SEVENTH FRAMEWORK PROGRAMME

Information & Communication Technologies
ICT

Cooperation Programme

NECOMA

Nippon-European Cyberdefense-Oriented Multilayer threat Analysis[†]

Deliverable D3.3: Security Information Exchange - Results

Abstract: This report describes the prototype implementation of the security information mechanism defined in D3.2 and produced in T3.2. The deliverable will also document experience gathered during the development phase.

Contractual Date of Delivery	31 May 2015
Actual Date of Delivery	31 May 2015
Deliverable Dissemination Level	Public
Editors	Dawid Machnicki (ATOS) Youki Kadobayashi (NAIST)
Contributors	All *NECOMA* partners

[†] The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7-ICT-2013-EU-Japan) under grant agreement n° 608533 and the Strategic International Collaborative R&D Promotion Project of the Ministry of Internal Affairs and Communication, Japan.

The *NECOMA* consortium consists of:

Institut Mines-Telecom	Coordinator	France
ATOS SPAIN SA	Principal Contractor	Spain
FORTH-ICS	Principal Contractor	Greece
NASK	Principal Contractor	Poland
6CURE SAS	Principal Contractor	France
Nara Institute of Science and Technology	Coordinator	Japan
IIJ - Innovation Institute	Principal Contractor	Japan
National Institute of Informatics	Principal Contractor	Japan
Keio University	Principal Contractor	Japan
The University of Tokyo	Principal Contractor	Japan

Contents

List of Figures

1 Introduction

This deliverable concludes Task 3.2 *Information exchange mechanisms* which aimed at providing means for easy communication and dissemination of threat data and knowledge, addressing various kinds of users, including security operators, data analysts and non-technical users.

This document is a report containing results and gathered experiences that were obtained during the implementation and application of the mechanisms presented in Deliverable D3.2: Security Information Exchange - Design, focusing on the implementation and observations, that followed the implementation, of machine-to-human and machine-to-machine interfaces foreseen as the NECOMA system internal and external interfaces.

The aim of this document is to provide an overview and conclusions that were deduced based on the usage of the implemented mechanism within NECOMA. Primarily this document will describe how the implementation of particular mechanisms was carried out, followed by explanation of the processes that involved the interfaces. Lastly a report, per interface, will be provided. It will describe how the interface performed, how they would satisfy the needs of particular target groups, and lastly, how they improved the processes involved.

In addition the document will provide all the experience gathered, throughout the project, on implementation and usage of different communication interfaces by various consortium members.

1.1 Data Storage

Although covered in deliverable D3.2: Security Information Exchange - Design, the data storage was not implemented due to the complexity and extent of requirements that were posed by security and privacy restrictions. It was concluded that this subject was extremely broad to be covered within the scope of the NECOMA project, although the design itself might serve as a recommendation to build a distributed data storage that would successfully incorporate various datasets in a secure way.

For the time being the datasets remain decoupled and their security and accessibility lays within the responsibility of the owners.

1.2 Structure of the Document

In order to transparently show the results, this document has been divided into three main chapters:

- Machine-to-Human Interfaces
- Bridging Machine-to-Machine Interfaces
- Additional Experience Gathered

The second chapter reports the usage of the NECOMAtter prototype, which serves as the main machine-to-human interface in NECOMA.

Chapter three will describe the utilisation and experience with the n6 interface that runs on top of various datasets and knowledge bases serving as a machine-to-machine interface.

Chapter four will comment on the experience and conclusions the consortium members derived from the usage of various communication means within the NECOMA context.

2 Machine-to-Human interfaces

This chapter will focus exclusively on the implementation and documentation of NECOMAtter, which is a system designed and established according to the motivation and challenges posed in relation to effective communication and dissemination of multilayer threat analysis information. We first explain specific challenges taken with the design of NECOMAtter, then present the internal architecture of the software with the defined interfaces to the users of NECOMAtter. The last section of this chapter provides an extensive view on the usage of NECOMAtter as part of the threat information analysis pipelining, data enrichment and threat mitigation.

2.1 Design Overview

NECOMAtter is a tool for human security analyst, various data sources, and network equipments, intended to ultimately facilitate man-machine collaboration. The followings summarize our motivation and challenges for the design of NECOMAtter:

1. **large amount of data**
   The data exchange must be efficient despite the large amount of data.
2. **diverse data**
   The number of data types can be large if we analyse multiple kinds of datasets.
3. **various stakeholders with different expertise**
   The information providers and users have diverse background and priorities, with different domain of expertise.
4. **ad-hoc collaboration is key**
   Collaboration among information providers and users cannot be predicted and should be formed spontaneously.

5. **improvised defences against improvised attacks**
   As attack campaigns are improvised, countermeasures should also be enabled and combined as necessary as the situation develops.

We took special care for those challenges by taking following approaches in the implementation of NECOMAtter.

A-1. **linkage with external data sources**
Since we assumed that the size of information tends to be big, it is not feasible to store all information in the system. We adopted hybrid approach, storing the summary of the information, meanwhile showing some pointers, e.g., URL, for a user who requests detailed information.

A-2. **individual agents for dealing with diverse kinds of data**
We introduce agent, called BOT for short, that generates messages based on the automated analysis of datasets based on specific set of algorithms.

A-3. **participation-oriented, data centric platform**
Analysts can find the useful information and exchange security information using NECOMAtter. A normal tweet can be read by anyone who is joining in NECOMAtter, and a user can retweet and comment to any tweet.

A-4. **mechanisms for ad-hoc collaboration as in social networks**
Following the chain of trust between users, the security information are exchanged and analyzed rapidly. It is a new *modus operandi* of exchanging and analyzing security information. We are also inspired by the concept of timeline, resembling a real-time list, as in tweets on Twitter. In the case of NECOMAtter, there are also various types of cyber threat information, and each NECOMAtter users have different demands for the information.

A-5. **improvise defenses from flat presentations of cross-layer information**
NECOMAtter should offer to improvise defenses, for instance by taking following steps: 1) NECOMAtter BOT finds indications or activities of attacks, 2) Operator find the notification and confirm the activities, and 3) group individual notifications with NECOMAtome, in order to take corresponding actions.

2.2 Core part of NECOMAtter

NECOMAtter is written in Python with Neo4J, a graph-database implementation. All the implemented source code is available on the github [1].

[1] https://github.com/necoma/NECOMAtter

We used a python-based web framework, Flask [2] as a web-based application. The offered interfaces are two-fold: 1) web-based UI for human interaction and 2) REST API for the machine-to-machine interaction, especially used by our BOT programs. The web-based UI is used for a typical web application with HTML or JavaScript while REST API uses JSON[1] for the data transport.

Figure 2.1 illustrates the structure of the python application of NECOMAtter. A BOT user in the figure interacts with the Flask module of NECOMAtter while a user of Web UI accesses resources through the Flask module as well as the one exposed by HTML and JavaScript. In the other words, Web UI is written in HTML and JavaScript and obtains the information provided by NECOMAtter via REST API of the Flask module.

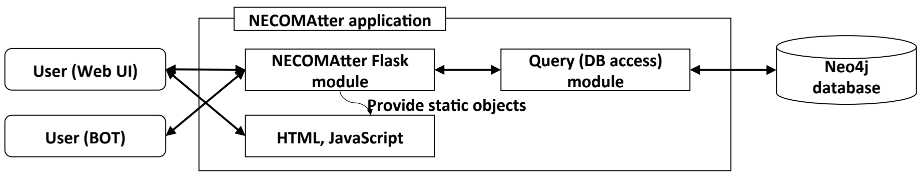


Figure 2.1: A class diagram of NECOMAtter.

The followings explain all of the entities that interact with NECOMAtter.

- User
  All the operations of NECOMAtter are user actions. The operations here are mainly related to accessing the API, which will be discussed later.

- tweet
  A tweet is an instance of each post by a user, human or a BOT, and recorded by NECOMAtter. A tweet is written in string of characters and treated as plain text for the REST API, but it can also be written in an encoded format like a link for a URL or formatted text written in markdown format for pretty-printing on the Web UI.
  Each tweet is assigned an identifier managed by NECOMAtter and can be retrieved with the relevant tweets for the other purposes.

- reply
  A reply is a tweet indicated as a reply message to the original tweet.

[2] http://flask.pocoo.org/

An URL, which points to a tweet, can list all of children tweets replying to the parent tweet.

- re-tweet
  A tweet can be regenerated by a user who wants to enlarge the audience of the particular tweet. This re-tweet resembles the forward operation of email client.

- timeline
  A timeline is a view provided by NECOMAtter Web UI listing tweets. A user can follow multiple users of interest and can obtain the tweets generated by the following users. Right after an account creation, a user has only a single follower who is himself, and the timeline only presents his own tweets and re-tweets.

2.2.1 API

All of internal information used by NECOMAtter are stored in the internal database. In order to obtain these information, NECOMAtter accepts client requests via HTTP or HTTPS. A request assigns a particular URL and processed by the data described in JSON format.

For instance, for a tweet function, an URL /post.json is assigned and NECOMAtter processes with the following HTTP POST message as a JSON input. As a result, the text part of "tweet TEXT" will be recorded as a tweet of a particular user.

```
{"user_name": "YOUR ACCOUNT NAME", "api_key": "YOUR AIPKEY",
 "text": "tweet TEXT"}
```

The followings briefly describe the other APIs implemented in NECOMAtter.

- /stream/regexp.json
  The API is used to observe newly generated tweets coming to NECOMAtter. Different from usual HTTP request, NECOMAtter server will not disconnect TCP connections for this type of request, then outputs a single line of tweet when user inputs with the regular expression matches tweets.

- /user/<user_name>.json
  An API to obtain tweets for a user named 'user_name'.

- /tweet/<tweet_id>.json
  An API to obtain a single tweet with the identifier 'tweet_id'.

- /tweet/<tweet_id>_tree.json
  An API to obtain a tweet and its relevant tweets in a reply tree.

- /timeline/<user_name>.json
  An API to obtain a timeline of a single user 'user_name'.

There are a couple of other available APIs implemented for Web UI operations.

2.2.2 NECOMAtome

NECOMAtome is used to summarize a set of tweets with a single entry accessible by a dedicated URL. As timeline is often filled with numerous random tweets, important information might be overlooked. The summary of a group of tweets contributes to the visibility of important tweets enriched by a user through an URL.

2.3 Pipelining analysis, data enrichment and mitigation

As we described, we develop and deploy NECOMAtter system for security information exchange. The data collection and analysis are performed by MATATABI system which we have also developed, then the analysis modules on MATATABI post the analysis results to NECOMAtter. Other analysis modules and operators watch the posted results on the timeline in NECOMAtter, which they use for further analysis or decision of launching the mitigation mechanisms.

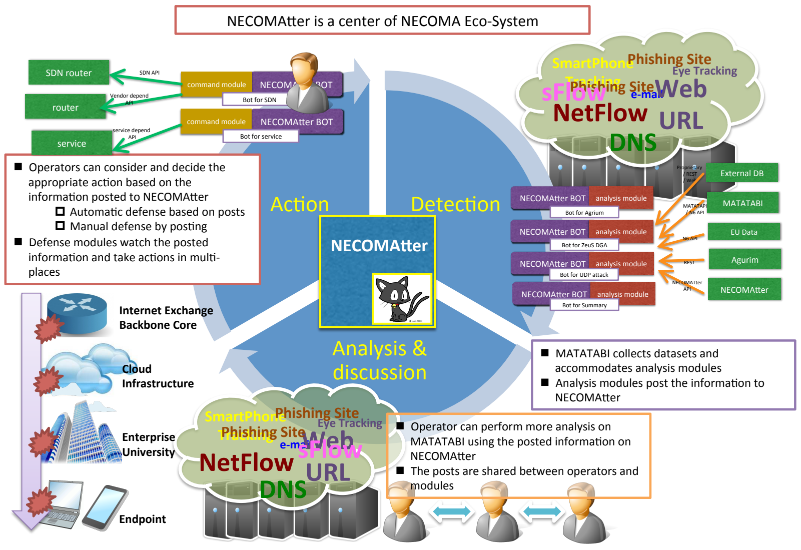
The entire system of NECOMA Project, which includes MATATABI, NECOMAtter, and DORANECO, can form a pipeline of data acquisition, analysis and mitigation; it can also be considered to be an ecosystem as shown in Figure 2.2.

The detection part collects datasets and stores the datasets into MATATABI system. The analysis modules running on MATATABI system detected unusual behaviors from the datasets and post the results to NECOMAtter system.

Data collecting modules collect data from various sources and convert the collected data into the appropriate format, then store them into the MATATABI. Subsequently, analysis modules called "NECOMAtter BOTs" perform analysis using the stored data and post the analysis results to NECOMAtter. The series of the tasks is running on a routine basis. This is the first part of NECOMA ecosystem.

The list below describes the BOT programs that generate tweets by means other programs such as threat analysis modules, queries to an external data source, or interact with other BOT programs:

- **RSS fetch BOT**
  The RSS fetch BOT collects the publicly available information of var-



- Operators can consider and decide the appropriate action based on the information posted to NECOMAtter
  - Automatic defense based on posts
  - Manual defense by posting
- Defense modules watch the posted information and take actions in multi-places

- MATATABI collects datasets and accommodates analysis modules
- Analysis modules post the information to NECOMAtter

- Operator can perform more analysis on MATATABI using the posted information on NECOMAtter
- The posts are shared between operators and modules

Figure 2.2: NECOMA Ecosystem

ious web sites via RDF Site Summary (RSS). For instance, a vulnerability information published by security companies can be tweeted to NECOMAtter.

- **UDP Fragment Volume BOT**
  The module analyzes the ratio of fragmented UDP packet volume on sFlow[4] datasets. The fragmented packets could be used for DNS cache poisoning attack. If the fragmented ratio is increased compared with other days, we could say that a suspicious behaviour is caused on the networks.

- **Phishing top 10 BOT**
  Top ten-lists are useful to grasp trends, and hence there are various top-ten lists in the context of cyber threat information. Within the NECOMA project, we aim at providing daily trends about phishing sites. "Phishing top 10 IP" BOT is designed for tweeting top-ten IP addresses of phishing servers that hosted phishing sites. "Phishing top 10 AS" BOT tweet the AS numbers where phishing servers belonged. Currently, we collect phishing sites' URL from external information sources such as PhishTank and Council of Anti-Phishing Japan, and

store these information at MATATABI, then extract top-ten IP addresses and AS numbers per day[3].

- **ZeuS DGA BOT**
  The module looks for DNS queries used by a botnet programs of Zeus-DGA with a given regular expression to match in our DNS query log dataset. In particular, if the dataset includes an entry like the following, the module tweets an entry to NECOMAtter[4].

  ```
  oq2cuaacaaknsmqaag46adosaqaabjafstfpeioqwoozaaaaahwbiaa3aaaabkp.
   ayldacaaiaaaqamy6sx5lss6umwaw7dt5rru3f3azshqcz7y.a.j.e5.sk.
  ```

- **Agurim report BOT**
  Agurim[2] report BOT reports a tweet in NECOMAtter when a certain condition is met in the available information in Agurim[5]: IF a volume of flows exceeds a pre-defined threshold AND if these flows are not observed in the past AND aggregated network prefix is longer than pre-defined prefix length. Since this BOT program is periodically running in 1 hour interval, the module can detect the flow anomaly within 1 hour delay.

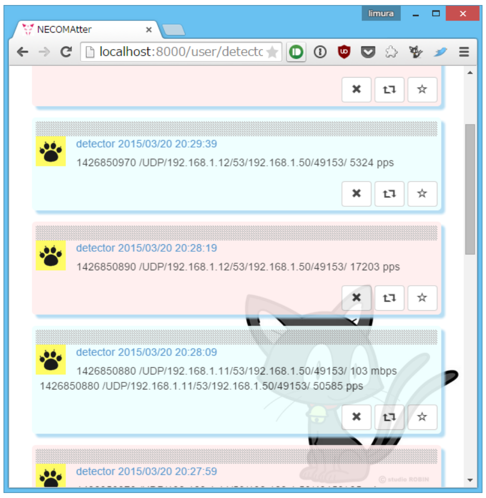
- **Twitter watcher BOT**
  The twitter watcher BOT reports top 10 entries obtained from a twitter query to which search with the particular hashtags such as #security and #vulnerabilities from one-day history[6].

- **Flow Watcher BOT**
  The flow watcher BOT reports most frequent observed flows of DNS traffic (with port number 53) within a specific time[7].

- **Phishing analysis BOT**
  Phishing analysis BOT is a NECOMAtter BOT to provide information about whether the website seemed to be phishing or not. To launch BOT, a NECOMAtter user sends a mention with the specified URL, such as @is_a_phish: `http://www.necoma-project.eu/` The BOT analyzes the URL with feature vectors, such as lifetime of domain, popularity of content, number of dots, thus calculating the phishing likelihood of the site, then outputs the likelihood in a reply to the original tweet.

[3]https://necomatter.necoma-project.jp/user/phishing_top10as, https://necomatter.necoma-project.jp/user/phishing_top10ip
[4]https://necomatter.necoma-project.jp/user/ZeuSDGA
[5]https://github.com/necoma/aguri2/
[6]https://necomatter.necoma-project.jp/user/twitter%20watcher
[7]https://necomatter.necoma-project.jp/user/FlowWatcher

- **DNS amplification attack mitigation BOT**
  The mitigation BOT works on a controller of mitigation mechanisms. The BOT feeds NECOMAtter timeline of DDoS detection BOT which is developed in WP2 tasks. When the BOT received a post which include information of DNS amplification attack[3], the BOT initiates a mitigation on the mechanism based on IP address of the attack, thus contributing to autonomic defense.



detector 2015/03/20 20:29:39
1426850970 /UDP/192.168.1.12/53/192.168.1.50/49153/ 5324 pps

detector 2015/03/20 20:28:19
1426850890 /UDP/192.168.1.12/53/192.168.1.50/49153/ 17203 pps

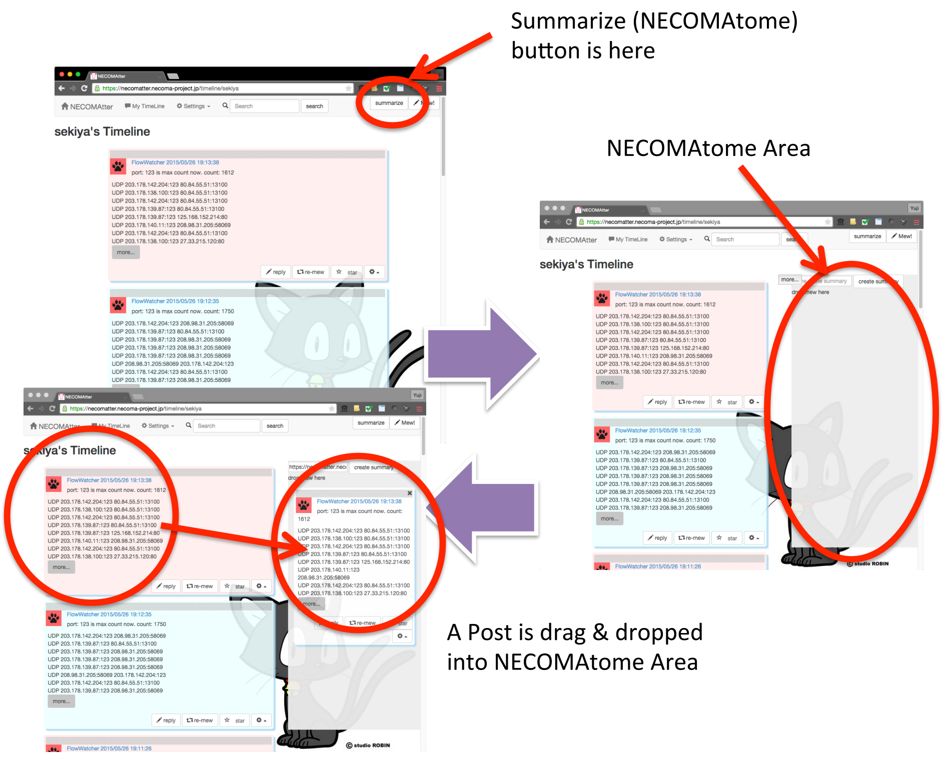
detector 2015/03/20 20:28:09
1426850880 /UDP/192.168.1.11/53/192.168.1.50/49153/ 103 mbps
1426850880 /UDP/192.168.1.11/53/192.168.1.50/49153/ 50585 pps

detector 2015/03/20 20:27:59

Figure 2.3: Tweets by DNS amp BOT.

The second part of NECOMA ecosystem is analysis and discussion on NECOMAtter. Based on the post (called tweet in NECOMAtter), operators share the information of trends, attacks, and malicious events. As shown in Figure 2.4, if an operator finds interesting relevant posts in NECOMAtter, he or she may copy and paste the posts to the work area (called NECOMAtome area in NECOMAtter), and try to perform more analysis using MATATABI system in order to investigate the traffic to/from their own organization. If needed, the operator can post a question to another timeline for all users in NECOMAtter, and generate discussion regarding the posted results or analysis results on NECOMAtter. As a result, an operator collect the related posts and make the summary page of the interested issue, then he or she can make discussion based on the summary (NECOMAtome) page as shown in Figure 2.5.

The last part of NECOMA ecosystem is launching appropriate action against incidents. If the operator find incidents or malicious events in the analysis results, the operator can launch the countermeasure methods, such as filtering, isolation, or mitigation through NECOMAtter posts. When the countermeasure BOTs find the launching commands on NECOMAtter posted



```
FlowWatcher 2015/05/26 19:13:38
port: 123 is max count now. count: 1612
UDP 203.178.142.204:123 80.84.55.51:13100
UDP 203.178.138.100:123 80.84.55.51:13100
UDP 203.178.142.204:123 80.84.55.51:13100
UDP 203.178.139.87:123 80.84.55.51:13100
UDP 203.178.139.87:123 125.168.152.214:80
UDP 203.178.140.11:123 208.98.31.205:58069
UDP 203.178.142.204:123 80.84.55.51:13100
UDP 203.178.138.100:123 27.33.215.120:80

FlowWatcher 2015/05/26 19:12:35
port: 123 is max count now. count: 1750
UDP 203.178.142.204:123 208.98.31.205:58069
UDP 203.178.139.87:123 80.84.55.51:13100
UDP 203.178.139.87:123 208.98.31.205:58069
UDP 203.178.139.87:123 208.98.31.205:58069
UDP 203.178.139.87:123 208.98.31.205:58069
UDP 208.98.31.205:58069 203.178.142.204:123
UDP 203.178.142.204:123 80.84.55.51:13100
UDP 203.178.139.87:123 208.98.31.205:58069

FlowWatcher 2015/05/26 19:11:26
```

Figure 2.4: Screen shot of NECOMAtome

by operators or another modules, it take the countermeasure action at the specified points.

After the countermeasure action was taken, the operators and analysis modules watch the analysis results posted on NECOMAtter whether the action was effective or not. If the attacks or incidents are protected by the action, they continue to watch the results. If the action doesn't work well, they will take conduct further analysis and take next action.

These series of the three steps are effectively forming a NECOMA ecosystem, and DORANECO system is developed for operators to make easy for the series. As a proof of the concept, Figure 2.6 shows the number of posts by modules and operators, respectively. The blue lines shows the posts from machines (BOTs) and red lines show the posts from humans.

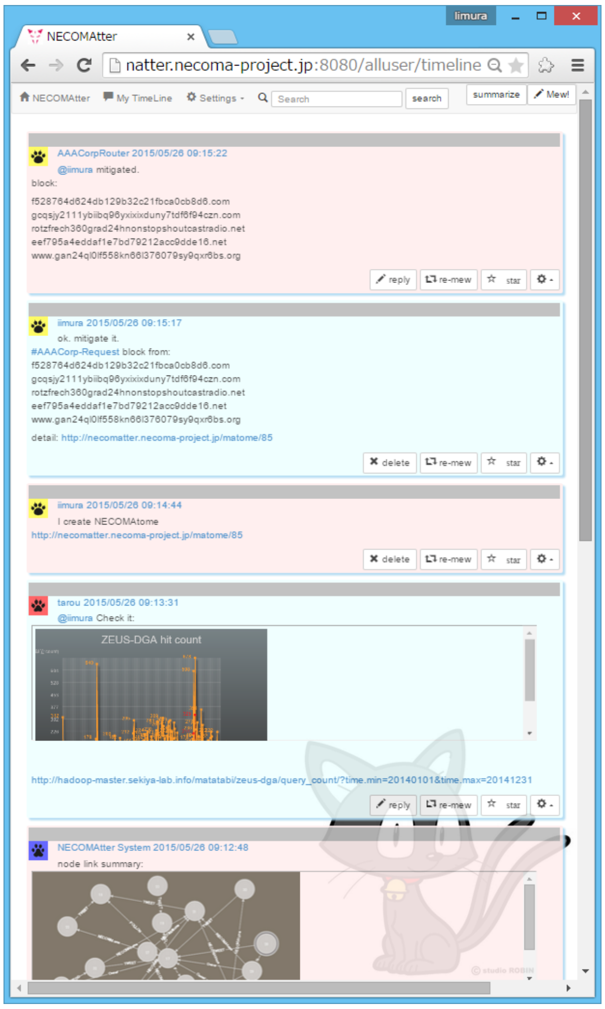
Figure 2.7 shows the three patterns of interaction, human to machine, machine to machine, and human to human.

The human to machine interaction is performed by (1) sending an analysis query to MATATABI system and (2) posting the rules of filtering or mitigation to action modules. All of the interaction in NECOMA ecosystem except for the confidential matter are exchanges through NECOMAtter system.

NECOMAtter system works like a messaging queue and shared memory in the ecosystem.

The machine to machine interaction is performed between analysis BOT and action BOT. When analysis BOT reports the result to NECOMAtter and the result including the command launching the paricular action BOT or the malicious behavior of the specific host, the action BOT find the post and take an appropriate action based on the information included in the post.

As for human to human interaction which is described in the previous part, NECOMAtome and discussion on the timeline facilitates interactions between humans.



AAACorpRouter 2015/05/26 09:15:22
@iimura mitigated.
block:
f528764d624db129b32c21fbca0cb8d6.com
gcqsjy2111ybiibq96yxixixduny7tdf6f94czn.com
rotzfrech360grad24hnonstopshoutcastradio.net
eef795a4eddaf1e7bd79212acc9dde16.net
www.gan24ql0lf558kn66l376079sy9qxr6bs.org

iimura 2015/05/26 09:15:17
ok. mitigate it.
#AAACorp-Request block from:
f528764d624db129b32c21fbca0cb8d6.com
gcqsjy2111ybiibq96yxixixduny7tdf6f94czn.com
rotzfrech360grad24hnonstopshoutcastradio.net
eef795a4eddaf1e7bd79212acc9dde16.net
www.gan24ql0lf558kn66l376079sy9qxr6bs.org
detail: http://necomatter.necoma-project.jp/matome/85

iimura 2015/05/26 09:14:44
I create NECOMAtome
http://necomatter.necoma-project.jp/matome/85

tarou 2015/05/26 09:13:31
@iimura Check it:
http://hadoop-master.sekiya-lab.info/matatabi/zeus-dga/query_count/?time.min=20140101&time.max=20141231

NECOMAtter System 2015/05/26 09:12:48
node link summary:

Figure 2.5: Discussion based on NECOMAtome

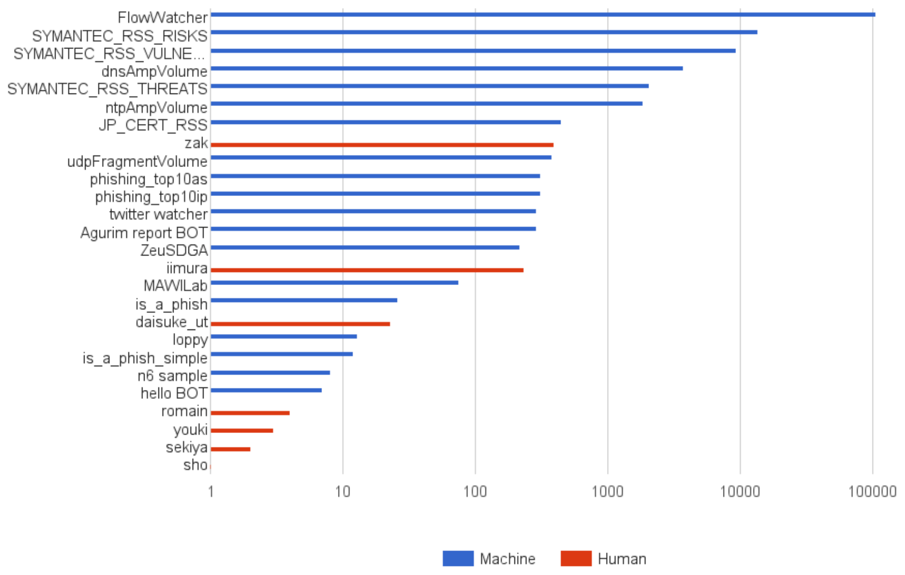


Figure 2.6: The number of posts on NECOMAtter

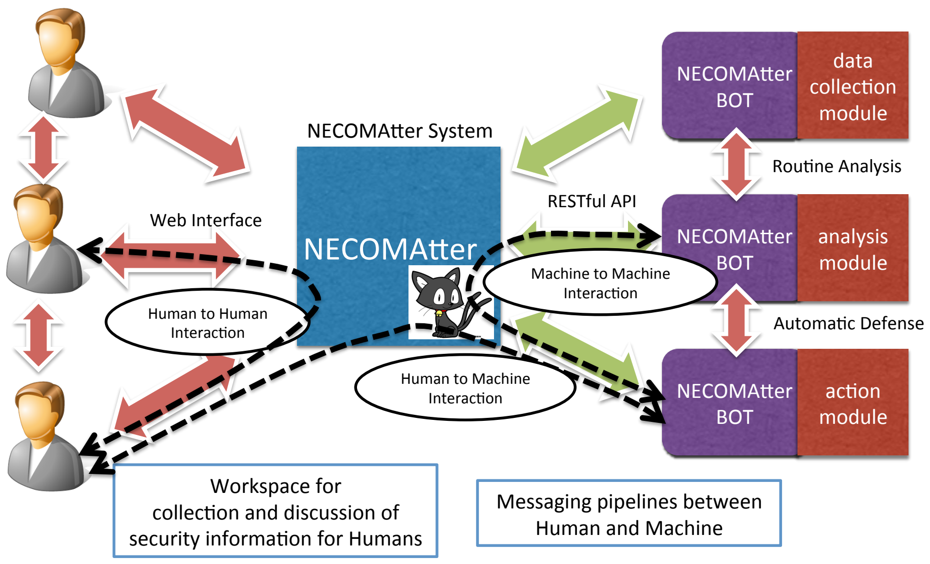

NECOMAtter System
Web Interface
NECOMAtter
Human to Human Interaction
Human to Machine Interaction
RESTful API
Machine to Machine Interaction
NECOMAtter BOT
data collection module
Routine Analysis
NECOMAtter BOT
analysis module
Automatic Defense
NECOMAtter BOT
action module
Workspace for collection and discussion of security information for Humans
Messaging pipelines between Human and Machine

Figure 2.7: Three types of interactions via NECOMAtter

3 Bridging Machine-to-Machine Interfaces

This chapter will be centred around the n6 API, which was elected as the main machine-to-machine interface for the NECOMA system. The n6 API is in constant development since the beginning of the project, thus, it was already extensively described in several deliverables, with the source code of the implementation and extensive documentation publicly available (for details please refer to section 3.2.1 Public release). Thus, this section focus mostly on describing the integration between NECOMA components and the n6 API. It also provides feedback and experience while coping with, and using, the n6 API.

3.1 n6–MATATABI integration

One of the implementations of the threat analysis platform within the NECOMA project is MATATABI, which was also described in several documents([5], Deliverable D2.1[1]) providing a scalable platform to analyse and detect various types of threats based on the *big-data* technology. It stores a huge amount of datasets in a cluster environment and generates various types of detected threats in its own data store. Thus, publishing these analysis results to other parties of interests would be beneficial to enrich their analysis results.

We used n6 SDK to provide the interface as an application programming interface (API) for this purpose. Figure 3.1 depicts an internal message flow to provide a data entry stored in MATATABI to the external entities via n6 API.

[1]Deliverable D2.1: Threat Analysis (November 2014): http://www.necoma-project.eu/m/filer_public/f0/65/f0654584-6fe2-40cd-8204-4781dfc6f7a8/necoma-d21r1345-public.pdf

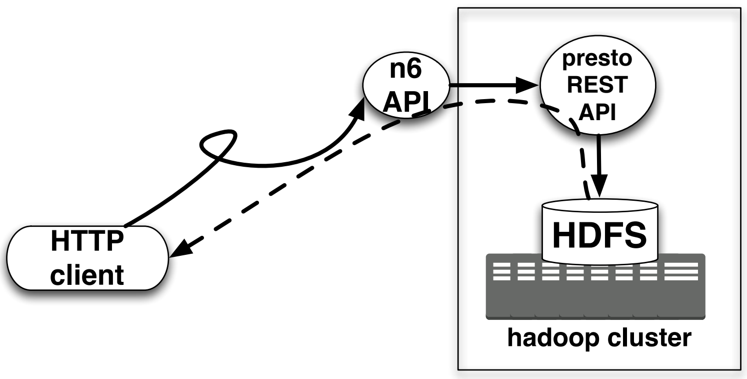


Figure 3.1: Overview of n6 MATATABI.

The integration of n6 and MATATABI is bridged by `Presto REST API`[2]. An HTTP request to query any information to MATATABI is firstly received by an instance of n6 API, then the request is forwarded as an SQL-like statement of presto-db[3], which eventually queries to MATATABI.

By using n6 RESTful API over HTTP transport, all the data stored in MATATABI can be published via API, however we only provide specific datasets in MATATABI because a query triggers huge computation load into MATATABI may cause an issue with a large amount of request. Thus, we limits its capability at the moment.

Table 3.1: An example of presto-db/Hive table scheme.

Column	Type	Null	Partition Key
srcip	varchar	true	false
srcport	bigint	true	false
dstip	varchar	true	false
dstport	bigint	true	false
label	varchar	true	false
nbdetector	bigint	true	false
confidence	varchar	true	false
dt	varchar	true	true

To publish a dataset via n6, we need to provide a mapping scheme between presto-db and n6. The following is an example of this mapping, where table 3.1 represents the original data scheme stored in MATATABI (in this case, we present a specific table named `mawilab`, which contains flow

[2]This API is not well documented but the information is summarized at https://gist.github.com/electrum/7710544.
[3]https://prestodb.io/

information with a label) and table 3.2 is a result of mapping to n6 API syntax.

Table 3.2: An example schema of n6 API.

MATATABI attribute	Corresponding attributes in the n6 API
category	category="flow-anomaly"
srcip	(address[0].ip / address[0].ipv6) & address[0].dir="src"
srcport	sport
dstip	(address[1].ip / address[1].ipv6) & address[1].dir="dst"
dstport	dport
label	name
nbdetector	*(omitted)*
confidence	confidence
dt	time

Based on the above mapping scheme, we can implement required API by a simple code by python. The figure 3.2 is a code snippet of this particular dataset.

```
class MawilabDataBackendApi(object):
  @staticmethod
  def parse(auth_data, params, query_result, **kwargs):
    columns = {'sport':1, 'dport':3, 'name':4,
               'nbdetector':5, 'confidence':6}

    # filter out with specified input parameters
    for data in query_result:
      for key, value in params.items():
        if params[key][0] != data[columns[key]]:
          skip = True
          break
      if skip is True:
        continue

    # generate output information
    yield {
      'source': 'mawilab',
      'category': 'flow-anomaly',
      'address': [{'ip': data[0], 'dir': 'src'},
                  {'ip': data[2], 'dir': 'dst'}],
      'sport': data[1],
      'dport': data[3],
      'name': data[4],
      'nbdetector': data[5],
      'confidence': convert_conf(data[6]),
      'time': datetime.strptime(data[7], "%Y%m%d"),
      }
```

Figure 3.2: A code snippet for n6 SDK.

3.2 Development of the n6 SDK

This section describes recent developments of the n6 SDK. The SDK is a software library that substantially simplifies implementing an n6-compatible REST API on top of arbitrary data sources.

3.2.1 Public release

In December 2014 NASK released the n6 SDK on an open source licence (GPL version 2)[4]. The source code is publicly available on *GitHub* code hosting service: https://github.com/CERT-Polska/n6sdk. Additionally, ex-

[4] GNU General Public License, version 2: http://www.gnu.org/licenses/gpl-2.0.html

tensive developer's documentation, including a step-by-step tutorial is available through the *Read the Docs* service: `http://n6sdk.readthedocs.org/`.

3.2.2 Recent improvements

This section covers improvements to the SDK introduced in versions 0.4.0 (released 2014-12-23) and 0.5.0 (released 2015-04-18).

3.2.2.1 Documentation

All the documentation is generated with Sphinx[5], which makes it easier to read and maintain. Contents of the documentation was significantly expanded, including docstrings present in the source code.

3.2.2.2 Query syntax

A new query parameter – *modified* – allows to select data by the time of addition to the database, which simplifies incremental fetching of data by clients. Additionally, an alternative way of specifying query parameters was introduced, which should increase compatibility with existing software.

3.2.2.3 Event categories

The following categories of events were introduced:

- `amplifier`: open DNS resolvers and similar services that can be used for DoS attacks
- `backdoor`: addresses of web shells or other types of backdoors installed on compromised servers
- `dns-query`: DNS queries and answers (no determination on legitimacy or maliciousness)
- `flow`: network traffic in layer 3 (no determination on legitimacy or maliciousness)
- `flow-anomaly`: anomalous network activity (not necessarily malicious)
- `fraud`: activities and entities related to financial fraud
- `leak`: leaked credentials or personal data
- `vulnerable`: addresses of vulnerable devices or services
- `webinject`: injects used by banking trojans

[5] Sphinx, Python Documentation Generator: `http://sphinx-doc.org/`

They reflect the growing diversity of types of information shared through the n6 API. Some of these categories were introduced to accommodate datasets provided by members of the NECOMA consortium.

3.2.2.4 Event attributes

Existing set of event attributes had to be extended as well, in order to provide common semantics for specific data elements that are inherent to newly introduced data sources. The following new fields were added to the n6 SDK:

- `dir`: role of the `address` field in terms of the direction of the network flow in layers 3 or 4. Possible values: `src` (address is the source of the flow) or `dst` (address is the destination of the flow)
- `email`: email address associated with the threat (e.g. source of spam, victim of a data leak)
- `iban`: International Bank Account Number associated with fraudulent activity
- `injects`: objects describing a set of injects performed by banking trojans when a user loads a targeted website
- `ipv6`: IPv6 address in the hexadecimal notation; `ipv6` and the previously existing `ip` are mutually exclusive – no more than a single address can be an element of the same `address` object
- `phone`: telephone number, national or international
- `rdns`: PTR record of the `.in-addr-arpa` domain associated with the IP address (without the terminal dot)
- `registrar`: name of the domain registrar
- `url_pattern`: wildcard pattern or regular expression triggering injects used by banking trojans
- `username`: local identifier (login) of the affected user
- `x509fp_sha1`: SHA-1 fingerprint of an SSL certificate in hexadecimal format

3.3 Feedback of n6 REST API usage

In this section we place ourselves in the context of a DDoS detection and mitigation platform, as examined in Task 3.3.3 and report here early experience in using the n6 API and the data NASK makes available through the

API. The idea for using the n6 API to confirm or infirm the identified attack sources to limit collateral damage.

3.3.1 Overview of the relevant capabilities of the n6 API

The malicious element in a DDoS attack can be identified in different ways, from source IP to a TCP or application session. The common criterion between our detection system and the n6 API is the source IP.

By studying the event categories in the API we identified potentially interesting categories:

- ’dos-attacker’, ’dos-victim’ and ’amplifier’ categories seem to be most relevant. Such event streams may allow us to monitor the common trends in terms of DoS attacks, vectors and sources. Such events could be used also to generate proactive mitigation rules. However, at the time of writing this report NASK was unable to share these datasets with the members of the NECOMA consortium but it is expected that at least partial access will be granted mid-2015.
- ’bots’ category can be used as a starting point to evaluate the confidence in the identification of an IP involved in a Distributed DoS, as botnets can be used to launch DDoS attacks.

We shall look in more detail in one usage of the ’bots’ category in the next subsection.

3.3.2 Using ’bots’ events for IP reputation

We consider different approaches to use events in the ’bots’ category, from a simple check of the presence of an IP in these events to taking into account time-related properties and the type of botnet being reported.

In the following section, with parsed and processed events we refer only to events in the category ’bots’. We define the value *score* as a floating number between 0 and 1. Higher the score, more confident we are in the identification of a malicious source IP address (i.e. we consider that the IP is likely to be a source of malicious traffic). Conversely a score zero means that the IP is unlikely a source of malicous traffic.

3.3.2.1 Simple evaluation

The simplest evaluation only checks if a source IP has ever been reported in the events. It means that we query the n6 API for a specific IP in the ’bots’ category. If the call returns one or more events, the IP score is set to 1.

Value of the confidence attribute value could be used in establishing the reputation score, but currently the value is always 1.

Should this change the scoring could be enhanced by counting the number of retrieved events and their related confidence and then scoring the IP depending on the average confidence.

This approach is limited, for example since:

- the starting date for the query window is far the API call can be quite long,
- IP reported a long time ago might no longer correspond to the same host (dynamic IPs) or the host might have been cleaned since.

The next approach tries to address these issues.

3.3.2.2 Time-based evaluation

This approach defines one possible way to take the event time into account when establishing the IP reputation

A first variant just uses the simple method described previously and limits the age of events taken into account.

A second variant consists of recording the history of an IP over an extended period (e.g. a year). We divide this period into equal slices (e.g. a day) and count the number of events (mapped with their confidence) for each slice. This time series allows us to monitor the activity of the IP over the extended period. We have considered following approaches in using this data:

- Identification of bursts in activity, and recent bursts result would result in higher score.
- Identification of activity trends. An increasing trend of activity would lead to a greater score than a decreasing activity.

3.3.2.3 Name-based evaluation

This approach uses the botnet name in the scoring. The name field of 'bot' category refers to the identification of the botnet to which the event belongs. At this point the taxonomy used for botnet identification might differ for each sensor. Thus we map each value to a malware/botnet and list the botnet's DDoS capabilities. DDoS capabilities can be:

- a boolean: the botnet is known to be used to launch a DDoS or not
- a list of the DDoS attacks/types that the botnet is used to launch, e.g. UDP floods or TCP floods.

For the first case bots known to participate in DDoS attacks would result in higher score. For the second type to be useful, the detection system should provide also information on attack type.

This approach has few prerequisites:

- we have to list all 'name' values and map them to a malware/botnet
- we have to construct and maintain the DDoS capabilities list for each malware/botnet. Such set up might be difficult as it seems there is no uniform database that provides such taxonomy.
- we have to correlate the DDoS detection alert attack details with the DDoS capabilities.

3.3.3 Summary

The use of the n6 API is intuitive and easy. Querying the API follows the common API syntax, e.g. use of '?' to begin parameters, '& to split parameters and the 'key=value' formating. Moreover fields to query are easily recognizable by looking at a result to a query with no parameters.

The taxonomy of categories is easily comprehensible and in addition for all terms are well defined in the documentation, should any uncertitude arise.

Querying the API over a long period (years) can take a certain time (up to tens of minutes). The fast exploitation of IP reputation might be difficult in such a case. Moreover, for queries requiring a long execution time the client loses the connection (python script, web browser) before receiving the end of the request (i.e. the closing JSON ']'). These performance issues depend on the characteristics of the database used for the backend and at the time of writing this report NASK is working on improving performance of some types of queries.

3.4 High Precision Phishing Detection REST API

The High Precision Phishing Detection REST API was created as a part of the machine learning based prototype that uses simple heuristics, derived from the host names, to detect suspicious web sites. The service allows submitting a URL (or a list of URLs) and responds with the classification and the probability of the classification in case of malicious class. Since it is directly connected to the analysis module database, the results are generated by current configuration parameters established by the stream learning component. The API is compatible with the n6 specification, although it's purpose is to estimate the category of the provided URL rather than allowing to query by the category itself.

Currently the service exposes two methods:

- **GET request**
  Allows submitting one URL as a GET request parameter.

  Example request:

  ```
  http://localhost:8080/prediction/?url=a.b.c.d.f.g.com
  ```

  Example response:

  ```
      {
          "url": "a.b.c.d.f.g.com",
          "category": "malurl",
          "confidence" : "high",
          "custom" : {
              "probability": 99.90774907749078
          }
      }
  ```

- **POST request**
  Allows submitting a list of URLs as POST payload.

  Example request:

  ```
  http://localhost:8080/prediction/
  ...
  [
      "google.com",
      "http://www.thelongestdomainnameintheworld.com/",
      "a.b.c.d.e.f.g.h.m",
      "http://https.www.cic.fr.fr.banques.cornerkebabonline.com/cic.fr/4c02/"
  ]
  ```

  Example response:

  ```
  [
      {
          "url": "google.com",
          "category": "benign-url",
          "confidence" : "high",
          "custom" : {
              "probability": null
          }
      },
      {
          "url": "http://www.thelongestdomainnameintheworld.com/",
  ```

```
            "category": "malurl",
            "confidence" : "low",
            "custom" : {
                "probability": 18.639053254437872
            }
        },
        {
            "url": "a.b.c.d.e.f.g.h.m",
            "category": "malurl",
            "confidence" : "high",
            "custom" : {
                "probability": 100
            }
        },
        {
            "url": "http://https.www.cic.fr.fr.banques.cornerkebabonline.com/cic.fr/4c02
            "category": "malurl",
            "confidence" : "high",
            "custom" : {
                "probability": 100
            }
        }
    ]
```

The service is currently being tested in a closed environment as one of the analysis modules in a DNS traffic analysis component. It is foreseen to expose the API for public access after the internal performance and security tests are finalised.

3.4.1 Experience in Usage and Utilisation

The prototype and the API are being actively utilised and tested as part of the DNS analysis module. Because the of the n6 specification flexibility and nature it can by easily extended and tailored to particular needs, e.g. incorporated directly to GUIs or other visualisation components. Also, because of it's light weight it is possible to obtain results in nearly real-time without overloading the analysis module. Though, in the case of analysing a list of URLs, it is recommended to submit them via the POST method described above, as requesting a list analysis, entry by entry, with the GET method caused performance issues.

4 Experience Gathered

This chapter reports experience gained during implementation and use of security information exchange mechanisms on a SSL/TLS dataset and provides insights on incorporating different cyber threat information exchange formats into the NECOMA ecosystem.

4.1 Deploying the n6 interface on top of the SSL/TLS dataset

The dataset provided by IMT comprises SSL/TLS hanshake information collected by probing servers of the whole IPv4 address space, and currently organised into several campaigns spanning across several time periods during individual years: 2010, 2011 and 2014. The acquisition of the information follows simple steps:

- a SYN-scan to detect servers with an opened 443 port;
- a TCP connection opening with the identified servers using a ClientHello message.

As of 2011, the message has been declined with various parameters in order to determine how servers respond to different stimuli.

By leveraging the n6 interface, we intend to make the dataset available to other parties. In particular, several tokens can be then used for multilayer analysis, leveraging server information such as the IP address, the timestamp of the probe, the protocol version and the ciphersuite chosen by the server. Certificate information is also provided, namely information about the issuer and the subject of the certificate.

Data is not provided as raw and we opted for a the use of MongoDB to store the parsed dataset. In Table 4.1, we provide a mapping between the entries in the table and the n6 fields.

Table entry	MongoDB type	n6 field
_id	ObjectId	N/A[1]
category	String	`UnicodeEnumField`
origin	String	`UnicodeEnumField`
record_version	String	`UnicodeEnumField`
stimulus	String	`UnicodeLimitedField`
timestamp	Date	`DateTimeField`
compression_method	String	`UnicodeEnumField`
source	String	`UnicodeField`
ciphersuite	String	`UnicodeLimitedField`
dport	String	`IntegerField`
issuer	String	`UnicodeField`
ip_address	String	`IPv4Field`
subject	String	`UnicodeField`
pem_certificate	String	`CertificateField`[2]

Table 4.1: Mapping between the SSL/TLS dataset information and the n6 fields

Valid credentials (login and password) have to be provided in order to access the database. Here follows some of the use cases considered in this implementation:

- Retrieve SSL/TLS information relative to a particular IP address:

```
http://<user>:<password>@<ip_address>:<port>/ssl_info.json?
ip_address=39.114.56.51,39.191.34.116,39.113.212.140

[
{
    "category": "ssl/tls_server_answer",
    "origin": "probe",
    "dport": "443",
    "timestamp": "2014-03-22T00:38:12Z",
    "source": "IMT",
    "record_version": "ERROR",
    "ip_address": "39.113.212.140"
},
{
    "category": "ssl/tls_server_answer",
    "origin": "probe",
```

[1]not present in the final object
[2]extended from `UnicodeField`

```
        "dport": "443",
        "timestamp": "2014-03-22T03:16:49Z",
        "source": "IMT",
        "record_version": "None",
        "ip_address": "39.191.34.116"
    },
    {
        "category": "ssl/tls_server_answer",
        "origin": "probe",
        "record_version": "TLSv1.0",
        "timestamp": "2014-03-20T01:40:26Z",
        "compression_method": "Null",
        "source": "IMT",
        "ciphersuite": "TLS_RSA_WITH_AES_256_CBC_SHA",
        "dport": "443",
        "issuer": "\"/C=KR/O=XNsystems",
        "ip_address": "39.114.56.51",
        "subject": "/C=KR/O=XNsystems"
    }
    ]
```

- Retrieve SSL/TLS information relative to a particular timestamp:

  ```
  /ssl_info.json?timestamp=2014-03-06T00:30:15Z
  ```

- Retrieve SSL/TLS information relative to an IP address at a particular timestamp:

  ```
  /ssl_info.json?ip_address=39.114.56.51&timestamp=2014-03-06T00:30:15Z
  ```

- Retrieve SSL/TLS information within a given timeframe:

  ```
  /ssl_info.json?timestamp.max=2014-03-06T00:30:15Z
  &timestamp.min=2014-03-05T23:39:33Z
  ```

- Retrieve SSL/TLS information before a certain date:

  ```
  /ssl_info.json?timestamp.max=2014-03-06T00:30:15Z
  ```

- Retrieve SSL/TLS information after a certain date:

  ```
  /ssl_info.json?timestamp.min=2014-03-05T23:39:33Z
  ```

- Retrieve SSL/TLS information relative to a particular IP address within a given timeframe:

```
/ssl_info.json?timestamp.max=2014-03-06T00:30:15Z
&timestamp.min=2014-03-05T23:39:33Z&ip_address=39.114.56.51
```

- Retrieve the certificate exposed by a given IP address in PEM format:

```
/get_certificate.json?ip_address=39.114.56.51
```

Presently, a machine-to-machine interface is to be deployed between the SSL dataset and the SSL/TLS analysis module, the module that computes a server score based on the certificate information, the protocol version and the ciphersuite chosen as described in Deliverable 2.1. The n6 interface is also to be implemented on top of the SSL/TLS analysis module in order to directly get the assessment of a particular server.

For the time being, the implementation of the n6 interface is still an ongoing process on the SSL dataset, therefore feedbacks can only be provided relating to the implementation of the interface. In our context, the use of the n6 interface as a mean to exchange knowledge proved to be flexible and relatively straightforward. No conversion concerns have been raised, due to the choice of a data storage technology supporting the JSON format natively, and the fact that it is the required output format of the n6 interface. Apart from the implementation of a custom authentication policy, no additions have been made to the current API, showing its comprehensiveness in our use case.

4.2 Experience from IODEF implementation

This section explains the relationship between NECOMA's threat information exchange scheme and existing information exchange schemes. Due to the number of the incidents is growing day by day, threat information needs to be exchanged among stakeholders.

4.2.1 Overview of Structured Cyber Threat Information

The Incident Object Description Exchange Format (IODEF) is an IETF standard for data representation. The aims of IODEF is providing framework for sharing information commonly exchanged by Computer Security Incident Response Teams (CSIRTs) about computer security incidents. Currently, IODEF is already used by security consortiums and vendors.

For example, IODEF is supported by the Anti-Phishing Working Group (APWG) [3], which is one of the biggest coalition against cybercrime, especially phishing. In order to collect threat information in a structured format, APWG provides a phishing and cybercrime reporting tool which sends threat

[3]http://www.antiphishing.org

information to APWG by tailoring information with IODEF format. The Advanced Cyber Defense Centre (ACDC) [4], is EU-wide activity to fight against botnets. ACDC provides a solutions to mitigate on-going attacks, as well as consolidating information provided by various stakeholders into a pool of knowledge. Within ACDC, IODEF is one of the supported schema for exchanging the information. Research and Education Networking Information Sharing and Analysis Center (REN-ISAC) [5], is a private community of the research and higher education members fro sharing threat information, and employs IODEF formatted-message to exchange information.

STIX [6] is another standard for exchanging threat, and was defined by MITRE cooperation. In comparison with IODEF, it supports additional contextual information regarding threat actors and their attack techniques, tactics, targets, and courses of action.

4.2.2 NECOMA and IODEF

In NECOMA project, we decided to employ n6 for exchanging threat information. As we mentioned, n6 uses an event-based data model for representation of all types of security information. Each event is represented as a JSON object with a set of mandatory and optional attributes. However, from the viewpoint of the dissemination of NECOMA contributions, the backward compatibility should be addressed in regard to collaborate with security consortiums and vendors.

n6 therefore supports alternative output data formats for keeping compatibility with existing systems - IODEF and CSV - although they lack some of the attributes that may be present in the native JSON format.

As a part of dissemination work, we promote n6 through our standardization activity in IETF. Currently, a member of the NECOMA consortium proposes an internet draft which aims at informing implementation from vendors and researchers who have used IODEF, in the Management Incident Lightweight Exchange (MILE) working group which is actively reviewing the base standards and extending them to meet the use cases for incident response and security teams. This member became a co-author of the draft since March 2014, and introduced n6 in the draft since November 2014.

4.2.3 Experience from implementation

Since IODEF and its XML Schema Document (XSD) were defined in RFC 5070, we thought that libraries for accessing XML attributes could be automatically created by code generators. However, the complexity of IODEF XSD hinders us to generate libraries. We explored a workaround, and

[4] http://acdc-project.eu
[5] http://ren-isac.net
[6] https://stix.mitre.org/

found that it can be done by converting from XSD to XML by serialization, and re-converting from the serialized XML to XSD. The generated class libraries for several programming language are available at: https://github.com/daisu-mi/IODEF-codegen

We also found several operational issues on using IODEF. In some cases, we were not sure about which attributes should be chosen. For example, Incident Type class is used to categorize incidents into several types. However, there already exist various incident classifications, and it is often hard to fit them into the type attribute of the impact class. The numbering of Incident ID needs to be considered. Otherwise, information, such as the number of incidents within certain period could be observed by document receivers. Additionally, in regard to the gap between n6 and IODEF, a country code was not supported in the IODEF format whereas the n6 format supports to describe the country code for an attack node.

5 Conclusion

The aim of this deliverable was to report on the results and gathered experience that was obtained during the implementation and application of the security information mechanisms as presented in Deliverable 3.2: Security Information Exchange – Design.

The mechanisms were comprised of 3 main categories: machine-to-human interfaces, machine-to-machine interfaces and data storage. This deliverable extensively covered the former two interfaces, as the subject of data storage was identified to be extremely broad and beyond the scope of this deliverable.

Task 3.2 catalyzed developments of information exchange mechanisms across both EU and Japanese consortiums, benefiting individual consortium members to take a new look at the security information exchange from variety of perspectives.

Security information exchange encompasses variety of traits: machine-to-human interfaces, machine-to-machine interfaces, as well as the data representation, conversion, aggregation and enrichment. Throughout the designs and implementations of proposed systems and interfaces, many of these important traits are highlighted and better understood. Furthermore, they were validated through diverse datasets and rich array of algorithms that were established in Workpackage 1 and 2, respectively.

NECOMA project further intends to validate the proposed designs and implementations through periodic test campaigns, Workpackage 4 demonstrators and ongoing collaborations with relevant projects.

Bibliography

[1] T. Bray. RFC 7159: The JavaScript Object Notation (JSON) Data Interchange Format, Mar. 2014.

[2] M. Kato, K. Cho, M. Honda, and H. Tokuda. Monitoring the Dynamics of Network Traffic by Recursive Multi-dimensional Aggregation. In *OSDI2012 MAD Workshop*, Oct. 2012.

[3] V. Paxson. An analysis of using reflectors for distributed denial-of-service attacks. *ACM SIGCOMM Computer Communication Review*, 31(3), July 2001.

[4] P. Phaal, S. Panchen, and N. McKee. RFC 3176: InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks, Sept. 2001.

[5] H. Tazaki, K. Okada, Y. Sekiya, and Y. Kadobayashi. MATATABI: Multi-layer Threat Analysis Platform with Hadoop. In *3rd International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*, Sept. 2014.