FFI-rapport 2009/00319

Oasis demonstration – secure information exchange between military and civilian systems

Raymond Haakseth and Morten Andreassen (Thales Norway AS)

Norwegian Defence Research Establishment (FFI)

11.02.2009

FFI-rapport 2009/00319

1086

P: ISBN 978-82-464-1539-0
E. ISBN 978-82-464-1540-6

Keywords

Nettverksbasert Forsvar

Tjenesteorienterte arkitekturer

Informasjonssikkerhet

Demonstrasjon

Approved by

Anders Eggen	Project Manager
Vidar S. Andersen	Director

English summary

In the fall of 2008 the FFI project 1086 Secure Pervasive SOA participated in a demonstration during the Oasis final event. Oasis (Open Advanced System for dISaster & emenergency management) was a four year research project funded by the EU funded project under the FP6 Information Society Technologies program that ended in December 2008. FFI was not a part of the Oasis consortium but was invited by one of the partners, Thales Norway AS, to take part in the demonstration. This document describes this demonstration where the main focus was information exchange between military systems and civilian emergency response management systems.

At a high level the challenge of such an information exchange can be divided into two, translation between data models and secure exchange cross security domains. The need to do translation comes from the fact that the systems did not use identical data models. The need to have secure exchange of information is based on the requirement to minimise the risk of information leakage. The solution outlined in this document, and showed during the demonstration, makes use of confidentiality labels that are bound to the information by a digital signature. The label is used to signalise the sensitivity of the information, which is checked against a policy by a guard before leaving the security domain. In addition this document and the demonstration described here involved use of a security infrastructure for identity management and tools for policy specification and for complying with these. The concept of role based access control was also introduced in this demonstration.

This document provides a description of the challenges and solutions demonstrated during the Oasis final event. This includes a description of the technical solutions that were designed and implemented for the demonstration. The technical solution created was a cooperative effort between FFI and Thales Norway AS. This document also provides an evaluation of the solutions. A brief description of the work and artefacts produced by the Oasis project is also included. It is assumed that the reader of this document has basic knowledge of Service Oriented Architecture (SOA) and Web Services.

Sammendrag

Høsten 2008 deltok FFI prosjekt 1086 Sikker gjennomgående SOA på en demonstrasjon i regi av forskningsprosjektet Oasis (Open Advanced System for dISaster & emergency management). Oasis var et fireårig prosjekt finansiert gjennom EUs sjette rammeprogram som pågikk ut 2008. FFI ble invitert til å delta i den siste demonstrasjonen som prosjektet avholdt av en av partnerne, Thales Norge AS. Dette dokumentet beskriver denne demonstrasjonen. Hovedfokuset for demonstrasjonen var utveksling av situasjonsdata mellom militære systemer og sivile krisehåndteringssystemer.

Grovt kan man dele utfordringene med en slik deling av informasjon i to, oversetting mellom dataformater og sikker utveksling av data mellom sikkerhetsdomener. Systemene som utvekslet informasjon brukte ikke samme dataformat, en oversettelse måtte derfor implementeres. Når det gjelder sikker utveksling av data mellom domener, er det risikoen for å slippe ut informasjon som er av feil gradering en må forhindre. Løsningen som er skissert i dette dokumentet forutsetter at en merker all informasjon med sensitiviteten og dette merket bindes til informasjonen med en signatur. Informasjonen kan da sjekkes av en beskyttelsesanordning (guard) mot et gitt regelverk (policy) før den forlater domenet. I tillegg beskriver dette dokumentet bruk av en sikkerhetsinfrastruktur for blant annet identitetshåndtering og spesifisering av regelverk. Bruk av rollebasert aksesskontroll er også introdusert.

Dette dokumentet beskriver demonstrasjonen ved å gi en oversikt over de problemstillinger en ønsket å belyse og mulige løsninger som ble tatt fram i prosjektet. Dette inkluderer også en beskrivelse av de tekniske løsningene som ble designet og implementert for demonstrasjonen. De tekniske løsningene ble tatt fram av FFI og Thales Norge AS i samarbeid. Dokumentet gir også en evaluering av hvor godt de tekniske løsningene løser de gitte problemstillingene. En introduksjon til arbeidet som er gjort i Oasis prosjektet er også inkludert. Dokumentet forutsetter at leseren har noe forhåndskunnskap til konseptet tjenesteorientert arkitektur (SOA) og Web Services.

Contents

Preface

The FFI project 1086 Secure Pervasive SOA has as one of its important methods of work experimentation and development of demonstrator software. These efforts provide a bottom-up approach to better test and verify the theories and hypotheses provided by the project. In the fall of 2008 one such opportunity for demonstration appeared, the Oasis Final Event. This event represented the end of a four year long EU funded research project investigating how civil emergency systems can be interconnected.

FFI was not one of the partners in the consortium making up the Oasis project. We were however invited to join the demonstration by Thales Norway AS that was one of the partners. Having worked together on previous experiments and demonstrations they noticed the close resemblance between what Oasis was doing on the civilian side and the problems we were trying to solve on the military side. FFI accepted the invitation since this gave us the opportunity to test our software in a combined military and civilian setting. Information exchange in such an environment is perhaps the most challenging and important scenario that we can use for testing our ideas of secure cross domain information exchange. This also gave us an opportunity to test and discuss our thoughts and ideas on a different audience than we normally do.

This demonstration is so far the last in a series of similar efforts including the Coalition Warrior Interoperability Demonstration (CWID) in 2006 and 2007 [2;3;10]. The actual demonstration took place in parallel with another experiment conducted by the project, the “Multi network II” exercise documented in [6]. The Oasis demonstration took place in Versailles France.

1 Introduction

The most important tenet of Network Enabled Capability (NEC), and its Norwegian equivalent Network Based Defence (NBD), is providing the correct information at the correct time to the correct user. In order to reach this level of ambition, the sharing of information needs to be both fast and dynamic. One of the most substantial obstacles to achieve this today is different security domains and the lack of mechanisms to share information between them. The level of interaction between domains may vary, and typically moving information from a low to a high side is allowed when using for instance a diode, but the other way around requires a manual review and release. At the same time Service Oriented Architecture (SOA) has been pointed to as a key enabler of NEC. SOA is an architectural principle based on loose coupling between services and service consumers, and uniform description of services which can be discovered on an ad-hoc basis. Previous work performed at FFI shows that the use of SOA in combination with object level security, including confidentiality labels and automatic guards to ensure a secure information exchange between domains, can contribute to solving the information exchange challenges within NEC.

Providers and consumers of information are not limited to military personnel or systems, exchange of information with civilian stakeholders are also becoming increasingly important. The need to both collect information from civilian sources and share information with civilian systems is foreseen to increase in the years to come. Integrated logistics is one of the many use cases that are highlighted for this capability. Another is information sharing with civilian emergency response agencies. In an emergency situation, information provided by a military system might be crucial for the performance of the response. This does however place strict requirements on security as the different domains have different confidentiality criteria. From a technical perspective a civilian system is just another provider or consumer of information. However from the policy perspective there is a larger challenge.

This document describes a demonstration of mutual information exchange between military and civilian systems. In addition to showing the technical solutions to ensure secure exchange of information, this demonstration also showed the mutual benefit of this. The demonstration was put together by the Oasis research consortium. The demonstration was an effort performed by five different companies. FFI and Thales Norway AS provided the military systems and the surrounding security solutions. The Oasis systems were developed by BAE Systems[1] and Elsag Datamat[2]. The scenario used in the demonstration was developed by Cranfield University[3].

It is important to note that all security concepts and software described in this document are experimental and should be treated as such. No effort was made to formally verify or certify the security functionality.

[1] http://www.baesystems.com (2008)
[2] http://www.elsagdatamat.com (2008)
[3] http://www.cranfield.ac.uk/ (2008)

This document is structured as follows; Section 2 provides a description of the Oasis consortium and the artefacts produced. Section 3 provides a more in-depth description of the objectives defined for this demonstration. The scenario used during the demonstration is outlined in Section4. Section 5 gives a description of the software used and Section 6 describes the actual execution of the demonstration. Finally, this document is concluded with an evaluation in Section 7 and a summary in Section 8.

2 Oasis

The Open Advanced System for dISaster & emergency management (Oasis) is an EU funded research project focusing on disaster and emergency management systems. The project is addressing the strategic objective "Improving Risk Management" of the European Commission FP6 Information Society Technologies program. The project started in 2004 and ended in 2008. A consortium consisting of 14 partners, both industry and academia, from nine countries made up the project participants. In addition, national representatives from different civil protection agencies composed a strategy group. This group functioned as a reference group and ensured that the interests of the end users were taken into account.

The overall goal of the project was to develop an information technology framework that will provide a basis for future disaster and emergency management systems. Use of an open and flexible architecture combined with existing and proposed standards was the key to developing the framework. This framework should enable the cooperation between units from different agencies ranging from the local, regional, national, to the international level. In the end the purpose of the framework is to enable information exchange in order to enhance situational awareness between players in a disaster or emergency situation.

Through the lifetime of the project several experiments and trials were performed in order to evaluate the products of the research group. In addition a final event was held at the EADS premises at Elancourt, France in October 2008. This event gathered over hundred participants who had the chance to watch demonstrations and get some hands on experience by trying out some of the systems.

More information on the Oasis project and the involved partners can be found on the project internet site[4]. As a final note, the Oasis project as described here must not be confused with the OASIS (Organisation for the Advancement of Structured Information Standards) standardisation organisation. For the reminder of this document when using Oasis the research project presented here are to be understood, and OASIS with capital letters are to be interpreted as the standardisation organisation.

[4] http://www.oasis-fp6.org/ (2009)

2.1 IT Framework

One of the core products emerging from the Oasis Consortium is the IT framework. In essence this is an architecture defined to support interoperability between different emergency management systems, both legacy and future systems based on the Oasis work. The IT framework is a result of the overall goal of Oasis, which is to define, develop and test a generic architecture for the management of emergency operations. The IT framework is to a large extent derived or based on the Tactical Situation Object (TSO), which is presented in Section 2.2. The IT framework also provides definitions of terms used to provide clarity and avoid confusion.

The architecture defined in the IT framework is heavily based on the concept of Service Oriented Architecture (SOA). This was a natural choice for the team as the flexibility and loose coupling between service consumers and producers fitted the interoperability constraints defined. In addition, the approach taken by the Oasis when defining the architecture focused on users and their processes which also fit well with the SOA approach. In total and taken all advantages and disadvantages of SOA into account it was judged as the best available approach since it best meets the requirements.

In addition to the SOA approach, a three-tier architecture has been defined that all Oasis components should comply with as far as possible. The different tiers allow for the separation of the presentation of information, the business process and the core services. This clear separation of concerns provides the flexibility to for instance create different presentations to the end user dependent her requirements and needs.

The three-tier architecture for decomposition of services outlined above provides a separation between views, business processes and core services. In addition to this, the Oasis architecture defines four service layers under what is called the Oasis service layer architecture. Each of these layers contains a set of services that have common or similar functionality. These layers are outlined in Figure 2.1. From the bottom, the first of these layers is named Communication, and provide functionality for establishing connections between components. Oasis Common Operating Environment (COE) is the second layer and provides basic services for the integration of Oasis modules. The third layer, C3I Applications, provides functionality for establishing and presenting a unified view of a situation and provides the main coordination functionality. The last layer, Decision Support Applications, is intended to provide more sophisticated support functionality to the decision makers.

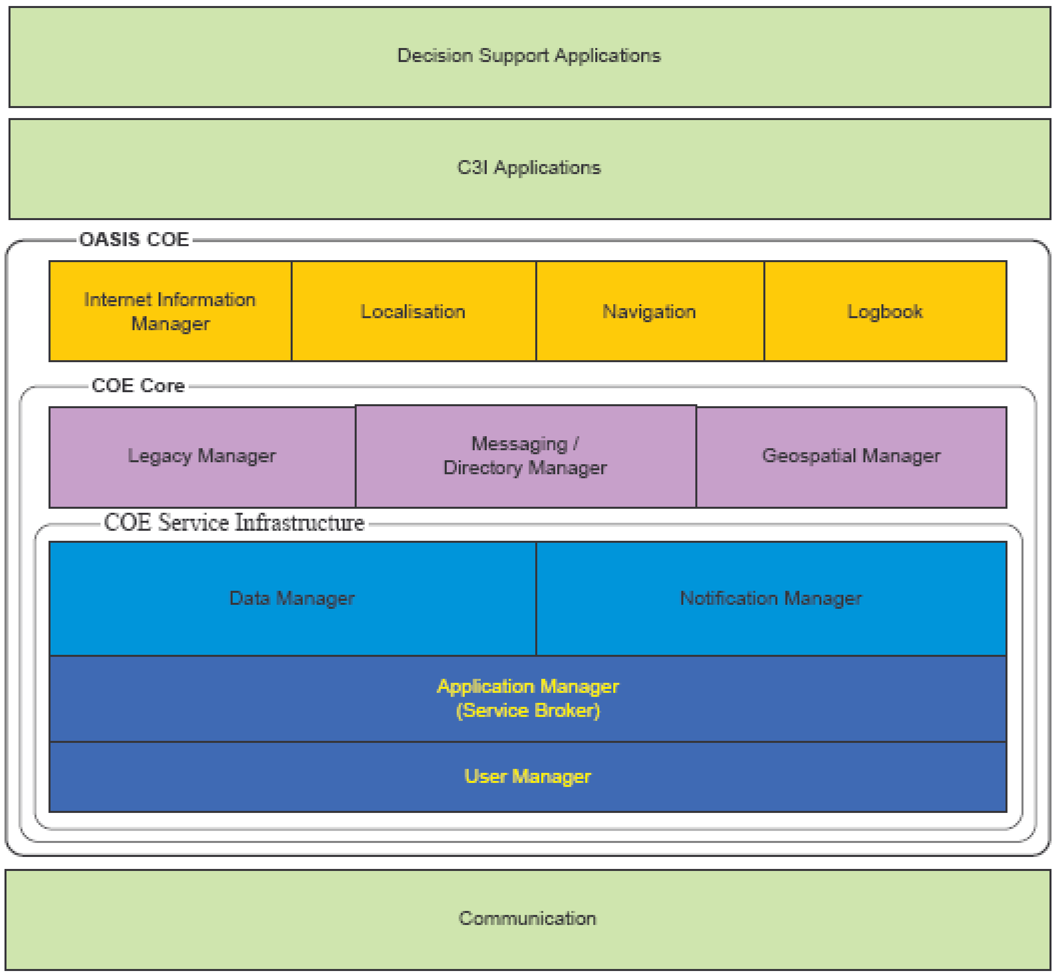

Figure 2.1 The Oasis Reference Architecture for emergency response management systems

The Oasis COE is itself broken into three smaller layers. Support for service invocation is handled by the COE Service Infrastructure layer, while the COE Core layer provides functionality that is common and needed by all application services. The COE support layer provides services not specific to emergency management, e.g. logging.

The IT Framework also makes some consideration about the requirements for security, specifically for confidentiality, trust and trust propagation. Due to the wide range of systems to potentially be connected in an Oasis setting, defining the requirements for confidentiality precisely is difficult, if not impossible. When interconnecting systems from various agencies confidentiality becomes an important issue and the Oasis IT Framework take this into account by allowing the configuration of trust. This configuration would involve defining which applications are allowed to share which information with other parties. In addition, according to the IT Framework this trust is not propagated automatically, but must instead be specified explicitly. The net result of this is that if system A and B has agreed to share information, system B cannot share information from A to C even if a trust agreement exists between B and C. The IT Framework does not go into details on how these mechanisms can or should be implemented.

It is important to note that the Oasis IT framework will evolve as new advances in information technology emerge. The architecture defined in the IT framework is supposedly created with evolution and extensibility in mind.

2.2 Tactical Situation Object (TSO)

The Tactical Situation Object was developed by the Oasis Consortium in order to represent a situation. The overall goal was to be able to provide a Common Operational Picture (COP) that could be shared with all participants in order to achieve a higher level of situational awareness. To achieve this, a mechanism was needed to both accurately represent the observations of the situation and exchange this. The mechanism needed to be interoperable with a wide variety of communication bearers and computer platforms. The information also needed to be shared between different levels within an organisation and between different organisations.

TSO was created by the Oasis Consortium from the bottom up, but not without investigating already existing solutions. Since civilian emergency response operations have some commonalities with military operations and systems, this was a natural starting point for this activity. Among the alternatives evaluated were the work provided by the Multilateral Interoperability Programme (MIP)[5] and the C2IEDM data model in particular. Even though this approach was abandoned due to differentiating requirements, TSO is still inspired by the experience gained by the military. Civilian initiatives were also investigated before defining the TSO. This also includes initiatives from the Organization for the Advancement of Structured Information Standards (OASIS)[6]. None of the different initiatives investigated were found to satisfy the requirements defined by the consortium and the work of defining the TSO were as a result started.

In general TSO can be defined as a structure for storing information related to an emergency situation. This information can be created by a particular observer to record a specific view of the situation or be used to provide a more unified view. The information is in a form that enables it to be transported and shared with other entities. Entities in this case encompass all devices ranging from hand held to more capable computers running command, control and planning applications, and also automated processes. To achieve this flexibility, XML was chosen as the language to encode the TSO information. The choice of using XML enables the information to be exchanged using a large range of mechanisms; a TSO structure can for instance be stored in a file, sent by email, used as a parameter in a remote call, or embedded in a Web Services interaction. If all other means for communication has broken down, TSO information can even be exchanged by memory sticks or a CD. TSO defines many information elements that can be used, it is however important to point out that most are optional. TSO also reuses other standards like the OASIS EDXL-DE [11] for enveloping and transporting TSO messages.

[5] http://www.mip-site.org (2008)
[6] http://www.oasis-open.org (2008)

From a high level perspective TSO is made up by four different types of elements. First is identification data. The purpose of this element is to uniquely identify the message so that it can be unambiguously identified later. This also includes information on the originator, time of creation, organisational level, confidentiality and urgency. In addition, this element can also provide information on relationships to other TSO instances and links to external sources.

The second main element of TSO is event information. This element is used to provide information about an ongoing event. Typically this would include the type of event, extent and criticality, location and associated geographical information, environmental consequences and assessments.

Description of resources is the third element provided by TSO. In a situation multiple resources from different agencies will be available and the purpose of this element is to share the existence of such resources. It is important to have an overview of which resources are available for the operation and which are already active. The description of resources includes information about the availability, location and the capability. A resource in this context can range from fire engines to human resources.

The purpose of the final element, mission description, is to inform others of current, planned and foreseen activities. This information is important in both planning and coordination and can contribute to a more efficient execution. Information typically conveyed on a mission includes status, teams and resources used and which events are handled.

The TSO is defined in two different documents. The first describes the XML schema and the elements described above. In addition, a dictionary is provided that defines the terms used relating to civil emergencies and their respective encoding. The dictionary provides terms to be used in many of the XML elements defined, and TSO does avoid the problem of free text interpretation. The information is well defined and can even be translated automatically to other languages.

The future of the TSO specification is at the time of writing uncertain. As with all research project and consortiums that are limited in terms of time the follow-up is dependent future custodians. The short term plans for TSO is to try to get it standardised, both through the International Standardisation Organisation (ISO) and the European Committee for Standardisation (CEN). The standardisation process often tends to be time consuming and what the result of this effort will be is too early to predict.

3 Demonstration Objectives

The overarching objective for the demonstration as a whole was to provide information exchange between military and civilian systems, and highlight the benefits for users of both the civilian and military systems. It should be mentioned that this demonstration was first and foremost a technical exercise, little or no effort was given to evaluating the value of the information exchange from an organisational perspective.

From the FFI perspective there were two main objectives defined for this demonstration. The first main objective for this demonstration was to show how information can be released from one security domain to another in trusted and secure way. Exchanging information between different security-domains is just one part of the problem, integrating systems in order to utilise the information is the second part. The second objective from the FFI perspective thus becomes system integration. These objectives are presented in more detail in sections 3.1 and 3.2 respectively. In addition, reuse of existing software developed at FFI as far as possible was an important goal underpinning the design and implementation phases.

From the perspective of the Oasis partners the integration of another information source was just one of the important aspects of the demonstration. Other key factors for the demonstration were incident association and the generation of a common operational picture (COP). Incident association showed how mechanisms provided by the Oasis system can be used to aggregate information from several independent organisations managing the same incident. The ability to hold information from all sources and sensors in the COP was also a key point of the demonstration, as well as distribution of this information using the TSO. These objectives are not detailed in this document.

Underpinning all other objectives was the use of Service Oriented Architecture (SOA) and the use of Web Services as the technology of choice for implementation.

3.1 System Integration

Interoperability between systems is the cornerstone when exchanging information. This can roughly be separated into interoperability at the data model level and interoperability of exchange mechanism. For this demonstration both levels of interoperability was important.

The Oasis system use the TSO format as the data model, while the experimental system used and developed by FFI heavily relies on the NATO Friendly Force Identifier (NFFI)[7]. These two data models are in many respects similar, they do however also have important differences. The challenge when translating between data models is to preserve the information as much as possible. It is however not always possible to provide a loss-less translation, and the challenge then becomes to identify the gaps. If the loss of information is too large when translating, the models should be declared incompatible and the attempt to merge them should be abandoned. One of the objectives for this demonstration was thus to find if a translation between the TSO and NFFI formats was feasible.

Interoperability at the level of information exchange mechanisms was the second challenge that needed to be solved in order to fulfil the overall system integration objective. Both the Oasis TSO and NFFI use SOA and Web Services to exchange information with other parties. As a result of this both have defined their own services. Assuming that the challenge of data model translation and integration can be solved there are several paths that can be taken to solve this. The first solution is to implement client functionality on both sides of the exchange. The client functionality would have to include translation as well. The drawback of this approach is that it might require extensive modifications to the systems. Another approach would be to implement a middle tier, e.g. a proxy solution, handling the information exchange both ways. This would function as a mediator and also do translation between the systems. The Oasis architecture also features an interface that can be used to integrate legacy systems. Using this approach the responsibility for integration is put on the owners and developers of the legacy system. Still this is also a viable solution.

3.2 Secure Release of Information

Sharing information between different systems is not always just a challenge of technical interoperability. When considering the confidentiality of information the use of physically and administratively separated security domains is often the case. In order to ensure the confidentiality, no or little information is exchanged between security domains. The strict separation of networks and systems according to confidentiality level ensures that all information is handled correctly. However since a system high domain can contain information of classification up to its own classification level the need to share information arise. Information can in certain cases be lifted from a lower classification domain to higher by using approved one-way diode solutions. The use of manual review and release control and air gaps separation is the most used method for transporting information from high to lower classification domains. Automatic solutions certified for exchange between certain classification levels are also available, but are limited.

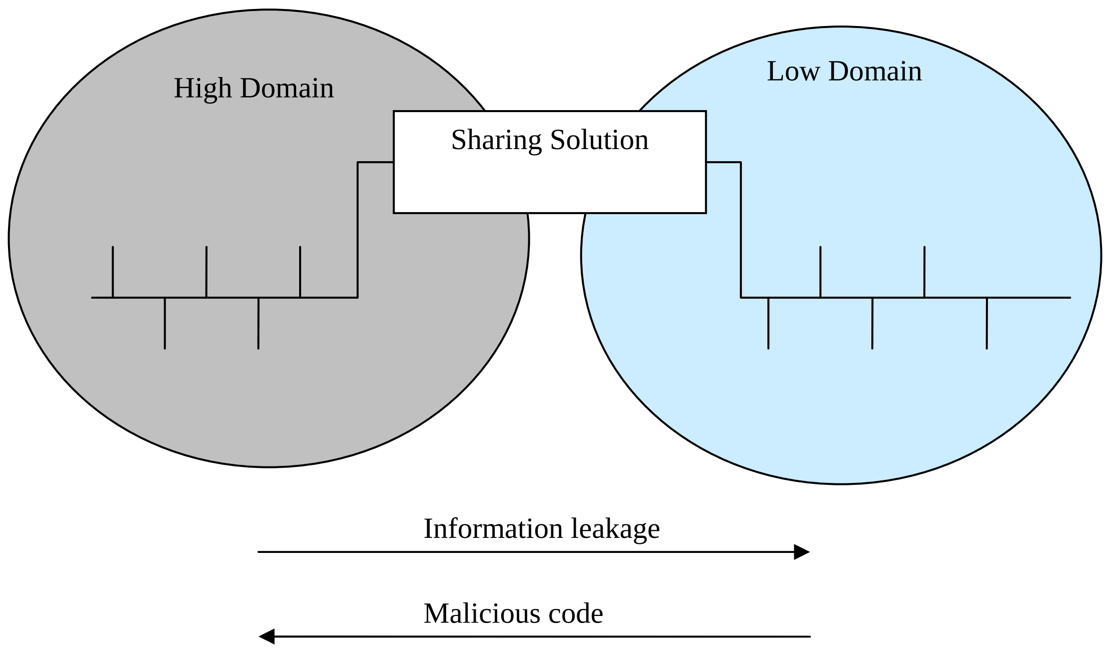

Figure 3.1 The sharing problem

To a large extent, the current solutions for information exchange between system high security domains do not satisfy all requirements. When considering emerging requirements of more information exchange it is clear that for instance the manual review and release process does not scale very well. What is needed is a secure and automated solution for two-way information exchange. Due to the inherent problems of sharing information between different security-domains this is a challenge that is hard to take on. The challenges are shown in Figure 3.1.

There are two generic problems when interconnecting different security domains of different classification levels and enabling two-way information exchange. First, since the high domain can contain information that is not to be shared with the lower classification domain, the need to ensure that no information leakage can happen is an absolute requirement. Second, when information flows from the low to the high domain, the integrity of the systems must be protected. The potential danger is that malicious code and other types of cyber attacks can enter the high domain and this must be avoided as far as possible. These challenges are very high-level and when diving into these, other challenges becomes visible, including issues regarding for instance cross-domain identity management, certificate and cryptographic key distribution and more.

For the demonstration described in this document only the secure release of information was within scope. The argument for only concentrating on one aspect is that the theme of cross-domain solutions is so complex that in order to achieve results the problem has to be broken down and handled separately.

In our approach two important building blocks are used to solve the challenge of secure information release between security domains. First, all information items must be marked with its sensitivity. This is achieved by using a label defined in XML that is cryptographically bound to the information in question. The label identifies the security policy to be used to handle the information, the classification of the information and any other handling categories to be applied to the information. In other words the label represents security metadata for the data/information in question, providing the basis for object level security. In order to be able to put trust into the association between data and metadata, a trustworthy binding is vital. In our solution this is achieved by using a digital signature over both items. The signature provides integrity protection for the metadata, data and the binding between them. As a result none of these three items can be modified without it being detected. It also provides authentication of the originator.

The label and binding to information enables a guard to inspect and filter information passed through it. The guard is thus the second important building block in our solution for the challenge of secure release of information. For the guard to be able to release information it needs to be able to trust the binding between the confidentiality label and data. Trust is ensured through checking the signature, if this does not validate, no further processing is required and the information exchange is stopped. Dependent on which key exchange algorithms are used and the wanted level of trust, certificate validation including revocation checking may also be needed. The guard needs to act according to a policy defining which information is allowed to leave the domain and which is to be stopped. There are several options for both defining and implementing the policy. Earlier versions of the guard have used a proprietary solution with a preconfigured local policy. It is however desirable to have a standardised solution to policy handling as well. The eXtensible Access Control Markup Language (XACML)[7] from OASIS is one alternative for this. In the XACML setting the guard becomes a Policy Enforcement Point (PEP) that can communicate with a Policy Decision Point (PDP). The latter makes the actual access control decision based on the defined policy. This policy is defined using the XACML policy language, which again is defined in XML. The effect of this is that the policy can be changed without reconfiguring the guard and a common understanding of the policy could be achieved thus enabling sharing of policy.

At the same time new mechanisms for handling user identity is emerging. In the forefront of this development is the Security Assertion Markup Language (SAML)[8] defined by OASIS. SAML can in short be described as a framework for defining and communicating user authentication, entitlement, and other attribute information. Single sign-on solutions is one of the use-cases for this functionality where a user authenticates to an identity provider, which issues assertions that can be used when interacting with different service providers. This eliminates the need for the service to implement its own authentication scheme and eases the burden of the user which only has one identity with e.g. one username and password. This might prove beneficial especially in cross-domain settings.

In short the secure release of information objective can be summarised in the following bullets:

- Investigate mechanisms for secure release of information between security domains.
- Investigate the use of object level security and labelling.
- Investigate the use of standards for defining and complying with policy.
- Provide initial investigation to identity management and role based access control.

4 Demonstration Scenario

The demonstration used a simplified scenario in order to show the audience the benefits of both the Oasis system and the information exchange with a military system. The scenario needed to be simplified due to the fact that the intended demonstration time was only 15-20 minutes. The consequence of this is that the scenario did not include a full scale response to the incident described, rather the scenario were tailored to highlight the specific advantages of the systems. The scenario was developed by Cranfield University.

The scenario included several actors. Civil emergency response was provided by police, fire and ambulance. All three represented by their separate control room and mobile units. The military involvement in the scenario included different military units, air traffic control and a tracking system. During execution of the scenario all participants were either simulated or played by an on-site operator. The execution did not use any operational personnel.

[7] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml (2009)
[8] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security (2009)

As expected the scenario was centred on a civil emergency response operation. Below an abridged version of the scenario storyline follows.

1. A NATO exercise is taking place over UK. As a part of this, three F-16s are on the wings flying in formation.
2. One aircraft reports to the air traffic control that it is in trouble. This information is relayed to the Police that inform the other services but take no further action at this time.
3. Air traffic control loses track of the aircraft.
4. The crash is reported from different sources:
    a. Police are notified by the air traffic control of the believed location of the crash and dispatch units to this location.
    b. Ambulance receives call from the public and mobilises units to the reported crash site.
    c. Fire receives another emergency call from the public and mobilise units to the third reported crash site.
5. There are now three different locations reported for the plain crash, none of them are accurate. The military tracking system does however have an accurate position due to the F16s onboard GPS but this information is not releasable at the moment.
6. Police arrives at the location reported by the air traffic control but cannot find the site of the plane crash. They request information from the RAF to get the accurate position of the plane crash.
7. Since the information that can be provided by the military system is very important for the civilian emergency handling it is decided to downgrade relevant information.
8. Information exchange is started between the military and the civilian systems. Information flows both ways, and together a common operational picture is created. It is important to notice that only information that is allowed to pass from the military system is transferred.
9. Now knowing that there is only one crash site and having the exact position of this the civilian resources are directed to this area.
10. The military provides information that the aircraft was carrying weapons. Hence the area needs to be secured.
11. Using the information provided by the civilian systems, the military HQ gives the order to send in personnel to act as scene guards. This information is also given to the civilian emergency response systems.
12. The situation is now getting under control. Cordons are now established, pilots are being transported to hospital and hazmat teams are doing their job.
13. While the situation of the aircraft crash is being handled, other civilian emergency response operations are being handled in parallel.

5 Demonstrator Description

One of the important goals when designing and implementing the software to be used for the demonstration described in this document was reuse of existing components developed at FFI.

Through a series of experiments and demonstrations a large base of ready code was available and it was thus decided to only develop new software when absolutely necessary. This enabled us to produce the software needed with a minimum of effort in both time and human resources.

The architecture of a software system can be viewed in different ways dependent on the anticipated receiver. As a result of this, the next subsections present different architectural views of the software system produced for the Oasis demonstration.

5.1 FFI Software Architecture Overview

The backbone of FFI part of the software produced for this demonstration was a generic demonstrator core developed for experimenting with for instance distributed picture compilation. One of the desired functionality provided by this software component was storing and retrieving NFFI tracks. In addition other software components could be reused for e.g. viewing of tracks, NFFI web services and clients. The guard component is also reused, but some extensions were needed to allow new security solutions. More detailed information on this demonstrator software and how it has been used previously can be found in [2-4].

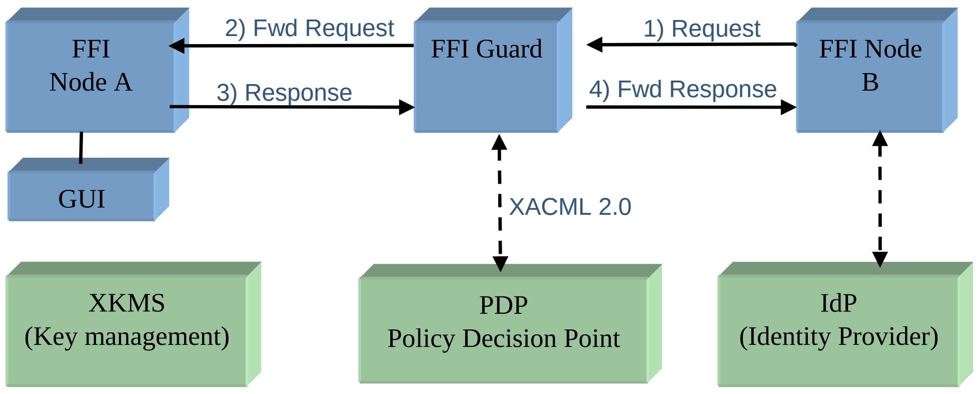

Figure 5.1 FFI Software Architecture

Figure 5.1 outlines the components used by the FFI part of the demonstration. The blue boxes in this figure represent functional components or services and the green boxes represent the supporting security infrastructure and its related services.

The figure depicts the FFI nodes A and B. In this context an FFI node is a software component with features for management and distribution of military track information. A track in this context is describing a military resource with attributes like identity and position. The format for exchanging this information is NFFI. An FFI node can collect track information in two different ways, either by using its local sensors or by using Web Services to retrieve this information from other nodes. Each FFI node implements a request/response Web Service for sharing of track information. This Web Service is defined in the NFFI specifications. Since this service is defined using Web Services, SOAP messages is used for communication. A request for NFFI tracks to the service may include different filters in order to limit the number of tracks returned. Filters may vary, and may include period of time, geographical area, maximum number of tracks, history of tracks and a number of others.

The task of the FFI Guard component is to control the traffic passing through it, both ways. Normally, and also in this demonstration, this component will be located at a security domain border. This component has two primary tasks. First, it should perform access control on requests entering the domain. Second it should check and filter messages leaving the domain. Filtering in this context means removing parts of the message that the user are not allowed to see, or parts that can not be transmitted to the other domain. The current implementation of the FFI Guard component acts as an HTTP proxy.

The supporting security infrastructure is composed by three components. The responsibility of the Identity Provider (IdP) is to authenticate users and provider the users attribute so that other services can take advantage of them. The Policy Decision Point (PDP) will use this information as one of the inputs when performing access control. The XKMS (XML Key Management Specification)[1] component can be used to perform certificate and key operations like validation. The FFI Guard and security infrastructure is further elaborated in Section 5.5.

To conclude the description of the overview of the internal FFI architecture, a word about the information flow. In Figure 5.1 the information flows from FFI Node A to B, and is initiated by the latter by issuing a request. This request is intercepted by the guard who perform access control in cooperation with the PDP. If access is granted the request is forwarded to the services provider, in this case FFI Node A. If access is denied, an error message is returned to the client. The response produced by the service provider is also intercepted by the guard. The guard checks the information and forwards only the items allowed by the policy to the service consumer. This process can also be reversed, i.e. FFI Node A retrieving information from FFI Node B.

5.2 FFI – Oasis Integration

The previous section introduced the FFI demonstrator system and outlined how this works. This section adds to this and introduces the integration between the FFI demonstrator and the Oasis demonstrator. Figure 5.2 provides an overview of the two different systems with integration points and protocols or specifications that is being used for information exchange. Since integration of security solutions between the two systems was not within the scope of this demonstration the components that make up the security infrastructure is omitted for clarity in this figure. Other smaller components have also been omitted for the same reason.

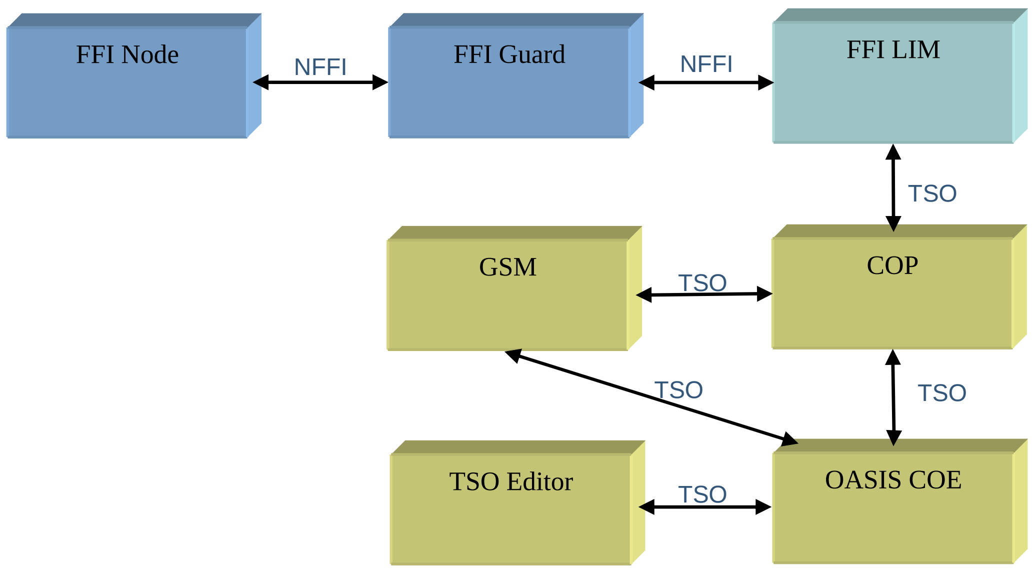

Figure 5.2 Combined deployment view

The three topmost components constitute the FFI demonstrator and are hence the responsibility of FFI. Compared to Figure 5.1 the FFI Node and the FFI Guard components can be recognised. However the second FFI Node is replaced with a component named FFI LIM in this figure. These nodes use the NFFI data format to exchange track information. The yellow components represent the Oasis system. These nodes use the TSO format defined by Oasis to represent and exchange the information. The Oasis components are not explained any further here, the interested reader is referred to the documentation provided by the Oasis consortium at their web site.

Early on in the integration process it was decided to use the same method as the Oasis system to integrate legacy systems. Even though the FFI demonstrator system is not a legacy system, it was decided that the same approach for integration could be used. As a result the Oasis integration API was used to send and receive information to and from the Oasis system. The most important component when integrating the systems is the FFI LIM. From the perspective of the FFI demonstrator this acts as a normal FFI Node, which means that it implements the NFFI request response Web Service and also implements a client for accessing instances of such a service. It also acts as a source of information from the perspective of the Oasis system. The FFI LIM component is thus the natural component in which to place the mapping or translation between the NFFI and TSO data formats. In order to facilitate information exchange between these systems the FFI LIM needs to actively retrieve data from each side and make it readily available for the other. This also includes handling of information looping and the removal of duplicate information if required.

5.3 Data Flow and Processing

This section takes a closer look at the interaction between the different components outlined in the previous section. Figure 5.3 provides a more detailed view of the components and how NFFI and TSO messages are exchanged. In the figure, black arrows represents data sent from the Oasis system to the FFI system and the red arrows represents data from FFI to Oasis.

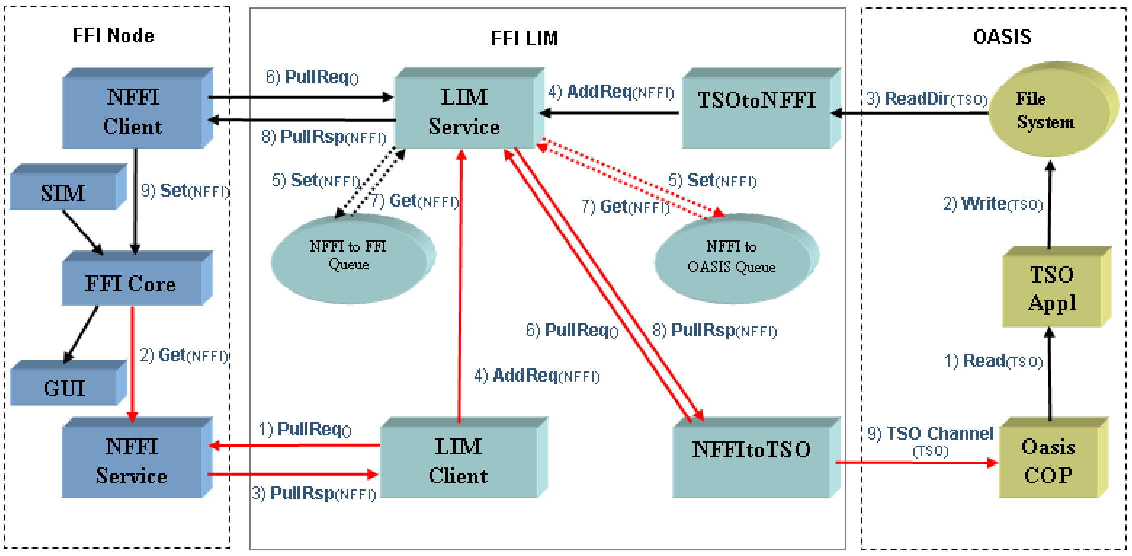

Figure 5.3   NFFI and TSO data flow

In the figure the components have been further broken down into sub-components or applications. We can recognise the FFI Node component on the left side. This has been broken down into five sub-components; these components are still shown in blue. In the middle we find the FFI LIM component that is responsible for the actual integration of the systems. In this figure this components have been broken down into six new sub-components, all coloured turquoise. Finally on the right hand side the Oasis system is shown in yellow. In this figure only the parts of the Oasis system that are relevant for the integration is shown. A description of the components shown in the figure follows below.

The TSO Application, named TSO Appl in the figure, gets TSOs from the TSO channel and writes them to a predefined directory in the File System. The latest TSOs are fetched from the channel at start up, and updates are received afterwards. It uses a predefined naming scheme when storing files in the File System.

TSOtoNFFI is a Java application that fetches new TSO files from the File System. One of the responsibilities of this application is translation from TSO to NFFI. It also performs filtering of TSOs, i.e. discarding TSOs that shall not be transferred. When finished with these tasks the resulting NFFI message is sent to the LIM Service.

The LIM Service is a service provider that implements the NFFI Web Service described previously. Tracks delivered to clients are fetched from two FIFO queues that it is managing, one for tracks destined for the FFI system and one for the Oasis system. It also receives messages from the TSOtoNFFI application. These are stored in the queue destined for the FFI system.

The FFI Node uses the NFFI Client application to fetch tracks from the LIM Service. In this demonstration the NFFI Client is configured to requests at regular time intervals. Received tracks are stored in the FFI Core. It should be noted that the NFFI Client can also be used to retrieve NFFI tracks from sources other than the LIM Service.

The FFI Core component provides storage for track information, and interfaces for both adding and retrieving tracks. Closely related to this component is the GUI which provides a graphical view of the tracks stored in the core. In this demonstration we used the Maria software from Teleplan AS[9].

The Simulator (SIM) application is used to add NFFI tracks into the FFI Core. This application allows us to play a predefined scenario. Other input sources can also be used including via files and the GUI.

The NFFI Service component is a service provider offering the NFFI service for retrieving tracks. It is connected to the FFI Core which provides the track base used. The result set returned from the service is dependent on the filters provided by the client in the request.

In order to retrieve tracks from the FFI Node the FFI LIM implements an application named LIM Client. Basically this is a client for the NFFI request/response Web Service. At regular time intervals it requests tracks from the FFI Node by using the NFFI Service. The tracks received are placed in the LIM Services queue destined for the Oasis system.

The final step in transferring tracks from the FFI Node to Oasis is performed by the NFFItoTSO application. This is a Java application that at regular time intervals fetches NFFI tracks from the LIM Service. It translates the received tracks from NFFI to the TSO format before forwarding it into the Oasis system. The translation results in a TSO file which is forwarded to the Oasis COP via a TSO channel.

As a final remark on Figure 5.3, it should be noted that the security components have been removed for clarity. During execution the guard will be placed between the FFI Node and the FFI LIM, intercepting all messages flowing between them as shown in Figure 5.2.

5.4 NFFI – TSO Mapping

The last, but not least, part of the integration between the two systems is translation or mapping between the two data models used. The NATO Friendly Force Identification (NFFI) is used by the FFI system and the Tactical Situation Object (TSO) is used by the Oasis system. This section summarise the mapping between these models implemented for the demonstration described in this document. The full mapping can be found in Appendix B.

[9] http://www.teleplan.no/ (2009)

NFFI is can be described as an XML based message format that can be used to exchange friendly force tracking information between NATO and national forces. A NFFI message can either consist of an unbound number of tracks or be empty. One track can have five main elements. The only mandatory element is the "positionalData" which among others provides the geographical location of the track. It also provides a description of the system that produced the track and the time that it was created. The combination of these elements, "trackSource" and "dateTime" is used to uniquely identify a track. The four remaining main elements are optional and these are used to for instance describe the track by symbol and name, and to describe the operational status of the track. The track can be further be detailed by using the "deviceSpecificData" and "detailData" elements.

The Tactical Situation Object (TSO) was described in detail in Section 2.2, but a short summary is presented here for completeness. The TSO can in short be described as an information structure to record a view of a situation as seen by a particular observer at a particular time and to transfer this view to another observer. The main elements of such a TSO message include descriptions of events, resources and missions. In addition each TSO message has its individual identification and description of its originator and creation time.

Since NFFI is a pure tracking model it does not provide support for event handling and mission planning. Providing a mapping between these elements from the TSO to NFFI is thus difficult if not impossible. Doing such an exercise would probably at least involve using NFFI elements in other ways than intended. Such usage could result in different and conflicting interpretations of NFFI elements, which is not beneficial in light of interoperability. The event and mission elements of TSO are thus not translated to NFFI. In addition, since the FFI backend is a tracking system it cannot handle this type of information.

The resource description of TSO, called TSO Resource, is very similar to a NFFI track and it was thus decided to implement a one-to-one mapping between these. The most difficult mapping was unique identification. In both models identification is important; in NFFI each track can be uniquely identified and in TSO each resource can be uniquely identified. For NFFI the combination of the track source and time is used as this unique identifier. However, in TSO time stamps are only given for a context. The mapping between was thus decided to be TSO resource id to NFFI track source. The result is that from the NFFI perspective each TSO resource is a track source that reports its own position and related information. And from the TSO point of view each NFFI track is a resource including the current position and related information.

In addition, the NFFI is not using context as in TSO. It was decided that this was to be handled by the integration software, more precisely the FFI LIM component. The other mapping between NFFI and TSO is straight forward and given in Appendix B.

5.5 Security Processing and Information Flow

Until now the main focus of this chapter has been integration of the FFI and Oasis system, and the information flow between them. Integration of the security mechanisms of the two systems was out of the scope of this demonstration and is thus treated separately in this section. The security processing is vital to the FFI system, especially to enable the secure release of information from one domain to another. Figure 5.4 and Figure 5.5 below outlines the components that either perform or make use of the security mechanisms. The figures contain the same components, but arrows are used to display the message flow in the two use cases, information flow from FFI to Oasis and vice versa. From the previous figures the five topmost components can be recognised. The security infrastructure components, coloured green can also be found in Figure 5.1. The XKMS component for key and certificate distribution is however missing from the figures below. This was taken out of the demonstration due to time constraints. Certificates was alternatively distributed and stored in advance at each node. As expected the figure does not show any of the Oasis specific components.

5.5.1 Exchange of information from FFI to Oasis

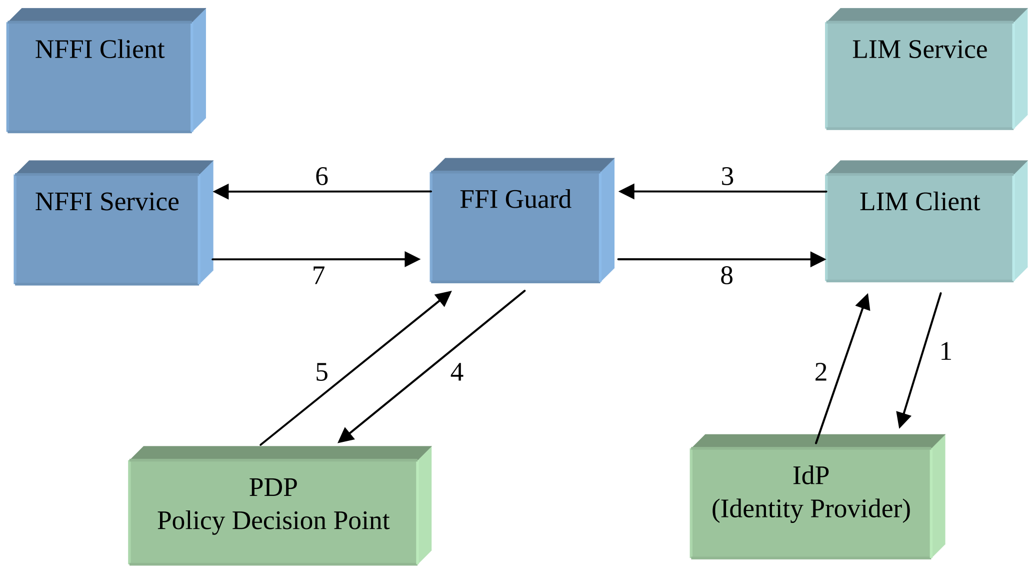

Figure 5.4 Security Processing – exchange of information from FFI to Oasis

Figure 5.4 shows which components are part of and the message flow involved in sending information from the FFI system to the Oasis system. The messages and what components that are generating them are explained below:

1) The interaction is started by the LIM Client component[10] which authenticates itself to the IdP. The IdP can be configured to use a variety of authentication methods, e.g. username and password. In this case authentication was performed using signatures and verifying the clients' certificate.
2) A SAML assertion is returned by the IdP. This assertion contains a set of attributes describing the client, including authentication information. In our setup a role attribute

[10] The information exchange follows the request/response pattern and is thus initiated by the client.

was included in addition to the clients’ certificate. It also includes the time of creation and validity of the assertion. The assertion is signed by the IdP in order to ensure the integrity. The assertion can now be provided by the client to other services as an authentication token.

3) The LIM Client builds an NFFI request message and includes the assertion received by the IdP. The whole message is signed by the LIM Client before it is sent to the NFFI Service via the FFI Guard.
4) The message is intercepted by the FFI Guard, which checks the signature and the assertion. The verification of the assertion includes a signature check. The FFI Guard also verifies that the certificate included in the assertion is the same certificate that has been used to sign the message. This limits the risk of a man-in-the-middle attack, where someone else might use the clients’ assertion. The FFI Guard plays the role of a Policy Enforcement Point (PEP) and needs to contact a Policy Decision Point (PDP) in order to authorise the request. It builds a XACML request message which among others include information from the assertion and the resource that is to be used, i.e. the NFFI service. This message is forwarded to the PDP component.
5) The PDP evaluates the request against the policy which is described in XACML. If this access control fails a deny message is returned to the FFI Guard. On the other hand if access is granted an accept message is returned. The accept message can contain obligations that the PEP must process either before the resource is used or after. In our case the PDP includes a clearance label to be used for filtering after the request has been processed. The content of the clearance label is decided by the clients’ role as described in the assertion provided by the IdP.
6) When the FFI Guard receives the decision from the PDP, it either generates a SOAP fault message and sends this to the client if the access is denied, or forwards the request to the NFFI service if accepted. In this case, it also stores the clearance label to be used later when filtering the information contained in the reply from the service.
7) The request is processed by the NFFI Service and a NFFI return message containing the tracks matching the search criteria is created. All tracks are then labelled according to sensitivity and signed using the mechanisms described in Section 3.2. The message is returned to the LIM Client via the FFI Guard.
8) Before releasing any information to the LIM Client the FFI Guard has to make sure that only information allowed to pass according to the policy is released. This is done by comparing the label of the information with a clearance label. This label is either pre-configured or fetched from the obligation received from the PDP, the latter has priority if present. The clearance label is a confidentiality label that essentially represents the highest sensitivity of information that can be released, e.g. NATO Unclassified. The confidentiality label of the information, in this case NFFI tracks, is compared to the clearance label. If it is of lower or equal sensitivity it can be released, if not it is removed. The filtered message is returned to the LIM Client.

The process outlined above makes use of both Policy Based Access Control and Identity Management. It also describes role as one of the attributes describing the user, in this case the

LIM Client component, providing the basis for Role Based Access Control. The intention here is to provide a more flexible security infrastructure where the policy can be changed easily, the user is provided with single sign-on features and the role the user is playing is vital to the privileges received. It should be noted that the effort described here is not exhaustive and lacks for instance a more dynamic role management system.

5.5.2 Exchange of information from Oasis to FFI

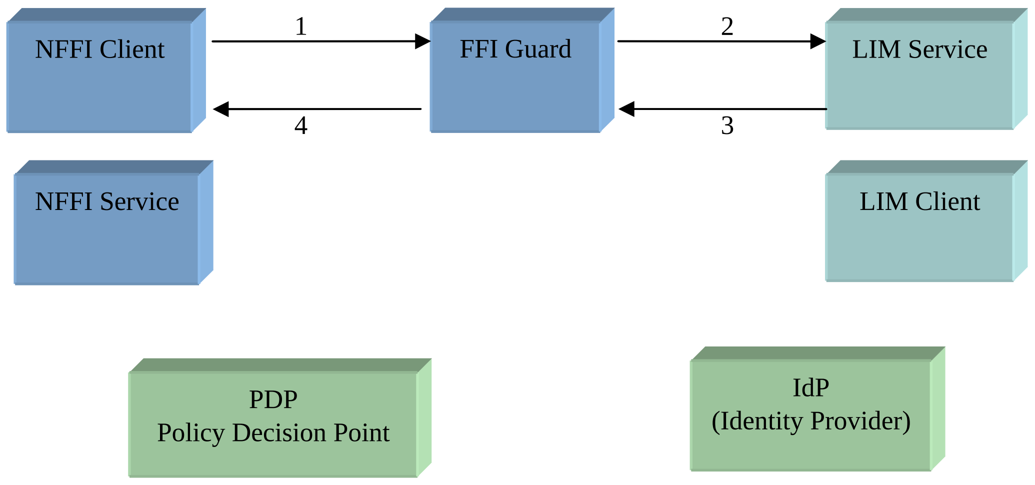

Figure 5.5 Security Processing – exchange of information from Oasis to FFI

Figure 5.5 shows which components are involved in sending information from the Oasis system to the FFI system, and the message flow between them. The messages and the components that are generating them are explained below:

1) The interaction is initiated by the NFFI Client component. It generates a NFFI request and sends this to the LIM Service via the FFI Guard. Before the request message is sent it is labelled with the sensitivity level and digitally signed.
2) The message is intercepted by the Guard which examines the message and performs release control. This process is identical to the one described under bullet eight in the previous description. If the request is releasable to the other domain it is forwarded to the LIM Service, and if not a SOAP Fault is returned to the NFFI Client. Even though the message in question here is a request, it is important to perform release control since these types of messages also can be a source of information leakage.
3) The LIM Service handles the request and returns an NFFI response message. In the setup used during demonstration no security mechanisms was used here.
4) The return message is forwarded to the NFFI client by the FFI Guard without any more processing. The guard deployed here is only concerned with the secure release of information and not to protect the internal domain from for instance virus attacks.

In this interaction only the basic security functions are used, and not the Identity Provider and the Policy Decision Point. This was first of all a choice made for this demonstration due to time limitations and not a more fundamentally based choice. Earlier versions of the FFI Guard used this interaction pattern when releasing information and the task of including for instance an identity provider was not prioritised for this demonstration. The value of both the Identity Management and Policy Based Access Control was shown in the other interaction pattern and this combined with the reuse of earlier work was the basis for this prioritising.

5.6 Implementation, Deployment and Source Code

The implementation of the system followed the description given previously in this chapter, and will not be described in any detail here. It suffices to say that all software was developed using the Java programming language, more specific the Java EE 5 SDK. Earlier versions of the demonstrator and components from this were reused as much as possible. The implementation was a cooperative effort between FFI and Thales Norway AS.

In addition to the use of standard Java EE, a variety of supporting open source frameworks were used and the software was deployed using various applications servers. The security components are largely based on open source software. The implementation of the Identity Provider is supported by the OpenSAML framework[11] and the implementation of the Policy Decision Point is supported by the JBOSS XACML and SAML-XACML frameworks[12]. These components were deployed in a WSO2 Web Services Application Server (WSAS)[13]. In addition, this application server includes an XKMS service that was planned to be used. However, as explained earlier the use of XKMS was dropped due to time constraints. Also the FFI Guard is based on open source software, namely the Apache TCPMon[14]. This is a utility that allows messages to be intercepted and viewed. It is mainly used as a debug tool, but it has been extended to include the necessary security functionality to function as a guard. The Apache Tomcat[15] servlet container was also used for deployment of various web services. And the FFI Core component was deployed in a Glassfish application server[16].

Further details and documentation are available in the source code of each component. For those with access the source code is available in the Subversion repository used by the FFI project 1086 Secure Pervasive SOA. For the software components developed by Thales Norway additional documentation is available in the repository.

[11] https://spaces.internet2.edu/display/OpenSAML/Home (2009)
[12] http://www.jboss.org/ (2009)
[13] http://wso2.com/products/create/wso2-web-services-application-server-wsas/ (2009)
[14] http://ws.apache.org/commons/tcpmon/ (2009)
[15] http://tomcat.apache.org/ (2009)
[16] https://glassfish.dev.java.net/ (2009)

6 Demonstration Execution

The EADS premises at Elancourt, France was chosen as the arena for the two day Oasis final event and thus also for the FFI – Oasis demonstration by the Oasis consortium. The final event was organised as a mix of plenary session and parallel demonstration sessions. In total four different demonstrations was given during the event. The FFI – Oasis demonstration was displayed to the audience four times. Each demonstration session was initiated with a short presentation giving the audience the background information needed. This was followed of the demonstration scenario which lasted 20 minutes, with a following ten minutes question and answer session. In addition, a session was held during the second day of the event allowing the audience to have a closer look at the systems. The demonstration was also captured on video for later display to audience that was not able to attend the Oasis final event. These videos feature screen captures of the various displays used during the demonstration.

During the actual demonstration the scenario outlined in section 4 was played by the two different systems. During which the narrator read the scenario script in order to give the audience the necessary input to understand what was happening. During the demonstration execution two screens were used simultaneously, one showing the displays of the FFI system and one showing the displays of the Oasis system. This allowed the audience to have a view of what information was available in the different systems and actually see the benefit of sharing it. The different departments involved in the scenario are outlined in Figure 6.1.

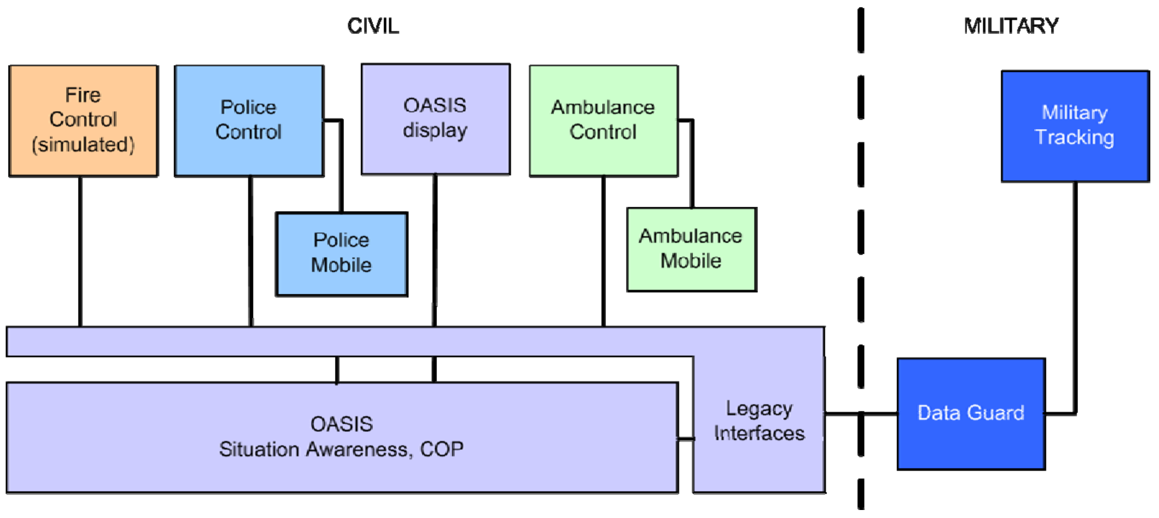

Figure 6.1 Demonstrator Architecture

For the team running the demonstration a two day final testing and rehearsal was performed in the days prior to the final event. During this time the necessary equipment and data networks was rigged, configured and tested as well. It should be mentioned that integration testing was performed earlier in the development process as well. The main purpose of these days was to fine tune the scenario and presentation, however last minute software changes were also performed when necessary.

The FFI deployment consisted of three laptop computers, essentially one for the integration software, one for the guard and one for the backend tracking system. Virtual machines were also used extensively during the demonstrations. Storing the state of these machines before and resetting after a demonstration proved both less time consuming and error prone than resetting the systems manually. This was especially valuable taken the short time between demonstrations into consideration. VMware Workstation[17] was used as virtualisation software.

During the demonstration all computers shared a Local Area Network (LAN) and could be addressed by all others. In order to achieve separation between the civilian side and the military side, two virtual security domains were established.

7 Evaluation

Evaluating the results from a demonstration as the one described in this document is difficult since the main goal is to show the attending audience the capabilities of the system and the benefits it has over already existing systems. Due to this fact, this section is divided into feedback from the audience and an assessment of the developed system and how well it conforms to the technological objectives defined. A section outlining future work identified during the development and demonstration phases is also included.

7.1 Technological Assessment

The experience gained from this activity provides an excellent foundation for future activities, including more exhaustive experimentation. Since demonstration of the technology and capabilities was the main purpose of this activity, an assessment can only be given on basis of the experience of designing, implementing and testing the software developed. This section is split into the same two main objectives as identified and described in Section 3.

7.1.1 System Integration

The first main objective defined was integration of the FFI system and the Oasis systems. This process involved both translating between data formats and providing an infrastructure for sharing the information. The translation between data formats was outlined in Section 5.4, and as expected when combining two different formats they do not align completely. In the case of NFFI and the TSO formats the latter is more expressive and the first can thus not handle all of these elements in this. However, we found that a one-to-one mapping could be provided between NFFI and a subset of the TSO format. Obviously this does not provide full interoperability between the models. Full interoperability was never a goal as the formats are not developed for the same purpose. These formats and the mapping between them can however be used to provide valuable information exchange between systems implementing these. The actual implementation of the translation process was done using plain Java programming. This does however produce highly specialised code with functionality limited to translating between these two formats. If the formats change the code has to change as well. Use of semantic technologies [5] would most

[17] http://www.vmware.com/ (2009)

likely have been a more future proof way of doing the translation. This would not only provide the mapping between the two data models, but also provide a foundation for creating a translation between them and other formats.

Even though both systems are based on Service Oriented Architecture and Web Services it was opted for an architecture were a middle tier was responsible for retrieving and delivering information to and from both systems. This additional middle tier used a file based interface to integrate with the Oasis system, and not Web Services. A better approach would have been to use SOA principles and Web Services to a larger extent when integrating the systems. Both systems are based on Web Services and as such it would have been a cleaner and more efficient integration to use this technology. However, the choice was taken as a consequence of how integration with both legacy and new software had been done in the Oasis system earlier. In addition, the fact that the Oasis system was based on the use of Web Services was not clear until the middle of the integration process and it was thus not time to revert this decision. A middle tier would have been necessary anyhow since the security mechanisms could not be implemented within the Oasis system. This being said the information exchange worked very well and the performance was satisfactory for this demonstration.

The choice was partly taken as a consequence of how integration with both legacy and new software had been done in the Oasis system earlier. The fact that it was not possible to implement the security mechanisms needed to pass the guard within the Oasis system was also a major factor for this.

All in all the integration of the systems was solvable and the mechanisms for exchanging information worked as expected. It must again be mentioned that this is based on the experiences made during the demonstration and pre-testing. The systems and the integration of these were not tested in more challenging environments.

7.1.2 Secure Release of Information

The secure release of information was the second main objective at the Oasis final event demonstration. The challenge of exchanging information between security domains is very complex and must be broken down into more manageable pieces. For this demonstration the more precise objectives were outlined in Section 3.2, together with an overall description of the challenge. Again it must be mentioned that demonstration of the capabilities of the technology was the main purpose of this activity and not exhaustive experimentation. This evaluation is thus influenced by this.

The first and second objectives stated are interlocked with each other. The first objective was to investigate mechanisms for secure release of information between security domains and the second was to investigate the use of object level security and labelling. In order to achieve secure and automatic release of information we used a combination of labels attached to information objects and a guard for enforcing the actual release control. In this case the label attached to the information is a confidentiality label which contains meta-data stating the sensitivity of the information. This can be used by the guard to filter and remove information that is not allowed to be passed to other domains. The solution has been proved to be working at previous experiments and demonstrations such as CWID and the use of an automatic guard for exchanging information between security domains are seen as a viable solution in the medium timeframe. The major challenge with this solution is to provide a high enough assurance level for the solution. This should not be a problem for the guard itself, as it is a stand-alone piece of software that has very limited functionality. However the process of creating and binding labels to the data object is more challenging and not solved by this effort. In our solution we used digital signatures and more specifically XMLDSig [12] which itself has some issues that must be resolved in order to get the assurance levels needed. There is also an unanswered question of which levels this solution can be used and what assurance levels are needed.

Investigating the use of standards for defining and complying with policy was the third objective stated. We used the eXtensible Access Control Markup Language (XACML) for both defining and for execution of policy. The XACML language was found to be very expressive and could without any problems be used to specify all policies used in this demonstration. The policy defined for our demonstration scenario was quite simple. How a more advanced XACML policy will behave is difficult to tell. The fear is that it might become too complex and will be difficult to verify. Often flexibility comes with a cost and this might be the case for XACML. Another desirable feature of XACML is the architecture defined which provides a very flexible infrastructure for easily defining new policies and distributing them. In addition, the fact that policy decision is removed from the enforcement means that for instance the guard doesn’t have to be implemented with a certain policy in mind. Again this provides more flexibility. The last advantage of XACML explored in this demonstration was the integration with SAML. A profile is provided for use of SAML attributes in the XACML decision process [9]. This enables the use of identity information provided by SAML to be used in the decision making. This demonstration has just scratched the surface of XACML and its capabilities. More experimentation is needed to make a firm recommendation on the usage of XACML. That said XACML seems to be a promising standard for defining and complying with policy [8].

The final objective defined was to provide initial investigation of identity management and role based access control. We used the Security Assertion Markup Language (SAML) for this purpose. SAML was introduced to the FFI demonstrator for this demonstration and as such the experience we have from using this technology is limited. One of the advantages of using SAML is that user identities do not have to be stored with all service providers. This simplifies the management of user credential substantially and is also a benefit for users as they are provided with single sign on capabilities. However in order to achieve this, a trust relationship has to be provided between the SAML Identity Provider and all relaying parties. In our solution trust were propagated by the use of digital signatures and certificates. Another advantage of SAML is the fact that any user attribute that can be expressed in XML can be included. Our solution made use of this to include the role of the user. When the user authenticates to the identity provider a role attribute was included in the assertion returned. This role was later used to determine access rights to services. In our solution roles was statically stored and linked to an identity within the identity provider. This solution is not very flexible and does not facilitate a dynamic environment where roles might change more frequently. In order to achieve support for this a more elaborate role management system must be used in combination with identity providers. From this demonstration it is too early to make a firm conclusion on the applicability of both SAML and the role based access control. At the time of writing these looks promising for use in a military system but more experimentation is needed.

In total, the security solutions worked as expected and the added value was shown during the demonstration. However more work is needed to ensure that these mechanisms can work properly in a military system. This would include testing in environments close to those found in operational settings.

7.2 Audience Feedback

The main purpose of a demonstration activity like the one described in this document is to provide the audience with a view of the capabilities and value added of the technology. The feedback from the audience is thus very important when evaluating the activity. One way of gathering feedback from audience is to use questionnaires. Due to the nature of the demonstration setting this was not possible in this case. The evaluation presented here is based on the feedback received through discussions and questions from the audience present. The audience consisted of a mix of people from civilian emergency response agencies, academia, military personnel and both civilian and military industry.

Overall the audience seemed to agree with our main purpose of the demonstration, which was to exchange information between military and civilian emergency response systems. The value of this information exchange was pointed out by several of the participants, and the real world requirements for this was emphasised. The need to share information with other civilian systems was also pointed out and logistics was pointed out as an example. This positive feedback shows that the scenario used during the demonstration was in some sense close to the needs of the real world and solves some of the requirements.

The FFI contribution was well received by the audience as well. The need for release control of information between security domains was well understood and it was pointed out that this was a concern for non-military systems as well. During the demonstrations several of the attending audience noted that the same requirement is valid when exchanging information between different emergency agencies. The concern was raised from both members of the police and the health care/ambulance community that they wanted to share information with each other, but not all the information. Some of the information within these systems are not of any use for the other party and only contribute to information overload, and some information is considered confidential and should thus not be shared at all. Examples of such information include confidential patient records and also confidential police records. It was noted that our solution of using object level security with labelling and automatic guard for redaction of information could solve this problem. The use of role-based access control also sparked some interest from the audience as the level of access within the civilian emergency systems might be different dependent on the role of the user. Example of such a role is an emergency room physician, who might have extended access rights when occupying this role but otherwise not.

In addition, the FFI contribution was looked upon with interest from a general information security perspective. We found that little work had been done on actually implementing security in the Oasis prototype system. The proposed standards and frameworks do mention security and takes into account the difference in requirements for different agencies when it comes to e.g. confidentiality. They also elaborate some on the issue of trust propagation. However, not much of this was implemented in the prototype. We showed the audience how some security mechanisms might be implemented, and this was well received.

7.3 Further Work

Even though the demonstration was a success and the mechanisms implemented worked as expected, some potential for improvements and need for further experimentation were identified. As mentioned several times in this document, this demonstration was only a part of a series of experiments planned. Some of the functionality has been tested before and this iteration represents a further refinement of these. And others like identity management and role based access control were introduced for this demonstration and need even further refinement. Below the main points of further work is outlined. These are not exhaustive. Cross domain change of information is very complex and these challenges are the ones that we will focus on in the time to come.

Development and testing of distributed identity management. In this demonstration we only used one single identity provider for both domains involved. This is not very likely in a real life setting where each domain will manage its own users. As a result of this, identity management will be distributed over several domains. The key question to answer is how trust should be propagated between identity providers of different security domains. The end goal should be that an access control entity should be able to make its access decision based on the attributes provided by an identity provider residing in a different domain. Distributed key and/or certificate management across domains is a related issue.

This demonstration also featured the use of XACML as a standard for defining and complying with policy. This work has just started and more is needed to determine its feasibility when used in a military environment.

In this demonstration software we have used XML Digital Signature in order to provide a binding between confidentiality labels and data. Through this and previous experiments and demonstration we have found that this specification is very flexible and can thus be used in many different ways. In our view there is a need to write and XMLDSig profile to ensure interoperability between different implementations and configurations. The initial work on a NATO profile for XMLDSig has started within the “NATO RTO RTG-031/IST-068 Task Group on XML in Cross-Domain Solutions” and will be taken further there.

The same group is also developing the confidentiality label and binding mechanisms further. At the time of writing two documents are being prepared; one for the actual label definition and one for the binding mechanism. Norway has the editorial responsibility of these documents. Once these are completed the implementation of these are to be included in the demonstrator software.

The security solutions described in this document has only been tested with the request/response interaction pattern. It is believed that other patterns, like the publish/subscribe pattern, is beneficial when used in a military setting. It is thus important to find out how the security mechanisms will handle such a change. Important questions to answer are whether there are requirements from the security mechanisms that will limit the utility of the exchange mechanism and vice versa. And are there possible adaptations that can be made to provide the best of both. In addition, military networks are often characterised by low bandwidth, high delay and frequent errors. It is also very important to find out whether the mechanism described here can be used in such a setting. The mechanisms are based on the use of XML which tend to be very verbose, which might not be suitable. Identifying where and when it is feasible to use these mechanisms are important.

8 Summary

The demonstration of information exchange between military and civilian systems performed at the Oasis Final Event was by all measures a success. It showed both how two different data formats, the NFFI and TSO, could be used to exchange information and a viable solution for securely exchanging information between security domains. The value of such an information exchange was also highlighted. Exchanging information between civilian and military systems is of many seen upon as one of the greatest challenges of cross-domain information exchange. The main differences between this scenario and one where two military domains are exchanging information is administrative and policy. From a technical perspective these are no different, and the same mechanism can be used to minimise the risk of leaking information.

The approach taken by the Oasis consortium for sharing of information and coordination between different emergency response agencies are very similar to the one envisaged for the NNEC. The use of Service Oriented Architecture to enable easier integration and discovery of services is one of the common denominators. In both cases fast and timely information exchange is needed, also between security domains. The solution presented in this document enables the automatic sharing of information between such domains, while at the same time minimising the risk of information leakage. This is achieved through the combination of confidentiality labels that are cryptographically bound to the information, and a guard that performs access control based on them.

Web Services is currently the most common technology used for implementing SOA. This technology is heavily based on the use of XML. Since our confidentiality label and binding mechanism are both specified using XML they can be integrated with Web Services without any problems. The security mechanisms used are either specified with Web Services in mind or can be integrated. This makes it a good fit for the implementation of future services to be deployed in the NEC environment. Even more, the confidentiality label specified can be used in combination with other data formats as well and can thus be used as a generic container for confidentiality meta-data. The need for expensive and proprietary solutions that is tightly coupled with either data formats or protocols are to some extent removed. Based on several years of testing and experimenting with the label and guard solution we feel that our solution has a great potential to solve the challenge of seamless information exchange within NEC.

During this demonstration we have showed important security functionality that should be implemented in the NEC environment in order to achieve the level of flexibility and efficiency envisage. In addition to the confidentiality label and guard solution we have been experimenting with for some time, we have introduced identity management, role based access control and new functionality for specifying and complying too security policy. From this demonstration it is not possible to make a final conclusion on the usability of these in a military environment. More work is required. The demonstration has on the other hand showed the potential benefits when using these mechanisms and we firmly believe that they are important to realising NEC.

As a final note it should again be emphasised that all software described in this document and used during the demonstration is purely experimental and no effort has been taken to have it formally certified.

References

[1] W. Ford, P. Hallam-Baker, B. Fox, B. Dillaway, B. LaMacchia, J. Epstein, and J. Lapp, "XML Key Management Specification (XKMS),"http://www.w3.org/TR/2001/NOTE-xkms-20010330/, 2003.

[2] R. Haakseth and M. Andreassen (Thales Norway), "NATO CWID 2007 Cross Domain Web Services Experiments," FFI-notat 2007/02302 (U), 2007.

[3] R. Haakseth, T. Gagnes, D. Hadzic, T. Hafsøe, F. T. Johnsen, K. Lund, and B. K. Reitan, "Experiment report: "SOA – Cross Domain and Disadvantaged Grids" – NATO CWID 2007," FFI-rapport 2007/02301 (U), 2007.

[4] D. Hadzic, K. Rose, K. Lund, and T. Gagnes, "An Extensible and Modular Software Test-Bed for Distributed Communication and Picture Compilation - Technical Documentation," FFI notat 2007/01968 (U), 2007.

[5] B. J. Hansen, T. Gagnes, R. Rasmussen, M. Rustad, and G. Sletten, "Semantic Technologies, Norwegian Defence Research Establishment (FFI) Report," 2007.

[6] F. T. Johnsen, T. Hafsøe, E. Skjervold, K. Rose, K. Lund, and N. A. Nordbotten, "MULTINETT II: SOA and XML Security Experiments with Cooperative ESM Operations (CESMO)," FFI rapport 2008/02344, 2009.

[7] R. Malewicz, "NATO Friendly Force Information (NFFI) (version 1.2) Interface Protocol Definition IP3, NC3A Working Document," 2006.

[8] N. A. Nordbotten, "XML and Web Services Security," FFI rapport 2008/00413, 2008.

[9] OASIS, "SAML 2.0 profile of XCAML v2.0," 2005.

[10] R. Rasmussen, A. Eggen, D. Hadzic, O.-E. Hedenstad, R. Haakseth, and K. Lund, "Experiment report: "Secure SOA supporting NEC" - NATO CWID 2006," FFI rapport 2006/00325 (U), 2006.

[11] M. Raymond, E. Webb, and P. I. Aymond, "Emergency Data Exchange Language (EDXL) Distribution Element, v. 1.0,"http://docs.oasis-open.org/emergency/EDXL-DE/V1.0, 2009.

[12] World Wide Web Consortium, "XML-Signature Syntax and Processing," 2002.

Appendix A Acronyms and Abbreviations

API	Application Programmer Interface
COP	Common Operational Picture
FFI	Norwegian Defence Research Establishment
IdP	Identity Provider
NBD	Network Based Defence
NFFI	NATO Friendly Force Identifier
MIP	Multilateral Interoperability Programme
NEC	Network Enabled Capability
NNEC	NATO NEC
Oasis	Open Advanced System for dISaster & emergency management
OASIS	Organisation for the Advancement of Structured Information Standards
PDP	Policy Decision Point
PEP	Policy Enforcement Point
SAML	Security Assertion Markup Language
SOA	Service Oriented Architecture
TSO	Tactical Situation Object
XACML	eXtensible Access Control Markup Language
XKMS	XML Key Management Specification
XML	eXtensible Markup Language
XMLDSig	XML Digital Signature

Appendix B NFFI – TSO Mapping Rules

SEND

The following table defines the TSO elements the FFI LIM shall be able to send to OASIS, i.e. translation from NFFI to TSO.

TSO Element	Value	Comments
Context. id	TSO.Context.Origin.**orig_id** + TSO.Context.**creation**	E.g. "FFI" + timestamp.
Context. mode	"ACTUAL"	Alternative: EXERCS
Context. msgtype	"UPDATE"	
Context. creation	<current time>	When the TSO was created by the FFI LIM
Context. level	"TACTCL"	
Content. urgency	"URGENT"	
Context. confidentiality	"UNCLAS"	
Context. origin. orig_id	NFFI. positionalData. trackSource. SourceSystem. **system**	
Context. origin. name	N/A	
Event.*		If the track "331-A1" (crashed F16) is included in the message, an event with id "FFI_01" is built based on this resource. The location is identical to the F16 resource location. This mapping is only valid for the Oasis NATO demo.
Resource. Type. class	NFFI. indentificationData. **UnitSymbol**	This will determine the GUI icon.
Resource. ID	NFFI.indentificationData. **ShortName**	The NFFI.positionalData. trackSource. **transponderID** could have been used too since transponderID should be unique in the NFFI domain.
Resource. ORG_ID	NFFI.positionalData. trackSource. **SourceSystem**	Note: In all TSO messages from FFI: Context.origin.id = Resource.orig_id
Resource. NAME	N/A	
Resource.	"CUR"	Current position. (The only option for "tracks".

geo. TYPE		
Resource. geo. POSITION. TYPE	“POINT”	
Resource. geo. POSITION. COORD.LAT	NFFI. positionalData. coordinates. **latitude**	Both use WGS 84
Resource. geo. POSITION. COORD.LON	NFFI. positionalData. coordinates. **longitude**	Both use WGS 84
Resource. geo. POSITION. COORD.HIGHT	NFFI. positionalData. coordinates. **altitude**	Both use meters above see level.
Resource. NATIONALITY	“NO”	
Mission.*	-	Not supported

RECEIVE

The following table defines the TSO elements the FFI LIM shall be able to receive from OASIS. Each element shall be mapped to a NFFI element, as defined in the table below. If a mapping is not defined (N/A), or the TSO Element is not listed, the TSO element shall be ignored.

TSO Element	NFFI Element	Comments
Context. creation	NFFI.positionalData. **dateTime**	Resource.datetime is usually not set, so context.creation is the best option.
Event.*	N/A	
Resource. Type. class	NFFI. indentificationData. **UnitSymbol**	We need to define the TSO types we will use during the demo (RESOURCE/TYPE) and a mapping to the “military types” supported by the FFI Demonstrator.
Resource. ID	NFFI.positionalData. trackSource. **transponderID** and NFFI.indentificationData. **ShortName**	
Resource. ORG_ID	NFFI.positionalData. trackSource. **SourceSystem**	
Resource. NAME	NFFI.indentificationData. **ShortName**	E.g. this vale overrides resource.id if set.
Resource. geo. TYPE	N/A	

Resource. geo. POSITION. TYPE	N/A	
Resource. geo. POSITION. COORD.LAT	NFFI. positionalData. coordinates. **latitude**	Both use WGS 84
Resource. geo. POSITION. COORD.LON	NFFI. positionalData. coordinates. **longitude**	Both use WGS 84
Resource. geo. POSITION. COORD.HIGHT	NFFI. positionalData. coordinates. **altitude**	Both use meters above see level. Skip?
Resource. NATIONALITY	N/A	
Mission.*	N/A	Not supported