


INNO-UTILITIES IPS-2001-42014	Deliverable 11 – System design and specification	
	Location:	N/A
	Meeting date:	N/A
	Responsible:	Dimitrios Tsigos, VIRTUAL TRIP, tsigos@vtrip.net
	Confidentiality:	INNO-UTILITIES Consortium confidential
	Distribution:	Project internal and Commission Project Officer
	Document Ref.:	WP4/D11 - PrototypeSpecification

INNO-UTILITIES

Deliverable11

System design and specification document

Editor: Dimitrios Tsigos, VIRTUAL TRIP

Suggested readers

INNO-UTILITIES consortium partners

European Commission Project Officer

Abstract

This document provides the systems' technical architecture specification of the INNO-UTILITIES demonstrator.

The INNO-UTILITIES demonstrator is a network-centric information system, which will be used for fighting telecommunication fraud by Telecom Operators and public authorities. Emphasis has been given on the security aspects of the demonstrator, as well as to its ability to function in a distributed environment, allowing for separation of administrative tasks among the participating organizations.

Disclaimer

This document contains material, which is copyright of certain INNO-UTILITIES consortium parties and may not be reproduced or copied without permission. The information contained in this document is the proprietary confidential information of certain INNO-UTILITIES consortium parties and may not be disclosed except in accordance with the consortium agreement.

The commercial use of any information in this document may require a license from the proprietor of that information.

Neither the INNO-UTILITIES consortium as a whole, nor a certain party of the INNO-UTILITIES consortium warrant that the information contained in this document is capable of use, or that use of the information is free from risk, and accept no liability for loss or damage suffered by any person using the information.

List of Authors

Dimitrios Tsigos, VIRTUAL TRIP

Vangelis Mihalopoulos, VIRTUAL TRIP

Thanassis Parathyras, VIRTUAL TRIP

Stelios Gasparis, VIRTUAL TRIP

Nikos Ntarmos, VIRTUAL TRIP

Alexandros Antonopoulos, University of Patras

Dimitrios Serpanos, University of Patras

Artemios Voyiatzis, University of Patras

Nikolaos Xipolias, University of Patras

Anastasius Gavras, Eurescom GmbH

Table of Contents

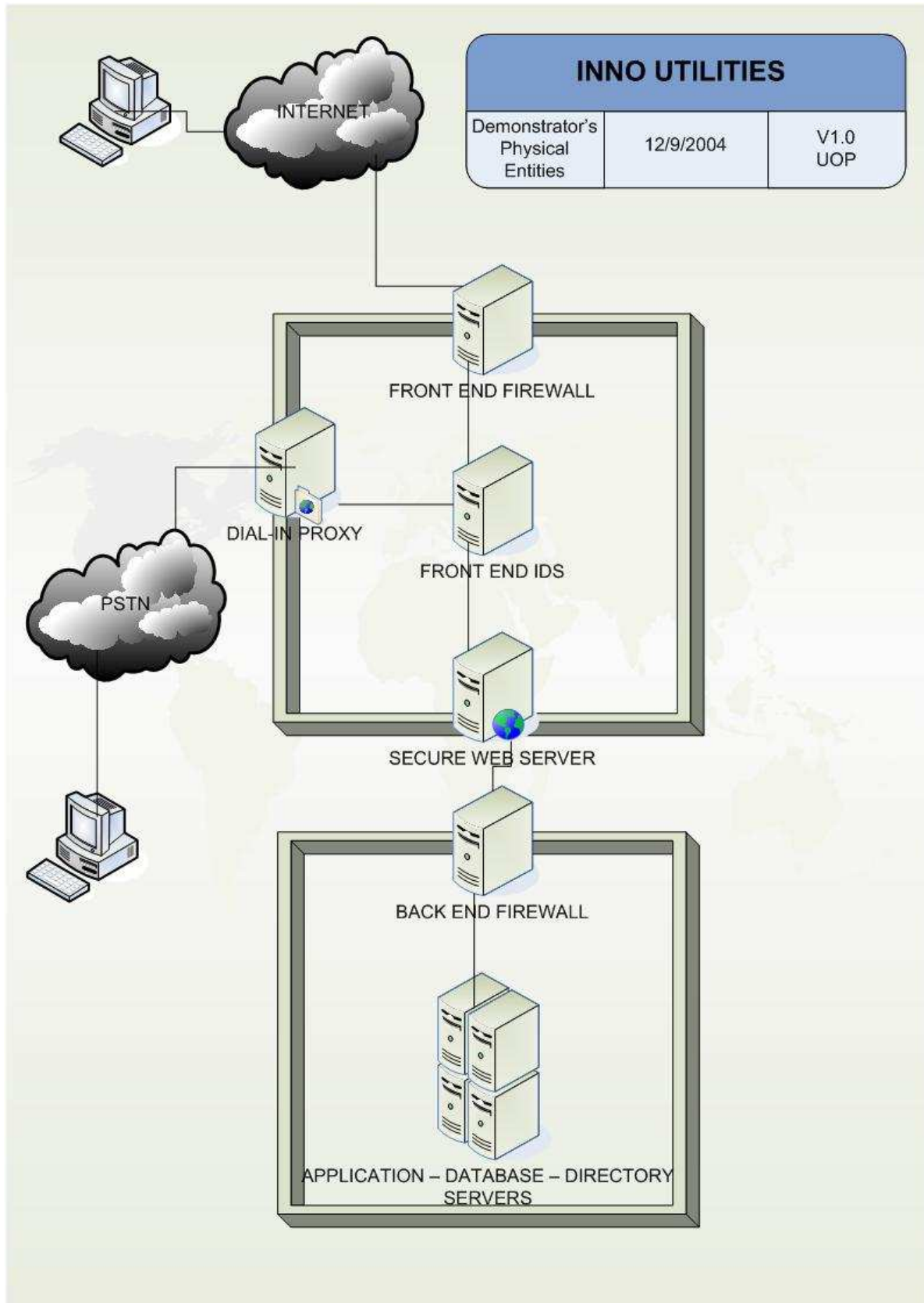
Executive Summary.....	Error! Bookmark not defined.
List of Authors.....	3
Table of Contents	4
Abbreviations	Error! Bookmark not defined.
Definitions	Error! Bookmark not defined.
1 Introduction.....	5
2 System’s technical architecture.....	6
2.1 Information tier specification	7
2.2 Network tier specification	12
2.2.1 Dial-In server.....	12
2.2.2 Web server.....	12
2.2.3 Secure web server	12
2.2.4 Intrusion detection system	13
2.2.5 Firewall [EC].....	14
2.3 Application tier specification	15
2.3.1 Policy Guard Module.....	16
2.3.2 Security Guard Module	16
2.3.3 Accounting module	17
2.3.4 Business process module	18
2.4 Presentation tier specification	20
2.4.1 User-Interface specification	20
2.4.2 Message formats.....	20
2.4.3 (Optional) Message integrity	20
2.4.4 (Optional) Message confidentiality.....	20
2.4.5 (Optional) SMS API specification.....	21
2.4.6 (Optional) SMS “dialogues” description	21
2.4.7 (Optional) VoiceXML API specification	21
2.4.8 (Optional) Telephone dialogues description	21
References	22
ANNEX	Error! Bookmark not defined.

1 Introduction

This document presents the system technical architecture of the Network-Centric Secure Information System for fraud-fight in the field of telecommunication and describes the design decisions behind it. This document will be the guideline for the development phase of the demonstrator.

According to the requirements identified in the assessment report, the system will be enhanced with strong security mechanisms. Although the system architecture follows an N-Tier design, the security architecture arises from a layered approach design, proposed at International Standard Draft: "ISO/IEC FCD 18028 IT Security techniques – IT Network Security – Part 2: Network Security architecture". The reason for the different treatment at the security and system architecture design is that it leads to a better result that takes into account both points of view. The combination of the two methods does not produce any inconsistency at the integration of the security and the system architecture, as the two methods are interrelated and not oppositional. The security considerations focus on the three layers (Infrastructure Security Layer, Service Security Layer, and the User Security Layer), and are then mapped to the additional Tiers of the system architecture. Moreover there is no separate security architecture design as the system and security architectures are combined in an organic rather than in a distinct way.

2 System's technical architecture



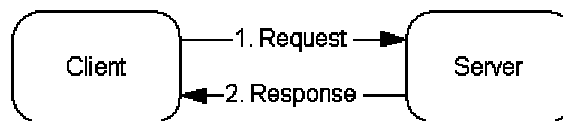
2.1 Information tier specification

- LDAP software and version

Based on the transactions that are going to be performed by the system, a directory-structured storage scheme has been selected. The information storage and security requirements are satisfied by the OpenLDAP software, release 2.2.17. This release is considered the latest stable release from the OpenLDAP Project and it will cover our needs for flexibility, efficiency, and security.

- OpenLDAP modules selection & configuration

The Directory Server configuration minimizes possible security threats and boosts the efficiency of the directory server. Fundamental configuration options include applying the Local Directory Service only for the local domain. There is not any further interaction with other Directory Servers for security reasons. A typical interaction in such a system involves a local client request and the corresponding local server's response as shown in the figure below.



Based on the system architecture, this operation model will satisfy all information storage and retrieval needs.

For security reasons, the Application Tier will be solely granted read-only access to the LDAP repositories. The LDAP data section will be kept on read-only media (e.g. on a CD-ROM) and updated off-line; the system administrator will have to take the system off-line, perform any maintenance operations (e.g. insertions/deletions to the LDAP repositories, change of access levels, etc.), and then bring the system back on-line. This procedure may be automated, but will in any case be available only to the system administrator or to local users of the server(s) hosting the LDAP data repositories.

- LDAP infrastructure dependencies & considerations

OpenLDAP is distributed as "Free Software", for any usage. It is distributed only as source-code, so there are not any officially available binary packages. OpenLDAP can be used with a variety of Operating Systems. We will be basing the OpenLDAP installation on a GNU/Linux distribution, Redhat GNU/Linux Enterprise. This distribution has proven stable and secure and its long life-cycle and continued support are valuable advantages.

The software updates for the directory server cannot be automated, because of the source-only distribution of the OpenLDAP Project software.

- Directory structure

- o Authentication tree

The Authentication tree contains credentials and other information for the users of the system. Authentication can be based on digital certificates, plain credentials or network based authorization, all of which can be stored in the Authentication tree.

- o Information tree

In the structure of the LDAP tree, the information nodes are modelled after the XML/DTD document that is listed in the revised version of the Assessment Report. We define the attribute names and their types and then we construct the IncidentReport objectClass, which will represent single Incident Reports in the Information Tree. Below is the Schema file used for this modelling:

```
#attributes for the IncidentHeader object

#basic attribute from which all others inherit

attributeType ( 1.3.6.1.4.1.22174.2.1.1 NAME 'SequenceNumber'
```

```

DESC 'Incident Sequence Number'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768}
SINGLE-VALUE )

attributeType ( 1.3.6.1.4.1.22174.2.1.2 NAME 'SubjectLine'
DESC 'Incident Subject Line'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.3 NAME 'Keyword'
DESC 'Keyword for the Incident'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.4 NAME 'CompanyProvidedInfo'
DESC 'Company that provided this info'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.5 NAME 'ContactPerson'
DESC 'Person that submitted this info'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.6 NAME 'ContactTelephone'
DESC 'Phone number for contact'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.7 NAME 'ContactFax'
DESC 'Fax number for contact'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.8 NAME 'ContactEmail'
DESC 'Email address for contact'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.9 NAME 'DateOfIncidentReport'
DESC 'Date this info was submitted'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.10 NAME
'DateOrPeriodOfIncident'
DESC 'Date of Period the Incident occurred'
SUP SequenceNumber )

```

```
#attributes for the IncidentReported object

attributeType ( 1.3.6.1.4.1.22174.2.1.11 NAME 'TypeOfFraud'
              DESC 'Type of fraud'
              SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.12 NAME
'TypeOfServiceAffected'
              DESC 'Type of Service Affected by the Incident'
              SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.13 NAME 'DescriptionOfFraud'
              DESC 'Description of the Fraud'
              SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.14 NAME 'DetectionMethod'
              DESC 'Method of detection for the Incident'
              SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.15 NAME 'Countermeasures'
              DESC 'Countermeasure applicable for the Incident'
              SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.16 NAME 'TypeOfEquipmentUsed'
              DESC 'Type of the Equipment used to make fraud'
              SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.17 NAME
'TypeOfEquipmentAffected'
              DESC 'Type of Equipment affected by this fraud'
              SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.18 NAME 'FraudstersName'
              DESC 'The Name of the Fraudster responsible'
              SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.19 NAME
'FraudstersOrganisationOtherIdData'
              DESC 'The Organization, or other identification data, of
the Fraudster'
```

```

SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.20 NAME
'CountriesCompaniesAffected'
DESC 'List of countries and Companies affected by this
Incident'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.21 NAME 'HotDestinations'
DESC 'Telephone numbers called by the Fraudsters'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.22 NAME 'LegalIssues'
DESC 'Legal Issues concerning the fraud'
SUP SequenceNumber )

attributeType ( 1.3.6.1.4.1.22174.2.1.23 NAME 'AdditionalIssues'
DESC 'Any additional information'
SUP SequenceNumber )

objectclass ( 1.3.6.1.4.1.22174.2.2.1 NAME 'IncidentReportHeader'
DESC 'A Fraud Incident Report'
SUP top STRUCTURAL
MUST ( SequenceNumber $ SubjectLine $ Keyword $
CompanyProvidedInfo $ ContactPerson $ ContactTelephone $ ContactFax
$ ContactEmail $ DateOfIncidentReport $ DateOrPeriodOfIncident ) )

objectclass ( 1.3.6.1.4.1.22174.2.2.2 NAME 'IncidentReported'
DESC 'A Fraud Incident Report'
SUP top STRUCTURAL
MUST ( TypeOfFraud $ TypeOfServiceAffected $
DescriptionOfFraud $ DetectionMethod $ Countermeasures $
TypeOfEquipmentUsed $ TypeOfEquipmentAffected $ FraudstersName $
FraudstersOrganisationOtherIdData $ CountriesCompaniesAffected $
HotDestinations $ LegalIssues $ AdditionalIssues ) )

objectclass ( 1.3.6.1.4.1.22174.2.2.3 NAME 'IncidentReport'
DESC 'A Fraud Incident Report'
SUP ( IncidentReportHeader $ IncidentReported ) STRUCTURAL
)

```

- o Policy tree

The Policy tree defines the information needed to control access to the information. The LDAP server (OpenLDAP) has capabilities for Access Control, but these will not be used in the Demonstrator context. The Policy information will be retrieved and used by the Application tier.

- Database server version

The database server of the system is a Relational Database System; the software used is the MySQL Database Server, release 4.1.7. The software is considered to be efficient, in heavy-load operation, and easy to customize, following all the necessary industry standards.

- Database server modules selection & configuration

The Database [MySQL] will be using the InnoDB tables. InnoDB provides MySQL with a transaction-safe (ACID compliant) storage engine with commit, rollback, and crash recovery capabilities while various built-in features increase multi-user concurrency and performance. Its CPU efficiency is probably not matched by any other disk-based relational database engine. InnoDB is used in production at numerous large database sites requiring high performance.

The Database schema that will be used will hold accounting information for the Demonstrator and a minimal part of the system logging. This information will be later used by the Application to produce accounting reports and to show basic statistics for the usage of the System.

The Database server will be configured to maximum security by:

1. Restricting network access, allow only the Application to access the service from the network.
2. Optionally, securing each connection by username and password, as long as the application allows it.
3. Optionally, use SSL to secure each connection, as long as the application allows it.

- Database server Infrastructure dependencies & considerations

MySQL software, including the InnoDB storage engine, is Free Software and is distributed under the FSF GPL [Free Software Foundation, General Public License] v2.

- Accounting and auditing information description

The accounting and auditing module will be using the functionality exposed by the database module to store and maintain information pertaining to the usage of the system by its users. The data stored in the database should allow for different interpretations and/or pricing schemes. In order to achieve these goals, the accounting and auditing module will maintain separate information for when users authenticate with the system and are granted access privileges, and for when users pose queries to the system. As of these, the data stored will include:

1. Authentication/privilege-granting information comprised of the exact time and type of the transaction, user identification data, and originating host information.
2. System usage information, comprised of the exact time each query was posed, user identification and originating host information, as well as the exact query sent to the application.

With the information defined above, accounting can be based on any combination of connection time, query complexity, result size, etc. Persistence of the auditing information is guaranteed by the usage of the MySQL/InnoDB engine as the storage backend.

2.2 Network tier specification

2.2.1 Dial-In server

A dial-in server is a host equipped with a modem and phone line, that allows other hosts with modems and phone lines to call and connect to it. The main reason for equipping the demonstrator with a dial-in access via modem is that certain potential customers have expressed the requirement to dial-in to the secure server via a dedicated modem and phone line. However the dial-in option will not be subject for Intrusion testing, thus a standard configuration as provided by the operating system will be used.

- Dial-In server version – No special dial-in server will be used. The default configuration by the used operating system will be configured to deliver this functionality.
- Dial-In server modules & configuration – Default modules and configuration only
- Dial-In server infrastructure dependencies & considerations – No dependencies. A call-back mechanism will be used to pre-registered modems (phone numbers). These phone numbers have to be made known during server set-up.

2.2.2 Web server

- Web server version

The web server of the system is the HTTP Server of the ASF (Apache Software Foundation). The server is considered to be very stable and is used by more than 60% of the websites on-line, making it the most widely used web server. The latest release 2.0.52 (currently) is going to be deployed, solving most of the known security and stability issues. Any critical security update will be applied in time.

- Web server modules & configuration

The HTTP server will include the following modules:

- mod_log_config, for logging of the requests made to the server

- Web server infrastructure dependencies & considerations

The HTTP Server is distributed under the Apache Software License v2.0, which is almost compatible with the FSF GPL.

2.2.3 Secure web server

- Secure web server version

As above in section 2.2.2, the web server of the system is the HTTP Server of the ASF (Apache Software Foundation). The latest release 2.0.52 (currently) is going to be deployed, solving most of the known security and stability issues. Any critical security update will be applied in time.

- Secure web server modules & configuration

The HTTP server will include the following modules

- mod_log_config, for logging of the requests made to the server
- mod_log_forensic, for forensic logging of the requests made to the server
- mod_ssl, provides strong cryptography using the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols

- Secure web server infrastructure dependencies & considerations [VT]

The HTTP Server is distributed under the Apache Software License v2.0, which is almost compatible with the FSF GPL. The mod_ssl module is distributed under a BSD-style license, which is compatible with the software's usage and goals.

2.2.4 Intrusion detection system

The front-end intrusion Detection System is a Network Intrusion Detection System (NIDS). The NIDS will not be just a passive NIDS. The purpose of a passive NIDS is to monitor all network traffic and to generate alerts in order for the Administrator to take any needed actions. Having in mind that quite often the weakest link in any system is the human factor (neglect of duties, social engineering attacks, etc) it is necessary to automate procedures as much as possible. Thus the need arises for the NIDS to be at least able to configure the front-end firewall and to block dynamically any suspicious host.

The NIDS used is Snort [2] software distributed under GPL (<http://www.snort.org/license.html>).

2.2.4.1 Intrusion detection system modules & configuration

A Network Intrusion Detection System is able only to monitor network traffic and detect any suspicious or malicious packets. It is obvious that beyond detection the NIDS needs to be able to log all necessary information, to alert the administrator (or any other responsible role) and optionally to update the firewall dynamically in order to prevent a suspicious host from accessing the network. These requirements correspond to a database like MySQL [3] for logging, ACID [4] (Analysis Console for Intrusion Detection) and Guardian [5] for controlling a firewall.

- MySQL: MySQL provides a safe enough environment for storing all necessary information. The Database Server for the Intrusion Detection System will be installed in the same system with the IDS. The database scheme used is included in Snort.
- ACID/ Barnyard: Analysis Control for Intrusion Detection is a PHP based application which is able to read Snort's data (stored in the DB) and to generate human readable graphs and statistics.
- Guardian: Guardian is a security program which works in conjunction with [Snort](#) to automatically update firewall rules based on alerts generated by Snort. Guardian is able to control the following firewalls:
 - ipchains
 - iptables
 - ipfwadm
 - ipfw (on FreeBSD)
 - ipfilter
 - Checkpoint firewall
 - Pix firewall

All software mentioned above is released under the GPL license.

2.2.4.2 Intrusion detection system infrastructure dependencies & considerations

Snort can be used with a variety of Operating Systems. GNU/Linux is used widely for NIDS platforms. In the system the distribution that will be used is RedHat GNU/Linux Enterprise.

The NIDS maintenance is divided in two parts. The first is about Snort as software and any bug fixes or new features added to the software. The second one has to do with the rules used for detection.

Snort's update can be done automatically. The issues that may arise are rule incompatibilities or plug-in incompatibilities.

Snort has a large set of rules used to detect many attacks. Beyond that, the user is able to develop and maintain a custom rule set. Since the attack set is not static Snort depends a lot on the rule set update mechanism. An NIDS with an outdated rule set is a useless NIDS. As a result the need for rule maintenance is one of the most important factors for the effectiveness of the NIDS. Furthermore it is crucial for Performance and Efficiency reasons to activate/deactivate rule sets based on the systems used. Rules which do not correspond to existing systems must be disabled. For example, if a system does not include Windows servers, then the IDS does not have to include rules regarding attacks against IIS or Windows RPC.

2.2.5 Firewall

- Firewall version

The firewall that will be used in the demonstrator will be a commercial version of the Astaro Security Linux V5 Firewall, which was kindly provided for evaluation and demonstration purposes by Astaro AG.

http://www.astaro.com/firewall_network_security/firewall_asg

- Firewall modules & configuration

The product provides a large number of features such as Stateful Packet Inspection (standard firewall capability), Application-Level Deep Packet Filtering, Security Proxies (for HTTP DNS SOCKS POP3 Ident and SMTP), NAT and Masquerading, DoS Protection, Traffic Shaping and QoS, and Detailed Reporting.

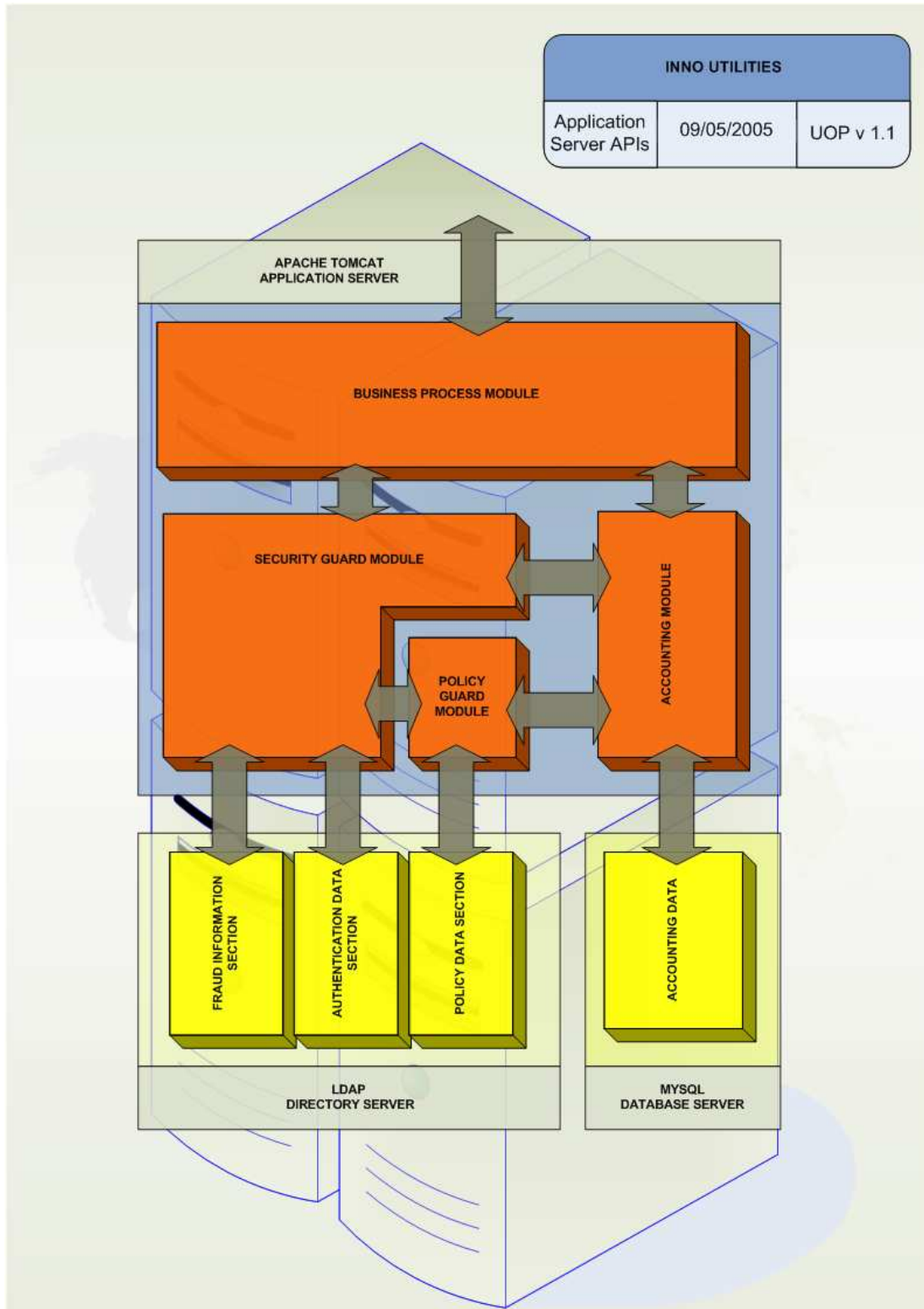
All non core modules that are necessary for the protection of the demonstrator will be removed from the installed configuration.

A standard configuration will be used.

- Firewall infrastructure dependencies & considerations – This product and its version have been chosen based on the excellent experience Eurescom GmbH made in protecting its own operational resources.

2.3 Application tier specification

The building blocks and modules of the application tier are outlined in the following figure and further defined in the rest of this section. Unless stated otherwise, the programming language of choice is the Java programming language, using the Java 2 Enterprise Edition SDK v1.4.



2.3.1 Policy Guard Module

The Policy Guard is a mechanism used by owners of resources to enforce policies. The main focus of the Policy Guard is to ensure that the permission policies are enforced.

The Policy Guard physically lies at the Application Server as a distinct module. The Policy Guard module will be developed using the same development tools and programming language with the rest modules of the Application Server.

2.3.1.1 General Description

The Policy Guard has the responsibility to query a Directory Server for the policy matching the credential-type of connection pair. The Policy Guard isn't active up to the moment that the Security Guard queries the Policy Guard providing the credentials-type of connection pair. Then in turn it queries the Directory server in order to retrieve the corresponding policy document. This document describes the client's permission regarding the information stored in the system. The Security Guard is responsible to interpret this document and to decide if the client has the necessary privileges to access the requested data.

2.3.1.2 Functionality

Based on the above the functionality of the Policy Guard is:

- The Security Guard module queries the Policy Guard module providing the user's credentials and her type of connection (API function)
- The Policy Guard module uses the credential-type of connection pair to retrieve the appropriate policy document from the Directory Server Module.
- The Policy Guard module returns the policy document to the Security Guard module (API function).
- The Policy Guard module informs the Accounting Module for the actions carried out.

2.3.1.3 APIs

The APIs of the Policy module that will be implemented are the following:

- Policy Guard module - Security Guard module API
- Policy Guard module - Accounting module API
- Policy Guard module - LDAP Policy Data Section API

Each API consists of specific global functions prototypes and data objects templates.

2.3.2 Security Guard Module

The Security Guard physically lies at the Application Server as a distinct module. The Security Guard module will be developed using the same development tools and programming language with the rest modules of the Application Server.

2.3.2.1 General Description

When a client requests some type of data from the Web Server, the Web Server forwards the request to the Security Guard (through the Business Process module) along with information regarding the client authentication mode, authentication information and her type of connection. After a successful authentication the Security Guard gives this set of information to the Policy Guard and waits for an answer that maps the specific combination of authentication mode-credentials-type of connection to a specific role in the system. If the mapped role has the necessary permissions to access the requested data then the Security Guard forwards the request to the Directory Server Component. Upon reception of the requested data after double-checking that they match the role's access rights it sends them back to the Web Server (which in turn sends them to the requesting client) and at the same time updates the auditing component with transaction information (client, data, time etc).

2.3.2.2 Functionality

According to the above, the Security Guard module should offer the following functionality:

- Retrieval of user's request and credentials from the Business Process module (API function)
- Request for user's credentials from the LDAP Authentication Data Section (API function)
- Validation of user's credentials (authentication) comparing the credentials retrieved by the Business Process module and the LDAP Authentication Data Section (Internal local function)
- Request for a user's credentials mapping to a specific predefined role from the Policy Guard module (API function)
- Retrieval of the result of the mapping of user's credential to a role from the Policy Guard module (API function)
- Request from the LDAP Fraud Information Section for the data access level properties of the user's requested data (API function)
- Decision extraction of which parts of the data requested will be provided to the user, comparing user's role access level and data's access level properties (Internal local function)
- Forwarding of the Security Guard's response that corresponds to the user's request, to the Business Process module (API function)
- Informing of the Accounting module for every action carried out.
- (Optionally) Functionality for external authentication mechanism support.

2.3.2.3 APIs

As described above a great percentage of the Security Guard's functionality consist of API functions for the interaction with other modules. The APIs that will be implemented are the following:

- Security Guard module - Business Process module API
- Security Guard module - Policy Guard module API
- Security Guard module - Accounting module API
- Security Guard module - LDAP Fraud Information Section API
- Security Guard module - LDAP Authentication Data Section API

Each API consists of specific global functions prototypes and data objects templates.

2.3.3 Accounting module

The module will be responsible for keeping accounting information. Data acquired from other modules, such as the identity of client and the information accessed, will be used by the module in conjunction with accounting policy, to produce accounting reports. The accounting data will be stored by the module to the accounting data storage, currently a database. Interfaced with the policy guard module, this software will be capable of retrieving information policy data and modify policies matching specific accounting patterns.

The module will implement an "Accounting Information API" to enable the retrieval and pre-processing of the account information stored in the system. This API will make it possible for the Business Process Module to access transparently all the accounting information with a uniform way of access.

2.3.3.1 Functionality

According to the above, the Accounting module should offer the following functionality:

- Maintenance (storage/retrieval) of detailed auditing information (API functions).
- Maintenance (storage/retrieval) of accounting patterns (API functions).
- Presentation of auditing/accounting information to the Business Process module (API function).

2.3.3.2 APIs

The APIs that will be implemented are the following:

- Accounting module - Security Guard module API
- Accounting module – Policy Guard module API
- Accounting module – Business Process module API
- Accounting module – MySQL Accounting Data section API.

Each API consists of specific global functions prototypes and data objects templates.

2.3.4 Business process module

This software will incorporate the main application logic. Using data from all the interfacing modules [“Security Guard Module”, “Accounting Module”, the Client request] it will control execution and data flow. Interaction with the users will be mainly through a Web Service (SOAP) interface.

2.3.4.1 Functionality

The main functionality of this module will consist of:

- Authenticating users of the system, through the functionality exposed by the Security Guard and Policy Guard modules.
- Processing of (read-only) queries over the fraud information.
- Maintenance of auditing/accounting data, through the functionality exposed by the Accounting module.
- Updating of the authentication, policy, and fraud data, stored in the LDAP backend, through the functionality exposed by the respective modules.
- (Optionally) Backing up the stored information onto magnetic or optical media (e.g. tape drives, DVDs, etc.) or hard-disk-resident files, in a secure and authenticated manner.

2.3.4.2 APIs

The Business Process module will provide external APIs, exposed by the Web Services of the system, as well as internal APIs for interfacing with the rest of the system’s modules.

More specifically, internal APIs will include:

- Business Process module – Security Guard module API
- Business Process module – Accounting module API

Each internal API consists of specific global functions prototypes and data objects templates.

Moreover, according to the above-mentioned functionality exposed by the Business Process module through its Web Service interface, the module will include the following APIs:

- Read-only query API.
- Update API.
- (Optionally) Back-up API.

2.3.4.3 Notes

A core design characteristic of the Inno-Utilities demonstrator is that it will support distributed operation, allowing for separation of administrative tasks among the participating organizations. This will be accomplished as follows. Each demonstrator instance must be aware of all other demonstrator instances and be able to query them. Then, given a user query posed at a given demonstrator instance, the Business Process module will retrieve all information available locally to the specific instance. It will then reuse the user authentication data to execute the exact same query on all other demonstrator instances. The returned result sets will then be merged into a single result set and returned to the user. Thus, (i) the distributed

operation is transparent to the end-user, and (ii) each demonstrator instance is allowed to have different policies and access rights for the information stored locally.

Furthermore, as far as updates on the LDAP data section are concerned, remember that, for security reasons, the Business Process module and all other Application Tier modules are not granted direct write access to the LDAP backend. Instead, the Update interface exposed by the Business Process module will process the user input and generate LDIF files, ready to be merged into the existing LDAP repositories. This last step, however, will be executed off-line; the system administrator will have to take the system off-line, do the merge, and then bring the system back on-line. Optionally, this procedure may be automated. In any case, however, write access will be granted only to the system administrator or to users local to the server(s) holding the LDAP repositories.

2.4 Presentation tier specification

2.4.1 User-Interface specification

The main method of interfacing to the Inno-Utilities server is via the exported web service(s). The server-side of the application will be implemented using Apache Foundation's Jakarta Java servlet container and Axis libraries.

Furthermore, for the implementation of the demonstrator we will build a Java-based web application to showcase the client-side of the Inno-Utilities system. Based on the requirements of the participating partners, the demonstrator front-end will support a set of predefined parameterized queries (i.e. queries on predefined combinations of index attributes). We have identified the following attributes as the three most frequently appearing in user queries: FraudstersName, TypeOfFraud, and FraudstersOrganisationOtherIdData. As of this, the demonstrator front-end will support only queries over these three attributes. Note, though, that this constraint applies only to the front-end and *not* to the query capabilities exposed by the Business Process module.

The application will communicate with the Inno-Utilities server using SOAP, as defined by the Java API for XML-Based RPC (JAX-RPC), over SSL-encrypted HTTP tunnels. To this end, we will be using Apache Foundation's Axis libraries v1.2RC3 or higher – a JAX-RPC v1.1-compliant implementation.

Since communication is done using SOAP and, consequently, XML, the presentation of results will be performed via XML/XSLT transformations, as supported by and implemented in the Apache Foundation's Xerces v2.6.2 or higher, and Xalan v2.6.0 or higher.

All Apache Foundation-derived software is distributed under the Apache Software License v2.0.

2.4.2 Message formats

The natural way of communicating with a web service is using SOAP. The current W3C standard for SOAP is SOAP v1.2. For the implementation of the demonstrator we will use the Apache Axis libraries – the upcoming Apache Foundation's SOAP implementation.

The Apache-Axis library is distributed under the Apache Software License v2.0.

2.4.3 (Optional) Message integrity

In order to prevent message tampering and protect message integrity, XML Signature standards will be applied. For the implementation of the demonstrator, we will use the Apache-XML-Security-J library Version 1.1. The Apache-XML-Security-J library supports "XML-Signature Syntax and Processing, W3C Recommendation 12 February 2002".

The Apache-XML-Security-J library is distributed under the Apache Software License v2.0. Note that message signing requires a Certificate Authority (CA) global to all Inno-Utilities demonstrator instances and, probably, another (subordinate) CA within each demonstrator instance. This may bring in additional dependencies and bindings and will surely increase the cost of building, deploying, and maintaining the demonstrator instances.

2.4.4 (Optional) Message confidentiality

In order to ensure message confidentiality, XML Encryption standards will be applied. XML Encryption provides the ability for fine-grained encryption of XML documents. It doesn't supplant the need for transport layer security such as SSL but rather complements it at the application level. For the implementation of the demonstrator, we will use the Apache-XML-Security-J library Version 1.1. The Apache-XML-Security-J library supports "XML Encryption Syntax and Processing, W3C Recommendation 10 December 2002".

The Apache-XML-Security-J library is distributed under the Apache Software License v2.0. Note that message encryption requires a Certificate Authority (CA) global to all Inno-Utilities demonstrator instances and, probably, another (subordinate) CA within each demonstrator instance. This may bring in additional dependencies and bindings and will surely increase the cost of building, deploying, and maintaining the demonstrator instances.

2.4.5 (Optional) SMS API specification

- SMS messaging will be used for immediate notification on new information available to the system, based on a publish/subscribe model in which interested parties can subscribe to information updates published as SMS notifications.

- SMS Messaging architecture
- Integration with SMS subsystem
- Subscription system
- Generator of SMS messages (filters and formats fraud information to be presented as SMS).

2.4.6 (Optional) SMS “dialogues” description

- SMS dialogues provide a form of minimal user interaction via SMS messages. Users will be able for example to subscribe/unsubscribe to specific feeds of information from their mobile phones by completing a specific interaction “dialogue” via SMS.

- Specification of SMS “dialogue” flow
- Integration with main storage and publishing system.

2.4.7 (Optional) VoiceXML API specification

- Voice XML will be used to publish fraud information to interested parties via voice.

- VoiceXML Forms
- Voice XML Grammar specification
- Security integration
- Generator of Voice XML (filters and formats fraud information to be presented as Voice XML).

2.4.8 (Optional) Telephone dialogues description

- Voice XML dialogues provide a form of user interaction via voice. Users will be able, for example, to subscribe/unsubscribe to specific feeds of information from their mobile phones by completing a specific interaction “dialogue”. When necessary calls can be transferred to specific offices/human agents and/or recorded.

- Specification of Voice XML dialogue flow.
 - Control Flow
 - Form Design
- Integration with main storage and publishing system.

References

- [1] <http://www.inno-utilities.org>
- [2] <http://snort.org>
- [3] <http://www.chaotic.org/guardian>
- [4] <http://mysql.com>
- [5] http://www.astaro.com/firewall_network_security/firewall_asg