

# **RESISTO:**

## **D4.5\_DESCRIPTION AND DEFINITION OF THE CORRELATOR INTELLIGENCE**



# RESISTO

## D4.5 - DESCRIPTION AND DEFINITION OF THE CORRELATOR INTELLIGENCE

<b>Document Manager:</b>	Alberto NERI	LDO	Editor
--------------------------	--------------	-----	--------

<b>Project Title:</b>	Description and Definition of the Correlator Intelligence
<b>Project Acronym:</b>	RESISTO
<b>Contract Number:</b>	786409
<b>Project Coordinator:</b>	LEONARDO
<b>WP Leader:</b>	TEI

<b>Document ID N°:</b>	RESISTO_D4.5_191126_V1.0	<b>Version:</b>	V1.0
<b>Deliverable:</b>	D4.5	<b>Date:</b>	26/11/2019
		<b>Status:</b>	Approved

<b>Document classification</b>	<b>Public</b>
--------------------------------	---------------

Approval Status	
<b>Prepared by:</b>	Alberto NERI, Francesco TORELLI, Lucio BRIGHELLA (LDO)
<b>Approved by: (WP Leader)</b>	Giuseppe CELOZZI (TEI)
<b>Approved by: (Coordinator)</b>	Alberto NERI (LDO)
<b>Approved by: (Advisory Board Leader)</b>	NA
<b>Security Approval (Security Advisory Board Leader)</b>	NA

## CONTRIBUTING PARTNERS

Name	Company / Organization	Role / Title
Alberto NERI, Catherine MINCIOTTI, Alessia GALLI, Massimo FRATINI, Emanuele AONZO, Fabio CIRINNA'	LDO	Editor/Contributor

## DISTRIBUTION LIST

Name	Company / Organization	Role / Title
PMT	RESISTO CONSORTIUM	NA
Markus MULLER	EC DG REA	EC Programme Officer
General Public	NA	NA

## REVISION TABLE

Version	Date	Modified Pages	Modified Sections	Comments
0.1	31/10/2019	All	All	First Draft
1.0	04/11/2019	All	All	Final Release

## COPYRIGHT STATEMENT



© 2018-2021 This document and its content are the property of the RESISTO Consortium. All rights relevant to this document are determined by the applicable laws. Access to this document does not grant any right or license on the document or its contents. This document or its contents are not to be used or treated in any manner inconsistent with the rights or interests of the RESISSO Consortium or the Partners detriment and are not to be disclosed externally without prior written consent from the RESISTO Partners. Each RESISTO Partner may use this document in conformity with the RESISTO Consortium Grant Agreement provisions.

The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation Program, under the Grant Agreement No 786409.

The views and opinions in this document are solely those of the authors and contributors, not those of the European Commission.

## PROJECT CONTACT



LEONARDO  
Via Puccini 2 – Genova (GE) – 16154 – Italy  
Tel.: +39 348 6505565  
E-Mail: [bruno.saccomanno@leonardocompany.com](mailto:bruno.saccomanno@leonardocompany.com)

## RESISTO PROJECT – PUBLISHABLE EXTENDED ABSTRACT

Communications play a fundamental role in the economic and social well-being of the citizens and on operations of most of the CIs. Thus, they are a primary target for criminals having a multiplier effect on the power of attacks and providing enormous resonance and gains. Also extreme weather events and natural disasters represents a challenge due to their increase in frequency and intensity requiring smarter resilience of the Communication CIs, which are extremely vulnerable due to the ever-increasing complexity of the architecture also in light of the evolution towards 5G, the extensive use of programmable platforms and exponential growth of connected devices. The fact that most enterprises still manage physical and cyber security independently represents a further challenge. RESISTO platform is an innovative solution for Communication CIs holistic situation awareness and enhanced resilience (aligned with ECSO objectives). Based on an Integrated Risk and Resilience analysis management and improvement process availing all resilience cycle phases (prepare, prevent, detect, absorb, etc.) and technical resilience capabilities (sense, model, infer, act, adopt), RESISTO implements an innovative Decision Support System to protect communication infrastructures from combined cyber-physical threats exploiting the Software Defined Security model on a suite of state of the art cyber/physical security components (Blockchain, Machine Learning, IoT security, Airborne threat detection, holistic audio-video analytics) and services (Responsible Disclosure Framework) for detection and reaction in presence of attacks or natural disasters. Through RESISTO Communications Operators, will be able to implement a set of mitigation actions and countermeasures that significantly reduce the impact of negative events in terms of performance losses, social consequences, and cascading effects in particular by bouncing efficiently back to original and forward to operational states of operation.

## EXECUTIVE SUMMARY

This deliverable illustrates the outputs of Task 4.5 “Complex Event Recognition (Cyber-Physical correlator)”, which is included in WP4 “Tools and techniques for Monitoring and Detection of cyber-physical threats”. Its aim is to describe the techniques used by the Correlator Intelligence Module to recognize physical and cyber threats. It will deal with how these techniques will be used within RESISTO, with a focus on techniques based on machine learning technologies and deterministic techniques based on Complex Event Processing (CEP) tools.

## CONTENTS

<b>RESISTO:</b>	<b>1</b>
<b>D4.5_DESCRIPTION AND DEFINITION OF THE CORRELATOR INTELLIGENCE</b>	<b>1</b>
<b>1. Introduction</b>	<b>9</b>
1.1. Context	9
1.2. Objectives and scope	9
1.3. Document Structure	9
1.4. Acronyms and symbols	9
<b>2. Correlator intelligence inside RESISTO</b>	<b>11</b>
<b>3. Complex Event Processing</b>	<b>14</b>
3.1. Events in CEP	14
3.2. Complex event patterns	15
3.3. Esper engine	15
3.4. Event pattern language	16
3.4.1. EPL queries	16
3.4.2. EPL patterns	17
<b>4. Machine Learning Recognition Strategy</b>	<b>18</b>
<b>5. RESISTO Applicability</b>	<b>19</b>
5.1. Cyber and physical protection of network	19
5.2. Identification of probability of service loss	20
<b>6. CONCLUSION</b>	<b>21</b>
<b>7. REFERENCES</b>	<b>22</b>

## LIST OF FIGURES

Figure 1 – RESISTO architecture.....	11
Figure 2 – Short Term Control Loop Functional Architecture.....	12
Figure 3 – Cyber / Physical Threats Correlator Functional Architecture .....	13
Figure 4 – Event Stream Processing and Correlation.....	14
Figure 5 – Event Stream Intelligence .....	16
Figure 6 – Use case 1: Testbed - Logical design .....	19
Figure 7 – Use case2: workflow .....	20



## 1. INTRODUCTION

### 1.1. Context

The present deliverable D4.5 “Description and definition of the Correlator intelligence” illustrates the outputs of Task 4.5 “Complex Event Recognition (Cyber-Physical correlator)”, which is included in WP4 “Tools and techniques for Monitoring and Detection of cyber-physical threats”.

### 1.2. Objectives and scope

The aim of D4.5 is to describe the techniques used by the Correlator Intelligence Module to recognize physical and cyber threats. The document will deal with how these techniques will be used within RESISTO, with a focus on techniques based on machine learning technologies and deterministic techniques based on Complex Event Processing (CEP) tools.

### 1.3. Document Structure

The document is structured as follows:

- Introduction: introduction to the document
- Correlator Intelligence within RESISTO: description of the module within the general architecture of RESISTO.
- Machine Learning Recognition Strategy: description of correlation strategies based on modern machine learning techniques
- Complex Event Processing: description of correlation strategies based on real-time deterministic techniques, on the model of CEP (Complex Event Processing) tools
- RESISTO Applicability: example of applicability of the techniques described in a use-case

### 1.4. Acronyms and symbols

<b>CEP</b>	Complex Event Processing
<b>CI</b>	Communication Infrastructure
<b>DDOS</b>	Distributed Denial of Service
<b>EPL</b>	Event Processing Language
<b>EU</b>	European Union
<b>ICT</b>	Information and Communications Technology
<b>KPI</b>	Key Performance Indicators
<b>HW</b>	HardWare
<b>ML</b>	Machine Learning
<b>SOC</b>	Security Operations Center

<b>SQL</b>	Structured Query Language
<b>TLC</b>	Telecommunication Line Controller
<b>WP</b>	Work Package

## 2. CORRELATOR INTELLIGENCE INSIDE RESISTO

The logical architecture of RESISTO is depicted in the figure below. The platform integrates two control loops both running on top of the Communication Infrastructure and strongly interlinked with each other.

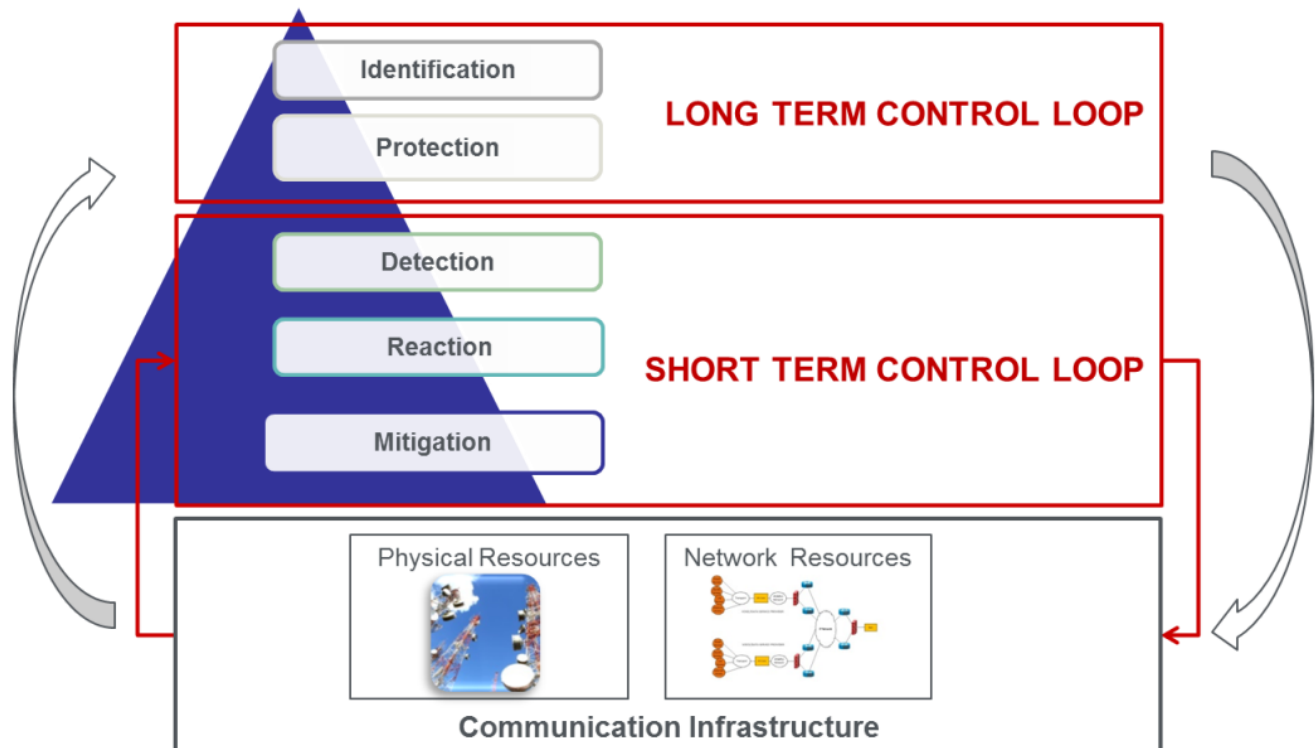


Figure 1 – RESISTO architecture

In detail, the *Long Term Control Loop* is an offline component that assesses asset vulnerabilities and security threats and consequently defines the configuration of the system, updating it on a periodic basis or when particular events take place. The *Short Term Control Loop* is a runtime component that reacts to attacks and threats that may impact the operational life of the system. It enhances situation awareness, immediate response and bouncing back up to forward even better states of systems as preselected by the long-term approach. In particular, the *Short Term Control Loop*:

- monitors the physical and cyber security status of the infrastructures, correlating the physical and cyber domain events and communication KPIs (Key Performance Indicators) to detect anomalies and provide early warnings on security attacks by detecting threats in advance;
- evaluates the attack impact with respect to performance degradation of detected anomalies and security attacks on the communication CI, and interlinked CIs if known, based on the cascading effect;
- supports decision making providing a qualitative and quantitative What-If analysis tool in order to evaluate the most resilient communication CI reconfiguration;

- drives reaction and mitigation by means of action workflows (composed of directives to intervention teams, physical protection devices activation) and, mainly, of orchestrated Communication Network reconfiguration and protection function activation.

The Short Term Control Loop functional control flow is reported below.

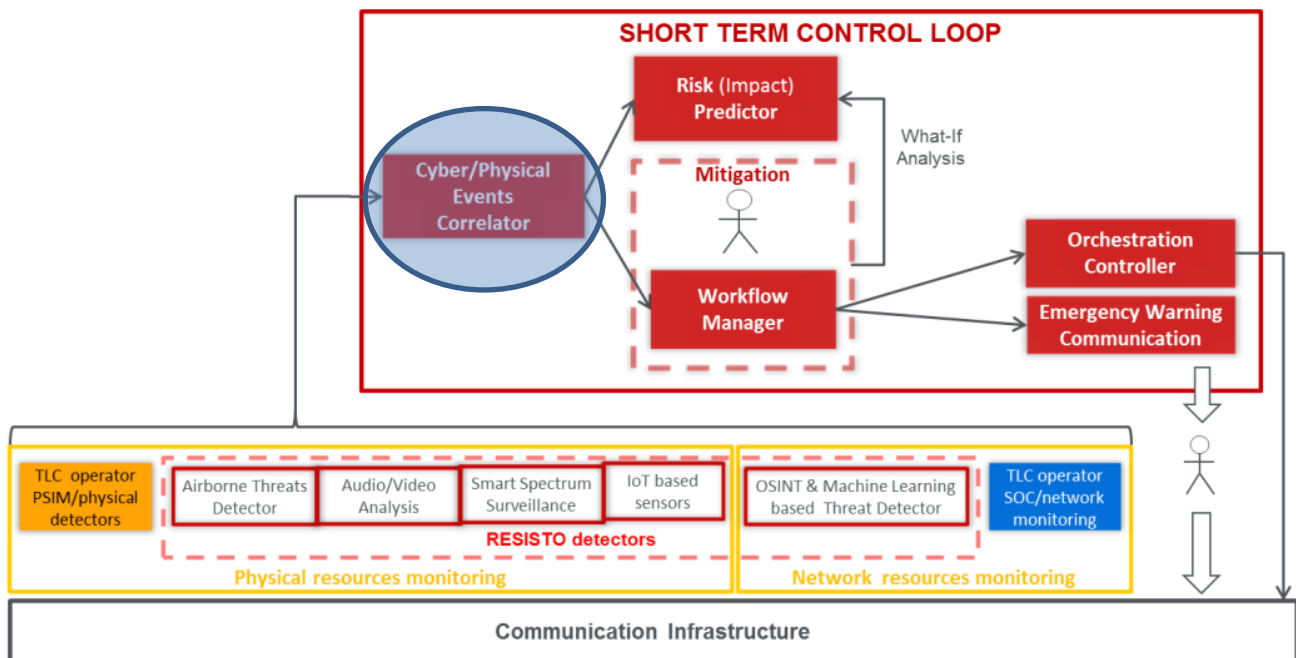


Figure 2 – Short Term Control Loop Functional Architecture

The module object of the document (circled in blue) is the Cyber / Physical Threats Correlator. This module deals with detecting threatening events by correlating the events of the "physical" and "ICT" worlds, using modern techniques and tools based on real-time methodologies. The Cyber/Physical Threats Correlator is composed of the following main components (figure 3):

- Correlator Engine: component correlating data source events and identifying potential threats based on a list of rules configured inside a Complex Event Processing component.
- Machine Learning (ML) Module: component which implements machine-learning algorithms allowing both identification of standard behavioral models based on the traffic originating from network data sources and automatic generation of alerts.
- Interconnection Layer: dedicated to data exchange between data originating from network data sources and the results produced by the Correlator modules performing data analysis.

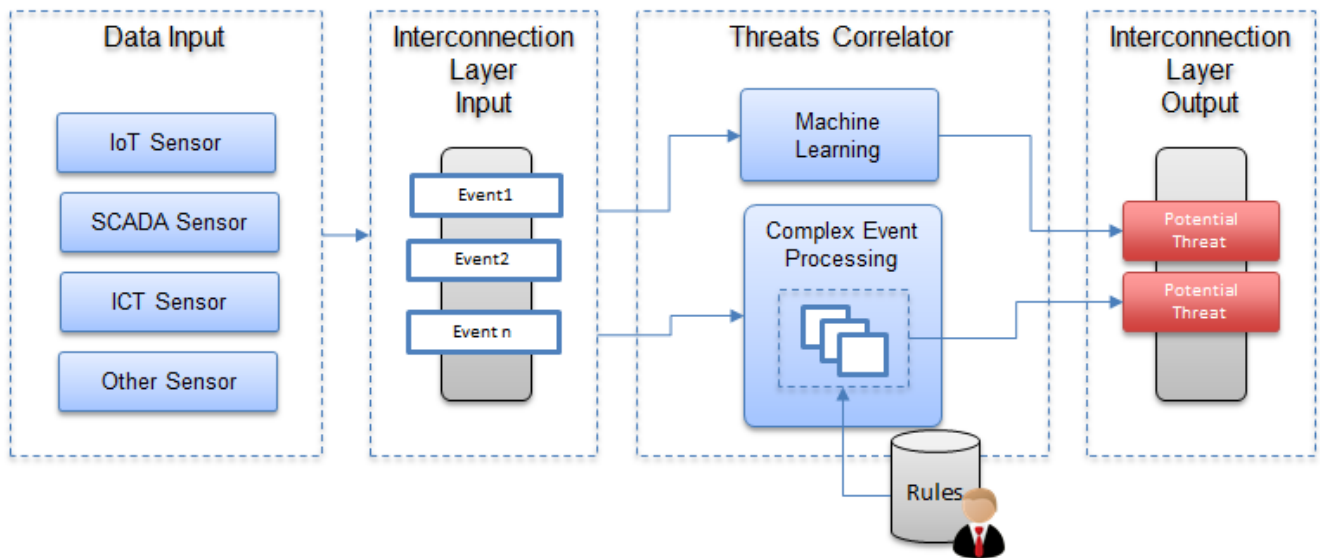


Figure 3 – Cyber / Physical Threats Correlator Functional Architecture

### 3. COMPLEX EVENT PROCESSING

Complex Event Processing (CEP) is a set of techniques and tools that help to manage an event-driven information system. A more complex definition claims that CEP is a network technology that creates actionable, situational knowledge from distributed message-based systems, applications and databases in real-time. It creates a capability to define, manage and predict events, situations, exceptional conditions, opportunities and threats in complex systems. CEP helps to process a huge amount of events in real-time or at least near real-time. Information systems, which implement CEP, are able to react to a set of events from the lowest to the top level, i.e., CEP is able to connect multiple low-level events with low severity to one or several complex events with high, even critical severity. A complex event can be viewed as an aggregation of multiple simple events. A good example of the relationship between simple events and complex events is the detection of a DDOS attack in a computer system. The computer server usually handles a certain throughput of user requests at a time. A single request represents a simple event. If the throughput is normal, the server responds to the user request. However, if the server observes a significant increase of throughput in a short time period, a DDOS attack is detected. This increase of throughput in a short period of time represents a complex event. This can only work if detection mechanisms are able to link low level events (web requests) together to detect a complex event (DDOS attack).

#### 3.1. Events in CEP

Most core elements of Complex Event Processing are events. In CEP terminology, an event is defined as an occurrence within a particular system or domain or a record of activity in a system.

The most important relationships between events (correlations) are:

- **Time:** a relationship according to which events are ordered (event A happened before event B). Time depends upon a clock. Usually, when the event occurs, a timestamp is assigned to the event according to the system clock.
- **Causality:** a dependence relationship between events. An event A depends upon the other events if this event occurs only because other events occurred. When event B depends on the A, we say that A caused B.
- **Aggregation:** an abstraction relationship. If event A consists of a set of events B, then A is an aggregation of all the events B.

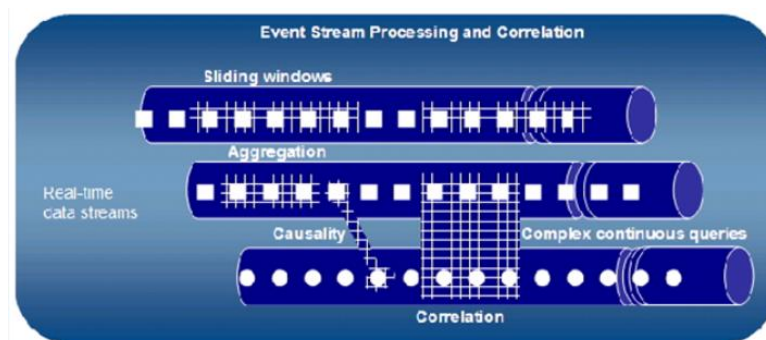


Figure 4 – Event Stream Processing and Correlation

### 3.2. Complex event patterns

To be able to work with complex events, it is necessary to define rules according to which they will be detected and processed. In CEP terminology, these rules are called patterns. In order to define complex event patterns a specific computer language called Event Pattern Language (EPL) is used. An event pattern language is a query language enabling the user to match a subset of events and compute an aggregation or to select events. Patterns described using this language, are quickly translated by the engine and immediately applied against event streams in order to quickly find sets of events that match the defined event patterns. An event pattern is a template that matches sets of events that the user wants to find. It describes not only the events themselves but also their causal dependencies, timing, data parameters and context. The pattern is similar to an SQL query, as it uses the same clauses as SQL (SELECT, FROM, WHERE). The pattern can also be a Boolean combination of multiple different patterns, because usually when the pattern matches, a new event is created and this event can be a part of another pattern. Derived from this, an event pattern is actually a description of a complex event with all its characteristics. One of the most important usages of patterns are event pattern rules. An event pattern rule is defined as a specific reactive rule that defines an action that should be executed whenever an event pattern is matched. This rule contains two parts: a trigger (an event pattern) and an action (an event created whenever the trigger is matched).

### 3.3. Esper engine

The Esper engine works like a database which is turned upside down. Instead of running queries against stored data, the Esper engine stores queries and applies them on flowing data. The data which flow through the Esper engine are called data streams. It is important to note that the engine reacts to events of event streams in real time. Unlike databases where queries are executed immediately after submission, the execution model is continuous because Esper is designed for performing real-time analysis with an immediate response. In Esper terminology data are represented as events and queries are represented as EPL (Event Processing Language) statements. EPL is an SQL-like declarative language for specifying expression-based event pattern matching and event series querying. EPL is a very powerful language which offers benefits ranging from simple functionality such as filtering and aggregating to complex functionality such as detection of the presence or absence of a combination of events. Esper introduces two mechanisms to process events:

1. "event stream queries",
2. "event patterns";

both use the EPL language.

Event stream queries fulfil the event stream analysis requirements of the CEP applications. They provide aggregation, filtering, joining, windows and analysis function functionality for processing event streams.

Event patterns can be used for detection of the presence or absence of events or of a combination of events. It also supports time-based correlation between events.



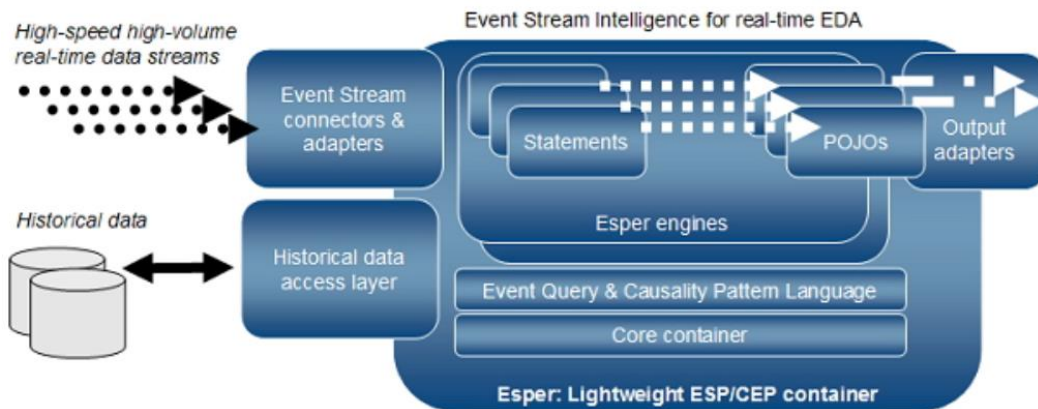


Figure 5 – Event Stream Intelligence

### 3.4. Event pattern language

EPL is a language for expression-based pattern matching and querying of data streams. EPL language syntax is similar to SQL with clauses like SELECT, FROM, WHERE, GROUP BY, HAVING and ORDER BY. However, in EPL, event streams replace data tables and the basic data unit is an event, not a data row. Because events are also data, the concept of joining, filtering and aggregation is leveraged in EPL in same way as in SQL.

Expressions in EPL are written in the form of EPL statements. EPL statements are divided into two main types:

- EPL queries
- EPL patterns.

#### 3.4.1. EPL queries

EPL queries provide filtering, joining, grouping and aggregation of features. They also support the creation of window views and application of function upon event streams. EPL queries are stored in the engine and they publish results to listeners when events match criteria specified in the given query. The listener has to be attached to the EPL query via the API before an EPL query is started. The most used and important clauses in EPL queries are SELECT, FROM, WHERE. They have meanings similar to those in SQL:

- the **SELECT** clause specifies which event properties or events are retrieved by the query.
- the **FROM** clause specifies the names of the streams from which the query reads.
- the **WHERE** clause specifies the search conditions according to which events are filtered. An example of a simple SELECT-FROM-WHERE query is shown in below.

```
select avg(price) from StockTick.win:time(30 sec) where
symbol='ABC'
```

The following EPL query returns the average price of “ABC” stock in the last 30 seconds. In addition to the SELECT clause, the FROM stream definition and the WHERE search condition, the query also



contains the aggregation function avg and a time window view lasting thirty seconds. Esper offers many kinds of built-in functions, such as sum, count, min, max, first, last, etc.; moreover, it allows the creation of user-defined functions. Another important feature of EPL are the views. They specify an expiry policy of events (data windows) and also derive data from events.

### 3.4.2. EPL patterns

EPL patterns are mainly used for the detection of new events in event streams. Event patterns match when the event or multiple events occur in a way that fulfils pattern definition. EPL patterns use a slightly different EPL syntax than EPL queries. EPL patterns consist of:

- pattern atoms;
- pattern operators.

The pattern atoms are the basic building blocks of patterns. The most common pattern atoms are:

- Filter expressions that specify searched events (StockTick(symbol='ABC', price > 100))
- Time-based observers which specify timer interval or time schedule (timer:interval (10 seconds))
- Custom Plug-in.

Pattern operators combine atoms and control the life cycle of patterns:

- Operators which control pattern sub-expression repetition: every, until
- Logical operators: and, or, not
- Operator followed by: ->
- Guards which are conditions to control the life-cycle of the subexpressions: timer:within, timer:withinmax, while-

An example of a simple EPL pattern is shown below.

```
every (e1=SensorEvent -> e2=SensorEvent (ID=e1.ID) ) }
```

The pattern shown above matches every SensorEvent which is followed by another SensorEvent with the same ID. The result will contain data from both events e1 and e2. The operator “every” defines which pattern should match every SensorEvent that fulfils the conditions. The expression inside the first parenthesis contains the followed-by operator.

## 4. MACHINE LEARNING RECOGNITION STRATEGY

The Machine Learning module will be implemented as a distributed module of the platform able to detect anomalous network traffic.

The operating principle of the machine learning module is the periodical scheduling of a statistical procedure based on the principal component analysis (PCA) statistical method. This approach creates a network profile called Digital Signature of Network Segment using Flow Analysis (DSNSF) that denotes the predicted normal behaviour of a network traffic activity through historical data analysis thus eliminating the need for prior knowledge about the nature and properties of anomalies. This trait leads to some advantages: the possibility of discovering new and unforeseen types of anomalies and the detection of insider attacks making it difficult for an attacker to know with conviction what malicious action can be carried out without being detected by the system.

The main idea of PCA is to reduce the dimensionality of a data set comprised of many correlated variables, retaining as much as possible of the variation of the data set.

The DSNSF is a network profile which estimates the traffic behaviour of a network segment. Wherever the real traffic exceeds or goes below the DSNSF, that time interval will be classified as an anomalous event, which is the central idea of a profile-based anomaly detection approach.

The DSNSF is based on network traffic data represented by SNMP-MIB data. Historical data retrieved from the past will be used as the first baseline for the elaboration of a statistical procedure. This initial repository will be increased as data is collected.

SNMP is a popular protocol for network management. It collects information from different types of network devices, such as routers, switches, and hubs on an Internet Protocol (IP) network. SNMP offers powerful statistical information about what is happening in network devices through a database of management variables called the Management Information Base.

The module will receive the SNMP-MIB data from the interconnection layer and determine the presence of any anomalies in the observed time interval.

The module will be divided into two parts:

- Traffic Characterization: responsible for extracting quantitative attributes (bits/s, packets/s and number of flows/s) from a dataset flow containing historical data about the network segment activity and consequently generating the corresponding DSNSFs.
- Anomaly Detection: responsible for analysing the DSNSF confidence bands (or thresholds) along with the real time network traffic observed from a network device (switch).

The module will be implemented as software modules developed in R or Python languages. The core of the elaboration will be included within a generic software application and mechanisms for the model update (Traffic Characterization) will be achieved by using one of two approaches:

- a batch operation periodically scheduled by the application, and configured by means of a configuration file,
- a REST web service which will receive invocations from outside in order to schedule the computation on demand.

Communication among the machine learning and rule-based correlators will be implemented by means of either synchronous communication (e.g. web service interaction) or asynchronous communication based on the publish-subscribe paradigm.

## 5. RESISTO APPLICABILITY

This section will show two use cases applicable to the RESISTO project, with a focus on identifying threats by the Cyber / Physical Correlator module.

### 5.1. Cyber and physical protection of network

The use case "Cyber and physical protection of network and network elements mechanisms used by critical services that impact users" aims to identify potential threats through a correlation of events coming from a test-bed designed to include most of the real network infrastructure elements, replicating most of the core network functionalities and services.

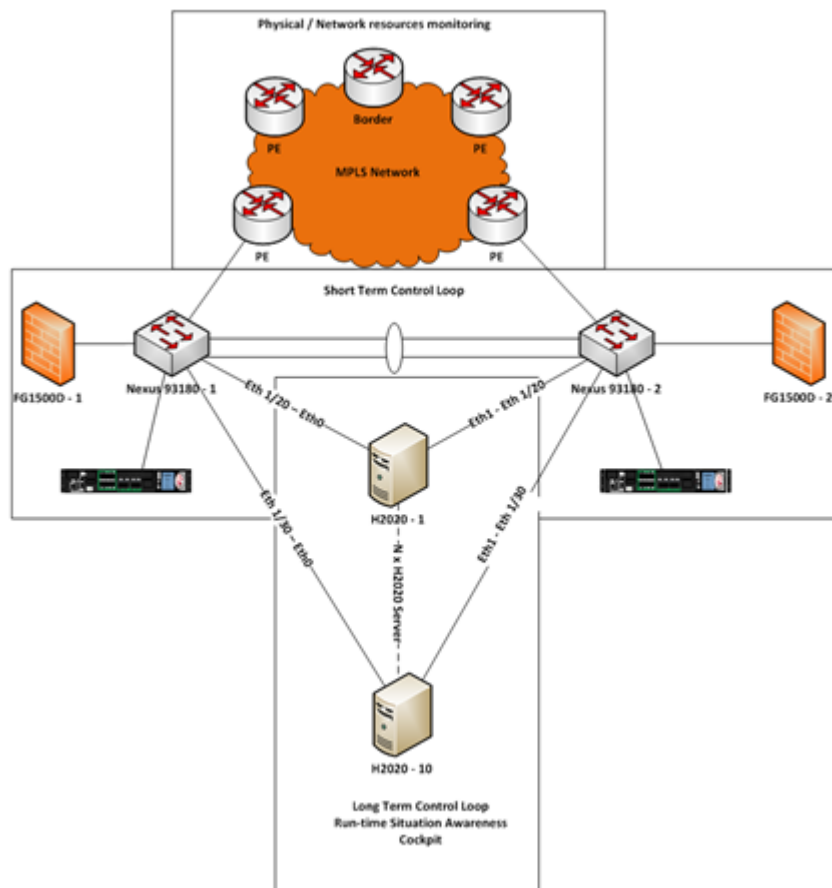


Figure 6 – Use case 1: Testbed - Logical design

The input data to the correlator are:

- IP/RAN/Core/Security network elements collected from generically named “sensors”.

Sensors will be implemented at all network boundaries, e.g. IDS/IPS, DoS mitigation platform, WAF.

Based on input data, the Correlator, in the Complex Event Processing module, can identify, through correlation rules, the following threats:

- DoS attack from inside (from a client) / DDoS attack from inside botnet (residential customers. B2B customers with large infrastructures)
- DDoS attack on border
- DDoS attack on peering point (in case of interconnection with other partners)
- Routing Table Poisoning on core network
- BGP hijacking
- External Network scanning, mapping and lateral movement
- Detecting Connectivity to botnets from internal network
- Power outage – UPS/Generator available

## 5.2. Identification of probability of service loss

The use case "Identification of probability of service loss" aims to identify a potential threat of loss of service using the Machine learning recognition module. The application of the scenario involves two phases:

- The training of a neural network, starting from the input data, in order to set a parameter that identifies the probability of loss of service.
- Recognition of the probability of loss of service through the execution of the trained network on real-time data.

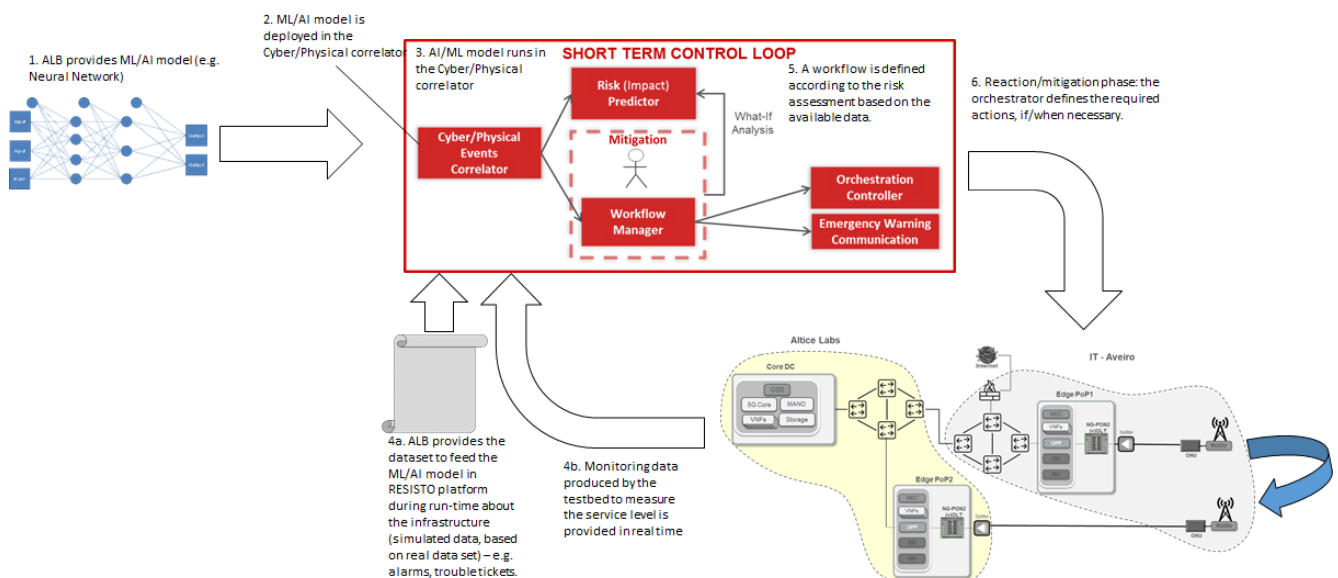


Figure 7 – Use case2: workflow

Figure 7 above shows the life cycle of the use case, in terms of:

- Data generation
- Training of the neural network
- Real-time recognition of the probability of loss of service

Operational workflow and identification of reaction actions after recognition.

## 6. CONCLUSION

RESISTO correlation process will be based on a holistic approach with the main objective of the component is to correlate events and identify potential threats. Events are acquired through an interconnection layer that handles the input / output messages. All the sources will transfer the log data into the raw messaging system, to be analyzed and correlated.

The correlation module is composed of two principal sub-components dedicated to identifying potential threats. The two components deal with analysis of data sources and automatic update of correlation rules. The Correlator operating workflow is composed of the following high level phases:

- Acquisition of data from external data sources through the interconnection layer
- Normalization of data in a standard format
- Transmission of the normalized data to the correlation engine and to the machine learning component
- Data correlation
- Data analysis using machine learning algorithms
- Updating of the correlation rules in:
  - o manual mode, by skilled operators
  - o automatic mode, by the machine learning component
- Threat identification
- Normalization of threat data using a standard format
- Forwarding of threat data to the interconnection layer.

## 7. REFERENCES

Apart from the references already denoted within the txt, the following ones were also considered:

INDEX	REFERENCE
1	RESISTO – Grant Agreement. Project Starting Date: May, 1 <sup>st</sup> 2018