The Brufence Project

WP 2: Design of mechanisms for the detection of cyber threats and attacks

Dimitrios Sisiaridis

Olivier Markowitch

innoviris.brussels
empowering research

sopra steria

ULB UNIVERSITÉ LIBRE DE BRUXELLES

Our focus

- Communication Systems
  - Collaboration Platforms
  - Managed File Transfer

Main targets

- We are looking mainly for:
  - Advanced Persistent Threats (APTs)
  - Zero-Day exploits (aka, zero-day attacks)
  - Distributed Denial of Service (DDoS) attacks

A look on current state-of-the-art approaches

- Proposals:

... *we should look inside logs after a breach took place, on auditing and non-repudiation* ...

... *we should analyse incoming traffic to detect known threats and attacks* ...

... *we should look primarly on authentication and authorization* ...

... *we should know our system better, continuously monitoring of everything* ...

Detection of threats and attacks

- Based on known signatures
  - misuse detection
  - high accuracy on known patterns
- Based on discovering patterns that deviate normal behaviour
  - completeness (but.. high number of false positives)
  - difficult to define what is 'normal' behaviour

Our proposal

- We need to combine the best techniques of each approach
  - define baseline profiles of normal system behaviour for the underlying network, users and services
  - increase performance by reducing the number of false positives utilizing known threats and attacks knowledge bases
- Try to be more proactive
  - it is about Risk Management
    - risk exists whenever there is at least a threat exploiting a system vulnerability
  - always, be prepared for the worst scenario
  - try to be a step ahead of the adversaries

What do we have?

- A plethora of logs, either structured, semi-structured or un-structured
  - Application logs, Web logs, DB logs, Filesystem logs, AAA logs, firewall logs, network raw traffic logs, syslogs, system event logs etc.
- Really, a lot of inputs
  - need to cope with... Big Data
  - and, with time-series analysis of temporal and spatial non-stationary data

Basic Axes in our research

- Behavioural Analytics
- Predictive Analytics
- Security Management
- Big Data analytics

Behavioural analysis of users and system entities

- Data aggregation from various sources
- Post breach analysis
  - in a constrained environment
    - Sandboxing and honeypots
- Lateral movement analysis
  - logging in from one account on one resource to another
  - sudden changes in behaviour are highly suspicious
  - Indications of sensitive data exfiltration and privilege elevation

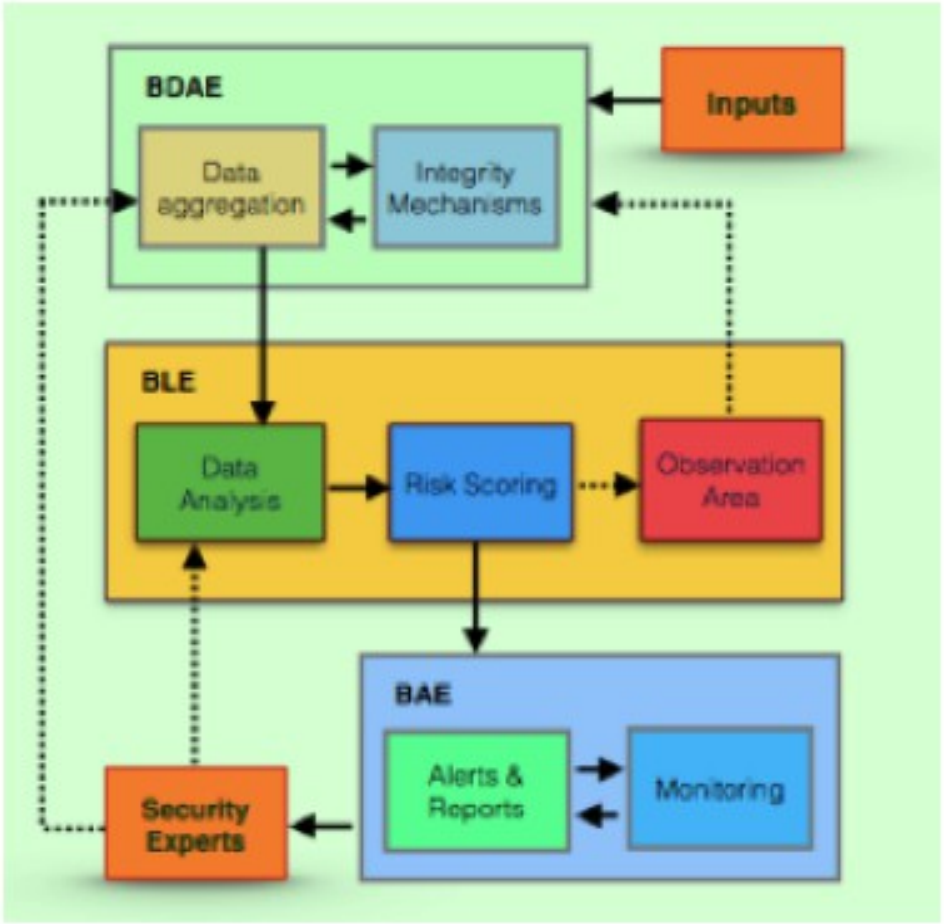
Security management

- Ensure data integrity, using:
  - public blockchains, as used in bitcoin industry
  - enhanced level of security of the employed algorithms in data analysis
- Security assessment of the basic building blocks
  - Using open source tools
- Verification of the deployed security protocols
  - using ProVerif, FDRII/CSP

Big Data Analytics

- Dynamic pattern matching
  - based on re-usable discovered patterns of interest
  - Complex Event Processing
- Machine Learning algorithms
  - based on outlier detection analysis
  - producing models of system behaviour

The proposed framework



Implementation steps

- Compute the behaviours of different entities within unlabeled or partially labeled datasets (un-supervised and semi-supervised outlier detection analysis)
- Present to the enterprise security analysts a small set of events or entities of high risk score (i.e. outliers)
- Collect analysts feedback (i.e. Labels of malicious or not events) through a UI engineering active learning techniques
- Apply supervised analysis using the feedback
- Use the supervised models in conjunction with semi-supervised and un-supervised models to predict threats and attacks
- The whole process in continuously repeated

Time Series Analysis

- On temporal and spatiotemporal data
- Activities/events are decomposed in time intervals on a basis either of a minute, hour, day, week, month
  - to maintain aggregations that can be used to satisfy real-time requirements
  - in terms of reduced computational effort

Performance measurements

- Metrics:
  - Improve attack detection rates
    - Best known detection rate: 86,8%
    - False positive rate: 4,4%
    - With 200 events daily labeled by the analysts
  - Reduce security noise
    - By reducing the number of alerts to the security analysts
- Results can be validated against labeled datasets provided by the enterprise, under a supervised analysis

The Data Acquisition Engine - BDAE

- The Pre-processing data module
  - Data cleaning process
  - Data normalization process
  - Data anonymisation and privacy-preserving processes
- The Data Integrity module
  - Using a public blockchain adapted from the bitcoin industry

The Learning Engine - BLE (I)

- The Exploratory Data Analysis module
  - Data preparation for ingestion / setup the pipeline
    - Using the Hadoop ecosystem
  - Data summary / data indexing (using uni- and multi-variate graphical and non-graphical methods)
    - Preliminary detection of false data interpretation or missing information
    - Checking for assumptions
    - Preliminary selection of appropriate models
    - Detection of relationships among attributes
    - Preliminary correlation and prediction inference

The Learning Engine - BLE (II)

- The Statistical Prediction Modeling module
  - Outlier detection analysis
    - Data correlation process
    - Complex Event Processing
    - Data classification process
    - Outlier scoring process
    - Formal Statistical Inference process

The Learning Engine - BLE (III)

- The Statistical Prediction Modeling module
  - Algorithms and Models
    - Supervised (labelled datasets)
      - using a combination of algorithms
    - Semi-supervised
      - enhanced with Active Learning
    - Un-supervised learning
      - quite common in the inputs
      - using a combination of algorithms in the absence of labels
        - e.g NLP (Natural Language Processing) and LSI (Latent Semantic Indexing) are utilized to extract weighted features of interest in un-structured data
      - algorithms trained on a large number of inputs

The Learning Engine - BLE (IV)

- Other continuous, iterative processes:
  - Results visualization / data indexing
  - Inference and data interpretation
    - Evaluation of models and algorithms
  - Challenging findings
    - Models and algorithms refinement
- Protection of the algorithms against integrity attacks
  - e.g. causative or exploratory attacks in the training phase

Cross-Module Processes (BDAE/BLE): Dynamic Pattern Matching

- Question:
  - Is there a knowledge base of threats and attacks to be used as a reference point?
- Answer:
  - Yes, the CAPEC and STIX databases of patterns for threats and attacks in cyber security
    - Based on the adaptation of the **kill chain model**, originally introduced by Lockheed Martin (USA)
      - accepted by OASIS, OWASP
      - supported by Cert-EU

The kill-chain model (I)

- An intelligence-driven, threat-focused approach to study intrusions from the adversaries perspective
- Originally inspired by the kill chain process, used by US military services to target and engage an adversary to create desired effects.
- It comprises of seven phases:
  - Reconnaissance
  - Weaponization
  - Delivery
  - Exploitation
  - Installation
  - Command and Control
  - Actions on Objectives

The kill-chain model (II)

- Each discrete phase of the intrusion is mapped to courses of action for the detection, mitigation and response to a threat or an attack
- Assumption: An adversary must progress successfully through each phase of the chain before it can achieve the desired objective
- Just one mitigation, implemented faster than the adversary evolves:
  - disrupts the chain
  - increases the cost the adversary should spend for a successful attack

The kill-chain model (III)

- The fundamental element is the **Indicator**
  - It corresponds to any piece of information that can describe a threat or an attack

- Indicators can be either:
  - Atomic: such as IP or email addresses
  - Computed: such as hash values or regular expressions
  - Behavioural: collections of computed and atomic indicators such as statements (i.e. sentences)

Implication of the kill chain model in the project (I)

- Detection of threats and attacks early in the chain, e.g. in the exploitation phase
  - By extending the scope of the detection:
    - to threats and attacks (threat-focused)
    - to system entities (baseline profiles of system normal behaviour)
  - to increase system's resiliency in terms of continuous risk management
  - to advance beyond the kill-chain model

Implication of the kill chain model in the project (II)

- Information security analytics leverages the discovery of Indicators in each phase of the chain (either atomic, computed or behavioural)
- An example: a zero-day attack that exploits a not-yet identified or mitigated system vulnerability
  - the adversary uses tools, tactics or infrastructure already employed in another campaign, as repeated or overlapping indicators:
    - detection of the attack can prioritize countermeasures increasing their effectiveness and performance

Implication of the kill chain model in the project (III)

- In case of a successful detection and mitigation:
  - Assumption: prior phases have already executed successfully by the adversary
  - Then, we are able to synthesize the remaining phases in order to understand adversary's intents, techniques and tactics
    - by identifying patterns and behaviours over time
      - using sandboxing and honeypots for observation
      - using our previous work on risk management utilizing:
        - Process calculi (lambda- and pi-calculus)
        - Godement calculus to express interactions between system entities

Implementation: why we chose python

- Existence of packages for tokenization in python, e.g. the nltk package
- STIX and CAPEC threat patterns specifications in XML and in python
- Spark implementations in scala, python and java

Datasets we use

- Public:
  - CSIC 2010 HTTP dataset
  - GureKDDcup datasets
  - CAIDA DDoS attacks 2007 dataset
  - UNB ISCX IDS 2012 dataset
  - Enron Email 2015 Dataset
  - Nuix EDRM Enron Email 2014 Dataset
- Private:
  - ULB's OwnCloud logs
  - **need for more 'real' datasets from MFTs and collaboration platforms**

BDAE: Data cleaning

- Logs are cleaned of either:
  - Extra characters that are generally meaningless for analyzing
    - e.g. word frequencies or meaningless patterns
  - Frequent occurring words that are meaningless for data analysis:
    - Using a stop-word-list

BDAE: Data normalization (I)

- Extract Indicators in logs of different data formats
  - They will allow us in later stages to detect Indicators of Compromise using statistical analysis and Complex Event Processing
- File Format conversions (bi-directional) using python scripts:
  - .pcap → .csv
  - .txt → .csv
  - .xml → .csv
  - .json → .csv
  - .avro → .json
  - .csv → .avsc
  - .csv → .avro
  - .avro → .avsc
  - .pst → .json
  - MIME messages → directories of .txt files

BDAE: Data normalization (II)

- Sniffers detect logs structure
  - e.g. look for the existence of headers
- Scanners are implemented for pattern matching using regular expressions
  - we have extended packages and modules for tokenization
    - by searching for all types of Indicators
- Compilers are used for optimizing Indicators pattern matching
- CAPEC and STIX threat pattern hierarchies are used for reducing false positives potentially produced by outlier detection analysis

Extracting Indicators

```
Headers: No#, Time, Source, Destination, Protocol, Length, Info

Values: ['1', '0.000000', '192.168.2.111', '69.147.125.65', 'TCP', '60', '1555  >  80 [FIN, ACK]
    Seq=1 Ack=1 Win=63839 Len=0']
-------------------------------
1. a single-valued attribute: 1
2. a single-valued attribute: 0.000000
3. a single-valued attribute: 192.168.2.111
4. a single-valued attribute: 69.147.125.65
5. a single-valued attribute: TCP
6. a single-valued attribute: 60
7. a multi-valued  attribute: 1555  >  80 [FIN, ACK] Seq=1 Ack=1 Win=63839 Len=0
   ---attribute features/values---
   1555
   80
   [FIN, ACK]
   Seq=1
   Ack=1
   Win=63839
   Len=0
```

FIN

ACK

new [attrib, value] pairs

BDAE: Data anonymisation

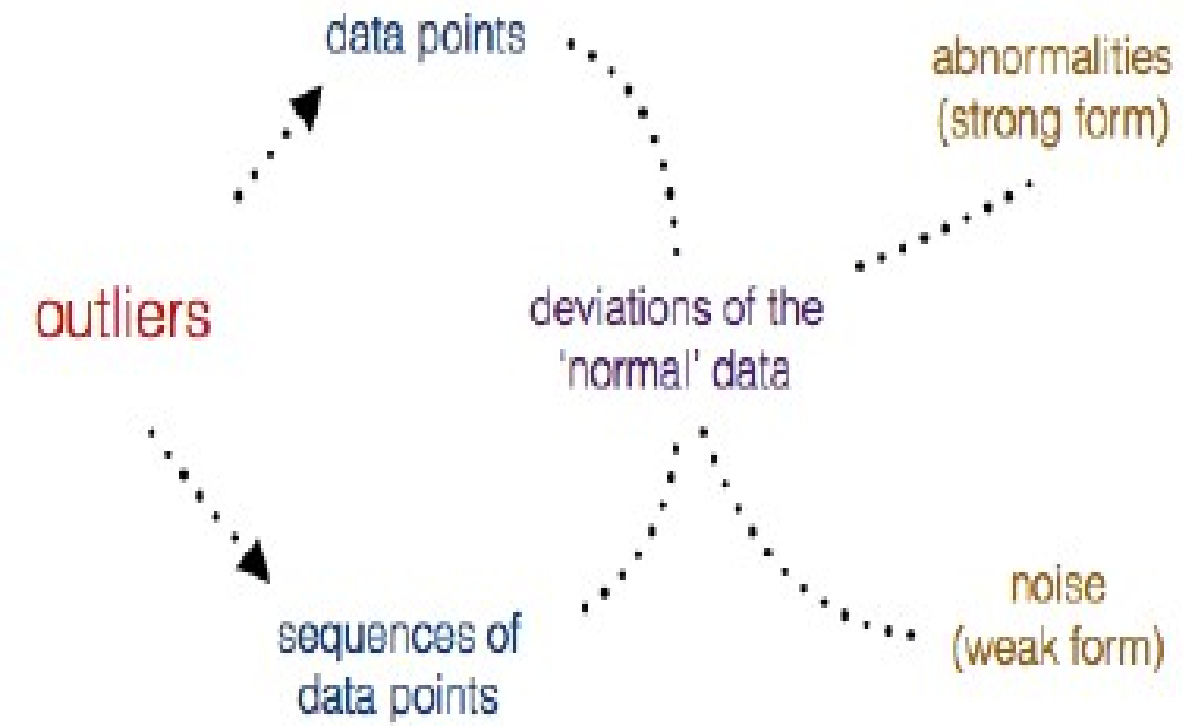
- Public datasets used were already anonymized
- Anonymisation techniques against re-identification on the OwnCloud logs:
  - Non-probabilistic:
    - anonymize their traces (i.e. IP addresses), tcp ports, payload, checksum and IP options, using CryptoPan, that uses the Rijndael cipher, and tcpanon
    - tokenization / randomization using mapping tables
  - Aggregation techniques
    - data sampling
    - perturbation techniques by adding noise to sensitive data
  - We are currently working on employing statistics and group-based anonymisation utilizing location privacy techniques using:
    - resampling data by using probability density functions

BLE: Current implementation of the pipeline for Big Data Analytics with the Hadoop ecosystem

- Normalized datasets are uploaded onto HDFS
- Internal and external tables are created in HIVE for quering and preliminary data correllation and time-series analysis
  - Tables are indexed using Impala
  - Data are imported to the indexed collection using:
    - Flume
    - Morphlines
  - We are examining alternatively the use of Kafka and Cassandra, as the message broker and the BigData storage option, respectively
- Streaming analysis will be engineered using Spark streaming

BLE: Outlier Detection Analysis for threat and attack detection

- types of outliers



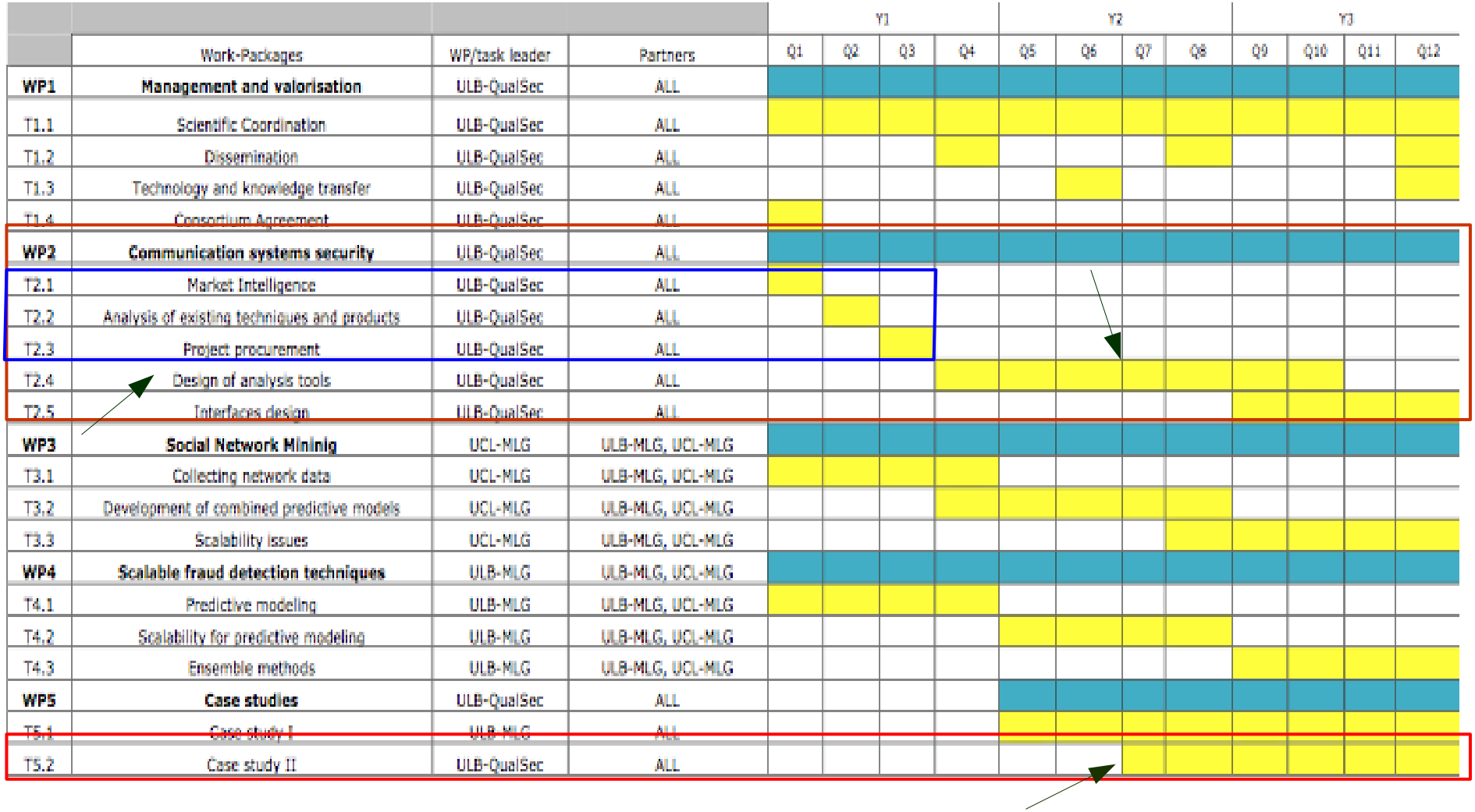
BLE: Data classification - Preliminary selection of the appropriate models

classifier	characteristics
Bayes classifier	the posterior probability is inferred in terms of the likelihood, prior probability and evidence
proximity-based classifiers	the number of k-nearest neighbours for each class is multiplied with the its corresponding cost; the majority class is chosen after the weighting process
rule-based classifiers	a rule relates a condition to a class label; frequent-pattern mining algorithms determine the relevant rules at a given level of support and confidence
decision trees	training data are recursively partitioned using entropy measures; instances of different classes are successively separated out at lower levels of the tree
SVM classifiers	hyperplanes optimally separate classes in order to minimise the expected error

BLE: Outlier scoring - Preliminary selection of the appropriate models

models for outlier scoring	characteristics
probabilistic	outliers are defined by the relative positions of the data values with respect to each other
proximity-based	a data point is defined as an outlier, if its locality (or proximity) is sparsely populated
linear	the residual distance of a data point to a lower-dimensional representation of the data
information theoretic	frequent pattern mining, histograms and spectral methods
frequency-based	measurements of the relative frequency of the comparison unit in the training sequences
ensemble methods	sequential or independent ensembles with respect to the number of dimensions
temporal	outliers defined using either the distance from previous or forecasted values in time-series

Brufence: Project Management

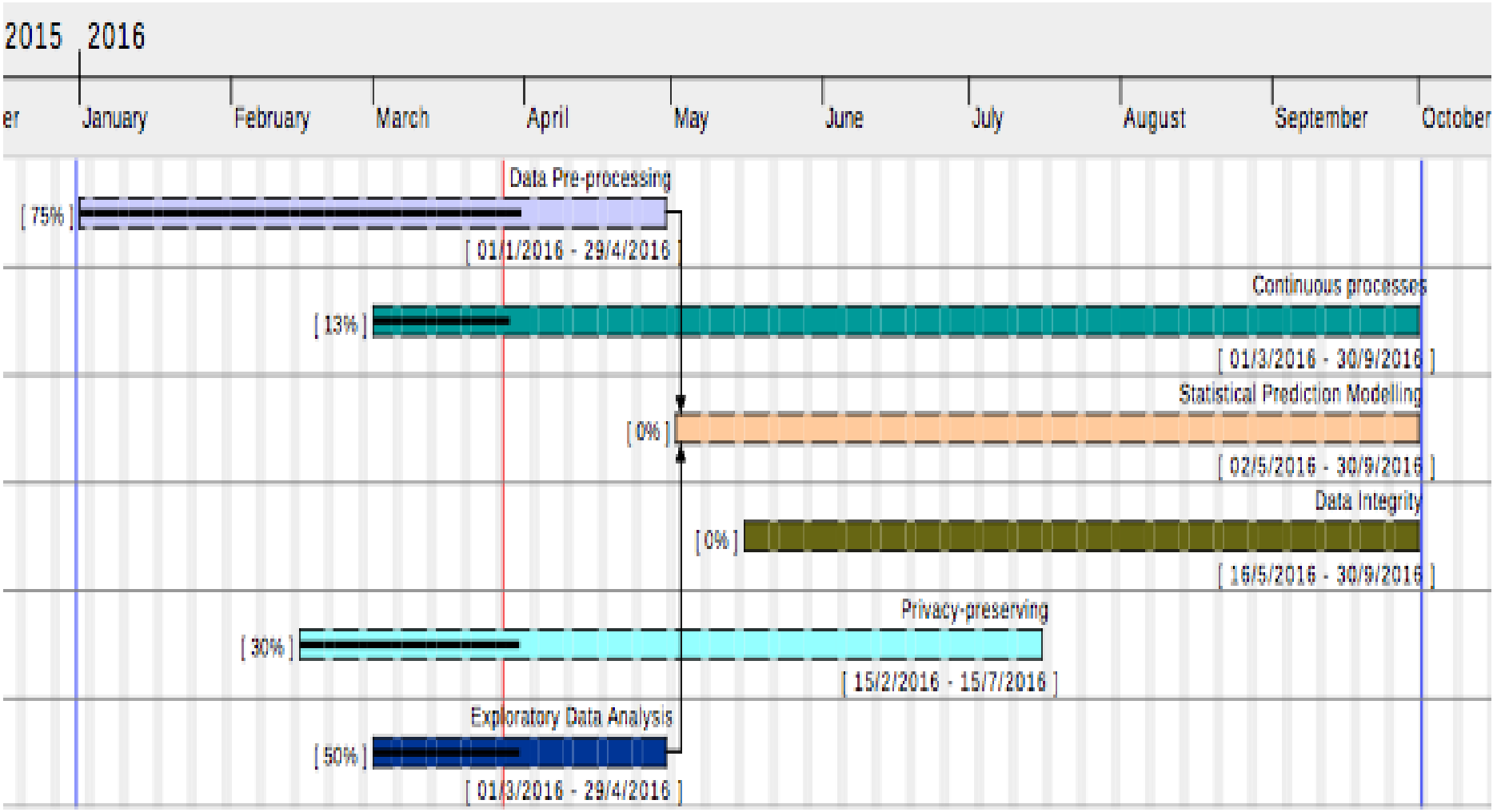


	Work-Packages	WP/task leader	Partners	Y1 Q1	Q2	Q3	Q4	Y2 Q5	Q6	Q7	Q8	Y3 Q9	Q10	Q11	Q12
WP1	**Management and valorisation**	ULB-QualSec	ALL												
T1.1	Scientific Coordination	ULB-QualSec	ALL												
T1.2	Dissemination	ULB-QualSec	ALL												
T1.3	Technology and knowledge transfer	ULB-QualSec	ALL												
T1.4	Consortium Agreement	ULB-QualSec	ALL												
WP2	**Communication systems security**	ULB-QualSec	ALL												
T2.1	Market Intelligence	ULB-QualSec	ALL												
T2.2	Analysis of existing techniques and products	ULB-QualSec	ALL												
T2.3	Project procurement	ULB-QualSec	ALL												
T2.4	Design of analysis tools	ULB-QualSec	ALL												
T2.5	Interfaces design	ULB-QualSec	ALL												
WP3	**Social Network Mininig**	UCL-MLG	ULB-MLG, UCL-MLG												
T3.1	Collecting network data	UCL-MLG	ULB-MLG, UCL-MLG												
T3.2	Development of combined predictive models	UCL-MLG	ULB-MLG, UCL-MLG												
T3.3	Scalability issues	UCL-MLG	ULB-MLG, UCL-MLG												
WP4	**Scalable fraud detection techniques**	ULB-MLG	ULB-MLG, UCL-MLG												
T4.1	Predictive modeling	ULB-MLG	ULB-MLG, UCL-MLG												
T4.2	Scalability for predictive modeling	ULB-MLG	ULB-MLG, UCL-MLG												
T4.3	Ensemble methods	ULB-MLG	ULB-MLG, UCL-MLG												
WP5	**Case studies**	ULB-QualSec	ALL												
T5.1	Case study I	ULB-MLG	ALL												
T5.2	Case study II	ULB-QualSec	ALL												

Next steps

- finalize pattern matching implementation
- continue work on:
  - the deployment of machine learning algorithms for outlier detection analysis in terms of BigData analytics
    - including implementations in public clouds, as AWS
  - privacy-preserving algorithms
  - protocols for data integrity, using:
    - public blockchains
    - enhanced security algorithms in data analysis e.g. to protect against data contamination in training phase
- collaborate with the Spices project in Complex Event Processing

WP2: Project Management



Thank you