 high level CI security model for availability risk indicator

Each site of Grid'5000 is represented as as service[2] that will aggregate service risk from the lower-level model described in the following Sections. The *Grid'5000*

[1] `http://www.grid5000.fr`

[2] The Sites represented by dotted lines in Figure 2 are part of the Grid'5000 project, but currently offline.

*provider* depends on all sites which allows to produce a global risk estimate for Grid'5000, taking the risk of the individual sites into account.

The network provider is modelled by defining each network link between two sites as a service. The risk estimate of each link aggregated from lower-level model of the network provider can be used at this decomposition level to estimate a risk for each network segment as well as a global risk estimate for the *Network provider*. Therefore, each service characterizing a network segment is modelled as a dependency of the *Network provider*. This representation was used since it follows the geographical structure of the network and it provides a good basis to model the lower decomposition levels as shown in the following Sections.

The dependencies between the network provider and the Grid'5000 provider were modelled in a way that each site physically connected to a network segment, depends on this network segment. In this way a changed risk in a network segment can be easily included in the risk estimation of the site that depends on this segment.

The main information sources to gather the information used at this decomposition level are the internal Grid'5000 documentation containing information about the high-level structure of Grid'5000, like the network backbone shown in Figure 1 and expert knowledge about Grid'5000 to account for the most recent changes in the the structure of Grid'5000.

### 4.2 The Network Backbone

Every network segment of the backbone presented in the Figure 1 is characterized by connection points within the dark fibre infrastructure provided by Renater[3] which allocates dedicated 10Gbit/s "lambdas" on a DWDM (Dense Wavelength Division Multiplexing) infrastructure for the Grid'5000 platform. Grid'5000 sites see each other inside the same VLAN at 10Gbps. A few bottleneck still exists, like the Lyon to Paris link, where the 10Gbps are shared between all the sites above Lyon and all the sites under Lyon.

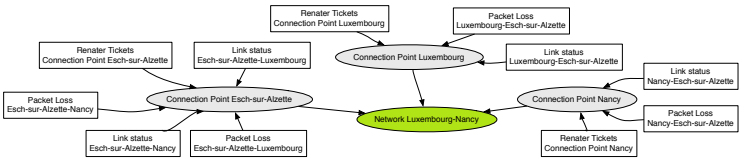


**Fig. 3.** CI security model for availability risk indicator of network segment Luxembourg-Nancy

Each network segment can be observed by incident monitoring as well as network performance measures, both of them performed by the backbone provider (Renater in France and Restena in Luxembourg). The incident monitoring takes the form of tickets which can be accessed by Grid'5000 administrators in real-time and are one of the following:

[3] `http://www.renater.fr/`

- *incident* : a sudden IP service disruption occurs on the network.
- *maintenance* : an operation will be performed on a network equipment that will (or not) interrupt the IP service.

Regarding backbone performance, Grid'5000 offers 3 different types of monitoring information: The *link status* (up or down), the *packet count* (reported by the SNMP counters of Renater/Restena switches) and the *packet loss* (PL) occurring on the link. Packet Loss can be caused by the saturation of a link. When router or switch buffers are unable to store packets, the packets are dropped. It may also be caused by a faulty equipment or a flapping link (link up/down/up/-down...).

### 4.3 The Network Backbone CI Security Model

The resulting CI security model for the network link Luxembourg-Nancy is presented in Figure 3. To account for space constraints in a publication, only the detailed decomposition of one network segment is presented in this work, but it should be noted here that the decomposition of all network segments follows a similar structure.

The network connection between the sites Luxembourg and Nancy is characterized by the three main connection points in Luxembourg, Esch-sur-Alzette and Nancy. In Figure 3 those are represented as services that are characterized by the measurements collected at the level of the routers at those locations (performance measures) and Restena tickets.

Normalization of base measurements according to the CI security model for Restena tickets is done following the information presented in Table 1. No ticket means normal operation or normalization level 1. A *maintenance* ticket is characterized as normalization level 3, since it will not interrupt the service, but a degradation in service can occur. An *incident* ticket is characterized as normalization level 5, since it implies an interruption of service.

From the performance measures presented in the previous Section, only *link status* and *packet loss* are taken into account, since they fully characterize the service availability status. *Link status*, as specified in Table 2a can be see as a binary base measurement where link up represents the normalized level 1 and link down represents level 5. The *Packet loss* measurement is normalized according to the boundaries presented in Table 2b, with rather restrictive bounds to reach level 1 or level 5.

**Table 1.** Base measurement normalization for Renater/Restina tickets

| Level | Description |
|---|---|
| 1 | Normal |
| 3 | *maintenance* ticket for the link is pending |
| 5 | *incident* ticket for the link is pending |

**Table 2.** (a): Base measurement normalization for link status. (b): Base measurement normalization for packet loss.

| Level | Description |
|---|---|
| 1 | Link `up` |
| 5 | Link `down` |

(a)

| Level | Description |
|---|---|
| 1 | Normal: $PL = 0\%$ |
| 2 | Degraded: $0 < PL \leq 33\%$ |
| 3 | Really Degraded: $33\% < PL \leq 66\%$ |
| 4 | Hightly Degraded: $66\% < PL \leq 99\%$ |
| 5 | $PL > 99\%$ |

(b)

The main information sources at this level of decomposition were Grid'5000 as well as Restena network documentation (as far as available to the Grid'5000 project), manuals and documentation of the used monitoring tools (Passilo) and expert knowledge, especially to define the base measurement normalization bounds.

### 4.4 Grid'5000 Site Infrastructure

Each site of Grid'5000 holds one or several computing clusters. The hardware configuration of each site is depicted in Figure 4a and is typically organized in the following way:

- An *access* server which serves as an isolated user frontend to access the resources of the site;
- one or several *adminfront* server(s) holding a set of virtual machines dedicated to platform services.
- the *computing nodes*, split among one or more clusters (depending on the site capacity).
- an *NFS* server (together with its associated disk enclosure) that act as a centralized and shared storage area for the site.
- the *site router* which interconnects all site components and enables communication to other sites of Grid'5000.
- a *local interconnect switch*, which provides an isolated network for the site internal components.

Apart from the general structure, Grid'5000 consists of several elements crucial for operation, especially in the context of availability: The *Puppet infrastructure*[4] (responsible for the configuration and the deployment of all grid services within Grid'5000), the *Platform Monitoring infrastructure* (a set of monitoring tools such as Nagios[5] or Ganglia [17]), *OAR* [18] (the resource manager of Grid'5000) and *Kadeploy*[6] (piloting the deployment of the computing nodes) as well as the *technical committee* (engineers responsible for the maintenance and the development of the platform) and the *users* of Grid'5000.

[4] `http://puppetlabs.com/`
[5] `http://www.nagios.org/`
[6] `http://kadeploy.gforge.inria.fr/`

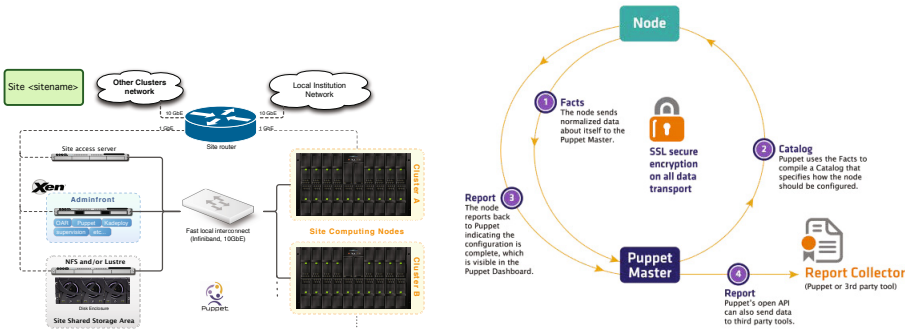


**Fig. 4.** (a): General overview of the hardware organization of each site. (b): General Puppet Workflow.

**The Puppet Infrastructure.** Puppet is an IT automation software based on a master/slave approach that helps system administrators to manage their infrastructure throughout its life cycle, from provisioning and configuration to patch management and compliance. Using Puppet, it is easy to automate repetitive tasks, quickly deploy critical applications, and pro-actively manage change in a scalable way. Puppet uses a declarative, model-based approach to IT automation. In particular, the desired state of the infrastructure's configuration is defined using Puppet's declarative configuration language. After a Puppet configuration is deployed, a Puppet agent is run on each node to automatically enforce the desired node state, correcting any configuration drift. As illustrated in Figure 4b, the Puppet agent of a node sends *Facts*, or data about its state, to the Puppet Master which compiles a *Catalog*, or detailed data about how the node should be configured, and sends this back to the Puppet agent. After making any changes to return to the desired state, the Puppet agent sends a complete Report back to the Puppet Master. In this work we propose to use the Puppet agent exit codes to classify the status of a given service. The exit codes are reported every time the Puppet agent is executed, which is every 30 minutes by default.

**The Platform Monitoring Infrastructure.** Many monitoring tools are used to reflect the status of the platform. The main one is Nagios which evaluates at the moment of writing 160 hosts and more than 629 service status. The different services currently monitored by Nagios plugins are summarized in Table 3.

Each service is checked by the Nagios daemon, either at regular intervals or On-demand as needed for predictive service dependency checks (such checks help to ensure that the dependency logic is as accurate as possible.) Services that are checked can be in one of four different states (`OK`, `WARNING`, `UNKNOWN` or `CRITICAL`).

**Table 3.** Nagios services monitored on Grid'5000

| ID | Service | ID | Service | ID | Service | ID | Service |
|---|---|---|---|---|---|---|---|
| N1 | APT | N6 | `/home` usage | N11 | LDAP | N16 | NTP |
| N2 | Conman consoles server | N7 | Ganglia | N12 | Load usage | N17 | OAR |
| N3 | Dhcp-proxy | N8 | HTTP | N13 | Mail server | N18 | PING |
| N4 | DNS | N9 | Kadeploy | N14 | MySQL | N19 | Squid |
| N5 | Filesystem usage | N10 | KaVLAN server | N15 | Nagios | N20 | SSH |

**The OAR and Kadeploy Infrastructure.** OAR is a versatile resource and task manager (also called a batch scheduler) for clusters and other computing infrastructures which is used in particular on Grid'5000 and provides a simple and flexible exploitation of a cluster. Its design is based on high level tools, more precisely a relational database engine (MySQL or PostgreSQL), a scripting language (Perl), a confinement system mechanism based on a feature proposed by recent (2.6) Linux kernels called `cpuset` and a scalable administrative tool (component of the `Taktuk` framework). It is composed of modules which interact only with the database and are executed as independent programs. Whereas the OAR service is managed by Puppet on Grid'5000, it integrates a set of internal checks of the computing nodes it pilots such that from the node status reported by OAR, it is possible to get information on the current status of the node.

For node deployment OAR is coupled with the Kadeploy middleware. Kadeploy belongs to the category of disk imaging and cloning tools and is used to install and customize servers. Users are able to deploy a new environment and customize computing nodes during their reservations. In parallel, other tools have been designed internally to qualify the good status of the computing nodes and detect altered environments. Among them, `phoenix` analyses the production environment deployed after user reservation and restarts the full deployment process up to 3 times if errors are detected (bad disk formatting, unhealthy system etc.), which can be used as measurement to determine the status of computing nodes.

### 4.5 Grid'5000 Site Infrastructure CI Security Model

For each site of Grid'5000 we identified 4 main factors that contribute to the availability risk of the site:

- **Cluster**: The hardware and software components that form a computing cluster.
- **Network connection**: The network interconnection between computing sites, as detailed in Section 4.2.
- **Environmental factors**: The supporting infrastructure needed to operate the computing cluster.

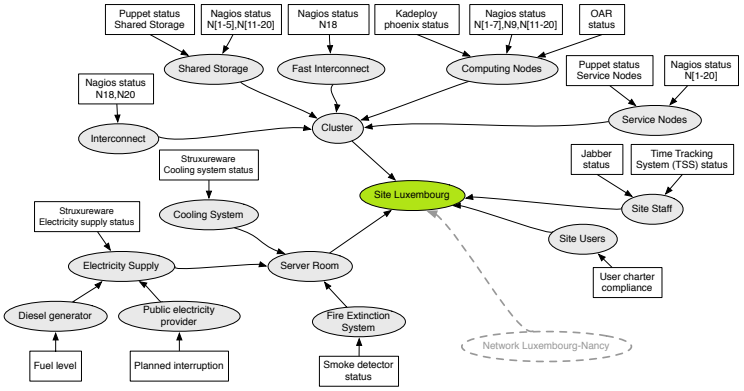


**Fig. 5.** Site Luxembourg CI security model for availability risk indicator

– **Human factor**: This describes how humans can influence the availability of the infrastructure. In Grid'5000, two types of human interaction was identified, by the *staff* operating and maintaining the grid and the *users* that can influence availability since they are granted substantial user rights to operate the grid.

The main information sources at this decomposition level were structural as well as organizational documentation of Grid'5000 as well as interviews with Grid'5000 administrators.

Each site of Grid'5000 follows the same basic structure, but further decomposition reveals differences in implementation details, for example different hardware components that allow different levels of monitoring as well as differences in the number of employed staff. In this work we illustrate the decomposition of the site Luxembourg and while the basic structure of the CI security model remains the same for each site, the outcome of the model depends on the actual implementation of the site.

The resulting CI security model for site Luxembourg can be seen in Figure 5. In the first level of decomposition, the node *Server Room* represents the environmental factors of the site, the *Site staff* and the *Site Users* represent the human factor, the *Network Luxembourg-Nancy* represents the network connection and the *Cluster* represents the actual computing cluster.

The main part of the site Luxembourg is the *Cluster* which can be further de-composed into the 5 major components (as illustrated in Figure 4a), the *Interconnect* (local interconnect to other institutions within site) and the *Fast Interconnect* (interconnect for site internal components), the *Computing Nodes*, the *Shared Storage* and the *Service Nodes* (a summation of supporting services like OAR, Puppet, Kadeploy, supervision or site access server, as listed in Figure 4a). The service state of the Interconnect and Fast Interconnect can be

described by the the state of services they are composed of, monitored by nagios plugins N18 and N20 (Interconnect) and N18 (Fast Interconnect). The Shared Storage state can be characterized by the output of the puppet configuration status for shared storage and the state of the services monitored by nagios plugins N1-N5 and N11-N20. The state of the computing nodes can be described by the output of Kadeploy phoenix status, the node status as monitored by OAR[7] and the state of the services monitored by nagios plugins N1-N7, N9 and N11-N20. The Service Nodes state can be characterized by the puppet configuration status for each service node[8] as well as the status of the services monitored by the nagios plugins N1-N20.

The base measurement normalization for the output of the nagios plugins (N1-N20) is listed in Table 4a. The nagios plugins report 4 different states for the services they monitor which can be translated to normalization levels. Since only 4 states are available, normalization level 4 was omitted.

Puppet configuration, reports exit codes which are translated into normalization levels as illustrated in Table 4b.

**Table 4.** (a): Base measurement normalization for Nagios service checks. (b): Base measurement normalization for Puppet-managed services.

| Level | Description |
|---|---|
| 1 | OK |
| 2 | WARNING |
| 3 | UNKNOWN |
| 4 | n/a |
| 5 | CRITICAL |

(a)

| Level | Description |
|---|---|
| 1 | Normal: exit code 0 |
| 2 | Configuration changes: exit code 2 |
| 3 | Puppet master is unreachable |
| 4 | Failures during the transaction: exit code 4 |
| 5 | Configuration changes and failures during the transaction: exit code 6 |

(b)

The node status reported by OAR is translated into normalization levels according to Table 5a and Kadeploy phoenix node status is normalized according to Table 6a.

The main information sources to obtain the structure of the Cluster were structural documentation of Grid'5000 clusters, the technical documentations as well as log output of the platform monitoring tools to determine base measurements and base measurement normalization and expert interviews to verify the information.

[7] Each available computing node has a seperate Kadeploy phoenox as well as OAR status base measurement. To maintain readability of Figure 5, those base measurements are summarized by *"Kadeploy phoenix status"* and "OAR status".

[8] A puppet status base measurement for each monitored service is available. To maintain readability of Figure 5, only a single base measurement, *"Puppet status Service Nodes"* was introduced.

The second important aspect of each site are the environmental factors. The basic idea of this decomposition is to capture data from the environment the site is located in to be able to monitor not only the Cluster itself, but also be able to react to a changing environment that can influence the operation of the cluster (like for example fires or power shortage). At the site Luxembourg the main place to capture environmental factors is the *Server Room* the cluster is located in. We identified three main elements which would be *Cooling* of computing equipment, the *Fire Extinction System* and the *Electricity Supply*. The *Electricity Supply* of the Site Luxembourg is composed of the main public electricity supply as well as backup energy supply by a *Diesel generator*. The state of the electricity supply can on the one hand be determined by the output of Struxureware software[9], a tool that monitors several sensors and aggregates a risk level which is presented to the operator. On the other hand, additional information about the availability status of the public electricity provider as well as a backup diesel generator can be given by monitoring notifications about planned interruptions of public electricity supply as well as the diesel generator fuel level. The base measurement normalization of the Stuxureware risk levels can be found in Table 5b. The provided risk levels of low, medium and high are mapped to the normalization values 1, 3 and 5 respectively. The base measurement normalization for the planned public electricity supply can is presented in Table 6b. The normalization bounds result from the analysis of the electricity supply: For up to 5 minutes, batteries guarantee operation of the cluster and the backup diesel generator should provide backup electricity supply for up to 30 minutes. The base measurement normalization for the diesel generator fuel level can be seen in Table 7a. Unfortunately, the generator does not report fine grained fuel level status, but only *full* or *not full*. The cooling system status can be determined by Stuxureware alerts. The base measurement normalization is identical to the electricity supply status and can be seen in Table 5b. The fire extinction system status can be determined by the smoke detector status, each smoke detector can have three different states: *no alert*, *defect* or *alert*. The base measurement normalization for the smoke detector status can be found in Table 7b.

**Table 5.** (a): Base measurement normalization for OAR based node status checks. (b): Base measurement normalization for Struxureware monitored services.

| Level | Description |
|---|---|
| 1 | Free: the node is unused and powered |
| 2 | Reserved: at least one job is running |
| 3 | Standby: the node is ready to wake up |
| 4 | Suspected |
| 5 | Down |

(a)

| Level | Description |
|---|---|
| 1 | Low Risk Assessment |
| 3 | Medium Risk Assessment |
| 5 | High Risk Assessment |

(b)

[9] `http://www.apc.com/`.

**Table 6.** (a): Base measurement normalization for Kadeploy phoenix status. (b): Base measurement normalization for planned electricity supply interruption.

| Level | Description |
|---|---|
| 1 | Node deployed without error |
| 2 | Node is re-deployed after error |
| 3 | Node is re-deployed a second time after error |
| 4 | Node is re-deployed a third time after error |
| 5 | Node is down |

(a)

| Level | Description |
|---|---|
| 1 | No interruption |
| 2 | 0-5 Minutes |
| 3 | 6-15 Minutes |
| 4 | 16-30 Minutes |
| 5 | 30 Minutes and more |

(b)

The main information sources regarding identification of environmental factors were interviews with personal working in the server room as well as technical documentation of the server rack and the fire extinction system.

The human factor in a complex system is generally hard to quantify. In this case study, there are two separate groups that can influence the availability of the site Luxembourg: The *Site Staff* and the *Site Users*. We have identified that the main factor concerning availability in the Site Luxembourg caused by *Site Staff* is absence due to vacation or sick leave. Collecting and monitoring the availability of each staff member within Grid'5000 is performed via a time tracking system (TTS). Another interesting tool that help to monitor the staff presence is the Grid'5000 Jabber service (a messaging service similar to MSN or Skype). Monitoring the Jabber online status can add additional information about availability of site staff since in the computing sector personal can be available and resolve possible problems without physical presence.

The base measurement normalization of the Jabber and TTS status can be found in Tables 7c and 8a. Each member of the site staff will have a base measurement for *"Jabber status"* and *"Time Tracking System (TTS) status"*, even though in Figure 5 only one of each base measurements is introduced for the sake of readability.

**Table 7.** (a): Base measurement normalization for diesel generator fuel level. (b): Base measurement normalization for smoke detector status. (c): Base measurement normalization for Jabber status.

| Level | Description |
|---|---|
| 1 | Full |
| 5 | Not full |

(a)

| Level | Description |
|---|---|
| 1 | No alert |
| 3 | Defect |
| 5 | Alert |

(b)

| Level | Description |
|---|---|
| 1 | Online |
| 5 | Offline |

(c)

The influence of *Site Users* to the availability of site Luxembourg is hard to determine. On the one hand, due to the infrastructure-as-a-service approach of Grid'5000, site users have quite substantial rights to interact with Grid'5000 infrastructure. On the other hand, due to the academic nature of Grid'5000 there are not many tools in place to monitor user actions. One factor we could identify is *User charter compliance* measured by a script designed to check on a regular basis the compliance of the user behaviour with the user charter. The user charter compliance measurement normalization can be found in Table 8b. User charter compliance is measured for each user, but for readability reasons only one base measurement *"User Charter Compliance"* was introduced in Figure 5.

**Table 8.** (a): Base measurement normalization for Time Tracking System status. (b): Base measurement normalization for user charter compliance.

| Level | Description |
|---|---|
| 1 | Available |
| 5 | Not Available |

(a)

| Level | Description |
|---|---|
| 1 | No violations of user charter compliance |
| 2 | 1 violation of user charter compliance |
| 3 | 2-3 violations of user charter compliance |
| 4 | 4-5 violations of user charter compliance |
| 5 | 5 or more violations of user charter compliance |

(b)

## 5 Results and Discussion

In this Section we would like to discuss the observations made during the case study in more detail. We want to highlight which parts of the case study provided the expected results according to the proposed dependency analysis methodology, but we also want to highlight some problems and unexpected obstacles we experienced during the case study. At the end we want to summarize our findings and draw a conclusion regarding the feasibility of the proposed methodology.

On the positive side, by conducting this case study we have shown that it is possible to represent a complex system like Grid'5000 using the abstract elements of the CI security model (CI services, base measurements and dependencies) by following the steps of the presented dependency analysis methodology. We have successfully established a model that includes the socio-technical structure of Grid'5000 and takes technical as well as human and environmental components into account allowing us to provide a holistic view on the availability risk of CI services at different organizational levels. We have also shown that hierarchical decomposition in this context is possible by decomposing higher level services into sub-components and treating them as services they provide to higher level services. We have shown that we can identify base measurements within the CI systems to be able to observe the system state. And finally, we could show that we could identify enough information sources within Grid'5000 (e.g. documentation or expert interviews) or from external sources (e.g. manuals) to establish the

CI security model. It was experienced that expert interviews are a very valuable source of information, especially on the higher levels of decomposition. The lower, more technical levels are usually well described by documentation/manuals. In any case it was experienced that, as expected, the graphical representation of the CI security model is a very useful tool to discuss the model structure and find flaws in the representation.

Aside from the positive aspects, some observations were made during the course of the case study that could have a negative implication on the outcome of future case studies. One major observation was that not all information can be made available due to confidentiality or privacy reasons. This was even experienced in this case study (e.g. employee records due to privacy concerns) in an academic environment which is usually more open than a commercial operator. If the CI security model should be applied in a more competitive commercial environment, restricted access to information due to confidentiality reasons could pose a problem. Even though risk alerts are abstract metrics and should help to maintain source confidentiality, providers might be reluctant to provide the necessary structural information (CI services and dependencies) to business partners and possibly competitors.

Another observation made during the case study was that information gathering was generally more difficult and time consuming than expected. One problem contributing to this observation were for example the organizational structure of some of the sites. For example, the server room in one site was the responsibility of the universities maintenance department and the administration of the cluster is done by research personnel. Information requests regarding server room equipment and actual hardware caused substantial delays because those requests were treated with different urgency by the maintenance department for mainly two reasons: a) The employees were already overworked with other, more important tasks and b) since the maintenance department and research personnel of the university are organized in different branches of the organizational hierarchy, there is no direct authority which would give requests for information more urgency. Although this scenario was specific to this case study, similar challenges are expected to be experienced in other big companies or complex infrastructures.

The last observation that was made during the case study was that base measurements, although available and observable, are not always recorded. Before the case study it was assumed that sensor data is recorded for a certain time period. In the CI security model, recorded sensor data can be used to learn risk probabilities from events that happened in the past, but without data records learning is not possible. In this case, risk probabilities need to be estimated by experts which makes the estimates more inaccurate and error prone.

To conclude this Section, we want to state that this case study has shown that it is possible to represent complex infrastructures as CI security model, but it requires some effort and dedication by the operators to achieve the goal. We also recognize that every complex infrastructure is different and that one case study can not proof or disproof the validity of our approach, but we could make

some general positive as well as negative observations that can serve as a basis for future discussions and case studies.

## 6 Conclusion and Future Work

In this work we presented a case study to provide a proof-of-concept validation of a CI dependency analysis methodology in the context of CI security modelling. CI security modelling provides a framework for on-line risk monitoring of CI services, taking into account the service state as well as the risk state of dependencies. A crucial part for the success of the model is a well performed CI analysis to identify the elements forming the CI security model, which are CI services (services provided to customers or internal services utilized to provide the service to the customer), base measurements (measurements to determine the state of a CI services) and dependencies between those elements. The presented case study was performed in the context of Grid'5000, a distributed academic computing grid operated by several universities in France and other countries like Luxembourg. The sites of the grid are connected by a dedicated high-speed fibre optic network connection. As a result of the case study, we have shown that, following the proposed dependency analysis methodology, it is possible to analyse a complex infrastructure and represent it as a CI security model. We have also seen that it is not trivial to gather the necessary information due to for example organizational reasons. Though it is not possible to provide a final verdict on the general feasibility of the methodology since every CI is different, the results suggest that the methodology will also work in other CI environments.

In future work, Bayesian network based risk estimation within the Grid'5000 CI security model as well as risk simulation based on attack or failure scenarios will be conducted. For this purpose a tool is currently implemented that allows risk estimation as well as risk simulation/emulation in the context of the CI security model. Given the opportunity, we also plan to conduct further case studies to validate the dependency analysis methodology as well as the CI security model in a broader context.

**Acknowledgements.** One of the authors would like to thank the Luxembourgish National Research Fund (FNR) for funding his PhD work under AFR grant number PHD-09-103.

## References

1. Schaberreiter, T., Kittilä, K., Halunen, K., Röning, J., Khadraoui, D.: Risk assessment in critical infrastructure security modelling based on dependency analysis (short paper). In: 6th International Conference on Critical Information Infrastructure Security, CRITIS 2011 (2011)
2. Rinaldi, S.M., Peerenboom, J.P., Kelly, T.K.: Identifying, understanding, and analyzing critical infrastructure interdependencies. IEEE Control Systems Magazine (2001)

3. Panzieri, S., Setola, R., Ulivi, G.: An approach to model complex interdependent infrastructures. In: 16th IFAC World Congress (2005)
4. Tolone, W.J., Wilson, D., Raja, A., Xiang, W.-N., Hao, H., Phelps, S., Johnson, E.W.: Critical infrastructure integration modeling and simulation. In: Chen, H., Moore, R., Zeng, D.D., Leavitt, J. (eds.) ISI 2004. LNCS, vol. 3073, pp. 214–225. Springer, Heidelberg (2004)
5. Eronen, J., Laakso, M.: A case for protocol dependency. In: IEEE International Workshop on Critical Infrastructure Protection (2005)
6. Eronen, J., Röning, J.: Graphingwiki - a semantic wiki extension for visualising and inferring protocol dependency. In: Proceedings of the First Workshop on Semantic Wikis – From Wiki To Semantics. Workshop on Semantic Wikis (2006)
7. Eronen, J., Karjalainen, K., Puuperä, R., Kuusela, E., Halunen, K., Laakso, M., Röning, J.: Software vulnerability vs. critical infrastructure - a case study of antivirus software. International Journal on Advances in Security 2 (2009)
8. Pietikäinen, P., Karjalainen, K., Eronen, J., Röning, J.: Socio-technical security assessment of a voip system. In: The Fourth International Conference on Emerging Security Information, Systems and Technologies, SECURWARE 2010 (2010)
9. Aubert, J., Schaberreiter, T., Incoul, C., Khadraoui, D., Gateau, B.: Risk-based methodology for real-time security monitoring of interdependent services in critical infrastructures. In: International Conference on Availability, Reliability, and Security, ARES 2010 (2010)
10. Aubert, J., Schaberreiter, T., Incoul, C., Khadraoui, D.: Real-time security monitoring of interdependent services in critical infrastructures. Case study of a risk-based approach. In: 21th European Safety and Reliability Conference, ESREL 2010 (2010)
11. Schaberreiter, T., Aubert, J., Khadraoui, D.: Critical infrastructure security modelling and resci-monitor: A risk based critical infrastructure model. In: IST-Africa Conference Proceedings (2011)
12. Schaberreiter, T., Bouvry, P., Röning, J., Khadraoui, D.: A bayesian network based critical infrastructure model. In: Schütze, O., Coello Coello, C.A., Tantar, A.-A., Tantar, E., Bouvry, P., Del Moral, P., Legrand, P. (eds.) EVOLVE - A Bridge Between Probability. AISC, vol. 175, pp. 207–218. Springer, Heidelberg (2012)
13. Rinaldi, S.: Modeling and simulating critical infrastructures and their interdependencies. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences (2004)
14. Sokolowski, J., Turnitsa, C., Diallo, S.: A conceptual modeling method for critical infrastructure modeling. In: 41st Annual Simulation Symposium, ANSS 2008 (2008)
15. Tan, X., Zhang, Y., Cui, X., Xi, H.: Using hidden markov models to evaluate the real-time risks of network. In: IEEE International Symposium on Knowledge Acquisition and Modeling Workshop, KAM Workshop 2008 (2008)
16. Haslum, K., Årnes, A.: Multisensor real-time risk assessment using continuous-time hidden markov models. In: Wang, Y., Cheung, Y.-M., Liu, H. (eds.) CIS 2006. LNCS (LNAI), vol. 4456, pp. 694–703. Springer, Heidelberg (2007)
17. Massie, M.L., Chun, B.N., Culler, D.E.: The Ganglia Distributed Monitoring System: Design, Implementation, and Experience. Parallel Computing 30 (2004)
18. Capit, N., Da Costa, G., Georgiou, Y., Huard, G., Martin, C., Mounié, G., Neyron, P., Richard, O.: A batch scheduler with high level components. In: Cluster Computing and Grid 2005, CCGrid 2005 (2005)