PANOPTESEC

FP7-610416

D2.1.1. - Deficiencies Evaluation

Work-Package	WP2	Deliverable	D2.1.1
Due Date	M6	Submission Date	2014-04-28
Main Author(s)	Nicolas Prigent (Supélec)		
Contributors	All project participants		
Version	V1.0	Status	Final
Dissemination Level	PU	Nature	R
Keywords	State of the Art, Deficiencies		

Part of the Seventh

Framework Program

Funded by the EC - DG Connect

Table of contents

Executive Abstract: In this document, we present the deficiency evaluation of the current automated cyber-defense decision support.

This objectives of this study is to provide directions for the PANOPTESEC project. These directions are presented in the Conclusion and Further Work section.

1 Context

Organizations have become increasingly dependent on networks and computer systems to support their business operations and services. As this dependency has grown, so have the motives and capabilities of cyber-attackers. Despite security measures and regardless of their motives, cyber-attackers are often able to penetrate networks and computer systems to extract valuable information (violating confidentiality), tamper with the information (violating integrity) and overload or otherwise prevent access to and systems services (violating availability).

Therefore, continuous vulnerability assessment and remediation as well as operational incident response capabilities are nowadays identified as not only useful but necessary. While software solutions and best practices are available, the numerous successful attacks that happened in the recent year have proved them insufficient. The objective of the PANOPTESEC project is to deliver a beyond-state-of-the-art prototype of an automated cyber-defense decision support system to demonstrate operational use of Dynamic Risk Approaches for Automated Cyber-Defense algorithms, architecture and design. PANOPTESEC will deliver this capability through an integrated and modular, standards-based integration of technologies that collectively deliver a beyond-the-state-of-the art capability to address cyber-vulnerabilities and incidents in real-time.

The expected PANOPTESEC organization is sketched in Figure 1.

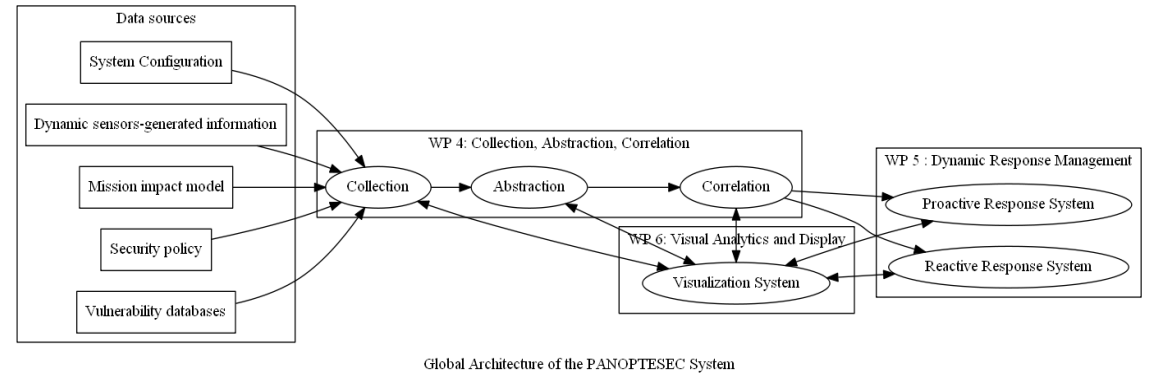


Figure 1 Organization

Five data sources will be used:

- The *system configuration* that describes the observed system, i.e. the network topology, the connected devices, the services that run on them, the users, the physical location of the entities, the physical security measures, etc.
- The *security policy* that describes the various permissions in the observed system.
- The *mission impact model* that describes the level of importance of each device and service.
- The *vulnerability databases* that contain information about the known vulnerabilities, whether they be present in the observed system or not.

- The *sensor-generated information* that is made of the various information dynamically created by the sensors that are deployed in the observed system and that describe what happens in it in real time.

Information from the five data sources are *collected*, *abstracted* and *correlated* to generate a consistent and concise enough view of the observed system and of the events that are happening on it.

This refined information is then used by the *proactive response system* that ensures that the system is properly configured and tries to minimize the risks. The *reactive response system* furthermore takes into considerations the evolutions of the system (and particularly, the alerts) to react accordingly to dynamically minimize the risks, for instance by modifying the system configuration.

Finally, the *visualization system* handles the interactions with the user. It reports the relevant information and allows the control of the various components of the system.

In this document, we analyze and present the current deficiencies that exist in the state of the art of automated cyber-defense decision support system. For each of the components of the architecture, we first present the state of the art. We then present the current deficiencies both in terms of products and research. Based on these deficiencies, we finally propose directions for the PANOPTESEC project.

2 Data Sources

In this section, we present the five types of data sources that will be used in the PANOPTESEC project. For each of these types of source, we first propose a description. We then present the state of the art. Finally, we present the current limitations.

2.1 System Configuration

The *system configuration* data source provides information about:

- The network topology.
- The devices (e.g. including OS and patch level).
- The applications and services.
- The users (who are they, where are they located, what are their roles).
- The physical layout (server room location and geographical position for instance).
- The authorities on each piece of equipment.

If the system configuration information can be provided manually by the administrator, it is nevertheless a complex and lengthy process. It is also error prone and leads to errors when the system evolves without the administrator knowing it (in case of attacks for instance, when an attacker installs a rogue server). Consequently, system configuration information is currently frequently built at least partially automatically.

In order to build the system configuration, two non-exclusive approaches can be used: the active approach and the passive one. In both cases, it is required to verify regularly the freshness of the information contained in the database.

The *active approach* can first consist in deploying agents on the monitored network. A software agent located on machine accesses to its configuration and the list of installed software. The various agents can be interrogated remotely to obtain the local information. For example, Nagios[1] allows deploying daemons on machines. Each of these processes provides the local information required to build the

[1] http://nagios.sourceforge.net/docs/nagioscore/3/en/activechecks.html

whole system configuration. The agents must often have administrator privileges in order to access relevant information. The fact that an attack can be conducted against an agent is a major risk and consequently it is the main drawback of this approach.

The active approach can also consists of interrogating machines and the services they host by sending packets to these services and by observing the replies. These answers, or the absence of an answer, infer information about the deployed software and on their configuration (for example, the port on which they are listening). The work of [50] relies on this type of approach to determine the local network topology. The main drawback of this approach is that it can disturb the machines and services that are interrogated.

The passive approach appears as an alternative to the active approaches that may permit to correct their drawbacks [33]. The passive approach relies on capturing of the packets exchanged in the network. Such observations allow identification of the topology of the network, the machines present in the network, the deployed services, their versions, etc. Several tools are available to perform a passive discovery of a network topology: p0f[2], EtterCap[3], Sourcefire RNA[4], etc. Of course, the drawback of the passive approach is the risk of erroneous interpretations of the captured data.

In the specific context of SCADA systems, it is generally admitted that the passive approach has to be chosen. In fact, these systems being known to be quite touchy from a robustness perspective, active system configuration discovery through messages could in fact lead to instability. Due to the specificities of these systems, they also do not accept general purpose monitoring agents to be run on them.

Current challenges and objectives of the PANOPTESEC project

Label	Importance	Difficulty
Collecting and maintaining safely system configuration in the context of SCADA systems is challenging. The PANOPTESEC project should propose some acquisition mechanisms for that purpose.	3	2
Managing the various system configuration description in a conceptually unified way is currently challenging, especially when SCADA systems are involved. This aspect will be handled by the PANOPTESEC project thanks to carefully crafted ontologies.	3	3
It is difficult to be confident about the system information that are collected when at least part of the system can be corrupt. The PANOPTESEC project will address this aspect through information aggregation and correlation.	3	3

[2] http://lcamtuf.coredump.cx/p0f3/

[3] http://ettercap.sourceforge.net

[4] http://www.sourcefire.com/security-technologies/cyber-security-products/3d-system/network-awareness

2.2 Security Policy

The Security policy shall describe the rights of the users, of the devices, what are the authorized network flows, what are the critical parts of the system. It constitutes a fundamental aspect of PANOPTESEC.

Access control is a security fundamental concern. It covers a wide area of applications. Using high-level statements, access control models allow a flexible and easy expression of security policies. The task of generating and managing such policies is not trivial.

Traditional ICT systems base their security policies on a number of existing models, providing more or less expressiveness and capabilities. In the historical and widespread Discretionary Access Control model (DAC), the owner of a resource has the right and power to define the access rules applying to it. Therefore, the overall policy of the organization is tributary to the unregulated decisions of individual users. Such a security model may be viable for single-user, personal platforms, but are usually avoided for professional information systems, since they do not allow the building and maintenance of a consistent policy across the system. The Mandatory Access Control (MAC) is a response to this imperfect situation, in the sense that it introduces the notion of a security administrator, a particular subject with the authority to define a policy then enforced by the system for all users. This basic principle may simply be implemented as a set of constraints over the discretionary power of users, or as the one source of information about access control, as in the multi-layer security models (MLS) with which it is sometimes mistaken for. In essence, the mandatory aspect of a security model is simply the ability to design impose a global security policy. Although it is an obvious requirement for PANOPTESEC as for any large-scale information system, this concept must be associated with others (related to expressivity and administration, for instance) to design an actual security model.

One of the simplest models for defining and administrating a security policy is probably the HRU matrix model. The said matrix puts in correspondence objects (resources) with subjects (users), each correspondence being associated with a set of rights related to access control and ownership. In a given system, an original access matrix is associated to a set of transformation functions, describing in which conditions and in which manner the access control matrix can be modified. A major problem of this apparently convenient model is that the problem of reachability of a given state of the matrix is undecidable, which makes it impossible to be certain that an undesirable situation will actually be forbidden by the policy. Even with this theoretical limitation aside, the administration of such a system proves inconvenient, being based on the explicit manipulation of all subjects and objects by an all-knowing administrator. This kind of shortcoming can be overcome by adding a level of abstraction to the rules, which can be done in several ways.

This is the main objective of the Role-Based Access Control (RBAC) model [92] and its variations (like attribute-based access control [93]). The principle is to abstract subjects or objects by assigning classes to them, and then manipulate primarily those classes when designing the policy. This principle is now widespread in many information systems and more complex security models, and several variants of RBAC have been given formal implementation specifications by the NIST.

Organization-Based Access Control (OrBAC) [94] generalizes this abstraction concept by dividing policies between an abstract layer, where security administrators define the general rules regulating access control and other security properties in terms of roles, views (abstracting away objects) and activities (abstracting away actions), and a concrete layer, in which a reasoning engine instantiate the actual rules applying to the objects, subjects and actions it has become aware of. The set of these relations allow security administrators to define organizations and sub-organizations, and to set up the management of the policies along the lines of a structured representation of human and architectural entities.

Another significant feature of OrBAC is the contextualization of its access control models, e.g., the ability to grant access based on the dynamic activation of system changes. This trend, usually referred to as context-aware access control models, concerns the enhancement of RBAC in order to incorporate

different criteria in access control decisions, rather than just which user, having which role, is performing which action on which data object.

The OrBAC model provides a robust modelling solution to address access, network and usage control policies in traditional IT systems. It is based on a very expressive and modular formalism that enables system administrators to separate the definition of security requirements from their concrete enforcement. To do so, it introduces a new entity, referred to as context, in which three abstract entities, namely roles, views and activities, represent the abstraction of the classical triple of elements in any access control entities (e.g., subjects, objects and actions).

The context entity in the OrBAC model provides the conditions in which a security rule must be activated. This allows the definition of dynamic security rules. It also provides to the model the concept of organization. The organization projects a well-organized group of active entities in which roles, views and objects are played by specific subjects, objects and actions.

The abstraction added by the OrBAC model aims at reducing the system administrator effort. Instead of directly defining concrete security rules, whose number in large organizations can be huge, it requires the definition of abstract rules, whose number is expected to be rather low. In the end, and based on an automatic derivation process, abstract rules are transformed into concrete rules, according to the set of active contexts. In the sequel, more information is provided.

2.2.1 OrBAC Security Rules

An OrBAC security rule indicates if a given subject has the permission to perform an action upon a given object. A security rule is modelled in OrBAC as a set of premises (i.e., subject, action and object) and a decision (e.g., either grant or deny access). The following definitions present the common concepts to be taken into account.

- *Subject:* Is the entity (e.g., person, process, user account) to which access to an object shall be granted.
- *Object:* Is the resource to which access shall be granted (e.g., a file or a directory).
- *Action:* Is the operation performed by a subject over an object (e.g., read, write, execute over the object).

The above three entities are referred as concrete entities, since they concretely interact with the access control mechanisms. Some more definitions follow.

- *Access Control Decision:* There are two fundamental conclusions that can be given upon an access control request: either to allow or deny it. Naturally, there might be many other decisions. In any case, it is important to note that the set of decisions shall always be a finite set.
- *Concrete Security Rule:* At the concrete level, a security rule describes relations among subjects, actions, objects and access control decisions. For instance, a given rule (*s*, *a*, *o*, *accept*) states that a subject *s* is granted access to perform an action *a* upon object *o*.

We assume that directly describing concrete security is a very hard and error-prone task. Rather, we assume that defining the security policy is done via the definition of high-level statements to express general security rules, regardless of the enforcement of the policy. These statements consist of an abstraction of premises. The goal is to reduce the number of rules. The following definitions state the necessary elements to define such abstract-level security rules:

- *Subject Abstraction:* Subjects can be abstracted using one or more of their attributes or using the role they are playing in the organization where the policy is being defined. We denote *Role* to such set of attributes.

- *Action Abstraction:* Actions are abstracted using one or more of their attributes or using the role they are playing in the organization where the policy is being defined. We denote *Activity* to such set of attributes.
- *Object Abstraction:* Objects are abstracted using one or more attributes or using the role they are playing in the organization where the policy is being defined. We denote *Activity* to such set of attributes. We denote *View* to such set of attributes.

After defining abstract entities, relations between abstract and concrete entities are implicitly or explicitly shall be expressed. The association between concrete entities and abstract entities is characterized by: (1) a many-to-many mapping of subject-to-role assignment relation; (2) a many-to-many mapping of action-to-activity assignment relation; and (3) a many-to-many mapping of object-to-view assignment relation. As a result, we assume finite sets of subjects assigned to each role; finite sets of actions assigned to each activity; and finite sets of actions assigned to each activity.

High-level statements must be encoded as abstract security to rules, according to the aforementioned mappings. Only premises are abstracted. The concrete decisions remain unchanged, but applied to the abstract elements. Therefore, at the abstract level, a security rule describes the relation between abstract elements and access control decisions. Furthermore, abstract elements can be organized in either flat or hierarchical structures. In case of hierarchical structures of abstract elements, the hierarchy relations among abstract elements shall be characterized as follows:

- Many-to-many mappings to declare role-to-role hierarchy relations.
- Many-to-many mappings to declare activity-to-activity hierarchy relations.
- Many-to-many mappings to declare view-to-view hierarchy relations.

Hierarchy structures are expected to simplify the security policy definition, since the final derivation into concrete security rules will benefit from inheritance mechanisms. In other words, we assume the following inheritance cases:

- Let r_2 be a sub-role of role r_1, then r_2 will inherit all abstract rules of r_1.
- Let a_2 be a sub-activity of activity a_1, then a_2 will inherit all abstract rules of a_1.
- Let v_2 be a sub-view of view v_1, then v_2 will inherit all abstract rules of v_1.

Finally, in case of using the aforementioned hierarchies, the security rules will be assumed to be divided in two groups: local rules and inherited rules.

2.2.2 Applying OrBAC in PANOPTESEC

The OrBAC model provides the necessary elements to express in a unique and unambiguous manner the security requirements expected by the PANOPTESEC system. It provides, as well, the necessary mechanisms to lead the response system of the PANOPTESEC, in terms of policy updates, refinement, enforcement and redeployment. As indicated in a companion deliverable D2.2, the PANOPTESEC system must provide a *response system* that must evaluate potential threats and attacks, assist administrators to select *mitigation actions*, and actuate by activating, e.g., the appropriate reconfiguration processes. The use of contextual OrBAC policies allows policy-driven reconfiguration, based on the following two steps: (1) an *instantiation* process, based on the activation of the appropriate contexts (e.g., a given threat or attack context); and (2) a *technical refinement* process, based on administration-driven templates. The latter can be complemented by an *enforcement process*, in order to communicate the appropriate package of configuration rules to the system reconfiguration process. A more in detail description of the two main processes follow.

The instantiation process involves the necessary derivation of security concrete rules. It establishes the mapping between the system's security reaction policy (defined a priori by the security operators of the system) and the use of contextual information, such as incoming attack alarms from SIEM devices,

potential cyber security attack paths that can affect the security of the system, or any other vulnerability advisory information. In the end, a dynamic activation of response actions will eventually occur by transforming the abstract representation of the policy into the concrete rules expected to interact with the security devices and mechanisms existing in the system. We refer the reader to [94], [95] and citations thereof, for a more detailed explanation about the instantiation process associated to the OrBAC model via contextual activation of abstract rules.

The technical refinement process is in charge of translating the set of concrete rules that will later be communicated to the security component of the systems. The process can be seen as an iterative set of transformations, each complying with a given security technology instantiated in the system (e.g., firewalls, routers and SIEMs). Each transformation can be conceived as the instantiation of those parts of the security policy that must be enforced in the system by means of the processed components. The global policy is, therefore, decomposed into several packages of local configurations. Several proposals in the literature have been proposed for the implementation of this functionality. Some sample solutions are presented in [96] and [97], in order to refine OrBAC policies into packages of instructions for firewalls and routers.

2.3 Mission Impact Model

In order to provide decision support for managing a critical infrastructure, e.g. in a SCADA setting, the presentation of explanations for recommendations presented to system analysts is mandatory. In particular, if trade-offs are considered between the removal of vulnerabilities and the provision of services, it is important for system analysts to understand the impact of actions on the overall goals of the computational system. These overall goals are represented in the form of so-called missions at some level of reasonable detail, and it is assumed that system analysts are trained such that they are familiar with those missions. Given these assumptions it is a reasonable requirement that recommendations for security maintenance activities are explained to analysts by describing the trade-offs w.r.t. the above-mentioned missions. Missions can be seen as “goals” or “commitments” of organizations to provide “services”, which, in turn, rely on “assets” such as, for instance, computational devices with IP address and port. For instance, if a mission is defined for committing to maintain a certain pressure level (the service) at certain water distribution point, the mission is (potentially) impacted if the pressure level cannot be controlled due to the shutdown of certain computers (or ports) being part of the (local) control system. The question is how to quantify such an impact in a reasonable way.

Current decision support systems are based on numerical scores, aggregating “importance values” of assets and services in different ways. Obviously, in this setting, computing “best actions” can be formalized as optimization problems for finding the actions that minimize the corresponding numerical scores. Unfortunately, only in rare cases can system analysts understand the advice for executing particular actions (e.g., applying a patch while a certain system is shut down) on the basis of the impact on numerical scores. Other actions not proposed might seem as suitable alternatives, and hence, rather than just recommending a particular action, it might be important to also communicate why other actions are ranked down. In order to mitigate the explanation deficiency of current security decision support systems, and to provide this kind of information, a so-called mission impact model is required.

Although basic data structures for mission impact modelling have been proposed [61], existing approaches to mission impact modelling [58], [59], [60], [62] provide just a starting point, since a formalization of the impact is not provided. Business process model provide the potential for a “deeper” model on how services contribute to a mission than simple is-composed-of-like modelling structures. As usual, the challenge here is to minimize modelling overhead.

There are approaches such as, e.g., [63], for assessing security risks for services in networked scenarios using quantitative measures. However, these kinds of models need to be complemented with information about a more high-level impact on the overall purposes of services, namely the mission of the organization providing services. Mathematical properties of scores used in engineered solutions

are rarely investigated, and, thus, often, mission impact modelling is intermingled with risk estimation in monolithic approaches. Risk estimation by definition is related to prediction of (potential) security incidents (and their proactive avoidance) whereas mission impact assessment also applies to reactive scenarios in which situation awareness is important.

A classic approach for assessing security risks of individual assets is generating detailed attack graphs [64]. However, a common deficiency of these methods is that they require a fully observable network. Thus, mission impact modelling should further be exploited for extending attack graph generation to hidden network configurations, i.e., an open world. Further, the selection of entry points is a crucial step in attack graph generations, for which the mission impact might be exploited.

While risk assessment approaches take into account the "value" of certain assets, which, in turn, contribute to the assessment of tasks or services, it is clear that value of assets can hardly be defined in a bottom-up fashion. Indeed, those values depend on the importance for a certain mission a particular organization complies to. Thus, mission impact modelling should be exploited for risk assessment (or risk estimation) in the predictive case as well as for assessing the situation in a reactive case. With mission impact modelling, risk assessment or situation awareness becomes more modular.

Furthermore, temporal aspects need to be taken into account. If vulnerabilities are (assumed to be) removed due to the execution of certain actions, the corresponding impact values or the resulting risk estimates are not immediately reset but just "decay" over time since passwords might not be changed immediately after a vulnerability is removed and unauthorized access might still be possible as an effect of the previous exploitation of a vulnerability).

Current challenges and objectives of the PANOPTESEC project

Label	Importance	Difficulty
New approaches for mission impact modeling need well-analyzed and streamlined data such that a definition of required notions for mission impact modeling can be derived at some suitable level of abstraction automatically (e.g., mission, tasks, services, assets, etc.).	3	1
Current approaches are of a monolithic nature and are not easily transferable to other domains, thus a modular architecture is required such that possibly parts (modules and models) can be reused in other contexts (e.g., a separation between mission impact modeling and risk quantification is to be achieved).	3	2
The state of the art for quantifying either mission impacts or risk estimates mostly uses numerical worst-case scoring values. This leads to a loss of dependency tracking possibilities such that explanations for decision making are hard to provide. The investigation of potential benefits of ontology-based solutions for logical impact modeling, i.e., for techniques beyond simple scoring approaches is a challenge.	3	3
Mission impact information (e.g., priorities on missions) should be used to determining suitable asset rankings based on expected scores to incorporate more information available in a particular situation.	2	3
Business process models are to be investigated in order to possibly improve dependency models (mission, tasks, services, assets), i.e., the benefit of primitive recursion and while constructs from business process models might be exploited for	1	3

mission impact modeling and risk estimation [61].		

2.4 Vulnerability Databases

The *vulnerability databases* describes the various known threats that exist against IT systems. Many vulnerability databases are available, some being them free, others not. The most common free/openly accessible vulnerability databases are The NIST *National Vulnerability Database (NVD)*[5], the Mitre Corporation *Common Vulnerability and Exposure (CVE)*[6], the *Open Source Vulnerability Database (OSVDB)*[7], Symantec *SecurityFocus*[8] and *Secunia advisories community edition*[9].

While these databases exhibit different interfaces and declare different numbers of vulnerabilities, they globally provide the same kind of information under comparable formats. Each vulnerability is described by a name, the vulnerable target (device/OS/application), the vulnerable versions, the effects of the vulnerability, the required privileges and situation (local, distant, etc.) and a severity. The severity is often computed according to the effect of the vulnerability and the required privileges. For instance, a local transient DoS is of small severity while a distant vulnerability providing privileged access is of high severity.

While using more than one database could seem interesting in order to have better coverage, it also induces many drawbacks. First, most vulnerabilities are in fact available in most of the databases, while having different names. A partial solution has been found *de facto* by using the CVE identifier in most of the vulnerability databases. Second, the slight differences between vulnerability descriptions make it difficult to create a common format to unify all the advisories. To that end, NVD proposes the SCAP (Security Content Automation Protocol) format suits which is sometimes considered heavy to use.

A second challenge when using vulnerability databases consists of not being overwhelmed by the amount of vulnerabilities that nowadays cannot be handled manually. A first step consists of selecting the vulnerabilities that are relevant to the current system. This requires having a precise knowledge of the devices/OS/applications that are deployed in the system (cf. section 2.1). While numerous proposals have been made, correlating vulnerability descriptions with the system description is still mostly a challenge.

A third challenge, finally, consists of generating the correct detection rules for IDSes based on vulnerability descriptions. Indeed, some databases propose sample exploits that are often used as-is to create detection rules. However, in many cases, these detection rules have been specifically tuned against the sample exploits for marketing purposes without totally encompassing the full extent of the threat. Consequently, slightly modified attacks still go undetected.

We should also mention that in the specific case of SCADA system, most of the vulnerabilities are kept secret due to the criticality of these systems. Some conference presentations have for instance recently been cancelled[10].

Current challenges and objectives of the PANOPTESEC project

[5] https://nvd.nist.gov/
[6] www.cve.mitre.org
[7] http://osvdb.org/
[8] http://www.securityfocus.com/vulnerabilities
[9] http://secunia.com/community/advisories/
[10] http://www.lemagit.fr/actualites/2240215853/CanSecWest-Eric-Filiol-renonce-a-son-intervention (in French)

Label	Importance	Difficulty
While many vulnerability databases are available, it is still difficult to choose which one proposes the most valuable information. Often, interesting fields are in fact in plain text and cannot be automatically processed. The ontologies proposed in the PANOPTESEC project should help in defining the most valuable information in our context and how to present them.	2	3
Vulnerability databases for SCADA systems should be investigated.	2	2

2.5 Sensor-generated Information

The *sensor-generated information* is made of automatically generated information that dynamically describes the current events on the observed system. It describes IT events such as user connections, device reboots, AV alerts, IDS alerts, but also physical world event such as the fact that someone entered legitimately or not a secure room, that a USB key was inserted in a server, etc.

In this section, we present three types of sensors (host-based, network-based and intrusion detection systems). By contrast with the first two sensors, intrusion detection systems do not only log what happens on the system but also analyze it to detect attacks or anomalies and subsequently emit alarms.

2.5.1 Host-Based Sensors

Host-based sensors are sensors located on a given host that collect information about the events currently happening on it. This operation is also referred to as *logging*. Each event is time stamped and logs that are generated are most often stored into a file. We should however underline that a recent trend consists in collecting logs in noSQL databases such as Splunk[11] or loggly[12].

Two complementary kinds of host-based sensors coexist: *Application-level sensors* and *OS-level sensors*. *Application-level sensors* only collect information from a given application. They can themselves be a distinct application or can be a software module included in the application itself. When the application generate its own logs (i.e. when the sensor is a software module), it is possible to provide information that is internal to the application (state of a given internal variable for instance) that it would not have been to collect from the outside of the application. However, the drawback of this approach is that the numerous existing format makes it sometimes difficult to handle them consistently. Furthermore, the semantic of the information it contains can be very specific to it.

By contrast, *OS-level sensors* gather information at the OS level. As a consequence, they consider the host globally and are often more coarse-grain. They often offer little information about the internals of the applications that caused a log entry to be generated. In order to offer a more consistent management of the various logs that are generated, standards are proposed in the current operating systems: the Unix based systems (Linux and Mac OS X for instance) use the syslog[13] format while MS Windows systems use the Windows Event Log API[14].

One of the most important challenge dealing with host-based sensors deals with how their outputs are managed: where are the outputs send? How are they protected? How to prevent an attacker from

[11] http://www.splunk.com/
[12] https://www.loggly.com/
[13] http://en.wikipedia.org/wiki/Syslog
[14] http://msdn.microsoft.com/en-us/library/windows/desktop/aa385780%28v=vs.85%29.aspx

accessing to them or from modifying them to cover their tracks? In other words, outputs of the host-based sensors have to be securely collected and protected in a way comparable to the other information in the system.

A second challenge currently consists in handling consistently the outputs of various sensors. While parsing and handling each file independently is currently reachable, arbitrarily correlating the information contained in files generated by multiple sensors is still quite difficult and requires skilled analysts to analyze the various pieces of information.

Current challenges and objectives of the PANOPTESEC project

Label	Importance	Difficulty
Collecting and storing securely the outputs of host-based sensors is particularly difficult, particularly when hosts can be compromised. An objective of the PANOPTESEC project will be to design a secure mechanism to collect and store logs from host-based sensors.	2	2
Correlating outputs from different host-based sensors is still an open problem. We expect that results from Semantic science (cf. Section 2.7) will help in this way.	3	3

2.5.2 Network Sensors

Network sensors (also called *network probes*) are used in information systems to capture network traffic, to reassemble applicative sessions, to analyze the traffic and possibly record, save or export this traffic. Some products provide network packet capture and/or network traffic analysis functionalities (often called Deep Packet Inspection). We should mention that while intrusion detection is not the primary functionality of such product, they are often used to feed IDS[15]. In fact, vendors of network probes often cite IDS (or security threat identification) as one of the use case of their product[16].

Description

Figure 2 illustrates the different functionalities of network probes and NIDS/NIPS. We try to distinguish functionalities that are implemented by almost all products (solid-border boxes) from those that are optional (dash-border boxes). Moreover, thick-border boxes illustrate the main functionalities.

We then describe in more details the two main functionalities of network probes, i.e. **capturing** and **decoding** network flows.

[15] http://home.regit.org/2011/09/oisf-brainstorming-planning-phase-3-take-3/

[16] http://www.ipoque.com/en/products/pace

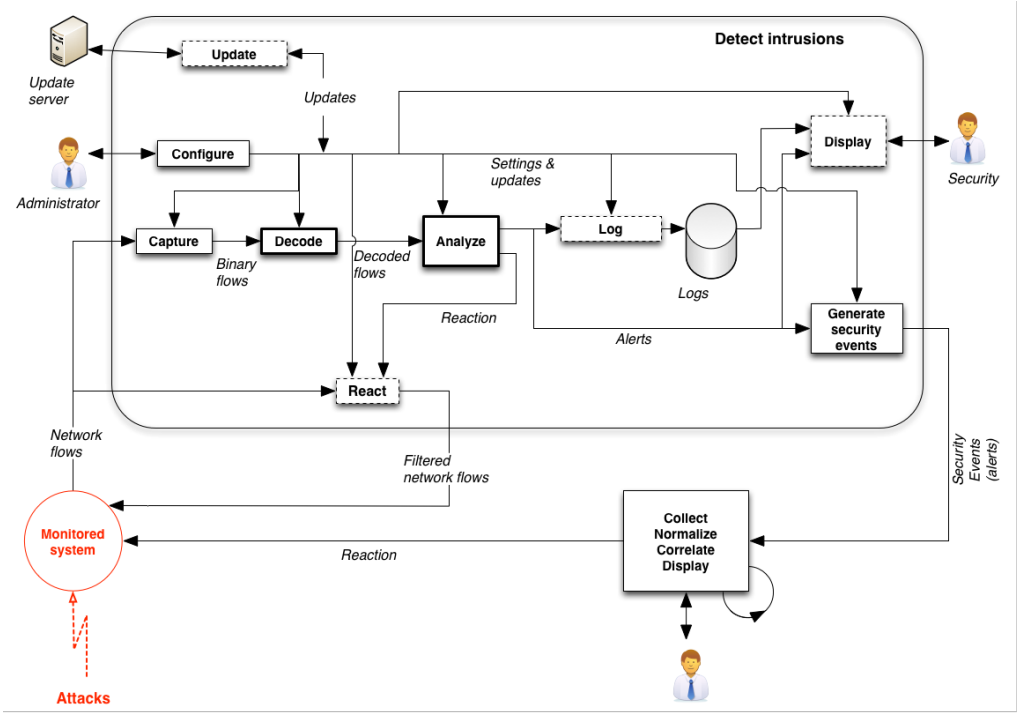


Figure 2: Functional description of network probes

The first functionality consists of capturing network flows. Network probes can capture two different types of network flows:

- Raw packets directly captured on network links or devices;
- Aggregated flows, sampled traffic and network statistics that can be provided by network devices (switches, routers, firewalls, etc.) or dedicated probes using IPFIX[17]/NetFlow or sFlow[18] technologies.

Products available on the market are often dedicated to one or the other of these types of input. Some products can handle both of them.

Products using the second type of network flows implement a so-called flow collector[19]. Compared to raw packets, sampling and aggregation reduce the amount of data that need to be collected and then analysed. This type of solution is very appealing when used in high-speed network (network core, ISP, etc.). However, a large part of the information contained in packet payloads is lost.

To capture raw packets, network probes use hardware devices such as classical network interfaces or high performance capture cards[20]. They can implement two different capture modes:

- Passive capture consists of sending a copy of each packet to the network probe;
- Inline capture consists of making the traffic flowing through the probe.

[17] http://tools.ietf.org/html/rfc7015, http://tools.ietf.org/html/rfc5103
[18] http://www.sflow.org/
[19] For example, ntop : http://www.ntop.org/solutions/flow-based-monitoring/
[20] For example, Endace DAG : http://www.emulex.com/products/network-visibility-products-and-services/endacedag-data-capture-cards/

Passive capture is achieved by setting the capture card in a special mode (promiscuous mode[21]) to capture all traffic it receives. In this case, the network probe is quite stealthy (no IP address is bind to the capture interface). Copies of network packets are provided by another network device[22]:

- The probe can monitor communications between a set of devices connected to the same hub or switch (using port mirroring)
- It can also monitor communications between two nodes using a network tap[23].

The choice of solution to capture network flows depends on several criteria: reaction behavior, impact on network performance (delay, throughput), traffic rate and architectural considerations.

Typically, inline capture mode is preferred when the probe needs to react to intrusions (IPS) as it can alter or delete packets. The probe could also use this mode to normalize the capture traffic and thus to defend itself against evasion attacks [70], [73]. However, the inline capture mode intrinsically alters network performances due to the packet processing delay. Moreover, a probe failure can induce network failures.

On the other hand, passive mode is stealthier and does not alter the performances or the availability of the monitored network. Network taps can achieve high-speed full duplex capture with no data loss. They have few impacts on the monitored network but are only suited to monitor a point-to-point link. Thus, monitoring large networks using network taps can require a lot of monitoring devices. Using port mirroring (SPAN or RSPAN) is a cheaper and often simpler alternative to monitor all the communications inside a given network (for example, a DMZ). However, this solution can suffer from bandwidth limitation and data loss.

The second major functionality of network probes consists of decoding binary flows. It is highly valuable to decode binary flows before analyzing them for the following raisons:

- The analysis often concerns data disseminated in a sequence of packets (e.g. a TCP flow) and cannot be performed packet per packet.
- Invalid packets will not be taken into account by the receiver and should not be analysed. However, they can be symptomatic of some attacks such as evasion attempts.
- The analysis can concern only some specific part of the captured flows such as specific protocol fields or applicative contexts (for example, the URL of HTTP requests) that need to be identified.
- Some fields or applicative contexts can be encoded differently depending on the protocol stack implementation. Normalizing them simplifies their analysis.

The different steps of the decoding rely on *a priori* knowledge concerning the specification of network protocols used in the captured flows. Thus, the number and the type of protocols that are supported by the probe constitute important choice criteria.

To decode binary flows, it is necessary to first identify the different protocols that are used and encapsulated in each packet. This helps to delimit the boundaries of protocol fields inside each packet. Identifying protocol can be challenging when encapsulation or proprietary protocol are used. For example, relying on well know TCP or UDP port used by convention is not a robust solution [74]. For complex protocols such as TCP, decoding also consists of verifying that the sequences of packets satisfy protocol grammars usually modelled as state machines.

[21] http://en.wikipedia.org/wiki/Promiscuous_mode
[22] http://www.ntop.org/pf_ring/port-mirror-vs-network-tap/
[23] http://en.wikipedia.org/wiki/Network_tap

By decoding the packet, the probe could check the conformance to protocol specification to identify invalid packets (e.g. ill-formed packet, packet with wrong field values or out-of-sequence packets). These invalid packets should not be analysed and can be noticed by raising alerts.

To help the analysis, decoding should also manage session reconstruction and defragmentation. Indeed, some applicative context could be spread among different packets or fragments. Fragmentation can appear at different levels:

- The Internet Protocol natively implement datagram fragmentation (see RFC 791[24] and RFC 815[25]);
- The Transmission Control Protocol provides reliable delivery of a stream of bytes between programs in connected mode. This flow is broken into a sequence of TCP segments that are encapsulated in IP datagrams. This mechanism is transparent for applications. Thus, a given application request or respond could be split into several TCP segments.
- Some applicative protocols such as DCE/RPC could also implement fragmentation at the application layer[26].

Finally, the probe can normalize some applicative contexts (for example HTTP URL[27]) before passing them to the analysis.

Products

The following products provide generic network probes that are not included in NIDS products but that could be used in security monitoring contexts:

- Traditional network tools such as tcpdump[28], Wireshark[29], netsniff-ng[30], NetScout Sniffer[31] or Microsoft Message Analyzer[32];
- nTop products[33];
- Qosmos DPI probe[34];
- Bivio DPI[35];
- Ipoque PACE[36];
- Cloudshield products[37];
- Solera Networks products[38];

[24] http://tools.ietf.org/html/rfc791
[25] http://tools.ietf.org/html/rfc815
[26] http://pubs.opengroup.org/onlinepubs/9629399/chap12.htm
[27] http://en.wikipedia.org/wiki/URL_normalization
[28] http://www.tcpdump.org/
[29] http://www.wireshark.org/
[30] http://netsniff-ng.org/
[31] http://www.netscout.com/products/enterprise/Sniffer_Products/
[32] http://blogs.technet.com/b/messageanalyzer/
[33] http://www.ntop.org/
[34] http://www.qosmos.com/products/deep-packet-inspection-probes-for-security/
[35] http://www.bivio.net/products/dpi/
[36] http://www.ipoque.com/en/products/pace
[37] http://www.cloudshield.com/products/technology/deep-packet-processing/

- cPacket products[39];
- Narus products[40];
- nPulse products[41].

Future work

Generally speaking, the main challenges concerning the capture and decoding are the following:

- Capturing and decoding traffic in high-speed network [68], [69];
- Protocol identification, especially when tunneling or encryption is used[42];
- Managing specific and undocumented protocols [75];
- Evasion attacks that to try to deceive the decoding [71], [72].

In the context of the PANOPTESEC project, we want to investigate the problematic of capturing and decoding traffic from ICS/SCADA environments. This is challenging as a lot of specific and often undocumented protocols are used in those environments. We plan to rely on automatic protocol reverse engineering tools such as Netzob[43] to automatically generate protocol parsers for network probes.

Protocol identification could be quite difficult to achieve. However, in the specific context of ICS, we can assume that the monitored system is under control, which limits the need to identify tunneled traffic.

Label	Importance	Difficulty
Capturing and decoding traffic in high-speed network	2	1
Protocol identification	1	2-3
Managing specific and undocumented protocols	3	3
Protection against evasion attacks	2	2

2.5.3 Intrusion Detection Systems

In PANOPTESEC, *Intrusion Detection Systems* (IDSes) are considered as data sources that provide information about the current attacks against the observed system.

Intrusion detection systems [158] can be categorized according to at least two features [27]:

- The location of the observation network-based intrusion detection (NIDS) VS host-based intrusion detection (HIDS)
- The reference they use to detect attacks, whether it is a base of known attacks (misuse-based intrusion detection), a base of know correct behavior (anomaly-based intrusion detection).

In this section, we successively describe NIDS and HIDS.

[38] http://www.soleranetworks.com/products/security-analytics/platform/
[39] http://cpacket.com/products/
[40] http://www.narus.com/solutions/narus-nsystem
[41] http://www.npulsetech.com/products/
[42] http://gray-world.net/projects/papers/html/cctde.html
[43] http://www.netzob.org/

2.5.3.1 Network-based intrusion detection

Network IDS aim at detecting attacks or intrusions by monitoring network traffic. To do that, they rely on different functions:

- They **capture** network traffic and provide them as binary blobs to the decoding and analyzing functions.
- They **decode** binary flows to identify the different protocol fields, to follow protocol sessions and to verify the compliance with protocol specifications.
- They **analyze** decoded flows to detect symptom of attacks or intrusions and raise alerts if necessary.
- They manage alerts:
- They can **log** them locally.
- They can **generate security events** and send them to other security devices such as SIEM.
- They can **react** by modifying or blocking malicious flows.
- They can **display** alerts (products often provide basic security visualization functionalities).
- They provide functionalities to **configure** the probes (managing signatures, setting parameters, etc.).
- They often provide functionalities to automatically **update** the product using external servers (belonging to the client organization, provided by the vendors or third parties).

The main functionalities are thus to capture, decode and analyze network flows. The two first functionalities are described in section 0 as generic network probes also provide them.

Analyzing network flows to detect intrusions is the main functionality of NIDS probes. In theory, such products are supposed to detect intrusions, i.e. effective violations of the security policy as, for example, confidential data leakage, modification of system configuration or denial of service. In practice, NIDS are often not able to distinguish failed attacks from successful ones, i.e. intrusions. Thus, NIDS can often only detect attack symptoms.

As mentioned before, the different intrusion detection approaches can be classified into two distinct families: misuse-based [77], [78] and anomaly-based approaches [79]. Most of the NIDS products implement misuse-based approaches. They rely on signatures of previously known attacks. These signatures are often expressed as a set of patterns[44] and the detection engine use well-known pattern-matching algorithms to detect some protocol fields or applicative contexts of the monitored traffic match these patterns. Some approaches rely on a more complex multi-events detection. In this case, signatures consist in several patterns linked by conjunction, disjunction, sequence, or counting operators [80], [81]. This could be seen as a form of correlation implemented at the probe level.

Some products implement anomaly-based approaches [82], [83]. They often rely on machine learning to infer a statistical or rule-based model of the legal behavior of the monitored system. Then, the detection engine compares the monitored traffic to this model and reports any deviation. This supposes to define an adequate similarity measure and to fix thresholds. Anomaly-based approaches could also rely on rules that are manually specified to describe the normal or legal behavior[45]:

- Policy-based approaches detect network flows that are not allowed by the security policy (for example the use of P2P protocols or connections to forbidden hosts)

[44] See, for example, the Snort signature language :
[45] For example using Bro IDS : https://www.bro.org/

- Specification-based approaches [84], [85] detect protocol anomalies by checking the conformance of the monitored traffic to protocol specifications.

NIDS products can also implement some specific and *ad hoc* techniques. For instance, sandbox analysis consists in identifying some specific contents (typically files) in the monitored traffic, to extract them from the traffic and to analyze them offline using typical anti-malware approaches[46].

Some products rely on reputation mechanisms provided by third parties [86].

2.5.3.2 Host-based intrusion detection

By contrast with NIDS, host-based intrusion detection systems try to detect intrusions by observing what is happening on hosts. To that end:

- They **monitor** what the files and events.
- They **analyze** these files and events to detect symptom of attacks or intrusions and raise alerts if necessary.
- They **manage** alerts:
- They can **log** them locally.
- They can **generate security events** and send them to other security devices such as SIEM.
- They can **react** by modifying or blocking malicious actions or putting malicious files in quarantine or even deleting them.
- They can **display** alerts (products often provide basic security visualization functionalities).
- They provide **configuration** functionalities to **configure** the probes (managing signatures, setting parameters and sensitivity levels, etc.).
- They often provide functionalities to automatically **update** the product using external servers (belonging to the client organization, provided by the vendors or third parties).

By contrast with NIDS that only can monitor network activities, HIDS run on hosts and are therefore able to consider what is happening on them and had no symptoms on the monitored networks. They are also not impacted by the fact that malicious communications may be encrypted. However, since they run on hosts, HODS are vulnerable in case of successful attacks: as soon as an attacker gained control of the host they run on, NIDS cannot be trusted anymore.

Anti-viruses are currently the most common HIDS available. The monitor the file system and the memory to detect malware. They all propose signature-based detection systems: anti-virus systems have a base of known malware signature to which it matches the various files that appear on the monitored system. While this approach is often quite efficient from a performance perspective (numerous large files can be checked in a limited file), it can only detect malware that is already known and for which a signature was made available. As a consequence, new virus are not detected and it is possible for an attacker to modify specifically a given malware to prevent detection.

Due to this fact, most anti-virus now propose other approaches. First, they can monitor file modification in a way similar to what tripwire [47] does. They also propose “heuristic” approaches that complementarily detect malware by monitoring what each piece of software does on the system. If the anti-virus detects that a piece of software has a specific behavior that seems malicious, it makes the hypothesis that it is malware. Currently, this approach is quite inefficient and only a few very specific malicious activities are detected. Furthermore, malicious activities are detected only when they already occurred. To limit these consequences, some anti-viruses propose sandbox detection: a new piece of

[46] For example, using Razoback : http://sourceforge.net/apps/trac/razorbacktm/
[47] http://sourceforge.net/projects/tripwire/

software that may be malicious is first launched in a sandbox to observe its behavior and the modifications it makes to the system. If it is malware, it will not be able to harm the system and anomalous behavior should be detected. While this approach is interesting, it is still vulnerable to sandbox detection and sandbox evasion.

Aside anti-viruses, other HIDS are available that monitor anomalous behaviors on systems. They can either collect information on the system by themselves (files, RAM, network communications, system calls or state of the kernel) or monitor the various logs that are generated.

2.5.3.3 Products

According to the NIST reference document [48] and technical data sheets provide by vendors, NIDS probes are actually branded into different types of products:

- Few products, which are often open-source, are still marketed as software. It is then necessary to install them on dedicated platform (including common OS, network cards or dedicated network capture interfaces, etc.).
- The vast majority of commercial IDS probes are now marketed as dedicated hardware products called *appliances*. These products are identified as IPS but they can be considered as NIDS that can block or alter malicious traffic. Most of them provide both passive and inline capture mode. The reaction functionality can be deactivated so that the product can be used as a pure NIDS. Editors often provide product-related services such as product and signature updates.
- Some products, which are similar to IDS, are marketed under different names. For example, ***Network Behavior analyzer*** can be seen as anomaly-based network IDS. They often rely on the analysis of network flows using IPFIX, Netflow[48] or similar technologies.
- Some vendors provide IDS probes for a specific domain. For example, some of them are dedicated to the wireless networks and are called *Wireless IPS*.
- Some vendors provide an IDS plugin for their networking hardware (switches, hubs, routers, firewalls, etc.). This plugin can be provided as additional hardware or software. They could also simply consist in specific license tokens that activate this functionality.
- Some vendors have generalized this approach by providing integrated security hardware solutions. These products, often named as ***Universal Threat Management*** appliances, embed several security mechanisms such as firewall, NIDS/NIPS, anti-malware, proxy, etc. They also provide other functionalities such as DHCP, DNS, etc.

More precisely, the following products are available:

- Open-source: Snort[49], Bro[50] and Suricata[51].
- NIPS: McAfee (Intel), SourceFire (Cisco), HP (TippingPoint), IBM (Internet Security Systems), Cisco, Juniper Networks, Top Layer Security (Corero), Radware, Stonesoft, NitroSecurity, Check Point Software Technologies, DeepNines (Netsweeper) and Enterasys Networks (Siemens Enterprise Communication).

[48] http://en.wikipedia.org/wiki/Netflow

[49] http://www.snort.org/

[50] http://bro-ids.org/

[51] http://www.openinfosecfoundation.org/

- UTM: Fortinet, SonicWALL, Check Point Software Technologies, Cisco, Juniper, WatchGuard, Netgear, Astaro, Cyberoam, NETASQ, IBM, Trustwave, Intoto, gateProtect.
- NBA: Arbor Networks, Cisco Systems, Mazu (Riverbed), Internet Security Systems (IBM), Q1 Labs (IBM), Lancope.

The following documents are useful to identify existing products:

- NIDS *appliances:* cf. Gartner (Magic Quadrant for Network Intrusion Prevention Systems[52])
- UTM: Magic Quadrant for Unified Threat Management[53])
- Frost & Sullivan 2007 (World Intrusion Detection and Prevention Systems Markets - N22B-74)
- Reports from evaluation laboratories: ICSA[54], NSSLabs for NIPS[55] and UTM[56]

Few products are targeting the ICS environment:

- Digital Bond provides Snort preprocessors and specific rules dedicated to SCADA[57]. Other vendors have used these tools to adapt their products.
- SRI adapted EMERALD, a NIDS they have developed, to detect intrusions in SCADA[58].
- The Centre for Secure Information Technologies developed rule-based NIDS probes dedicated to SCADA[59].

HIDS are most often marketed as anti-virus. VirusTotal[60] is a web service that allows remote file verification using the most widely available anti-viruses:

- AegisLab (AegisLab)
- Agnitum (Agnitum)
- AhnLab (V3)
- Antiy Labs (Antiy-AVL)
- Aladdin (eSafe)
- ALWIL (Avast! Antivirus)
- AVG Technologies (AVG)
- Avira (AntiVir)

[52] *http://www.stonesoft.com/export/download/pdf/IPS_MQ_2010_208628.pdf*

[53] *http://www.virtua.si/fileadmin/user_upload/Fortinet_Gartner_UTM_Report_2010.pdf*

[54] *https://www.icsalabs.com/products?tid[]=4222*

[55] *http://www.nsslabs.com/research/network-security/network-ips/*

[56] *http://www.nsslabs.com/research/network-security/utm/*

[57] https://www.digitalbond.com/tools/quickdraw/
[58] http://www.csl.sri.com/papers/PST2010/pst2010.pdf
[59] http://www.csit.qub.ac.uk/News/Events/Belfast2013/Filetoupload,379555,en.pdf
[60] https://www.virustotal.com/

- Baidu (Baidu-International)
- BitDefender GmbH (BitDefender)
- Bkav Corporation (Bkav)
- ByteHero Information Security Technology Team (ByteHero)
- Cat Computer Services (Quick Heal)
- CMC InfoSec (CMC Antivirus)
- Commtouch (Command Antivirus)
- ClamAV (ClamAV)
- Comodo (Comodo)
- Doctor Web, Ltd. (DrWeb)
- Emsi Software GmbH (Emsisoft)
- Eset Software (ESET NOD32)
- Fortinet (Fortinet)
- FRISK Software (F-Prot)
- F-Secure (F-Secure)
- G DATA Software (GData)
- Hacksoft (The Hacker)
- Hauri (ViRobot)
- Ikarus Software (Ikarus)
- INCA Internet (nProtect)
- Jiangmin
- K7 Computing (K7AntiVirus, K7GW)
- Kaspersky Lab (Kaspersky)
- Kingsoft (Kingsoft)
- Malwarebytes Corporation (Malwarebytes Anti-malware)
- McAfee (VirusScan)
- Microsoft (Malware Protection)
- Nano Security (Nano Antivirus)
- Norman (Norman Antivirus)
- Panda Security (Panda Platinum)
- PC Tools (PCTools)
- Qihoo 360 (Qihoo 360)
- Rising Antivirus (Rising)
- Sophos (SAV)
- Sunbelt Software (Sunbelt antivirus)

- SUPERAntiSpyware (SUPERAntiSpyware)
- Symantec AntiVirus
- TotalDefense (TotalDefense)
- Trend Micro (TrendMicro, TrendMicro-HouseCall)
- VirusBlokAda (VBA32)

Other HIDS are available, such as Verisys[61], Tripwire that monitor modifications on the system. OSSEC[62] is quite complete open source solution that perform log monitoring, integrity checking, MS Windows registry monitoring and rootkit detection and works on most platforms.

Current challenges and objectives of the PANOPTESEC project

The main challenge of IDS in general is the famous tradeoff between True Positive Rate (sensitivity or recall) and False Negative Rate (fall-out). We expect IDS probes to detect a wide range of attacks while raising few false alerts. Thus, they must ideally exhibit a high detection rate (and so a low FNR) and a low FPR. This supposes that the model used by the IDS about malicious (for misuse-based approaches) or legitimate (for anomaly-based approaches) behaviors must be both complete and correct. In practice, this objective is hard to achieve and a tradeoff have to be found. To choose an IDS probe or to find the optimal configuration, users can rely on ROC curves [87] but have to take into account the base-rate fallacy [88].

Sensitivity and fall-out highly depends on the type of detection approach implemented in the IDS. Thus, the number of attacks covered by the signature database intrinsically limits the detection rate of misuse-based approaches. They cannot detect attacks exploiting so-called "zero-day" vulnerabilities and the signature database must be updated continuously to maintain an acceptable detection rate. However, this approach is quite sensitive. On the contrary, anomaly-based approaches can theoretically detect previously unknown attacks but suffer from poor sensitivity. Indeed, the inferred model often fails to capture all the legal variations of user behaviors. These somewhat caricatured results should be qualified, as they vary against the different types of attacks. However, they highlight the need to combine different approaches and so different IDS probes.

Most of the existing NIDS products are designed for the monitoring of traditional IT systems. One of the main challenges is to adapt these products or to design knew IDS that are well suited to the ICS environment [89]. This can be achieved by using traditional signature-based approaches, like the tools provided by Digital Bond. However, this approach supposes to specify a representative set of attack signatures. This could be quite challenging as only few attacks have been reported for such environment.

Another appealing approach consists in designing a spec-based mechanism that takes into account the specific workflow of the industrial process or the protocol behavior [90], [91].

Label	Importance	Difficulty
Modeling industrial process for spec-based IDS	3	2

2.6 Alert Abstractions

As stated earlier, an intrusion detection system is usually made of several probes deployed on the monitored information system (at either the network, system or application level). In each probe, a sensor gathers information on the activities and generates audit events. These events are analyzed by IDSes that rely on misuse or anomaly based techniques to detect malicious activities. An IDS (also called an analyzer) generates raw alerts that are transmitted to a manager (or a SIEM) in charge of

[61] http://www.ionx.co.uk/
[62] http://www.ossec.net/

dealing with these alerts. In Figure 3, the two main above-mentioned entities (namely a probe and a manager) are depicted.

Alert correlation aims to addresses the fact that the flow of raw alerts is usually huge. First, each IDS tends to generate too many alerts (and a lot of them are false alerts also called false positives); second, the multiplicity of the IDSes contributes also to increase the number of these alerts and false positives.

As a consequence, the administrator faces a large amount of data that requires to be analyzed. Therefore, to assist the administrator and avoid an information overload, additional actions have to be performed before by the manager. In particular, the manager must contribute:

1. To enrich the content of the generated alerts.
2. To aggregate the alerts in order to lower the amount of data to be analyzed.
3. To provide a high level analysis of the low level observed alerts and to produce incident reports and/or indicators related to the security of the global system.
4. To provide a succinct and compact presentation of the most relevant results.

In this document, we define the alert-level correlation process as the whole set of steps required to perform alert aggregation. In the literature and among the providers of security tools, the term *alert correlation* is often used in a very general way to refer to different objectives and used techniques. Herein, *correlation* covers all the steps of the process proposed to aggregate raw alerts and to facilitate their analysis. Alert aggregation is the targeted final objective while correlation is the name of the process used to reach this purpose. This process is itself composed of several consecutive steps, as illustrated on the right side of the Manager box in Figure 3.

Each of these steps has a precise objective. *Normalization* of alerts is a prerequisite for the detection of relationships between alerts from various sources. *Enrichment* and *verification* improve accuracy of the generated information. *Aggregation* of alerts into meta-alerts generates only high level alerts. Finally, the global analysis meta-alerts flow determines the impact of the detected attacks, their consequences for the system or the organization and deals with report generation.

These sequential steps are logically linked. Nevertheless, it is sometimes required to adopt an iterative approach and to go back to some previous steps of the process. For example, the enrichment and the verification can be applied successively to raw alerts, to enriched alerts or to meta-alerts resulting from the aggregation process.

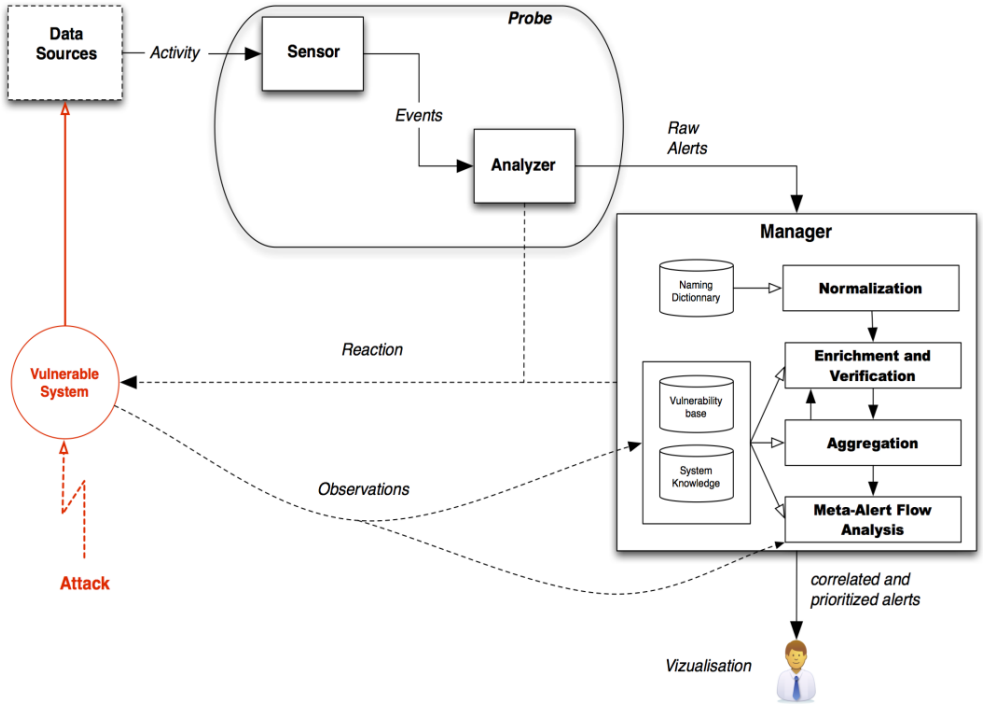


Figure 3: Online Correlation System Architecture

As shown on *Figure 3*, complementary information can be required during some steps of the correlation process. For example, *normalization* requires, among others, a dictionary of synonyms that identifies the different possible names used to refer to a same information (e.g., several ways to name similar vulnerabilities). *Enrichment* and *verification* also require to have an additional knowledge about the assets that characterize the monitored system (computers, operating systems, services, topology, open port on a firewall, etc.) and the known vulnerabilities that affect them. Of course, the content of these two databases can also be used during the next steps. In particular, this knowledge is very useful during the analysis of the flow of meta-alerts. These knowledge bases are updated by automatically or manually extracting information either from a specification or from an observation of the system.

The four steps described above are corresponding to first level functions. Their purposes can be refined as follows:

1. *Alert normalization*: the normalization of alert messages focuses on their structure (their fields) and also on their content (the value of their fields). Usually, the syntactic normalization is distinguished from the semantic normalization which is more complex. The normalization process (at least the syntactic one) is a prerequisite for the aggregation of alerts and alert correlation.
2. Alert enrichment and alert verification:
   - *Alert enrichment* consists of adding external knowledge to the alert. For example the IP address of the attacked machine if the alert is generated by a host IDS that does not provide this information.
   - *Alert verification*, consists of determining if an attack has been successful, i.e., if the generated alert corresponds to a real successful attack (intrusion identification, false positive elimination).
3. Alert aggregation: three different levels are identified.

- *Alert fusion* consists of merging information carried by alerts produced by several probes of the same type for a given attack.
- *Elementary attack identification* consists of merging simple alerts that characterize a given type of elementary attack.
- *Intrusion scenarios identification* consists of merging alerts that characterize the elementary attacks of a complex attack scenario.

4. Global Risk Analysis based on on-going Scenarios:
   - *Analysis of the impact* of the attack scenarios identified for a given system.
   - *Ranking of alerts* to prioritize them, depending on both the previous evaluation and the alert processing policy.

The five first correlation functions listed above (normalization, enrichment, verification, fusion, and elementary attack recognition) can be viewed as *low-level alert correlation*. They are actually a prerequisite for the last three functions. In the remainder of this section, we investigate each low level function and provide a more detailed description for each of them. The next function (intrusion scenarios identification) is a more high level function dedicated to correlation of previous built meta-alert that belong to the same global attack scenario. The two last correlation functions are linked to dynamic risk analysis: given an on-going attack scenario that has been identified, determining the impact on the attacked systems and prioritization of the meta-alerts to be presented to the system administrator for reaction. These three last functions, which are more related to the intended enhancements in the PANOPTESEC context, are presented later.

Note that some low level correlation functions can be implemented in the probe itself. For example, such a choice can be made for the implementation of the syntactical normalization, verification and fusion (when facing the problem called scission - see below). However, a probe usually has to satisfy hard temporal constraints. Therefore, only a limited number of tasks that are not time consuming can be implemented at this level.

2.6.1 Alert normalization

Alert normalization consists of transforming each alert so that it conforms to a standard format that can be used by any other component of the correlation system. This operation is required when several probes emit alerts in several formats.

The need for normalization has been clearly identified in previous works such as [141], [125] or [119]. Each of these works considers the normalization as a prerequisite to any correlation activity.

Normalization can take two different aspects. *Syntactical normalization* consists of renaming semantically equivalent fields from a source format to the destination format. For example, a field that contains the IP address of a packet can have a different name in the source format and the destination format. In the IDMEF format that will be described later, the field that contains this information is called *Alert.Source.Node*. *Semantic normalization* consists of translating information contained in the source format alert into an information that has a well identified meaning in the destination format. For example, if an alert reports the exploitation of a vulnerability using its Bugtrack reference, we can replace this reference by its equivalent CVE reference. Actually, the definition of the new alert relies on additional knowledges that are not carried by the original alert itself. For example, this translation process can rely on naming dictionaries, as those depicted in *Figure 3*.

In [121], the authors identify the main tasks to perform during the normalization process. They actually define four steps to normalize alerts: (1) conversion of the initial alert into a string representation, (2) creation of a new alert in accordance with the selected target format (here, IDMEF), (3) analysis of the content of each field of the new alert and finally (4) semantic normalization of the fields of this alert, using a dictionary.

Very few formats are actually standards. As a consequence, most tools use a proprietary format. However, the IDMEF format (Intrusion Detection Message Exchange Format), summarized on *Figure 2*, was defined by the IETF (RFC 4765, March 2007) and is recognized as a potential solution that is already widely used in the academic community. The description of the fields in this format is out of the scope of this document. For a complete description, the reader is invited to refer to the RFC.

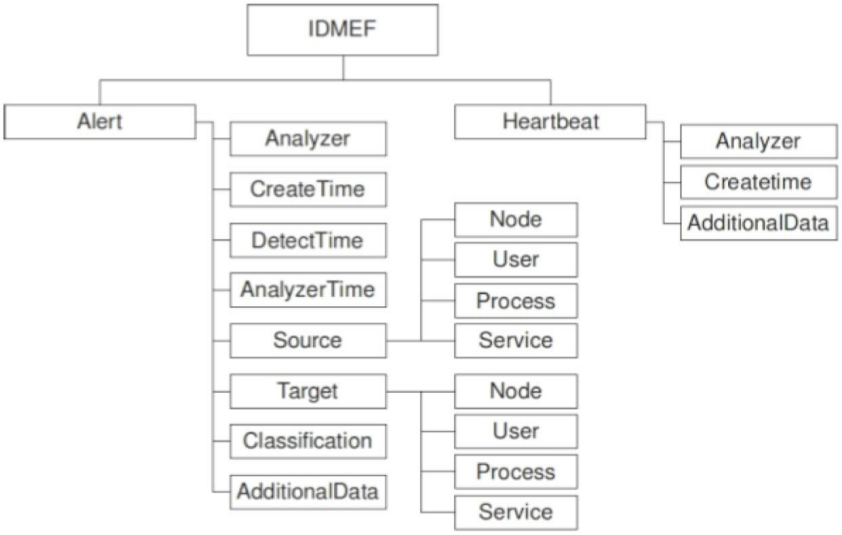


Figure 4: IDMEF Format

The IDMEF format aims at standardizing alerts, not events. That is why the MITRE proposes an event format[63] called CEE (Common Event Expression) that aims at describing logs (see *Figure 3*). Even if this format is originally devoted to the description of events, it is generic enough to cope as well with the description of alerts. Again, this document does not pretend to describe all the particularities of this format: for this, the reader can consult specific documentation about MITRE.

[63] http://cee.mitre.org/docs/overview.html

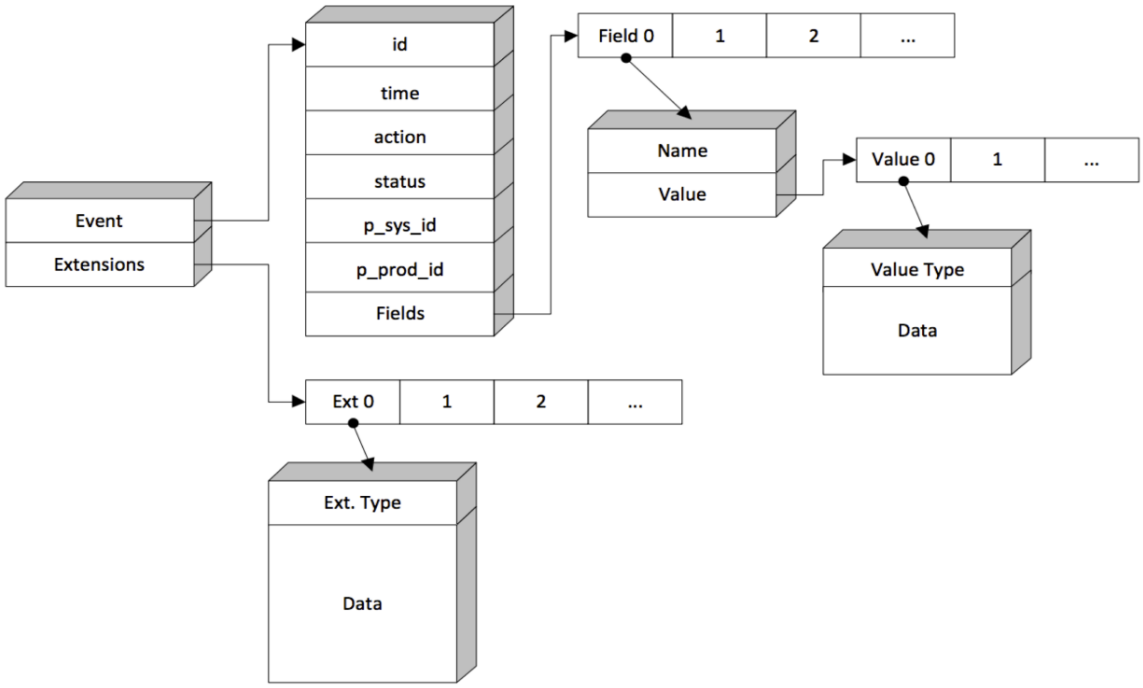


Figure 3: MITRE CEE Format

Finally, various proprietary formats such as CBE (IBM), CEF (Arcsight), OLF (eIQnetworks) have been defined. By definition, each of these formats is used only within the tools proposed by their designers. Yet, this limitation is perhaps less effective in the particular case of the Arcsight CEF format which is recognized by other tools, such as Prelude Pro.

2.6.2 Alert enrichment

Alert enrichment consists of adding information to an alert. This information is not present in the original alert and must be provided by external sources. In the following, we identify some possible types of additional information and we explain why this information is useful for the correlation process.

Generally speaking, the complementary information can be related to:

1. The machines, their configuration, installed software, the network topology: these information items are collected in the system configuration.
2. The prevention and detection tools deployed on the supervised system: this information is contained in the system configuration.
3. The known vulnerabilities, and especially the ones that are present in the supervised system: this information is contained in the vulnerability database.

Very few logical models describe how to organize these data items. In [128], a logical model describes how relevant information on the system can be obtained. For example, when an alert emitted by a host-IDS does not provide the IP address of the involved machine, this particular information has to be extracted from the system configuration and has to be added to the alert.

In [144], the authors propose to enrich in a passive way each alert with the context of its production. This context includes various information: the type and the version of the Operating System, its expected vulnerabilities, the services available on the machine, their vulnerabilities, the open ports, the simplified network topology, information on the network flows, on the users and processes, and on the possible modifications on the file system. This work, even if it is not really implemented, precisely

identifies several types of information that may be of interest during the alert verification step. The next section focuses on this step.

2.6.3 Alert Verification

Alert verification consists of verifying whether an alert represent a successful attack. This requires verification whether (1) the system is vulnerable to the reported attack and (2) traces of the attack can be observed on the system. To perform these two controls, an active or a passive strategy can be adopted [122].

Here, the words "passive" and "active" have a meaning that is different from those they have in the previous section dedicated to data enrichment. The passive verification consists of using data previously collected and stored in the system configuration and in the vulnerability database (see below). By contrast, the active verification consists of:

1. Dynamically obtaining information on the configuration of the system. In practice a similar information is already available in the system configuration. However, as the new information is obtained dynamically, its freshness is sure.
2. Verifying if the consequences of the attack are observable on the system. These consequences can be, for example, the modification of a service behavior, the existence or absence of a particular information in the file system, the modification of the state of a communication port, the existence or absence of a particular line in a log, etc.

Table 1 offers a summary of the four possible verification cases. It shows that the verification of the attack traces cannot be performed in a passive way. It also proposes examples of implementations that we describe in the following.

	Passive approach	Active approach
Vulnerable configuration	[128]	[122] [123]
Attack traces on the system	-	[122] [123] [127]

Table 1: Alert Verification and references

The passive verification requires to have, in addition to the system configuration, a vulnerability database containing all currently known vulnerabilities (whether they are actually present or not in the system). This vulnerability database can be imported in the system automatically using bases such as bugtrack[64] or CVE[65] that are publicly available.

Logical links must be defined between the system configuration and the vulnerability database, in the way similar to what is done in M4D4 [128]. These links allow passive verification of whether or not the system is vulnerable. For example, if the system configuration describes the deployment of software of a given version and the vulnerability database defines that this version of this software has a vulnerability, then we can deduce that an alert reporting an exploit of this vulnerability has a lot of chance to be a true positive. Actually, this model (which implementation is made in Prolog) allows forging more complex requests than this very simple example.

To illustrate a particular form of active verification, we describe here the work of [122] [123]. This work relies on a dynamic verification of the existence of vulnerabilities. Instead of relying on a vulnerability database defined prior to the execution (passive verification), they use a tool to scan vulnerabilities (e.g., Nessus). To be able to verify a given alert, this alert must provide a CVE

[64] http://www.bugtrack.net/

[65] http://cve.mitre.org/

reference. The set of NASL scripts in Nessus is searched for this reference. If no script is found, the alert is enriched with information defining that it is probably a false negative. If the script is executed successfully, the alert is considered to be verified and is a true positive.

A second type of active verification is proposed in the same article. It consists of verifying on the target the presence of attack traces. The description of the action that must be performed to check the traces must be added to the alert during the enrichment phase. An agent deployed on each possible node is dedicated to the collection of the information that has to be verified on this node. The communication between this agent and the correlation engine must be secure to keep a high level of authenticity, confidentiality and integrity. In practice, it relies on SSL communications.

In the previous work, the nature of the control action performed to verify the alert is previously known (a test of the presence of a vulnerability or a check of a trace on the system). The reference [127] proposes another approach. Instead of specifying manually what must be verified, the authors try to automatically generate verification rules. These rules aim to take into account additional information that characterized the state of the attacked machine. For example, answers to the following questions are gathered: has a port state been modified? Did the machine powered on or off? Is any relevant information contained in logs? Using the learning of network traffic, verification rules are generated for each Snort alert raised for this network traffic. This article does not explain the details of the algorithm used to generate these rules, but it illustrates the relevance of such an approach on a small set of generated rules.

2.6.4 Alert Aggregation

In [141], the authors define the alert fusion activity. It consists of creating a meta-alert that merges all information data carried by alerts produced by several probes of the same type for a given attack.

Actually, the production of multiple alerts for a given attack can have several causes:

1. It can be the consequence of IDS redundancy on the system: the same IDS is deployed at several places in the system. This redundancy implies that many independent detections of the same attack occur in the system.
2. It can be the result of an incorrect behavior of an IDS whose detection algorithm can be inaccurate or whose configuration can be incorrect. This IDS can however detect that an event includes many suspect characteristics and thus produces multiple alerts. This phenomenon is called *splitting*.
3. Finally, an IDS can produce multiple alerts when a periodic abnormal phenomenon occurs (this phenomenon is not necessary due to an attack). This phenomenon is called *recurrence*.

These three phenomena are illustrated in *Figure 6*.

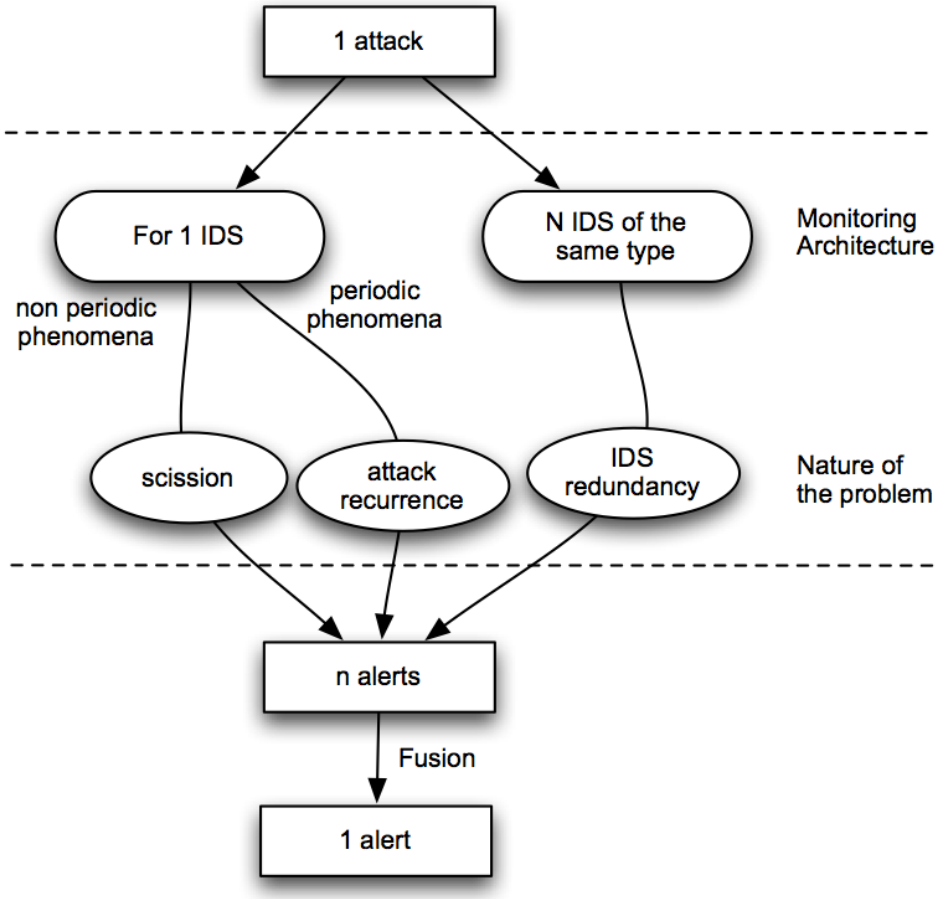


Figure 6 : Alert Fusion is required

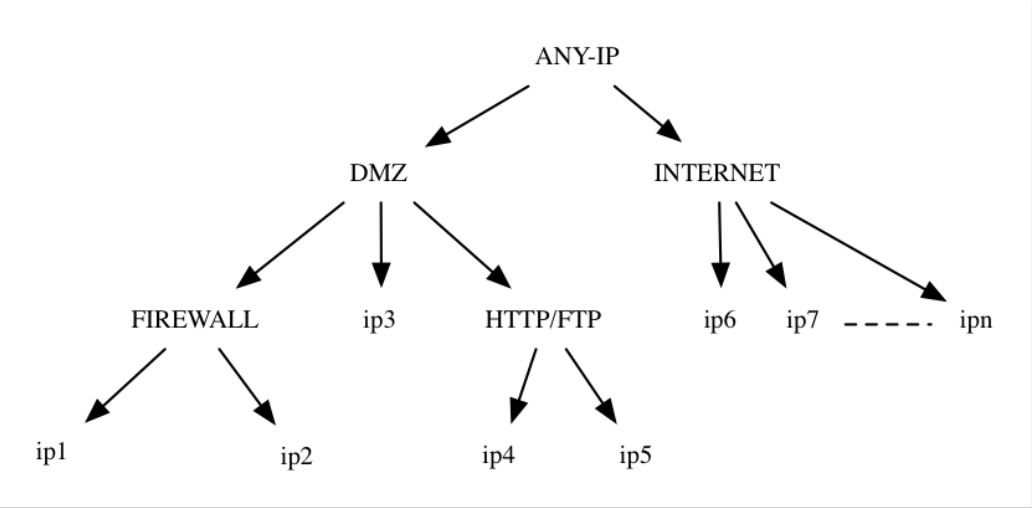
Fusion of alerts has to merge similar alerts provoked by these several phenomena.

As far as we know, a very simple approach is generally used to solve the redundancy problem: it consists of merging all identical alerts that occur in a given time frame. We must note here that the redundant IDSes are of the same type. As a consequence, the alerts they emit for a given attack are at least comparable and at best identical after the normalization phase.

The problem of *splitting* should be treated at the probe level. Indeed, the IDS should have all required information to limit its output to one alert for each analyzed event. However, we must define precisely the content of the emitted alert. This can be difficult, as the alert must report the multiple suspect characteristics of the analyzed event. We did not find any article in the literature relative to this point.

In the literature, the articles focus essentially on the merging of recurring alerts. In the following, we focus on this problem.

Two types of approaches are used. The first approach requires discovery of rules that link the alerts by exploiting an alert base. The second approach consists of defining rules that specify what properties the merged alerts must share. Moreover, some articles describe approaches that try to avoid the use of explicit merging rules: we call them *implicit* approaches.



Example of hierarchy generalization

We can expect that similar alerts have the same cause (i.e., they are the result of the same attack). According to this principle, [120] focuses on the creation of similar alert clusters. The similarity between alerts is defined using a distance: the distance between two alerts is the (weighted) sum of the distances between the several attributes of these alerts. The distance between two attributes relies on the concept of a "generalization hierarchy" that can often be illustrated using a tree, but is more generally defined using an acyclic oriented graph (cf. *Figure 5*). The leaves of the tree represent the attributes or the set of attributes of the primary alerts (on the figure, the IP addresses we can find in the packets). The parent nodes are in fact generalized attributes regarding potentially many axes (on the figure, the type of service provided by the machine that owns this address or the sub-network that owns this address). If we have such a generalization hierarchy for each attribute, a generalized alert will be an alert whose attributes are generalized. We can then define the distance between two attributes as the sum of the shorter paths between a common parent or the attributes themselves. As a consequence, we can define the distance between two alerts (the sum of the distances between their different attributes) and the mean distance between a generalized alert and a cluster of primary alerts (the normalized sum of the distances between primary alerts and generalized alerts). The heterogeneity of an alert cluster is then defined as the shortest mean distance between this cluster and the set of possible generalized alerts. A generalized alert whose average distance to the cluster corresponds to the cluster heterogeneity is called "covers". The problem consists thus in discovering alert clusters whose cardinal is sufficiently big and whose heterogeneity is minimal (for a given set of generalization hierarchy). The meta-alert sent will correspond to a cluster cover.

In [126], the authors propose to learn alert association rules inside an alert database. The discovery of association rules is a well-known problem in the data mining community. In particular, it is heavily used in the field of commercial transaction analysis to identify the frequent buying combinations. Due to the fact that frequent alerts, distributed on a long period, are generally the consequence of normal phenomena (i.e., non-intrusive), Manganaris proposes the modeling of the normal behavior of a system as association rules issued from an alert base. In their approach, a transaction is made of alert frequency peaks, i.e., alert groups whose inter alert delay is weak. A transaction is the basic unit of the analysis. The analysis of the transactions reveals association rules between the alerts in a given group, and permits establishment of the profile of normal behavior of an analyzer. Indeed, the observed deviation between the alerts generated by the analyzer and its profile allow isolation of the intrusive activities. In this approach, fusion of alerts is thus not based on a similarity measure on the value of the attributes but on frequency relationships. This way, it handles the recurring alerts.

In [126], the authors focus also on the fusion of recurring alerts. Thanks to data mining techniques, the authors propose to identify alert attributes that appear frequently in a correlated manner. They are thus

able, using a learning phase, to build a model that allows detection of new occurrences of attacks similar to attacks that have previously occurred. Like in the previous approach, this method allows discovery of the fusion model during a learning phase.

The three articles previously presented require fusion rules explicitly defined by the administrator or obtained during a learning phase. In the following, we focus on implicit approaches.

In [142], the authors propose an approach of alert fusion relying on the principle that frequent alerts, distributed on a long period, are generally produced by normal periodic phenomena (i.e., non intrusive). However, contrary to previous work, the order of alerts inside each cluster is taken into account. An "alert signal" is built by counting the number of alerts of a given type (a different type for each Snort signature, in the experimentations) in a given time frame (5 minutes and one hour are proposed by the authors). The alert signal is used instead of the individual alerts. In practice, the authors note that for some types of alerts, each alert carries very low information, by contrast with the alert signal that reflects the suite of alerts. In particular, they note that this signal contains many periodic phenomena that are considered normal. As a consequence, it is required to filter these phenomena in order to identify disruptions in signal. The hypothesis is that these disruptions are the consequences of intrusions. A non-stationary auto-regressive model is used for modeling the regularities that can thus be filtered. The remaining irregularities must also be analyzed by the administrator.

In [143], the authors propose a relatively simple approach, where a notion of distance allows merging similar alerts. The distance between two alerts is the weighted sum of the distances between the alert attributes. These distances are computed using a similarity function. In the contrary, the interesting concept of "expectation of similarity" is introduced by the authors. It allows weighting of the attributes depending on the type of attack. For example, for a SYN FLOOD attack, it is not necessary to focus on the source address similarities, as they are most of the time masqueraded by the attacker. The authors propose also a "minimum of similarity", defined for each attribute, that represents the maximal distance for which the alert must be merged (if a single attribute exceeds its minimum of similarity, the global similarity of the alert is considered to be null). The authors expect to use this technique at different levels by relying on different weights or similarities. At the first level, they aggregate alerts emitted by a given probe (solution for the split problem). At the second level, they aggregate alerts emitted by different probes (the weight of the attribute defining the probe is thus lowered). We focus on this type of approach in the next section (remember that the fusion process deals with alerts emitted by the IDS of the same types).

Challenges for detection in ICS/SCADA systems

Current challenges and objectives of the PANOPTESEC project

Label	Importance	Difficulty
Detecting attacks vs. detecting intrusions	3	3
Misuse-based vs. anomaly-based	3	3
Spec-based and policy-based IDS	3	3
ip/domain black list and reputation systems	3	3
False positive vs. false negative, base-rate fallacy	3	3
Language of signatures, automatic generation of signatures, vulnerability-based signatures	3	3
Incremental learning, feature selection, unbalanced classes.	3	3
Challenges for detection in ICS/SCADA systems	3	3

2.7 Ontologies and Semantic Science

The main contribution of Semantic Science in the area of data collection and correlation lies in the use of a mathematical framework and mixed-method for problem solving driven by Semantic Web technologies.

Correlation engines usually rely on rule-based triggering procedures. The complexity of managing many (possibly conflicting) rules in a knowledge base makes the system hard to update. PANOPTESEC proposes a component for the correlation engine based on "Description Logics". This will allow system designers and operators to easily interact with the engine and keep the knowledge base dynamically updated, while still ensuring optimal computational time.

2.7.1 Data collection

R. Sawilla, et al. [98] present the concepts behind the proposed Automated Computer Network Defence (ARMOUR) Technology Demonstration Project (TDP). Their work identifies the need for data correlation and pre-processing; however, their approach is focused on the risk analysis and response aspects without addressing current challenges in data collection and pre-processing.

Splunk[66] performs data collection oriented to indexing and storage. It provides further aided navigation and searching through data (sensors, networks, databases, virtual machines, servers, security devices, etc.). One missing feature is that Splunk does not support the abstraction of data. Because of the way it organizes data, it is performant for indexing but not capable of translate data into concepts. This is a technological limitation.

PANOPTESEC data collection aims at capturing data in real time, both from external sources (like sensors) and internal (like information coming from attack graphs, users commands input in visualization system). The system must be able to handle different data formats, exchanging information with all PANOPTESEC components build or used in every Work Package

PANOPTESEC data collection system does not need to immediately store the data but to prepare them to pass to an abstraction phase.

2.7.2 Abstraction

L. Stojanovic, et al. [101] discuss the advantages of using ontologies to represent knowledge in a correlation engine. Stojanovic approach has strong limitations because of using IF-THEN rules for implementing correlation operations.

W. Li et al. [99] discuss the use of ontologies to represent knowledge about attacks in a multi-agent architecture. The approach lacks however a clear and easily modifiable integration with mission impact models.

Recently the Rule Interchange Format (RIF) has been created to define a standard way to exchange rules among rule systems, RIF is oriented on exchanging instead of defining rules in order to cover more than one rule modelling. The limitation of this way of abstracting is that it does not allow expression of logic relations among concepts. This makes correlation more difficult and limited.

PANOPTESEC Abstraction System will take data from the Collection Component and will transform them into elements belonging to a Knowledge Base. The approach will not use an IF-THEN-ELSE formalism but an ontology in which the relations between the concepts are expressed with a strong semantic language.

Thanks to its logic-driven language, this representation will easily support the correlation phase.

[66] http://www.splunk.com

2.7.3 Correlation

N. Cuppens-Boulahia, et al. [100] describe the use of ontologies in the context of a complete attack response system, where knowledge from the ontology is integrated in system policies for attack management. However, their approach relies on rules as well.

Existing correlation engines for security applications (McAfee Advanced Correlation Engine, Skyline Communications DMS Correlation Engine, Quantum Secure SAFE Event Correlation Engine) cannot easily incorporate new or dynamic information, due to the fact that they're designed to establish correlations between events by executing and managing business rules and business processes and workflows. This "operational" approach is not well suited to implement a more "declarative" approach, where knowledge about the system and the attacks can be added to the engine without specifying all the related workflows and rules.

The PANOPTESEC method of correlation is not only based in statistic approaches but also on formal reasoning and logics. This permits performance of more general correlations reasoning so as to allow the system to automatically link risks and protective decisions and thus to evolve better while responding to cyber-attacks.

3 Dynamic Risk management

In order to achieve a comprehensive yet accurate mitigation of possible and ongoing attacks on the managed ICT system, the PANOPTESEC Security Management System will rely on a Response System that continuously quantify risks and decide how to respond smartly to cyber-threats that target on the monitored ICT system.

Unlike early work that only considered the impact factor, the PANOPTESEC Response System aims to minimize the risks of both possible and ongoing attacks, noting that a risk is evaluated as a function of the likelihood and the impact of a threat/an attack. We then intend to adopt a "quantitative" risk-aware approach that provides a comprehensive view of the threats, by considering, (i) the likelihood of success of the considered attacks, (ii) their induced impact, and (iii) the cost and impact of the possible responses. Moreover, the proposed Response System aims at dynamically managing the security of a monitored ICT system proactively and reactively.

The Response System for the Dynamic Risk Management that is designed may then be schematically decomposed in two complementary chains of treatment: one for the Proactive Risk Management, which state of the art and deficiencies of existing solutions are detailed in the *Proactive Risk Management* section, and another for the Reactive Risk Management, which state of the art and deficiencies of existing solutions are presented in the *Reactive Risk Management* section. This decomposition may be further refined for each chain of treatment in three stages for, Situation Awareness, Risk Assessment, and Response Assistance. A simplified diagram of the functional architecture of a DRM Response System is presented in Figure 10.

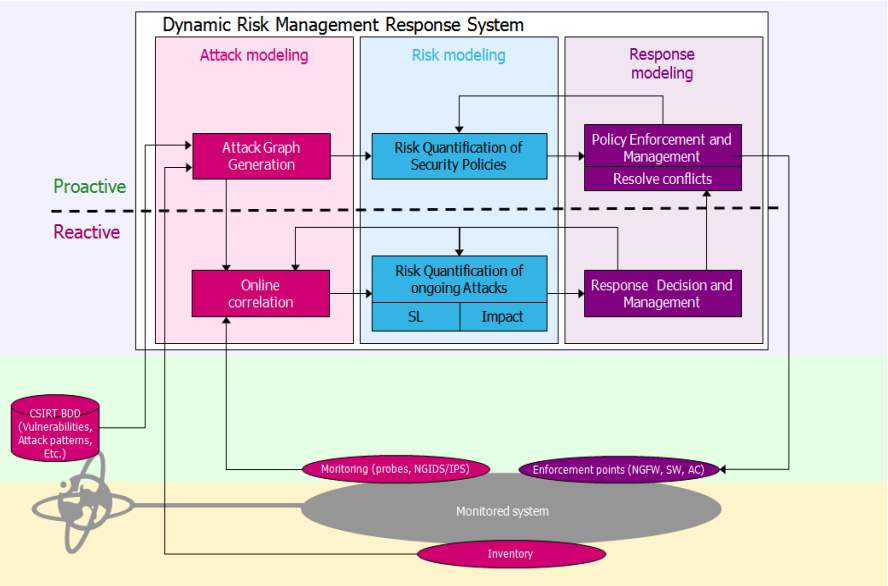


Figure 10

3.1 Proactive Risk Management

The *proactive risk management* chain of treatment consists of managing dynamically and permanently the risk during the life-cycle of a system. On a daily basis, new vulnerabilities are discovered, with a continuous growth in the complexity and the dynamicity of today's ICT systems. Our intent is then to automate this proactive risk management process, which enables the security officers to permanently update their risk evaluation process and deploy the needed security measures.

For the Attack Awareness stage, a commonly admitted approach in security of ICT systems is to rely on attack graphs as means to express attack scenarios. In this context, generation methods must be adapted to automatically compute topological attack graphs based on the knowledge of the collected topology and the vulnerabilities inventory on the monitored ICT system. The Risk Assessment stage, on its side, may rely on models and algorithms to assess the Success likelihood based on attack graphs. The impact of potential attacks must also be considered, by modeling the service dependencies in the monitored system. Eventually, when it comes to the Response Assistance, strategic security measures (e.g. patching, filtering, NAC, etc), formally modeled as security policies (e.g. RBAC or OrBAC as presented *earlier*, should be updated and activated with respect to the decision of the Security Officer to reduce the evaluated risks.

In addition to the attack graph tool, there is a need for a component to analyze inputs from the live system (configurations, topology, assets), matching them with an overall security policy and proposing preventive actions, even in the absence of known attacks.

3.1.1 State of the art and limitations

State-of-the-art efforts in attack and risk modeling have made advances in these areas, however they lack an integrated approach combining attack and risk modeling, nor do they include the aspects of response assistance.

- S. Jajodia, et al. propose a means to assess risk (likelihood dimension) based on a Bayesian approach. However, they do not consider response aspects, nor do they distinguish between the proactive and reactive aspects.

- R. Lippmann, et al. [112] also adopt a Bayesian approach and propose to generate attack graphs (NetSPA), to assess risk in a proactive aspect. However, they do not consider the integration of response aspects.
- X. Ou, et al. [64] propose attack graph generation methods called "Multi-host, Multi-stage Vulnerability Analysis Language" (MulVAL). Their approach for risk assessment is based on a Bayesian approach for the proactive assessment of the likelihood based on attack graphs. Their model is not generally applied to reactive assessment problems and they do not consider response aspects.

Some evidence of the opportunity for a fully integrated approach involving attack, risk and response modeling has surfaced in recent years. However a scalable operational system is not yet available. R. Sawilla, et al. [98] present the concepts behind the proposed Automated Computer Network Defense (ARMOUR) Technology Demonstration Project (TDP). Their work demonstrates that a fully-integrated and automated cyber-defense system is necessary, possible and not yet available. They propose a similar attack modelling, risk analysis and response cycle to the PANOPTES proposal based on a planned open source integration framework. However, their approach relies on some techniques noted above (i.e., MulVal), does not address aspects of threat likelihood in the risk quantification model, and has yet to be developed into an operational system.

3.1.2 Products, Processes, Services and their deficiencies

Leveraging Attack Awareness & Risk Assessment:

- Vulnerability scanners (e.g. Nessus, etc.)
- Attack graphs generators (e.g. Skybox, Redseal, etc.)
- Traditional Risk Assessment methods and processes

Leveraging Response Assistance:

- Risk treatment methods and processes
- Patch management tools (e.g. simple Patch Managers, enhanced PM like Skybox, etc.)
- Firewalling policy management and NAC management tools (e.g. fwbuilder, Skybox, Sourcefire; etc.)

Objectives of the PANOPTESEC project:

- Requirements in terms of data sources: system configuration, assets, mission impact model;
- Requirements regarding the policy language and engine (contextualization, description and evaluation of assets, possible recommended actions...)

We propose to address and handle threats based on attack surface-like methodologies [A,B]. Such methodologies allow evaluation of the security of complex dynamic systems and minimization of their vulnerability to attack. The objective is to isolate vulnerability exploitation and reduce potential damage to organizations while guaranteeing the most appropriate selection of countermeasures.

The proposed methodology should allow conduct of measurements in the absence of resources like source code, and include aspects such as monetary impact during the evaluation of potential damages. The results of the evaluation shall guide in the decision of selecting from a list of potential response actions. Key elements of the evaluation are the definition of the system (e.g., tangible and intangible assets vulnerable to known threats), the specification of vulnerabilities (e.g., specified w.r.t. the volume of assets they affect) and the description of potential countermeasures (e.g., scored by the percentage of vulnerabilities they can handle). Evaluation of results consists of the identification of the most appropriate response w.r.t. the aforementioned composition of functions (i.e., system, vulnerability, and countermeasure functions).

The proposed approach should coexist and complement other proactive models, such as service dependency based models (guided by likelihood rather than impact). The consolidation of both models into a single holistic approach shall allow tracking of vulnerability propagation across the elements of the system, while maximizing the coverage of the response both in terms of likelihood and impact.

3.2 Reactive Risk Management

The *reactive risk management* system acts complementary to the *proactive risk management* one. Indeed, some of the risks might be accepted without being eliminated. However, during operations, some of the detected attacks may occasionally see their risk rise to an unacceptable level, e.g. when the likelihood of an attack increases drastically towards its success for instance the attacker used a zero day. It may also happen that new opportunities appear for some attackers which open to them new means to threaten critical assets of the monitored ICT system (e.g. if the filtering rules of a compromised firewall are modified by an attacker). This kind of situation stresses the need for a reactive part for any organization willing to permanently and dynamically manage their risks.

By contrast with the proactive risk management system, the reactive risk management system refers to the risk management after the detection of an ongoing attack. Consequently, an insightful decision to deploy an efficient "tactical" response has to be determined in order to eliminate, or mitigate, the ongoing attack. Then, possible conflicts must be managed to produce a *security policy instantiation* that complies with both strategic and tactical response requirements before enforcement.

On the *attack awareness stage*, correlation techniques are a widely admitted means to give meaning to elementary intrusion events of various kinds (e.g. alerts from IDS/IPS based on signatures or behaviors, firewall logs, antivirus alerts, etc.), and follow the progress of ongoing attack scenarios among an important volume of collected data. As for the *proactive response system*, *attack graphs* may be a convenient means to support this high-level correlation and represent in (near) real-time the progression of simultaneous intrusions in the monitored ICT system.

Regarding *risk assessment*, accurate *impact and success likelihood* assessment must be derived from models of the ongoing attacks, the state of the monitored ICT system and an *a priori* knowledge about the value of identified critical assets threatened by the attacks.

For the *response assistance*, *tactical response* measures opportunities that may act on attacks progress or induced impacts must be determined. Then, their impacts (i.e. on the attacks, but also on the security policy) and cumulative cost (i.e. deployment cost, and cost induced by services' QoS decrease) has to be assessed in (near) real-time to prioritize the various responses measures possibilities regarding their return on investment, and, determine the best moment to activate (or deactivate) them. Note that possible conflicts between the "tactical" and "strategic" responses have to be managed.

Based on this assessment, a security officer may take the decision to deploy or automate the response measures representing the best trade-off between risk and benefit.

In this section, we present the reactive response system. We first show how low level alerts generated by the intrusion detection systems are correlated to generate meaningful alerts. We then present the way complex events are processed.

3.2.1 Attack Awareness

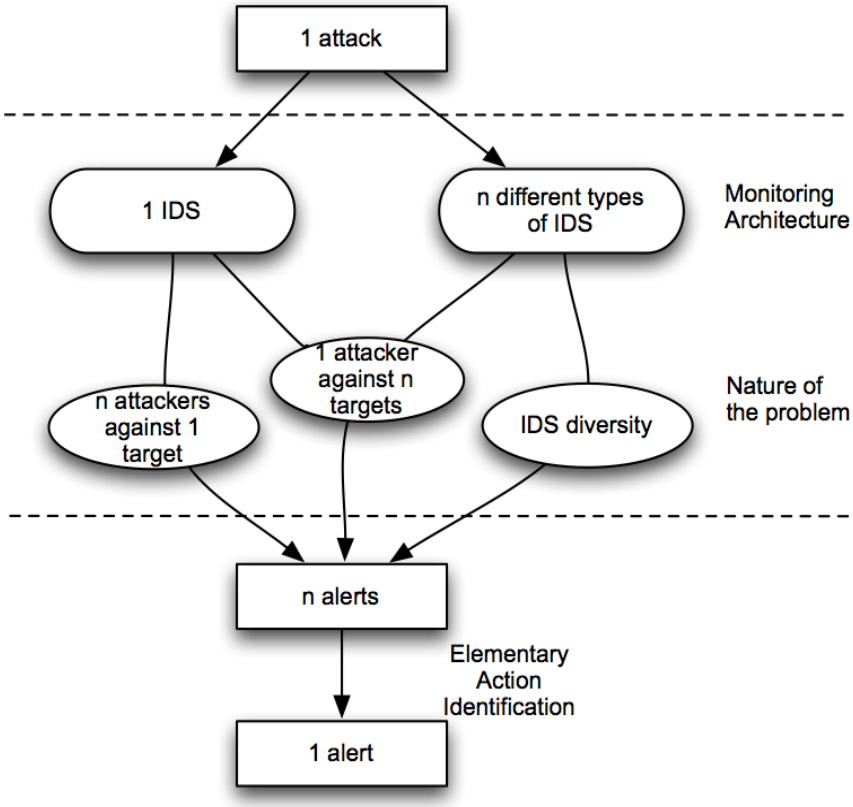
In order to react efficiently, a first step consists of improving the information about the attacks. While we presented previously alert-only information augmentation, we present in this section methods take advantage of information other that the alerts to augment their meaning.

3.2.1.1 Elementary Attack Identification

While the fusion process we presented earlier deals with alerts issued from IDSes of the same type (often Network IDS), the elementary attack identification consists of merging alerts issued from

heterogeneous sources (diversified IDS, Host based IDS, Network IDS (see *Figure 8*). This phase is identified as "session recognition" in [141].

When the IDSes are diversified, a given attack can generate several alerts. These alerts are emitted by IDSes of different types. However, these alerts are raised in a very short time frame. As a consequence this can be a criteria to merge them (if the alerts occur in a very short time frame, they are merged). In order to enhance the merge precision, we can rely on the alert enrichment process result to establish additional logical links between the alert attributes. For example, if an attack is conducted against a precise service, a network alert will be correlated with a system alert only if (1) the alerts occur in a short time frame and (2) the system alert indicates that the concerned process is listening on the port dedicated to this service.



Elementary attacks recognition

As illustrated on *Figure 8*, N alerts can also be produced by a same IDS when an attack is launched from N sources or, in contrary, when it targets N machines in the supervised network ("N to 1" attacks, "1 to N" attacks or "N to N" attacks). In this case, these alerts must be merged. This type of merge corresponds to the notion of "focus recognition" defined in [141].

In [134], the first step consists of identifying simple patterns: 1 attacker targets N machines, for example by scanning the network; N attackers target the same machine simultaneously, e.g., during a Distributed Deny of Service (DDOS) and N attackers target N machines. The second step consists of applying data mining techniques to show up the most frequent patterns and to signal them to the operator. This approach is similar to a scenario identification that we treat hereafter.

3.2.1.2 Intrusion Scenario Identification (multi-step correlation)

Performing an attack often requires multiple steps (for example, a port scan, then an exploit against a service discovered, followed by a malware upload, etc.). The goal of the identification of a scenario is to aggregate the alerts or meta-alerts produced during each of these steps. The meta-alerts result from the correlation functions previously described, as illustrated on *Figure 7*.

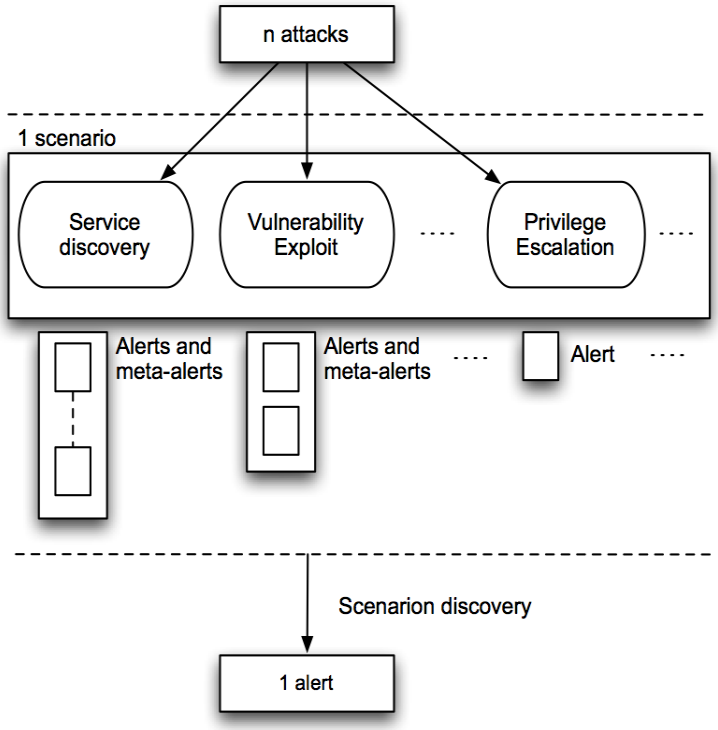


Figure 7: Alert Flows and Scenario identification

In the sequence of steps leading to the intrusion, each step is dependent on the success of the previous phase. The alerts are thus causally dependent. The methods used to identify the intrusion scenarios must then be able to rebuild these dependencies.

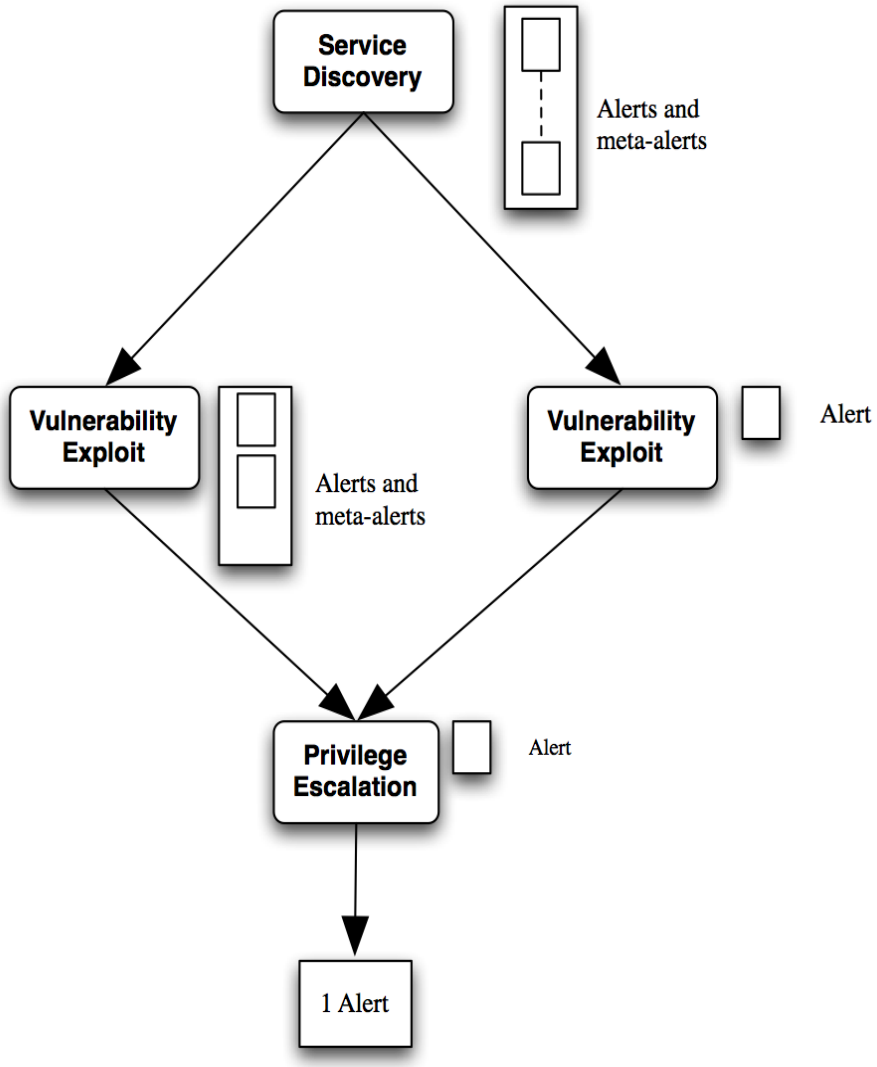


Figure 8: Explicit Scenario for intrusion recognition

In the literature, we distinguish three types of approaches to detect intrusion scenarios:

- The explicit approaches consist of specifying the expected attack scenarios.
- The semi-explicit approaches describe the links between dependent states (expressing pre and post condition on the system state that is modified by the attack steps).
- The implicit approaches rely on metrics instead of the specification of rules. We can find such approaches in [143] and [131].

3.2.1.2.1 Explicit correlation approaches

Explicit correlation approaches intend to explicitly define scenarios stored in a base of rules and then use this database to recognize known attack scenarios at runtime. A scenario example is given on *Figure 8*. In this example, we present a finite state automaton that specifies that a possible scenario is identified if a sequence of alerts and meta-alerts is emitted. This sequence exhibits a scan of services, an exploit of at least one vulnerability, and then a privilege escalation.

The main difficulty of these approaches is the definition of rules and their exhaustiveness. In order to describe a rule, a language is required. In [139], the authors describe a language called *ADeLe* (Attack Description Language) whose goal is to describe scenarios. The language provides a way to describe (1) temporal properties on the alerts (which allows to order them), and (2) to specify links between these alerts (in particular links between the value of their attributes). The time windows dedicated to the recognition of a scenario permits to cut explicitly partial recognition. This is useful to drop the recognition of scenario that will never complete. Each rule is compiled into a finite state automaton by the correlation tool called *GnG* whose role is to detect scenarios at runtime.

In [118], the authors describe another general language to describe scenarios which is a declarative high level language relying on first order temporal logic. In addition to explicit cuts (as in *GnG*), the tool called *Orchids* is able to cut implicitly partial scenario recognitions. As in *GnG*, scenarios are transformed in finite state automatons to perform detection at runtime.

STATL [117] is a third example of language dedicated to scenario description. In practice, this language permits expression of explicitly finite state automatons that are used at runtime to detect the attack scenarios.

We can see that online correlation is heavily based on the use of finite state automatons that permit the identification of scenarios in the flow of alerts that are generated in the system.

In [66], Morin and Debar propose a misuse based correlation component. The definition of misuse correlation is similar to misuse intrusion detection: known malicious or undesired sequences of alerts are searched in the stream of alerts received by the correlator. In this approach, the alarms are checked against a set of multi-events patterns (or signatures) expressed using the "chronicle" formalism proposed by Dousson [65]. A proof-of-concept correlator is built upon Dousson's chronicle recognition system (CRS).

Recognition of chronicles is based on a formalism in which time is fundamental. This contrasts with classical expert systems, which base their reasoning on rules, relegating time information to the background. A chronicle is thus a set of events linked together by time constraints.

The available time information allows ordering and the specification of time spans between two occurrences of events. The predicates used in chronicles are *hold*, *event*, *noevent* and *occurs*. The *hold* predicate represents persistence of the value of a domain attribute over an interval, without knowing when this value was reached. The *event* predicate expresses a time stamped instance of a pattern. An *event* has no duration. *Events* denote a change of the value of a domain attribute. The *nonevent* predicate expresses forbidden events, i.e. events whose occurrence leads to the invalidation of a chronicle instance during the recognition process. The *occurs* predicate is a counting predicate.

Based on these predicates, Morin and Debar propose a few examples of chronicles dedicated to alert correlation. For instance, the following chronicle allows detecting the Nimda worm:

```
chronicle nimda[?source, ?target] {
   occurs((1,2),alarm[iis_code_red_ii_root_exe,?source,?target], (t,t+2000))
   occurs((1,4),alarm[iis_decode_bug,?source,?target],(t,t+2000))
   occurs((1,14),alarm[iis_cmd_exe,?source,?target],(t, t+2000))
   occurs((1,3),alarm[web_dot_dot,?source,?target],(t,t+2000))
   occurs((1,2),alarm[iis_unicode,?source,?target],(t,t+2000))
   occurs((1,1),alarm[iis_unicode2,?source,?target],(t,t+2000))
   occurs((1,1),alarm[iis_unicode3,?source,?target],(t,t+2000))
   occurs((1,1),alarm[iis_decode_bug3,?source,?target],(t,t+2000))
   occurs((1,1),alarm[iis_decode_bug2,?source,?target],(t,t+2000))
   occurs((1,1),alarm[iis_decode_bug4,?source,?target],(t,t+2000))
   when recognized {
          emit event(alarm[nimda, ?source, ?target], t);
   }
```

}

A Nimda attempt is characterized by a burst of 10 distinct alarms. Each alarm can occur several times (for instance, the **`iis_cmd_exe`** occurs 1 to 14 times at each attempt). All the alarms should have the same source and the same target, and they all occur within a 2s time window (the resolution here is 1 ms). When such an alarm burst occurs, a synthetic alarm Nimda_worm_attempt is reported. Only one synthetic alarm represents the 30 original ones.

Beside the examples given in the paper, there is unfortunately neither efficiency evaluation nor operational comparison with other correlators.

Alerts are not the only relevant information that should be correlated. In fact, many research activities have been led on event management. Indeed, detecting event patterns (sometime referred to as *situations*) and reacting to them are in the core of Complex Event Processing (CEP) and Stream Processing (SP), both of which play an important role in IT technologies. Business intelligence, air traffic control, collaborative security, complex system software management are examples of such applications.

CEP is an active research field recently used for the development of monitoring and sense-and-respond applications @Luckham. It addresses two crucial prerequisites in building highly scalable and dynamic systems:

- Mediation of the information in the form of events.
- Detection of relationships among them, i.e. temporal relationships that can be identified by defining correlation rules (often called Event Patterns).

In CEP, three fundamental concepts are defined:

- Event streams,
- Correlation rules, and
- Event engine.

An *event* is a representation of a set of conditions in a given time instant. Events belong to streams: events of the same type are in the same event stream [3].

Correlation rules are commonly defined by SQL-like queries (but also other types of queries are possible [7]) used by the *event engine* in order to correlate events that is to discover temporal and spatial relationships between events of possibly different streams (to filter only the relevant ones, or to perform calculation among them).

The *event engine* takes as an input the streams and correlates events from the event streams according to the correlation rules. The type of correlation that can be analyzed can be very complex as events can be correlated both spatially and temporarily, it can be joined and filtered, new streams can be created at runtime and also new events can be created composing other events generating what is commonly called *complex events*.

CEP is a widespread approach to the on-line correlation as it is able to handle the complexity of defining the element of the correlation itself (e.g. pattern detection, event filtering, event aggregation over time etc...) and it is applied in several different contexts ranging from business process management and automation (process monitoring, BAM, reporting exceptions, operational intelligence) to finance (algorithmic trading, fraud detection, risk management), distributed systems (network and application monitoring, intrusion detection, SLA monitoring) and sensor network applications (RFID reading, scheduling and control of fabrication lines, air traffic).

At first glance, the correlation done through CEP may resemble classical approaches to data analysis performed on classical SQL databases. However, CEP engines do not query a repository to look for events matching some conditions, but instead they trigger customized actions as the flow of events

comes in and matching event conditions making it appealing for application that requires on-line approaches to support near real-time monitoring and detection. Technically there are other important differences between how data are processed in classical DBMSs and the Data Stream Management Systems. Table in Fig. 3.6 emphasize these differences.

Data Base Management Systems (DBMS)	Data Stream Management Systems (DSMS)
Persistent relations	Transient streams
One-time queries	Continuos queries
Random access	Sequential access
"Unbounded" disk store	Bounded main memory
Only current state matters	Historical data counts
No real-time services	Real time requirements
Relatively low update rate	Multi arrival and variable rate
Data at any granularity	Data at fine granularity
Assume precise data	Data imprecise

Main differences between DBMS and DSMS.

Several CEP engine are centralized in nature, in the sense that all the events flow toward the CEP that takes care of the correlation, i.e. sources generate simple events, which are continually pushed to a processing site where the registered complex events are evaluated trough continuous queries, trigger, or rules.

In [11], the authors note how this model is neither efficient (as it requires communicating all base events to the processing site) nor necessary, as only a small fraction of all basic events eventually make up complex events. Their goal is to avoid continuous global acquisition of data without missing any complex events of interest, as specified by the users. For this purpose, they generate a cost-based multi-step detection plan on the basis of the temporal constraints among constituent events and event frequency statistics. Each step in the plan involves acquisition and processing of a subset of the events with the basic goal of postponing the monitoring of high frequency events to later steps in the plan. As such, processing the higher frequency events conditional upon the occurrence of lower frequency ones eliminates the need to communicate the former in many cases, thus has the potential to reduce the transmission costs in exchange for increased event detection latency. Plans are represented with extended finite state machines and the authors provide two algorithms, a dynamic programming solution and a heuristic method, for generating the plans with the goal of optimizing the overall monitoring cost while respecting the latency constraints.

Implementing centralized plan-based techniques in mobile delay tolerant networks is not feasible as they require distributed and scalable solutions in order to be executed on top of mobile devices. For this reason, in @CRLK11 the authors present a solution, called Comet, to distribute the computation effort among nodes in the system. Comet shares the task of detecting complex events (CEs) among multiple nodes, with each node detecting a part of the CE by aggregating two or more primitive events or sub-CEs. The tricky part is the distribution of CEs and to this aim the authors build a cost-and-delay efficient tree and incorporates a two-phase push-pull conversion heuristic that employs both single-target and multi-target pulls. Finally, they show that this solution, in most cases, produces significantly better plans than existing centralized mechanism.

The work presented in [15] even though is placed in the sensor networks environment (it faces energy consumption problem) presents interesting solutions for complex event detection. Complex events are sets of data points hiding interesting patterns. These patterns are difficult or even impossible to capture using traditional techniques such as thresholds. Moreover, system users may not know, a priori, what they are looking for. The presented algorithm allows the user to either specify the pattern they are looking for and then search for proximity to that pattern or makes an autonomous decision on what is interesting based on monitoring some normal data. In order to understand how much events match a

pattern or are outliers the concept of distance, rather than threshold, is used. Distance is evaluated by simple machine learning techniques. There are three main phases:

- *Learning phase*, where the algorithm keeps track (learns) of the distances it has seen.
- *Detection phase* during which distances are calculated. An event is flagged if the observed distance is greater than the maximum distance observed during learning.
- *Escalation phase* is a temporally deferred operation which is used to increase the accuracy of the algorithm (by pruning false positives). It determines whether the point of change, from the previous phase, was a true or false positive.

So a distance metric is used either to perform exact-matching or approximate-matching of some user supplied pattern either to perform non-parametric event detection.

Recently, CEP has been also used to detect specific type of attacks (or basic step of a multi-stage attack). Numerous proposals are emerging, in the literature, that use CEP as a support to correlate alarms detected mainly through signature or anomalies based techniques.

In [5] Ficco and Romano propose an Intrusion Detection and Diagnoser System (ID2S) where a CEP is used to capture causal relationships among intermediate alarms. The system correlates alarms on the base of temporal and logical constraints. The presented ID2S architectural model consists of a collection of software components that transform raw attack symptoms into high-level intrusion scenario to alert the administrator or/and to perform a recovery action. The paper deeply describes the correlation process and the system architecture devised to implement it, the ontology representing the knowledge-base schema used by the detection and diagnosis process and ends with an example of attack scenario recognition (i.e. the complex queries performed by the CEP to detect an SQL injection scenario). As the authors argue, a common weakness of this correlation technique, also known with the name of signature-based, is that it requires specific knowledge about the attacks. Therefore, the major limitation of this technique is that it cannot understand if an unknown attack is in place, since its behavior is not defined.

The accuracy and speed of current IDSs can be significantly improved by using event-processing techniques that can parse and analyze traffic beyond the network layer and detect more sophisticated attacks, which require correlation between multiple protocol data units. This is shown in @APP_EE_MCN where the authors place event processing in the core of the IDS and propose novel algorithms to efficiently match vulnerability signatures. In particular, they describe how, through deep content inspection and state maintenance, it is possible to detect complex and sophisticated attacks (i.e. Conficker worm), that current IDSs fail to detect due to their limited capability of regular expressions in representing vulnerabilities.

Due to the expensive, incomplete and hard to maintain nature of the signature-based approach, Hobbach and Seeger, in the work @AMusingCEP, motivate substantial improvements of CEP technology by making the behavior of the infrastructure dynamic and by switching the detection paradigm from signatures to anomalies. They propose dynamic CEP infrastructure able to capture the normal behavior of monitored objects and to create all necessary continuous queries for the detection of anomalies automatically. Additionally, the infrastructure is adaptive to changes in the application context at runtime. This dramatically reduces the effort of building and maintaining CEP applications and ensures a high quality at any time. The user has still the opportunity to define queries and can so extend the monitoring logic. This results in the best of both worlds: an automatically created anomaly-based monitoring setup that can be extended by the user with little effort. They not only review traditional application domains of CEP showing how, better capabilities for the management of anomalies can improve existing applications, but also introduce the management of security anomalies in computer systems as a novel and challenging domain. In the last quarter of 2011 33% of web malware was *zero-day* malware. The ratio is growing and consequently today's signature based monitoring systems will become less effective and important.

As shown in [6], anomaly detection techniques can be used in several environments (i.e. cloud computing) where the main issue is given by attacks acting at different layers (e.g. cross-VM side channel attacks). The main authors' goal is to close this gap presenting a context-based anomaly detection framework. First they create a model to describe workflows, services and infrastructures. The workflow model, as used for compliance detection, is nothing more than a series of CEP rules that are verified by the CEP engine. Second they profile users, services and hosts, where any outliers have an impact on the activity profiles of the given entity. These outliers are used to generate fingerprints which are used to compare entity's behavior to others, but also to measure potential deviation of its own behavior over time. Finally, the authors perform anomaly detection by clustering fingerprints: less is the number of data instances inside a cluster, more probable is that those data instances are in fact outliers. The key feature of this framework is the ability to keep multiple profiles of entities on various layers and to link detected anomalies and semantic gaps up to workflows in order to detect anomalies and false-positives.

A recent application of CEP occurred in the CoMiFin European Project [1]. The aim of CoMiFin is to develop a middleware for collaborative protection of networked financial players from cyber-attacks by analyzing network data sources coming from several financial institutions. In [14] the idea of the project and the *Semantic Room* abstraction is presented.

The SR abstraction has been defined as complex threats are extremely difficult to detect locally by single organizations in isolation. For this reason information sharing among groups of organizations is pushed. Exploiting the collaboration, different entities shares their network data that are analyzed by a CEP engine in the semantic room to detect if a specific attack pattern is possibly occurring at different site (i.e. CEP performs temporal and spatial correlation to detect complex events).

After having considered recent research results we now quickly describe some available tools/engines supporting CEP.

Esper [2] is an open source Event Stream Processing (ESP) and Complex Event Processing engine available for both Java and .NET (called NEsper). It is able to process large amount of data represented in the form of messages or in the form of streams, by filtering, correlating and combining events. In particular, Esper is able to trigger actions defined by users (and expressed in the POJO format) as soon as some conditions occur over the input stream. Such conditions are expressed as query in an SQL-like language called *Event Processing Language* (EPL). EPL supports all SQL's conventional constructs (i.e., Group By, Having, Order By, Sum, etc...) and, in addition, it adds further constructs, like `pattern`, that enable the developer to spot complex correlations among events. As a consequence, EPL is able to define conditions to easily detect *patterns* among events.

InfoSphere Streams [12] is a stream processing platform developed by IBM allowing the development of applications aiming at quickly ingest, analyze and correlate information as they arrive from thousands of real-time sources. InfoSphere Streams can handle very high data throughput rates, up to millions of events or messages per second. While there are other systems that embrace the stream computing paradigm, InfoSphere Streams takes a fundamentally different approach for continuous processing and differentiates with its distributed runtime platform, programming model, and tools for developing continuous analytic applications. In particular, InfoSphere Streams is able to handle data originated from sensors, cameras, news feeds, stock tickers, or a variety of other sources, including traditional databases. The InfoSphere Streams architecture shares some similarities with CEP systems but its main advantage is its supports for higher data rates and a larger sets of data types. It also provides infrastructure support to address the needs for scalability and dynamic adaptability, like scheduling, load balancing, and high availability.

JBoss Drools Fusion [8] is an independent module (but still completely integrated with the rest of the platform) with the following main features:

- understand and handle events as basic elements of the platform,
- select a set of interesting events in a cloud or stream of events,

- detect the relevant relationships (patterns) among these events and
- take appropriate actions based on the patterns detected.

In the Drools philosophy, events have unique and distinguishing characteristics and they are related by strong temporal constraints. Drools Fusion understand events by what they are and allow users to model business rules, queries and processes depending on the occurrence or absence of events and supports temporal reasoning, i.e. it offers a complete set of temporal operators to allow modeling and reasoning over temporal relationships between events. In addition, Drools Fusion supports reasoning over absence of events i.e., it allows to model rules and processes that react to the absence of events and it supports sliding windows for event correlation.

TIBCO BusinessEvents [13] is a complex event processing solution that abstracts and correlates meaningful business information from the events and data circulating inside the company information systems. The key features offered by this CEP are about the modeling, the rule engine and the way in which it captures the events. Business Event requires an UML-based state model to describe how applications and services interact as part of activities and processes. BusinessEvents enables the definition of causal and temporal relationships between events trough the definition of rules and queries over event streams used to match event patterns.

3.2.1.2.2 Implicit Correlation Approaches

The explicit correlations require specification of relations between the alerts or between the states of the system. These relations must be expressed by an expert. This can be particularly difficult to describe. The implicit approach does not require such an explicit information. The implicit correlation consists of bringing out relations between alerts, without pre-established schemes. These relations can be similarities between some alert attributes, frequency or statistical correspondences between the alerts.

In [10], the authors propose an implicit approach that consists of computing a distance between alerts that were beforehand enriched. For each alert, the probability that it is effectively part of a scenario is measured. This probability relies on three indicators that are then weighted. The first indicator is the probability that an alert of type *j* follows another alert of type *i*. The second indicator is a temporal value comprised between 0 and 1: the closer the alerts are to each other temporally, the higher its value. The third indicator is a value between 0 and 1, depending on the distance of the source IP of the two alerts.

A second implicit approach described in [132] uses a probabilistic approach similar to the one used in [143], that we have previously presented in alert fusion section. The correlation probability between two alerts depends on a prior knowledge that describes the probability that an attack type is the consequence of another attack type. Actually, it is a variant of the semi-explicit approaches. The same authors propose in [131] to correlate alerts using statistical and temporal properties. The proposed approach relies on Granger causality test (a variable is correlated with a variable x if y is better predicted by the suite of x and y previous values than only by the y previous values).

3.2.1.2.3 Semi-explicit Correlation Approaches

The semi-explicit correlation generalizes the explicit approach. Like the explicit approach, it uses expert knowledge on the attacks. However, instead of recognizing pre-established scenarios, the approach intends to detect inconsistencies in the pre-requisites (pre-conditions) and the consequences (post-conditions) of the attacks that reveal the evolution of an intrusion scenario.

A set of work is based on this idea and has been initiated by [137]. The authors propose an attack description language, Jigsaw, which focuses on the expression of pre-requisites of an attack and its effects from the point of view of the attacker. We remark that Jigsaw aims at describing attacks, i.e. basic steps of an intrusion, and not at describing intrusion scenarios. As far as we know, the authors do not provide an operational semantics of their language.

The Lambda language proposed by Cuppens and Ortalo [115] allows specification of the pre-requisites and the effects of attacks. A semi-explicit correlation is thus proposed in [114]. Each elementary attack in the scenario is defined by a pre-condition and a post-condition. Two types of correlations are in practice proposed:

- The correlation post-pre consists of linking an elementary attack whose post-conditions satisfy the pre-conditions of another attack. This type of correlation denotes an evolution of the attacker intrusion scenario.
- The correlation post-post consists of merging two attacks whose effects coincide. It denotes several different attempts of the same attack to reach its objective.

3.2.2 Risk Assessment

3.2.2.1 Analysis of the impact of scenarios on the system

During this phase of the correlation process, the meta-alerts should most certainly be due to real attacks. The final objective of the correlation process is thus to determine the impact of the detected intrusion on the system and its assets.

The correlation functions previously presented put in relation activities on the system: activities are linked with some others or with information in the system configuration. However, to evaluate the impact of an attack on the system, it is sometimes necessary to have complementary information describing the links between the different components and services deployed on the system.

In the example on *Figure 9*, the mail server uses the DNS service, the Webmail interface is using the web server and the mail server. The Web server relies on the DNS proxy that depends on the DNS service. If we suppose that the DNS service is attacked, all services depending on this service may be compromised. The complementary information describing the links between the components and services in the system allows identification of the set of components and services impacted by the attack against the DNS service. New meta-alerts are emitted to inform the administrator about the impact of the attack.

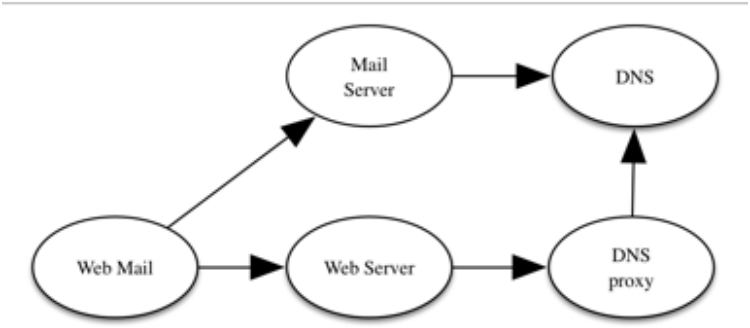


Figure 9: Service Dependencies

Very few articles treat specifically the problem of impact analysis. We can although cite (@PFV02). This work defines criticality levels for the several types of attacks against the system. These levels are used to compute a global impact score for each alert. This score permits to classify the attacks. We explain this aspect in the next section.

We can also cite (@Lag10) that focuses on the impact analysis in the DRA project (Dynamic Risk Assessment). The approach to real time determination of the impact of attacks on the assets, services and missions of the highest criticality in the monitored organization. The approach deals with a risk analysis statically computed at the system initialization. This analysis is updated dynamically

depending on the events that are produced in the system (new software versions, knowledge of new vulnerabilities, alert reception).

DRA combines the use of an automatic attack tree generation and a computation of a risk analysis. Using the description of the monitored system, an attack graph is generated automatically by using the MulVal[67] tool. The AssetRank (@SO08) tool is also used to emphasize in the results given by MulVal the most critical nodes of the attack graph. AssetRank provides a way to identify the most probable attack path depending on multiple criteria (exploit simplicity, stealth, target criticality, etc.). Finally, the PILAR[68] tool allows the computation of the risks induced by an attack, similarly to a risk analysis. This computation exploits the system configuration, their dependencies, their criticality and their state (normal, vulnerable, exposed or compromised).

3.2.2.2 Alert prioritization

The alerts and the phenomena they emphasize have not an absolute criticality level. Indeed, this level depends on the monitored system, its objectives, and the context in which it is executed.

Consequently, all the services proposed by a system do not have the same criticality levels. These levels must be included by the administrator in the system configuration. We can then use this information to determine the priority level of the alerts: for example, an alert involving a service of high criticality will have a higher priority. This approach is very simple by design, but constitutes a natural way to finalize the correlation process.

Moreover, since we try to display only the alerts that have the highest semantic, i.e., the meta-alerts that are the results of the global correlation process (from the alert fusion to scenario recognition), we can consider that the meta-alerts have a higher priority than the alerts from which they are issued.

Few academic works exist on the subject. However, we can cite (@AAB08) that proposes an approach for alert notation to determine the criticality of the alert: this notation relies on the elements used during the correlation process and the sensitivity level of the attacked service. In (@NJ08), attack graphs are used to compute the criticality level of an alert by measuring the distance in the graph between the action that triggered the alert and the assets that are the most sensitive.

3.2.3 Leveraging Response Assistance

We presented in the previous sections how alerts and events are correlated. In the context of the PANOPTESEC project, we do not only intend to inform the user about what is happening on the system but also want to automatically propose and/or deploy reactions to mitigate the attacks.

3.2.3.1 Adaptive Intrusion Response System

The response model presented in an example of automated and adaptive response system presented the design and the implementation of an Adaptive Intrusion Tolerant Systems (ADEPTS) for containing intrusions in large and distributed systems. ADEPTS relies on intrusion graphs (I-Graph) which model the attacks goals. An I-Graph represents all the possible paths that the attack can undertake from a given service to its neighbor. Consequently, the response system determines the possible spread of the attacks. The system maps the alerts generated by IDSs to I-Graph nodes, and estimates the likelihood of the attack spreading, based on the alerts confidence levels. Finally, the most appropriate response is selected, by targeting the nodes with the highest likelihood, in order to contain the ongoing attack. The selection procedure considers: (i) the survivability metric that assesses the disruption of the response for legitimate system activities, (ii) the previous success of the response, and (iii) the confidence index

[67] http://people.cis.ksu.edu/~xou/mulval/

[68] http://www.ar-tools.com/en/index.html

which evaluates the confidence that the determined intrusion is indeed taking place. The response has the goal of preventing the escalation of the intrusion and the possible spread from one service to another. Moreover, ADEPTS supports an automatic update of the response effectiveness, without the need of the administrator's intervention.

The continuous feedback of the response effectiveness makes ADEPTS an adaptive response system. It is also an automated response system since there is no need for administrator's intervention. Moreover, ADEPTS is considered preventive: the response system tries to anticipate the future nodes in the I-Graph, and blocks them in order to contain the ongoing attack. Another advantage of the proposed system is its capability to handle unknown alerts. This is done by creating a generic node per host. As this node represents an unknown vulnerability, it is connected to all of the other nodes in the I-Graph. Another important advantage of ADEPTS is the cost-sensitivity. The response selection procedure is based on the response index. Response index depends on the effectiveness index of the response and the disruptiveness index, which are specific to the response itself. Thus, ADEPTS selects the response which offers the highest difference between these two metrics.

However, ADEPTS has several limitations:

- ADEPTS considers only a single type of response: containment of ongoing attack. Containment implies restricting the effect of the ongoing attack to some of the services, and preventing the attack from spreading to other parts and services in the monitored system. In other words, ADEPTS aims at containing the detected attack rather than halting it completely. It is obvious that this type of response is not convenient for all types of attacks (e.g. DoS attacks).
- To determine the spread of the attacks, ADEPTS relies on I-Graphs, which are manually specified and developed. The graph is static and acts as auxiliary information used in conjunction with the actual intrusion detection system. If the monitored system changes, the I-Graph needs to be updated accordingly using expert's guidance and knowledge.
- ADEPTS adopts a cost-sensitive approach, it selects the response action with the highest difference between the response effectiveness and disruptiveness during normal activity, without considering the damage of the attack itself, and the possibility that it may or may not exceed the benefits of the response.

3.2.3.2 Cost-sensitive Intrusion Response System

Stakhanova et al. propose a cost-sensitive intrusion response system. This model compares the costs of deploying a response with the costs of damage caused by a non-responded attack. Additionally, a methodology is proposed to adapt responses in a changing environment by considering the previously applied countermeasures through a feedback. The method proposed in this approach uses two very simple metrics: (i) the response cost RC, and (ii) damage cost DC. While the former represents the impact of a countermeasure on a system, the latter generally quantifies the impact incurred by the monitored system due to the attack's occurrence. Similarly to other proposed systems, the authors recognize that assigning accurate values to those metrics is challenging, and requires a high level of expertise and knowledge. First, a preliminary set of applicable countermeasures is selected. A countermeasure belongs to this set if:

$$DC \times CL > RC$$

where CL is the *confidence level*, i.e. the probability that the attack is actually occurring.

Second, the most appropriate countermeasure of the previous set will be selected. The selection procedure is based on metrics: (i) the *success factor* SF, and (ii) the *risk factor* RF. The former is the percentage of times that this response succeeded in the past, whereas the latter represents the negative impact that this response has on the monitored system. Clearly, the countermeasure which provides

maximum benefit with minimum risk is selected. Thus, the response *rs* that has the highest expected value *EV(rs)* is selected to counter the attack sequence *S*, given that:

$$EV(rs) = Psucc(S) \,.\, SF + Prisk(S) \,.\, (-RF)$$

where *Psucc(S)* is the probability that attack sequence *S* occurs, while *Prisk = 1 - Psucc(S)*. The *success factor* is updated permanently: it is increased by one if the response succeeds in terminating the attack, otherwise it is decreased by one.

The response system proposed by Stakhanova et al. is automated and cost-sensitive. It is also preventive because the response is triggered before the attack completes. Additionally, the response system is adaptive since the Success Factor is updated each time a response is launched whether successfully or not. On the other hand, a main challenge is setting accurate measurement of the cost factors. This requires a high level of knowledge of the system and overall expertise. Moreover, these values must be updated while considering the dynamic nature and continuous evolution of monitored systems. Last but not least, the authors defined the risk of a response as an impact. This is again a major and common misuse of risk.

3.2.3.3 Intrusion Response System considering resources dependencies

Jahnke et al. model the monitored system using directed graphs. Afterwards, quantified metrics derived from these graphs are computed in order to evaluate the effects of a response action. As a consequence, the most appropriate response can be identified and launched. This paper considers only the availability property of the monitored system, while confidentiality and integrity properties are excluded. Similarly to previous work, the authors consider the users and the services as resources. Every resource has a certain availability metric *A(r)* expressed between 0 and 1. The availability value of a resource can be both inherent, and a result of the propagation of modified availabilities of other resources on which the considered one depends. Therefore, the authors define for each resource *r* an *intrinsic availability AI(r)* and a *propagated availability AP(r)*. The availability of *r* can be computed as:

$$A(r) = AI(r).\, AP(r)$$

We note that different kinds of dependencies can exist between resources. The users depend on services in the monitored system, and services often rely on other services. For example a network communication system depends on the availability of a directory service. In order to model the dependencies between the system's resources, the authors define a directed dependency graph. Whenever a resource *r* depends on another resource *s*, the edge *(r,s)* is added to the graph. Each edge is associated with a subjective weight *w(r,s)* (between *s* and *r*). The dependency types that are considered by the authors are: mandatory, alternative, combined, m-out-of-n and indirect.

Each time the diagnosis systems indicate a modification in the availability of a resource, the availability of resources which directly or indirectly depend on the modified one, need to update their values immediately. The propagated availability values *AP(r)* are updated using the dependency graph and an inverse breadth-first-search (BFS) algorithm. Finally, the availability of the entire system *A(G)* is computed.

Regarding response, the effectiveness of a candidate countermeasure can be evaluated. More precisely, the system's graph for the system state prior and after the activation needs to be generated. The candidate countermeasure which leads to the highest A(G) is selected.

The limitation of this model is that it focuses only on the availability of the resources. Neither the confidentiality nor the integrity of the resources is considered when selecting the appropriate countermeasure. Another major limitation is the manual and error-prone work that consists of setting the values of the weight *w(r,s)* of the dependencies.

While Jhanke et al. consider only the availability of resources in the monitored system, Kheir et al. extend the previous approach, by considering the confidentiality and integrity dimensions in the dependency graphs. The confidentiality and integrity properties of a resource may only be verified if the resource itself is available. Hence, Kheir et al. consider the effect of the availability level both on confidentiality and integrity levels. The major challenge is the system's modelling, given that the model has three layers: confidentiality, integrity and availability. In the dependency graph, each edge is associated with a 3x3 weight matrix, while in the previous work each edge was associated with a single weight value. Consequently, the system modelling process requires even more effort and higher expertise's level.

Moreover, Kheir et al. consider only the following types of countermeasures:

- Countermeasures that deactivate (shut down) a given component (e.g. delete file, remove account, kill process, stop service, quarantine host, etc.). A deactivated component does not propagate confidentiality and integrity costs. However, as it is unavailable for its dependent resources, it propagates availability cost.
- Countermeasures that alter dependencies: dependency alteration makes only the dependency unavailable for a given antecedent resource with responses (e.g. block connection, unmount file system, change process owner). It makes the dependency unavailable only for the concerned resource, but not all the children of a specific resource.
- Countermeasures that restore resources and partially reduce the attack impacts were not considered in this model.

3.2.4 Products, Processes, Services and their deficiencies

3.2.4.1 Security Information and Event Management description (SIEM)

Security Information and Event Management description (SIEM)

As introduced by Gartner's analysts Mark Nicolett and Amrit T. Williams in 2005 [102], and since take up in each Gartner's Magic Quadrant report by Mark Nicolett and Kelly M. Kavanagh [103], Security Information and Event Management (SIEM) is a technology that characterizes the software, appliances and managed services covering both the Security Event Management (SEM) and Security Information Management (SIM) functions for the management of security events across networks, systems and applications of an ICT infrastructure.

Security Information Management is often referred to the Security Management technologies dealing with the long-term storage, analysis and reporting of log and event data generated by network pieces of equipment, systems and software deployed in an ICT infrastructure. These features are commonly referred to as Log Management and have the purpose to extract trends that may support the discovery of "bad behaviors", the reporting about compliancy over regulatory requirements (e.g.[104], [105], etc.) or forensic.

On the other hand, Security Event Management deals with the Security Management technologies enabling the real-time monitoring and incident management of security-related events from network and security devices, systems and software of the ICT infrastructure. The SEM features usually rely on various tools for enrichment and deep analysis of security-related events and a Correlation Engine to extract targeted behaviors of interest that may be used to qualify the security incidents.

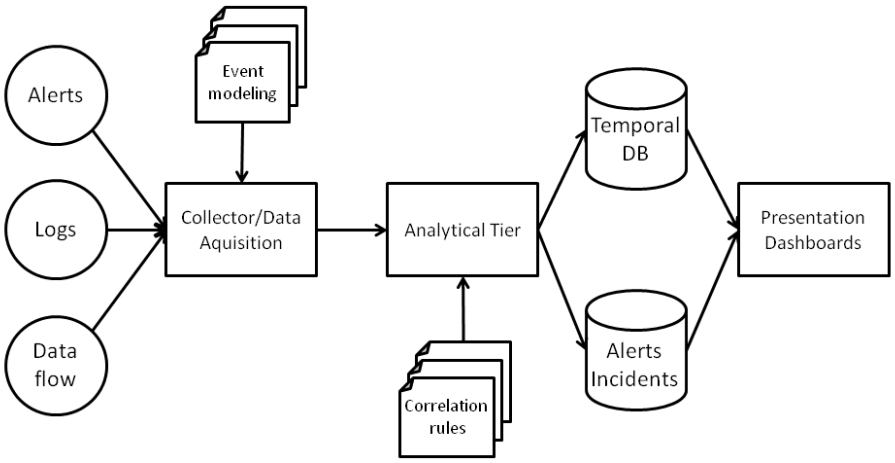


Figure 10 - Logical Security Information and Event Management architecture

The simplified logical architecture of a typical Security Information and Event Management system may be represented by *Figure 10*

Main capabilities

Although a wide variety of reasons may guide an entity to deploy a SIEM infrastructure, security practitioners Adrian Lane and Mike Rothman [111] retain three objectives or use cases as valid:

- *React faster to attacks*: The main issues faced by security analysts managing the Information Security of an ICT infrastructure are, the large number of piece of equipment and software managed, the huge volume of data and events related with security generated in a decentralized way with various specific semantics, and the high ratio of noise over real evidence of attacks hidden within those data and events. Moreover, analysts often face the lack of a global technical view (e.g. a detailed up-to-date topology and inventory of the infrastructure) and business context (e.g. identified critical assets for the business of the entity, clear link between assets and supporting assets) that would not enable them to properly prioritize the resolution of problems that security alerts raise. In this context, security analysts need tools that help them to extract rapidly pertinent information from a wide volume of data and present this information in the clearest way so that deciding how to react is eased.
- *Improve efficiency of existing technical management teams*: Nowadays, as companies and entities rely more and more on ICT for the operation of their business, their infrastructures tend to grow in size and complexity. Moreover, the regulatory requirements on those infrastructures are becoming stricter each day. Definitely, the volume and sophistication of attacks is also increasing. On the other hand, ICT management teams are always demanded to enhance their monitoring scope, whereas they often have to face a cost center reality that prevents those teams to grow in size proportionally.
- *Automate the compliance verification*: A security that is not measured is potentially an inefficient one. Moreover, knowing is better than ignorance in the security domain because it permit the setup of contingency and recovery plans. We then have seen the proliferation of a number of Security Control and Compliance standards emerging in the past twenty years. Some of them are simply guidelines and have no value of obligation (e.g.[105], [106], [107] and [108]), but some of them are regulatory requirements depending on the business domain in which a company or an entity evolve (e.g. [104], [109], etc.). When the managed infrastructure is extremely large, composed of ten-thousands of devices and applications, spread over hundreds of geographical locations, security practitioners may have difficulties plan the preparation for audits if this preparation is not done on a strict and regular basis. Moreover, the time required to get prepared

for an audit usually impairs their capacity to respond to security alerts. In those contexts, there is a need for automation to collect required data to establish that security controls are in place. Obviously, it may reduce the time and repetitive annoyance of establishing such kind of information in anticipation of an audit.

From the various capabilities of SIEM platforms that such systems may share with both SIM and SEM, the three listed use cases enable to identify the main capabilities that are required for a Security Information and Event Management system:

- *Aggregation*: one of the base feature of a SIEM is its ability to aggregate (i.e. collect and interpret) data coming from a wide variety of sources in an ICT infrastructure in a central and accessible point. This ranges from networking and security pieces of equipment or software, like switches and routers, network services (e.g. IPBX and Media Gateways) and applications (e.g. proxies and Web servers, groupware servers etc.), firewalls and authentication servers, IDS/IPS and anti-viruses.
- *Normalization*: While aggregation merges data representing dissimilar events in a common repository, the normalization purpose is to filter data attributes representing information of the same semantic level (e.g. usernames, application, activity, network addresses, event time, protocols or services etc.) in a generic template. This step then omits in the resulting events any information not complying with the semantic level of the generic template. Data normalization enable to store most important information on original events in a consistent way, which make it easier for indexing and efficient searching over huge volume of data. While, crucial on early SIEM systems, this feature is now less useful as indexing and databases techniques for huge volume of data has constantly evolve on such platform.
- *Correlation*: It is usually admitted that correlation is the process by which a meaning is extracted from several related events of the same or different kinds. Correlation tends to provide a broader context to correlated clusters of events than the individual event that composes the cluster. Correlation often makes use of *a priori* information on events or on the ICT infrastructure to link together events into meaningful clusters. A wide variety of correlation techniques exists from the simpler clustering of alert bursts, up to the more complex which try to forecast the intention or plan of an attacker. SIEMs platforms usually implement automation for the most simple correlation techniques. And, the more complex scenario recognition is often left to human security analysts.
- *Alerting*: It is the process by which the information on fully qualified incidents, produced by mean of the events correlation followed by the analysis and assessment by security experts, are issued to an appropriate destination so that actions can be taken. Usually, this information is issued though alerts, sent to identified Decision-Makers (for administrative actions), Administrators (for technical actions) or programmatic agents (for instant and automatic technical actions). The Alerting process should usually rely on policies, which identifies the predefined responses to be triggered for each identified suspect event. Where and how the alerts are sent (e.g. email, SMS, SNMP, specific API calls), the kind of information in the alert, and the way the criticality is determined should also be defined in the rules of the policies.
- *Dashboards*: As for any kind of monitoring system SIEMs have the need to represent collected, managed and assessed data in a synthetic and (semi-)real-time shape. The main purpose is to assist Decision-Maker, Experts and Administrators in seeing patterns of various activities, or activities which deviate from a standard pattern. Dashboards should then give the ability to show an overview of the global state of the monitored/protected ICT system based on an increasing number of events continuously collected. Usually, a specific dashboard is dedicated to a specific user, or kind of user, of a SIEM system, so that a role based approach of dashboards is generally adopted. Dashboards also need to be customizable in term of sources of data used and shapes for the representation of those data to adapt to the kind of user to which it is dedicated.

- *Compliance*: As most companies and entities tend to focus a lot on Security Controls deployed in the monitored/protected ICT system, and the documentation about the conformity of those Security Controls, the collection of proof of compliance of the Security Controls, which has to be done for each Audit's occurrence, stays a cumbersome, repetitive and error-prone process relying mainly on Administrators. Moreover, while most of the Security Controls Conformity standards address the same set of controls, each of them has their own rule. A process that automates most of the collection of proof of compliance and prepares them in the format for the different standards reduces the burden of Administrators, and the risk for errors inherent to any fastidious and repetitive human process. As many companies and entities of the same activity domain has to conform to the same rules, modern SIEM platforms now propose template for the most common Security Controls standards (e.g.[105], [104], [109], etc.).
- *Workflow*: Information Security Management is by definition a collaborative process which requires organization, timeliness and clear Separation of Duty (SoD). A workflow enables, to keep the track of any action with the correct identification of who has done the action, and to guarantee that administrative actions are taken without conflict between different levels of responsibility (Decision-Makers, Expert, and Administrators) that may impair the integrity of the reporting.
- *Data Retention*: As data normalization is a process of filtering relevant information and correlation the process of determining causality dependencies between different events, some details about the initial information and events are necessarily lost. Data retention is the process that guarantees the secure archival of non-normalized original events. Indeed, when a technical analyst qualifies a security incident, he or she sometimes needs to substantiate his conclusions by producing the original data in order to prove the correct foundation of his opinion. This may happen when an auditor demand to demonstrate that a security control works as expected and efficiently, or when a forensic analysis is required. Moreover, regulatory requirements on data retention exist in most countries.
- *Forensic*: it is the process by which a verified problem or a qualified incident is investigated to find the root cause. Forensic analysis should also provide the ability to extract information on the context of an incident to determine the extent of the incurred damages. Forensic capabilities of SIEMs should then enable to search efficiently (e.g. as quick as possible) across data and logs, of any node of the monitored/protected ICT infrastructure collected and stored by the platform, based on criteria. These features are an assistance to aggregate a huge among of information extracted from stored logs in a timely manner.

Typical architectures

Usually, it is considered [111] that architectures for a SIEM deployment can be of four different kinds: flat central, hierarchical, ring and mesh collection. Each of them has specific advantages which may be used to attain specific purposes. Current SIEM products are often able to adapt to those specific kinds of architecture for their deployment, which will optimizes the platform regarding various criteria like, coverage, scalability, analysis performance, reporting performance, storage or cost.

Moreover, we can remark that most products are rather specialized regarding SIM and SEM functions. Even if most products offer nowadays both type of features, some of them are more versatile in the Log Management (LM) domain, while others are more adapted to correlation. An effective SIEM deployment may then require the collaboration of two products which may guide the definitive architecture of the SIEM platform.

Flat Central

The *Flat Central* architecture (cf. *Figure 11*) corresponds to the oldest kind initially used in early SIM platforms. It is composed of a unique collection point that centralizes all the logs and data aggregated by the various sensors and log sources. In this kind of deployment, all the burden of

storage, normalization, and correlation is dedicated to a central point, often represented by a single appliance.

The main advantage of this architecture is the simplicity. The data cannot suffer from any synchronization issues and all policies (e.g. storage, log, correlation rules, alerting and reporting etc.) are centralized at a single location. Obviously, its main disadvantage is the scalability in term of collection, aggregation, correlation and storage. Nowadays, this kind of architecture is no more adapted to other case than small and medium entities with feeble data collection needs and with the purpose to produce basic compliancy for moderate size ICT infrastructures.

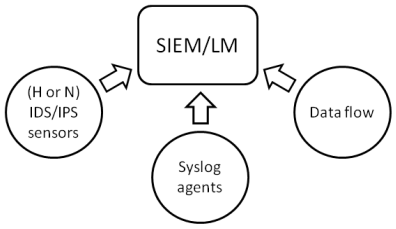


Figure 11 - SIEM Flat Central architecture

Hierarchical

The *Hierarchical* (cf. *Figure 12*) deployment model relies on a central treatment point. But, unlike the Flat Central model, the various sensors and data sources agents are not directly reporting their alerts and collected data to the central treatment point. They rather send their information to intermediary treatment points which are themselves connected to the central treatment point.

This kind of architecture is particularly adapted to protect/monitor ICT infrastructures spread over several physical locations with limited/moderate bandwidth between them. This enables to dedicate the simplest treatment, and the storage of raw data of events (i.e. future forensic analysis), to intermediary/regional treatment points. While the central treatment point manages the most complex level incident by aggregating and correlating lower level already qualified events, then alerting and reporting to the higher level of the hierarchy.

The main advantage of this kind of architecture is the scalability across large ICT infrastructures in term of bandwidth and storage performance. Another positive consequence of this kind of deployment model is the ease in implementing SoD. Overall, this architecture render possible to distribute many functions on several lower responsibility nodes of the architecture, which ease the management of large data sets. On the drawback side, the complexity in configuration and management processes of the SIEM platform may be an obstacle. This render this kind of architecture particularly adapted for large entities, or for service providers of Information Security Management.

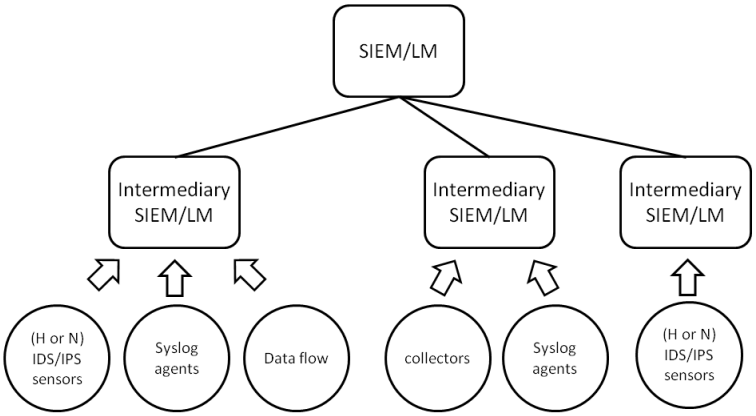


Figure 12 - SIEM Hierarchical architecture

Ring

Like the two previous models, the *Ring* architecture (cf. *Figure 13*) relies on a central point of treatment. And, like the Hierarchical model, it is based on intermediate treatment points connected to the central point on one side, and to sensors and data sources agents on the other side. But, unlike the hierarchical model, the intermediary points are not of the same kind as the central point. In fact, the intermediary nodes only assume the role of Log Managers, and implement also distributed reporting. All the treatment burden of aggregation, correlation, analysis, and alerting is dedicated to the central treatment point which implements most of the real SIEMs' capabilities.

Additionally to the scalability in term of storage and distributed reporting, the advantage of this architecture is the cost reduction that may result from the use of two different and specialized products: a cheap one for Log Management and distributed reporting, and a more expensive one for a high performance central SIEM component. On the other hand, the use of two different products may lead to overcomplicating the integration of products not necessarily designed to work in unison.

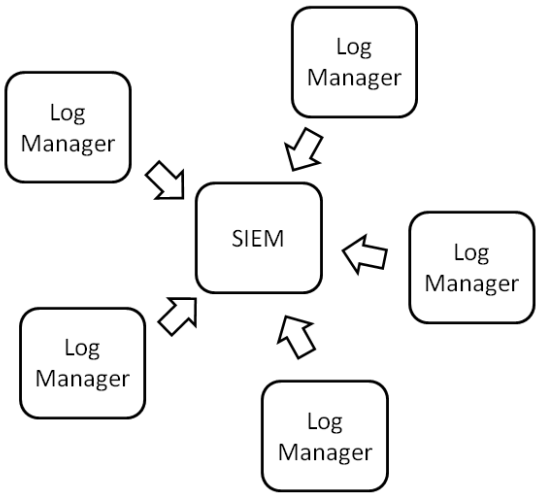


Figure 13 - SIEM Ring architecture

Mesh

The *Mesh* architecture (cf. *Figure 14*) rely on the model of a group of interrelated sub-systems, performing each a part of a global system by sharing a common information basis over a communication layer. The global system then performs the functionalities of a global SIEM system, but with independent components. Each component performing a limited number of features (e.g. normalization, aggregation, correlation, analysis, alerting, or reporting), but in a much optimized way, and may be interconnected to any other component of the global system through a communication layer with a standard interface.

The obvious advantage of such architecture is the flexibility. It enables, for instance, to aggregate, correlate and analyzes specific parts of the monitored/protected ICT infrastructure independently. Indeed, depending on the needs, the communication can range from a full mesh to a hierarchy. Moreover, components from different provider may be associated (e.g. in order to use the most powerful component of each kind), provided that they share common interfaces. On the drawback side, the complexity in the deployment and management of such architecture may discourage entities with ICT infrastructure of large size to protect/monitor. Moreover, it is very difficult to implement an efficient Separation of Duties and a clear segregation of data over the whole system.

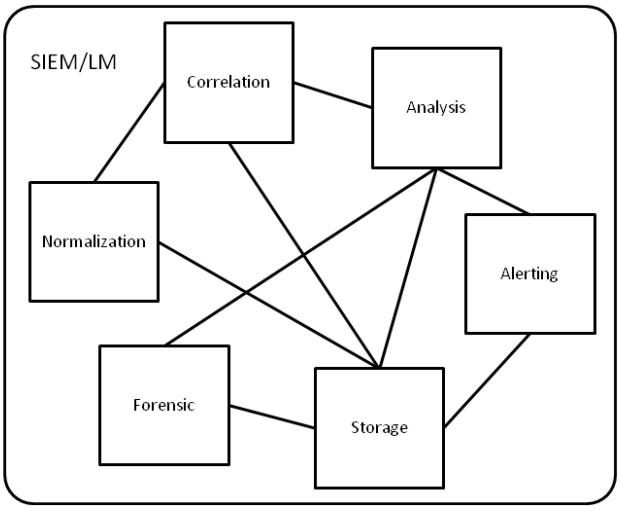


Figure 14 - SIEM Mesh architecture

It is worth noting that these diverse deployment architectures are not exclusive. Depending on the monitored/protected ICT infrastructure architecture, purpose and performance requirements, a mixing of several kind of deployment model may be done. But, constrains may exist depending on the product selected to implement a SIEM platform, as not all SIEM products can manage every architecture kind. Moreover, most products only implement less than three of these models.

Market and products

As for 2014, and after more than ten years of existence, the SIEM technology market can be assessed as mature. The competition is very high with a wide number of companies proposing SIEM platforms. Gartner analysts in their latest market study [103] lists sixteen major players is this domain, with one half of them now positioning themselves as Managed Security Services Providers (MSSP). Nevertheless, it seems, by the beginning of 2014, that quite 60% of the market is ruled by five large companies: HP, IBM, McAfee, EMC-RSA and Splunk.

The rest of the market is animated by a list of companies, that Gartner recognizes as less important and categorizes them in four groups: Leaders (LogRhythm), Challengers (Tibco-LogLogic, NetIQ, SolarWinds and Symantec), Visionaries (AlienVault) and Niche Players (Sensage, Trustwave, Tenable Network Security, EiQ Networks and EventTracker).

The Gartner analysis also identifies a group of smaller vendors, with small market shares or local representation: AccelOps, CorreLog, FairWarning, Lookwise, Tripwire Log Center, Tango/04 and Tier-3.

Deficiencies

The following list is among the main deficiencies of SIEM systems that we identify:

- *Limited Scalability*: Current architectures of SIEM systems rely on a centralization of the main treatment capabilities crucial for the analysis (e.g. correlation, storage). There is a need to improve the architecture distribution, by moving the query and analysis capabilities down, closer to the event sources.
- *Poor Analytics*: The analysis techniques to find the root cause, which are implemented in current products, are poor. The process then relies heavily on human expertise to, qualify the root cause and assess the risk (i.e. consequences and likelihood) of a threatening event.
- *Simple Correlation*: Often, only the simplest correlation (e.g. Normalization, aggregation) are implemented in SIEM products. But, the means that enable to express the complex relations

between events using correlation rules are uneasy to exploit and manage. SIEMs platform are then poorly efficient against targeted, multi-steps attacks and breach detection (e.g. APT).

- *Bad Filtering*: The exponential growth of the data produced by sensors render the experts' analysis cumbersome. The most simple correlation techniques do not permit to filter efficiently the most critical events, alerts and incidents among the huge amount of data collected by modern SIEMs platform.
- *Limited IT (NOC) Integration*: SIEM system are poorly integrated or interfaced with the ICT infrastructure topology and inventory management. It is then difficult to give context to events, alerts and incidents in order to assist the human experts in assessing root causes, impacts/consequences and the likelihood of threatening events.
- *Limited Vulnerability Assessment Integration*: Usually there is no tight link between SIEM platform and vulnerability assessment tools. Even if this interaction exists, no automatic scenario discovery is implemented in SIEM system in order to assist the human expert in its analysis. The exploitation of vulnerability inventory has to be done by an expert.
- *No Business level vision*: SIEM are mainly dedicated to the technical monitoring of ICT infrastructures. But, they are not adapted to the monitoring and management of businesses, processes or missions of the entity that rely on this infrastructure.
- *Poor Technical Abstraction*: SIEM platforms manage very technical and low-level information. This management revolves around the 3-tuple network traffic, packet density and IP addresses, which make it difficult for human analysts to assess concrete risks and deduce actionable remediation measures. The management paradigm should rather evolve to a more context centric approach, revolving around transactions, services (or applications) and users, making it easier to analyze for human experts.
- *No Risk level vision*: SIEM systems are dealing with technical monitoring. But, there is no integration with rational and clear Risk Assessment and Management processes. This kind of link implemented directly in SIEM tools could help human experts to assess the risk of threatening events.
- *No automated Response*: SIEM platform main function is to alert the correct person or group of persons that a threatening event has been detected. Base on the alerts, procedures are operated by a SOC team, a crisis cell and the ICT infrastructure administrators. Actually, as the filtering, correlation and analysis are not accurate enough, and provided that no consequences assessment of the possible responses exists in SIEM platforms, no pertinent responses could be deployed automatically by the system.

3.2.4.2 Security Operation Centers

A commonly admitted definition of Information Security Operations Center (ISOC) (e.g. McAfee2013 [159]), is the unit that performs monitoring, assessment, protection and defence of an organization's ICT infrastructure, through standardized and well-defined processes. As stated by many companies proposing ISOC services and solutions (e.g. [160] and [161]), a ISOC is composed of people, processes, and technologies enabling the centralized incident-response to Advanced Persistent Threats (APT), and reports to a Chief Security Office (CSO) or Chief Information Security Office (CISO). Similarly to their more general counterpart managing at a wider level (i.e. at physical and human level) the security of an organization, Security Operations Center, an ISOC deals with security issues both from a organizational and technical point of view.

ISOC generations

The emergence of ISOCs services followed a constant evolution during the four past decades. But, this has not been purely continuous, and ISOCs went through different eras of growing threats, which guided their evolution in stages, as HP remarked it in one of their white papers (cf. [162]):

- *First generation (1975 - 1995)*: First abuses have existed from the beginning of the automation of telecommunication systems in the early 1970's, when phreakers were able to take advantages of systems for their own profit. But, the real exploitation begins with the emergence of Internet with its flagship technologies, Ethernet and TCP/IP, in the end of the 70's and in the 80's. It then thrive in the end of the 80's with the appearing and development of nuisance programs and malicious codes (e.g. Christmas Tree Exec, and Morris Worm), now well-known as viruses and worms. But, in an era where few companies had networked computers, the global impact of those codes were minimal. From the first malicious program's debut up to more complex attacks performed by humans in the mid 90's (e.g. Mitnick's attack on the 25th of December 1994 [163]), several security tools appeared: antivirus (e.g. McAfee antivirus software in 1987), firewalls, proxies and first intrusion detection systems (i.e. NetRanger in 1995). In order to manage those individual and early security tools and address the issues caused by the threatening programs, Security Operations were created in some organizations. Initially operated by a single person, with networking and systems (i.e. Unix) skills, the Security Operations dealt by that time with the management of security devices and pieces of software. Real attempts to formalize a Information Security Operation Center appeared in government and military organizations in the beginning of the 90's (e.g. creation of the USAF AFCERT in October 1993). But, the analysis done in those ISOCs were mainly unstructured.

- *Second generation (1996 - 2001)*: This period is characterized by the widespread of new viruses and worms, which may sometimes paralyze the Information System (IS) of corporations and government organizations, in an era of local internetworking of computers development and rapid penetration of the interconnection of Information Systems to the Internet. In parallel to this threats boom, the security tools and products also thrive, with a particular focus on the development of Intrusion Detection Systems (e.g. the development of Snort begins in 1998) and Vulnerability Scanners (e.g. SATAN, SAINT, SARA, Nessus etc.). With the tools explosion, private companies begun to offer security monitoring (e.g. for log analysis of first commercially available IDSs) and management services (e.g. for the wide diversity of Firewalls), creating the Managed Security Services Provider's model. With professionalization, ISOCs initiated a formalizing of processes and procedures for intrusion response and vulnerability management. Although early SEM and Log Management tools emerged at the end of the period, technically, ISOCs teams' management of events remained largely based on scripts, dedicated tools management consoles and internally developed tools.

- *Third generation (2002 - 2006)*: On the threat side, this era is characterized by an expansion and professionalization of attackers with the monetization of attack capabilities (e.g. Botnets, 0days vulnerabilities), and the organization of cybercrime syndicates around identity theft, financial information theft, and threatening of IS for money extortion. We saw very high impact malwares causing major disruption on the Internet appear (e.g. SQL Slammer, Blaster), the emergence of targeted attacks, and the "militarization" of the cyberspace with first nation-states cyber-attack campaigns (e.g. Operation Titan Rain performed by China against US military and defense contractors sites). New usage of the Internet also emerged (i.e. Social Networks, peer to peer file sharing etc.), with their own threats and security issues (e.g. the Samy worm spreading across MySpace). While government, military, and MSSPs had already well-established technical processes, large private sector organizations began to create their own ISOC entities. The period is also very active on the development and enhancement of SEM and SIM tools' capabilities, and saw progressively emerge the concept of SIEM as a fusion the two. In parallel with the formalization of crisis management procedures by ISOC's teams to address the challenges of new threats and usages, ISOC units progressively replaced their internal tools, scripts and dedicated monitoring consoles with first generation of centralized SIEM platforms. The focus of ISOCs of that generation is primarily placed on early detection capabilities. But, with the development of security and data protection standards (e.g. Sarbanes-Oxley, PCI-DSS, HIPPA etc.), and the appearance of the first security breach

notification laws (e.g. California data security breach notification law SB 1386, Cal. Civ. Code 1798.82 and 1798.29, in 2003), a new focus appear on prevention and compliance.

- *Fourth generation (2007 - 2013)*: The period was marked by the explosion in volume and sophistication of attacks, the media exposure of high impact attacks and the new motivations of those targeted attacks (i.e. "hacktivism", cyber-espionage, sabotage). With the wider deployment of security controls and tools in large organizations, and the widespread of security breach notification laws all over the world (e.g. European Union Directive on Privacy and Electronic Communications in 2009), many breaches have been reported during the period (e.g. RSA breach in 2011). ISOCs then became aware that intrusions could not be avoided in every case, and make their mission evolve from a full detection and prevention of every event to information exfiltration (e.g. Data Leak Prevention) detection and breach containment. Processes and procedures then changed to concentrate on the most impacting breaches for the organization, timely escalation and remediation, and forensics capabilities for compliance and post-mortem analysis.

- *Fifth generation (2014 onward)*: ISOCs enter in a new era of exponentially growing cyber-attacks. Attackers are now massively collaborating and the attacks representing the higher risk for government, military and private organizations are not necessarily the most visible (i.e. APT). ISOC units must continue to evolve in order not to be out-paced by attackers. A new generation of ISOC must emerge by adopting a holistic approach, combining the broader information from security devices and most advanced SIEM platforms, with the most advanced analytics techniques, and human-driven intelligence shared between cooperating white hats.

Mission and perimeter

With the exponential growing of attacks against which modern organization has to protect, and with the growing number of deployed devices and software related to Information Security in modern ICT infrastructures, an efficient ISOC need a scrupulous planning. The first step must then be the definition of a mission and a perimeter for any ISOC unit:

- *The mission*: it is a key statement that specifies at a very high level the purpose of the ISOC existence. It usually defines one or several clear objectives for the unit (e.g. detect attacks from the Internet, monitor PCI-DSS compliance, respond to detected incident and forensic analysis etc.). This definition should be agreed and confirmed by the organization from which the ISOC depends (i.e. its internal or external customers, and sponsors) and each member of the ISOC.

- *The perimeter*: (technical perimeter, operational hours, customers etc.) In order to accomplish its mission the ISOC unit needs to know precisely what is the perimeter under its responsibility. First, the ISOC need to know what is the technical perimeter (e.g. what ICT infrastructures are under its monitoring, what are the data and security events that will be available to feed its analysis, which part of the infrastructures will be directly managed by the ISOC teams and by others, etc.). Important information to determine is the operational hours. This may vary from a five working days, eight hours a day (i.e. 8/5) to a "follow the sun" service (i.e. 7 days a week, 24 hours a day, 365 days a year, 24/7/365).

Base on the definition of the ISOC mission and perimeter, the various services that will be operated by the unit can be derived precisely.

Organization

Once defined, one of the key parameter for the successful running of an ISOC is the organization and decomposition of the responsibilities. In most cases, modern ISOCs adopt a high level distinction between the Management people and the Technical staff.

Among the Management people, an ISOC usually has a Director which is responsible for any decision required to ensure that the global mission of the ISOC is accomplished. As such he is liable for the objectives accomplishment in front of the organization hierarchy (e.g. CSO or CEO). The Director must also guarantee the obligation to provide the means (e.g. funding, technical means etc.) by the

internal or external customers. He is also responsible of the interfacing with the other entities of the organization. Other managers may also be required depending on the ISOC perimeter, the shape of the organization and the customers of the ISOC's services, which may be internal or external customers in the case of a MSSP. Those SOC Managers should be in charge of a part of the perimeter or a part of the customers (e.g. one SOC manager per customer). They are responsible for all aspects that ensure the correct operations for their perimeter, and are the primary point of contact for their customers. They usually report to the Director. Definitely, a Staff Manager may also be required to manage the assignment and planning of technical staff in order to address the ISOC Manager's needs.

Regarding the Technical staff, the major part is composed of Analysts. Nowadays, most deployed and operated ISOCs are of the fourth generation, and tend to adopt an organization which decompose the analysis in three levels. A diagram of how is organized a typical ISOC with three level of analysis can be found in Figure 15:

- *Level One Analysis*: Level One Analysts are the first line of analysis. They are responsible of the detection of the most common security incidents (e.g. already known and documented incidents). They base their analysis on correlation of several security events (e.g. alerts raised by security devices, like IDS/IPS, Authentication failures, etc.) and researches on various sources intelligences about targets, attackers, exploits and threats. Depending on their analysis, they determine the cause of the event (e.g. a real attack, or a misconfiguration of a device etc.), and fix a severity or priority to the event. If the Analyst is unable to determine the cause, or if he assess that the security event requires more advanced investigation or a remediation (e.g. in case of a confirmed intrusion), he may escalate the issue. In some cases, a Level One Analysts may be supervised by a more experimented Level One Analysts.

- *Level Two Analysis*: Level Two Analysts address the detection of more complex events that cannot be assessed by a Level One Analysts. They are usually more experimented, which give them the ability to handle more advanced attack situations. In some cases, Level Two Analysis may be handled by the SOC manager himself, who will be responsible of the reporting to the customer. As for the first level the Analyst will decide if a remediation or escalation is required (e.g. for a more advanced analysis in case a new kind of attack is suspected).

- *Level Three Analysis*: The upper level of analysis make usually use of the CERT (or CSIRT) resources. They are responsible for the most complex attack analysis (e.g. new kind of attack), reporting and remediation. CERT's experts are not managing the most complex analysis in the same timeframe than Level One and Two Analysts. They usually take more time (e.g. up to 90 days after an incident) to perform a deep forensics analysis, and report on an incident.

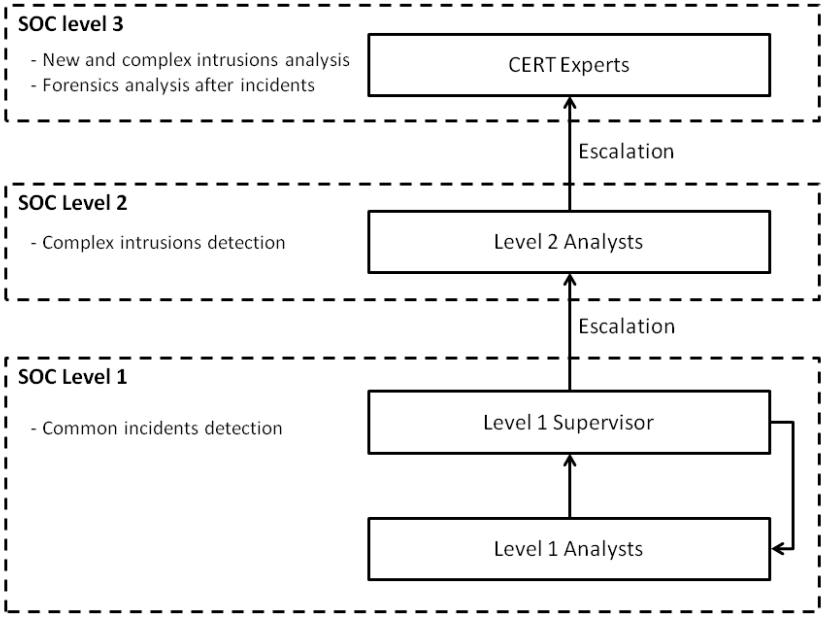


Figure 15 - Typical three level of analysis ISOC process diagram

Apart from the Analysts, the ISOC Technical staff should also be composed of a number of specialists or experts to complement the Analysts skills: A SOC Architects to manage technical evolutions of the ISOC's perimeter (e.g. integration of new customers) and migrations of the technical platforms on which the ISOS's services relies; Administrators, for technical management of systems and networks; Security engineers and administrators to manage the security devices, software and solutions.

Finally, for the remediation after an incident is detected, every kind of the ISOC Technical staff profiles, and possibly external profiles if the remediation actions are not under the responsibility of the ISOC teams (e.g. the NOC, or systems and networks Administrators of the customers itself) should be available.

Deficiencies

Here after are among the main deficiencies of ISOCs that we may identify:

- *Poor correlation automation*: detection is based on correlation mainly done by experts. Actually, ISOCs lack advanced automatic correlation tools based on analytics or data mining algorithms.

- *Incident focus*: Main focuses of ISOCs are on security events and incidents rather than risks. But, the most dangerous attacks are targeted (e.g. APT), may be constituted of different attack steps, and are usually flowed among irrelevant security events or false positive. Moreover, ISOCs lack tools to guide Analysts to detect most complex attacks based on multiple steps and retries.

- *Limited integration with NOC*: Technical perimeter of ISOC is rather narrow and focused on managing security devices to monitor the network infrastructure. But, to assess precisely the threats, ISOC would need to have an up-to-date vision of the entire network infrastructure topology and inventory. Actually, ISOCs usually lack an efficient integration with NOCs and their tools, which would also enable ISOC's experts to deploy remediation to address security incidents timely.

- *Limited business level vision*: ISOCs manage technically ICT infrastructures. But, they usually lack a global vision of the mission or business of the organization. ISOCs security events' analysis are then necessarily low level and cannot efficiently prioritize the most critical incidents for the organization.

• *Poor reactivity*: As ISOCs are mainly based on paper processes and procedures, remediation deployment necessarily take some time and is prone to human errors. But, currently, most ISOCs lack tools for the automation of remediation actions.

4 Visualization and interactions

Visualization can represent synthetically and often aesthetically a significant amount of data and events in order to facilitate the understanding and manipulation. In the security field, visualization has been proposed in order to cope with increasing volumes of data and the major limitations of purely automated analysis. It allows in particular to provide an efficient representation of a general situation, to search for information intuitively or to detect patterns and details inaccessible to the study of raw data.

4.1 State of the art and limitations

In Visualizing Data [1], Ben Fry offers a 7-steps process for visualization:

- *Acquire*: data is retrieved during this stage. In the context of PANOPTESEC, data can be contextual information about the monitored system (the machines, their services running, their address / their name, their location etc.), or information reported by the machines (the system logs of each machine and for each service) or by the security services (the firewalls or the intrusion detection systems).
- *Parse*: information is translated into a manageable format. In the context of PANOPTESEC, it may be for example the unification of all alerts in a common format.
- *Filter*: formatted information is filtered to retain only those that are relevant. In the context of PANOPTESEC, this step corresponds for example to the alert aggregation / correlation steps.
- *Mine*: the stage during which an automatic process allows to find relevant information among those that have been filtered. In the context of PANOPTESEC, this point corresponds to high-level aggregation / correlation of alerts.
- *Represent*: a visual model is chosen to render the data in graphical form. It is the heart of the visualization process but only a part.
- *Refine*: the representation is improved to make it more synthetic or aesthetic (choice of color codes, icons, artifacts, for example).
- *Interact*: the user changes the representation by interacting with the visualization tool to perform actions depending on what he or she sees on the initial view.

These seven steps are involved in the visualization process but the steps of representation, refining and interaction form the core of it. Moreover, even if the steps in visualization are strictly ordered, some retroactions are present. For instance, following a representation, the user may wish to obtain additional information or modify the filter rules on data. Besides, during the interaction stage, the user may have retroactive effect on data mining (ask for details on data) or refine the representation (highlight graphical elements for example).

From a security perspective, visualization is a part of a global solution of a monitoring system. It can providing four complementary functions in the security process:

- *Dashboard*: the tool allows real-time monitoring of the protected system. In this context, the visualization tool must allow to collect and display a clear and concise manner information on the current status of the monitored system.
- *Analysis*: the tool allows to visually mine the collected data to better understand the situation, especially in the case of proven security situations (*post-mortem analysis*). In this context, the

visualization tool must allow to query interactively the different data sources to find additional information about the security event studied using a suitable representation.

- *Reaction*: the tool provides the operator with one or more appropriate corrective measures with easy deployment. In this context, the visualization tool must have precise information on the security mechanisms and situation in the monitored system. Moreover, it must often have administrative rights (or a way to acquire them) on the monitored system in order to implement the changes. As such, it is a privileged application that must be protected accordingly.
- Communication: the tool simplifies the explanation of security situations. This communication can be internal allowing to exchange information about processing, analysis and documentation of events (ticketing) between experts. It can also be external and simplify the reporting of security situations to non-experts.

Visualization techniques have been successfully applied to the network monitoring and security in several areas. However, most proposals do not address the security situation awareness problem or address it only in a limited way (e.g., do not address proactive and reactive chains, do not involve risk analysis results or other security metrics, and do not involve a response model).

In the following we describe several research proposal and visualization tools, from both academia and industry that share some of the goals of PANOPTESEC.

In [145], Yurcik proposes a flow visualization tool that allows operators to detect and investigate anomalous internal and external network traffic. It mainly uses parallel coordinates visualization in which nodes are represented on three axes. It can be animated over time to reveal some trends in specific interval times. It improves situational awareness of a security administrator by providing an intuitive view of IP flows using link analysis. However, this approach is limited to an analysis of traffic on the network, without considering a broader approach to encompass other aspects like risks, attack and response model.

In [146], Conti et al. explores the application of visualization techniques to aid in the analysis of malicious and non-malicious binary objects, in particular of network packets. However, the approach used is heavily focused on packet structure and low-level analysis (payload analysis) and does not encompass topics regarding network security, network topology and risk/attack models.

In [147], Takeda and Koike elaborates on the inspection of computer log-files, using two techniques for manual log inspection: information visualization and statistical analysis. It is loosely tied to cyber security issues, and it does not cope with representation of an ongoing situation, limiting its approach to "a posteriori" analysis. However, the same authors describes in [148] a method to detect anomalous user activities, such as intrusion, from a huge amount of computer logs. Exploiting the theory described in the paper, a real system has been developed. It is a tool for monitoring and auditing user behaviors on a server used by a small group of people. It focuses on the activities of accessing to a server, logging to a server, and substituting a user with another one. Using a concentric-disk representation of the operations, the operator can analyze the 3 types of log and evaluate risks for the system. To that end, the authors use layered concentric disks representations, log summarization and rules specification: each disk can belong to a different rule, to distinguish particular risk, e.g., access to the server from abroad.

In [149], Koike and Ohno focused on Snort logs and proposed a visualization that uses a time-diagram. Each attack is displayed as an icon whose shape and color indicate the type and priority of the attack. The system overlays a histogram of the attack on each icon, helping to avoid overwhelming the screen with large numbers of identical icons. Abdullah et al. also addressed the issue of attacks overwhelming in [150]. They propose a novel visualization showing alarm activity within a network. Despite being a tool for analyzing alarm activity in real-time and aiding the security operator in making decision, this proposal does not consider risk assessment and lacks the concepts of pro-active and reactive phases. STARMINE [151] addresses the same problem by integrating three different

kinds of representations: logical, geometrical, and temporal. [152] goes a step further and provides a holistic view of network security to help detect malicious activities, and presents critical information concerning network activity in an integrated manner.

Aside these proposals that are dedicated to IT systems, [153] presents a Cyber Situational Awareness Visualization tool dedicated to smart grids systems that visualizes cyber trust, evaluated in terms of a mathematical model consisting of availability, detection, and false alarm trust values as well as a model of predictability. It provides visualization of trust in the nodes of a network in order to allow a user to make critical decisions. To that end, it uses bar charts to show the level of a variable measured from the sensors (i.e. Voltage), geographical map: to show the nodes and the level of alerts in each of them and line charts: to model the attacks with respect to the effects that these had on the nodes.

[154] is another proposal that shares some objectives with PANOPTESEC. Indeed, it provides visualization technique that allows for reviewing the state of the network, locating potentially problematic anomalies and prioritizing the anomalies for detailed analysis and remediation. Its goals are to provide support for situational awareness (high-level understanding of available data-network health) and to help decision making. To that end, it integrates and abstract various data sources.

By contrast, [155] focuses on a single data-source (Nessus vulnerability detection results) that has already been abstracted. To that end, it uses Web-based treemaps and linked histograms to discover and manage vulnerabilities on the networks. Treemaps are zommable and the analyst can obtain on demand detailed information about results of the Nessus analysis.

From a real-time monitoring perspective, [156] is probably one of the most impressive proposal both from the perspective of the size of the data it can handle and from an aesthetic point of view: Inoue et al. present a novel real-time 3D visualization engine that enables operators to grasp visually and in real-time a complete overview of alert circumstances and provides highly flexible and tangible interactivity. This alert system is based on large-scale darknet monitoring, where the darknet is a set of unused IP addresses whose monitoring is an effective way of detecting malicious activities on the web. The GUI of the tool is fully based on the 3D visualization of the darknet and its properties.

Aside these research results, VIAssist is probably the most advanced product in terms of visualization of security-related data. It offers a pseudo-realtime network data analyzer that assists that analyst that monitors the network, and allows to analyze and react to some suspicious or malicious activities. It proposes bar charts for data analysis, graphs to analyze links among nodes, pie charts to better understand the communications started/received by a specific node and offers a user friendly environment for issuing queries.

In general, visual analytics techniques and technologies are mature. It is currently quite easy to select proper representations for a given single dataset [157]. However, visualization has currently only partially addressed when multiple data sources are used in a fully integrated way. For instance, current network management products that include visual analytics technique or technologies often lack integration with security relevant data and current security management products neglect attack, risk and response models, decision response support, and correlation of security data between proactive and reactive chains.

For the specific case of ACEA, the currently adopted security system basically relies on the RSA Envision (storing of information from 1 year) that integrates different products' logs:

- FW logs (Juniper, Cisco, Fortinet)
- IDS/IPS Logs (HP Tipping Point, Radware)
- Anti-virus logs (Symantec End Point Protection)
- VPN (SSL) Concentrator logs (Juniper Client to LAN, Cisco LAN to LAN)
- Active Directory (administrative events)
- Network appliance logs (Cisco)

- All administration logs for above devices
- Anti-Spam (Symantec) connected to exchange
- Proxy web filtering logs (Websense) (blocking URLs and cross site scripting, etc. But not HTTPS scanning – due to personal privacy and labor laws)

From a preliminary analysis we found the following limitations:

- The number of licenses for RSA is at the limit;
- It is not clear what each visualization represents, what information is collected and how it is used;
- It is not possible to use all data, but only to monitor a subset of the system: some more general data may be lost and different perspective and dynamic adaptation of data could be helpful.
- There is no generic framework for representing data: pre-processing based on the sensor may not capture all the relevant data.

Tool	Attack model	Situation	Effects of attack	Risks	Automatic solutions	Semi-automatic solutions	Logs analysis
VIAssist	X	√	X	√	X	X	√
CyberSAVE	√	√	√	√	X	X	X
VisFlow Connect-IP	X	X	X	X	X	X	√
Tudumi	X	X	X	√	X	X	√

Comparisons between the various tools in the context of Visualization

In this matrix, we compare the most significant products, VIAssist, CyberSAVE, VisFlowConnect-IP, and Tudumi against the Panoptesec main characteristics (Attack model, Situation, Effect of the attack, Risks, Automatic solutions, Semiautomatic solutions, Logs analysis). It is clear that some relevant issues (e.g., automatic and semi-automatic solutions) are not addressed by current proposals.

4.2 Objectives of the PANOPTESEC project

The PANOPTES project will build on such an architecture, leveraging the work in visual analytics to adopt the use of an advanced visual analytics engine, producing a fully integrated visual analytics environment, in which the latest developments in network data visualization will be combined with models contributing to security analysis (mission impact model, attack model, risk model, response

model), correlation between the actual network and the models, and operator interaction as necessary to support automatic and semi-automatic responses.

Current challenges and objectives of the PANOPTESEC project

Label	Importance	Difficulty
Representing abstracted data from multiple sources has currently be only partially addressed. One of the objectives of the PANOPTESEC project will be to propose adequate coordinated representations for the various data that will have been collected, correlated and abstracted by the other components of the system.	3	3
Reaction has currently only been very partially addressed in the current proposals. The PANOPTESEC will propose some representations to help the analyst in understanding the situation, the consequences of the reconfigurations and will help him or her in deploying them.	3	3

5 Integration framework technologies

5.1 Technology Platform

5.1.1 Approach

This section addresses the key technological aspects behind the design and engineering of PANOPTESEC system as a whole. While primary objective of PANOPTESEC project is evolution in the state-of-the art in security, a key challenge is to integrate them together in a robust and secure manner

The technology assessment approach being adopted for the selection of suitable technologies to be used within PANOPTES considers a number of steps:

1 Identify the technology domains which are considered applicable to PANOPTESEC.

2 Identify the criteria against which candidate technologies within a technology domain shall be assessed.

3 Perform an assessment of candidate technologies and associated products, including the overall best integrated selection of candidates.

4 Based on the previous points, define a technology assessment roadmap, identifying where further analysis and investigations are required before the final selection of the PANOPTESEC technologies can be completed.

5.1.2 Initial Assumptions

There are some basic assumption and constraints which shall scope the technology selection.

- **Operating systems**: Linux is the selected baseline Operating System for PANOPTESEC on the basis of its open nature, for the processing functions. Compatibility with Windows is highly desirable for the execution environment front-end (user interface).

- **Programming language:** Java is the preferred language for all components. This implies that all technology candidates have to offer a native interface towards Java based implementations. Note that this assumption does not rule out the capability to develop and/or integrate

components implemented in a different language, for whatever reason (e.g. performance hot-spots, integration of legacy implementations, etc.). Further, it does not represent a constraint on the implementation of the selected products themselves (e.g. a product can be implemented in C++, provided that it supports Java based implementations in its interfaces).

5.1.3 Technology Selection Criteria

The technologies' selection criteria model is based on the FURPS+ model developed at Hewlett-Packard. The FURPS categories of the basic model cover the following aspects:

- Functionality - Feature set, Capabilities, Generality.
- Usability - Human factors, Aesthetics, Consistency, Documentation
- Reliability - Frequency/severity of failure, Recoverability, Predictability, Accuracy, Mean time to failure
- Performance - Speed, Efficiency, Resource consumption, Throughput, Response time
- Supportability - Testability, Extensibility, Adaptability, Maintainability, Compatibility, Configurability -Serviceability, Installability, Localizability, Portability

The + was added to the model acronym in order to emphasize various attributes:

- Design requirements
- Implementation requirements
- Interface requirements
- Physical requirements

The PANOPTESEC specific concerns lead to add the following criteria:

- Costs – Implementation costs, Possession costs, Configuration costs, Adaptations costs.
- Security - Authentication, authorization, encryption
- Marketability - Licensing, Property rights, Export rights

5.1.3.1 Functionality

Functional requirements are also supported by non-functional requirements, which impose constraints on the design and implementation. This criterion addresses the capacity of the candidate technology to fulfil the level of quality of the functional requirements.

For example, this criterion shall assess the system capacity to perform its mission in due time, managing the routine dataflow and potential peaks.

5.1.3.2 Usability

Usability qualifies to which extent the system can be easily operated and in particular "learnability" and the "user friendliness" from the end user point of view. Thus, it addresses human factors, aesthetics and consistency in the user interface and documentation. It may encompass the following aspects:

- Consistent and integrated Look & Feel,
- Support users with different levels of expertise,
- Customizable for different projects/missions,
- Efficient and robust,

- Easy to learn (including training aspects),
- Good level of the available documentation.

This criterion is a concern for PANOPTESEC as it has a direct impact on the operational costs (i.e. minimizing the number of operators).

5.1.3.3 Reliability

This criterion assesses the ability the candidate technology to support the system in performing its required functions under stated conditions for a specified period of time.

This criterion encompasses the following aspects:

- Availability (the amount of system "up time"), amount of time between failures or the ratio of time when the system is operational and accessible when required for use
- Ability to recover from failure without further impact,
- Accuracy and exactitude of system calculations
- Integrity and consistency (no data loss neither corrupted)
- However, the maintainability is also an important part of reliability engineering, as the latest may affect at least the two fist listed items.

5.1.3.4 Performance

This criterion qualifies the capacity of the candidate technology to support PANOPTESEC in fulfilling its required functions within a given set of constraints. For example this may comprise:

- Response time for a given function (e.g. start-up and shutdown time),
- Duration of the completion time for a given function,
- Capacity; like throughput (rate of processing work), data bandwidth / transmission time, number of connected facilities,
- Optimization of computing resource(s),
- High availability of the system (e.g. recovery time).

In particular, a high level of performance is required for the PANOPTESEC system to meet the stringent requirements relative to sensor data processing. This criterion shall assess the degree to which the system is able to perform its mission in due time, managing the routine dataflow and potential peaks.

5.1.3.5 Supportability

This criterion addresses the various elements which contributes to easily supporting PANOPTESEC during its lifetime:

- Testability: High degree of test automation,
- Extensibility: New components expecting during long life time and to build end users' applications,
- Portability: Easiness with which the system can be transferred from one hardware or software environment to another. Needs to support mainstream OS and different OS version over product life time,

- Adaptability: Facilitate tailoring for different projects and system applications, open interfaces (interoperability with other systems). This also qualifies the capacity of the solution to adapt to potential internal or external changes, in a timely and cost-effective manner
- Maintainability: OSS approach, proven and main stream technologies. This criteria addresses the easiness with which the system can be modified/upgraded to correct faults, improve performance or other attributes, or adapt to a changed environment
- Compatibility: Avoid niche or single-vendor solutions, use Commodity PC/workstation hardware, reduce OS dependencies,
- Configurability: Where needed but no more to keep things simple and efficient,
- Installability: Automated where possible, follow platform standards, remote deployment,
- Scalability: Very broad range from single laptop to control centre. This also addresses the ability of the system to improve its performance after adding hardware, proportionally to the capacity added.
- Community: Qualifies the development and users profiles of the candidate solution.

5.1.3.6 The Model +

- Design requirement: Specifies or constrains the options for designing a system. For example, a relational database may be explicitly required.
- Implementation requirement: Specifies or constrains the coding or construction of a system. Examples include required standards, implementation languages, operating systems and resource limits.
- Interface requirement: Specifies an external item with which a system must interact, or constraints on formats or other factors used within such an interaction.
- Physical requirement: Specifies a physical constraint imposed on the hardware used to house the system — shape, size, or weight, for example.

5.1.3.7 Costs

This specific criterion is reflecting the qualitative cost assessment of the system implementation and deployment. It includes the following items:

- Implementation costs: Assess the impact of the candidate solution on the development costs (i.e. licenses, schedule and risks - Maturity).
- Possession costs: Includes both costs to own (license fee) and to use (maintenance fee) the selected solution,
- Configuration costs: Addresses the complexity to parameterize the selected technology for each application (end user system),
- Adaptations costs: Concerns the potential cost impact on elements/component added to PANOPTESEC in order to build the end user system.

5.1.3.8 Marketability

This specific criterion includes the following items, on the basis the PANOTESEC will be marketed as a commercial product.

- Licensing policy: Licensing model and price policy applied to the candidate solution..
- IPR: This element qualifies the legal aspect attached to the use of the candidate solution and its impact on PANOPTESEC business cases.

- Export rights: Identification of export restrictions and procedures. Assessment of the compliance towards PANOPTESEC driving constraints (e.g. no export restriction)

The candidate technology shall ideally have at-least two viable open source implementations. Only open source license can be considered which do not contaminate PANOPTESEC software itself with the same license conditions (i.e. viral). This does not exclude the use of licenses, such as GPL, but only when it does not contaminate PANOPTESEC software which will have its own licensing conditions. Commercial products may be considered, but the selection must be fully justified (e.g. is the only product that could fully meet the requirements) and only if it is possible to easily replace the product with an alternative (perhaps with some loss of capability).

5.1.3.9 Security

This criterion assesses the degree to which the system is able to protect its data and restrict its access to authorize users. It includes the following items:

- Authentication: Guarantees the origin of the incoming entities (e.g. configuration data, commands, service requests…),
- Authorization: This element is particularly addressing the capacity to filter user access to elementary functions. The selected technologies shall not put any restriction on this capability,
- Encryption: read protection of data (stored, exchanged).

This aspect shall be checked very soon as inappropriate solutions may jeopardize the capability to reach the expected level of security for the final deployments.

Additionally, the final selection shall take into account the compatibility between the different technology domains, to achieve the best global selection, rather than just the best candidate within each individual technology domain

5.2 Technology Domains

In this section we identify the technology domains and the criteria against which each technology candidate within a technology domain are analyzed for their suitability, Technology domain is basically an area where adoption of 3rd party technologies is considered potentially effective. For example, component framework is a technology domain which we want to study for which we will select a number of different candidate technologies (SCA, OSG, etc.).

5.2.1 Run-Time Environment

In following sections provide a preliminary list of technology domains. This is only a first iteration and is expected to evolve in future versions of this document.

5.2.1.1 Component Framework

Overview

The component framework supports the development, composition, deployment and execution of the PANOPTESEC software components, which form the building blocks of an system.

Selection Justification

Implementation of the component framework identified in the system concept and functional requirements

Related Technology Domains

Service Integration Platform, Security, Data Persistence

5.2.1.2 Service Integration Platform

Overview

The service integration platform supports the integration of a complete system and its integration with external systems using a service orientated approach.

The service integration platform provides the basis for the top level integration of a complete PANOPTESEC system. It includes various integration features, such as flow control, message transformation, etc. A critical feature is performance (data throughput and latencies) as there is an overhead going through the message bus itself and also the need for transformation logic to be performed.

With regards to integration of the service integration platform shall:

- be decoupled from the selected component framework to enable easier evolution of the two products
- provide good integration of services of provides / consumers running within the component framework
- Be open to the integration of different communication protocols

Selection Justification

Integration features, such as flow control, message transformation, etc, are complex to implement. The selection of a proven third party product is therefore considered the most appropriate approach.

Related Technology Domains

Component Framework, Messaging, Security

5.2.1.3 Communication and Data Distribution

Overview

Support for the efficient distribution of messages and data within PANOPTESEC, including the marshalling and serialization of data.

Message-oriented middleware (MOM) is software or hardware infrastructure supporting sending and receiving messages between distributed systems. MOM allows application modules to be distributed over heterogeneous platforms and reduces the complexity of developing applications that span multiple operating systems and network protocols. The middleware creates a distributed communications layer that insulates the application developer from the details of the various operating system and network interfaces. APIs that extend across diverse platforms and networks are typically provided by MOM

In addition, many inter-application communications have an intrinsically synchronous aspect, with the sender specifically wanting to wait for a reply to a message before continuing (see real-time computing and near-real-time for extreme cases). Because message-based communication inherently functions asynchronously, it may not fit well in such situations. That said, most MOM systems have facilities to group a request and a response as a single pseudo-synchronous transaction.

Selection Justification

Message-oriented middleware applies to PANOPTESEC because it enables the capability to let components/subsystems communicate with each other in a decoupled way. Because the message is self-contained, it does not have explicit dependencies on the components/subsystems that handle and

pass around messages. Those components/subsystems can evolve independently. Messaging is stateless and when applied correctly, allows certain parts of PANOPTESEC to become statelessness as well.

Related Technology Domains

Service Integration Platform

5.2.1.4 System Run-time Management

Overview

Support a common approach for the management and administration of PANOPTESEC run-time system and legacy platforms. The purpose is to provide a harmonized approach to run-time management of all the systems that comprise PANOPTESEC.

Selection Justification

The System Run-Time Management component is a key aspect in future commercialization of PANOPTESEC system, for flexible deployment on client networks.

Related Technology Domains

Component Framework, Service Integration Platform

5.2.1.5 Logging and Tracing

Overview

Logging and tracing of events and messages within PANOPTESEC.

In the System Context document in the context of messaging. Although messaging and logging are connected, we address them separately with regards to technology assessment and describe the relationship to both messaging and data archiving in general.

A number of logging frameworks exist and are candidates for selection for the logging solution within PANOPTESEC. Many of these frameworks breakdown logging into three logical components (see Wikipedia):

- Logger – this is responsible for capturing the message to be logged, including any associated metadata and passing it to the logging framework. The logger provides the interface with PANOPTESEC components which are the source of log messages.
- Formatter – the logging framework uses a format to take the log message (including meta-data) and formats its output. This enables a clean separation between the context of a log message and how the message is presented in its output form. This allows the format to be easily changed by either replacing the formatter or by changing the configuration settings of an existing formatter. Different formatters could also be used depending of the message type.
- Handler – the logging framework passes the formatted message a handler for disposition. The handler could be responsible for sending the formatted to a display, writing it to a file / database for storage, etc. The selected handler will depend on the message (e.g. message type) and more than one handler can be invoked for a given message. For example, it could be that the log framework is configured that trace messages are only processed (for performance reasons) by the local log manager.

Related Technology Domains

Communication and Data Distribution

5.2.1.6 Data Persistence

Overview

Covers the framework used for the persistence of data within the preparation and run-time environment. This covers data definitions, software configuration data, software component state, etc.

Selection Justification

Data persistence attributes the reliability of parts of the system. An increased reliability follows from the fact that state can be restored even after a restart of a component/program/subsystem/system.

Which objects are to be managed persistently needs to be determined. It is already safe to state that configuration data objects for example are good candidates for this.

Related Technology Domains

Data Modelling and Editing

5.2.1.7 Data Archiving

Overview

The efficient storage and retrieval of time ordered data received and generated at run-time. Performance is a key driver and where the complete history of all samples is stored.

5.2.2 Software Development Environment

This section identifier the technology domains for the PANOPTESEC software development environment based in the following assumptions:

- Distributed Development Teams
- Establishment of Common Software Development Environment used across all teams working on PANOPTESEC development

5.2.2.1 Requirements Management

Support for the management of requirements throughout the project life-cycle.

5.2.2.2 Design Methodology and Tooling

Design language and the supporting tools used for the design and the generation of documentation and software artefacts.

5.2.2.3 Integrated Development Environment

Provides comprehensive facilities for software development, such as source code editor, compiler, debugger and which supports the integration of other facilities into a single environment.

5.2.2.4 Configuration Management

Supports the storage, configuration and change control of PANOPTESEC software and documentation.

5.2.2.5 Build System

The building of deployable PANOPTESEC system artefacts from PANOPTESEC sources (e.g. generation of binaries from source code).

5.2.2.6 Installation and Deployment Management

Support for the installation and deployment of PANOPTESEC into the target run-time environment.

5.2.2.7 Testing Framework

Supporting test framework(s) used for the verification and validation of PANOPTESEC system at various levels (i.e. unit, integration and system)

5.2.2.8 Software Quality Measurements

Support the production of PANOPTESEC software quality metrics (e.g. compliance against coding standards, requirements coverage, code coverage, performance, etc.).

5.2.2.9 Continuous Integration

Implement the continuous processes of applying quality control within PANOPTESEC development (i.e. small pieces of effort, applied frequently).

5.2.2.10 Issue Tracking and Reporting

The tracking and reporting of issues during PANOPTESEC development and maintenance.

5.2.2.11 Code Reviews

Support for code reviews within a distributed environment.

5.2.2.12 Document Authoring

Support for technical documentation production (e.g. User Manuals, Training Material, etc) so that document content can be created in a presentation-neutral form and published in a variety of formats, such as HTML or PDF.

5.3 Technology Selection

5.3.1 Component Framework

OSGi is presented as the baseline component framework as it is a mature technology with a high level of adoption in middleware implementations. Other advantages include:

- A number of high quality open source implementations are available, including those with liberal licenses (e.g. Felix, Equinox, etc).
- Good support for modularizations and plug ability, where variations and extensions of a core system can easily be accommodated.
- Remote services specification supports the externalization of services outside of the JVM.
- Blueprint Container specification provides a dependency injection framework for OSGi to support application level development (based on Spring Dynamic Modules).

EJB (JEE) are very mature with a large user community and dominate within the Enterprise Application domain. EJB is however considered as being rather heavyweight within the JEE technology stack.

Spring is also widely used but recommended for exclusion based on it not being standard based.

SCA is considered a good technology that can theoretically fulfil component framework needs. However the adoption level, availability of good open source implementations and the activity of the user community has been lower than initially expected and doubts remain over the long term relevance of SCA within the market.

The component framework supports the development, composition, deployment and execution of PANOPTESEC software components, which form the building blocks of an PANOPTESEC system. This domain covers:

- The development by providing an IDE to build and assemble components ;
- The deployment: components and groups of components, named composites, are deployed among several distributed containers. A deployment is described in a deployment plan ;

The execution: The containers host the components and execute them. They provide adequate bindings for communication between distributed components.

Distributed PANOPTESEC components communicate through a platform that a message based communication. As PANOPTESEC is an oriented service platform, the service contract shall be built upon this low message layer. Applications should not see messages but only calls to operations of services. Messages are just a transport. Endpoints must be also configurable so that they can be easily adapted to local and distributed deployments of a composite. Switching between in-process and remote communication shall be easy and transparent for components. As a result, how the components communicate does not concern only the domain "Communication and Data Distribution" but also the Component Framework in terms of services interfaces and bindings.

Securing OSGi applications tends to imply the use of Java SE security mechanisms including JAAS. As PANOPTESEC is itself a security-centric application, it implies that components must run as authenticated users, which implies a JAAS Login at composite component startup. Calls to encapsulated implementation components are to be made within a JAAS Security Context, providing controlled access to resources within the OSGi runtime. The component binding layer responsible for message encryption, validation between the composite components.

5.3.1.1 Technology Candidates

The following candidates have been identified, avoiding commercial products.

Sub Domain	Product	Licence	Analysis
Component Container	Apache Felix	EPL	Apache Felix is a pure Java OSGi container implementing OSGi R4. In the Apache world, it is the base for Karaf, ServiceMix and Fuse which is a good proof of quality. Apache Felix is a pure Java OSGi container implementing OSGi R4. Its packaging is very clear, its configuration also. It offers a web console to inspect the bundles deployed and their states. Apache Felix publishes maven plugins to integrate the development of OSGi bundles within the Maven generation. The Apache Felix subprojects are many. The community is active
	Eclipse Equinox	EPL	Equinox, like Felix, is a pure Java OSGi container implementing OSGi R4. It is used by the Eclipse platform, it is important to note that Equinox offers non standards mechanism around OSGi. Eclipse Equinox is the OSGi container (implementing OSGi R4) of Eclipse and so of all Eclipse RCP applications. It may be an advantage to have the

Sub Domain	Product	Licence	Analysis
			same container for applications with human interfaces and applications without human interfaces. However Equinox remains more complex than other containers as it uses a native launcher. Equinox has a complex configuration composed of several files with the same information, sometimes for Eclipse, sometimes for OSGi land. Equinox is full compliant with OSGi specifications but it adds some Eclipse features, which can be tricky when building pure OSGi modules.
	JBoss OSGi	LGPL	JBoss OSGi is included in the widely used JBoss Application Server. JBoss OSGi was originally created to be used in JBoss Application Server and is generally used through the application server. The standalone version is not widely used. The product is well documented. JBoss OSGi 1.1.1 has been released in June 2012. It is a very young product compared to Felix or Equinox.
	JBoss AS		
	Apache Geronimo		
	Glassfish		
	JOnAS		
	CIAO		
	OpenFusion		
	OpenCCM		
IoC Container	BluePrint		The OSGi standard includes an IoC container specification, which is BluePrint. BluePrint IoC is available on both Gemini Enterprise Bundles and in Karaf.
	Spring DM		SpringDM is generally used when Spring framework is also used. SpringDM was contributed to the Eclipse project, becoming Gemini Dynamic Modules. Given the end-of-lining of Spring DM, and the existence of the Blueprint standard for OSGi, it is recommended to discard Spring DM.

Sub Domain	Product	Licence	Analysis
	Google Juice		Google Juice is an invasive injection framework as it requires a specific design. Given the existence of the Blueprint IOC standard for OSGi, it is recommended to discard Google Juice

5.3.1.2 Technology Selection

The selection has been reduced to Felix or Equinox for the component container, whereas for the IoC Container sub-domain BluePrint is clearly recommended as it is the standard proposed by OSGi.

5.3.2 Service Integration Platform

5.3.2.1 Introduction

The Service Integration Platform will support the integration of the complete PANOPTESEC based system, including the integration with external systems based on a service-oriented approach.

It provides the top level integration of the complete PANOPTESEC system and must offer various features such as message routing, message transformation etc. Key drivers of the Service Integration Platform will be performance (data throughput and latencies) and transformation logic. Given the discarding of the Distributed Composition Component sub domain, those aspects will be recommended to be treated by the domain.

Moreover, the Service Integration Platform must:

- Be decoupled from the Component Framework to allow the two domains to evolve independently.
- Provide good integration for service providers / consumers exposed by the Component Framework.
- Support different communication protocols.
- Support distributed component composition.

The main requirement areas covered by the Service Integration Platform are:

- System deployment with regards to remote clients, component distribution and scalability.
- Performance.
- Monitoring and Control Model with regards to the access layer support.
- Service Provision and Integration also covering the integration of legacy systems.
- Security

The Service Integration Platform is closely related to Component Framework, the Communication and Data Distribution and the Security and must be compatible with each. Moreover, the Service Integration Platform should allow several communication and data distribution solution. It is important to verify that the candidates integrate seamlessly.

The Service Integration Platform should be based on a standardized security-model to authorize, authenticate and audit use of the platform. This will reduce the coupling with the Security domain. However, the security candidates will have to be validated against the Service Integration Platform, also because the security constraints may affect the performance of the platform.

5.3.2.2 Sub Domain Breakdown

To evaluate fully a further breakdown of the domain is required.

- **Enterprise Server Bus**. The ESB provides Invocation, Routing, Mediation, Messaging, Transformation and Integration of Services and Data Flows, either amongst themselves and with external applications and clients. Typically an ESB uses a common data serialization format allowing the integration between endpoints that may internally speak a different language.
- **Enterprise Integration Patterns.** The EIP technology offers the Routing, Mediation and Transformation functionality of an ESB. These patterns can be seen as accepted solutions to recurring problems within a given context. They provide a framework for designing, transforming and routing messages. The frameworks allow to integrate different applications using the required patterns. A varity of technologies and communication protocols are supported such as HTTP, FTP, JMS, EJB, JPA, RMI, JMS, JMX, LDAP, Netty etc. They can be deployed not necessarily in an ESB but as standalone application, in a web container (e.g. Tomcat or Jetty), in a JEE application Server (e.g. JBoss AS or WebSphere AS), in an OSGi environment or in combination with a Spring container. The ESB relies on an EIP implementation for the message routing and the mediation engine. Therefore the two domains are closely connected.
- **Service Remoting.** Service Remoting provides the capability of Components to make invocations on services residing in a separate process. Its functionalities are often included in an integrated ESB solution.

5.3.2.3 Technology Candidates

Sub Domain	Product	Licence	Detailed analysis
ESB	Apache ServiceMix 4	Apache	ServiceMix is based on an OSGI container Felix and add best frameworks like spring, camel, cxf, activemq, ode. The OSGI container permits easily to add bundles as hibernate. Usability: ServiceMix doesn't provide any added service based on standard JMX. Reliability: ServiceMix use ActiveMq that can persist messages. Supportability: Camel is use to implement EIP. The OSGI container Felix allows adding many components into the ESB. Costs: Development and integration are very easy with help of many frameworks: Maven plugin for Felix OSGi bundle, EIP implementation with Camel.
	OpenESB	CDDL	Open ESB is an ESB built with JBI technology. Moreover the product runs in a Java application server. This will strongly increase the integration efforts into the component framework based on OSGi. Given the OSGi-orientation of the platform, it is recommended to discard all JBI oriented ESBs. This will strongly increase the integration efforts into the component

Sub Domain	Product	Licence	Detailed analysis
			framework based on OSGi.
	Mule	CPAL	Mule ESB is a widely used ESB. It implements its own complete EIP framework. The Apache Camel integration support and OSGi deployment are not supported natively.The Mule ESB has a solid track record in organizations that have deployed it into production. MuleSoft also has the highest number of downloads of any open source ESB provider, though it is difficult to measure the real impact of this activity on production deployments. It is being released under CPAL. Mule ESB is based on a proprietary component architecture. Given the OSGi-orientation of PANOPTESEC platform, it is recommended to discard Mule ESB.
	Fuse ESB		
	Jboss ESB		
EIP	Apache Camel		Apache Camel offers many, many components (even more than Mule) for almost every technology you could think of. If there is no component available, you can create your own component very easily starting with a Maven archetype. Camel also provides Spring integration capabilities. Moreover, it offers a XML DSL: Readability is better than Spring Integration and almost identical to Mule. Apache Camel also offers DSLs for Java, Groovy and Scala. The programming DSLs are very easy to read (even in more complex examples). These programming DSLs have better IDE support than XML (code completion, refactoring, etc.). Camel has a connector to integrate some channel spring-integration in camel route if necessary.
	Spring Integration		Spring Integration is based on the well-known Spring project and extends the programming model with integration support. You can use Spring features such as dependency injection, transactions or security as you do in other Spring projects. It is almost no effort to learn Spring Integration if you know Spring itself. Nevertheless, Spring Integration only offers very rudimentary support for technologies – just „basic stuff“ such as File, FTP, JMS, TCP, HTTP or

Sub Domain	Product	Licence	Detailed analysis
			Web Services. Mule and Apache Camel offer many, many further components. Integrations are implemented by writing a lot of XML code (without a real DSL). You can also use Java code and annotations for some stuff, but in the end, you need a lot of XML. The visual designer for Eclipse (called integration graph) is not as good and intuitive as its competitors. The selection Apache Camel for EIP is made over Spring.
Service Remoting	Apache CXF		Apache CXF is an open source services framework. CXF helps you build and develop services using frontend programming APIs, like JAX-WS and JAX-RS. These services can speak a variety of protocols such as SOAP, XML/HTTP, RESTful HTTP, or CORBA and work over a variety of transports such as HTTP, or JMS. CXF includes a broad feature set, but it is primarily focused on the following areas: • Web Services Standards Support: CXF supports a variety of web service standards including SOAP, the WS-I Basic Profile, WSDL, WS-Addressing, WS-Policy, WS-ReliableMessaging, WS-Security, WS-SecurityPolicy, WS-SecureConverstation, and WS-Trust (partial). • Frontends: CXF supports a variety of "frontend" programming models. CXF implements the JAX-WS APIs. CXF JAX-WS support includes some extensions to the standard that make it significantly easier to use, compared to the reference implementation: It will automatically generate code for request and response bean classes, and does not require a WSDL for simple cases. Apache CXF is one of the products integrated in Apache ServiceMix
	Apache Axis2		

Sub Domain	Product	Licence	Detailed analysis
	Metro		
	Fuse Service Framework		
	Eclipse Communication Framework		

6. Conclusion

In this document, we presented the current deficiencies of the various technologies that would be of interest in the PANOPTESEC project. We summarize them and provide directions for the PANOPTESEC project in the following table.

Label	Importance	Difficulty
Collecting and maintaining safely system configuration in the context of SCADA systems is challenging. The PANOPTESEC project should propose some acquisition mechanisms for that purpose.	3	2
Managing the various system configuration description in a conceptually unified way is currently challenging, especially when SCADA systems are involved. This aspect will be handled by the PANOPTESEC project thanks to carefully crafted ontologies.	3	3
It is difficult to be confident about the system information that are collected when at least part of the system can be corrupt. The PANOPTESEC project will address this aspect through information aggregation and correlation.	3	3
New approaches for mission impact modeling need well-analyzed and streamlined data such that a definition of required notions for mission impact modeling can be derived at some suitable level of abstraction automatically (e.g., mission, tasks, services, assets, etc.).	3	1
Current approaches are of a monolithic nature and are not easily transferable to other domains, thus a modular architecture is required such that possibly parts (modules and models) can be reused in other contexts (e.g., a separation between mission impact modeling and risk quantification is to be achieved).	3	2
The state of the art for quantifying either mission impacts or risk estimates mostly uses numerical worst-case scoring values. This leads to a loss of dependency tracking possibilities such that explanations for decision making are hard to provide. The investigation of potential benefits of ontology-based solutions for logical impact modeling, i.e., for techniques beyond simple scoring approaches is a challenge.	3	3
Mission impact information (e.g., priorities on missions) should be used to determining suitable asset rankings based on expected scores to incorporate more information available in a particular situation.	2	3
Business process models are to be investigated in order to possibly improve dependency models (mission, tasks, services, assets), i.e., the benefit of primitive recursion and while constructs from business process models might be exploited for mission impact modeling and risk estimation [61].	1	3

While many vulnerability databases are available, it is still difficult to choose which one proposes the most valuable information. Often, interesting fields are in fact in plain text and cannot be automatically processed. The ontologies proposed in the PANOPTESEC project should help in defining the most valuable information in our context and how to present them.	2	3
Vulnerability databases for SCADA systems should be investigated.	2	2
Collecting and storing securely the outputs of host-based sensors is particularly difficult, particularly when hosts can be compromised. An objective of the PANOPTESEC project will be to design a secure mechanism to collect and store logs from host-based sensors.	2	2
Correlating outputs from different host-based sensors is still an open problem. We expect that results from Semantic science (cf. Section 2.7) will help in this way.	3	3
Capturing and decoding traffic in high-speed network	2	1
Protocol identification	1	2-3
Managing specific and undocumented protocols	3	3
Protection against evasion attacks	2	2
Modeling industrial process for spec-based IDS	3	2
Detecting attacks vs. detecting intrusions	3	3
Misuse-based vs. anomaly-based	3	3
Spec-based and policy-based IDS	3	3
ip/domain black list and reputation systems	3	3
False positive vs. false negative, base-rate fallacy	3	3
Language of signatures, automatic generation of signatures, vulnerability-based signatures	3	3
Incremental learning, feature selection, unbalanced classes.	3	3
Challenges for detection in ICS/SCADA systems	3	3
Representing abstracted data from multiple sources has currently be only partially addressed. One of the objectives of the PANOPTESEC project will be to propose adequate coordinated representations for the various data that will have been collected, correlated and abstracted by the other components of the system.	3	3
Reaction has currently only been very partially addressed in the current proposals. The PANOPTESEC will propose some representations to help the analyst in understanding the situation, the consequences of the reconfigurations and will help him or her in deploying them.	3	3

7. References

[1] The COMIFIN project web site, http://comifin.eu
[2] Esper Project Web Page, http://esper.codehaus.org/
[3] O. Etzion and P. Niblett. Event Processing in Action. Manning Publications Co., Greenwich, CT, USA, 1st edition, 2010.
[4] Farroukh, A.,Sadoghi, M., and Jacobsen, H. Towards Vulnerability-Based Intrusion Detection with Event Processing. In *Proceedings of the 5th ACM International Conference on Distributed Event-Based Systems* (pp. 171-182). ACM. (2011).
[5] Ficco, M., and Luigi, R. A Generic Intrusion Detection and Diagnoser System Based on Complex Event Processing. In *Data Compression, Communications and Processing (CCP)* (pp.275-284) (2011)
[6] Gander, M., Felderer, M., Katt, B., Tolbaru, A., Breu, R., and Moschitti, A. Anomaly Detection in the Cloud: Detecting Security Incidents via Machine Learning. *In Trustworthy Eternal Systems via Evolving Software, Data and Knowledge* (pp. 103-116). Springer Berlin Heidelberg. (2013).
[7] Gedik, B., Andrade, H., Wu, K., Philip S. Y., Myungcheol, D. SPADE: The System S Declarative Stream Processing Engine. *in Proceedings of ACM SIGMOD international conference on Management of data*, 2008
[8] JBoss Drools Fusion web site, http://www.jboss.org/drools/drools-fusion.html
[9] Lodi, G., Aniello, L., Di Luna, G. A., and Baldoni, R. An event-based platform for collaborative threats detection and monitoring. *Information Systems*, 39, 175-195. (2014).
[10] D. C. Luckham. The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001. 35, 58
[11] Mert, A., Ugur, C., Nesime, T. Plan-based Complex Event Detection across Distributed Sources. *in Proc. VLDB Endow.* 1, 1, 66-77 VLDB Endowment (2008)
[12] IBM InfoSphere Streams web site, http://www.ibm.com/software/data/infosphere/streams/
[13] Tibco Business Event Web Site, http://www.tibco.com/products/business-optimization/complex-event-processing/businessevents/default.jsp
[14] Verissimo, P. E., Baldoni, R., Bortnikov, V., Chockler, G., Dekel, E., Laventman, G., Lodi, G., Montanari, L., CoMiFin Architecture and Semantic Rooms. *Collaborative Financial Infrastructure Protection*, 2012: 85-98
[15] Zoumboulakis, M., and Roussos, G., Escalation: Complex event detection in wireless sensor networks. *In Smart Sensing and Context* (pp. 270-285). Springer Berlin Heidelberg. (2007).
[16] Benjamin Morin and Hervé Debar. "Correlation of intrusion symptoms: an application of chronicles." Recent Advances in Intrusion Detection. Springer Berlin Heidelberg, 2003.
[17] Christophe Dousson and Pierre Le Maigat. "Chronicle Recognition Improvement Using Temporal Focusing and Hierarchization." IJCAI. 2007.
[18] Qin, Xinzhou, and Wenke Lee. "Attack plan recognition and prediction using causal networks." Computer Security Applications Conference, 2004. 20th Annual. IEEE, 2004.
[19] Demolombe, Robert, and Ana Mara Otermin Fernandez. "Intention recognition in the situation calculus and probability theory frameworks." Computational Logic in Multi-Agent Systems. Springer Berlin Heidelberg, 2006. 358-372.
[20] Zhu, Bin. Alert correlation for extracting attack strategies. University of New Brunswick (Canada), 2005.
[21] Stakhanova, Natalia, Samik Basu, and Johnny Wong. "A taxonomy of intrusion response systems." International Journal of Information and Computer Security 1.1 (2007): 169-184.

[22]Debra Anderson, Teresa F. Lunt, Harold S. Javitz, Ann Tamaru, and Alfonso Valdes. Detecting unusual program behavior using the statistical components of nides. Technical Report SRI-CSL-95-06, SRI International, Computer Science Laboratory, May 1995.

[23]Mansoor Alicherry, M. Muthuprasanna, and Vijay Kumar. High speed pattern matching for network ids/ips. In Proceedings of the Proceedings of the 2006 IEEE International Conference on Network Protocols, ICNP '06, pages 187–196, Washington, DC, USA, 2006. IEEE Computer Society.

[24]Lothar Braun, Alexander Didebulidze, Nils Kammenhuber, and Georg Carle. Comparing and improving current packet capturing solutions based on commodity hardware. In Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, IMC '10, pages 206–217, New York, NY, USA, 2010. ACM.

[25]David Brumley, James Newsome, Dawn Song, Hao Wang, and Somesh Jha. Towards automatic generation of vulnerability-based signatures. In SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy, pages 2–16. IEEE Computer Society.

[26]Scott A. Crosby and Dan S. Wallach. Denial of service via algorithmic complexity attacks. In SSYM'03: Proceedings of the 12th conference on USENIX Security Symposium, pages 3–3. USENIX Association.

[27]Hervé Debar, Marc Dacier, and Andreas Wespi. Towards a taxonomy of intrusion-detection systems. Computer Networks, 1999.

[28]Holger Dreger, Anja Feldmann, Michael Mai, Vern Paxson, and Robin Sommer. Dynamic application-layer protocol analysis for network intrusion detection. In Proceedings of the 15th conference on USENIX Security Symposium - Volume 15, USENIX-SS'06, Berkeley, CA, USA, 2006. USENIX Association.

[29]Terrie Diaz. Common criteria evaluation and validation scheme validation report, tippingpoint intrusion protection system (ips) e-series (5000e, 2400e, 1200e, 600e, 210e), software version: 2.5.3.6933. Technical report, Science Applications International Corporation, 2008.

[30]Steven T. Eckmann, Giovanni Vigna, and Richard A. Kemmerer. Statl: an attack language for state-based intrusion detection. Journal of Computer Security, 10(1-2):71–103, 2002.

[31]Francesco Fusco and Luca Deri. High speed network traffic analysis with commodity multi-core systems. In Proceedings of the 10th annual conference on Internet measurement, IMC'10, pages218–224, New York, NY, USA, 2010. ACM.

[32]Joseph Gasparakis and Peter P. Waskiewicz. Design considerations for efficient network applications with intel® multi-core processor-based systems on linux. Technical report, Intel, July 2010.

[33]L. Todd Heberlein, Gihan Dias, Karl N. Levitt, Biswanath Mukher-jee, Jeff Wood, and David Wolber. A network security monitor. In Proceedings of the 1990 IEEE Computer Society Symposium on Research in Security and Privacy, pages 296–304.

[34]HP. Hp s intrusion prevention system (ips) n series, 2011.

[35]Mark Handley, Vern Paxson, and Christian Kreibich. Network intrusion detection: evasion, traffic normalization, and end-to-end protocol semantics. In Proceedings of the 10th conference on USENIX Security Symposium - Volume 10, pages 9–9. USENIX Association.

[36]Mike Hall and Kevin Wiley. Capacity verification for high speed network intrusion detection systems. In Andreas Wespi, Giovanni Vigna, and Luca Deri, editors, Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID'2002), volume 2516 of Lecture Notes in Computer Science, pages 239–251. Springer.

[37]IBM. Opensignature user guidelines, 2007.

[38]Christopher Kruegel, William Robertson, and Giovanni Vigna. Using alert verification to identify successful intrusion attempts. Practice in Information Processing and Communication (PIK), 27(4), 2004.

[39]Vinod Ganapathy, Liu Yang, Rezwana Karim and Randy Smith. Improving NFA-based signature matching using ordered binary decision diagrams. In Proceedings of the 13th International Symposium on Recent Advances in Intrusion Detection (RAID 2010).
[40]McAfee. Mcafee network security platform: The next-generation network IPS, 2012.
[41]Vern Paxson. Bro: A system for detecting network intruders in real-time. In Proc. of the 7th Usenix Security Symposium, pages 31–51.
[42]Jean-Philippe Pouzol and Mireille Ducassé. Formal specification of intrusion signatures and detection rules. In Proceedings of the 15th IEEE Computer Security Foundations Workshop (CSFW'02), pages 64–76. IEEE Computer Society.
[43]Jean-Philippe Pouzol and Mireille Ducassé. From declarative signatures to misuse ids. In W. Lee, L. Mé, and A. Wespi, editors, Proceedings of the Fourth International Symposium on the Recent Advances in Intrusion Detection (RAID'2001), LNCS, pages 1–21.
[44]Thomas H. Ptacek and Timothy N. Newsham. Insertion, evasion, and denial of service: Eluding network intrusion detection, January 1998.
[45]Jon Postel. Rfc 791 internet protocol - darpa internet program protocol specification, 1981.
[46]Randy Smith, Cristian Estan, and Somesh Jha. Backtracking algorithmic complexity attacks against a nids. In ACSAC '06: Proceed- ings of the 22nd Annual Computer Security Applications Conference, pages 89–98. IEEE Computer Society.
[47]Randy Smith, Cristian Estan, and Somesh Jha. Xfa: Faster signature matching with extended automata. In SP '08: Proceedings of the 2008 IEEE Symposium on Security and Privacy, pages 187–201. IEEE Computer Society.
[48]Karen Scarfone and Peter Mell. Guide to intrusion detection and prevention systems (idps), February 2007.
[49]S. E. Smaha. Haystack: An intrusion detection system. In Proceedings of the 4th Aerospace Computer Security Application Conference, pages 37–44.
[50]Umesh Shankar and Vern Paxson. Active mapping: Resisting nids evasion without altering traffic. In SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy, page 44. IEEE Computer Society.
[51]Stonesoft. Stonesofttm ips-3205, 2011.
[52]Hemant Sengar, Duminda Wijesekera, Haining Wang, and Sushil Jajodia. VoIP intrusion detection through interacting protocol state machines. In 2006 International Conference on Dependable Systems and Networks (DSN 2006), pages 393–402. IEEE Computer Society, 2006.
[53]Greg Taleck. Ambiguity resolution via passive OS fingerprinting. In Giovanni Vigna, Erland Jonsson, and Christopher Krügel, editors, Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID'2003), volume 2820 of Lecture Notes in Computer Science, pages 192–206. Springer.
[54]Chinyang Henry Tseng, Tao Song, Poornima Balasubramanyam, Calvin Ko, and Karl Levitt. A specification-based intrusion detection model for OLST. In Proceedings of 8th International Symposium on Recent Advances in Intrusion Detection (RAID '2005).
[55]Mythili Vutukuru, Hari Balakrishnan, and Vern Paxson. Efficient and robust TCP stream normalization. In IEEE Symposium on Security and Privacy, pages 96–110.
[56]Matthias Vallentin, Robin Sommer, Jason Lee, Craig Leres, Vern Paxson, and Brian Tierney. The NIDS cluster: Scalable, stateful network intrusion detection on commodity hardware. In Proceedings of the 10th International Conference on Recent Advances in Intrusion Detection, RAID'07, pages 107–126, Berlin, Heidelberg, 2007. Springer-Verlag.
[57]P. Manadhata. An Attack Surface Metric. PhD thesis, School of Computer Science, Carnegie Mellon University, 2008.
[58]Barreto, Alexandre B.; Costa, Paulo C. G., and Yano, Edgar (2012) A Semantic Approach to Evaluate the Impact of Cyber Actions to the Physical Domain. In Proceedings of the 7th

International Conference on Semantic Technologies for Intelligence, Defense, and Security (STIDS 2012). Costa, P.; Laskey, K. (eds.), pp. 64-71. Fairfax, VA, USA, October 23-26, 2012.

[59]Barreto, Alexandre B.; Costa, Paulo C. G.; and Yano, Edgar (2013) Using a Semantic Approach to Cyber Impact Assessment. In Proceedings of the 8th International Conference on Semantic Technologies for Intelligence, Defense, and Security (STIDS 2013). Laskey, K., Emmons, Y.; Costa, P.C.G. (eds.), pp. 101-108. Fairfax, VA, USA, November 2013.

[60]Goodall, J.R. Secure Decisions Div., Appl. Visions, Inc., Northport, NY, USA ; D'Amico, A. ; Kopylec, J.K. CAMUS: AUTOMATICALLY MAPPING CYBER ASSETS TO MISSIONS AND USERS. In Proc. Military Communications Conference. MILCOM 2009. IEEE; 2009

[61]Jakobsen, Gabriel, Mission Cyber Security Situation Assessment using Impact Dependency Graphs. 2011 Proceedings of the 14th International Conference on Information Fusion (FUSION). IEEE; 2011"

[62]Sawilla, R.E.; Wiemer, D.J., Automated computer network defence technology demonstration project (ARMOUR TDP): Concept of operations, architecture, and integration framework. 2011 IEEE International Conference on Technologies for Homeland Security. IEEE; 2011"

[63]Chow, S.; Alcatel, Ottawa, Ont. ; Gustave, C. ; McFarlane, B. ; Wiemer, D. et al., Situation Monitoring and Analysis of Security Risk for Networked Services, Proc. Military Communications Conference – MILCOM 2006. IEEE, 2006.

[64]Ou, Xinming, Sudhakar Govindavajhala, and Andrew W. Appel. "MulVAL: A logic-based network security analyzer." *14th USENIX Security Symposium*. 2005.

[65]Christophe Dousson, Suivi d'évolutions et reconnaissance de chroniques, Thèse de doctorat, Université Paul Sabatier, 1994.

[66]Benjamin Morin and Hervé Debar, Correlation of Intrusion Symptoms: An Application of Chronicles, Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID'2003), 2003.

[67]A. Abou El Kalam, R. El Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miège, C. Saurel et G. Trouessin. Organization Based Access Control. IEEE 4th International Workshop on Policies for Distributed Systems and Networks (Policy 2003), Lake Come, Italy, June 4-6, 2003.

[68]Braun, L., A. Didebulidze, et al. (2010). Comparing and improving current packet capturing solutions based on commodity hardware. Proceedings of the 10th annual conference on Internet measurement. Melbourne, Australia, ACM**:** 206-217.

[69]Fusco, F. and L. Deri (2010). High speed network traffic analysis with commodity multi-core systems. Proceedings of the 10th annual conference on Internet measurement. Melbourne, Australia, ACM**:** 218-224.

[70]Handley, M., V. Paxson, et al. (2001). Network intrusion detection: evasion, traffic normalization, and end-to-end protocol semantics. Proceedings of the 10th conference on USENIX Security Symposium - Volume 10, Berkeley, CA, USA, USENIX Association.

[71]Ptacek, T. H. and T. N. Newsham (1998). Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection, http://www.securityfocus.com/data/library/ids.ps.

[72]Shankar, U. and V. Paxson (2003). Active Mapping: Resisting NIDS Evasion without Altering Traffic. SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy, Washington, DC, USA, IEEE Computer Society.

[73]Vutukuru, M., H. Balakrishnan, et al. (2008). Efficient and Robust TCP Stream Normalization. IEEE Symposium on Security and Privacy.

[74]Dreger, Holger and Feldmann, Anja and Mai, Michael and Paxson, Vern and Sommer, Robin, **Dynamic application-layer protocol analysis for network intrusion detection**, *Proceedings of*

the 15th conference on USENIX Security Symposium - Volume 15, 2006, USENIX-SS'06, Berkeley, CA, USA, USENIX Association

[75]Georges Bossert and Frederic Guihery and Guillaume Hiet, **Towards Automated Protocol Reverse Engineering Using Semantic Information**, *ASIACCS*, 2014

[76]Debar, H., M. Dacier, et al. (1999). "Towards a Taxonomy of Intrusion-Detection Systems." Computer Networks.

[77]Smaha, S. E. (1988). Haystack: An Intrusion Detection System. Proceedings of the 4th Aerospace Computer Security Application Conference.

[78]Heberlein, L. T., G. Dias, et al. (1990). A network security monitor. Proceedings ot the 1990 IEEE Computer Society Symposiumon Research in Security and Privacy, Oakland, CA.

[79]Anderson, D., T. F. Lunt, et al. (1995). Detecting Unusual Program Behavior Using the Statistical Components of NIDES. Menlo Park, CA, SRI International, Computer Science Laboratory.

[80]Eckmann, S. T., G. Vigna, et al. (2002). "STATL: an attack language for state-based intrusion detection." Journal of Computer Security **10**(1-2): 71-103.

[81]Pouzol, J.-P. and D. Mireille (2001). From Declarative Signatures to Misuse IDS. Proceedings of the Fourth International Symposium on the Recent Advances in Intrusion Detection (RAID'2001).

[82]Hadžiosmanović;, Dina and Simionato, Lorenzo and Bolzoni, Damiano and Zambon, Emmanuele and Etalle, Sandro, **N-Gram against the Machine: On the Feasibility of the N-Gram Network Analysis for Binary Protocols**, *Research in Attacks, Intrusions, and Defenses*, Springer Berlin Heidelberg, 2012, 7462, Lecture Notes in Computer Science

[83]Sommer, Robin and Paxson, Vern, **Outside the Closed World: On Using Machine Learning for Network Intrusion Detection**, *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, 2010, SP '10, pp. 305--316, Washington, DC, USA, IEEE Computer Society

[84]Tseng, C. H., T. Song, et al. (2005). A Specification-based Intrusion Detection Model for OLSR. Proceedings of 8th International Symposium on Recent Advances in Intrusion Detection (RAID '2005)

[85] Sengar, H., D. Wijesekera, et al. (2006). VoIP Intrusion Detection Through Interacting Protocol State Machines. Proceedings of the International Conference on Dependable Systems and Networks, IEEE Computer Society**:** 393-402.

[86]Amann, Bernhard and Sommer, Robin and Sharma, Aashish and Hall, Seth, **A Lone Wolf No More: Supporting Network Intrusion Detection with Real-time Intelligence**, *Proceedings of the 15th International Conference on Research in Attacks, Intrusions, and Defenses*, 2012, RAID'12, pp. 314--333, Berlin, Heidelberg, Springer-Verlag

[87]Alvaro A. Càrdenas and John S. Baras and Karl Seamon, **A Framework for the Evaluation of Intrusion Detection Systems**, *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*, 2006, pp. 63--77, Washington, DC, USA, IEEE Computer Society

[88]Axelsson, Stefan, **The base-rate fallacy and its implications for the difficulty of intrusion detection**, *Proceedings of the 6th ACM conference on Computer and communications security*, 1999, CCS '99, pp. 1--7, New York, NY, USA, ACM

[89]B. Zhu and S. Sastry. *SCADA-Specific Intrusion Detection/Prevention Systems: A Survey and Taxonomy*. In Proceedings of the First Workshop on Secure Control Systems (SCS'10), Stockholm, Sweden, 2010.

[90]Steven Cheung, Bruno Dutertre, Martin Fong, Ulf Lindqvist, Keith Skinner and Alfonso Valdes. Using Model-based Intrusion Detection for SCADA Networks. *In Proceedings of the SCADA Security Scientific Symposium*. Miami Beach, Florida, January 2007.

[91]Lin, Hui and Slagell, Adam and Di Martino, Catello and Kalbarczyk, Zbigniew and Iyer, Ravishankar K., **Adapting Bro into SCADA: Building a Specification-based Intrusion Detection System for the DNP3 Protocol**, *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, 2013, CSIIRW '13, pp. 5:1--5:4, New York, NY, USA, ACM

[92] Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based access control models. IEEE Computer, 29 (1996) 38–47.

[93] Yuan, E., Tong, J.: Attributed Based Access Control (ABAC) for Web Services. In: 2005 IEEE International Conference on Web Services. (2005) 561–569.

[94] Abou-El-Kalam, A., Baida, R., Balbiani, P., Benferhat, S., Cuppens, F., Deswarte, Y., Miège, A., Saurel, C., Trouessin, G.: Organization Based Access Control. In: IEEE 4th International Workshop on Policies for Distributed Systems and Networks. (2003) 120–131.

[95] Cuppens, F., Cuppens-Boulahia, N.: Modeling Contextual Security Policies. International Journal of Information Security, 7 (2008) 285–305.

[96] Preda, S., Cuppens, F., Cuppens, N., Garcia-Alfaro, J., Toutain, L.: Dynamic deployment of context-aware access control policies for constrained security devices. Journal of Systems and Software, 84 (2011), 1144-1159.

[97] Hachem, N., Garcia-Alfaro, J., Debar, H.: An Adaptive Mitigation Framework for Handling Suspicious Network Flows via MPLS Policies. In: 18th Nordic Conference on Secure IT Systems. (2013) 297-312.

[98]R.E. Sawilla; D.J. Wiemer, "Automated computer network defence technology demonstration project (ARMOUR TDP): Concept of operations, architecture, and integration framework," Technologies for Homeland Security (HST), 2011 IEEE International Conference on , vol., no., pp.167-172, 15-17 Nov. 2011.

[99]W. Li; S. Tian. 2010, “An ontology-based intrusion alerts correlation system”. Expert Syst. Appl. Vol. 37, no. 10, October 2010.

[100] N. Cuppens-Boulahia; F. Cuppens; J.E. Lopez De Vergara; E. Vazquez; J. Guerra; H. Debar, "An ontology-based approach to react to network attacks," Risks and Security of Internet and Systems, 2008. CRiSIS '08. Third International Conference on , vol., no., pp.27-35, 28-30 Oct. 2008.

[101] Ljiljana Stojanovic, Andreas Abecker, Nenad Stojanovic, Rudi Studer: Ontology-Based Correlation Engines. ICAC 2004: 304-305

[102] Nicolett, Mark & Williams, Amrit T. - “Security Information and Event Management Technology Evaluation Criteria”, 1H05, ref. G00126899, published May 16, 2005 by Gartner Inc.

[103] Nicolett, Mark & Kavanagh, Kelly M. - “Magic Quadrant for Security Information and Event Management”, ref. G00246886, published May 7, 2013 by Gartner Inc.

[104] Sen. Sarbanes, Paul & Rep. Oxley, Michael G. - “Sarbanes-Oxley Act of 2002”, United State of America Public Law 107-204, 30th July 2002, available online at http://www.gpo.gov/fdsys/pkg/PLAW-107publ204/html/PLAW-107publ204.htm.

[105] PCI Security Standards Council, LLC - “Payment Card Industry (PCI) Data Security Standard, Requirements and Securtiy Assessment Procedures”, Version 3.0, Nov. 2013, available online at https://www.pcisecuritystandards.org/documents/PCI_DSS_v3.pdf.

[106] ETSI, TISPAN, Methods and protocols - “Method and proforma for Threat, Risk, Vulnerability Analysis”, ETSI TS 102 165-1, v4.2.1, Dec. 2006, available online at http://portal.etsi.org/mbs/Referenced%20Documents/ts_10216501v040201p.pdf.

[107] ISO/IEC International Standard - “Information technology - Security techniques - Information security management systems - Requirements”, ISO/IEC 27001:2013, Version 25 Sep. 2013

[108] ISO/IEC International Standard - “Information technology - Security techniques - Evaluation criteria for IT security”, ISO/IEC 15408:2009, Version 3.1 Rev. 4.

[109] Sen. Kennedy, Edward M. & Sen. Kassebaum Baker, Nancy L. - “Health Insurance Portability and Accountability Act of 1996”, United State of America Public Law 104-191, 21st Aug. 1996, available online at www.gpo.gov/fdsys/pkg/PLAW-104publ191/html/PLAW-104publ191.htm.

[110] [Lane, Adrian - “Understanding and Selecting SIEM/LM: Deployment Models”, 22nd June 2010, available online at https://securosis.com/blog/understanding-and-selecting-siem-lm-deployment-models.

[111] Lane, Adrian & Rothman, Mike - “Understanding and Selecting SIEM/Log Management”, Version 2.0, 25th Aug. 2010, available online at https://securosis.com/assets/library/reports/Securosis_Understanding_Selecting_SIEM_LM_FINAL.pdf.

[112] Ingols, K.; Chu, M.; Lippmann, R.; Webster, S.; Boyer, S., "Modeling Modern Network Attacks and Countermeasures Using Attack Graphs," *Computer Security Applications Conference, 2009. ACSAC '09. Annual* , vol., no., pp.117,126, 7-11 Dec. 2009

[113] K. Alsubhi, E. Al-Shaer, and R. Boutaba. Alert prioritization in intrusion detection systems. In Network Operations and Management Symposium, 2008. NOMS 2008. IEEE, pages 33 –40, april 2008.

[114] Frédéric Cuppens and Alexandre Miège. Alert correlation in a cooperative intrusion detection framework. In Proceedings of the IEEE Symposium on Security and Privacy, pages 202–215, 2002.

[115] Frédéric Cuppens and Rodolphe Ortalo. LAMBDA: A Language to Model a Database for Detection of Attacks. In H. Debar, L. Mé, and S. F. Wu, editors, Proceedings of the Third International Workshop on the Recent Advances in Intrusion Detection (RAID’2000),number 1907 in LNCS, pages 197–216, October 2000.

[116] Oliver M. Dain and Robert K. Cunningham. Building scenarios from a heterogeneous alert stream. In Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, 2001.

[117] Steven T. Eckmann, Giovanni Vigna, and Richard A. Kemmerer. STATL: an attack language for state-based intrusion detection. Journal of Computer Security, 10(1-2):71–103, 2002.

[118] Jean Goubault-Larrecq and Julien Olivain. A smell of orchids. In Runtime Verification, volume 5289 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2008.

[119] Jorge Herrerias and Roberto Gomez. A log correlation model to support the evidence search process in a forensic investigation. In Proceedings of the Second International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE’07), 2007.

[120] Klaus Julisch. Clustering intrusion detection alarms to support root cause analysis. ACM Transactions on Information and System Security, 6(4), 2003.

[121] Dongyoung Kim, Hyo Chan Bang, and Jung-Chan Na. Intrusion alert normalization method using awk scripts and attack name database. In Advanced Communication Technology, 2005, ICACT 2005. The 7th International Conference on, volume 1, pages 608 – 611 Vol. 1, feb. 2005.

[122] Christopher Kruegel and William Robertson. Alert Verification - Determining the Success of Intrusion Attempts. In Workshop on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA), 2004.

[123] Christopher Kruegel, William Robertson and Giovanni Vigna.Using alert verification to identify successful intrusion attempts.Practice in Information Processing and Communication (PIK), 27(4), October 2004.

[124] Philippe Lagadec. Visualisation et Analyse de Risque Dynamique pour la Cyber-Défense. In Proceedings of SSTIC'2010, pages 3–31, 2010.

[125] Xuejiao Liu, Debao Xiao, and Xi Peng. Towards a collaborative and systematic approach to alert verification. Journal of Software, 3(9):77–84, 2008.

[126] Stefanos Manganaris, Marvin Christensen, Dan Zerkle, and Keith Hermiz. A data mining analysis of rtid alarms. Web proceedings of the 2nd International Workshop on Recent Advances in Intru- sion Detection (RAID'99), http://www.raid-symposium.org/raid99, September 1999.

[127] Frederic Massicotte, Yvan Labiche, and Lionel C. Briand. Toward automatic generation of intrusion detection verification rules. In Proceedings of the Annual Computer Security Applications Conference (ACSAC'2008), 2008.

[128] Benjamin Morin, Ludovic Mé, Hervé Debar, and Mireille Duccassé. M4d4: a logical framework to support alert correlation in intrusion detection. Information Fusion, 10(4):285–299, October 2009.

[129] Steven Noel and Sushil Jajodia. Optimal IDS sensor placement and alert prioritization using attack graphs. Journal of Network and Systems Management, 16:259–275, 2008. 10.1007/s10922-008-9109-x.

[130] Phillip A. Porras, Martin W. Fong, and Alfonso Valdes. A mission impact-based approach to infosec alarm correlation. In Andreas Wespi, Giovanni Vigna, and Luca Deri, editors, Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID'2002), volume 2516 of Lecture Notes in Computer Science, pages 95–114. Springer, 2002.

[131] Xinzhou Qin and Wenke Lee. Statistical causality analysis of infosec alert data. In Giovanni Vigna, Erland Jonsson, and Christopher Krügel, editors, Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID'2003), volume 2820 of Lecture Notes in Computer Science, pages 73–93. Springer, September 8-10 2003.

[132] Xinzhou Qin and Wenke Lee. Discovering novel attack strategies from infosec alerts. In Proceedings of the 9th European Symposium on Research in Computer Security, Sophia Antipolis, pages 439–456, 2004.

[133] John P. Rouillard. Real-time log file analysis using the simple event correlator (sec). In LISA '04: Proceedings of the 18th USENIX conference on System administration, pages 133–150, Berkeley, CA, USA, 2004. USENIX Association.

[134] R.Sadoddin and A. A. Ghorbani. An incremental frequent structure mining framework for real-time alert correlation. Computers & Security, 28(3-4):153–173, 2009.

[135] Reginald E. Sawilla and Xinming Ou. Identifying critical attack assets in dependency attack graphs. In Proceedings of the 13th European Symposium on Research in Computer Security: Com- puter Security, ESORICS '08, pages 18–34, Berlin, Heidelberg, 2008. Springer-Verlag.

[136] Umesh Shankar and Vern Paxson. Active mapping: Resisting NIDS evasion without altering traffic. In SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy, page 44, Washington, DC, USA, 2003. IEEE Computer Society.

[137] Steven J. Templeton and Karl Levitt. A requires/provides model for computer attacks. In Proceedings of the 2000 New Security Paradigms Workshop (NSPW'00), pages 31–38, September 2000.

[138] Yohann Thomas, Benjamin Morin, and Hervé Debar. Improving Security Management through Passive Network Observation. In First International Conference on Availability, Reliability and Security (ARES), 2006.

[139] Eric Totel, Bernard Vivinis, and Ludovic Mé. A Language Driven Intrusion Detection System for Event and Alert Correlation. In Proceedings ot the 19th IFIP International Information Security Conference, pages 209–224, Toulouse, August 2004. Kluwer Aca- demic.

[140] Risto Vaarandi. Real-time classification of IDS alerts with data mining techniques. In IEEE MILCOM Conference, 2009.

[141] Fredrik Valeur. Real-Time Intrusion Detection Alert Correlation. PhD thesis, UNIVERSITY OF CALIFORNIA, 2006.

[142] J. Viinikka, H. Debar, L. Mé, A. Lehikoinen, and M. Tarvainen. Processing intrusion detection alert aggregates with time series modeling. Information Fusion, 10(4):312–324, 2009.

[143] Alfonso Valdes and Keith Skinner. Probabilistic alert correlation. In W. Lee, L. Mé, and A. Wespi, editors, Proceedings of the Fourth International Symposium on the Recent Advances in Intrusion De- tection (RAID'2001), number 2212 in LNCS, pages 54–68, October 2001.

[144] Min Xiao and Debao Xiao. Alert verification based on attack classification in collaborative intrusion detection. In Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007., volume 2, pages 739 –744, 30 2007-aug. 1 2007.

[145] William Yurcik: "VisFlowConnect-IP: A Link-Based Visualization of NetFlows for Security Monitoring",FIRST 2006 Baltimore Maryland USA.

[146] G. Conti, J. Grizzard, M. Ahamad, and H. Owen: "Visual exploration of malicious network objects using semantic zoom, interactive encoding and dynamic queries", VizSEC 2005.

[147] T. Takada and H. Koike.: "Mielog: A highly interactive visual log browser using information visualization and statistical analysis", LISA 2002, Philadelphia, Pennsylvania, USA.

[148] T . Takada and H. Koike : "Tudumi: Information visualization system for monitoring and auditing computer logs", Information Visualization 2002.

[149] H. Koike and K. Ohno. "Snortview: Visualization system of snort logs", VizSec2004

[150] K. Abdullah, C. Lee, G. Conti, J. A. Copeland, and J. Stasko: "IDS rainstorm: Visualizing IDS alarms", VizSEC 2005.

[151] Y. Hideshima and H. Koike. "Starmine: A visualization system for cyber attacks", APVIS2006, Tokyo, Japan. CRPIT, 60.

[152] S. Foresti, J. Agutter, Y. Livnat, S. Moon, and R. Erbacher: "Visual correlation of network alerts", Computer Graphics and Applications, IEEE (Volume:26 , Issue: 2), 2006.

[153] William J Matuszak, Lisa Dipippo and Yan Lindsay Sun. "CyberSAVE: Situational Awareness Visualization for Cyber Security of Smart Grid Systems , VizSEC 2013.

[154] Robert F. Erbacher. " Cyber Command Gauge Cluster: Visualization Design for Immediate High-Level Situational Assessment", VizSEC 2012.

[155] Harrison-Spahn-Iannacone. "NV: Nessus Vulnerability Visualization for the Web, VizSEC 2012.

[156] Inoue, Daisuke and Eto, Masashi and Suzuki, Koei and Suzuki, Mio and Nakao, Koji. "DAEDALUS-VIZ: Novel Real-time 3D Visualization for Darknet Monitoring-based Alert System, VizSEC 2012

[157] Christopher Humphries, Nicolas Prigent, Christophe Bidan and Frédéric Majorczyk, ELVIS, Extensible Log VISualization, Vizsec 2013.

[158] Denning, Dorothy E. "An intrusion-detection model." *Software Engineering, IEEE Transactions on* 2 (1987): 222-232.

[159] McAfee Foundstone Professional Services - “Creating and Maintaining a SOC - The details behind successful Security Operations Centers”, McAfee White Paper, 2013, available online at http://www.mcafee.com/fr/resources/white-papers/foundstone/wp-creating-maintaining-soc.pdf.

[160] EMC - “RSA Security Operation Management - Orchestrate Intelligence, Process, and Resources in the SOC”, EMC Data Sheet, 2013, available online at http://www.emc.com/collateral/data-sheet/h12427-ds-rsa-security-operations-management.pdf.

[161] Hewlett-Packard - “Security Operations - Building a successful SOC”, HP Business white paper, ref. 4AA4-6169ENW, April 2013, available online at http://h20195.www2.hp.com/v2/GetDocument.aspx?cc=us&doclang=EN_US&docname=4AA4-6169ENW.

[162] Hewlett-Packard - “SOC generations: HP ESP Security Intelligence and Operations Consulting Services”, HP Business whitepaper, ref. 4AA4-6539ENW, May 2013, available online at http://h20195.www2.hp.com/V2/GetDocument.aspx?docname=4AA4-6539ENW&cc=us&lc=en.

[163] Tsutomu Shimomura - “Technical details of the attack described by Markoff in NYT”, email sent to comp.security.misc, 25 Jan 1995, available online at http://www.cs.berkeley.edu/~daw/security/shimo-post.txt.