# Endreport

# CyberDEW
# A “Distributed Early Warning” System for Cyber Security

**Cyber Security**

| | |
|---|---|
| **Researchtheme** | **Malware** |
| **SBIR projectnumber:** | **SBIR13C043** |
| **Projectitle:** | **CyberDEW A “Distributed Early Warning” System for Cyber Security** |
| **Contact:** | **Frans Jansen** |
| **Executed by:** | **Thales Nederland B.V.** |
| **Project start date:** | **September 9, 2013** |
| **Project end date:** | **Februari 28, 2015** |

Feb 27, 2015

THALES

© THALES NEDERLAND B.V. and/or its suppliers
This information carrier contains proprietary information which shall not be used, reproduced or disclosed to third parties without prior written authorization by THALES NEDERLAND B.V. and/or its suppliers, as applicable

# TABLE OF CONTENTS

# 1. MANAGEMENT SUMMARY

**Introduction**
The goal of the project was to create a system (CyberDEW) that detects in an early stage, cyber attacks or parts thereof (e.g. elements of the reconnaissance phase). From the outset of the project it has been clear that a "hard" deterministic, detection of an attack on basis of "weak signals" is very difficult, if not impossible. Therefore the term "detection" should be understood in the probabilistic sense. The CyberDEW system generates a number (belief) indicating the probability that the attack for which CyberDEW has been setup, is taking place. The bespoke number can subsequently be assessed by a natural person (security officer) or by a knowledge system (e.g. a Security Information and Event Management System (SIEM)).

**Does it work?**
Detection in CyberDEW is based on the premise that isolated effects (distributed weak signals) emanating from an attack in themselves contain insufficient information but that the combination of bespoke weak signals on basis of knowledge of the attack (domain knowledge) can lead to useful information about the presence of such attack.

The majority of the team effort has been put in 1) obtaining attack characteristics (domain knowledge) from real data, 2) building a scalable and flexible engine that uses the weak signals from distributed sensors to yield probabilities of ongoing attacks and 3) building an attractive and intuitive demonstration that combines the two and shows the potential of the combined technologies. These are,
1) model (hypothesis) based reasoning on basis of domain knowledge;
2) workflow- and service-composition;
3) sensor technology;
4) secure data distribution.

The basic idea of detection using (distributed) weak signals on basis of domain knowledge is shown in the demonstration. The available project resources were insufficient to develop a null-version of a product. Further areas of research and development lay in the area of improving the user-interface towards the detection-engine(s) such that attack knowledge can more easily be translated in algorithms and algorithm parameters (by using for instance a model-based software development tool). The development of (in-system, user-action) sensors requires further attention.

**Is there a market?**
From discussions with various potentially interested parties it has become clear that the concept of distributed detection combined with probabilistics is deemed to be useful and elegant. The complexity of detecting advanced attacks requires substantial effort and demands product teams that continuously stay abreast of the latest developments and emerging malware. The technologies provided and combined in CyberDEW have the potential for creating a dynamic and evolvable intrusion detection system (IDS). Such novel technology would generate valuable information that is fed into SIEM and/or other information analysis fabrics.

**Societal and other effects**
Detection of advanced attacks is a complex and demanding task, in particularly when carried out in isolation. It is therefor believed that bundling efforts is essential. The probabilistic fusion of attack information from various sensors, over multiple entities (companies, sectors, nations) and flexibly combining that with other types of information could give valuable indications of ongoing attacks thereby alerting cyber security personnel. This subsequently could lead to cost-reductions. The framework and technology developed and combined in the CyberDEW project offers exactly a platform for such intelligent bundling.

Other "spin-offs" are the activities performed by Hogeschool Rotterdam (HR) namely laying the base of a curriculum in intrusion detection as well as the submission of a paper on preventing (spear)phishing attacks.

## 2. PROJECT EXECUTION

The following activities have been carried out.

| Phasing & distribution of tasks | By |
|---|---|
| 1. Overall Program Management | TCS-NL |
| *2.* Architecture<br>*Involved the mapping of required functionality onto technology such as "DPIF" (see 3.1.3.4) and "Martello"(see 3.1.3.5)* | TRT-NL |
| *3.* Processing Node & Leaf Node<br>*Research into & development of applicable algorithms* | TRT-NL |
| *4.* Sensors<br>*Development of new sensors (e.g. event-generation upon opening mail, detection of (encrypted) downloads. Research into user-originated event generation (phishing detection)* | HR, TRT-NL |
| *5.* Attack knowledge derivation via dataset analysis<br>*Analysis of NCSC, HR, ASP4ALL (MinV&J) datasets.* | WODC |
| *6.* Market exploration & business plan<br>*Various meetings with potential users.* | TCS-NL |
| 7. Demonstration building | TCS-NL |

Deviations with respect to the original phasing and tasks,

- Significantly more time than initially planned has been spent on finding specific attack patterns. The hope had been that analysis of honeypot data (from NCSC) would yield specific patterns. This was not the case, the same held for other data sources.
- The testing of the efficiency of the solution could not be performed in the absence of accurate domain models. However, the working of the principle has been tested with simplistic domain models distilled from documented cyber-attacks.
- Over time it became more and more clear that there is large value in offering a framework capable of dynamically coping with the always-changing characteristics of cyber-attacks.

**Communication & cooperation**

A total number of 12 face-to-face project meetings have been organized. The initial frequency was 1 meeting per 4-5 weeks. To increase efficiency, and limit the amount of hours spent, from September 2014 on the face-to-face meeting frequency has deliberately been decreased to a minimum, where a bi-weekly conference call (bridge) had been started.

From the outset of the project the 4 project parties have had their own, only slightly overlapping, task-areas. WODC focused on finding attack-patterns, Hogeschool Rotterdam focused on sensors, Thales Research focused on technology and detection algorithm while Thales Cyber Security took project management and the translation to the market as its tasks.

Problem, architecture & market

The activities executed during the project focused on two streams:
1) finding a realistic attack pattern that could not be detected before but could be detected by CyberDEW
2) creating an architecture and platform that would be able to detect the newly found attack patterns (see above) and be flexible enough to benefit from over time gained knowledge of the attack.

Finding an applicable attack pattern has turned out to be complicated. We have analyzed information from NCSC (anonymized honeypot data), from HR (netflow data) and from a party hosting government websites. However, the data collected was either purely attack data (honeypot) or unlabeled (netflow). Finding attacks in such unlabeled data is very time consuming and difficult without explicit domain and local network knowledge. It was therefor decided to base architecture/solution development on a relevant and well-documented distributed attack, the so-called "Nitro-attacks".

# 3. FINDINGS & CONCLUSIONS

The goal of the project was to create a prototype of and an assessment on the business case for a product capable of detecting complex attacks. A complex attack is characterized as an attack taking place in multiple phases, potentially from multiple places. A phase spreads out over a timeframe varying from a second to many days or even longer. Detection is based on the premise that effects emanate from the attack. These effects (e.g. a connection made with a Command and Control (C&C) center, encrypted data being sent to an unknown destination) in themselves unveiling little information are regarded as "weak signals". The combination (fusion) of bespoke weak signals however on basis of knowledge of the attack (domain knowledge) can lead to useful information about the presence of such attack.
Section 3.1 discuss the topics of problem description and complex attacks; detection: weak signals, domain knowledge and the engine realizing detection. Section 3.2 discusses the economic perspective while section 3.3 discusses benefits from the societal perspective.

## 3.1. Technical perspective

### 3.1.1. Problem

Cyber security incidents are usually detected by using various sensors that, in turn, are distributed across various locations and organizations. Cyber security attacks can be detected locally by using local sensors and/or can be detected centrally by using the information collected from distributed sensors. Detecting cyber attacks at a local level may provide a shortsighted vision on ongoing cyber attacks. This occurs in those scenarios where a large number of victims at various locations are attacked simultaneously or sequentially. Every sensor at a site collects a small amount of evidence about the ongoing attack, and therefore per definition produces weak signals about the attacks (i.e., with a low accuracy, certainty, etc.). When these sensors join forces and share their information at a central point[1], the resulting set of evidences can improve the quality of the inference process about the attack. Sharing raw information with a central node, on the other hand, may not be fruitful due to (a) creating information overload at the central point, (b) losing domain knowledge due to not communicating all the detailed contextual information to the central point, and (c) revealing business/privacy sensitive information to the central point. Therefore, solutions based on a centralized architecture (e.g., Security Operations Center, SOC) are inappropriate in largely distributed and cross-organizational settings. In such settings, therefore, one should seek a balance between local and central processing of information such that at the local sites the domain knowledge is fully exploited and just relevant and aggregated information is communicated to the central point, and at the central site a high-fidelity view of ongoing attacks is constructed (i.e., with a high accuracy, certainty, etc).

#### 3.1.1.1. Example: Nitro Attack

An attack satisfying the characteristics of a complex attack is the Nitro Attack [1] . The Nitro-attack targeted a total of 29 companies in the chemical sector and another 19 in various other sectors. Primary purpose was industrial espionage. The attack wave started in late July 2011 and continued into mid-September 2011. Many computers in various parts of the world were infected. The attackers started the attack by sending specifically targeted email. Upon opening an infected attachment or upon clicking a link the Poison Ivy (PIVY) RAT would be installed. Subsequently PIVY would contact the C&C server after which the infected computer is instructed to provide IP-addresses, names and dumps of the computer(s) in the workgroup/domain. Next the attackers would traverse the network and exfiltrate intellectual property. Five events that are phenomena of a Nitro-attack have been identified. See the figure 1 below.

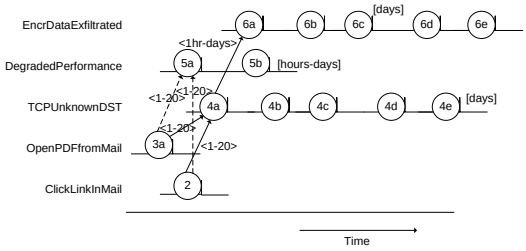



**Figure 1: Nitro-related events: sequence & time**

[1] Note that local sensors can share information in a peer-to-peer way, which inflicts communication processing load on the local nodes (the peers).

### 3.1.2. Solution

Collaborative and cooperative solutions are used among organizations [2] and are proposed for, for example, preserving user privacy [3], identity management [4] and intrusion detection and prevention [5]. In the context of IDSes the authors of [6] define cooperative intrusion detection as ”a distributed system, where participants exchange information for intrusion detection and prevention”. In such a system individuals can work also alone, but by cooperating they gain some benefits. A collaborative IDS, on the other hand, is ”a dynamic distributed system where participants form new organizational structures that can adapt to different roles to fulfill tasks not solvable by a participant on its own, i.e. the result must substantially differ from the individual functionality” [6]. Benefits of collaboration and cooperation can be at architectural level (to improve scalability and availability), team level (to compensate shortcomings of individuals), and big picture level (to improve overall awareness, providing a so-called weather report or dashboard) [6] and [7].

In distributed and cross organizational settings, as mentioned above, one should seek a balance between local and central processing of information such that at the local sites the domain knowledge is fully exploited and just relevant and aggregated information is communicated to the central point, and at the central site a high-fidelity view of ongoing attacks is constructed (i.e., with a high accuracy, certainty, etc.). Figure 2 shows the hierarchical structure of information processing within CyberDEW platform, where information resulting from level 1 (i.e., the belief that a certain type of attack is ongoing) is fed into the next higher level while level 1 details are not by default communicated (but could be retrieved when needed e.g. for forensic purposes). Our proposal as to the structure of a scalable, adaptable and hierarchical approach is shown in figure 2.

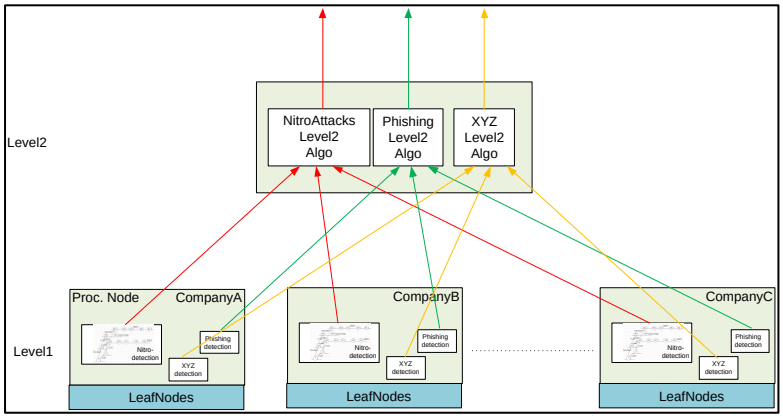
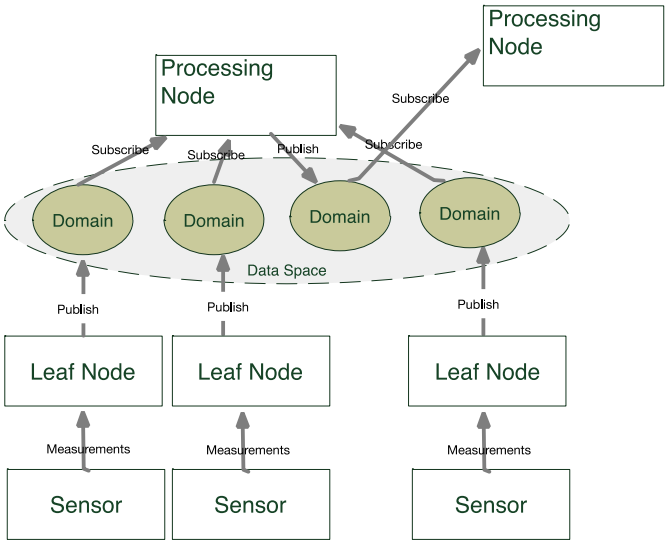



**Figure 2: Hierarchical abstraction of information**

The type of information handled at levels 1 and 2 is entirely different. Assuming level 1 represents the branches of a detection tree the leaf nodes typically collect weak-signals such as IDS-information, or the opening of an e-mail attachment. As shown later this type of information is processed using an auto regressive hidden Markov model (AR-HMM) algorithm that relates events occurring over time. Level 2 at least receives from level 1 beliefs that a certain attack is taking place. This type of information is entirely different in its meaning as well in its behaviour over time. The algorithm applied at level 2 shall therefor have a different character. Selection of the algorithm and its parameters is up to the security officer as it is domain-specific.

### 3.1.2.1. CyberDEW Architecture

The CyberDEW architecture satisfies the need for scalability by supporting hierarchy and abstraction. Moreover does it support flexibility, required to cope with ever evolving malware. The architecture is made up of leaf nodes and processing nodes and uses a data distribution system to send updates from leaf nodes to processing nodes and between processing nodes. This architecture is shown in Figure 3.

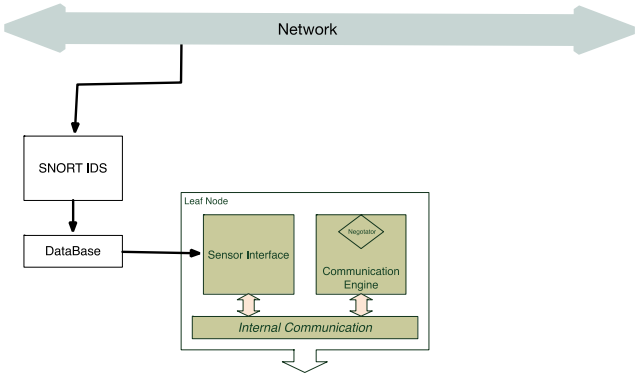



CyberDEW leaf nodes take measurements from sensors (such as an IDS) and *may* perform some initial processing on them (such as counting the number of alerts from a given target over the last period). However, equally, the leaf node could take and publishes all of the information from the sensor.
CyberDEW processing nodes take the information published by one, or more, leaf nodes and perform some operations upon it. A processing node could contain an algorithm, or an interface to a human expert. CyberDEW processing nodes publish their results in the same data space, but a different data domain. Processing nodes can also subscribe to data produced by other processing nodes in the hierarchy. This allows the results from the sensors to be aggregated by the processing nodes and uploaded to the higher-level authorities as required.

**Figure 3: CyberDEW functional architecture**

## Leaf Node

A CyberDEW leaf node can be implemented with any sensor, as long as the sensor and the leaf node use an agreed interface. Figure 4 shows an example where Snort places information in a Database (as normal) and the Leaf node reads from this database. There exists a processing plugin that can read from such databases.




Other leaf nodes can be created that directly interact with the IDS and this will be required for two-way communication with the sensor (i.e. controlling the IDS to increase sensitivity). The current implementation supports *Select* statements from *MySQL* but using a different database would be a straightforward matter.

**Figure 4: Leaf node with a Snort IDS sensor**

## Processing Node

A Processing Node is architecturally identical to a leaf node with the exception that instead of the interface to the sensor, there is a processing plugin. This processing plugin contains either an algorithm or an interface to a human sensor.

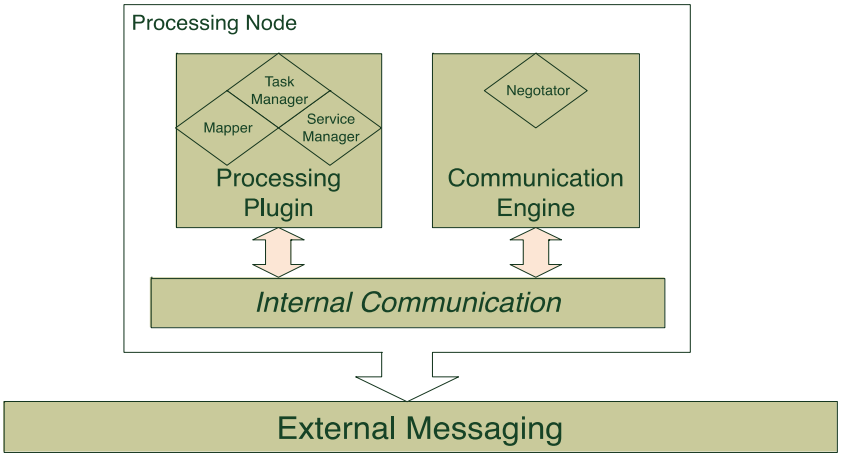
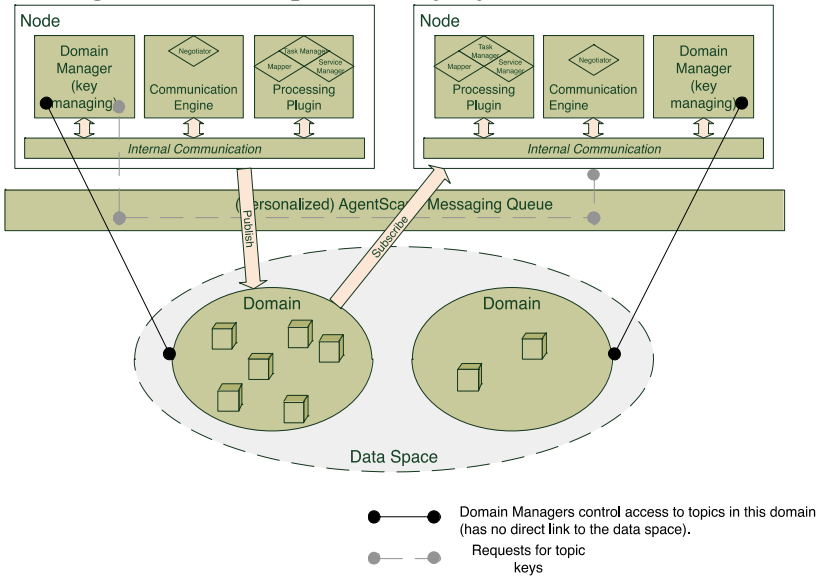



Figure 5 shows the structure of the Processing Node. The external messaging is, in the case of Cyber-DEW, the publish/subscribe system. Processing nodes are implemented by using the Dynamic Processing Integration Framework (DPIF) from Thales Research and Technology, NL.

**Figure 5: Processing Node Structure**

## Securing access to data produced by CyberDEW nodes




As the information travels between CyberDEW nodes, it is secured using the Martello framework (see figure 6). Martello allows data to be encrypted such that only authorised users can decrypt it. Martello operates on the assumption that data objects are secured, but the actual encrypted data itself is accessible to all users of the system. Leaf nodes and processing nodes use Martello invisibly without any change (outside of configuration/ management) to the existing architecture.

**Figure 6: Securing CyberDEW nodes**

## Middleware and Software modules

Both DPIF and Martello need a middleware for a number of basic operations, such as messaging and accessing the data space. The current implementation uses ActiveMQ as the middleware with *Hazelcast* providing the shared data space. These are both freely downloadable. The DPIF and Martello frameworks are Thales NL proprietary code. Therefore, they will be delivered as binaries, where required.

### 3.1.3. Applied technologies

The detection/posing-a-belief of an attack on basis of weak signals is based on the premise that the weak signals occur in a specific pattern(s), over place and over time. Prior to building the detectors the patterns/properties of an attack in terms of weak signals must be derived. Subsequently algorithms must be found that best match the characteristics of the attack. The section below addresses the various techniques and technologies.

#### 3.1.3.1. Techniques of finding patterns

The objective was to detect malicious behaviour, that aim to compromise the security of an enterprise, in a large volumes of netflow data, where a flow is defined as a unidirectional sequence of packets between a given source and destination. Therefore a processing method was developed that first reduced the large volumes of netflow data in a step-wise way and then categorised the pattern that is produced by the set of flows from each IP address that is contacting the enterprise as benign, malicious or suspicious.

The real world dataset derived from a large, high-speed network contained 5 months of continuous netflow data from 11 independent routers, resulting in 240 Gbyte of data. To reduce this dataset to a manageable size the following selection criteria were used: 1) only select flows for which the source IP address was from outside the enterprise. 2) Classify for each individual router the pattern that the flows of a source IP address creates. 3) Limit pattern matching to a period of at maximum 24 hrs. 4) As a sufficient number of flows is needed to match a pattern, only IP addresses that showed ≥35 flows per day were taken into account.

First IP addresses from well known and trusted internet companies such as Google, IBM, Microsoft etc. were identified. This to find out what kind of patterns their flows created to determine what could be considered benign. Each source IP showing similar patterns was therefore also classified as benign, the remaining IP source were either classified as malicious if it showed a pattern that was previously identified as a possible cyber attack (mainly a port scan) or suspicious if it did not show any similarities with known patterns.

#### 3.1.3.2. Results of finding patterns

An algorithm was developed to classify each pattern that the flows of a single source IP address creates. It was found that the large majority of the source IP addresses can be described by 5 different patterns that are all classified as benign. In total over 99% of all source IP addresses can be described with 15 patterns to detect benign cases and 6 patterns for the malicious cases, with the remaining <1% of the source IP addresses showing an unknown patterns that were classified as suspicious. Testing the algorithm against a dataset from a different enterprise showed it behaved equally well, suggesting that the process of finding malicious behaviour via exclusion is valid for a wide range of networks.

As already pointed out, no applicable attack patterns that could be useful for CyberDew were found. Instead, the patterns that were classified can mainly be attributed to port scans, which are thought to be the reconnaissance step for the potential intrusion of a network at a later time. Since the dataset consisted of 11 independent routers (sensors), the detected port scans could still be used to study the effectiveness of a distributed defence against attacks. Over 50% of the port scans are detected at multiple sensors, with the large majority of the port scans only occurring once at a sensor. This makes fast reaction times (in the order of seconds), to inform the other sensors about a potential port scan/attack, crucial to make a distributed defence worthwhile.

#### 3.1.3.3. Detecting attack patterns

In order to be able to automatically detect attack patterns produced by (multi-stage) attack different methods can be deployed. In this section two such methods are discussed, namely auto regressive hidden Markov model (AR-HMM) and Dempster-Shafer model. While the former is implemented in our demo for detecting a multi-stage Nitro attack, the latter is explained as alternative approach in order to emphasis the flexibility of the CyberDEW platform in accommodating various algorithms.

## AR-HMM

An AR-HMM is a probabilistic model that can be used to detect complex intrusion attacks consisting of multiple attack stages. When an attacker is executing a certain stage of the attack this will not be directly observable. However, the execution of this stage can result in sensors registering the symptoms of this attack stage. For example, if an attacker is in the reconnaissance stage of the multi-stage attack an IDS might report a port/IP-scan. The port scan is an observable symptom of the reconnaissance attack stage.

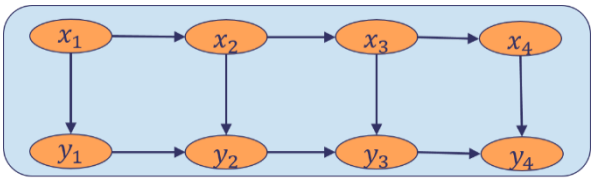



**Figure 7: An example of an AR-HMM with 4 time slices where the $x_t$ node represents the attack stage and the $y_t$ node represent the observation (symptom) at time *t*.**

The Nitro attack is an example of a multi-stage attack which can be detected through an AR-HMM by observing the symptoms in a certain order that is characteristic for the Nitro attack. In other words, the type of symptoms and the order in which these symptoms appear tells us something about the likelihood that a Nitro attack is being executed on the system. This likelihood is computed with the help of an AR-HMM.

An AR-HMM is a specific instance of a *dynamic Bayesian network* [8] that consists of two parts: a qualitative and quantitative part. The qualitative part of an AR-HMM encompasses the structure that consists of a hidden process layer (e.g. the variables $x_t$ in figure 7) and an observable layer (e.g. the variables $y_t$ in figure 7). The states of the variables in the hidden process layer represent the stages of the attack, while the states of the observable variables in the observable process layer represent the symptoms of an attack that we can observe. In AR-HMM the sequence of the attack stages and observation is modeled over time, where the states of these variables are observed or estimated for each time slice. Each time slice is composed of an $x_t$ and $y_t$ variable at time $t$. The quantitative part of the model encompasses the parameters. Two types of parameters need to be specified, namely the transition conditional probability parameters $P(x_t|x_{t-1})$ and the emission conditional probability parameters (also sensor model parameters) $P(y_t|x_t, y_{t-1})$. Based on the structure and the parameters of the AR-HMM we can compute the posterior probability $P(x_t = final\_attack\_stage|\epsilon)$ i.e. the chance that the attacker is in the final stage of his attack at the current time t given the set of observed symptoms $\epsilon$. If this probability is high that means that it is very likely that the Nitro attack is being performed on the system.

In a realistic environment the symptoms of the attack are often mixed with observations not coming from the attack (these are called parasitic observations) or missing attack observations which make the detection of the attack more difficult. AR-HMM are very robust models against parasitic and missing observations. The reason for this is that AR-HMM are able to remember the already seen observations sequences (or part of it) while these sequences are mixed with parasitic or missing observations. This behavior is very similar to rule-based detectors which have perfect memory, except that rules lack the possibility to handle missing observations. Also, AR-HMM are able to decrease the belief of an attack being present when over a long period there are no observations supporting the attack. This property is called delay. Since rules are deterministic this is also something that cannot easily be modeled with rule-based systems.

## Dempster-Shafer

To detect a cyber attack such as port-scans the AR-HMM approach produces two probability values corresponding to two events of being "attack" and being "no-attack". Sometimes the system cannot be certain about the possible outcomes due to, for example, lack of enough evidence or invalid evidences. The Dempster-Shafer (DS) theory models such uncertainties by defining a basic probability assignment to all subsets of the set of possible outcomes. For example, if the set of possible outcomes for a random variable X is {attack, no-attack}, the DS approach defines/assigns basic probabilities of subsets {}, {attack}, {no-attack} and {attack, no-attack}; while the Bayesian probability defines the probabilities of events {attack} and {no-attack}. In the DS approach, the empty subset {} always gets a basic probability value 0; and subset {attack, no-attack} symbolizes the fact that the detection system is uncertain (i.e., cannot make choices) about either of the outcome. This way of uncertainty modeling is an advantage of the DS approach. In the beginning of an inference process, for example, there are usually not enough evidences. The DS approach, therefore, can start from a completely uncertain state (i.e., where subsets {attack}, {no-attack} and {attack, no-attack} have basic probability assignments 0, 0 and 1, respectively) and allow the observed evidences form the belief in each of the subsets gradually. The probability model (e.g., the one used in the AR-HMM approach), on the other hand, requires the designer to (somehow) determine a-priori probability distribution over two random variable outcomes of {attack} and {no-attack}, even when there are not enough evidences (for example, in the beginning of the inference process).

The DS algorithm can be used instead of the AR-HMM approach for detecting (multistage) cyber attacks at CyberDEW processing nodes. To this end, the DS basic probability assignment should be updated with the DS combination operation (see [10]) for all *subsets* of the outcomes of random variable X. For example, if a Nitro attack consists of 5 stages, then

random variable X has 5 outcomes and in the DS approach one needs to calculate $2^5$ values of the DS basic probability assignment per update interval, while in the AR-HMM approach one needs to calculate only 5 probability values per update interval. Compared to the AR-HMM approach, therefore, the DS approach inflicts some extra processing complexity. Note that in the DS approach one does not need to overwhelm the end-user/decision-maker with all probability values calculated. It suffices to inform her/him only about those outcomes with, for example, most significant basic probability assignment values. In some occasions, determining which outcomes to filter out might complicate the DS decision-making process. Considering the advantages of the DS and AR-HMM approaches mentioned, one may decide to use DS approach in the beginning of an inference process when there are not enough evidences and switch to the AR-HMM approach at some point when there are enough evidences.

### 3.1.3.4. Dynamic Process Integration Framework (DPIF)

DPIF is a flexible environment that supports fast creation of systems that can efficiently and reliably analyze very complex patterns consisting of weak signals. In particular, DPIF supports:

- Plug&Play for fast adaptation of the detection capabilities with negligible effort ➔ maintain a rich detection system with a small team.
    - New types of sensors (e.g. IDS capabilities, server logs…) can be quickly added by the operators and exploited by advanced analysis processes, such as detectors of multi-stage attacks.
    - New attack models of arbitrary complexity can be added on the fly, as they become available.
- Dynamic creation of information flows between the different components. The resulting information flows support selective data-pull and data-push ➔ find the right type of the data source supplying the right information under the right conditions (e.g. part of the system network, in a specific time interval, etc.).

In DPIF each sensor and each detection process has a proxy, i.e. a "process driver" that (i) makes the sensor or detection process interoperable with other sources and processes and (ii) allows dynamic information management. Proxies are based on the Dynamic Process Integration Framework (DPIF), a recently introduced technology enabling cost and time efficient creation of advanced Service Oriented Solutions [11]. In particular, these proxies introduce the following functionality (see also [10] and [11] for more information)

- Make a process visible in a dedicated processing environment ➔ make the process discoverable. Make a service provided by a specific component interoperable and composable ➔ introduce a "LEGO" approach by providing uniform interfaces.
- Put a process into a context (e.g. associate a process with the data from a certain area, time, source, etc.).
- Translate between a local process and data objects in a common processing environment using the right interoperability standards.
- Mechanisms for invoking and controlling services.
- Filtering mechanisms/negotiation using multiple criteria.
- Automated creation of workflows and their maintenance.

### 3.1.3.5. Secure Data Distribution System (Martello)

Martello is an approach to defining and implementing an end-to-end (multi-level) security solution for data distribution systems. This solution aims at protecting the confidentiality and integrity of data objects for their entire lifetime, regardless of the security of the storage and communication media. One of the most critical factors when distributing information between partners is controlling when, where and to whom this information is passed. The fundamental principle within Martello is that data remains under the control of the data owner at all times. This ensures that the owner's requirements regarding privacy are maintained.
Martello is implemented using a publish-subscribe mechanism to share information between different organizations, using a shared data space. Information stored in the shared space is controlled by the organization that created it, and access is granted to others via their Domain Manager (DM). The domain manager uses an access control policy to determine whether or not a specific entity should be granted access and what access rights to grant (and for how long etc.). This access control mechanism is dependent on the application domain.
A Martello system is made up of one or more domain managers, each one controlling (indirectly) access to data owned by that domain. These domain managers provide the key exchange mechanism for data producers and consumers. The domain manager provides the ability to perform authentication, authorization and non-repudiation within the system. The Martello library is implemented in two main parts:

- domain manager: manages the users and the cryptographic keys;
- Martello client: manages the keys granted to the specific clients.

Typically, programmers do not have to be concerned about the cryptographic keys or access rights used by Martello – the library takes care of the keys and allows the programmer to manage the data as before. When data is produced, the **encrypt** method is called before putting the data in the shared data space. Similarly, when data is consumed, the **decrypt** method is called before the data is handled by the original application.

### 3.1.3.6. Weak event detection

**Mail Detector Sensor**

In the Nitro attack profile, one of the causal points is the detection of mail attachments being opened. In CyberDEW, we developed a sensor that monitors this activity and reports (via a leaf node) this activity. This sensor operates by monitoring the attachments directory on the client machine at the operating system level. Any file access activities are detected and then reported to the relevant processing node. A proof of concept implementation was developed on top of Mac OS X using Microsoft Outlook. This uses the *fs_usage* command found in Mac OS to monitor the directory path and processes the output to determine if a relevant file is accessed. In the complete system, equivalent sensors for other operating systems will be built.

**Download Detector**

Snort[2] is a powerful (de facto standard) open source network intrusion detection system (NIDS) which can also be used as an network intrusion prevention system (NIPS). The Snort detection engines utilizes signature, protocol and anomaly based inspection methods. Rule-based packet inspection depends on well written detection rules. These rules describe know anomalies, malicious traffic, etc. New rules to detect previously unknown anomalies or malicious traffic can be acquired from Sourcefire. For acquiring the most recent rule-sets a paid subscription is required. Users can also develop their own rules to identify attacks.The widespread use of Snort being highly configurable and the de facto standard for IDSes makes Snort the primary choice for CyberDEW. Snort depends on other third party software for its operation. Libpcap[3] for packet capturing. Barnyard2 for converting Snort's unified logging output to a MySQL database. Any Snort alert logged to a MySQL database can be used by CyberDEW (see figure 8).
These days the payload of most network traffic is encrypted using SSH or SSL. Rule-based IDSes are unable to check the contents of a packet with an encrypted payload [12] which renders these rule-based IDSes useless. Fortunately Snort has the ability to detect network-traffic anomalies by the use of preprocessors. There are several off the shelf preprocessors available for Snort. If none of these preprocessor are adequate, a dynamic preprocessor starter kit (DPX) is also available. The DPX C-source code can be modified to meet one's requirements.
Using DPX and modifying its code was our approach for the development of a preprocessor to detect an encrypted download. An operational CyberDEW preprocessor able to detect an encrypted download was developed.
The definition of download is important. Any data transferred could be qualified as a download. The CyberDEW preprocessor is adding up the size of the packets within an encrypted-data-flow. If within a preconfigured time-frame this data-volume exceeds a preconfigured amount an alert is generated. Most malware file sizes seems to be between 100 and 400 kB[4]. Most interesting are downloads from previously unseen IP source addresses. For this the preprocessor is keeping track of all IP-source addresses in a binary-tree and only 'forgets' if the system will run out of memory. Most alerts are false positives in terms malware download detection. Making the distinction is beyond the scope of the CyberDEW preprocessor alone. The CyberDEW preprocessor can be added to an already installed and operational Snort IDS.

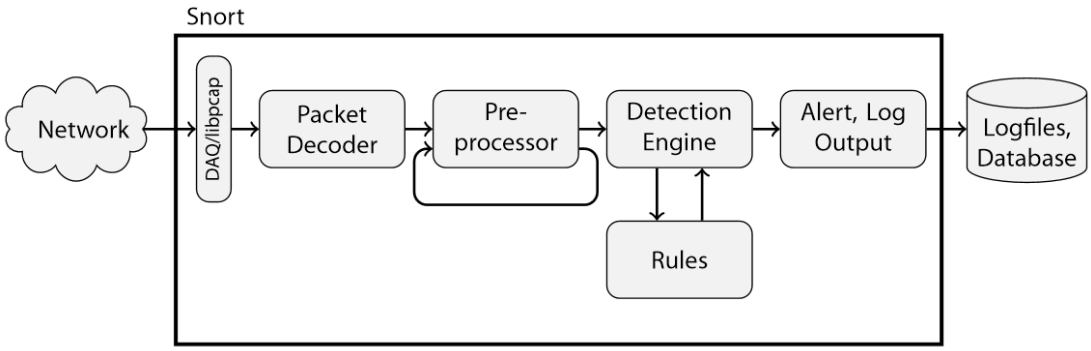



**Figure 8: Snort alerts logged to MySQL database**

[2] https://www.Snort.org/
[3] http://www.tcpdump.org/

[4] https://nakedsecurity.sophos.com/2010/07/27/large-piece-malware/

**User-generated alerts**

Through using social engineering tricks, cyber criminals send malicious spear phishing emails, which appear to be sent by a trusted source, to compromise the assets of individuals and organizations. In [12] the project researchers at Hogeschool Rotterdam studied users' ability to perceive, identify and react upon email (spear) phishing attacks. Inspired by the objectives of the CyberDEW project, the researchers sought ways of enabling users to detect and react on email (spear) phishing attacks collectively and cooperatively. Based on a survey of user centric solutions they integrated a number of promising solutions to harness the collective intelligence of users in a corporate environment. The paper reports on the design of the integrated solution and on the user study in three experimental steps for evaluating the devised solution. The preliminary study showed that the combination of active warnings, embedded education and reporting is promising. Therefor, it is for future work to collect and combine user generated weak alerts at CyberDEW processing nodes in order to deliver stronger alerts to the detection process of (multi-stage) cyber attacks.

### 3.1.4. Prototype

The developed prototype shows the main elements of a solution to detect a Nitro-attack (see figure 9). As stated before the CyberDEW architecture is a hierarchical, scalable and flexible approach to collecting and representing belief that certain attacks are ongoing.

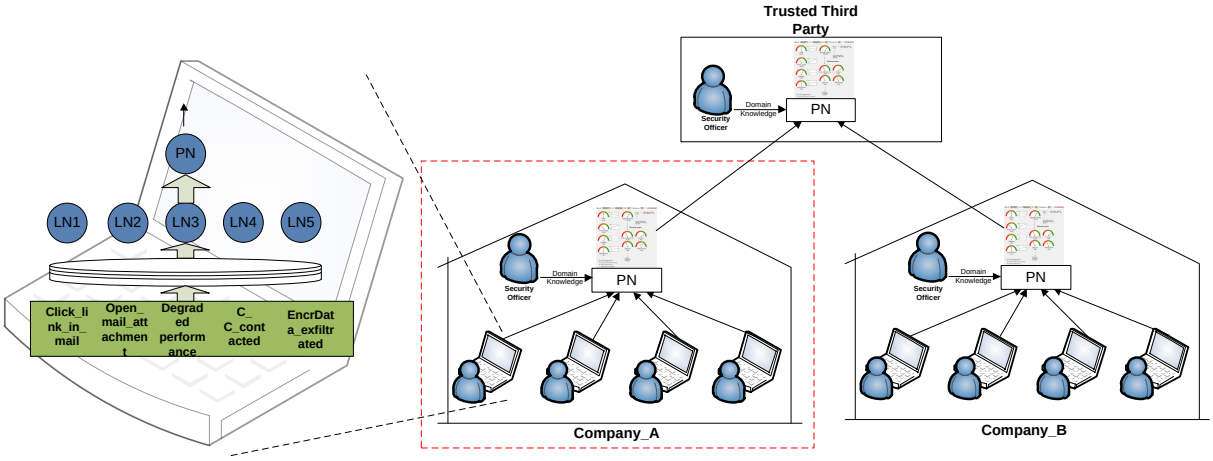



**Figure 9: Prototype, overview**

The red-dotted box in figure 9 shows the section that was prototyped. It represents a company (A) utilizing CyberDEWs Nitro-attack detection mechanism. It consists of two layers of Processing Nodes: the user-level and the security-officer level. The user-level employs the bespoke AR-HMM for generating a belief. Each user (laptop, desktop etc) has sensors installed to detect for instance the clicking of a link in an e-mail or the opening of an e-mail attachment. Other sensors, such as C&R contacted (C_C_contacted) and encrypted data exfiltrated (EncrData-exfiltrated) are typically placed at the network level. The security officer (SecOff) level will typically utilize other algorithm(s) (e.g. counting the number of times a belief has exceeded a certain threshold). Note that the security officer should have the ability to modify algorithms and /or their parameters on basis of actual (cyber threat)) knowledge. A dashboard type of graphical user interface (GUI) was created to assist the security officer in reading and interpreting the various (4) Nitro-attack beliefs.

### 3.1.5. Test results

Behavior of the prototype was tested using a setup that emulated the events created at the user-level (see figures 10 through 13). For this purpose a (programmable) event-generator was created. This generator allowed to generate events for each user.

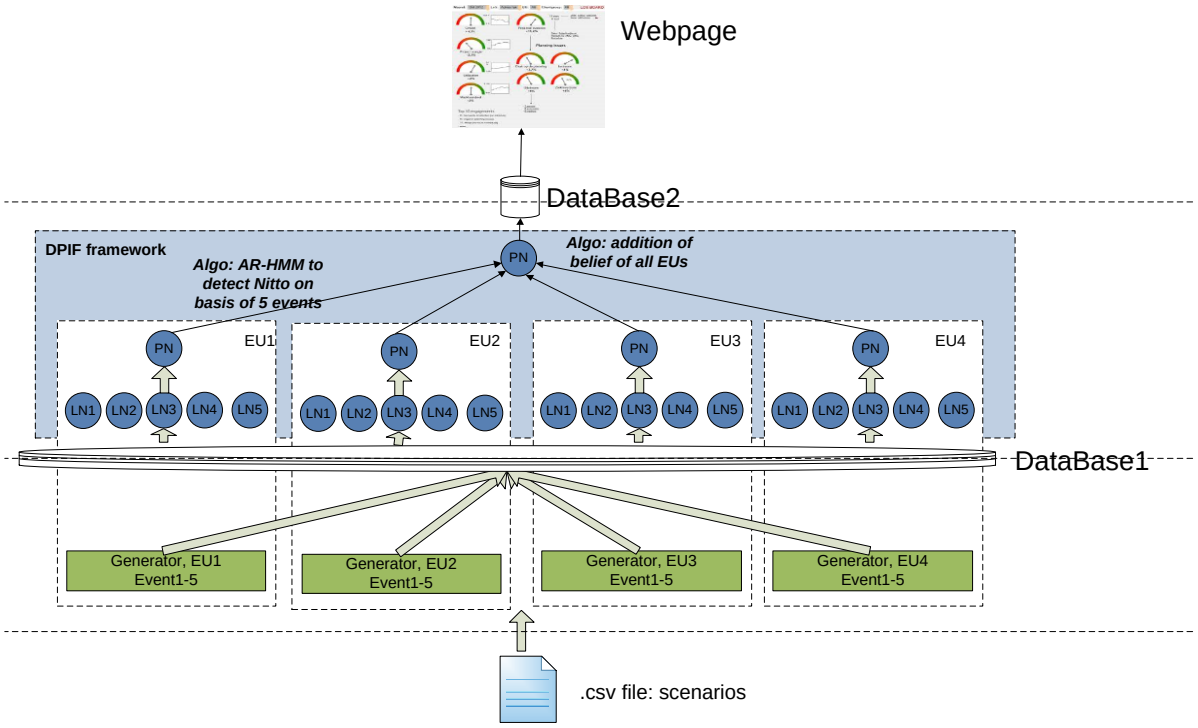



**Figure 10: Prototype setup**

At the top the web-page, providing an intuitive interface to the security officer is shown. The DPIF-framework runs 4 instances of the AR-HMM algorithm. The 4 beliefs are collected by another algorithm could be anything but in this case is a mere threshold-counter. The setup was created by means of 3 laptops, each featuring one or more aspects.

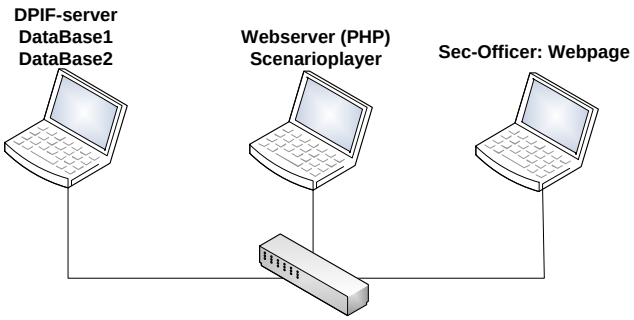



**Figure 11: Realization of prototype**

The various scenarios can be programmed (.xls) and played thus emulating the sequencing of the various weak-signals. Note that also non-ideal sequences are modeled. An example is shown below in figure 12.


**Figure 12: Sequence generator, multiple users**

The x-axis shows time, the y-axis shows which event is activated (circle). The window the security officer watches is shown below in figure 13.


**Figure 13: Security Officer window**

The window shows (speed-meter) for each of the 4 end-users the instantaneous belief that a Nitro-attack is ongoing. On top the cumulative number of beliefs that were > 50% , over all of the 4 users is shown. At the bottom the value of the belief of user 1 as a function of time is shown. Various sequences have been tested. It was shown that also non-ideal sequences (with noise, or with elements missing) generated an, albeit lower, belief.

### 3.1.6. Conclusions, considerations and future work

It was shown that the AR-HMM (see 3.1.5 ) is able to detect multi-stage attacks that produce a specific order of observable symptoms. In addition, AR-HMM are still able to detect such attacks under missing and parasitic observations.

- Consideration: AR-HMM, like any algorithm, requires parameterization on basis of knowledge about the attack. The expertise of the domain expert about a certain attack needs to be translated into AR-HMM parameters. It is well known from Bayesian network literature that parameter elicitation is a challenging task. Nevertheless, through experiments it can be shown that the detection performance of the AR-HMM for multi-stage attacks is

marginally influenced by deviations in the parameters up to a certain degree. This means that the AR-HMM is tolerant to a certain degree of error (deviations from the true parameters) in the parameterization.
- Future work: The applicability of AR-HMM for the detection of multi-stage attacks can significantly be improved if existing deterministic rules for the detection of similar attacks can be automatically transformed into AR-HMMs (softening of deterministic rules). Given the tolerance of AR-HMM in parameterization errors this automated conversion might be possible. Further experiments are required to substantiate this.

The alerts produced at leaf-nodes (like the (spear) phishing alerts issued by end-users) can generally be considered as weak alerts because they individually have a rather low quality (e.g., rather high false negative and/or positive ratios).
- Consideration: CyberDEW processing nodes combine and fuse such weak signals and yield stronger indications of cyber attacks according to a strategy. This strategy can depend on the objectives (e.g., improving the collective false positive/negative ratios of homogenous alerts, improving the quality of the heterogeneous alerts as in the case of multi-stage cyber attacks) and on the quality of the input weak alerts (e.g., the relative magnitude of false positive and negative ratios), see [12] and [14] for specific examples.
- Future work: Combing homogenous weak signals, issued by multiple (distributed) sensors, as CyberDEW processing should further be investigated. In every deployment and usage setting, the CyberDEW data fusion process should be designed and customized according to, for example, the available sensors, the quality of sensory signals, and the objectives of data fusion.
- Consideration: DS approach shows promising capabilities in modeling uncertainty.
- Future work: It is worthwhile to design, realize and evaluate the DS approach in detecting multi-stage cyber attacks (like the Nitro attack). A systematic comparison between the performances of the DS and AR-HMM approach can be instrumental to guide practitioners in effectively customizing CyberDEW to specific deployments settings.

Detection of the Nitro-attack is based on very generic events (opening mail-based attachment, clicking link in the mail).
- Consideration: It is imaginable that real-world systems, where most of bespoke weak-signals do not have a link with a (Nitro-)attack will insufficiently trigger. Extension of the set of weak-signals with a "strong indicator" (like an event being generated at the initialization of the PIVY RAT) would be expected to strongly enhance the Belief function.
- Future work: Algorithms like AR-HMM should be extended to take into account the relative weight of certain weak-signals. Some weak events required the design of new sensors (ClickLinkInMail, OpenMailAttachment, Degraded-Performance). It has to be seen to what extent the creation of user-level sensors is useful and can be picked up by IT-system vendors.

## 3.2. Economic perspective

A demonstration setup generating the Belief value that a Nitro-attack is in progress was created. More general it is stated that technology was developed that probabilistically detects the presence of attacks on basis of knowledge about the effects (weak signals) coming with such attacks. This technology could be an element of the next generation SIEM that collects and shares information. The commercial potential of the technology has been acknowledged, further development is required to shape this into a value proposition.

At the end of Phase1 the idea was to have created a product that would detect otherwise hard to detect patterns. Detection would be based on knowledge about the distributed, weak signals emanating from the attack. On basis of applying that knowledge, weak signals will result in local beliefs. At the next higher level beliefs will be combined with beliefs (about other attacks) and context knowledge (e.g. OSINT). This way an indication if a certain attack or group of attacks is active will be created.
In the course of the project bespoke idea has been shared with potentially interested parties (VNB, ING, IBM, KPN, ThalesGroup, ASP4ALL). From the resulting discussions the following observations were made,

a) The concept of distributed detection combined with probabilistic analysis is deemed to be elegant. The feature of a secure connection with a Trusted Thirds Party as well as the protection of data are crucial factors. The small size of the project team, related to the complexity of the problem at hand (detection of complex & distributed attacks) is called problematic.
b) Detection of attacks (and particularly of Advanced Persistent Threats) is extremely hard. The reasons: the (elements of the) attack are so widely spread over place and time that, if there are already specific phenomena, these almost certain will be completed hidden in the noise.
c) The business model (sign up for a service where the capability of probabilistic detection of attacks is offered) relies on a continuous update to include the latest attacks/attack patterns. Sustainingly offering such service demands a large and broad product team.

d) Products (from large players) offering distributed detection are already being offered. CyberDEW is unique in the aspect that it offers a flexible and scalable framework which yields the probability that a certain attack(s) is taking place. Technology applied in CyberDEW is potentially interesting for complementing existing SIEM functionality.

## 3.3. Contribution to the solution of societal issues

Situation awareness gains importance these days, specially in the area of cyber security. The CyberDEW platform offers an infrastructure for exchanging cyber security related information and for reasoning about that exchanged information. In addition to detecting the probability of (0-day) cyber attacks one can use the CyberDEW platform for situation awareness through analysis and visualization of cyber security trends (for example, how fast the number of botnets grows in a specific sector or country, and what the impact of a mitigation mechanism is on the cyber security resilience of a sector or an organization). The knowledge gained in executing the CyberDEW project is expected to be shared in other projects within the Ministry of V&J that are aimed at situation awareness (for example, which information can be shared among cooperating partners, and what the information privacy and sensitivity implications are in sharing this information)

## 3.4. Other effects on society

Two lecturers of Rotterdam University of Applied Sciences were involved in execution of the project. They carried out applied research and development, and thereby gained (new and multidisciplinary) knowledge within the project. The insight gained and the new problems and solutions identified have enabled the lecturers to define new assignments (i.e., projects for graduation and for minor Enabling Networked Society) and to prepare new lesson materials for their students. One student of the university carried out his graduation project within TRT-NL in an area related to the CyberDEW project. The knowledge gained through the CyberDEW execution benefitted various education disciplines of the university, like Informatics (topics on data science, data analytics, data visualization, and machine learning), Technical Informatics (topics on infrastructure security, intelligent malware detection and prevention, firewalls, distributed security, threat analysis, and data processing and fusion) and Media Technology (topics on human machine interaction for involving users and experts in the defense shield). The project has resulted in two papers [12] and [14] (to be) submitted to two conferences.

## 3.5. Governance, legislation and Intellectual Property.

Within a distributed intrusion detection system there are two types of privacy to be considered. The first is the privacy of the participants. Organizations may not be willing to participate and share information if this information is privacy-sensitive and may give a competitor an advantage. There are technical solutions to deal with this type of privacy which are discussed in section 3.1.3.5. The second type of privacy is that of the cyber attackers. Sharing information about the possible threat of a cyber attack cannot be done effectively without sharing the identity (i.e. the IP-address) of the potential attacker. In order to prevent the attack some idea of where it originates is required. In its basic form a distributed intrusion detection system very soon implies sharing suspicious IP-addresses with other organizations in a fast and automated fashion, a practice resembling blacklisting. In the Netherlands blacklisting is subject to privacy legislation and is monitored by the Data Protection Authority (DPA, in Dutch: College Bescherming Persoonsgegevens, CBP). All blacklists need to be registered with and approved by the DPA. Using a blacklist without the approval of the DPA is not allowed. See https://www.cbpweb.nl/nl/onderwerpen/zwarte-lijst.

However, there are some complications. The current legislation is very much aimed at the physical world and not the virtual world. For example, to blacklist someone there needs to be a very strong suspicion of guilt including hard evidence and suspects need to be informed of being blacklisted. But in order to be effective in the virtual world, a distributed intrusion detection system needs to blacklist and share within a couple of seconds. It is unlikely that within this timeframe all the conditions for blacklisting can be met. Thus before commercially developing this product, the developer will have start talks with the DPA. Another complicating factor may be that privacy legislation is country specific. This may complicate sharing across borders. The DPA may advise on this problem.

# 4. CONCLUSIONS AND COMMERCIAL OUTLOOK

CyberDEW technology provides a technology for the next generation SIEM that collects and shares information. In the SBIR CyberDEW Phase 2 project various technologies have been combined and further developed. This has resulted in the design of an architecture capable of generating belief values at multiple hierarchical levels, that indicate a potential ongoing attack Generation of the belief values is based on knowledge about the inter-relation (in space and time) of the weak signals that emanate from a cyber attack. For demonstration and learning purposes sensors and algorithms were developed to detect the so called “Nitro-attacks”. The processing of sensor-information, together with expert-knowledge is performed in a flexible and scalable framework, DPIF. Information is securely transmitted using the Martello shared data space. In order to bring the concepts & technology to the product-level the following extra work is required,

- Algorithms such as the developed Auto-Regressive Hidden-Markov-Model (AR-HMM), require an intuitive interface that enables domain-experts to easily enter knowledge (intuitions, knowledge about unequal relevance of weak-signals) and translate that to algorithm parameters.
- Relevant new weak-signal generators, picking up evidence at the end-user level, need to be developed.
- Sharing of security-related information needs to be brought into practice using the technology offered by Martello.
- Standard interfaces need to be developed for facilitating bidirectional knowledge transfer between CyberDEW technology on one hand and existing SIEM-technology on the other hand (e.g. STIX, TAXII).

The CyberDEW concept relies on knowledge about cyber attacks and the weak signals emanating from them. To offer a service that offers algorithms (qualitative and quantitative) for detecting cyber attacks requires an organization for retrieving this knowledge. The productized CyberDEW technology will most likely be sold as a license to SIEM-vendors.

## 5. REFERENCES


[1] E. Chien and G. O'Gorman, "the Nitro attacks", *Technical Report, Symantec*, 31 Oct. 2011.

[2] J. Schafer, (2010). "Security collaboration best practices guide," *whitepaper by InterAction* [Online]. Available at http://www.eisf.eu/resources/item.asp?d=3228

[3] J. Kolter, T., Kernchen and G. Pernul, "Collaborative privacy - a community-based privacy infrastructure," *In Proceedings of IFIP Information Security Conference Emerging Challenges for Security, Privacy and Trust (SEC)*, Volume 297, pp. 226-236, 2009.

[4] M. Linden, D., Simonsen, A. Solberg, I., Melve and M. Tveter, "Kalmar Union, a confederation of Nordic identity federations," *In Proceedings of TERENA Networking Conference (TNC),* Malaga, Spain, 2009.

[5] C.V. Zhou, C., Leckie and S. Karunasekera, "A survey of coordinated attacks and collaborative intrusion detection," *In Computers and Security*, 29 (1), 124-140, (2010)

[6] R. Bye, S., Albayrak and S.A. Camtepe, "Collaborative intrusion detection framework: characteristics, adversarial opportunities and countermeasures," *In Proceedings of International Conference on Collaborative methods for security and privacy (CollSec),* 2010.

[7] R. Bye, S.A. Camtepe and S. Albayrak, "Teams rather than individuals: collaborative intrusion detection," *In Proceedings of Security Research Conference on Future Security*, Berlin, 2010.

[8] K. P. Murphy, "Dynamic Bayesian network: representation, inference and learning," PhD dissertation, University of California, Berkley,,USA, 2002.

[9] S. Choenni and H.M. Blanken, "A Dempster-Shafer approach to physical database design," *In Proceedings of the 10th International Conference on Artificial Intelligence: Methodology, Systems, and Applications (AIMSA'02)*, pp. 111-121, London, UK, Springer-Verlag, 2002.

[10] A. Penders, G. Pavlin and M. Kamermans, "A collaborative approach to construction of large scale distributed reasoning systems," *International Journal on Artificial Intelligence Tools,* 20(6), pp. 1083-1106, 2011.

[11] G. Pavlin, M. Kamermans and M. Scafes, "Dynamic process integration framework: toward efficient information processing in complex distributed systems," *Informatica* 34(4), pp. 477-490, 2010.

[12] Foroushani, V.A. ; Comput. Eng. Dept., Yazd Univ., Yazd ; Adibnia, F. ; Hojati, E. : Intrusion detection in encrypted accesses with SSH protocol to network public servers in Computer and Communication Engineering, 2008. ICCCE 2008. p. 314 - 318

[13] N. Stembert, M.S. Bargh, S. Choenni and F. Jansen, "Towards an integrated solution to prevent email (spear) phishing attacks by enabling collaborative human intelligence", *Submitted to the 23rd European Conference on Information Systems (ECIS)*, Münster, Germany, 2015 (under review).

[14] R. Cornelisse, M.S. Bargh, S. Choenni, J.P.G. Sleddens, D. Moolenaar and L.V. de Zeeuw, "On detecting anomalous behaviour in NetFlow data," (under preparation).