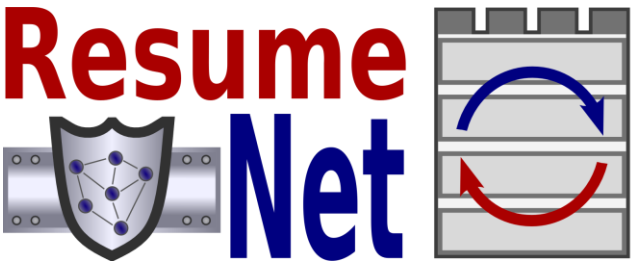
Resilience and Survivability for future networking: framework, mechanisms, and experimental evaluation



Deliverable number:	D3.2
Deliverable name:	Service surveillance and detection of challenging situation (interim)
WP number:	3
Delivery date:	30.06.2010
Date of preparation:	05.07.2010
Editor:	C. Lac (FT)
Contributors:	B. Delosme (FT), C. Dousson (FT), N. Kheir (FT), C. Lac (FT), P. Smith (ULANC)
Internal reviewer:	S. Martin (ULg)

Summary

This interim deliverable, result of the task 3.4 entitled "Service surveillance and detection of challenging situations" is about the Detection/Remediation/Recovery (DR^2) phases of ResumeNet's D^2R^2+DR strategy.

The task aims to build a framework for monitoring any service requiring a certain level of resilience. To this end, probes need to be inserted at the proper location in order to detect abnormal events. Their outputs, called *alarms*, are analyzed and treated using a correlation engine which requires as input a clear definition of challenging situations, including the resilience metrics used, for the service under study. The outcome of this analysis, i.e., a challenge detection called *alert*, will trigger remediation, e.g., with the help of policies deployment/modification. Continuous monitoring and event analysis provide information about the end of the threat, leading to actions to recover the system, bringing the service back to its normal performance operation.

As challenge detection is also realized in ResumeNet, but at a network level (task 2.2), the general architecture adopted in the project for this DR^2 purpose, including the Publish/Subscribe distributed store for challenges and remediation, is described. After a general description of the service monitoring problem, the approach proposed for this task in presented. Basic monitoring principles will be then surveyed, focusing on general policy, techniques, data, and maintenance operations. The scope of events correlation, i.e., fault localization, will finally be presented.

The correlation engine to be used is based on chronicles recognition, which will be described, followed by some applications of this technique in networks and services alarms analysis for security-related objectives (intrusion detection, reflexive DDoS attack), or dependability-related concerns (recovery actions monitoring, handover initiation in mobile systems).

The illustration of the Remediation phase, i.e., reaction to an alert generated by the correlation analysis, is done with the use of a dynamic policy engine, through the experimentation scenario "Communicating objects' data platform" which will deployed as a service use case in task 4.4. The conclusion reviews the principal contributions of this deliverable, as well as some aspects of the work planned during year 3 which will be reported in the final version of D3.2.

Table of contents

1. Introduction....4
2. Detection and remediation architecture....5
3. Service monitoring....6
3.1. Problem description....6
3.2. Proposed approach....7
3.3. Monitoring principles....8
3.4. Events correlation....12
4. Chronicle recognition system....13
4.1. Chronicle recognition....14
4.2. Chronicle learning....15
4.3. Experiments and applications....17
5. Sketching remediation in a service use case....18
5.1. Architecture of the communication objects' data environment....19
5.2. PubSub platform experimentation....21
5.3. Securing approach....22

1. Introduction

To ensure networks and services resilience, ResumeNet's D^2R^2+DR strategy first starts with a Defence phase, happening typically during the system design. Despite a careful robust building, a large variety of factors may occur during the operations, leading to undesired effects:

- in huge complex systems, faults are always left after test actions, leading sooner or later to failures;
- abnormal (and unpredicted) environment conditions, including natural disasters, can happen unexpectedly;
- faulty human intervention may disturb the functioning of a system[1];
- attackers actions, from inside or outside, target criteria such as security, availability, …

It is therefore necessary to realize the next steps of the strategy, i.e., Detection (of a challenge), Remediation (e.g., by deploying a temporary solution in order to mitigate the damage), and Recovery (i.e., returning back to normal operations after the threat has disappeared). The Diagnostic and Refinements steps, aiming to "do better" next time can be developed further for optimization and reusability purposes.

This interim deliverable, result of the task 3.4 entitled "Service surveillance and detection of challenging situations" is about the Detection/Remediation/Recovery (DR^2) phases of the strategy. The task aims to build a framework for monitoring any service requiring a certain level of resilience. To this end, probes need to be inserted at the proper location in order to detect abnormal events. Their outputs, called *alarms*, are analyzed and treated using a correlation engine which requires as input a clear definition of challenging situations, including the resilience metrics used, for the service under study. The outcome of this analysis, i.e., a challenge detection called *alert*, will trigger remediation, e.g., with the help of policies deployment/modification. Continuous monitoring and event analysis provide information about the end of the threat, leading to actions to recover the system, bringing the service back to its normal performance operation.

As the task 2.2 concerns also challenge detection, but at a network level, a harmonization is needed. Chapter 2 describes the architecture adopted in ResumeNet for this DR^2 purpose, including the Publish/Subscribe distributed store for challenges and remediation. After a general description of the service monitoring problem, we will present the approach proposed for this task in the chapter 3. Basic monitoring principles will be then surveyed, focusing on monitoring modelling, general monitoring policy, monitoring techniques, monitoring data, and maintenance operations. The scope of events correlation, i.e., fault localization, will finally be presented. The correlation engine we will use is based on chronicles recognition, described in the chapter 4. Some applications of this technique in networks and services alarms analysis for security-related objectives (intrusion detection, reflexive DDoS attack), or dependability-related concerns (recovery actions monitoring, handover initiation in mobile systems), are then detailed.

[1] this list does not include predictable failures such as hardware ones, as the Defence phase should have taken this factor into account (e.g., through redundancy)

Finally, the Remediation phase, i.e., reaction to an alert generated by the correlation analysis through the application of a dynamic policy engine, is shown in the chapter 5 using the experimentation scenario "Communicating objects' data platform" we will deploy for the service use case of task 4.4. We will resume in the conclusion the principal contributions of this deliverable, as well as some aspects of the work planned during year 3 which will be reported in the final version of D3.2.

2. Detection and remediation architecture

As mentioned earlier, the ResumeNet project uses a common challenge identification architecture for both network and service-level resilience. This architecture is summarized in Figure 3.1. Within the project, we do not advocate the use of the same mechanisms for network and service level detection, as the nature of detection will have to be quite different. For example, for network level detection, lightweight mechanisms that operate on potentially high-volumes of network traffic will be required, whereas at the service level, alarms will be generated by detection mechanisms that typically operate on much smaller volumes of data. Of course, there are benefits to be gained from mechanisms at the network and service level interacting with each other to improve detection – a multi-level approach to resilience is one of the project's key principles. Through the use of a common architecture, we leave the exploration of these interactions available for future work.

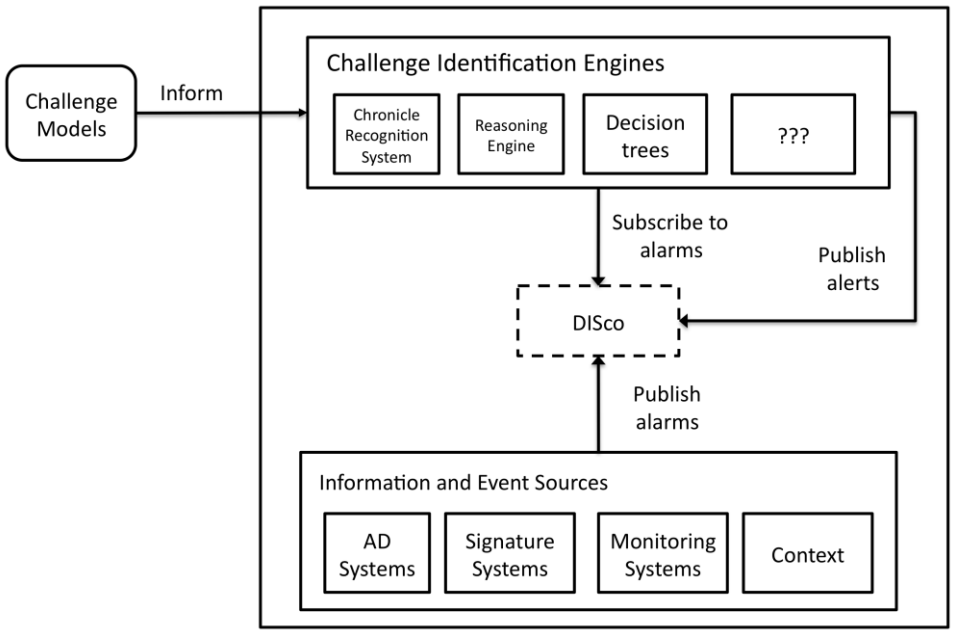


Figure 2.1: The ResumeNet challenge identification architecture

As a starting point, models are defined that represent the challenges the networked system may face, such as mis-configurations and attacks. The approach to challenge identification

presented in this deliverable advocates the use of chronicles to model challenges. Other work [Ste04a] advocates the use of directed bi-graphs to model the relationship between faults and their symptoms. Models should be developed based on the outcome of a risk assessment, such as the one advocated in ResumeNet deliverable D1.1 [Sch09].

A number of information and event sources generate alarms, which are published to a distributed information store, DISco (initially described in D2.2a [Chi10]). DISco acts as a middleware, decoupling sensors from identification engines and providing control on the amount of exchanged information independently of publishers and subscribers activities. It supports pre-processing activities such as aggregation and filtering of alarms in order to extract context from raw network status and automates bookkeeping of evidences for diagnostic and refinement (+DR) phases.

Identification engines subscribe to alarm events, which are correlated using the models previously discussed. Here, we advocate the use of a Chronicle Recognition System (CRS), discussed in Section 4. A survey of similar fault localization techniques is presented in [Ste01]. The approach the project advocates for network level challenge identification will be described in ResumeNet deliverable D2.2b, which aims to take account of the deployment constraints of, for example, the information and event sources.

3. Service monitoring

3.1. Problem description

During a service operation, supervision of its activities is a natural task needed to verify that all operations are performed as planned and its outcomes lie in the expected value window. Ensuring network optimal performance is an important key for a service provider, aiming in particular at churn reduction[2]. Network performance monitoring is mandatory for quick failure detection, so that these can be corrected as soon as possible [Chi92].

Nevertheless, correction cannot be carried out immediately. In most cases, it is not a single factor that explains all difficulties and failures, nor is there any one level of monitoring that can detect every issue. It is possible to have attackers from inside, or outside; misconfigurations of any aspect of the system introducing artificial or unnecessary bottlenecks in the network; other issues like hardware or software failures generating erroneous output; or non-intentional operational errors leading to failures. Moreover, monitoring should enable detection of out-of-date software or security measures, which can introduce vulnerabilities into the system, and program usage that goes against company policy. Due to increased number of network devices, network size, service profusion, variety of access networks and bandwidth, all these activities cannot be done by a single supervisor (administrator), i.e., an autonomous and automated system is necessary.

[2] applied to a customer base, the churn rate refers to the proportion of contractual customers or subscribers who leave a supplier during a given time period

3.2. Proposed approach

The first step in challenge detection consists in efficient monitoring. This is done with different tools processing the information received from network sensors or probes (i.e., computers, mobile phones, network devices) to check if all operations are performed as expected, inline with a list of rules:

- performance requirements, e.g., delay;
- dependability needs, e.g., availability;
- security concerns, e.g., confidentiality.

The purpose of these tools is not only to fix network problems on time, but also to prevent network failures, maintain network availability, reliability, and security and detect/eliminate internal and external threats.

In a second step, data captured by the probes, resulting in *alarms*, is analyzed. The different measurements are correlated to detect if challenges have occurred (*alert* generation) or not. The probes' data can refer to network status quo (indicating whether performance is acceptable or below a specified degree); information concerning network traffic (ensuring that some links will not become network bottlenecks); results of bandwidth analysis determining whether sufficient bandwidth capabilities exist for the service running on that network (otherwise the network cannot sustain data transfer above a certain threshold and the QoE[3] will be significantly affected).

After obtaining and transmitting the alert to a dynamic policy engine and taking into account any contextual data reflecting the state of the system at a given moment, the next step is to take optimal decisions to restore the required functionalities of the supervised network or system. A feedback loop is needed at this level to pursue context monitoring. The remediation operation will request an automated incident/intrusion detection response to threats and implies that challenge characterization is reliable and the diagnoses trustable; otherwise, remediation actions may lead to disastrous and harmful side-effects, such as self-inflicted denial-of-service.

[3] Quality of Experience represents service reliability, comprising service availability, accessibility, access time and continuity, and service comfort, reported by the end user [Mas07]

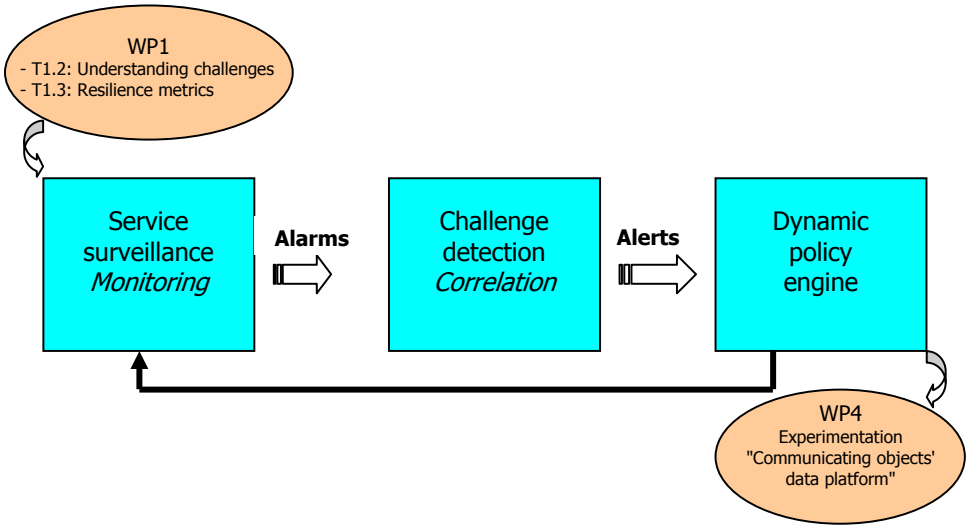


Figure 3.1: Overall description of the approach

In the context of ResumeNet (Fig. 3.1), preliminary inputs to challenge detection are obtained from WP1 (1.2 "Understanding challenges" and 1.3 "Resilience metrics"). An experimentation of this service surveillance and detection approach is done in WP4 by the way of a communicating objects' data platform (see the study described in chapter 5).

3.3. Monitoring principles

The implementation of a monitoring system must find patterns of possible faults and define what information will be regarded as an event or which particular event must be investigated. It should also specify network components to understand their functionality and role in the system, how the supervising architecture looks like (i.e., number of probes, areas of the network with the highest probability of failure), what are the probes configuration (e.g., which entries and data fields are sent to the centralized monitoring servers, what format, for these files, should be used), what are the responses to events (e.g., incident handling, operational problems if a management infrastructure component fails), what is the status of the management infrastructure (e.g., topology upgrade, infrastructure's components update), and what are the requirements for security of the management infrastructure.

After defining requirements and goals, the next steps to perform supervising and monitoring processes need to take into account applicable laws, regulations and existing organizational policies. The goals should be the reduction of risk, time and resources needed to perform management functions [Ken06]. In case of a network failure, the simplest monitoring agents should have the capacity to warn. For normal operation, these agents have to monitor constantly to seek for threats or to create reports displaying different factors with a variety of methods.

These methods are based upon time or geographic location, and improve the detection of the system parts that are causing problems. They are used to gather statistical data concerning network performance because, sometimes, an abnormal operation can be the beginning of a failure [Zea03]. Using this information, monitoring agents will hopefully detect, isolate, and/or correct the malfunctions, find the best way to keep network QoS/QoE within imposed requirements and, eventually, recover the failure.

● General policy

The best way to build a surveillance tool is to examine/analyze the application performance on a large scale. With such analysis in the context of the overall network performance, interactions between the different network levels and areas are not ignored and probable effects of further changes can be predicted. Analyzing data, received from sensors, requires enough support for the recognition of temporal components, which can be used to reduce false alerts and improve the efficiency of response to problems. Under these conditions, it is possible to ignore duplicate events and change processing rules based on the time of arrival, for example; or have the possibility to activate/deactivate rules based on the existence of other events [Rou02].

Minor events can be solved at the host level. However, they must be reported to the infrastructure level because these events can be reviewed there and help in the identification of additional instances of the same activities and patterns that cannot be seen at the individual host. At the infrastructure level, it is possible to group disparate messages from different sources into a single thread. It is possible to miss the pattern of a single event, but taken together, these patterns can indicate a problem that needs to be investigated. In these conditions, an improper grouping of events can lead to overestimating, or underestimating, the emergency severity. Moreover, analysis of the monitoring process at the infrastructure level can use more sophisticated tools and higher computational resources compared with host level [Ana04].

Performing effective review and analysis, at both host and infrastructure levels, should take into account: the organization's policies (to detect violations of these); the characteristics of common attack techniques, or common problems that might arise, and how these might be recorded by the system; the type of software used to perform analysis (e.g., log viewers, log reduction scripts, database query tools) and how these software will be configured. All the requirements must be defined clearly, must be mandatory for the entire network and must contain sufficient information, e.g., which types of events at each sensor should be recorded, how often each type of event should be logged, which type of entries should be transferred to a management infrastructure, how the confidentiality, integrity and availability (CIA) of each type of monitoring data should be assured.

● Monitoring techniques

Two techniques can be generally identified: router based (built into the routers themselves, it does not require additional installation of hardware/software, but offers little flexibility to perform checking on various critical services), and non-router based (providing more flexibility,

but requiring additional hardware/software to be installed). Some agents run a data collection script, testing processes, services and protocols.

The different types of tools used for network monitoring and analysis can be classified, depending on the data acquisition technique, as follows: analyzer/"sniffer", application/services monitoring, flow monitoring, FTP, network security, SNMP tools, topology, VoIP, finger printing, mapping, infrastructures monitoring, packet capture, path characterization, ping, Round Robin Database tool, throughput tools, Traceroute.

Some monitoring tools examine data transit through a network, looking for patterns of traffic that indicate hostile attacks, or periods of frequent use - by supervising, not only the levels of traffic, but also the specific data transmitted, the quantity of packets being sent, the source/destination, and ports of traffic, allowing for identification of policy violating traffic.

Other techniques for monitoring and analysis use local traffic flow information. In these methods, instead of requesting network devices to send the flow information to a monitoring host, it is possible to use a packet "sniffer" which collects locally the flow information. A "sniffer" can be either hardware or software, which mainly intercepts and collects the local traffic. After recording the traffic, this "sniffer" provides the function to decode and simply analyze the packet contents. Different versions of Telnet, HTTP, SNMP, FTP, and other protocols can be monitored with "sniffing" utilities such that every packet transmitted within the network is decoded and revealed.

● Monitoring data

A monitoring data file represents a record of the events occurring in the system/network, composed of information entries related to a specific event that has occurred. It can describe troubleshooting problems, or can be used for optimizing system/network performance such as operations and audits, or for demonstrating compliance with regulations, recording the actions of network's components. It provides useful data for investigating malicious activities, and contains records related to security.

Some applications generate their own monitoring data files, while others use the monitoring capabilities of the OS on which they are installed. They can monitor different types of information, such as:

- client requests and server responses (helpful in reconstructing sequences of events and determining their apparent outcome);
- user authentication or other related detailed information, e.g., recording the sender of a command, the requested address and type of response provided by the server, network's resources accessed by each host, successful and failed authentication attempts, account changes (e.g., account creation/deletion or privilege assignment), security events (i.e., brute force password access or escalation of privileges), traces of a tracked application's user, useful information such as the number of transactions occurring in a certain period (e.g., minute, hour), its size (e.g., e-mail message size, file

transfer size), system or device startup and shutdown, applications failures or configuration changes [Ken06].

New entries for monitoring data files will be generated continuously on an ongoing basis, or some sources, running periodically, can generate entries in batches, often at regular intervals.

Monitoring data are most beneficial for: (i) identifying/investigating suspicious activity involving a particular host or a particular area of the network, (ii) identifying security compromises and operational failures, (iii) producing statistics to identify the network's status. After a suspicious activity is identified, the data are checked with more details to get explicit information on the activity.

Monitoring must be configured to capture the necessary information in the desired format and locations, as well as to retain them for the appropriate period of time. Configuring data sources is often a complex process, including the choice (based on policies) of sensors, or host components, participating in the management infrastructure.

Four aspects of the data sources need to be considered [Zea03]:

- data generation, taking into account the maximum monitoring data entries to be recorded - actually, excessive monitoring can cause operational problems such as system slowdowns or even denial-of-service conditions;
- data storage - different situations can happen: (i) no storage for entries with little or no value, (ii) storage at system level only for entries that might be of some value or interest to the administrator, (iii) storage at both system and infrastructure level for entries deemed to be of particular interest which should be retained in the system and also transmitted to the management infrastructure, (iv) storage at infrastructure level only, specially when it is not possible to keep them at the system's level;
- data disposal - one important part of configuring data sources is monitoring data rotation, preferably both at a regular time and when a maximum size is reached;
- data security, both at infrastructure and system level, where CIA of monitoring data need to be guaranteed.

● Maintenance operations

To ensure a reliable monitoring process, diverse operations must be performed regularly:

- check the monitoring status of all data sources to verify that each source is enabled, configured properly and functions as expected;
- check monitoring data rotation and archival processes to ensure that data are archived and cleared correctly, and that old data are destroyed once they are no longer needed, the testing being done through automated, or manual, means of the remaining available data space;
- check for upgrades and patches for monitoring software;
- ensure that each system's clock is synchronized to a common time source so that its timestamps will match those generated by other systems;

- reconfigure monitoring as needed based on factors such as policy changes, audit findings, technology changes and new security needs, document anomalies detected in monitoring data settings, configurations and processes (it might indicate malicious activity, deviations from policy and procedures and flaws in monitoring mechanisms).

3.4. Events correlation

Having generated a complete image of the system, exploitation of correlation techniques for data taken from different sensors and filtering is critical to ensure service quality. The use of information about known vulnerabilities will also provide a faster way to respond to exceptional situations, using relationships between event patterns and actions to take. When multiple alarms overload the system, a proper event correlation mechanism may single out the alarms that represent noise, and help focus attention and prompt response only to relevant alerts.

This event correlation phase, also known as fault localization [Ste04b], is the process of deducing the exact source of the challenge (a root cause) from the set of observed alarms. Alarm correlation is the process of grouping alarms related by having the same root cause. Fault localization is subject to complications resulting from complexity unreliability, and non-determinism of communication systems. Some problems to be addressed by a fault localization technique, as proposed by Steinder and Sethi [Ste04b], are the following:

- fault evidence may be ambiguous, inconsistent, and incomplete;
- a fault management system should provide means to represent and interpret uncertain data within the system knowledge and fault evidence;
- an event management system should be able to isolate multiple simultaneous related, or unrelated, root causes;
- a fault localization should try to find the optimal solution according to some accepted optimality criteria;
- fault localization in large networks should be performed in a distributed fashion;
- a fault localization process has to take into account temporal relationships among events.

In the last item, i.e., the representation of time, events are related not only causally, but also temporally. The fault localization process has to provide means to represent and interpret the time associated with an event occurrence, as well as a technique for correlating events related with respect to the time of their occurrence and duration. Numerous paradigms have been proposed upon which faults localization techniques were based. These paradigms derive from different areas of computer science: artificial intelligence (rule-based systems, neural networks, model-based systems, decision trees, case-based systems), fault propagation models (code-based techniques, dependency graphs, Bayesian networks causality graphs, phase structured grammars), and model traversing techniques using object oriented representation of the system [Ste04b].

The mechanism we propose to use in ResumeNet is part of the first category, i.e., expert systems related techniques for challenge detection. The process is driven by an inference engine according to event-correlation rules, defined using a high level language. Rule conditions are

expressed as patterns, which test alarm frequency and origin, as well as values of alarm attributes. Tests on temporal relationships among correlated events are defined, checking whether the correlated alarms were received within a certain time-window. The next chapter describes in more detail this chronicle recognition mechanism.

4. Chronicle recognition system

Chronicle recognition approaches are challenging techniques for alarm correlation and diagnostic when on-line efficiency is required and/or when time is relevant for aggregating alarms. These approaches rely on a set of patterns, named *chronicles*, which represent the possible evolutions of the observed world. This set constitutes the chronicle base. More practically, a chronicle is a set of observable events which are time constrained and is labelled by a situation. In an alarm driven monitoring context, each (ab)normal situation corresponds to one (or more) chronicle(s); the events of a chronicle are the *alarms* and the constraints refer to their occurrence time. The chronicle recognition system (CRS) is in charge of analyzing the alarm input stream and of identifying, on the fly, any pattern matching a situation described by a chronicle.

The main positive point with these approaches is their high efficiency due to the symptom-to fault knowledge they rely on. The counterpart is the difficulty of acquiring and updating this expertise-based knowledge. Learning techniques have been proposed to remedy this problem and two kinds of techniques have been investigated. On one hand, one can process *unsupervised learning* where alarm logs are analyzed to discover frequent patterns in a data-mining way; the extracted patterns have then to be qualified by an expert. On the other hand, *supervised learning* requires to have collected positive and negative examples for each fault, but provides directly significant chronicles [Cor00].

4.1. Chronicle recognition

● Representation

While expert systems base their reasoning on rules, relegating time information to the background, recognition of chronicles is based on diagrams of evolution in which time is fundamental. Chronicles represent possible evolutions in what is observed. A chronicle is a set of events, interlinked by time constraints, and whose occurrence may depend on the context. In the supervision framework, these events are alarms referring to the supervised system and time information enables to sequence them and even to specify time spans between two occurrences. For example, in the chronicle of Figure 4.1, the alarms are partially sequenced and the temporal constraint between A2 and A5 means that it can be at least one second, but no more than three seconds, between their occurrences.

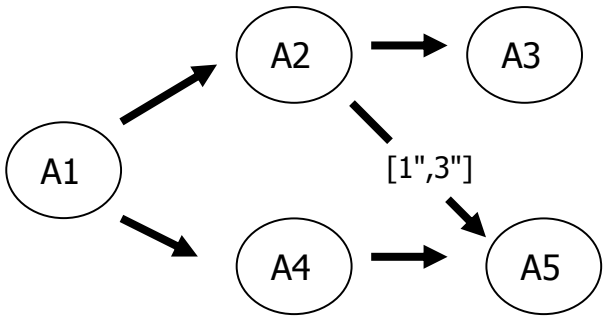


Figure 4.1: An example of chronicle

For complexity reasons, this formalism relies on time-points as elementary primitives and on the associated time-point algebra [Der82]. A time constraint between two time-points t_1 and t_2 is represented by an interval, $[I^-; I^+]$, which corresponds to the lower and upper bounds on the temporal distance from t_1 to t_2.

● Algorithms

A chronicle recognition system must detect in real-time any subset of alarms of the input stream that matches the set of constrained alarms of a chronicle model (i.e., according to all time constraints). A matching of some alarms with an alarm subset of a chronicle is a partial instance of the chronicle model. When a complete matching is found, the chronicle is recognized and the associated action (e.g., printing a message alerting the operator), if any, is performed by the system.

There are basically two ways to modify a chronicle instance: a new alarm can arrive (and can be integrated into an instance), or time passes without anything happening and makes some alarm date deadlines violated (all chronicles waiting for any alarm before this deadline will be destroyed). For this purpose, the system manages its own internal clock by receiving clock updates from the environment (e.g., from the management platform).

[Dou93] suggested an algorithm based on a complete forecast of the possible dates for each not yet occurred event; this set (called *time window*) is reduced by propagation of the dates of observed events through the graph of time constraints of the chronicle. The algorithm is incremental - each event is integrated as soon as it occurs - and is performed over a single reading of the input stream (the system does not manage a record of observed events). This method has a high-performance algorithm, partly due to a preliminary off-line compilation.

Others algorithms of chronicle recognition have been proposed. To mention only those taking a similar approach, [Fon98] presents the problem from the point of view of the compatibility of two time constraint networks, one of which corresponds to the model to be recognized and the other, the *session*, relates to the constraints between the observed events.

The approach described in [Lev94] is close to text pattern recognition techniques. It is based on a finite-state automaton, whose transitions correspond to the occurrence of events. Such algorithms are extremely efficient for recognizing sequences, but their performance is affected by quantitative time constraints and by the use of other than sequential structures. It is also very affected when events are collected in a non-chronological order (e.g., when events come from many sources of a distributed system).

Symbolic scenario recognition can also process the scenarios uniformly as a set of constraints (like the event manager of ILOG/JRules based on a modified RETE algorithm for processing time constraints [Ber02]. [Lab97] also uses a postponed verification of the chronicle time constraints for recognition.

These approaches for dealing with temporal constraints can be divided in two families: a first one (including most of them), which recognizes scenarios by an analysis of the past, and a second one (in which CRS belongs), which performs an analysis of scenarios that could be completed in the future.

4.2. Chronicle learning

● Unsupervised learning: log frequency analysis

The alarms log analysis approach is inspired by data mining techniques: it is based on a frequency analysis of actual alarm logs of the monitored system to discover some frequent chronicles [Vu99]. Identifying the most frequent patterns is relevant in order to reduce the number of alarms displayed to the operator: if it corresponds to a dysfunction, the set of alarms is aggregated before displaying it to the human supervisor; if not, the corresponding alarms are filtered. No extra knowledge is being used, i.e., the rule qualification (aggregation or filtering) must be performed by an expert.

● Supervised learning based on simulation

The supervised approach relies on a behavioural model of the system. The basic idea is to take profit of the efficiency of chronicle recognition for on-line monitoring of a network and to rely on a behavioural model of the system to automatically acquire, by learning, the set of chronicles. This approach can be applied to fault management in telecommunications networks.

The first step consists of the model simulation and aims in collecting a set of sequences of time stamped alarms resulting from well-identified faults. This set of labelled sequences constitutes the learning base. A learning module takes it as input and outputs a set of discriminating chronicles, the Chronicle Base (Figure 4.2).

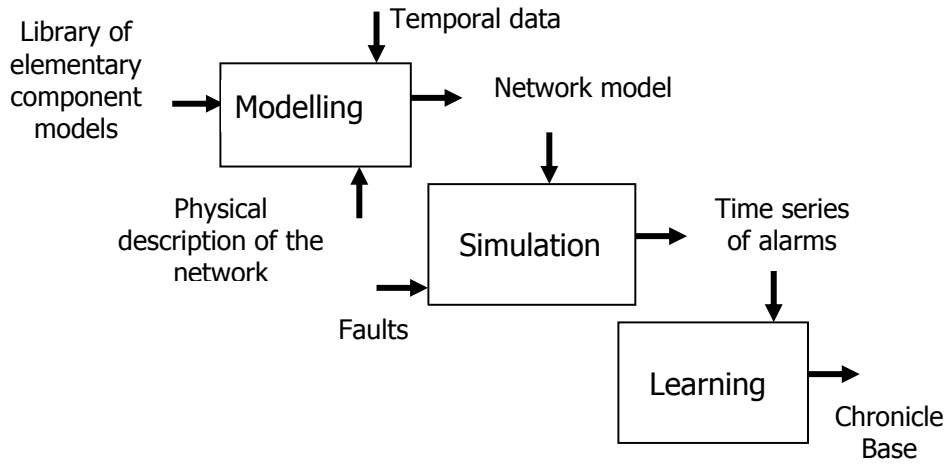


Figure 4.2: A fault management architecture in telecommunications networks

A monitored telecommunications network is made up of interconnected components. Each component can send or receive messages via its ports. To represent this communication between components, communicating finite state machines can be chosen. Each component is associated to an automaton and each connection (or channel) is included in the model.

The behavioural model of each component is described by a communicating finite state machine, whose nodes are component states and edges are state transitions. Each transition is triggered by the reception of exactly one message and is represented as follows (Figure 4.3):

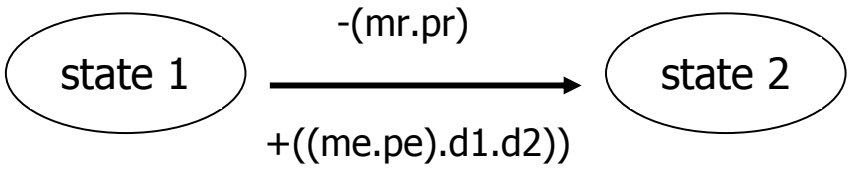


Figure 4.3: Component behaviour

It means that when the component receives a message *mr* on port *pr* at time *t*, it instantaneously switches from *state*1 to *state*2 and sends the message *me* on port *pe* at time *t* + *d* where d $\in$ [d1, d2].

Another way to automatically build chronicles from a behaviour modelling of the monitored system is to use unfolding techniques, introduced by [Mil93] in order to exhaustively identify all the possible evolution of a faulty situation. [Gue04] proposes a description of state transitions based on Petri nets with time constraints addition which allows, given a faulty state, the off-line generation of all the chronicles relevant for this fault. This approach is similar to the figure 4.2 except that the unfolding (simulation stage) exhibits immediately the chronicle base.

4.3. Experiments and applications

● Recovery actions monitoring

GASPAR is devoted to the analysis of alarms issued by the equipments of a telecommunications network [Bib96]. The supervision purpose is to monitor the behaviour of each network component when a fault occurs: the project provides a monitoring of the automatic recovery actions in order to detect when a problem is not correctly solved by the automatic procedures. When a station is dysfunctioning, a backup one is automatically activated to perform the desired service (1 backup for N stations). The chronicles detect when this backup works properly and when the process is aborted (i.e., the backup station is not activated). Other chronicles survey also when the nominal station works again (i.e., when the backup station could be switched off): this last operation is performed by a human operator. The main characteristic of this work is to take profit of the efficiency of a chronicle-based recognition for the on-line alarm analysis, and to propose an off-line automatic acquisition of the chronicle base by simulating the model and applying Inductive Logic Programming (ILP) learning techniques.

● Intrusion detection

A multi-alarm misuse correlation (i.e., known malicious or undesired sequences of events are searched in the data stream) component based on the chronicles formalism allows to reduce the number of alarms shipped to the operator, and enhances the quality of the diagnosis provided. It has been shown that the use of chronicles for alarm correlation in IDS has solved some of intrusion detection issues like alarm overload, false positives and poor alarm semantics [Mor03]. This project relies only on the recognition process, no learning feature was studied since the chronicles (intrusion signatures) are written by a human expert.

● Handover initiation

An optimally working mobile system requires tight cooperation and an information stream that flows impeccably between its components [Dou07]. It is not usually the case as applications and system components must each, in isolation, acquire data, create a knowledge base, and maintain it with diligence. In the scope of the Ambient Networks project [Nie04], chronicle recognition was introduced in a trigger management framework, allowing the generation of synthetic triggers based on events and measurements. These triggers can be generated efficiently, and are reliable in terms of event temporal correlation. The main objectives of using CRS here is to detect specific context-sensitive situations relevant to the handover. The recognition of chronicles gives richer information to the knowledge base, which is used by other entities. As for the previous project, chronicles are written by a human.

● Traffic supervision

For the supervision of an IP core network, one should consider huge amount of data and high rate (several dozen of thousands events per second). In order to be able to process efficient and quick on-line and real-time analysis, it is necessary to have the capability of control on the input event streams in order to focus the processing mainly on the relevant data. [Dou09] has introduced a protocol devoted to give control and focus capabilities to the analysis module, depending on its current needs. The flow rate reduction allows more efficient and less resource consuming processing. This technique has been applied to the signature recognition of a reflexive DDoS attack on a server. The attack is done through servers (called naïve), which are infected and so, send erroneous requests to the targeted (victim) server; as naïve servers are numerous, the victim is overloaded by all these wrong requests. The objective is not to detect when this attack takes place, but to identify the naïve servers responsible for it. This work is devoted to drastically improve the recognition performance.

5. Sketching remediation in a service use case

We will exploit for this use case a technical platform built in the framework of the French national project ICOM (see ResumeNet Deliverable D4.1b for more details), and allowing exchanges between applications through heterogeneous hardware and software. This intra/inter-enterprise infrastructure links various identified objects (RFID, 1D/2D barcodes, NFC, …) to a company information systems and fixed/mobile terminals and/or, to a lesser extent, the objects to each other.

This multi-networks and multi-communicating objects middleware will allow real-time processing of large volumes of information, i.e., event exchanges related to the objects with the applications and their information systems located locally, or not, and belonging, or not, to the distributor company. The numerous applications are related to logistics (parcels follow-up, inventory, …), sales functions (items traceability), or marketing (merchandising).

5.1. Architecture of the communication objects' data environment

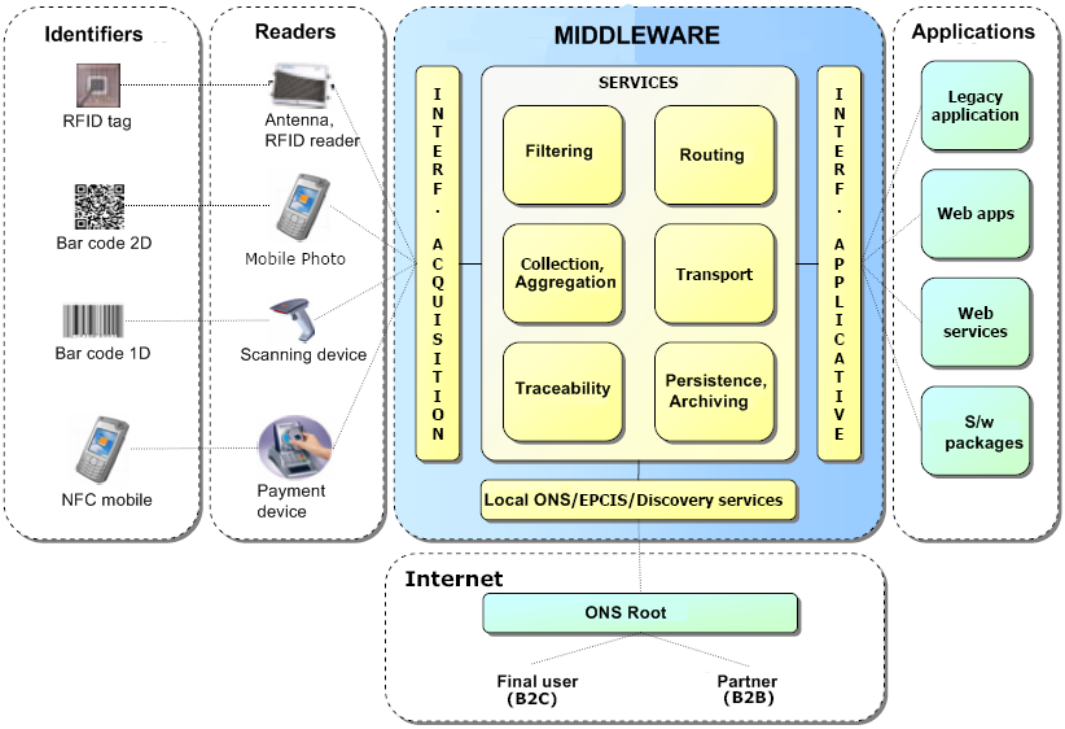


Figure 5.1: ICOM architecture

Figure 1 describes the two main parts of the ICOM project: in the physical area, the objects are identified through their label (RFID, 1D/2D barcodes, NFC) by mass-market readers (cameras embedded in mobiles) or dedicated terminals (RFID antenna/reader, reader device, payment terminal) and the data are sent to the logical area. Based on an open source RFID software platform and EPCglobal standards (ALE[4], EPCIS[5]), the middleware consists of different modules:

- *collection* and treatment by *aggregation* → the acquisition/formatting of large data volumes allows the unification of different label types;
- *filtering*, *routing* and *transport* based on the contents in order to send only useful information and adapt them to the application requirements (see below);
- *traceability*/*persistence* for adding to the objects' observation events business information, process verification and *archivage* in databases;
- linked to an ONS Root, a distributed objects' name server (*local ONS*) processes the partners queries (B2B) or customers queries (B2C), guiding them to the requested services.

The ICOM middleware is finally interfaced with various enterprise applications: Web/existing applications, Web services, ...

4 Application Level Events (http://www.epcglobalinc.org/standards/ale)
5 EPC Information Services (http://www.epcglobalinc.org/standards/epcis)

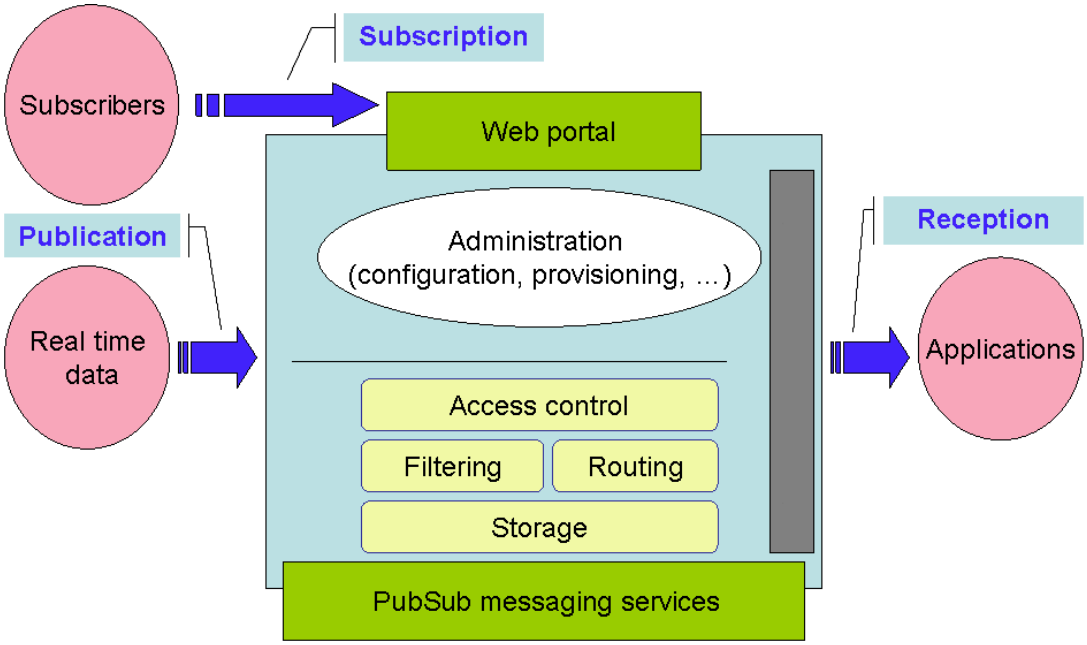


Figure 5.2: Principal functions of the data platform

The part of ICOM we are dealing with concerns the filtering functions and routing information with the use of a PubSub-based platform decoupling message senders and recipients (Fig. 5.2). This platform is based on a network of XML routers using hardware to process messages and allowing very high performance, the network covering itself a network of (traditional) IP routers.

Through this platform:

- contents-based routing is done from selection filters (subscriptions) of XML documents through the shortest path to end nodes before distribution to the recipients;
- each company can manage two types of users (data publisher/sender, data subscriber/client) and closed user groups;
- publishers broadcast their messages to authorized recipients;
- subscribers only receive relevant data, thanks to contents filtering (subject, EPC code, …), through chosen communication channels (Web service, Java Message Service, RSS, SMTP, SMS, ...);
- messages are stored until they have been issued;
- in addition to access control for both publishers and subscribers, a Web portal manages the administrative tasks (client configuration management, accounts provisioning, ...).

5.2. PubSub platform experimentation

Figure 5.3 describes the on-going experimentation for the ICOM project [Lac10]:

- site 1 is the source of data, i.e., a hypermarket in which a panel of volunteers feeds the event contents through their purchases - the items, identified by barcodes, are read by mobile phones and the information, sent by WiFi to a LAN, are routed through a secure VPN link to a processing center located in the remote site 2;
- different modules of the ICOM architecture are housed in the site 2 (collection and processing by aggregation, tracking/persistence, ...) which, after receiving the information, queries the hypermarket database to retrieve various information about the items (price, ...) before transmitting them for publishing to the PubSub platform;
- site 3 hosts the platform which constitutes the core of our study; some basic security mechanisms are already implemented: authentication, firewall, ... - this is also where the data subscriber applications are linked (not represented in the figure).

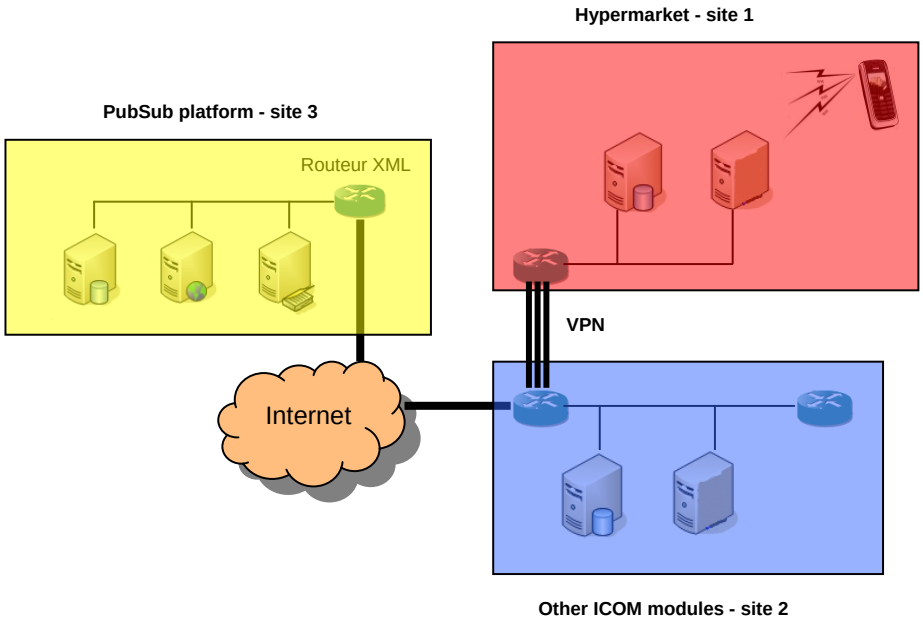


Figure 5.3: Diagram of the ICOM experimentation

5.3. Securing approach

The changing environment of the PubSub platform (multiple events from various sources, frequent access by new customers, ...) requires a dynamic security policy. This need is also motivated by the diversity of threats facing the platform. The response to these threats can be

achieved by applying a security policy using contextual rules, allowing both to specify the system normal behaviour and its nominal behaviour in the presence of threats. The process includes several steps.

● Specification of the nominal behaviour of the system, i.e., the normal security policy applied to the platform in the absence of any external risk:

- the legitimate contract administrator is allowed to create subscriber/publisher accounts related to one (or several) closed user group(s) (CUG);
- registered subscribers have the permission to receive data sent to the platform by publishers belonging to their CUG;
- legal publishers can publish in the platform interesting data for the members of their CUG;
- authentication of the platform's legitimate users (contract administrator, subscribers, publishers) is done via an LDAP directory server.

● Specification of the policy in reaction to threats, i.e., safety rules related to threat contexts. The specification of these rules is as follows:

- analysis of the risks facing the platform in order to define a policy reaction following a challenge detection; these risks include criteria such as:
  (i) confidentiality, e.g., (a) bypassing password check or publishers/subscribers filter - this can involve both a legitimate user or an intruder (unknown from the system) through mechanisms such as "sniffing", (b) having been able to guess the contract administrator's password, a registered subscriber may change its membership to a CUG and get data published for someone else, or an illegitimate user can create a subscriber account to receive published data;
  (ii) integrity, e.g., forgery, at the PubSub service provider level, of data transiting through the XML routers - having been able to guess the contract administrator's password, a publisher may change its registered membership to a CUG and publish false information, or an illegitimate user can create a publisher account to broadcast false information;
  (iii) availability, e.g., DDoS attack against a Web server of the platform or accidental failures of non-redundant equipment (XML router, identification server, ...).
- definition of threat contexts characterizing the identified risks, and activated when events related to such risks are detected. We note that the concept of threat context for threat response is different than contexts for alert correlation and detection diagnosis. Actually, context-based alert correlation introduces contexts as the interrelated conditions in which an alert is triggered, and which characterize the existence, or occurrence, of certain events, i.e., challenges. These contexts are created by certain rules defined within the alert correlation module, due to the execution of specific rules. Contexts thus act as event stores, and events can be added to contexts as they occur. However, Or-BAC contexts are used in a different way: these do not constitute event stores, and are not created by specific rules. Or-BAC contexts are

actually used to describe environmental parameters, and characterize the global conditions under which the target system evolves at a given time. Or-BAC introduces normal contexts (e.g., working time, spatial contexts, etc.). It also introduces threat contexts that make precise the events threatening the normal system behaviour, and which require actions to be taken by the system in order to counter those events (e.g., DoS, network congestion, temporary failure, etc.). In order to characterize the threats defined within the risk assessment process described in the previous paragraph, we introduce the following threat contexts:

(i) to guess the password (of the contract administrator, of another user, …), brute force attack can be deployed, hence the definition of context "brute_force" to be activated in this case;

(ii) the interception of data transmitted over a network ("sniffing") requires the opening for listening purpose of a suspicious port - "port_opening" is defined to characterize this threat context;

(iii) a DDoS attack is commonly expressed by an abnormal bandwidth occupation, coupled with the context "service_denial";

(iv) in the case of an accidental equipment failure, the context includes the lack of response from the monitored network elements[6], a situation one could associate with the context "equipment_failure".

- definition of actions to undertake, as contextual rules, in the case the threat contexts are activated, e.g.,

(i) to cope with a brute force attack, the immediate deactivation of the account experiencing the aggression is necessary;

(ii) upon detection of a port opening, two actions, focusing on the attacker or the system, are possible: block the suspicious port and/or switch the current sessions to a safe mode such as the use of the SSL protocol, e.g., transition from http (port 80) to https (port 443);

(iii) restrict the use of resources by prohibiting access to the platform for new users/customers is a natural reaction in case of DDoS attack;

(iv) the implementation of passive redundancy (setting up a second less efficient server, for example) can compensate hardware/software failures.

● Definition of the mapping functions connecting the threat contexts to certain attributes of alert messages. These alerts specify the platform components related to the ongoing threat: these components are the only ones concerned by the activated challenge context. For threats caused by internal clients of the platform, only these customers will be subject to countermeasures.

Access control policies include both permission and prohibition rules which are applied to *subjects* seeking to carry out *actions* on *objects*. Some rules specify the criteria to be applied during nominal operations, forming the *operational policy*. Other rules are only applied in the

[6] Nagios (www.nagios.org) is the type of tools one could use for this purpose

case of threats against the system, composing the *reaction policy*. The transition from one type of policy to another is triggered by contextual constraints.

We have chosen for this study the OrBAC (Organization Based Access Control) model [Abo03] to specify the contextual policies. This security policy formalism proposes to separate the mechanism of context activation from the one of associated rules activation. For this purpose, OrBAC defines the predicate *hold* connecting a context to a triplet (*subject*, *action*, *object*). The context activation is reduced to the instantiation of the *hold* predicate for the adapted triplet.

Concerning the PubSub platform, the activation of *hold* predicates is reduced to the verification of certain attributes in alert messages identifying the current challenge. Therefore, to characterize the threat contexts, *hold* predicates are proposed, e.g.,

- brute force attack:

  hold (any, login, target_account, *brute_force*) :-
  alert (source, target, classification),
  map_service (target.service, login),
  map_account (target, target_account),
  map_context (classification, "x login failures")

- port opening:

  hold (subject, connect, wifi access, *port_opening*) :-
  alert (source, classification)
  map_subject (source, subject)
  map_context (classification, "open tcpdump")

- service denial:

  hold (any, service, target_host, *service_denial*) :-
  alert (source, target, classification),
  map_service (target, service),
  map_hostname (target, target_host),
  map_context (classification, "flood attack")

- equipment failure:

  hold (any, any, service, *equipment_failure*) :-
  alert (target, classification),
  map_context (classification, "server down");
  map_target (target, failed_service),
  passive_redundancy (failed_service, service).

The *hold* predicates retrieve some attributes from the alert messages using static mapping functions. These relate the elements that constitute the triplet (*subject*, *action*, *object*) to information contained in the alert message. Following the activation of these threat contexts, the appropriate security rules to be triggered are the following:

prohibition (attacker, connect, wifi access, *port_opening*)
hold (subject, connect, wifi access, *port_opening*) ==>
prohibition (subject, connect, wifi access)

prohibition (any_user, authenticate, account, *brute_force*)
hold (any, login, target_account, port_opening) ==>
prohibition (any, login, target_account)

prohibition (external_user, connect, wifi_access, *service_denial*)
hold (any, service, target_host, *service_denial*) ==>
prohibition (any, connect, wifi_access)

obligation (administrator, activate, redundant_service, *equipment_failure*)
hold (any, any, service, *equipment_failure*) ==>
obligation (admin, activate, service)

● Disabling threat contexts after a latency phase depending on the level of risk that activated this context: a classification of threat contexts, according to the precariousness of the risk associated with each of them, needs to be established.

For each threat context, one needs to define both the disabling condition and the appropriate verification. Indeed, following a context activation, the corresponding disabling condition must be monitored. For this purpose, a checking rule is applied: if the rule is satisfied, the condition for disabling the threat context is considered "done" and the context is inactivated (Table 5.1).

Context	Disabling condition
port_opening	Condition: the attacker IP address is no longer recognized in the network Verification: a ping of the attacking machine should not get response
brute_force	Condition: the account under a brute force attack is disabled Verification: check activation parameters of the target account in the authentication server
service_denial	Condition: the bandwidth occupation is again below the threat threshold Verification: obtain a new occupation index of the bandwidth (e.g., using Nagios)
equipment_failure	Condition: the redundant machine was switched on Verification: positive response to a ping request for this machine

Table 5.1: Disabling conditions for threat contexts

● Deployment of the dynamic security policy, preceded by a translation phase of this policy into a set of configurations that apply to the various checkpoints (routers) of the platform, e.g.,

- for an LDAP authentication server and in the case of brute force attack, i.e., *prohibition (any, login, target_account)*:
  Ldap: access to dn.base = "target_account" by dn.base = "admin_contrat_account" none
- for a gateway to access to a network implementing Netfilter and in the case of port opening, i.e., *prohibition (subject, connect, wifi access)*:
  iptables -t filter -I INPUT -i subject -j REJECT
- for a gateway to access to a network implementing Netfilter and in the case of service denial, i.e., *prohibition (any, connect, wifi_access)*:
  iptables -t filter -I INPUT -j REJECT
- for a server reboot in Linux environment and in the case of equipment failure, i.e., *obligation (admin, activate, service)*:
  sudo \etc\init.d\redundant_service start

6. Conclusion

This interim deliverable, result of the task 3.4 entitled "Service surveillance and detection of challenging situations" deals with the Detection/Remediation/Recovery (DR^2) phases of ResumeNet's D^2R^2+DR strategy.

After a short justification of the DR^2 steps in a resilience strategy, we have described our project's challenge identification architecture used both for the network level, and the service level. It is made of four principal building blocks: (i) the understanding of challenges through different types of modelling; (ii) the information and event sources, including the monitoring systems (probes, …) and the environment context; (iii) a set of various possible engines for challenge identification; (iv) a distributed information store based on the PubSub paradigm.

The approach followed for the monitoring, the challenge detection, and the remediation for a communication service is then detailed. Diverse basic monitoring principles have then been surveyed. Our intention for this interim deliverable is not to describe in detail each of the items mentioned in this section, and exhaustivity of the monitoring principles is not the objective. The aim is to give a short insight of some working paths to be followed for building the final version of the deliverable.

The natural task coming after a service monitoring job is the events correlation. This phase, also known as fault localization, is based on numerous paradigms derived from different areas: artificial intelligence, fault propagation models, and model traversing techniques using object oriented representation of the system. The technique we plan to adopt for this project is part of the expert system techniques for challenge detection. Chronicle recognition system is then described, as well as some experiments/applications in the domain of alarms treatment: (i) recovery actions monitoring, (ii) intrusion detection, (iii) handover initiation, (iv) traffic supervision.

Finally, one of the service use cases to be exploited in WP4, a communication objects' data environment, was the subject of a study about remediation[7], i.e., after an alert has been issued by the correlation engine.

[7] As reported in ResumeNet deliverable D6.4c [Fes10], the work in Task D3.4 has been delayed, following the resignation of one of FT's main contributors. As a consequence, the illustration of the use of a challenge identification engine (by the way of the chronicle recognition system) as input for this remediation phase has not been initiated yet. It is part of the work programme planned for year 3 of the project, and will be reported in the final version of this deliverable.

References

[Abo03] A. Abou El Kalam et al., Organization-Based Access Control, IEEE 4th International Workshop on Policies for Distributed Systems and Networks (Policy), Lake Como, Italy, June 4-6, 2003, pp. 120-131

[Ana04] K.G. Anagnostakis, S. Ioannidis, S. Miltchev, and J. Ioannidis, Real-time log file analysis using the simple event correlator (sec), 18th USENIX System Administration Conference (LISA), Atlanta, GA, USA, November 14-19, 2004

[Ber02] B. Berstel, Extending the RETE algorithm for event management, 9th International Symposium on Temporal Representation and Reasoning (TIME), Manchester, UK, July 7-9, 2002

[Bib96] S. Bibas et al., Alarm driven supervision for telecommunication networks: I- Off-line scenario generation and II- On-line chronicle recognition, Annals of Telecommunications, Vol. 51, N° 9-19, 1996, pp. 493-508

[Chi92] D.M. Chiu, and R. Sudama. Network monitoring explained: design and application, Prentice Hall, 1992

[Chi10] L. Chiarello, S. Martin, M. Fischer, M. Fry, G. Popa, C. Rohner, and P. Smith, D2.2a: First draft on new challenge detection approaches, ResumeNet Deliverable, February 2010

[Cor00] M.-O. Cordier, and C. Dousson, Alarm driven monitoring based on chronicles, 4th Symposium on Fault Detection Supervision and Safety for Technical Processes (SAFEPROCESS), Budapest, Hungary, June 14-16, 2000

[Der82] D.V. McDermott, A temporal logic for reasoning about processes and plans, Cognitive Science, Vol. 6, 1982, pp. 101—155

[Dou93] C. Dousson, P. Gaborit, and M. Ghallab, Situation recognition: representation and algorithms, 13th International Joint Conference on Artificial Intelligence (IJCAI), Chambéry, France, August 28 - September 3, 1993, pp. 166-172

[Dou07] C. Dousson, K. Pentikousis, T. Sutinen, and K. Mäkelä, Chronicle recognition for mobility management triggers, IEEE Symposium on Computers and Communications (ISCC), Aveiro, Portugal, July 1-4, 2007

[Dou09] C. Dousson, and P. Le Maigat, Controlling the event streams to become autonomous, 5th International Conference on Autonomic and Autonomous Systems (ICAS), Valencia, Spain, April 20-25, 2009

[Fes10] A. Fessi, M. Karaliopoulos,, C. Lac, M. Schöller, and P. Smith, D6.4c: Third periodic progress report, ResumeNet Deliverable, March 2010

[Fon98] D. Fontaine, and N. Ramaux, An approach by graph for the recognition of temporal scenarios, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 22, N°3, 1998, pp. 387-403

[Gue04] B. Guerraz, and C. Dousson, Chronicles construction starting from the fault model of the system to diagnose, 15th International Workshop on Principles of Diagnosis (DX), Carcassonne, France, June 23-25, 2004

[Ken06] K. Kent, and M. Souppaya. Guide to computer security log management - Recommendations of the National Institute of Standards and Technology, NIST Special Publication 800-92, September 2006

[Lab97] P. Laborie, and J.P. Krivine, Automatic generation of chronicles and its application to alarm processing in power distribution systems, 8th International Workshop of Diagnosis (DX), Mont Saint-Michel, France, September 15-18, 1997, pp. 61—68

[Lac10] C. Lac, N. Kheir, and B. Delosme, Securing a communicating object data platform, LambdaMu 17, La Rochelle, France, October 5-7, 2010 (in French)

[Lev94] F. Lévy, Recognising scenarios: a study, 5th International Workshop of Diagnosis (DX), New Paltz, NY, USA, October 1994, pp. 174-178

[Mas07] L. Mason, T. Drwiega, and J. Yan, Managing traffic performance in converged networks, 20th International Teletraffic Congress, Ottawa, Canada, June 17-21, 2007, pp. 17-21

[Mil93] K. McMillan. Symbolic model checking: an approach to the state explosion problem, Computer Science, Carnegie Mellon University, 1993

[Mor03] B. Morin, and H. Debar, Correlation of intrusion symptoms: an application of chronicles, 6th International Symposium on Recent Advances in Intrusion Detection (RAID), Pittsburg, PA, USA, September 8-10, 2003

[Nie04] N. Niebert et al., Ambient networks: an architecture for communication networks beyond 3G, IEEE Wireless Communications, Vol. 11, N°2, April 2004, pp. 14-22

[Rou02] J.P. Rouillard, Efficient packet monitoring for network management, 8th IEEE/IFIP Network Operations and Management Symposium (NOMS), Florence, Italy, April 15-19, 2002

[Sch09] M. Schöller, and P. Smith, D1.1: Understanding challenges and their impact on network resilience, ResumeNet Deliverable, August 2009

[Ste01] M. Steinder, Fault localization in communication networks: a survey, Technical Report 2001-01, CIS Department, University of Delaware, February 2001

[Ste04a] M. Steinder, and A.S. Sethi, Probabilistic fault diagnosis in communication systems through incremental hypothesis updating, Computer Networks, Vol. 45, N°4, July 2004, pp. 537-562

[Ste04b] M. Steinder, and A.S. Sethi, A survey of fault localization techniques in computer networks, Science of Computer Programming, Vol. 53, 2004, pp. 165-194

[Vu99] T. Vu Duong, and C. Dousson, Discovering chronicles with numerical time constraints from alarm logs for monitoring dynamic systems, 16th International Joint Conference on Artificial Intelligence (IJCAI), Stockholm, Sweden, July 31 – August 6, 1999

[Zea03] S. Zeadally, E. Yaprak, Y. Li, and X. Che, A survey of network performance tools for computer networking classes, Computers and Advanced Technology in Education (CATE), Rhodes, Greece, June 30 - July 2, 2003