Application Based IDS Reporting in the ERP system SAP R/3

Odd Christian Hauge

Master's Thesis
Master of Science in Information Security
30 ECTS
Faculty of Computer Science and Media Technology
Gjøvik University College, 2007

Avdeling for
informatikk og medieteknikk
Høgskolen i Gjøvik
Postboks 191
2802 Gjøvik

Faculty of Computer Science
and Media Technology
Gjøvik University College
Box 191
N-2802 Gjøvik
Norway

Abstract

Ordinarily, IDS deal with threats from sources external to the organizations' computer networks. Still, it is common knowledge that organizations' internal threats pose a greater risk and higher costs in lost revenue. External threats tend to get more publicity and focus since the internal problems in the organizations are typically kept confidential due to concerns with loss of reputation and good standing with the public or the marketplace.

In addition to the risks involved with lack of internal control in computer networks, there are two main drivers for such a type of IDS. For one, the IS departments and IS consulting firms have to report deliveries of SLA (Service Level Agreements) to management. Parts of SLA status meetings can cover reporting of intrusions and misuse in the computer systems. Secondly, the concept of compliance has, in fact, become a major factor in computer systems management for larger organizations. The SOx (Sarbanes Oxley) legislation for corporations registered on the NYSE is a major initiator to this trend.

The largest share of assets in the industrialized world is managed in ERP systems. The most widely used ERP system, is SAP. This thesis presents a prototype IDS solution based on using SAP's own Security Audit Logging, in-house defined access roles, and the organization's own SOD matrix. The research for this new IDS solution compares the performance of running the system in standard mode versus running it with anonymization and a false positive reduction trait.

Sammendrag

En IDS vil vanligvis varsle om tegn på innbrudd som kommer fra kilder utenfor en organisasjons egne datanettverk. Misbruk og innbrudd er likevel mer sannsynlig fra en kilde internt i en organisasjon. Omkostninger forbundet med internt misbruk er ofte høyere, men generelt har oppmerksomheten likevel en tendens til å dreie seg mot truslene som kommer fra kilder utenfor organisasjonens egne datanettverk. Interne problemer på dette området vil vanligvis holdes skult på grunn av organisasjonens hensyn til sitt renommé.

I tillegg til større risiki med tanke på manglende internkontroll i data nettverk, finnes det to andre faktorer som taler for en IDS inne i selve applikasjonen. For det første, så rapporterer IS avdelingene og IS konsulent-firma SLA (Service Level Agreements) leveranser til ledelsen. Deler av SLA statusmøter kan inneholde rapportering som omhandler innbrudd og misbruk av informasjonssystemene. For det andre, så har compliance konseptet faktisk blitt en hovedsak i forvaltningen av informasjonssystemer i større organisasjoner. SOx (Sarbanes Oxley) lovgivningen for aksjeselskaper som er registrert på New York børsen er en pådriver til denne trenden.

Den største delen av aktiva i den industrialiserte verden er forvaltet i ERP systemer. Det desidert mest utbredte ERP systemet er SAP. Denne masteroppgaven legger frem en prototyp for en IDS løsning basert på SAPs egne sikkerhetslogging, egenutviklede aksessroller og organisasjonens implementerte SOD matrise. Forskningen i forbindelse en sammenligning av ytelsen til systemet når deg brukes i standard modus kontra anonymisert og en ”false positive reduction” egenskap.

This thesis is dedicated to my wife and best friend, Kari Jean.

You dared me to dream big and you told me go for it!

Acknowledgement

Thank you…

Bente K. Kleiven for massive help with ABAP coding.
Terje Broch for Enabling my Studies.
Morten Jostein Aronsen and Johnny Roger Johnsen for ABAP help and hints.
Hydro IS Partner for Accommodating my Information Security Study.
Sidsel & Einar Hassel Hauge for “putting me up”. Thanks for all the good times.
Even Christian Hauge for drawing the tickets used for anonymization by substitution.
It’s all fun & games.

Table of Contents

List of Figures

List of Tables

List of Charts

Introduction

Several cases of fraud involving electronic transfer make us wonder how the fraud can go on for many years and why, when the amounts of money involved are huge, does this go undetected? Even after years of widely publicized scandals, the threat of internal misuse of computer systems is faced with the following issues: [2]

- Inside threat is an increasing problem
- Underreporting persists
- The success rate in attacks is high
- The cost in lost revenues is also high

This project has developed an IDS internal to an ERP application as a direct response to the problem description above. Furthermore, the IDS can be used on a regular basis in security monitoring, which is on the agenda for security audits and in SLA meetings. Corrective actions based on the logging and IDS reporting will also enhance the general awareness of information security issues in the organization. The main motivating factor for the project is to create an effective tool for sustaining a meaningful process for managing security logging as one of the main components for security monitoring. The logging in itself serves no purpose other than being possible source for a potential investigation. If the security log is managed and processed properly, it could in itself trigger and show the need for corrective actions and thus the organization could choke an influx of internal threats.

Our IDS is an original, application based for SAP and includes functionality with characteristics for both anomaly detection and misuse detection. The data handled for reporting incidents must be treated with confidentiality, and it is therefore necessary to be able to run this IDS with anonymization in the reporting module. This anonymization is based on simple, random substitution. To increase performance of the system, one facility for reduction of false positives is offered for each characteristic of the IDS. We want to investigate and measure if the system's performance with reduction of false positives and anonymization activated. Our research question is "How is the performance of this SAP IDS when running with reduction of false positives and anonymization?".

Most larger organizations using SAP have already established most of the building blocks used in our IDS prototype. At the same time, most of these organization have not considered that these building blocks can be used for an application based IDS. This IDS utilizes the security audit logging, access roles, an SOD matrix, change records, and considers a more general security policy. Typically, upon an ERP implementation the company acquiring this software will analyze how to best ensure the needs for segregation of duties, (SOD). Internal audit will typically propose that an SOD matrix is formalized, if the implementation project has not done this work. The SOD matrix describes how to segregate tasks, such as for instance, purchasing and goods receipt, between workers. Another obvious example for segregating tasks is to ensure that access to create accounts is not granted with access to post payments. In order to protect the organization against fraud and theft, it is necessary to analyze the level risk for such activities. In the SAP system there are many SOD risks such as the two examples described, so many that a whole library of these risks has been created in a tool called the Virsa Compliance Calibrator.

When investigating incidents that might indicate fraud or theft, we anticipate a known problem with IDS – they generate false positives. Consequently, this project will explore ways of reducing this problem, but also offer an method for analyzing the results with anonymization. The findings from our IDS system can be used by external audit companies for running statistics and measure level of compliance in the SAP system. When submitting these reports to the audit company, it may be necessary to anonymize some information in order to keep confidentiality level such that the incidents cannot readily be traced back to a certain user. Additionally, anonymization can prevent irritation among the most seasoned users of the system. Naturally, people have a negative reaction to questioning perceived as inquisitional when they are merely performing the job duties that have been placed upon them.

By anonymizing the report it is also acceptable to make this research paper public and display parts of the logging and reporting. The bits of information in the figures and details of this report is not enough to reveal any sensitive personal or company information, but the information provides better illustrations and explanation of our research project. The complete set of log files is retained by the author and the supervisor as research data. Complete source code listings for the different parts of the IDS can be obtained from the author.

Our hypothesis, then, states that is possible to make a simple application based IDS reporting facility in an ERP system to better monitor signs of misuse and anomalies, and it is possible to increase performance in this system by reduction of false positives even when running the system with anonymization.

What is good IDS performance? We base our definition of IDS performance to the four factors described in [5] and [6], where the IDS is said to be

- Comprehensive
- Timely
- Comprehensible
- Accurate

The goal for the project is to present a method for detecting the most common intrusion detection issues in the company's ERP system, while maintaining simplicity. The goal can be summarized in the four bullet points below:

- Simplicity
- Automate security monitoring for SLA meetings and security audits
- Create an effective and proactive process for managing the security audit log
- Improve overall security awareness in the organization

We chose the three most commonly asked intrusion detection related questions raised by the firm in SLA meetings, and the three most commonly asked questions raised by the security audit teams investigating the SAP systems. One False Positive Reduction measure is tested for each type of incident reporting. Upon completion of the project we see that the auditors is interested in the misuse part of the IDS, while the system owner, or customer, is more interested in the anomalies.

The case for simplicity is multilateral. We had to make the project manageable within the time frame of the period allotted for a masters thesis. The system cannot be as comprehensive in design as would be required from a commercially available product. We made an effort to make the research comprehensible to as wide an audience as possible. The areas of ERP security administration, ABAP programming, ABAP workbench and -customizing, logging configurations, SOD risks, etc can become quite complex, yet the incident reporting for our prototype system is easy to understand. While the IDS reporting system is easy to understand, producing the basis for the system is not a quick and simple task. To make a further simplification, we decided to include only one simple false positive reduction for each misuse and anomaly characteristic.

1.1 *Some Definition for this Project*

Intrusion Detection in computer systems and networks is defined as “any set of actions that attempt to compromise the integrity, confidentiality, or availability of a resource” [24]

Access Roles, RBAC: Computer Access and Authorizations concept as described in [7] and [8]. The role-based access control is the main tool for enforcing the security and authorization policies in the SAP system. It segregates access to perform transactions and postings, and what type of displays a user can produce with his or her user account.

Application-Based IDS, “A host-based intrusion detection and prevention system that performs monitoring for a specific application service only, such as a Web server program or a database server program.” [4]

Customizing - Method in the R/3 System with which you install SAP functionality in your company quickly, safely, and cost-effectively, tailor the standard functionality to fit your company's specific business needs, and document and monitor the implementation phases in an easy-to-use R/3 project management tool. Customizing in the R/3 System is done via the Implementation Guide (IMG).

Emergency logon: User account with all access rights, to be used in an emergency situation. This accounts is unlocked after proper approval and monitored while in use. The user account is locked and deactivated as soon as the emergency is handled, and the log from its use is stored for audit trail purposes.

End user: A non-IS worker in the organization who is using the computer application to perform business functions.

False positives, “An instance in which an intrusion detection and prevention technology incorrectly identifies benign activity as being malicious.” [4]

IDS, “Intrusion Detection System: Software that automates the intrusion detection process.” [4]

IS user: A consultant or worker in the IS department involved in management, development, and operation of the computer system.

Privileged user: A user of the application with dual skills and access to perform both business critical and application critical functions.

RFC, Remote Function Calls, is communication between applications of different systems in the SAP environment, and it includes both connections between SAP computer systems, and between SAP computer systems and non-SAP computer systems. RFC is the standard SAP interface for communication between SAP systems. The RFC calls a function to be executed in a remote system.

SOD, Segregation of Duties principles to enforce separation of postings in the system between workers to secure against fraud, undue influence, misappropriation of funds, and embezzlement.

SOD Matrix, list of roles which shows which types of access that will introduce SOD risk in the computer system.

False negatives, the IDS is missing incidents in the report engine.

User Master Record, a user account registry with for instance access rights, defaults, parameters, and address information.

Workbench Organizer - Tool for managing central and decentralized software development projects in the ABAP Workbench.

We define dualism, for the purpose of this paper, as term for describing a predicament in many systems to which is often described in audits: The operation of the system itself should be handled in the IS department while the operation of the business should be handled outside the IS department.
We therefore have to classify the most crucial transactions as either postings for the business or IS operations for maintaining the application. This definition is in contrast to a generally held interpretation of the word dualism, which is commonly oriented towards the fields of theology and philosophy , but the concept about dualism is the same:

> "In general, the idea is that, for some particular domain, there are two fundamental kinds or categories of things or principles." [11]

1.2 Assumptions

An IDS assumes that normal and intrusive activities are observable in the computer system. Based on audit trail facilities and logging we have looked at in SAP, we believe that the updates performed in the application layer is recorded and retrievable in this version of the system.

An important performance feature of IDS is comprehensiveness. We do not claim that our prototype IDS covers all types of misuse and anomaly that could occur in the application layer, however, it is important that nothing is missing in the log, otherwise the system will produce false negatives. We assume that the logging performed in SAP R/3 security audit logging is complete and accurate. This assumption is made after an in-depth review of all SAP notes related to Security Audit Logging. This review is described in chapter 2, about the review of previous work.

The same performance criteria holds for the Compliance Calibrator which maps out the SOD risks. This tool is the state of the art for analyzing segregation of duties, SOD, risks in access roles. We observe that the Compliance Calibrator is comprehensive for the SAP standard transaction codes and authorization objects. Each SOD risk is described in a short text and full text field. Since SAP has purchased the company that developed the Compliance Calibrator, we have good reason to believe that this product has achieved a good level of trust.

Based on experience, we see that access rights distributed to end users is a rather frequent activity. The access roles that are assigned to the end user records (user accounts), are more static in nature. The updates in the access roles are infrequent. The small class of users that can be described as “heavy SAP users” or super users is a group of employee with less turn over in the organization. The assumptions we can make is that there will be a certain amount of errors in the reporting due to changes within the time span for the report. The log itself and the Compliance Calibrator are correct, but most of the other building blocks for the IDS assume that there are no changes in access for the time period reported. It takes the state of the system at the time of the reporting as correct for the whole reporting period. We should then keep in mind that there is a chance for small percentages of errors in the report, and the reports should cover a few days or weeks rather than months at a time.

The audit log itself must be in a secure place, and we believe it is for SAP. Attackers have a history of change the audit logs to hide their traces [12], but the audit log for SAP is stored at the UNIX level, and only the operations managers have access at the operating system for this system. Still, care must be taken to ensure that the audit files collected and stored in tables for this project, is not accessible to anyone outside the project group.

1.3 Problem Description

Organizations that utilize ERP systems typically have security audits. Additionally, if the operation of the SAP system is outsourced, there are SLA meetings on a regular basis. The IS security staff is summoned and asked to produce information and reports in the areas of intrusion detection and execution of access administration. This information has a tendency to be gathered manually during audits, and it is a time consuming effort. Audits have an uncanny ability to pop up at the least convenient time for the IS staff, for example during the implementation of a bigger project. The ability to collect information and produce the reports demanded by the auditors are hampered by the sheer amount of time this activity requires. Consequently, there is a case for automating the reports that the auditors request most frequently.

As far as the SLA meetings, the question of intrusive activity is raised in our case every month, however, there is no requirement to monitor this activity. For the organization in our case, the IS staff was somewhat dumbfounded and uneasy when they had to answer this question each month without really referring to any type of specific security monitoring procedures or activities in that area. What then, if serious breaches of the security policy are discovered long after they happen. This could seriously embarrass and tarnish the reputation of the local IS professionals.

The research in [14] concludes that failure to act on indications of criminal or unethical practices is itself unethical, even if one has no direct part in the practices. Ethical behavior requires that you do right, not that you simply don’t do anything wrong. The company of which we are doing this research, used to have an internal IS department, but had to outsource this activity to the corporations subsidiary for IS services. The company retained ownership of the SAP system and operated under the assumption that the IS staff still had the responsibility of monitoring and reporting the computer security.

The above assumption was problematic for the outsourced company. The cornerstone marketing strategy for the outsourcing subsidiary was to create a “product” for each services provided. This strategy was said to be a very effective cost reduction measure. In this framework, the security monitoring should not be provided unless this service was offered as a product that the customer was willing to purchase. The outsourced subsidiary thus called upon the IS staff to do the security monitoring only if the customer was willing to pay for this “product”. The staff could only do the work provided that they had a proper product number to which they could bill the customer for their time.

The security audit activities and SLA meetings both show that there is a considerable cost and ethical dilemma involved in the SAP security monitoring activity for the company. Much information about the activities and potential security issues can be discovered in the security audit logging of the SAP system, but this effort is time consuming. A highly efficient and automated processing of this log is the key to a satisfactory security monitoring level, and a monitoring activity level that does not infringe upon the cost and ethical issues.

In the case of the information gathered from security audit logs the time consumption problem is multiplied by the number of application servers, since each server generates one log file for each 24 hour period in its own file system. The SAP standard reporting facility for the security audit log only collects data for the application server currently in use by the operator. In other words, you have to log on to each application server and produce the report for each server, and eventually try to combine the log by downloading the log files to PC format and combine them in a spreadsheet for further processing. The PC format, however, is grossly inadequate since we need to be able to process several million records at a time if we want produce reports for several days at a time.

The management of the security audit logging is furthermore exasperated by the fact that the task of monitoring log files is quite monotonous work. This activity tends to deteriorate in IS departments over time. An efficient application based IDS should reduce the time involved in this monitoring activity considerably, and minimize the need for manually checking the log files directly. It is also capable of finding issues that are too complex or too hidden to be discovered manually.

When the auditors are on site and they are collecting data and reports, more or less personal or company sensitive information regarding the organizations staff and their work processes could be revealed. This is problematic since most of the auditors are external to the company, that is, they work for audit firms.

The relevant information during SLA meetings includes attempted break-ins, attempts at unauthorized postings, and attempts of illegal downloading of data. During security audits, however, the staff may be asked to participate in more complex security analysis including investigation of self administration of access rights, segregation of duties (SOD) risks, and tendencies for one person to both run computer operations transactions and financial postings. The last issue described, were one person committing computer administration tasks and end user postings simultaneously, we call dualism in our research paper. The term is used due to the local jargon in the company, but it is possible that this term is also used elsewhere to describe the same issue.

Dualism is problematic because normally an IS consultant does not have the credentials for understanding the company impact of financial postings or other transactions. Likewise, the end users typically do not understand the consequences of computer systems operations tasks. Consequently, many IS security audits has a focus on proper separation of IS and business tasks between its departments and staff members.

1.4 High Level Description of SAP IDS

In short, the IDS is based on an adapted implementation of SAP's own Security Audit Logging with additional building blocks. Existing SAP standard user access tables and tables produced for this project are parts of this system. The tables developed for this project is a list of SOD risks, a substitution table for anonymization. The systems anonymization feature is based on a simple substitution of the characters for relevant data in the log. The SAP standard tables used are the tables for access roles and transaction codes, user authorization update history, and user groups tables.

For added agility and flexibility in the system, we gather all the log files from all application servers and place the records in one table customized for this IDS prototype. The focus area for the IDS on the particular system we are researching does not take into account remote function calls (RFCs), since this, so far, has been outside the scope of attended SLA meetings and audits. It is for our purpose, convenient to avoid RFC logging since this more than doubles the amount of records in the log.

The data collector in the IDS prototype only fetches unique records from the security audit log files. The duplicate records are gathered in an error log, and this error log is examined for multiple attempts to crack passwords. If the error log is large, it would indicate automated intrusive attacks on the system.

We propose only one measure of reduction of false positives for each incident report for the sake of simplicity. The incident reports can be classified into misuse detection and anomaly detection. When the project part of the research was completed we realized that the SLA meetings typically ask for information in the anomaly detection area, while the audits are more interested in the misuse area. Reporting in the anomaly area is more straightforward than misuse, since the misuse detection requires more components in our IDS. For anomalies we can for the most part use the security audit log directly.

1.4.1 Misuse Detection

The three types of incident reporting in the misuse detection facility is

- Self administration of access rights
- High risks in Segregation of Duties
- Violations in Dualism Principle

The self administration of access rights simply checks if a user has run the user administration transaction code on his or her own user master record. The operator of the IDS can select reduction of false positives for any of the incident reports. In the case of self administration of access rights, the IDS performs an additional check of the incident against the user authorization update history table, USH04. This table shows actual user authorization changes and the IDS compares each user ID of the authorization update against the field for the user ID of user administrator in each record.

The high risk of segregation of duties is predetermined by checking all the transaction codes available in the access roles with a tool called the Compliance Calibrator by Virsa Technologies. The result is downloaded from the Virsa Compliance Calibrator, to a PC spreadsheet program for reformatting. The reformatted spreadsheet is then uploaded to SAP, into a table customized for our IDS. The detection engine in the IDS gathers all instances of SOD high risk violations and reports all incidents and how many users are involved in creating all the incidents. The incident report does not consider whether the user has access to the particular tcode or not. To reduce the number of false positives, the system removes all users who have approved access to run transaction with high SOD risk. These users are registered in the SOD user group.

Breach of dualism happens when one user is attempting to run postings and IS operational tcodes in the same time period. The incident report is produced by checking the access roles in the SAP standard table AGR_TCODES. The naming convention of the access roles used indicates whether it is designed for IS or for end user execution. A false positive reduction is issued by removing all users classified as 'privileged'. In this context a privileged user obtained approval from management to run postings and IS operations transaction codes.

1.4.2 Anomaly Detection

The three types of incident reporting checked and classifies as anomaly detection is

- Logon Failures
- Authorization Failures when executing transactions
- Excessive downloads

The incident reporting for critical logon failures and authorization failures are obtained from the security audit log table without any further resources. When run with reduction of false positives the user master record table, USR02, is checked for existence of the user ID. The idea is to exclude typing errors for the user id at logon time.

Authorization failures are generated when a user is attempting to run a tcode which either does not exist or is not available in the users access rights. Again, the typing errors are eliminated when running in the false positive reduction mode.

Excessive downloads is a matter of assessment. The log itself is the resource, just as the other two anomaly detection characteristics. The reduction of false positives is reducing the users in question to the contractors. The biggest concern in this area is that contractors are downloading company data for their own knowledge management and analysis.

1.5 About the Infrastructure

The research for this paper is based on work we have done at Norsk Hydro Oil & Energy. From this point on we refer to Norsk Hydro Oil &Energy as just O&E. O&E is the owner of the SAP system, but on October 1, 2007, the oil and gas division of O&E is merged with Statoil, to form the company StatoilHydro. The details surrounding the future of the system we have used, or further development of our IDS work is unknown.

We have used O&E's SAP update system, R/3 Release 4.6C, with the IBM's version of the UNIX operating system called AIX. This SAP system has four application servers and one database server. The users do not connect directly to the database server. The underlying database is Oracle Release 10.2, and it is approaching one terabyte in size. This SAP system is in use by ca. 7,500 employees and consultants, and all the primary modules of SAP has been in use for the past ten years. However, there is virtually no activity in the Sales & Distribution part of the system, since the O&E is an *upstream* organization. The added complexity and significance for this particular system is in the area of Joint Venture Accounting for the North Sea Oil and Gas operations.

2 Review of Previous Work

We have looked for an IDS specifically designed for SAP R/3, but we have reached the following conclusion: An application based IDS based on SAP's own security audit logging in concert with other access and authorizations information in this ERP system is not available. We therefore view our system as an original prototype. Although the system we are proposing is by no means all-embracing, we intend to prove that we can develop a reporting facility with IDS characteristics. For better performance, the system is equipped with both anomaly detection and misuse detection. As there is no IDS for SAP R/3 as far as we know, we can examine the previous work done with the components of the system we are implementing.

Since our results are based on information we extract from the Security Audit Log, we had to carefully plan and implement this functionality of the SAP R/3 system. This chapter consequently deals with the evolvement and experience of our security audit logging in SAP. Incompletion or other errors in logging must be avoided by any means, so this implementation must be painstakingly accurate. Any shortcomings in the logging facility will produce false negatives.

In SAP's own technical support web pages there are many documents describing challenges, bugs, and configuration details in the logging process. We downloaded all these documents and examined each one of them carefully to see if anything applied to our configuration.

In the beginning of the project, we frequently ran into the problem with of "empty audit files" as described in SAP notes [28]. This problem was not related to display issues in the SAP GUI interface or conversion issues, [25][26], but rather that the logging was not active and the files on the OS level was not being generated as described in [32]. After running through a helpful checklist in this note [28], we finally managed to track down some details in our configuration that was missing. Also, it is not self evident to a person how to establish the logging, whether he or she is familiar with SAP or not. The procedure for saving the logging configuration and activating the logging is not always consistent and user friendly. Sampling tests of the logging had to be exercised time and again. The note [33] for failures in changing security audit parameters was explored.

When the logging itself was producing records in a satisfactory manner, our next concern is system performance, which in the case for SAP is actually response times. The Security Audit Logging was previously not in use mainly because the system administrators had performance related concerns. A proof of concept had to be presented in the test system, and the logging in the production environment had to gradually be expanded to finally include all relevant details for the research. The performance aspect resource in this development considered notes [36] and [39], in addition to the spot checks we performed by frequently refreshing and checking the "Computing Center Management System", CCMS, load distribution information and response times. It was by note [36] we finally decided not to record the remote function call, (RFC), events, due to their large number of entries.

The "Frequently asked Questions" regarding authorizations in general and Security Audit Logging in particular is listed in Notes [47] and [38]. It includes a discussion regarding maxing out the logging file size. Potentially, this is a severe problem since the file closes the recording simply terminates.

We are particularly interested in logging user actions, so we studied additional methods to track these actions. [30] For instance, logging changes to tables in the ABAP Workbench, SE11 and subsequent evaluation with transaction SCU3. And, for instance tracking statistical data for user behavior with STAT.

The system log must also be taken into consideration, to understand system related security events. It is necessary to get familiar with how the two logging systems differ, and realize that the security audit logging complements the system log. The system log primarily record information that may signal system problems, and is not deactivated. The system log does, for instance, not contain any personal data, so the question of anonymization should not have interest when using the system log for forensics and investigative purposes. [30][48]

[35] is a note showing a potential problem with log management, were "Transaction SM18 always deletes all files, regardless of their specified age", so not to risk this problem we decided to keep all log files until all of them were processed for our purposes. An attempt was made at the end of the project, deleting all log files prior to January 1, 2007. This clean up work fine, the other log files remained on the system. When purging log files the operator can also run into authorization problems if access to authorization object S_ADMI_FCD is not provided. [31] Also, SM18 transaction could produce errors with minimum age as described in [34].

3 Experimental Work

3.1 Prototype, Detailed Description

Rebecca Gurley Bace describes an IDS with both a misuse engine and an anomaly engine with a flow diagram. [5] A similar representation is made for our prototype in figure 1. The misuse engine is on the left side and while the anomaly is on the right.

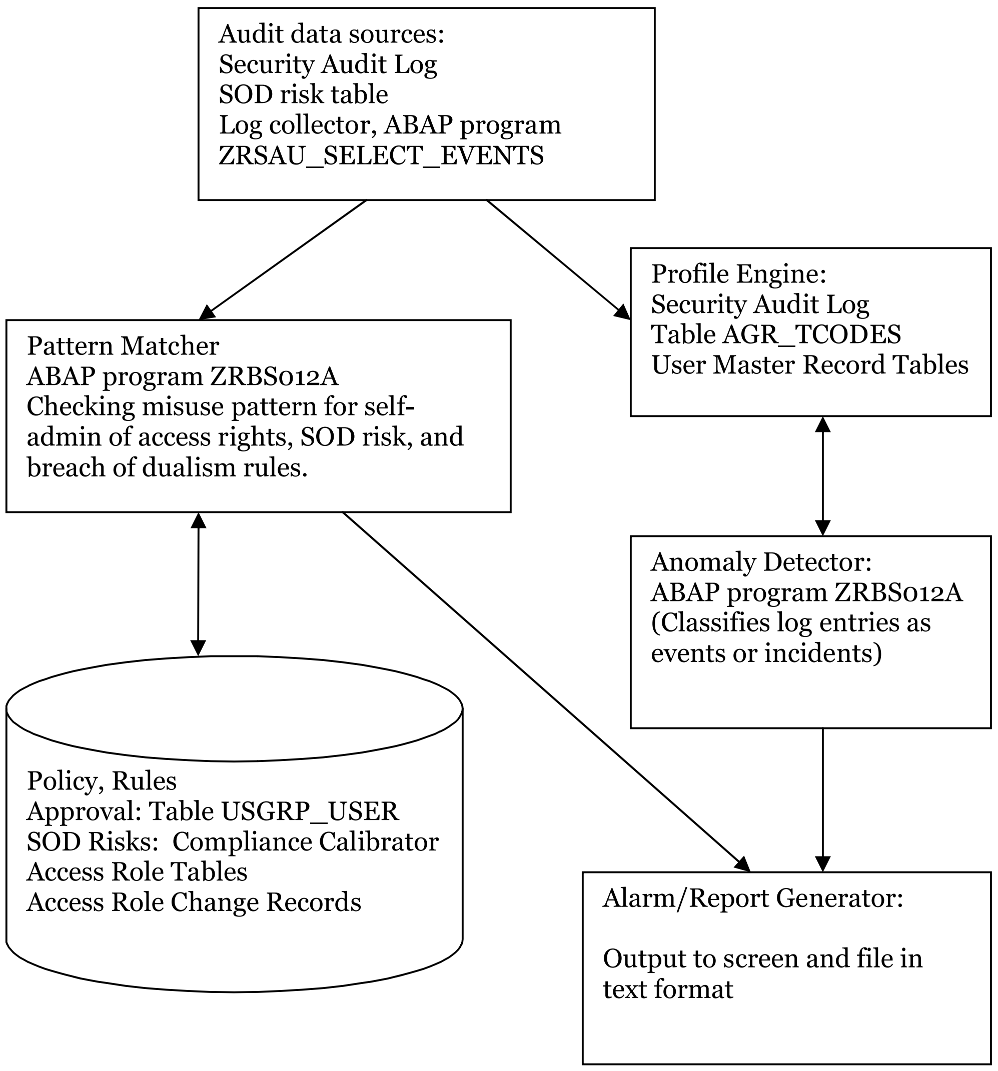

Audit data sources:
Security Audit Log
SOD risk table
Log collector, ABAP program
ZRSAU_SELECT_EVENTS
Profile Engine:
Security Audit Log
Table AGR_TCODES
User Master Record Tables
Pattern Matcher
ABAP program ZRBS012A
Checking misuse pattern for self-admin of access rights, SOD risk, and breach of dualism rules.
Anomaly Detector:
ABAP program ZRBS012A
(Classifies log entries as events or incidents)
Policy, Rules
Approval: Table USGRP_USER
SOD Risks: Compliance Calibrator
Access Role Tables
Access Role Change Records
Alarm/Report Generator:
Output to screen and file in text format

Figure 1

To achieve the results we seek in this IDS, we had to carefully plan and implement the security audit logging. The planning had to take into account the risk for affecting response time on the system. Although initial testing and calculations did not indicate response time problems, we kept the amount of logging to the minimum required for our IDS. After reviewing the logging details for this IDS, we found that it matches the description for security monitoring in the standard [3] save for the RFCs (described in chapters 1.1, 1.4, and 2) and the time and date stamping of log offs.

The gain in efficiency of the logging was considerable when excluding the RFCs and the log offs; more than a 60% reduction of the size of the log files. Otherwise we see a great similarity between what we are logging and what is suggested in the standard as far as authorization errors, terminal IDs, failed access attempts, and so on. The file access in this IDS's case is, however, limited to monitoring the download activity.

To enable the logging when the system is restarted, certain parameter settings has to be included in the startup profile. Suggested parameter settings are available from SAP. Table 1 describe the parameter settings as submitted to the systems administrator for this SAP system. It was first implemented in the quality and assurance, and development SAP systems for testing. The gradual implementation was coordinated with the owner of the SAP system and the system maintenance schedules.

Parameters	Suggested settings
RSAU/enable Activates the security audit Value set to 0 turns log off (default) Value set to 1 turns log on	RSAU/enable = 1
RSAU/max_DISKSPACE/local Size of the daily audit log in bytes Default is 1000000 (1MB)	RSAU/max_DISKSPACE/local = 3000000, 3MB (quite large for audit login)
RSAU/local/file D:\usr\sap\<SID>\dvebmgs00\log\++++++++.AUD Defines the directory and name of audit log file Directory must be defined in operating system ++++++++ Is the date in yyyymmdd format (you actually type the “+” characters) 00 in directory name is the instance number	RSAU/local/file = Admin’s preference or <sid>\DVEBMGS02\log\++++++++.AUD (for central instance) RSAU/local/file = Admin’s preference or <sid>\D02\log\++++++++.AUD (for app servers)
RSAU/selection_slots Number of filter slots available to define audit log selections Default = 2 Max in 4.5x = 4 Max in 4.6x = 10	RSAU/selection_slots = 5 NOTE*** Do not add these parameters to the default profile or central instance will do all the logging and performance on central instance might be effected.

Table 1

A security audit filter has to be created via transaction code SM19 and the details in the filter settings is listed in appendix 8. Experimental work was done in the QA and development system to gain experience with how to read the log and how to gather the right events in the log. All along the risk issue of affecting response time in the system was tested.

The program for gathering the log files did require much time and effort from us and this program is described in chapter 3.1.1

The detection engine's characteristics is listed in table 2.

type	Pattern/Security Policy/Incid.	Action/ Description	Reduction of False Positives
Anomaly	Failed logon Table USR02	Check if ID and Password incorrect	Valid IDs only
	Authorization Failure Table AGR_TCODES	Check if tcodes are approved for business process	Existing tcodes only
	Download activity	Tally amount of users performing downloads	Check the enterprise contractors only
Misuse	Maintain own access, match user admin ID with user account ID	Check SAL table generated from security audit log files	USH04 table confirmation
	SOD risk incidents as defined by Compliance Calibrator	Check for two and three transaction code combinations in the SOD table	Is the SOD action approved?
	Dualism breach as defined by naming convention in access roles	Check if transaction codes used by one user are in IS roles and business roles	Is the action done by a privileged user?

Table 2

For easy access during the experimentation period, we collected the most frequently user transaction codes in our own IDS menu, as displayed in figure 1. Here we could access the log collecting program, the SOD upload program, the detection engine and the security audit log configuration facilities directly.

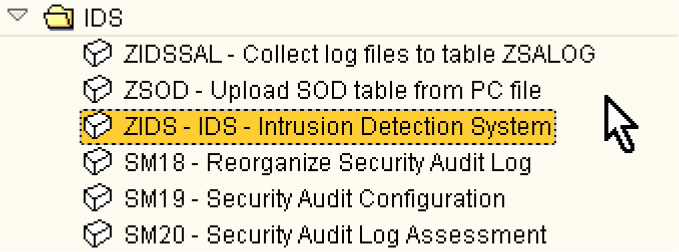
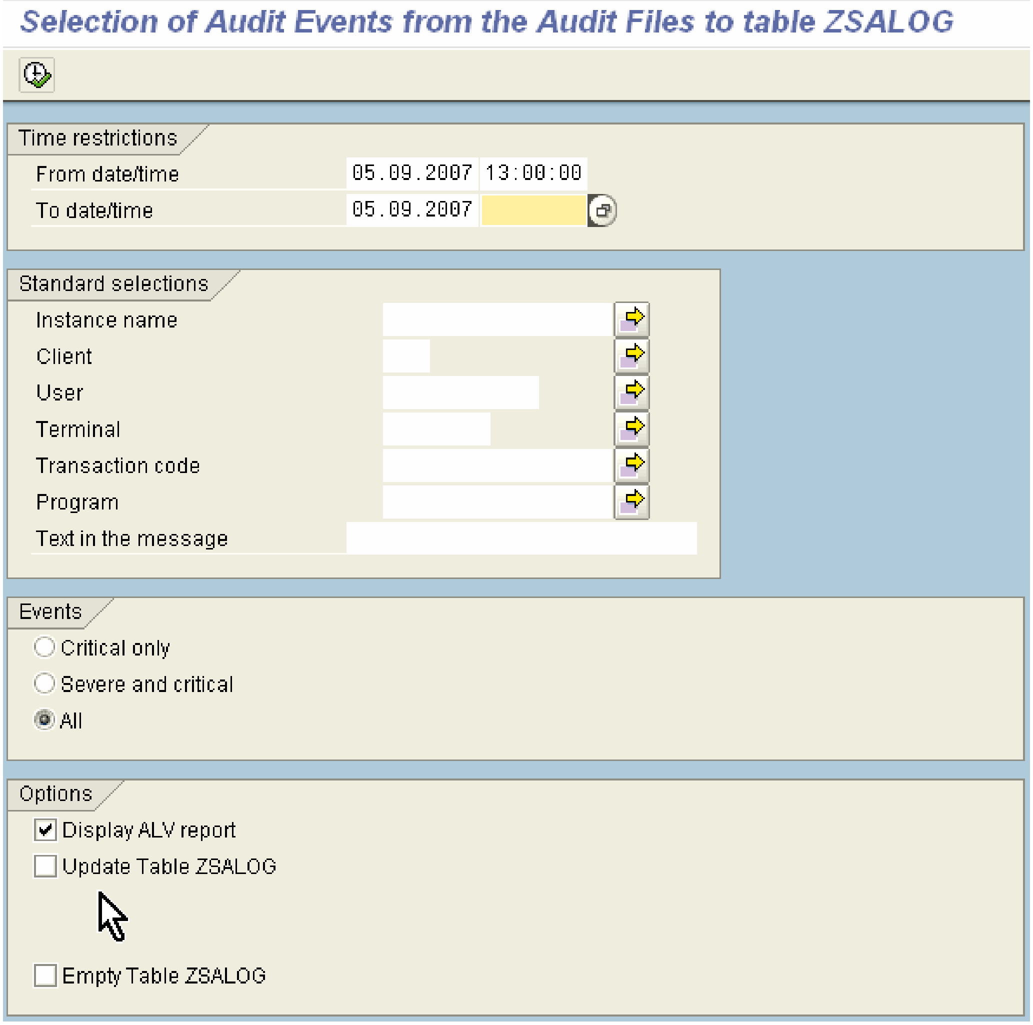

IDS
ZIDSSAL - Collect log files to table ZSALOG
ZSOD - Upload SOD table from PC file
ZIDS - IDS - Intrusion Detection System
SM18 - Reorganize Security Audit Log
SM19 - Security Audit Configuration
SM20 - Security Audit Log Assessment

Figure 2

3.1.1 Gathering Log Files

The log files from each application server are collected and appended to a large table with security audit logging records using an in-house developed ABAP program called ZRSAU_SELECT_EVENTS. For easy access we created a transaction code for this program called ZIDSSAL. The security audit logging creates one file for each day and night (24 hours), for each application server. The files are in text format and each line feed is specified with a q-like character. We treat each line as a record for a database table. The ZRSAU_SELECT_EVENTS program gathers these "records" from all the log files for a user specified time period and appends the records to table ZSALOG. A screen shot of the user interface to this program is shown in figure 3, while the core logic is listed in appendix 3. The code for this program is by far the lengthiest of all the programs in this system.



Selection of Audit Events from the Audit Files to table ZSALOG

Time restrictions

From date/time 05.09.2007 13:00:00

To date/time 05.09.2007

Standard selections

Instance name

Client

User

Terminal

Transaction code

Program

Text in the message

Events

○ Critical only

○ Severe and critical

● All

Options

☑ Display ALV report

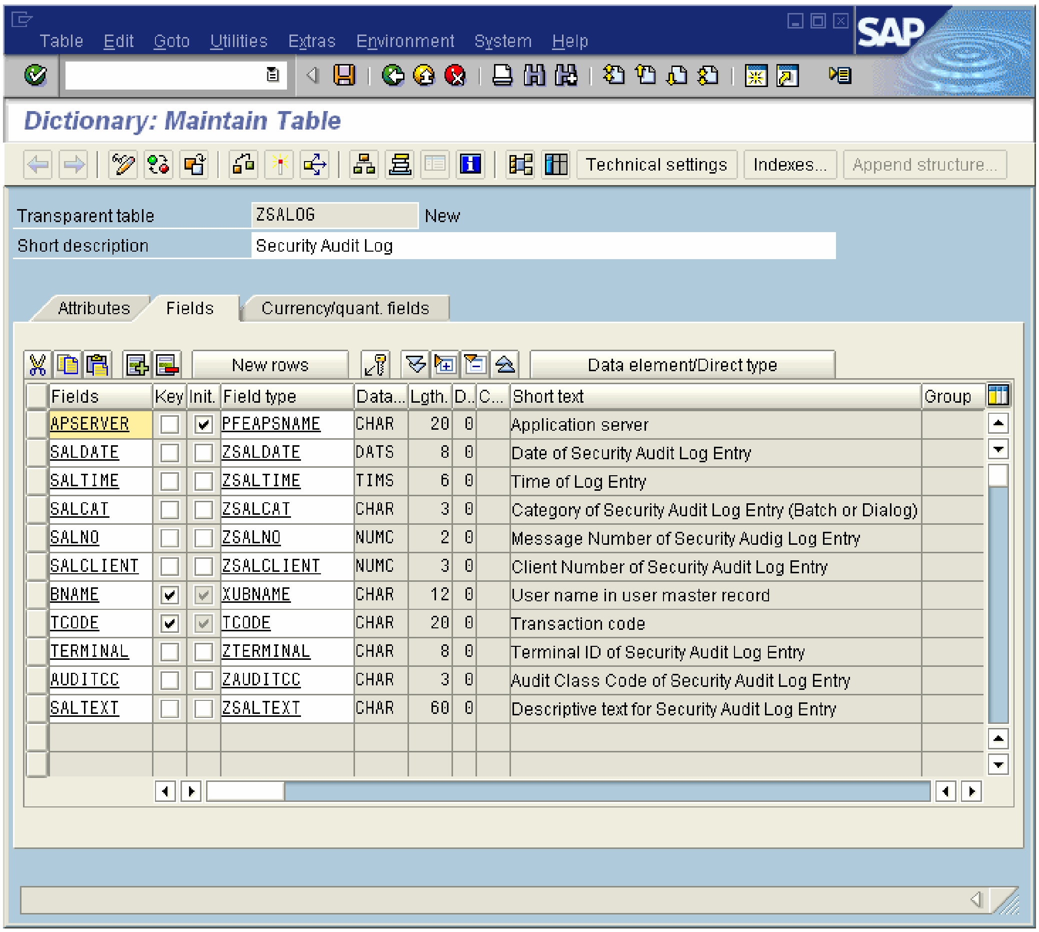
□ Update Table ZSALOG

□ Empty Table ZSALOG

Figure 3

When executing the log collector with the update table option and a valid time and date span, records from each log file is extracted and the ZSALOG table is populated. It is necessary to "flush" the table by separately executing the program using the "Empty Table ZSALOG" option at the bottom of the input screen, before a new sample period is collected. This functionality was used and tested against a manual flushing of the table. The result was the same. The functionality for "Standard selections" and "Events" was gradually excluded from the experimental work, since these functionalities did not add much value to the research we pursued.

An ABAP Dictionary screenshot of the ZSALOG table is displayed in figure 4. This table is designed to hold ten million records, which we find to be sufficient for processing one month of information.



Dictionary: Maintain Table

Transparent table ZSALOG New

Short description Security Audit Log

Fields	Key	Init.	Field type	Data...	Lgth.	D...	C...	Short text	Group
APSERVER	□	☑	PFEAPSNAME	CHAR	20	0		Application server	
SALDATE	□	□	ZSALDATE	DATS	8	0		Date of Security Audit Log Entry	
SALTIME	□	□	ZSALTIME	TIMS	6	0		Time of Log Entry	
SALCAT	□	□	ZSALCAT	CHAR	3	0		Category of Security Audit Log Entry (Batch or Dialog)	
SALNO	□	□	ZSALNO	NUMC	2	0		Message Number of Security Audig Log Entry	
SALCLIENT	□	□	ZSALCLIENT	NUMC	3	0		Client Number of Security Audit Log Entry	
BNAME	☑	☑	XUBNAME	CHAR	12	0		User name in user master record	
TCODE	☑	☑	TCODE	CHAR	20	0		Transaction code	
TERMINAL	□	□	ZTERMINAL	CHAR	8	0		Terminal ID of Security Audit Log Entry	
AUDITCC	□	□	ZAUDITCC	CHAR	3	0		Audit Class Code of Security Audit Log Entry	
SALTEXT	□	□	ZSALTEXT	CHAR	60	0		Descriptive text for Security Audit Log Entry	

Figure 4

3.1.2 Detecting Maintenance of Own Access

For the past two years, the auditors first check on this SAP system is an investigation into whether anyone has granted himself or herself access on the system. Since this is viewed as a serious breach of the security policy, we include this activity as the first characteristic of our IDS prototype.

The code in the detection engine performs three main operations on the security audit log:

1. Selects all log entries with user admin where user administrator equals name of user account.
2. If the FPR flag is set, the SAP standard table USH04 is checked for actual updates.
3. Anonymizes the fields user name, terminal ID, and description if the anonymization field is set.

The records in the security audit log extracted for this operation are those which include running SU01, the user administration transaction code. If the person running this transaction is changing his or her own user master record, there may be a potential problem. In some cases a person will run this transaction to make corrections to the address, phone number, or other details not related to access rights.

The SU01 transaction code is typically restricted, and may have additional security facilities attached in the authorization. Such is the case on the system we used for our IDS. As expected, this part of the IDS produced the least amount of incidents. At the end of the month of March, an in-house developed feature to secure against this type of misuse was introduced, so the second half of the testing period produced no incidents at all. The second half is therefore omitted from the table and the graph in the results and discussion chapters.

3.1.3 Detecting SOD Risks

A user authorization group called SOD was introduced to the system, to identify users who have approval for running transaction codes with SOD risks as defined by the Virsa Compliance Calibrator. A special table called ZSODTCODES3 was created and populated with the SOD risks. The records in this table contains three fields for transaction codes and one field for a risk description. The transaction code fields are intended for transaction codes which will result in high SOD risks as defined by the Compliance Calibrator. The transaction codes create SOD risks when used in two part or three part combinations.

The work involved in identifying the transaction codes with high SOD risks is not a small task. We gathered a list of all transaction codes in use on the SAP system and generated one large access role for all these transaction codes. This role produced as expected a large number or SOD risks, and we extracted all the high risks from the result. We then downloaded the result from the Compliance Calibrator to a PC file. This file was then reformatted to fit into a PC spreadsheet. We developed a new ABAP program for uploading this spreadsheet into the ZSODTCODES3 table.

The code extracting SOD risks performs the following operation

1. Selects all log entries with SOD risks in use of two or three tcode combinations.

2. If the FPR flag is set, the users in the user authorization group SOD are omitted.
3. The user name, terminal ID, role, and description in the record are alternatively anonymized.

So, the reduction of false positives in this case involves checking the user master record for membership in the SOD authorization user group. The membership in this group shows that the user has approval for running operations that includes such SOD high risk. The SAP standard table USGRP_USER houses the authorization user groups and user ID memberships to these groups.

3.1.4 Dualism, IS Operations vs. Postings

In this paper we have defined "dualism" as the running both IS operations transactions and actual business postings. The transaction codes run by the IS department staff are segregated from the transaction codes run by the business, in the naming convention of the access role. All access roles with their affiliated transaction codes are found in the SAP standard table AGR_TCODES.

For detecting dualism issues, the code performs the following extraction

1. Selects all log entries for users running tcodes from both IS and end user roles.
2. If the FPR flag is set, the users in the privileged (approved) users are omitted.
3. The user name, terminal ID, role, and description in the record are alternatively anonymized.

In this case, the reduction of false positives is done by checking the user master record for membership in the PRIVILEGED authorization user group. The membership in this group shows that the user has approval for running operations that could result in breach of the dualism principle. The SAP standard table USGRP_USER houses the authorization user groups and user ID memberships to these groups.

3.1.5 Both Privileged User and SOD User

Finally, a comment has to be added for an anticipated class of users that could be both approved for violating dualism principles and running transactions that produce SOD risk. We decided to crown the privileged user as the higher, meaning that if a privileged user runs transaction codes with SOD risks, the user is treated as one who has permission to run these transactions. For the system we used at the time, this was simple and convenient since there were only two privileged users.

3.1.6 Detecting Logon Failures

Critical logon failures are set to be registered in the security audit log. This includes logging all logon attempts that fail due to entering wrong or non-existent password or user ID. This version of the SAP system is not case sensitive, so the system disregards the difference between upper case and lower case letters. The log also registers user account lock due to several incorrectly entered passwords, and removal of such locks, since this is classified as critical.

The logic in the ABAP code for the detection engine

1. Tallies all logon failures.
2. Ignores all logon attempts with non-existing user IDs if FPR is selected.
3. Anonymizes the user name, terminal ID, role, and description if this is chosen.

3.1.7 Detecting Authorization Failures

Authorization failures are registered as incidents in the security audit log every time a user is attempting an operation not included in his or her access rights. Typically, it involves trying to run a transaction code from the transaction code field, or from the SAP menu.

The detection engine will

1. Tally all authorization failures.
2. Dismiss all non-defined tcodes as keying errors if FPR is selected.
3. Anonymizes the user name, terminal ID, role, and description if this is chosen.

3.1.8 Detecting Download Activity

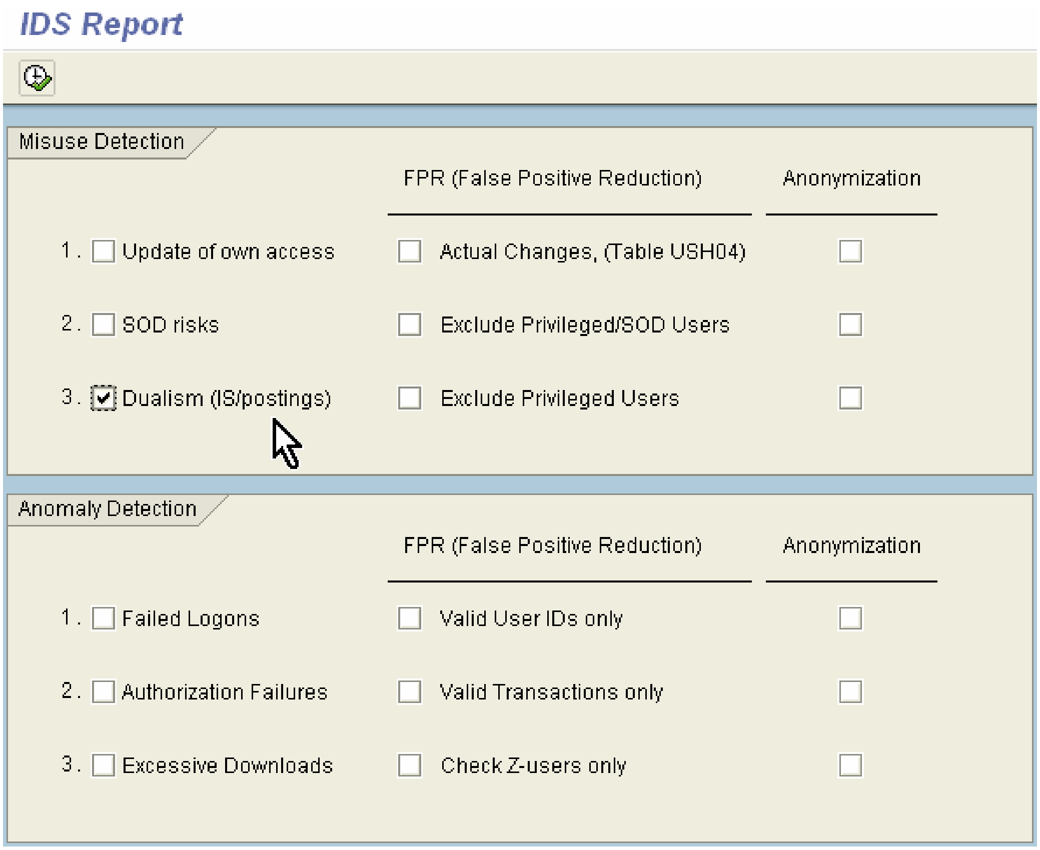
Finally, the last characteristic for anomaly detection is to investigate the download activity in the system. We want to check if someone is downloading and gathering large amounts of documents with information which can be misused by competitors or vendors. Practically all reports and documents in SAP are available for downloading, provided that the user has necessary authorizations.

The detection engine

1. Counts the number of users who downloads information from the system
2. Extracts the enterprise users only
3. Anonymizes the users' names, terminal IDs, roles, and description field, if chosen.

3.1.9 Running the Detection Engine

The in-house developed ABAP program functioning as a detection engine & decision engine is called ZRBS012A in SAP P01. The core logic for the ABAP code is in appendix 4, and a screen shot of the user interface to the program is shown in figure 3. The button with a clock and a green check mark is the "Execute button" in SAP. With this detection engine, you can chose what to check for the time period you specified in the log collector.



IDS Report

Misuse Detection

	FPR (False Positive Reduction)	Anonymization
1. □ Update of own access	□ Actual Changes, (Table USH04)	□
2. □ SOD risks	□ Exclude Privileged/SOD Users	□
3. ☑ Dualism (IS/postings)	□ Exclude Privileged Users	□

Anomaly Detection

	FPR (False Positive Reduction)	Anonymization
1. □ Failed Logons	□ Valid User IDs only	□
2. □ Authorization Failures	□ Valid Transactions only	□
3. □ Excessive Downloads	□ Check Z-users only	□

Figure 5

3.2 Results and Discussion

3.2.1 Statistical Analysis

Statistical techniques for analyzing this data do not apply in the traditional sense as it is used in medical and sociology research.[15] The "sample" we use in this case is the entire "population", or rather a comprehensive set of events and incidents, as this is one performance criteria for an IDS.[5] Our main focus is not to make any type predictions about future trends or investigate possible "treatments". We make suggestions as examples for what to do as corrective actions in some cases, however, these suggestions are simply added with the intent to further understanding for the project. The main focus is to evaluate the performance and accuracy of a self developed IDS when we try this system with reduction of false positives and anonymization. From the IDS reports one must be able to consider several states of system "health" issues, in the security risk area. So, we are checking the efficiency of "diagnosing" problem areas, or discovering incidents as they relate to information security risks.

The logging period used for registering the results, started January 10 and ended on June 30 in year 2007. Each month is divided into three parts, so the first third is from the first of the month until the tenth, the second third is from the eleventh until the twentieth, and the last third is the remaining days of the month, the 21st until the 28th-31st. Thus, we have, each time, a log table that is small enough to process with acceptable response time. Each characteristic was checked three times in the IDS detection engine, once in standard mode, once with one type of reduction of false positives, FPR, and once with FPR and anonymization.

We want to research the performance of the system when running in standard mode compared with running with FPR and anonymization. Running the system with just FPR was done to confirm accuracy of the log and to research the anonymized results. Statistical techniques would apply to some of the suggestions we present in the chapter five, for instance, when introducing thresholds and alerts in the IDS.

3.2.2 Anonymization

As expected, there is a one-to-one relationship in the false positive reduction reports produced with and without anonymization. The anonymization itself is performed as a last step in the reporting facility, and this approach was decided quite early in the project. The initial thought was to run the data through the detection engine with anonymization in the input data, however, this approach was quickly discarded as we found no practical need to run the detection engine this way. The only "gain" from running the system with anonymization in the input data is that the code in the detection engine gets more complicated and more prone to errors.

The results check for the anonymized reports compared with the standard false positive reduction reports, when examining the log directly, and when checking the PC spreadsheets.

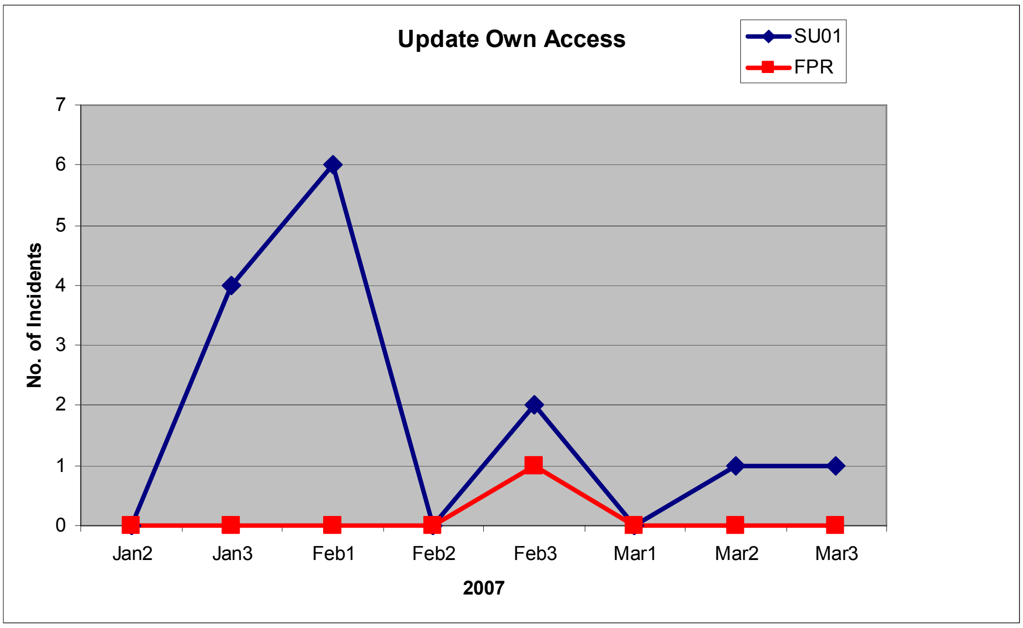
3.2.3 Misuse Detection, Maintaining Own Access

The reporting used for our work for this IDS characteristic spans from January to March in year 2007. During the remaining period, from April to June, there were no violations in the area of administration of own access rights. This result was not surprising to us, since all user administrators were blocked from access to change their own accounts in the month of March and a high focus on this security breach was broadcasted in the IS department. The access barrier was accomplished by creating a special authorization user group for each administrator, and taking those into account in the user administration access roles. Each authorization administrator got his or her own access role fitted for the authorization user group setting.

The figures for our findings are listed in table 3, and the single hit during the third period in February was found to be an update performed on the emergency logon, which already had access to all functions on the system. The corresponding chart for this table is found in graph 1. As we can deduce from the logging and the reports produced from the log, there are few incidents. Still, such a security breach is serious when it happens. In SOX audits there must be no incidents at all for the control objective[23], so one incident is one too many. The magnitude of this problem, however, is at such an infrequent level that further handling of the numbers in the incident reporting is not as interesting for this characteristic as for the others. We have registered only one actual incident.

	Jan2	Jan3	Feb1	Feb2	Feb3	Mar1	Mar2	Mar3
	0	4	6	0	2	0	1	1
FPR	0	0	0	0	1	0	0	0
FPR, ANON	0	0	0	0	1	0	0	0

Table 3



Graph 1

Performance wise, the result is accurate based on inspection in the user change document records in SAP's SUIM (User Information System) and based on manual inspection of the Security Audit Log directly.

3.2.4 Misuse Detection, SOD

The Segregation of Duties characteristic of the IDS detection engine was the most complicated and interesting part of the project. The response time for checking this type of misuse was far longer than other characteristics of the detection engine. Creating the basis for SOD checking is a long process with a considerable learning curve. Much of the work involved also revolves around trying to match correct formats for the different tools to be used for creating our prototype.

We first had to extract the transaction codes in use from the system. We then uploaded our list of transaction codes for SOD checking in the Compliance Calibrator by Virsa. For us, the easiest way to accomplish this was to first place every tcode into one large access role, and then check this role for transaction code SOD risks. The result from the Compliance Calibrator was downloaded to PC format and reworked in a PC spreadsheet program. When we had a list of tcodes in two and three transaction code combination with SOD risk, we uploaded this file to SAP by using our own program called Z_SOD_UPLOAD3, which is listed in appendix 2.

Due to its complexity and relevance to internal misuse, we checked this characteristic both for number of risks in total and number of users creating these risks. Conclusions regarding the ratio between number of users and number of incidents should be considered. For example, if a few users commit many SOD risks, you could raise the question whether the organization has adapted proper segregation of duties controls.

	Jan2	Jan3	Feb1	Feb2	Feb3	Mar1	Mar2	Mar3	Apr1
Total incidents	4388	6718	6102	5665	6446	5976	5936	6600	3109
SOD user	79	83	78	76	69	77	73	85	53
FPR, Total	1562	1645	766	1223	875	1111	735	893	496
FPR, user	26	28	23	25	24	23	18	23	14
ANON, Total	1562	1645	766	1223	875	1111	735	893	496
ANON, user	26	28	23	25	24	23	18	23	14

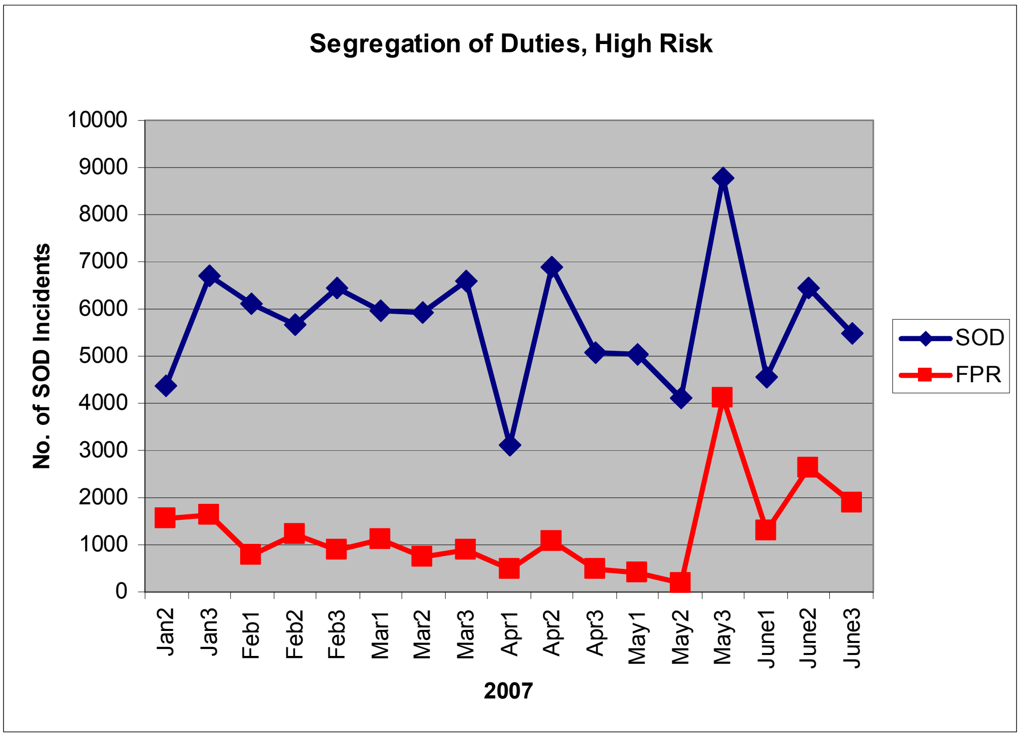
Table 4

	Apr2	Apr3	May1	May2	May3	Jun1	Jun2	Jun3
Total incidents	6878	5080	5019	4103	8779	4567	6444	5467
SOD user	75	69	75	62	85	72	75	75
FPR, Total	1064	492	420	175	4097	1291	2642	1882
FPR, user	21	17	18	13	24	18	20	24
ANON, Total	1064	492	420	175	4097	1291	2642	1882
ANON, user	21	17	18	13	24	18	20	24

Table 5

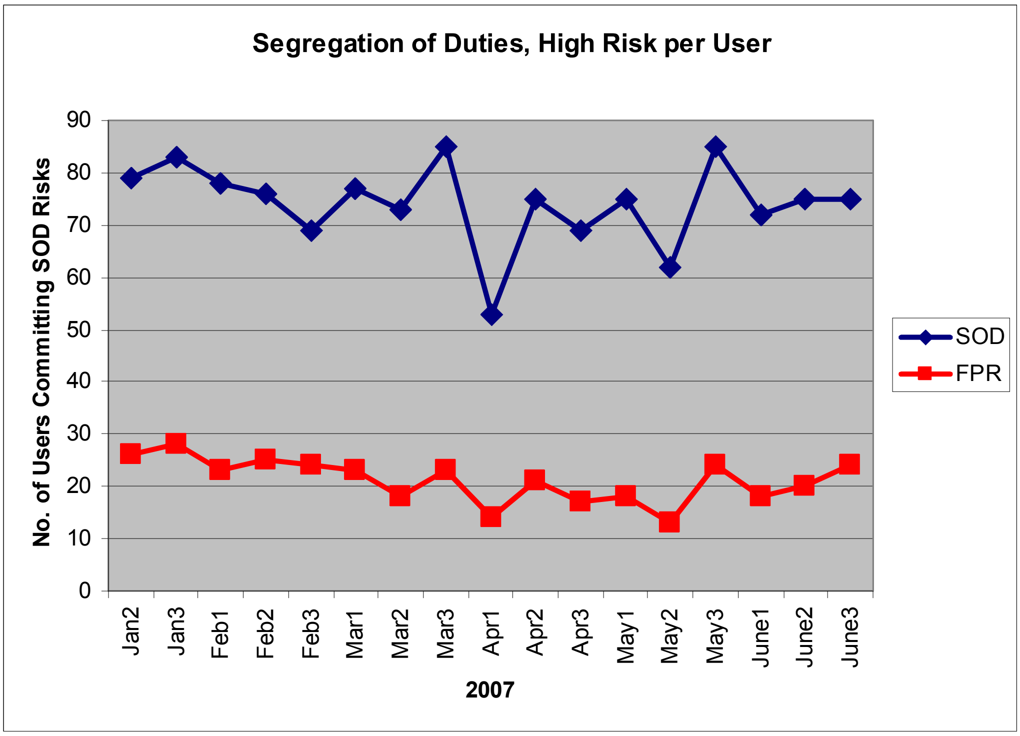
The total number of incidents and the number of users producing these incidents are listed in table four and five. The number of high SOD risks will vary from SAP system to SAP system as the different companies adapt different work routines for their employees. If the tendency is to assign a few “SAP gurus” or so called experts in the organization who take care of most of the postings, then the system is likely to produce more SOD high risk incidents.

As an example, relevant questions for the organization is whether these few experts, over time, will gain undue decision making power and will filter information to their own benefit. These questions could be considered aside from the direct risk of fraud and theft.



Graph 2

Graph two and graph three represents the SOD risks in total and per user, respectively. We can see some resemblance in the shape, with a spike for the third measurement in May and a low for the first measurement in April for example. The reduction of false positives is 79% for the total and 72% when measured per user.



Graph 3

The reduction when applying false positives is significant for the SOD high risk characteristic, but what does it mean? We define high SOD risk as an incident if it is done without management approval. If it is done with management approval we classify it as an event. The IDS concentrates the report to cover the incidents, and omits the events when it runs in FPR mode.

We define an authorization user group called SOD and place the users approved for SOD high risk transaction combinations into this group. Thus we can distinguish this special user group from the rest of the user community.

Corrective actions aims to reduce the number of incidents to zero. The corrective actions for the SOD report can be categorized in two groups. One group is the attempted SOD high risk transactions that resulted in authorization error, and the other is a SOD high risk transactions that ran without authorization error but was done by users who did this without approval.

Since the user community is dynamic in their access rights and authorizations, it is not likely that the number of incidents ever will reach zero as far as SOD high risks in this IDS. For instance, a user could have his approval revoked at the end of a measurement period, but still be an approved user at the beginning of the measurement period when he or she ran transactions with SOD high risk.

3.2.5 Misuse Detection, Dualism

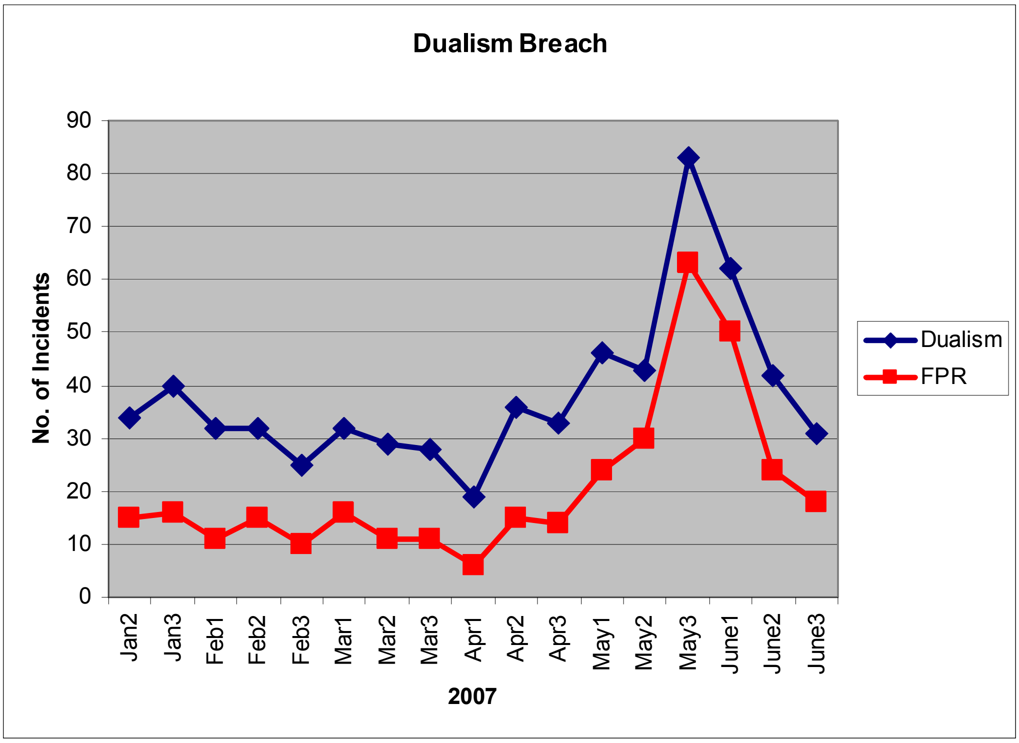
The amount of dualism breaches in the system is listed in table six and 7. An example of a dualism report screenshot is in appendix 12.

	Jan2	Jan3	Feb1	Feb2	Feb3	Mar1	Mar2	Mar3	Apr1
	34	40	32	32	25	32	29	28	19
FPR	15	16	11	15	10	16	11	11	6
FPR, ANON	15	16	11	15	10	16	11	11	6

Table 6

	Apr2	Apr3	May1	May2	May3	Jun1	Jun2	Jun3
	36	33	46	43	83	62	42	31
FPR	15	14	24	30	63	50	24	18
FPR, ANON	15	14	24	30	63	50	24	18

Table 7



Graph 4

Average false positive reduction rate is 50%, which is considerable.

Again, the false positive reduction does not remove actual dualism, but it concentrates to the incidents that occurred without approval. The system's label for approval to commit dualism in transaction codes, is that the user is member of the authorization group for "privileged users".

Corrective action to follow each investigation of dualism should be divided in two. The first action is taken with incidents that occurred with authorization failure. The proposed action is to inform the end user of access request procedures, and why this information is supplied to the user. Thus, the end user becomes aware of the fact that authorization errors are followed up, logged, and monitored. The second corrective action revolves around approval. The incident occurred without authorization error, which means that the end user is not member of the “privileged users” authorization user group.

Dualism breach is a not a wide spread phenomena for this system, and as such, this problem should be maintainable to the level where the IDS will run without any reported incidents when running with false positive reduction.

3.2.6 Anomaly Detection, Login Failures

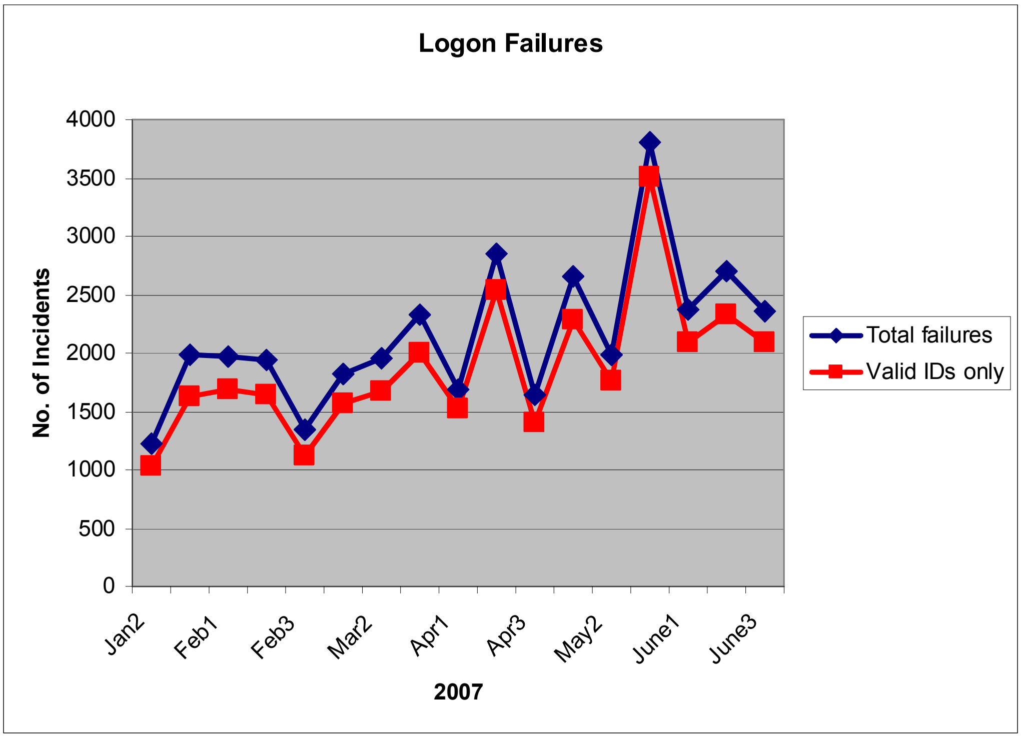
In a system where many users have and off-shore rotation schedule of two weeks on the job and two weeks off, we expect many logon failures. The logon failures are listed in tables eight and nine.

	Jan2	Jan3	Feb1	Feb2	Feb3	Mar1	Mar2	Mar3	Apr1
	1229	1992	1965	1938	1336	1817	1948	2333	1693
FPR	1035	1626	1686	1649	1123	1572	1679	2000	1520
FPR, ANON	1035	1626	1686	1649	1123	1572	1679	2000	1520

Table 8

	Apr2	Apr3	May1	May2	May3	Jun1	Jun2	Jun3
	2854	1646	2659	1978	3807	2376	2699	2365
FPR	2539	1396	2284	1758	3509	2096	2331	2097
FPR, ANON	2539	1396	2284	1758	3509	2096	2331	2097

Table 9



Graph 5

When analyzing the results from the login failures, we see that there is an average of 13% reduction of incidents if we run the system with false positive reduction. The low is an 8% difference in the last period for the month of May. This was also the peak for overall activity in the system due to the off shore revision stop. The biggest difference was 18% in the last period for January. In any case, dismissing the invalid user IDs as typing mistakes does not seem to have a significant impact on the amount of incidences.

3.2.7 Anomaly Detection, Authorization Failures

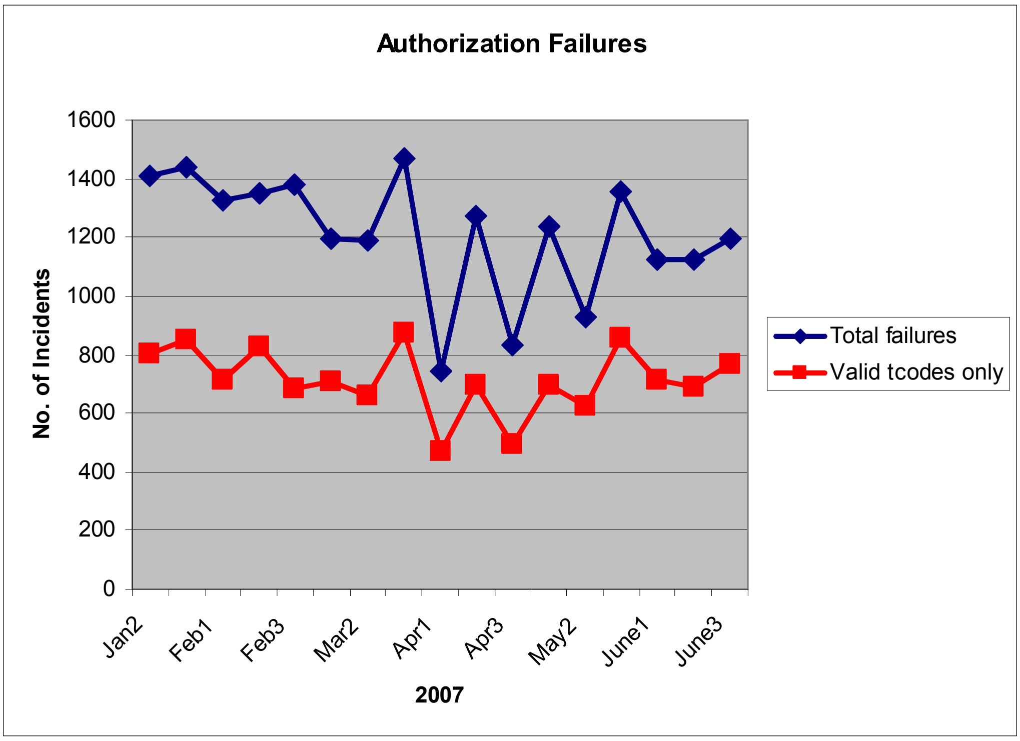
The amount of authorization failures are listed in tables ten and eleven. A example screenshot of the authorization errors report is in appendix 11.

	Jan2	Jan3	Feb1	Feb2	Feb3	Mar1	Mar2	Mar3	Apr1
	1408	1441	1326	1352	1379	1198	1192	1469	741
FPR	805	849	713	827	684	707	659	876	470
FPR, ANON	805	849	713	827	684	707	659	876	470

Table 10

	Apr2	Apr3	May1	May2	May3	June1	June2	June3
	1271	835	1238	925	1359	1122	1122	1194
FPR	696	494	693	625	854	715	688	768
FPR, ANON	696	494	693	625	854	715	688	768

Table 11



Graph 6

The average reduction in number of incidences is 41% when running the system with false positive reduction, which is somewhat useful. It looks as though there is quite a bit of misspellings of transaction codes. What share is due to the users' memory and what is just misspellings is unknown. With a high of 51% and a low of 32%, the variation is not more than 10%.

Repeat offenders should receive a notification of the access request procedure at the very least.

3.2.8 Anomaly Detection, Download

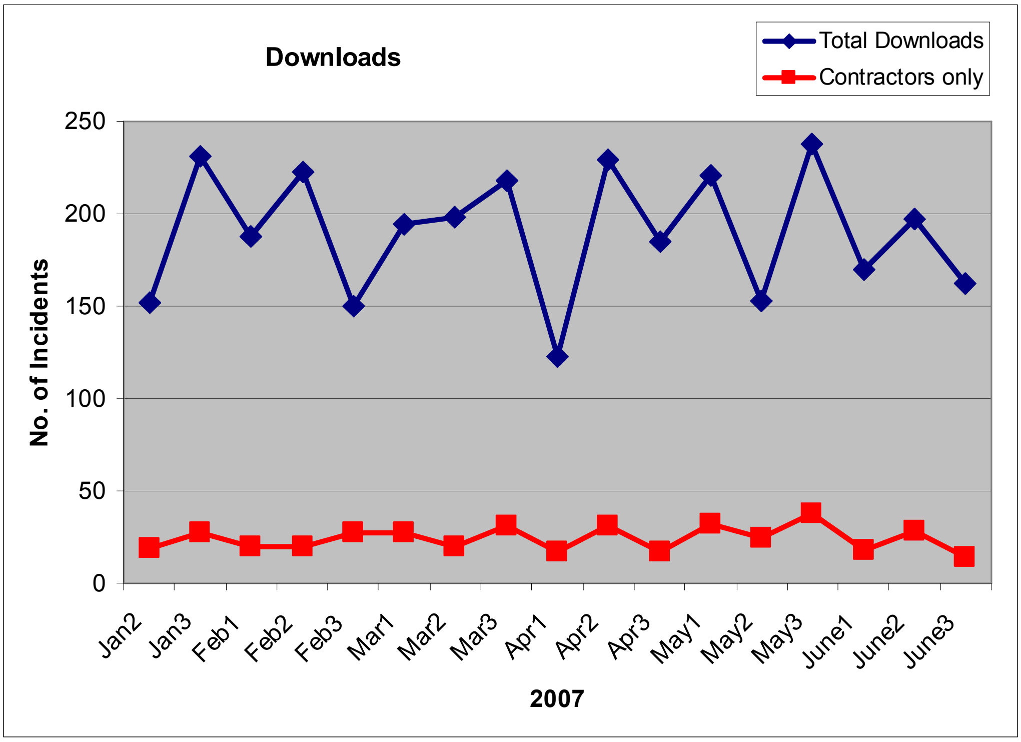
Finally, the download activity is listed in tables twelve and thirteen.

	Jan2	Jan3	Feb1	Feb2	Feb3	Mar1	Mar2	Mar3	Apr1
	152	231	188	223	150	194	198	218	123
FPR	19	27	20	20	27	27	20	31	17
FPR, ANON	19	27	20	20	27	27	20	31	17

Table 12

	Apr2	Apr3	May1	May2	May3	June1	June2	June3
	229	185	221	153	238	170	197	162
FPR	31	17	32	25	38	18	28	14
FPR, ANON	31	17	32	25	38	18	28	14

Table 13



Graph 7

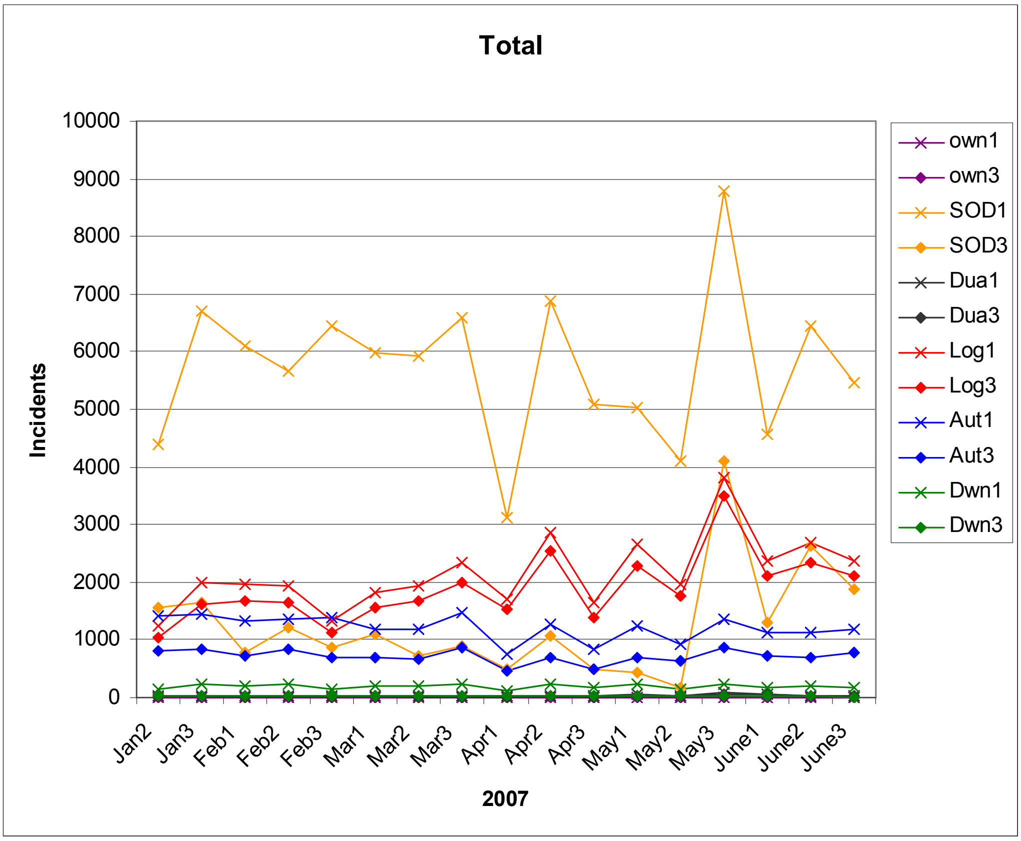
The false positive reduction rate is 87% for incidents involving downloads. With a high of 91% and a low of 82% the reduction in incidents is rather uniform for this time period.

Z-users primarily off-site contractors, A-users primarily on-sight hired personnel

4 Conclusions

The total activity is summarized in graph 8, and the pattern for level of activity is visible. The spike in the third period of May happened because of a revision stop for the oil rigs. During a revision stop, the oil production is cut off and this is a rather hectic period for the company. To contain cost, the window for a shut down period must be as narrow as possible, and consequently, the activity is high.

The low point for graph 8 is the first time period of April. Easter week happened on the first nine days of April in 2007, and traditionally, Norwegians go on vacation during the week preceding the Easter holidays. The activity level on the SAP system during Easter week is reflected in the IDS reporting.



Graph 8

As mentioned in the introduction of this paper, the performance of an IDS is based on how comprehensive, timely, comprehensible and accurate it is. In this case, how timely the system is, generally is left up to the operator. The reports can be run at leisure or at a predefined schedule. The reports should be generated at the end date for each log collecting period, to minimize the error rate that stems from dynamics in the building blocks. This is discussed in the 1.2 Assumptions chapter. How timely this IDS is when running in standard mode compared with running it with reduction of false positives and anonymization makes no difference, as long as we base the reports on the same time periods.

4.1 Misuse Detection

The misuse detection reports are readily comprehensible to security auditors since they deal with these issues in depth. Others will likely have a learning curve in understanding what the serious issues are behind terms like dualism and SOD risks.

4.1.1 Maintaining Own Access

Self maintenance of access rights in the system is easy to understand, as any network user would soon agree that a system with financial transactions cannot be based on a "help yourself to authorizations" approach. This activity is considered such a serious transgression that the report itself should contain few incidents. Since the vast majority of updates in the user's own user master record normally is changes to his personal data, such as phone number and address information, the report will typically be drastically reduced in number of incidences when it only covers actual authorization changes. The comprehensibility of the reports is therefore greater when eliminating the "noise" from the personal information updates and using the false positive reduced reports.

The same holds for accuracy. If a user changes his address or phone number, printer settings or parameter id's in the user master record it should merely be considered a event and not an incident. Therefore, the false positive reduction report is more accurate, since it checks for actual changes in the user master record. This part of the IDS is actually 100% accurate for the research we performed. The report was checked against the change documents in the User Information System (SUIM), and we found no discrepancies.

4.1.2 High SOD Risks

High amounts of incidences for high SOD risks may indicate that the organization of the work duties should be reconsidered. For this project, we reduced the incidence report for high SOD risk to those without approval. One cannot categorically say that a high number of high SOD risks is fine as long as the users have approval. Likewise, there is no fixed levels for risk in this area since the whole business process has to be considered before any risk assessments. The question of how accurate is one way of executing the report versus another, is more complex for segregation of duties risks. While the false positive reduction report drastically reduces the amount of "incidences" and thus is more comprehensible, the total number of high SOD risks cannot be disregarded.

4.1.3 Breach of Dualism Principle

Achieving complete segregation of user into two groups the one group only performs IS operations related transactions and the other only does postings and business transactions is practically very difficult to achieve. It is possible to reduce the number of users with this type of authorization to a low number and monitor the activities for these privileged users. The reduction of false positives did in this case reduce the number of incidences on average by 50%. By offering some focus on corrective action for these real incidences, that is, the users who commits or attempts to commit dualism breach without approval, it is likely that the number of incidents become less than one half dusin per reporting period. Breach of dualism does not imply that the user is attempting electronic theft, this is more likely in the SOD risk category.

4.2 Anomaly Detection

Any anomaly report in this IDS is likely to have a low learning curve. The issues are quite straightforward and it is easy to explain what the potential incidents mean. Again, running the system with false positive reduction and anonymization versus running it without, only makes minor changes in how comprehensible they are. If a smaller report means improving comprehensibility then the shorter false positive reduction reports are better. If anonymization makes the reports less comprehensible, then running the system without this feature is obviously easier to understand.

4.2.1 Logon Failures

Indications of intrusive actions at logon includes patterns of repeated and failed attempts with user ID and password. A manual inspection of a trained computer security officer the log is in this case likely to be more effective than simple, automated analysis. The log in our IDS will, however, show how many incidences of logon failures happen in the time period. The false positive reduction feature for this characteristic, was to discard invalid user IDs. Should an unusually large gap between the two modes for reporting happen, then this could indicate break in attempts. The difference in comprehensiveness between the two reports is not significant. The reports are accurate as we check them against SAP's User Information System records, which has a facility for checking failed logons.

The conclusion for the logon failure reports is that there is an advantage in comparing the two reports to verify that they do not differ significantly in amount of incidences. A larger than average amount of logon failures is obviously a warning sign for intrusive attacks.

4.2.2 Authorization Failures

The IDS makes no discrimination in what transaction code is attempted. Some transaction codes are more significant than other in what impact they have in the system, or how they affect the financial accounting. This reporting facility must therefore be viewed as a first step towards analyzing the authorization failures. It is not likely that an attacker will attempt to run transaction codes by entering many alphanumeric combinations. The result from this activity would be too incidental. An attacker is more likely to know exactly which transactions to use for the malicious activity. We can thus conclude that the report with false positive reduction gives a more accurate and true picture of the problems with failed authorizations. The false positive reduction report is not 100% accurate, since the user will sometimes misspell a transaction code or have an error in recollection and by chance hit a valid transaction code for which he or she does not have access. As we see it, it is not possible for an IDS to validate these incidents as false positives.

4.2.3 Excessive Download Activity

According to the assessment made by the management, only the contractors' download activity is significant enough to be labeled as incidents. The same concern holds for download activity as for logon failures. A larger than average amount has to be investigated. Ideally the IDS should discover a large amount of downloads from one single user regardless of whether he or she is a contractor or a hired employee. This is suggested in the last chapter for further work. For this IDS the rule is to report the contractors only and disregard the other downloads. Since the reduction is around 90%, the comprehensiveness of the report is affected. The amount of incidents by contractors could typically fit on one screen page for this report.

5 Suggestions for Further Work

A clear weakness in the IDS is the download activity and dualism reporting. These characteristics should be treated like the SOD risk characteristic where the total number of incidents and not just number of users committing the incidents is reported. From our viewpoint, this is the most obvious piece to suggest for an improvement.

A nice feature for the IDS would be if it could check if user administrator has created user account without approval, and actually used this account. This should be possible to some extent by checking the terminal ID and cross check uploaded approval records.

Automation in the SOD Risk Signature Base is preferable, since this would eliminate the need to upload a new SOD risk table every time a new transaction code with SOD risk is implemented in the access roles.

Further reductions of false positives is possible – it is only limited by the developers' own imaginations.

Alert mechanisms are often a vital part of IDS. Alerts for this system is not possible on-the-fly, since it is a reporting system not an agent. Some alert flags could be indoctrinated in the reports, however, so the operator of the IDS is notified of areas that requires special corrective actions. Thresholds particularly for downloads, distribution of SAP_ALL profile to user master record, logon failures, authorization errors and so on, could all trigger an alert.

Blacklists (hot lists) and whitelists are [6] used in some IDS and potential candidates for such registry are for instance user ID lists, user groups, terminal IDs, and critical transactions.

6 References

Ref Nr Authors, book/article, web address, or other detail.

[1] Dennis C. Brewer: Security Controls for Sarbanes-Oxley Section 404 IT Compliance: Authorization, Authentication, and Access, 2005
ISBN-13: 978-0-7645-9838-8

[2] Jon Petter Syvertsen: Insider Threat, Gjøvik University College, 2007

[3] ISO. ISO Standard ISO/IEC 17799: Code of practice for information security management. 2001.

[4] George Mohay, Alison Anderson, Byron Collie, and Olivier de Vel: Computer and Intrusion Forensics (Artech House Computer Security Series), ISBN 1-58053-369-8

[5] Rebecca Gurley Bace: Intrusion Detection, 2000, ISBN 1-57870-185-6

[6] NIST, (National Intritute of Standards and Technology), Karen Scarfone and Peter Mell: Guide to Intrusion Detection and Prevention Systems (IDPS), Special Publication 800-94, February 2007

[7] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman: Role-based access control models. IEEE Computer, 29(2):38{47, February 1996.

[8] Ravi Sandhu: Role Activation Hierarchies. Proceedings of 3rd ACM Workshop on Role-Based Access Control, Fairfax, Virginia, October 22-23, 1998

[9] Alan Rickayzen, Jocelyn Dart, Carsten Brennecke, and Markus Schneider: Practical Workflow for SAP - Effective Business Processes using SAP's WebFlow Engine

[10] Alan Calder, Steve Watkins: IT governance: a manager's guide to data security and BS 7799/ISO 17799, 2005

[11] Stanford Encyclopedia of Philosophy,
http://plato.stanford.edu/entries/dualism/

[12] Dieter Gollmann: Computer Security, ISBN 0-471-97844-2

[13] Matt Bishop: Computer Security Art and Science, ISBN -201-44099-7

[14] Donn B. Parker: Ethical Conflicts in Computer Science and Technology
(no ISBN)

[15] Leedy, Paul D: Practical Research, ISBN 0-13-124720-4

[16] Witcha Chimphlee, Mohd Noor Md Sap, Abdul Hanan Abdullah, Siriporn Chimphlee, Surat Srinoy: To Identify Suspicious Activity In Anomaly Detection Based On Soft Computing. Proceedings of the 24th IASTED International Multi-Conference “Artificial Intelligence and Applications”, February 13-16, 2006 Innsbruck, Austria.

[17] Peter Best, School Of Accounting, Queensland University of Technology, Brisbane, Australia: SAP R/3 Audit Trail Analysis, SAPPhire 2000 Brisbane Conference, 4th Annual SAP Asia Pacific

[18] Mario Linkies and Frank Off: SAP Security and Authorizations, SAP Press / Galileo Press

[19] SAP Security Guide, SAP Press

[20] IBM Consulting Services: SAP Authorization System, SAP Press

[21] Authorizations Made Easy, SAP Press

[22] Users and Roles (BC-SEC-USR), SAP Press

[23] Jill Gilbert Welytok: Sarbanes-Oxley for dummies,
ISBN-13:978-0-471-76846-3

[24] Richard Heady, George Luger, Arthur Maccabe, and Mark Servilla. The architecture of a network level intrusion detection system. Technical Report CS90-20, Department of Computer Science, University of New Mexico, August 1990

[25] SAP Note 747615 - Tool for converting files from one code page to another,
service.sap.com (https://websmp102.sap-ag.de/support)

[26] SAP Note 752859 – SAPICONV: A tool for converting the encoding of files,
service.sap.com (https://websmp102.sap-ag.de/support)

[27] SAP Note 23611 - Collective Note: Security in SAP Products,
service.sap.com (https://websmp102.sap-ag.de/support)

[28] SAP Note 875835 - SecAudit: Analysis finds no audit events,
service.sap.com (https://websmp102.sap-ag.de/support)

[29] SAP Note 30724 - Data protection and security in SAP Systems,
service.sap.com (https://websmp102.sap-ag.de/support)

[30] SAP Note 139418 - Logging user actions,
service.sap.com (https://websmp102.sap-ag.de/support)

[31] SAP Note 134542 - Security audit: Authorization to delete audit logs,
service.sap.com (https://websmp102.sap-ag.de/support)

[32] SAP Note 164665 - Security Audit: Audit files are created,
service.sap.com (https://websmp102.sap-ag.de/support)

[33] SAP Note 173743 - Security Audit: Changing parameters does not perform,
service.sap.com (https://websmp102.sap-ag.de/support)

[34] SAP Note 181173 - SM18: Error with minimum age,
service.sap.com (https://websmp102.sap-ag.de/support)

[35] SAP Note 198646 - Security Audit: SM18 collective note,
service.sap.com (https://websmp102.sap-ag.de/support)

[36] SAP Note 429343 - SecAudit: Performance-aspects,
service.sap.com (https://websmp102.sap-ag.de/support)

[37] SAP Note 443460 - Formatted Reporting: Viewing, service.sap.com (https://websmp102.sap-ag.de/support)

[38] SAP Note 539404 - FAQ: Answers to questions about the Security Audit Log, service.sap.com (https://websmp102.sap-ag.de/support)

[39] SAP Note 849167 - KM security audit logging - performance impact, service.sap.com (https://websmp102.sap-ag.de/support)

[40] SAP Note 115224 – SQL Audit, service.sap.com (https://websmp102.sap-ag.de/support)

[41] SAP Note 986996 – Best Practice for SAP CC Rules and Risks, service.sap.com (https://websmp102.sap-ag.de/support)

[42] SAP Note 987032 – Combined Analysis Functionality in SAP CC, service.sap.com (https://websmp102.sap-ag.de/support)

[43] SAP Note 986420 – How to configure organization rules in SAP CC 5.1, service.sap.com (https://websmp102.sap-ag.de/support)

[44] SAP Note 987031 - Instructions on how to create Organizational Rules, service.sap.com (https://websmp102.sap-ag.de/support)

[45] SAP Note 1026576 - AND OR NOT Logic in Compliance Calibrator, service.sap.com (https://websmp102.sap-ag.de/support)

[46] SAP Note 1006083 - Virsa Access Controls Suite- STMS to SAINT Migration,
service.sap.com (https://websmp102.sap-ag.de/support)

[47] SAP Note 549485 - FAQ, security, authorizations, authorization buffer, service.sap.com (https://websmp102.sap-ag.de/support)

[48] SAP Security Audit Log, Release 4.6c, SAP Press
http://help.sap.com/printdocu/core/Print46c/en/data/pdf/BCCSTADM/BCCSTSAL.pdf
(last visited September 18, 2007)

1. Appendix – Code for Anonymization Table

ABAP program Z_SELECT_SUB, a suggestion for how to fill a simper substitution table used for anonymization of security audit log.

```
************************************************************************
*                                                                      *
   REPORT Z_SELECT_SUB Line-Size 255.
*                                                                      *
************************************************************************
*                                                                      *
*  Fill a substitution table with exhaustive random characters in same *
*  character space, to be used for anonymization for internal          *
*  application based IDS.                                              *
*                                                                      *
*    date     init  description                            transp req  *
*  --------   ----  -------------------------------------  ----------  *
[redacted]
*                                                                      *
*                                                                      *
************************************************************************
*                                                                      *
* Declaration section                                                  *
*                                                                      *
   tables  ZANOSUB.

   data: begin of itab occurs 0,
            p_clear like zanosub-clear,
            p_subst like zanosub-subst,
         end of itab.

   data: i like zanosub-clear.
*                                                                      *
************************************************************************
*                                                                      *
* Input section                                                        *
*                                                                      *
   refresh itab.
   selection-screen begin of block b1 with frame title text-001.
*                                                                      *
   parameters: p_clear like zanosub-clear,
            p_subst like zanosub-subst.
*                                                                      *
   selection-screen end of block b1.
*                                                                      *
   translate p_clear to lower case.
   translate p_subst to upper case.

   itab-p_clear = p_clear.
   itab-p_subst = p_subst.

   append itab.
   write: / ' last entered clear text character: ', itab-p_clear, /,
            ' last enetred substistuted character: ', itab-p_subst.
   i = itab-p_clear.
*                                                                      *
* Update substitution table ZANOSUB                                    *
*                                                                      *
   loop at itab.
      insert into zanosub values itab.
   endloop.
   write /.
*                                                                      *
* Screen output for verification                                       *
*                                                                      *
   clear zanosub.
   select * from zanosub order by clear.
      write: zanosub-clear.
   endselect.
   write /.
*                                                                      *
   clear zanosub.
   select * from zanosub order by clear.
      write: zanosub-subst.
   endselect.
   write /.
   write /.
*                                                                      *
   clear zanosub.
   select * from zanosub order by clear.
      write: / zanosub-clear,'=', zanosub-subst.
   endselect.
*                                                                      *
   clear zanosub.
   select * from zanosub into itab where clear = ' '
                                      or clear = 'a'
                                      or clear = 'b'.
   endselect.
         write: / 'count = ', SY-DBCNT.

*                                                                      *
* End of program Z_SELECT_SUB                                          *
*                                                                      *
************************************************************************
```

2. Appendix – Code for Uploading Tcodes with SOD Risks

ABAP Program Z_SOD_UPLOAD3, for uploading a PC spreadsheet file to SAP R/3. The PC file contains a list of tcodes with high SOD risk in two comination and three combination formats.

```
*************************************************************************
*                                                                       *
   REPORT Z_SOD_UPLOAD3 no standard page heading.
*                                                                       *
*************************************************************************
*                                                                       *
* Upload SOD tcodes documented in an excel file and insert into table   *
* ZSODTCODES using the function module: ALSM_EXCEL_TO_INTERNAL_TABLE    *
*                                                                       *
*    date    init  description                         transp req   *
*  --------  ----  --------------------------------------- ----------   *
[redacted]
*                                                                       *
*************************************************************************
*                                                                       *
* Declaration section                                                   *
*                                                                       *
   tables  ZSODTCODES3.

   data: itab like alsmex_tabline occurs 0 with header line.

   TYPES:
      begin of t_record,
         posttcode like zsodtcodes3-tcode,
         SODtcode1 like zsodtcodes3-SODtcode1,
         SODtcode2 like zsodtcodes3-SODtcode2,
         SODtext   like zsodtcodes3-SODtext,
      end of t_record.

   DATA: it_record type standard table of t_record initial size 0,
         wa_record type t_record.

   DATA: gd_currentrow type i.

   PARAMETER p_infile like rlgrap-filename.
*                                                                       *
*************************************************************************
*                                                                       *
* Input section                                                         *
*                                                                       *
   call function 'ALSM_EXCEL_TO_INTERNAL_TABLE'
       exporting
         filename    = p_infile
         i_begin_col = '1'
         i_begin_row = '1'
         i_end_col   = '4'
         i_end_row   = '2075'  "Adjust this for number of rows
       tables
         intern      = itab
       exceptions
         inconsistent_parameters = 1
         upload_ole  = 2
         others      = 3.
*                                                                       *
* Initialize the internal table                                         *
*                                                                       *
   sort itab by row col.
     read table itab index 1.
     gd_currentrow = itab-row.

     loop at itab.
       if itab-row ne gd_currentrow.
         append wa_record to it_record.
         clear wa_record.
         gd_currentrow = itab-row.
       endif.

       case itab-col.

         when '0001'.                           "Posting tcode
           wa_record-posttcode = itab-value.

         when '0002'.                           "First SOD tcode
           wa_record-SODtcode1 = itab-value.

         when '0003'.                           "Second SOD tcode
           wa_record-SODtcode2 = itab-value.

         when '0004'.                           "Risk text
           wa_record-SODtext   = itab-value.

       endcase.
     endloop.
   append wa_record to it_record.
*                                                                      *
* The internal table IT_RECORD holds the data from the Excel file.     *
*                                                                      *
************************************************************************


************************************************************************
*                                                                      *
* Register in table ZSODTCODES and display data                        *
*                                                                      *
************************************************************************
*                                                                      *
  loop at it_record into wa_record.
    write:/     sy-vline,
*                                                                      *
* Screen output for manual verification                                *
*                                                                      *
           (12) wa_record-posttcode, sy-vline,
           (12) wa_record-SODtcode1, sy-vline,
           (12) wa_record-SODtcode2, sy-vline,
           (120) wa_record-SODtext, sy-vline.
*                                                                      *
* Update table ZSODTCODES                                              *
*                                                                      *
      insert into zsodtcodes3 values wa_record.
  endloop.
*                                                                      *
* End of program Z_SOD_UPLOAD                                          *
*                                                                      *
************************************************************************
```

3. Appendix - Code for Collecting the Security Audit Log Files

ABAP Program ZRSAU_SELECT_EVENTS collects all audit log files from application servers within specified time span. The time span is obtained by operator input.

```
.
.
.
START-OF-SELECTION.

  IF NOT e_zsalog IS INITIAL.
    IF NOT s_alv IS INITIAL.
      CALL FUNCTION 'POPUP_WITH_2_BUTTONS_TO_CHOOSE'
      EXPORTING
       diagnosetext1 = 'You have also chosen : '
       diagnosetext2 = ' Display ALV report '
 diagnosetext3 = 'If you choose Empty Table ZSALOG-this will be ignored'
        textline1     = ' '
        textline2     = ' '
        textline3     = 'Choose : '
        text_option1  = 'Empty ZSALOG'
        text_option2  = 'Exit - reselect'
        titel         = 'Empty table ZSALOG chosen'
      IMPORTING
        answer        = popup_answr.

      IF popup_answr <> '1'.
      ELSE.
        PERFORM delete_zsalog.
        EXIT.
      ENDIF.
    ELSE.
      PERFORM delete_zsalog.
      EXIT.
    ENDIF.
  ENDIF.

  IF  e_zsalog IS INITIAL.
    PERFORM eval_destinations  USING gt_dests g_local.
* Prepare list processing by ALV
    PERFORM build_alv_fieldcat USING g_tabname.
    PERFORM modify_alv_fieldcat.
    PERFORM build_fcode_excl_tab USING gt_excluding.
    PERFORM build_eventtab       USING s_alvgrd gt_events.
    PERFORM build_header_lines USING gt_dests.

* Save selections for remote selection
    g_start_date = strtdate.
    g_start_time = strttime.
    g_end_date   = enddate.
    g_end_time   = endtime.
    PERFORM save_selection_parameters USING
                      g_start_date g_start_time
                      g_end_date   g_end_time
                      g_class      g_sever.

    CLEAR event_stat[].

    IF g_local = 'X'.
      PERFORM read_local_auditlog USING gt_audit_file_stat
                                        gt_rec_cnts
                                        s_audir  s_aufn
                                        g_start_date  g_start_time
```

```
                    g_end_date    g_end_time
                    g_stats.

* Check if any files where processed
   DESCRIBE TABLE gt_audit_file_stat LINES g_files_processed.
   IF g_files_processed = 1 AND g_error = 'R'.
     WRITE: / gt_msg_408.
     EXIT.
   ENDIF.

  ELSE.
*   Evaluate audit files from default directory (profile params)
*   Do remote selection
   PERFORM read_remote_auditlog USING gt_dests gt_entriestab[].
  ENDIF.

* Check if any files where processed
  DELETE gt_audit_file_stat WHERE filename(1) = '<'.
  DESCRIBE TABLE gt_audit_file_stat LINES g_files_processed.
  IF g_files_processed = 0 AND g_error <> 'R'.
    WRITE: / gt_msg_405.
    EXIT.
  ENDIF.
  IF g_local IS INITIAL.
    DESCRIBE TABLE gt_entriestab LINES g_entread.
  ELSE.
*   Local processing does not create a table entriestab
    DESCRIBE TABLE gt_outtab LINES g_entread.
  ENDIF.
  IF g_entread = 0.
    IF g_error <> 'R'.
      WRITE: / gt_msg_406.
    ENDIF.
    EXIT.
  ENDIF.

  IF g_local IS INITIAL.
*   Select audit entries and build output table for ALV
   PERFORM select_entries_from_auditlog USING gt_entriestab[] g_stats.
  ENDIF.

  SORT  gt_outtab.
  DELETE ADJACENT DUPLICATES FROM gt_outtab COMPARING ALL FIELDS.

  LOOP AT gt_outtab.
    MOVE gt_outtab-alguser TO out_zsalog-bname.
    MOVE gt_outtab-algtcode  TO out_zsalog-tcode.
    MOVE gt_outtab-alginst   TO out_zsalog-apserver.
    MOVE gt_outtab-algdate   TO out_zsalog-saldate.
    MOVE gt_outtab-algtime   TO out_zsalog-saltime.
    MOVE gt_outtab-alslgttyp TO out_zsalog-salcat.
    MOVE gt_outtab-algfileno TO out_zsalog-salno.
    MOVE gt_outtab-algclient TO out_zsalog-salclient.
    MOVE gt_outtab-algterm   TO out_zsalog-terminal.
    MOVE gt_outtab-algsubid  TO out_zsalog-auditcc.
    MOVE gt_outtab-algrepna  TO out_zsalog-salprog.
    MOVE gt_outtab-algtext   TO out_zsalog-saltext.
    APPEND out_zsalog.
  ENDLOOP.

  IF NOT s_zsalog IS INITIAL.
    PERFORM update_zsalog TABLES out_zsalog.
  ENDIF.
```

```
    IF NOT s_alv IS INITIAL.
      PERFORM display_alv_list.
    ENDIF.
  ENDIF.

*---------------------------------------------------------------------*
*       FORM update_zsalog                                            *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
FORM update_zsalog TABLES p_zsalog STRUCTURE zsalog.
  DATA count TYPE i.
  DATA count2 TYPE i.
  CLEAR count.

  LOOP AT p_zsalog.
    SELECT  SINGLE *
        FROM zsalog
        WHERE  bname     = p_zsalog-bname
        AND    tcode     = p_zsalog-tcode
        AND    apserver  = p_zsalog-apserver
        AND    saldate   = p_zsalog-saldate
        AND    saltime   = p_zsalog-saltime
        AND    auditcc   = p_zsalog-auditcc
        AND    salprog   = p_zsalog-salprog
        AND    salcat    = p_zsalog-salcat
        AND    salno     = p_zsalog-salno
        AND    salclient = p_zsalog-salclient
        AND    terminal  = p_zsalog-terminal
        AND    saltext   = p_zsalog-saltext.

    IF sy-subrc <> 0.
      INSERT zsalog FROM p_zsalog.
      IF sy-subrc = 0.
        count = count + 1.
        IF count = 1000.
          COMMIT WORK AND WAIT.
          count2 = count2 + count.
          WRITE:/ 'inserted & commited : ', count2,' recs'.
          CLEAR count.
        ENDIF.
      ELSE.
        WRITE:/ 'ERROR, no insert ZSALOG:', sy-subrc,' rec:', p_zsalog.
      ENDIF.
    ENDIF.

  ENDLOOP.

  count2 = count2 + count.
  COMMIT WORK AND WAIT.
  WRITE:/ 'Upd & commited totally : ', count2,' recs'.


ENDFORM.
.
.
.
```

```
*&---------------------------------------------------------------------*
*&      FORM read_remote_auditlog
*&---------------------------------------------------------------------*
*       Contact remote server via remote function call RSAU_READ_FILE
*       and build a common table (ot_entriestab) with audit records
*----------------------------------------------------------------------*
*  -->  pt_all_dests   table with all rfc-destinations to contact
*  <--  ot_entriestab  table with all remote selected audit records
*----------------------------------------------------------------------*
FORM read_remote_auditlog USING    pt_all_dests  TYPE ty_all_dests
                          CHANGING ot_entriestab TYPE ty_audit_t_log.

.
.
* Get selections for remote selection
  CALL FUNCTION 'RSLG_SEL_DUMP'
       IMPORTING
            itself = ta.
  CALL FUNCTION 'RSAU_SEL_DUMP'
       IMPORTING
            itself = ta_audit.

* Now start working and ask all active applications servers for
* sending their AudiLog data to me.

  LOOP AT pt_all_dests INTO ls_all_dests.

    IF ls_all_dests-access <> 0.
      CLEAR ls_rec_cnts.
      MOVE 'R'                 TO ls_rec_cnts-kind.
      MOVE-CORRESPONDING ta    TO ls_rec_cnts.
      MOVE ls_all_dests-access TO ls_rec_cnts-errrc.
      APPEND ls_rec_cnts TO gt_rec_cnts.
    ELSE.
      ta-rfcdest = ls_all_dests-rfcdest.
      CLEAR ls_rec_cnts.
      MOVE 'R'              TO ls_rec_cnts-kind.
      MOVE-CORRESPONDING ta TO ls_rec_cnts.

*     Read that file
      CLEAR it_entriestab_2[].
      CLEAR audit_file_stat_rfc[].

*     Use auxiliary table, so that the growing main table is
*     not copied to each computer and then back again.

      CALL FUNCTION 'RSAU_READ_FILE'
        DESTINATION ls_all_dests-rfcdest
        EXPORTING
          selection             = ta
          selection_audit       = ta_audit
          file_no_old           = file_no
        IMPORTING
          end_reason            = end_reason
          end_info              = end_info
          counters              = l_recordcnt
          file_no_new           = file_no
        TABLES
          syslog_in_table       = it_entriestab_2
          audit_file_stat       = audit_file_stat_rfc
        EXCEPTIONS
          system_failure        = 2  MESSAGE last_rfc_mess
          communication_failure = 3  MESSAGE last_rfc_mess.
      IF sy-subrc <> 0.
*       save error information for later display
        ls_rec_cnts-errrc   = sy-subrc.
        ls_rec_cnts-errmess = last_rfc_mess.
      ELSE.
*       copy new entries from last server into global table
        LOOP AT it_entriestab_2 INTO ls_entriestab_2.
          ls_entriestab = ls_entriestab_2.
          APPEND ls_entriestab TO ot_entriestab.
        ENDLOOP.
*       save audit file statistic
        LOOP AT audit_file_stat_rfc INTO ls_audit_file_stat_rfc.
          ls_audit_file_stat = ls_audit_file_stat_rfc.
          APPEND ls_audit_file_stat TO gt_audit_file_stat.
        ENDLOOP.
        MOVE-CORRESPONDING l_recordcnt TO ls_rec_cnts.
      ENDIF.
*     save record statistic
      APPEND ls_rec_cnts TO gt_rec_cnts.
      CLEAR l_recordcnt.
    ENDIF.

  ENDLOOP.

* I have done some work.
* I did not update anything in the database, but I spent some
* time and will tell the ABAP/4 processor, that I finished a
* step and that I am still alive.
*
  COMMIT WORK.

  LOOP AT gt_rec_cnts INTO ls_rec_cnts.
    IF ls_rec_cnts-errrc <> 0.
      PERFORM display_rfc_errors USING gt_rec_cnts.
      g_error = 'R'.
      EXIT.
    ENDIF.
  ENDLOOP.

ENDFORM. "read_remote_auditlog

*&---------------------------------------------------------------------*
*&      FORM eval_destinations
*&---------------------------------------------------------------------*
*       Build a table with all rfc-destinations of servers belonging
*       to this system
*----------------------------------------------------------------------*
*  <--  ot_all_dests   table with all rfc-destinations to contact
*----------------------------------------------------------------------*
FORM eval_destinations CHANGING ot_all_dests TYPE ty_all_dests
                                o_only_local TYPE as4flag.


.
.
  CLEAR ot_all_dests[].

*  DESCRIBE TABLE server LINES l_entries.
*  BKK : P01 - several dest  TS1 - no remote servers
  IF sy-sysid <> 'P01'.
*   No server specified, evaluate local system and switch to local
*   evaluation. Read profile params to get audit dir and filename.
    o_only_local = 'X'.
    CALL 'C_SAPGPARAM' ID 'NAME'  FIELD 'DIR_AUDIT'       "#EC CI_CCALL
```

```
                       ID 'VALUE' FIELD s_audir.
    CALL 'C_SAPGPARAM' ID 'NAME'  FIELD 'FN_AUDIT'        "#EC CI_CCALL
                       ID 'VALUE' FIELD s_aufn.
    EXIT.
  ELSE.
*   get all active server and select some or all
    CALL FUNCTION 'RFC_GET_LOCAL_DESTINATIONS'
         TABLES
              localdest     = lt_local_dests
         EXCEPTIONS
              not_available = 1.
    IF sy-subrc <> 0.
      MESSAGE e404(sl).
    ENDIF.

    IF  NOT ( l_entries = 1 AND server-sign = 'I' AND
          server-option = 'CP' AND server-low = '*' ).
*     evaluate selected server
      LOOP AT lt_local_dests INTO ls_local_dests.
        CHECK NOT ( ls_local_dests-rfcdest IN server ).
        DELETE lt_local_dests INDEX sy-tabix.
      ENDLOOP.
    ENDIF.
  ENDIF.

  DESCRIBE TABLE lt_local_dests LINES l_entries.
  IF l_entries = 0.
    MESSAGE e404(sl).
  ENDIF.

* Get the list of all applications servers, which are currently
* active.
  LOOP AT lt_local_dests INTO ls_local_dests.
    MOVE-CORRESPONDING ls_local_dests TO ls_all_dests.
    MOVE               0              TO ls_all_dests-access.
    APPEND ls_all_dests TO ot_all_dests.
  ENDLOOP.

* Get the list of all applications servers, which are known to
* the computer center management system.
  CALL FUNCTION 'RZL_GET_BA_DESCR'
       EXPORTING
            betriebsart_name           = ' '  "all
       TABLES
            instance_description_table = lt_spfid.

* Note down those applications servers, which are known to the
* computer center management system, but which are currently not
* active.
  LOOP AT lt_spfid INTO ls_spfid.
    READ TABLE ot_all_dests WITH KEY rfcdest = ls_spfid-apserver
         INTO ls_all_dests.
    IF sy-subrc <> 0.
      MOVE-CORRESPONDING ls_spfid          TO ls_all_dests.
      MOVE               ls_spfid-apserver TO ls_all_dests-rfcdest.
      MOVE               4                 TO ls_all_dests-access.
      APPEND ls_all_dests TO ot_all_dests.
    ENDIF.
  ENDLOOP.

ENDFORM. "eval_destinations
```

```
*&---------------------------------------------------------------------*
*&      FORM get_local_instance
*&---------------------------------------------------------------------*
*       Get the name of the local instance
*----------------------------------------------------------------------*
*  <--  o_instance   name of the local instance
*----------------------------------------------------------------------*
FORM get_local_instance CHANGING o_instance TYPE spfid-instname.
.
.
  CALL 'C_GET_CPU_ID' ID 'CPUID' FIELD local_host.        "#EC CI_CCALL
  local_instance = local_host.
  CONCATENATE local_instance sy-sysid INTO local_instance
    SEPARATED BY '_'.

  CALL 'C_GET_SYSTEM_NUMBER' ID 'SYSTEM' FIELD sapsystem. "#EC CI_CCALL
  CONCATENATE local_instance sapsystem INTO local_instance
    SEPARATED BY '_'.

  o_instance = local_instance.
ENDFORM.                    "get_local_instance
.
.
.
```

```
*&---------------------------------------------------------------------*
*&      FORM save_selection_parameters
*&---------------------------------------------------------------------*
*       Save select options and parameters for a remote selection with
*       function module RSAU_READ_FILE. This interface does not
*       support selection tables (select options).
*----------------------------------------------------------------------*
*  -->  p_start_date     start date to select audit files
*  -->  p_start_time     start time to select audit files
*  -->  p_end_date       end date to select audit files
*  -->  p_end_time       end time to select audit files
*  <--  o_class          describes the audit classes to select
*  <--  o_sever          describes the audit severity to select
*----------------------------------------------------------------------*
FORM save_selection_parameters
                           USING
                              p_start_date TYPE d  p_start_time TYPE t
                              p_end_date   TYPE d  p_end_time   TYPE t
                           CHANGING
                              o_class      TYPE rsausel-subclasid
                              o_sever      TYPE rsausel-severity.

  PERFORM save_selection_date_range USING
                  p_start_date  p_start_time
                  p_end_date    p_end_time.

  PERFORM save_selection_classes  USING o_class.
  PERFORM save_selection_severity USING o_sever.
  PERFORM save_selection_user.
  PERFORM save_selection_tcode.
  PERFORM save_selection_terminal.
  PERFORM save_selection_report.

  CALL FUNCTION 'RSLG_SEL_ST_WHICHLOG'                    "note 865422
    EXPORTING file = 'L'.     "'L', 'R' or 'A' is valid here

ENDFORM.                     "save_selection_parameters

*&---------------------------------------------------------------------*
*&      FORM save_selection_classes
*&---------------------------------------------------------------------*
*       Compute and save select information for audit classes
*----------------------------------------------------------------------*
*  <--  o_clssum     describes the audit classes to select
*----------------------------------------------------------------------*
FORM save_selection_classes CHANGING o_clssum TYPE i.
.
.

  ADD rsau_class_other    TO sum_cls.
  ADD rsau_class_login    TO sum_cls.
  ADD rsau_class_tastart  TO sum_cls.
  ADD rsau_class_report   TO sum_cls.
  ADD rsau_class_rfclogin TO sum_cls.
  ADD rsau_class_user     TO sum_cls.
  ADD rsau_class_rfccall  TO sum_cls.
  ADD rsau_class_system   TO sum_cls.

  IF sum_cls = 0. MESSAGE e401(sl). ENDIF.

  clsmap = sum_cls.
  CALL FUNCTION 'RSAU_SEL_ST_SUBCLASID'
       EXPORTING
            clsmap = clsmap.
  o_clssum = sum_cls.
ENDFORM.                    "save_selection_classes

*&---------------------------------------------------------------------*
*&      FORM save_selection_severity
*&---------------------------------------------------------------------*
*       Compute and save select information for audit severity
*----------------------------------------------------------------------*
*  <--  o_sevsel     describes the audit severity to select
*----------------------------------------------------------------------*
FORM save_selection_severity CHANGING o_sevsel TYPE i.
.
.
  IF low      = 'X'.
    sev_sel = rsau_seve_low.
  ELSEIF medium  = 'X'.
    sev_sel = rsau_seve_med.
  ELSEIF high    = 'X'.
    sev_sel = rsau_seve_high.
  ENDIF.

  IF sev_sel = 0. MESSAGE e402(sl). ENDIF.

  CALL FUNCTION 'RSAU_SEL_ST_SEVERITY'
       EXPORTING
            sever = sev_sel.

  o_sevsel = sev_sel.
ENDFORM.                    "save_selection_severity
*&---------------------------------------------------------------------*
*&      FORM save_selection_user
*&---------------------------------------------------------------------*
*       Use low value from selection table if this table contains only
*       one value
*----------------------------------------------------------------------*
FORM save_selection_user.
.
.
  DESCRIBE TABLE user LINES l_entries.
  IF l_entries = 1 AND
     user-sign = 'I' AND user-option = 'EQ'.
    l_user = user-low.

    CALL FUNCTION 'RSLG_SEL_ST_USER'
         EXPORTING
              user = l_user.
  ENDIF.

ENDFORM.                    "save_selection_user
```

```
*&---------------------------------------------------------------------*
*&      FORM save_selection_tcode
*&---------------------------------------------------------------------*
*       Use low value from selection table if this table contains only
*       one value
*----------------------------------------------------------------------*
FORM save_selection_tcode.
.
.
  DESCRIBE TABLE tcode LINES l_entries.
  IF l_entries = 1 AND
     tcode-sign = 'I' AND tcode-option = 'EQ'.
    l_tcode = tcode-low.
    CALL FUNCTION 'RSLG_SEL_ST_TCODE'
         EXPORTING
              tcode = l_tcode.
  ENDIF.
ENDFORM.                    "save_selection_tcode
*&---------------------------------------------------------------------*
*&      FORM save_selection_report
*&---------------------------------------------------------------------*
*       Use low value from selection table if this table contains only
*       one value
*----------------------------------------------------------------------*
FORM save_selection_report.

  DATA: l_entries TYPE sy-tabix,
        l_repna   TYPE rslgentr-slgrepna.

  DESCRIBE TABLE report LINES l_entries.
  IF l_entries = 1 AND
     report-sign = 'I' AND report-option = 'EQ'.
    l_repna = report-low.
    CALL FUNCTION 'RSLG_SEL_ST_REPNA'
         EXPORTING
              repna = l_repna.
  ENDIF.
ENDFORM.                    "save_selection_report
*&---------------------------------------------------------------------*
*&      FORM save_selection_terminal
*&---------------------------------------------------------------------*
*       Use low value from selection table if this table contains only
*       one value
*----------------------------------------------------------------------*
FORM save_selection_terminal.
.
.
  DESCRIBE TABLE terminal LINES l_entries.
  IF l_entries = 1 AND
     terminal-sign = 'I' AND terminal-option = 'EQ'.
    l_term = terminal-low.
    CALL FUNCTION 'RSLG_SEL_ST_TERM'
         EXPORTING
              terminal = l_term.
  ENDIF.
ENDFORM.                    "save_selection_terminal
```

```
*&---------------------------------------------------------------------*
*&      FORM save_selection_date_range
*&---------------------------------------------------------------------*
*       Check and save parameters that describe the date and time range
*----------------------------------------------------------------------*
*  -->  p_start_date     start date to select audit files
*  -->  p_start_time     start time to select audit files
*  -->  p_end_date       end date to select audit files
*  -->  p_end_time       end time to select audit files
*----------------------------------------------------------------------*
FORM save_selection_date_range USING p_start_date TYPE d
                                     p_start_time TYPE t
                                     p_end_date   TYPE d
                                     p_end_time   TYPE t.

  IF  p_start_date <> '        ' AND p_start_date <> '00000000'
  AND p_start_time <> '      '   AND p_start_time <> '000000'
  AND p_end_date   <> '        ' AND p_end_date   <> '00000000'
  AND p_end_time   <> '      '   AND p_end_time   <> '000000'.

*   All four fields given.
    IF p_start_date > p_end_date.
      MESSAGE e170(sl).
    ELSEIF p_start_date = p_end_date  AND
           p_start_time > p_end_time.
      MESSAGE e170(sl).
    ENDIF.
  ELSEIF  p_start_date <> '        ' AND p_start_date <> '00000000'
  AND     p_end_date   <> '        ' AND p_end_date   <> '00000000'.

*   Two date fields given.
    IF p_start_date > p_end_date.
      MESSAGE e170(sl).
    ENDIF.
  ENDIF.

  IF p_start_date = p_end_date AND p_start_time = p_end_time.
    MESSAGE e170(sl).
  ENDIF.

  CALL FUNCTION 'RSLG_SEL_ST_TIMES'
       EXPORTING
            sdate = p_start_date
            stime = p_start_time
            edate = p_end_date
            etime = p_end_time.

ENDFORM. "save_selection_date_range
.
.
.
```

```
*&---------------------------------------------------------------------*
*&      Form  get_audit_files
*&---------------------------------------------------------------------*
*       Create an audit file list from a given directory
*       Similar to function RSAU_LIST_AUDIT_FILES (SECU)
*       Audit directory and filenames must be specified and are not
*       read form profile parameters.
*----------------------------------------------------------------------*
*  -->  p_dir_audit    directory where to find audit files
*  -->  p_fn_audit     file name pattern to select audit files
*  <--  ot_aufile_list returned audit file list
*----------------------------------------------------------------------*
FORM get_audit_files USING    p_dir_audit    TYPE rsauflist-file_name
                        value(p_fn_audit)    TYPE rsauflist-dir_name
                     CHANGING ot_aufile_list TYPE ty_t_rsauflist.

.
.
.

* When using extended filenames change the number pattern '######'
* otherwise the files will not be recognized by later CP
  TRANSLATE p_fn_audit USING '#+'.

  CALL 'C_DIR_READ_FINISH'                                  "#EC CI_CCALL
      ID 'ERRNO'  FIELD file-errno
      ID 'ERRMSG' FIELD file-errmsg.

  CALL 'C_DIR_READ_START' ID 'DIR'    FIELD p_dir_audit   "#EC CI_CCALL
                          ID 'ERRNO'  FIELD file-errno
                          ID 'ERRMSG' FIELD file-errmsg.
  IF sy-subrc <> 0.
    MESSAGE ID 'TC' TYPE 'I' NUMBER '004'
                 WITH file-errno 'C_DIR_READ_START'
                       file-errmsg p_dir_audit.
    EXIT.
  ENDIF.

  DO.
    CLEAR file.
    CALL 'C_DIR_READ_NEXT'                                  "#EC CI_CCALL
      ID 'TYPE'   FIELD file-type
      ID 'NAME'   FIELD file-file_name
      ID 'LEN'    FIELD file-len
      ID 'OWNER'  FIELD file-owner
      ID 'MTIME'  FIELD file-mtime
      ID 'MODE'   FIELD file-mode
      ID 'ERRNO'  FIELD file-errno
      ID 'ERRMSG' FIELD file-errmsg.
    file-dir_name  = p_dir_audit.

    MOVE sy-subrc TO file-subrc.
    CASE sy-subrc.
      WHEN 0.
        CLEAR: file-errno, file-errmsg.
      WHEN 1.
        EXIT.
      WHEN OTHERS. " SY-SUBRC >= 2
        ADD 1 TO errcnt.
        IF errcnt > 10.
          MESSAGE ID 'TC' TYPE 'I' NUMBER '004'
              WITH file-errno 'C_DIR_READ_NEXT'
                   file-errmsg p_dir_audit.
          EXIT.
        ENDIF.
    ENDCASE.

*   * Does the filename contains the requested pattern?
*   * Then store it, else forget it.
    IF file-file_name CP p_fn_audit.
      lp_mdate = file-mtime / 86400.    "convert to days
      lp_mdate = lp_mdate + lc_unix_date.  "add Unix-Start
      file-mdate = lp_mdate.            "convert to type date
      MOVE-CORRESPONDING file TO ls_rsauflist.
      APPEND ls_rsauflist TO ot_aufile_list.
    ENDIF.
  ENDDO.

  CALL 'C_DIR_READ_FINISH'                                "#EC CI_CCALL
      ID 'ERRNO'  FIELD file-errno
      ID 'ERRMSG' FIELD file-errmsg.
  IF sy-subrc <> 0.
*   WRITE: / 'C_DIR_READ_FINISH', 'SUBRC', SY-SUBRC.
  ENDIF.

ENDFORM.                    "get_audit_files

*&---------------------------------------------------------------------*
*&      Form  print_audit_file_stat
*&---------------------------------------------------------------------*
*       Create ALV header table with information from audit file
*       statistic table, i. e. records read, records selected.
*----------------------------------------------------------------------*
*  <->  pt_rsaufinfo    table with audit file statistics (back sorted)
*  <--  ot_eol_comment  returned ALV header table
*----------------------------------------------------------------------*
FORM print_audit_file_stat CHANGING
                              pt_rsaufinfo   TYPE ty_t_rsaufinfo
                              ot_eol_comment TYPE slis_t_listheader.

  DATA: c_header     TYPE slis_listheader-info.
  DATA: ls_line      TYPE slis_listheader,
        ls_rsaufinfo TYPE rsaufinfo,
        l_instname   TYPE rsaufinfo-instname,
        l_p1(8)      TYPE p,
        l_p2(8)      TYPE p,
        l_p3(8)      TYPE p.

  CLEAR ot_eol_comment[].

  c_header = 'Statistics for Analyzed Audit Log Files'(036).
  ls_line-typ  = 'H'.
  ls_line-info = c_header.
  APPEND ls_line TO ot_eol_comment.

  SORT pt_rsaufinfo BY fileno.

  LOOP AT pt_rsaufinfo INTO ls_rsaufinfo.
    CLEAR ls_line.
    ls_line-typ  = 'S'.
    IF ls_rsaufinfo-instname <> l_instname.
      ls_line-key  = ls_rsaufinfo-instname.
      l_instname   = ls_rsaufinfo-instname.
    ENDIF.
    ls_line-info = ls_rsaufinfo-filename.
    APPEND ls_line TO ot_eol_comment.

    CLEAR ls_line.
    ls_line-typ     = 'S'.
    ls_line-key     = ' '.
    ls_line-info+2  = gt_text_024.
    ls_line-info+16 = gt_text_025.
    ls_line-info+32 = gt_text_026.
    APPEND ls_line TO ot_eol_comment.

    CLEAR ls_line.
    ls_line-typ  = 'S'.
    ls_line-key  = ' '.
    l_p1 = ls_rsaufinfo-reccnt.
    l_p2 = ls_rsaufinfo-recgood.
*    l_p3 = ls_rsaufinfo-maxsize.
    WRITE l_p1 TO ls_line-info(12).
    WRITE l_p2 TO ls_line-info+14(12).
*    WRITE l_p3 TO ls_line-info+28(12).
    IF ls_rsaufinfo-endreason = 'E'.
      WRITE gt_text_028 TO ls_line-info+35.
    ELSE.
      WRITE gt_text_027 TO ls_line-info+35.
      WRITE ls_rsaufinfo-endreason TO ls_line-info+40.
    ENDIF.
    APPEND ls_line TO ot_eol_comment.
  ENDLOOP.

ENDFORM.                    "print_audit_file_stat

*&---------------------------------------------------------------------*
*&      Form  get_severity_stat
*&---------------------------------------------------------------------*
*       Get severity statistics about selected audit events
*----------------------------------------------------------------------*
*  -->  pt_event_stat   table with audit file statistics
*  <--  o_sev_high      number of high events
*  <--  o_sev_med       number of medium events
*  <--  o_sev_low       number of low events
*----------------------------------------------------------------------*
FORM get_severity_stat USING pt_event_stat TYPE ty_event_t_stat
                       CHANGING o_sev_high o_sev_med o_sev_low.

  DATA: ls_event_stat   TYPE ty_eventstat,
        l_severity      TYPE tsl1d-severity.

  LOOP AT pt_event_stat INTO ls_event_stat.
*   get subclasid and severity for this message from TSL1D
    CALL FUNCTION 'RSAU_HANDLE_TSL1D'
         EXPORTING
              aulogarea  = ls_event_stat-area
              aulogsubid = ls_event_stat-subid
         IMPORTING
              severity   = l_severity.

    CASE l_severity.
      WHEN rsau_seve_high.
        ADD ls_event_stat-cnt TO o_sev_high.
      WHEN rsau_seve_med.
        ADD ls_event_stat-cnt TO o_sev_med.
      WHEN rsau_seve_low.
        ADD ls_event_stat-cnt TO o_sev_low.
    ENDCASE.

  ENDLOOP.
ENDFORM.                    "get_severity_stat
*&--------------------------------------------------------------------*
```

```
*&      Form  init_selection_texts
*&---------------------------------------------------------------------*
*       Fallback selection texts handling
*----------------------------------------------------------------------*
FORM init_selection_texts.

  gt_text_024 = 'Read'(024).
  gt_text_025 = 'Selected'(025).
  gt_text_026 = 'Errors'(026).
  gt_text_027 = 'Yes'(027).
  gt_text_028 = 'No'(028).

  t_tx_001 = 'Time Restrictions'(001).
  t_tx_002 = 'From Date/Time'(002).
  t_tx_003 = 'To Date/Time'(003).
  t_tx_004 = 'Standard Selections'(004).
  t_tx_006 = 'Events'(006).
  t_tx_008 = 'Options'(008).

  t_tx_053 = 'Instance name'(053).
  t_tx_054 = 'Client'(054).
  t_tx_055 = 'User'(055).
  t_tx_056 = 'Terminal'(056).
  t_tx_057 = 'Transaction Code'(057).
  t_tx_058 = 'Program'(058).
  t_tx_059 = 'Text in the Message'(059).
  t_tx_071 = 'Only Critical'(071).
  t_tx_072 = 'Severe and Critical'(072).
  t_tx_073 = 'Every'(073).
  t_tx_082 = 'Display ALV'(082).
  t_tx_085 = 'Update ZSALOG'(085).
  t_tx_086 = 'EMPTY ZSALOG'(086).

  gt_msg_405 = 'No data was found on the server'.
  gt_msg_406 = 'The result set for this selection was empty'.
  gt_msg_408 = 'Could not open audit file (does not exist)'(040).
  gt_msg_488 = 'Error delete ZSALOG '(088).
  gt_msg_412 = 'This layout variant does not exist'(041).

ENDFORM.                    "init_selection_texts
*&---------------------------------------------------------------------*
*&      Form  delete_zsalog
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM delete_zsalog.

  DELETE FROM zsalog WHERE bname IS not NULL.
  IF sy-subrc <> 0.
    WRITE: / gt_msg_488 NO-GAP, ' R C : ' NO-GAP, sy-subrc.
    EXIT.
  ENDIF.
  COMMIT WORK AND WAIT.

ENDFORM.                    " delete_zsalog
```

4. Appendix – Core Code for the Detection Engine

ABAP Program ZRBS012A detects misuse and anomaly incidents as noted in this research paper.

```
.
.
START-OF-SELECTION.

  CLEAR anonym_field.

  SELECT * INTO TABLE itab FROM zsalog.

* Maintenance of own access rights
  IF NOT %accown IS INITIAL.
    MOVE text-102 TO ialv-rep_type.
    LOOP AT itab
        WHERE tcode = 'SU01'.
      IF itab-saltext CS itab-bname.
        IF NOT %ush04 IS INITIAL.
          SELECT SINGLE * FROM ush04 WHERE
               modda = itab-saldate AND
               bname = itab-bname.
          IF sy-subrc = 0.
            MOVE-CORRESPONDING itab TO ialv-zsal.
            IF NOT %ano1 IS INITIAL.
              PERFORM field_anom_ialv.
            ENDIF.
            APPEND ialv.
          ENDIF.
        ELSE.
          MOVE-CORRESPONDING itab TO ialv-zsal.
          IF NOT %ano1 IS INITIAL.
            PERFORM field_anom_ialv.
          ENDIF.
          APPEND ialv.
        ENDIF.
      ENDIF.
    ENDLOOP.
  ENDIF.
*
* SOD Risks
  IF NOT %sod IS INITIAL.
    SORT itab BY bname.
    LOOP AT itab.
      AT NEW bname.
        CLEAR: itabz,itabz_sod1,itabz_sod2,count, w_sod, w_priv.
        REFRESH: itabz,itabz_sod1,itabz_sod2.
        IF NOT %prvusr IS INITIAL AND NOT itab-bname IS INITIAL.
          SELECT SINGLE * FROM usr02
               WHERE bname =  itab-bname
                 AND ( class = 'PRIVILEGED'
                  OR   class = 'SOD' ).
          IF sy-subrc = 0.
            w_priv = 'X'.
          ENDIF.
        ENDIF.
      ENDAT.

      CHECK w_priv IS INITIAL.

      MOVE-CORRESPONDING itab TO itabz-zsal.
      MOVE-CORRESPONDING itab TO itabz_sod1-zsal.
      MOVE-CORRESPONDING itab TO itabz_sod2-zsal.
      APPEND itabz.
      APPEND itabz_sod1.                " temp storage1 for this user
      APPEND itabz_sod2.                " temp storage2 for this user
      AT END OF bname.
* Check all tcodes for user with SOD risk in 2 or 3 tcode combination
        LOOP AT itabz.
          SELECT * INTO TABLE itab_sod FROM zsodtcodes3
               WHERE tcode = itabz-zsal-tcode.
          IF sy-subrc = 0.
            LOOP AT itab_sod.
              READ TABLE itabz_sod1
                   WITH KEY zsal-tcode = itab_sod-sodtcode1.
              IF sy-subrc = 0.
                IF itab_sod-sodtcode2 IS INITIAL.
* User has two tcodes SOD combination (in zsodtcodes3) in zsalog
                  CONCATENATE text-104 '- 2 trans comb.'
                  INTO ialv-rep_type.
                  MOVE-CORRESPONDING itabz TO ialv.
                  IF NOT %ano2 IS INITIAL.
                    PERFORM field_anom_ialv.
                  ENDIF.
                  APPEND ialv.
                  MOVE-CORRESPONDING itabz_sod1 TO ialv.
                  IF NOT %ano2 IS INITIAL.
                    PERFORM field_anom_ialv.
                  ENDIF.
                  APPEND ialv.
                  ADD 1 TO count_sod.
                  MOVE 'X' TO w_sod.
                ELSE.
                  READ TABLE itabz_sod2
                       WITH KEY zsal-tcode = itab_sod-sodtcode2.
                  IF sy-subrc = 0.
* user har komb av 3 transer (som rec i zsodtcodes3) i zsalog
                    CONCATENATE text-104 '- 2 trans comb.'
                    INTO ialv-rep_type.
                    MOVE-CORRESPONDING itabz TO ialv.
                    IF NOT %ano2 IS INITIAL.
                      PERFORM field_anom_ialv.
                    ENDIF.
                    APPEND ialv.
                    MOVE-CORRESPONDING itabz_sod1 TO ialv.
                    IF NOT %ano2 IS INITIAL.
                      PERFORM field_anom_ialv.
                    ENDIF.
                    APPEND ialv.
                    MOVE-CORRESPONDING itabz_sod2 TO ialv.
                    IF NOT %ano2 IS INITIAL.
                      PERFORM field_anom_ialv.
                    ENDIF.
                    APPEND ialv.
                    ADD 1 TO count_sod.
                    MOVE 'X' TO w_sod.
                  ENDIF.
                ENDIF.
              ENDIF.
            ENDLOOP.
          ENDIF.
        ENDLOOP.
        IF NOT w_sod IS INITIAL.
          ADD 1 TO count_sod_user.
          CLEAR w_sod.
        ENDIF.
      ENDAT.

    ENDLOOP.

  ENDIF.

  CLEAR: count,post_role,is_role,anonym_field..
*
* Dualism check
  IF NOT %dualism IS INITIAL.

    PERFORM dualism_tables_build.

    MOVE text-301 TO ialv-rep_type.
    SORT itab BY bname.
    LOOP AT itab.
      AT NEW bname.
       CLEAR: itabz,count,is_role,post_role,tcode_exclpost,tcode_exclis.
        REFRESH itabz.
        SELECT agr_name FROM agr_users INTO w-agr_name
         WHERE uname = itab-bname.
* check if both IS&Posting roles:
          IF sy-subrc = 0.
            IF w-agr_name(2) = 'Z:'.
              is_role =  'X'.
            ELSEIF w-agr_name(2) = 'Z_'.
              post_role =  'X'.
            ENDIF.
          ENDIF.
        ENDSELECT.
      ENDAT.
      IF is_role = 'X' AND post_role = 'X'.
        MOVE-CORRESPONDING itab TO itabz-zsal.
        APPEND itabz.
*check if both exclusive IS/Posting transactions in audit log
        READ TABLE gtpostexcl WITH  KEY tcode = itab-tcode.
        IF sy-subrc = 0.
          tcode_exclpost = 'X'.
        ELSE.
          READ TABLE gtisexcl WITH  KEY tcode = itab-tcode.
          IF sy-subrc = 0.
            tcode_exclis = 'X'.
          ENDIF.
        ENDIF.
      ENDIF.
      AT END OF bname.
        IF is_role = 'X' AND post_role = 'X'
        AND tcode_exclpost = 'X' AND tcode_exclis = 'X'.
          IF NOT %privel IS INITIAL.
            SELECT SINGLE * FROM usr02
                 WHERE bname =  itab-bname
                   AND class = 'PRIVILEGED'.
            IF sy-subrc <> 0.
              LOOP AT itabz.
                MOVE-CORRESPONDING itabz TO ialv.
                IF NOT %ano3 IS INITIAL.
                  PERFORM field_anom_ialv.
                ENDIF.
                APPEND ialv.
              ENDLOOP.
              ADD 1 TO count_du.
            ENDIF.
          ELSE.
            LOOP AT itabz.
```

```
              MOVE-CORRESPONDING itabz TO ialv.
              IF NOT %ano3 IS INITIAL.
                PERFORM field_anom_ialv.
              ENDIF.
              APPEND ialv.
            ENDLOOP.
            ADD 1 TO count_du.
          ENDIF.
        ENDIF.
      ENDAT.
    ENDLOOP.
  ENDIF.

* Excessive downloads:
  IF NOT %excdwnl IS INITIAL.
    REFRESH itabz.
    MOVE text-311 TO ialv-rep_type.
    LOOP AT itab
        WHERE auditcc = 'Y'.
      IF %zusr IS INITIAL.
        MOVE-CORRESPONDING itab TO itabz-zsal.
        APPEND itabz.
      ELSE.
        IF itab-bname(1) = 'Z'.
          MOVE-CORRESPONDING itab TO itabz-zsal.
          APPEND itabz.
        ENDIF.
      ENDIF.
    ENDLOOP.
    SORT itabz BY zsal-bname.
    LOOP AT itabz.
      ADD 1 TO i.
      AT NEW zsal-bname.
        IF i > 5.
          ix = sy-tabix - 1.
          READ TABLE itabz INDEX ix.
          MOVE-CORRESPONDING itabz TO ialv.
          IF NOT %ano6 IS INITIAL.
            PERFORM field_anom_ialv.
          ENDIF.
          APPEND ialv.
        ENDIF.
        CLEAR i.
      ENDAT.
    ENDLOOP.


  ENDIF.

* Authorization failure:
  IF NOT %failaut IS INITIAL.
    MOVE text-211 TO ialv-rep_type.
    LOOP AT itab
        WHERE auditcc = '4'.
      IF %validtc IS INITIAL.
        MOVE-CORRESPONDING itab TO ialv-zsal.
        IF NOT %ano5 IS INITIAL.
          PERFORM field_anom_ialv.
        ENDIF.
        ADD 1 TO count_af.
        APPEND ialv.
      ELSE.
        SELECT SINGLE * FROM agr_tcodes WHERE
        tcode = itab-tcode.
    IF sy-subrc = 0.
     MOVE-CORRESPONDING itab TO ialv-zsal.
     IF NOT %ano5 IS INITIAL.
      PERFORM field_anom_ialv.
     ENDIF.
     ADD 1 TO count_af.
     APPEND ialv.
    ENDIF.
   ENDIF.
  ENDLOOP.
 ENDIF.
*

* Logon failure:
 IF NOT %faillog IS INITIAL.
  MOVE text-112 TO ialv-rep_type.
  LOOP AT itab
    WHERE auditcc = '2'.
   IF %validus  IS INITIAL.
    MOVE-CORRESPONDING itab TO ialv-zsal.
    IF NOT %ano4 IS INITIAL.
     PERFORM field_anom_ialv.
    ENDIF.
    ADD 1 TO count_lf.
    APPEND ialv.
   ELSE.
    SELECT SINGLE * FROM usr02 WHERE
    bname = itab-bname.
    IF sy-subrc = 0.
     MOVE-CORRESPONDING itab TO ialv-zsal.
     IF NOT %ano4 IS INITIAL.
      PERFORM field_anom_ialv.
     ENDIF.
     ADD 1 TO count_lf.
     APPEND ialv.
    ENDIF.
   ENDIF.
  ENDLOOP.
 ENDIF.
*
.
.
.
```

5. Appendix Table ZSALOG

Customized Table ZSALOG which contains unique records from the collected security audit logs.

Fields in this table:
BNAME, characters of length 12, used for the user ID name from the user master record
TCODE, characters of length 20, Transaction code
APSERVER, characters of length 20, which contains the name of the application server
SALDATE, in date format of 8, contains the date
SALTIME, in time format of 6, contains the time for the incident
AUDITCC, characters of length 3 is the field for Audit Class Code

SALPROG	CHAR	40	Program name
SALCAT	CHAR	3	Category of Security Audit Log Entry (Batch or Dialog)
SALNO	NUMC	2	Message Number of Security Audig Log Entry
SALCLIENT	NUMC	3	Client Number of Security Audit Log Entry
TERMINAL	CHAR	8	Terminal ID of Security Audit Log Entry
SALTEXT	CHAR	60	Descriptive text for Security Audit Log Entry

6. Appendix Table ZSODTCODES3

Customized Table ZSODTCODES contains combination of tcodes with high risk in segregation of duties. This is just a simple table with four fields in our case. The first three fields in the record are reserved for transaction codes and the last field is for the SOD description.

7. Appendix Table ZANOSUB

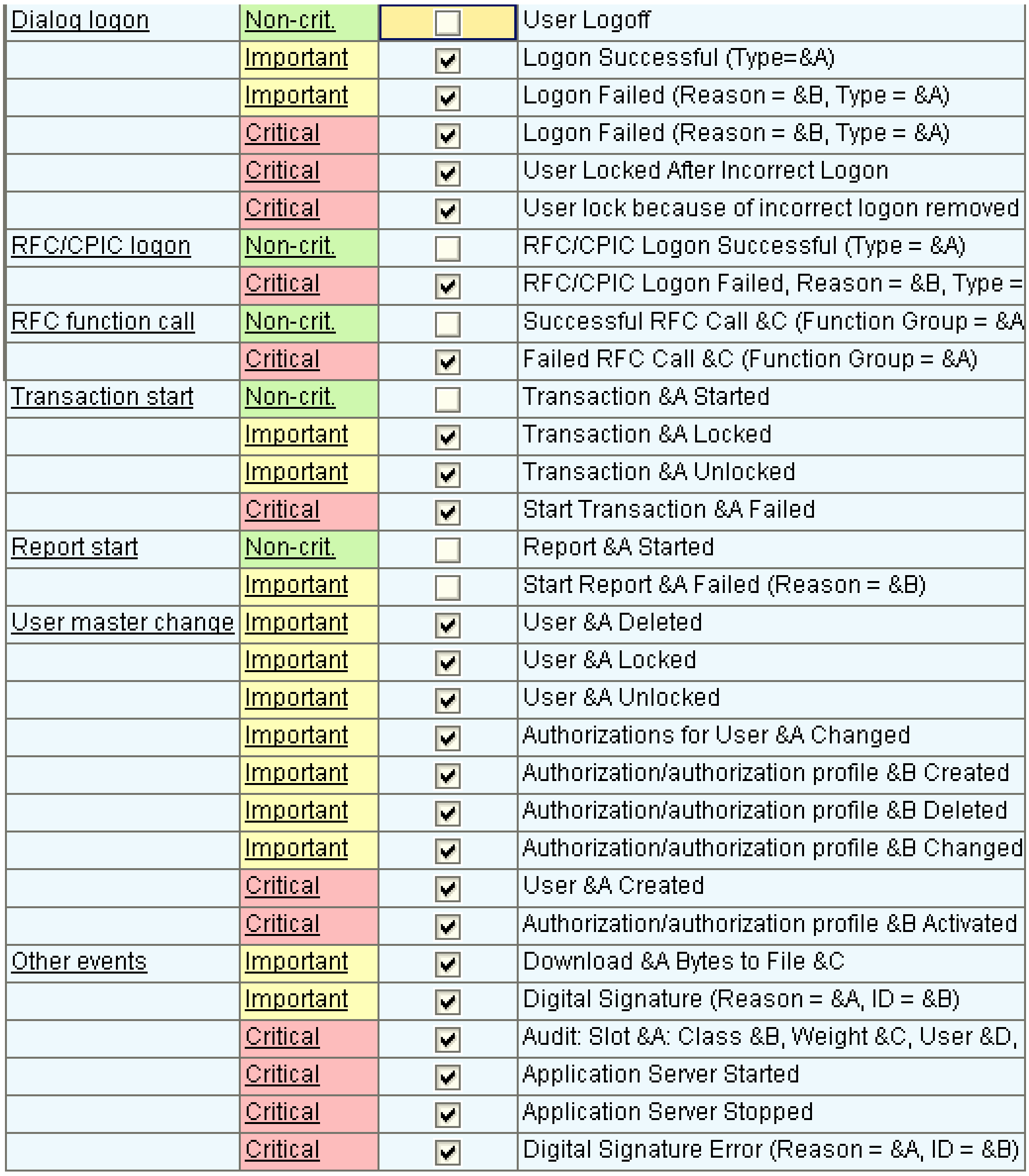
Customized Table ZANOSUB used for anonymization by substitution.

Size category 0, Data records expected: 0 to 300.000

Fields in this table:
CLEAR, character of length 1, used for clear text character.
SUBST, character of length 1, used for substitution.

8. Appendix - Security Audit Filter Settings

Security Audit Filter settings for our system. To ensure logging of all transaction code activity, the non-critical transaction code start was also checked. Thus we could get at complete SOD risk analysis.



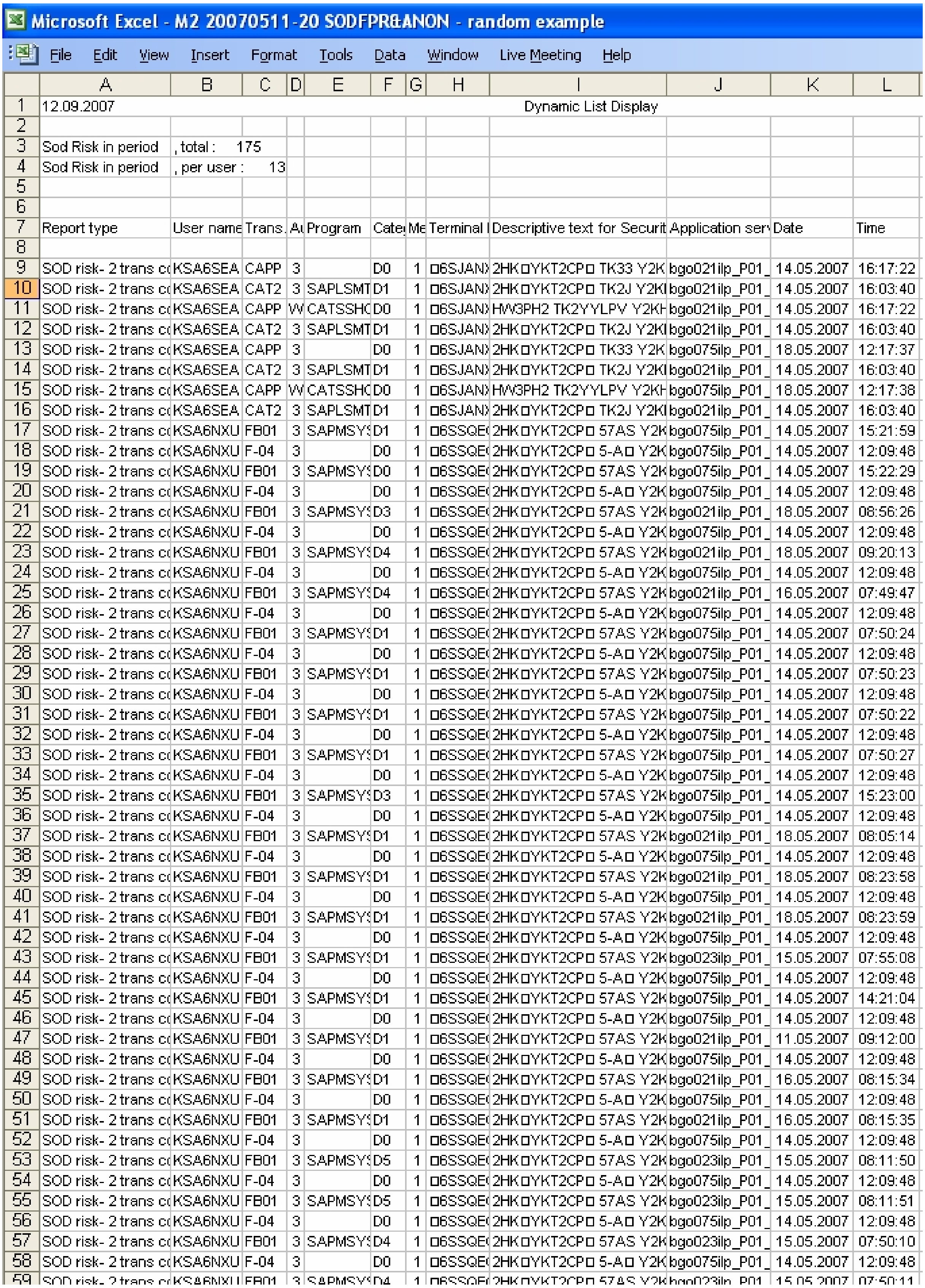
Dialog logon	Non-crit.	☐	User Logoff
	Important	☑	Logon Successful (Type=&A)
	Important	☑	Logon Failed (Reason = &B, Type = &A)
	Critical	☑	Logon Failed (Reason = &B, Type = &A)
	Critical	☑	User Locked After Incorrect Logon
	Critical	☑	User lock because of incorrect logon removed
RFC/CPIC logon	Non-crit.	☐	RFC/CPIC Logon Successful (Type = &A)
	Critical	☑	RFC/CPIC Logon Failed, Reason = &B, Type =
RFC function call	Non-crit.	☐	Successful RFC Call &C (Function Group = &A
	Critical	☑	Failed RFC Call &C (Function Group = &A)
Transaction start	Non-crit.	☐	Transaction &A Started
	Important	☑	Transaction &A Locked
	Important	☑	Transaction &A Unlocked
	Critical	☑	Start Transaction &A Failed
Report start	Non-crit.	☐	Report &A Started
	Important	☐	Start Report &A Failed (Reason = &B)
User master change	Important	☑	User &A Deleted
	Important	☑	User &A Locked
	Important	☑	User &A Unlocked
	Important	☑	Authorizations for User &A Changed
	Important	☑	Authorization/authorization profile &B Created
	Important	☑	Authorization/authorization profile &B Deleted
	Important	☑	Authorization/authorization profile &B Changed
	Critical	☑	User &A Created
	Critical	☑	Authorization/authorization profile &B Activated
Other events	Important	☑	Download &A Bytes to File &C
	Important	☑	Digital Signature (Reason = &A, ID = &B)
	Critical	☑	Audit: Slot &A: Class &B, Weight &C, User &D,
	Critical	☑	Application Server Started
	Critical	☑	Application Server Stopped
	Critical	☑	Digital Signature Error (Reason = &A, ID = &B)

9. Appendix - Security Audit Log Codes

Audit Codes

AU	0	Audit - Test. Text: &A
AU	1	Logon Successful (Type=&A)
AU	2	Logon Failed (Reason = &B, Type = &A)
AU	3	Transaction &A Started
AU	4	Start Transaction &A Failed
AU	5	RFC/CPIC Logon Successful (Type = &A)
AU	6	RFC/CPIC Logon Failed, Reason = &B, Type = &A
AU	7	User &A Created
AU	8	User &A Deleted
AU	9	User &A Locked
AU	A	User &A Unlocked
AU	B	Authorizations for User &A Changed
AU	C	User Logoff
AU	D	User Master Record &A Changed
AU	E	Audit Configuration has Changed
AU	F	Audit: Slot &A: Class &B, Weight &C, User &D, Client &E
AU	G	Application Server Started
AU	H	Application Server Stopped
AU	I	Audit: Slot &A Inactive
AU	J	Audit: Active Status Set to &1
AU	K	Successful RFC Call &C (Function Group = &A)
AU	L	Failed RFC Call &C (Function Group = &A)
AU	M	User Locked After Incorrect Logon
AU	N	User lock because of incorrect logon removed
AU	O	Logon Failed (Reason = &B, Type = &A)
AU	P	Transaction &A Locked
AU	Q	Transaction &A Unlocked
AU	R	&A &B Created
AU	S	&A &B Deleted
AU	T	&A &B Changed
AU	U	&A &B Activated
AU	V	Digital Signature Error (Reason = &A, ID = &B)
AU	W	Report &A Started
AU	X	Start Report &A Failed (Reason = &B)
AU	Y	Download &A Bytes to File &C
AU	Z	Digital Signature (Reason = &A, ID = &B)
AV	1	Recording audit events was stopped (reason=&A)
AV	2	Error &A occurred when reading table &B key = '&C' (LOC = &D)
AV	3	Warning: Maximum level of the security audit log file exceeds &A percent

10. Appendix – Anonymized Partial Misuse Detection Log

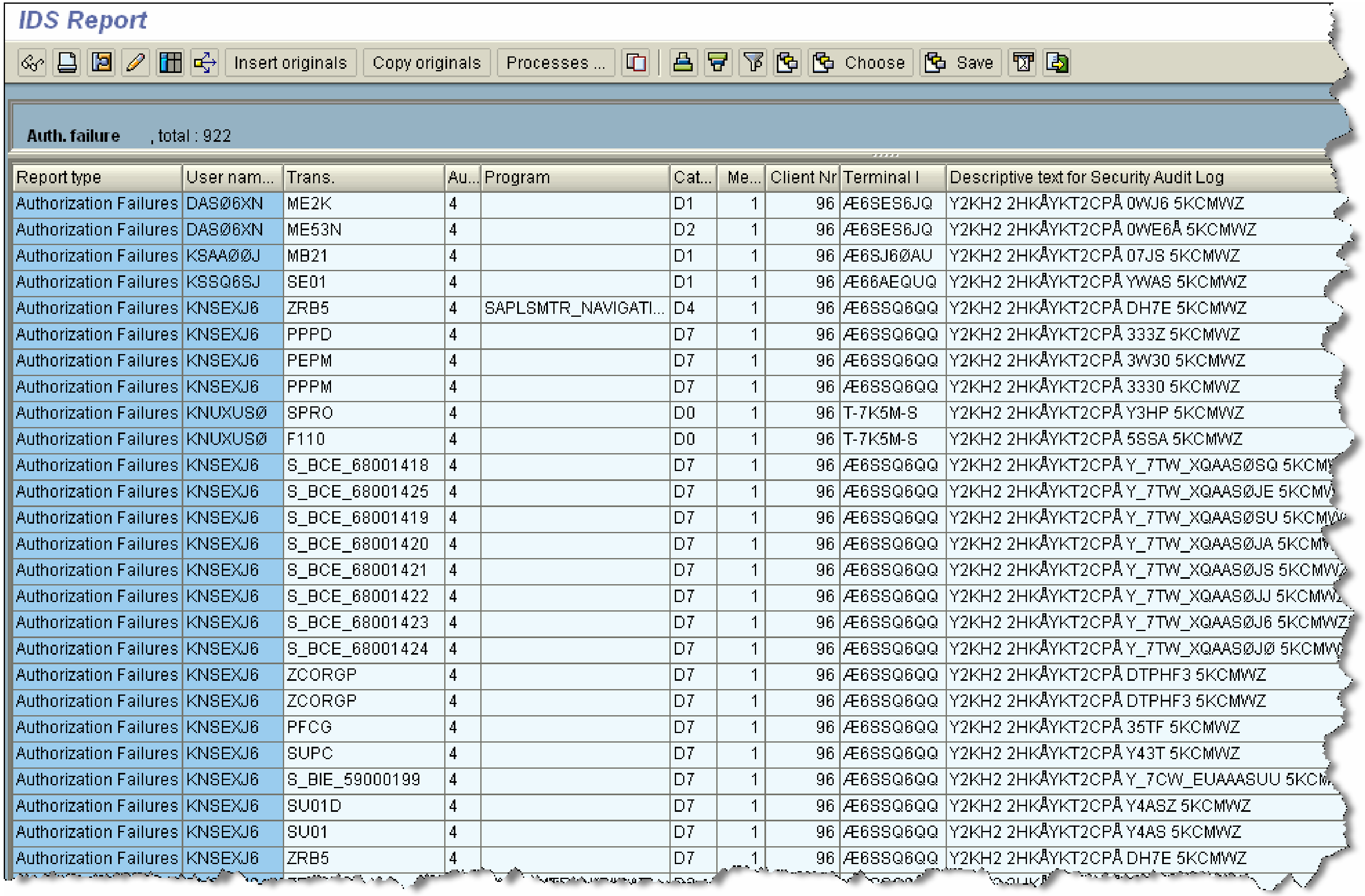


Microsoft Excel - M2 20070511-20 SODFPR&ANON - random example

	A	B	C	D	E	F	G	H	I	J	K	L
1	12.09.2007								Dynamic List Display			
2												
3	Sod Risk in period	, total :	175									
4	Sod Risk in period	, per user :	13									
5												
6												
7	Report type	User name	Trans.	A	Program	Cate	Me	Terminal I	Descriptive text for Securit	Application ser	Date	Time
8												
9	SOD risk- 2 trans c	KSA6SEA	CAPP	3		D0	1	□6SJAN	2HK□YKT2CP□ TK33 Y2K	bgo021ilp_P01_	14.05.2007	16:17:22
10	SOD risk- 2 trans c	KSA6SEA	CAT2	3	SAPLSMT	D1	1	□6SJAN	2HK□YKT2CP□ TK2J Y2K	bgo021ilp_P01_	14.05.2007	16:03:40
11	SOD risk- 2 trans c	KSA6SEA	CAPP	W	CATSSH	D0	1	□6SJAN	HW3PH2 TK2YYLPV Y2K	bgo021ilp_P01_	14.05.2007	16:17:22
12	SOD risk- 2 trans c	KSA6SEA	CAT2	3	SAPLSMT	D1	1	□6SJAN	2HK□YKT2CP□ TK2J Y2K	bgo021ilp_P01_	14.05.2007	16:03:40
13	SOD risk- 2 trans c	KSA6SEA	CAPP	3		D0	1	□6SJAN	2HK□YKT2CP□ TK33 Y2K	bgo075ilp_P01_	18.05.2007	12:17:37
14	SOD risk- 2 trans c	KSA6SEA	CAT2	3	SAPLSMT	D1	1	□6SJAN	2HK□YKT2CP□ TK2J Y2K	bgo021ilp_P01_	14.05.2007	16:03:40
15	SOD risk- 2 trans c	KSA6SEA	CAPP	W	CATSSH	D0	1	□6SJAN	HW3PH2 TK2YYLPV Y2K	bgo075ilp_P01_	18.05.2007	12:17:38
16	SOD risk- 2 trans c	KSA6SEA	CAT2	3	SAPLSMT	D1	1	□6SJAN	2HK□YKT2CP□ TK2J Y2K	bgo021ilp_P01_	14.05.2007	16:03:40
17	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo075ilp_P01_	14.05.2007	15:21:59
18	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
19	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D0	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo075ilp_P01_	14.05.2007	15:22:29
20	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
21	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D3	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	18.05.2007	08:56:26
22	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
23	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D4	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	18.05.2007	09:20:13
24	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
25	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D4	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	16.05.2007	07:49:47
26	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
27	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo075ilp_P01_	14.05.2007	07:50:24
28	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
29	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo075ilp_P01_	14.05.2007	07:50:23
30	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
31	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo075ilp_P01_	14.05.2007	07:50:22
32	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
33	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo075ilp_P01_	14.05.2007	07:50:27
34	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
35	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D3	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo075ilp_P01_	14.05.2007	15:23:00
36	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
37	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	18.05.2007	08:05:14
38	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
39	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	18.05.2007	08:23:58
40	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
41	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	18.05.2007	08:23:59
42	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
43	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo023ilp_P01_	15.05.2007	07:55:08
44	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
45	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo075ilp_P01_	14.05.2007	14:21:04
46	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
47	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	11.05.2007	09:12:00
48	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
49	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	16.05.2007	08:15:34
50	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
51	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D1	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo021ilp_P01_	16.05.2007	08:15:35
52	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
53	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D5	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo023ilp_P01_	15.05.2007	08:11:50
54	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
55	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D5	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo023ilp_P01_	15.05.2007	08:11:51
56	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
57	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D4	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo023ilp_P01_	15.05.2007	07:50:10
58	SOD risk- 2 trans c	KSA6NXU	F-04	3		D0	1	□6SSQE	2HK□YKT2CP□ 5-A□ Y2K	bgo075ilp_P01_	14.05.2007	12:09:48
59	SOD risk- 2 trans c	KSA6NXU	FB01	3	SAPMSY	D4	1	□6SSQE	2HK□YKT2CP□ 57AS Y2K	bgo023ilp_P01_	15.05.2007	07:50:11

11. Screenshot of Authorization Errors Report

Additional columns with information continues to the right of this screenshot. User name, Terminal ID, and description are anonymized fields.

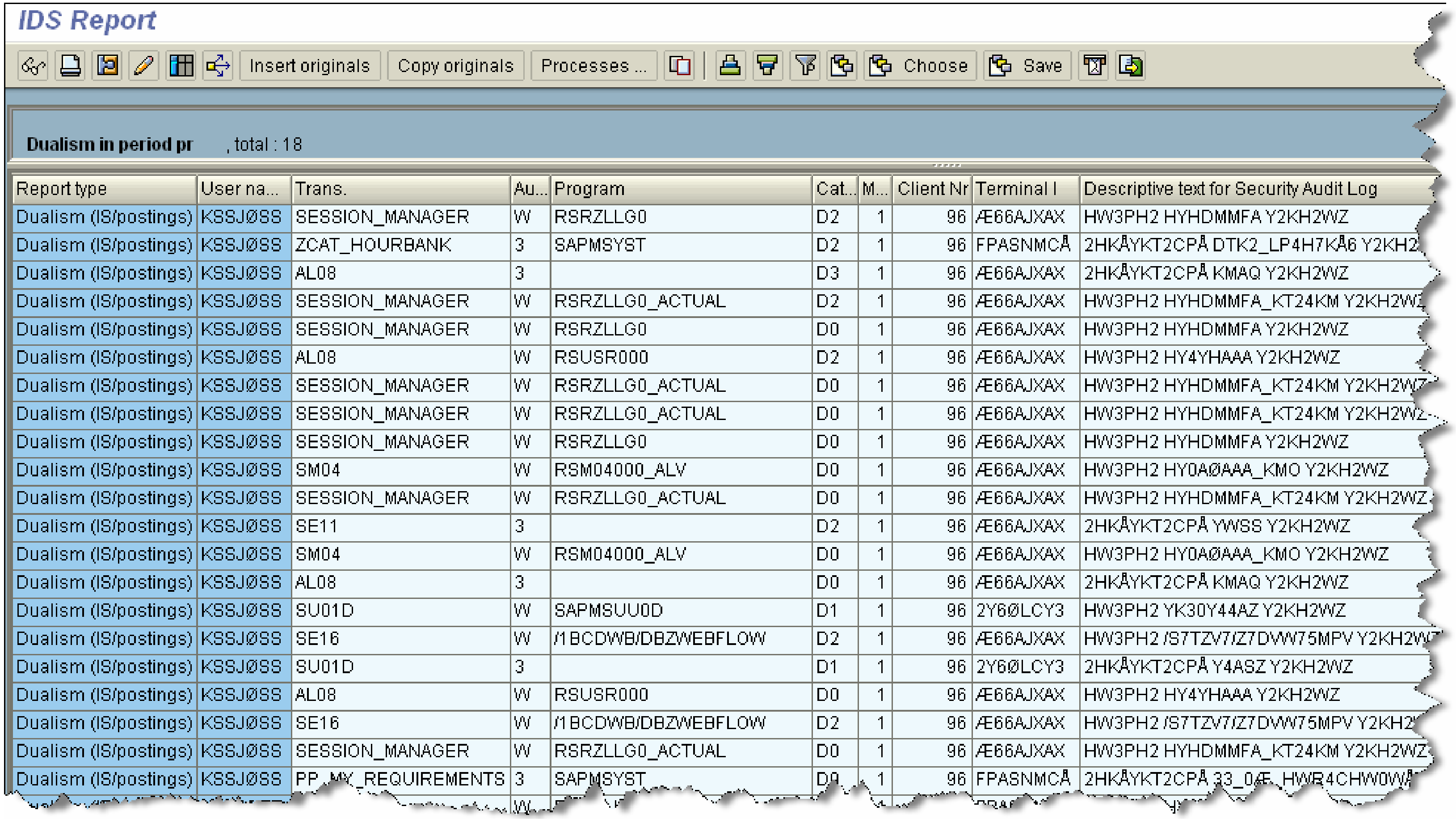


Auth. failure , total : 922

Report type	User nam...	Trans.	Au...	Program	Cat...	Me...	Client Nr	Terminal I	Descriptive text for Security Audit Log
Authorization Failures	DASØ6XN	ME2K	4		D1	1	96	Æ6SES6JQ	Y2KH2 2HKÅYKT2CPÅ 0WJ6 5KCMWZ
Authorization Failures	DASØ6XN	ME53N	4		D2	1	96	Æ6SES6JQ	Y2KH2 2HKÅYKT2CPÅ 0WE6Å 5KCMWZ
Authorization Failures	KSAAØØJ	MB21	4		D1	1	96	Æ6SJ6ØAU	Y2KH2 2HKÅYKT2CPÅ 07JS 5KCMWZ
Authorization Failures	KSSQ6SJ	SE01	4		D1	1	96	Æ66AEQUQ	Y2KH2 2HKÅYKT2CPÅ YWAS 5KCMWZ
Authorization Failures	KNSEXJ6	ZRB5	4	SAPLSMTR_NAVIGATI...	D4	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ DH7E 5KCMWZ
Authorization Failures	KNSEXJ6	PPPD	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ 333Z 5KCMWZ
Authorization Failures	KNSEXJ6	PEPM	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ 3W30 5KCMWZ
Authorization Failures	KNSEXJ6	PPPM	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ 3330 5KCMWZ
Authorization Failures	KNUXUSØ	SPRO	4		D0	1	96	T-7K5M-S	Y2KH2 2HKÅYKT2CPÅ Y3HP 5KCMWZ
Authorization Failures	KNUXUSØ	F110	4		D0	1	96	T-7K5M-S	Y2KH2 2HKÅYKT2CPÅ 5SSA 5KCMWZ
Authorization Failures	KNSEXJ6	S_BCE_68001418	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7TW_XQAASØSQ 5KCM
Authorization Failures	KNSEXJ6	S_BCE_68001425	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7TW_XQAASØJE 5KCMW
Authorization Failures	KNSEXJ6	S_BCE_68001419	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7TW_XQAASØSU 5KCMW
Authorization Failures	KNSEXJ6	S_BCE_68001420	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7TW_XQAASØJA 5KCM
Authorization Failures	KNSEXJ6	S_BCE_68001421	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7TW_XQAASØJS 5KCMW
Authorization Failures	KNSEXJ6	S_BCE_68001422	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7TW_XQAASØJJ 5KCMW
Authorization Failures	KNSEXJ6	S_BCE_68001423	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7TW_XQAASØJ6 5KCMWZ
Authorization Failures	KNSEXJ6	S_BCE_68001424	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7TW_XQAASØJØ 5KCMW
Authorization Failures	KNSEXJ6	ZCORGP	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ DTPHF3 5KCMWZ
Authorization Failures	KNSEXJ6	ZCORGP	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ DTPHF3 5KCMWZ
Authorization Failures	KNSEXJ6	PFCG	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ 35TF 5KCMWZ
Authorization Failures	KNSEXJ6	SUPC	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y43T 5KCMWZ
Authorization Failures	KNSEXJ6	S_BIE_59000199	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y_7CW_EUAAASUU 5KCM
Authorization Failures	KNSEXJ6	SU01D	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y4ASZ 5KCMWZ
Authorization Failures	KNSEXJ6	SU01	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ Y4AS 5KCMWZ
Authorization Failures	KNSEXJ6	ZRB5	4		D7	1	96	Æ6SSQ6QQ	Y2KH2 2HKÅYKT2CPÅ DH7E 5KCMWZ

12.Screenshot of Dualism Report

Additional columns with information continues to the right of this screenshot. User name, Terminal ID, and description are anonymized fields.



Dualism in period pr , total : 18

Report type	User na...	Trans.	Au...	Program	Cat...	M...	Client Nr	Terminal I	Descriptive text for Security Audit Log
Dualism (IS/postings)	KSSJØSS	SESSION_MANAGER	W	RSRZLLG0	D2	1	96	Æ66AJXAX	HW3PH2 HYHDMMFA Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	ZCAT_HOURBANK	3	SAPMSYST	D2	1	96	FPASNMCÅ	2HKÅYKT2CPÅ DTK2_LP4H7KÅ6 Y2KH2
Dualism (IS/postings)	KSSJØSS	AL08	3		D3	1	96	Æ66AJXAX	2HKÅYKT2CPÅ KMAQ Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SESSION_MANAGER	W	RSRZLLG0_ACTUAL	D2	1	96	Æ66AJXAX	HW3PH2 HYHDMMFA_KT24KM Y2KH2W
Dualism (IS/postings)	KSSJØSS	SESSION_MANAGER	W	RSRZLLG0	D0	1	96	Æ66AJXAX	HW3PH2 HYHDMMFA Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	AL08	W	RSUSR000	D2	1	96	Æ66AJXAX	HW3PH2 HY4YHAAA Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SESSION_MANAGER	W	RSRZLLG0_ACTUAL	D0	1	96	Æ66AJXAX	HW3PH2 HYHDMMFA_KT24KM Y2KH2W
Dualism (IS/postings)	KSSJØSS	SESSION_MANAGER	W	RSRZLLG0_ACTUAL	D0	1	96	Æ66AJXAX	HW3PH2 HYHDMMFA_KT24KM Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SESSION_MANAGER	W	RSRZLLG0	D0	1	96	Æ66AJXAX	HW3PH2 HYHDMMFA Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SM04	W	RSM04000_ALV	D0	1	96	Æ66AJXAX	HW3PH2 HY0AØAAA_KMO Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SESSION_MANAGER	W	RSRZLLG0_ACTUAL	D0	1	96	Æ66AJXAX	HW3PH2 HYHDMMFA_KT24KM Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SE11	3		D2	1	96	Æ66AJXAX	2HKÅYKT2CPÅ YWSS Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SM04	W	RSM04000_ALV	D0	1	96	Æ66AJXAX	HW3PH2 HY0AØAAA_KMO Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	AL08	3		D0	1	96	Æ66AJXAX	2HKÅYKT2CPÅ KMAQ Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SU01D	W	SAPMSUU0D	D1	1	96	2Y6ØLCY3	HW3PH2 YK30Y44AZ Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SE16	W	/1BCDWB/DBZWEBFLOW	D2	1	96	Æ66AJXAX	HW3PH2 /S7TZV7/Z7DVW75MPV Y2KH2W
Dualism (IS/postings)	KSSJØSS	SU01D	3		D1	1	96	2Y6ØLCY3	2HKÅYKT2CPÅ Y4ASZ Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	AL08	W	RSUSR000	D0	1	96	Æ66AJXAX	HW3PH2 HY4YHAAA Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	SE16	W	/1BCDWB/DBZWEBFLOW	D2	1	96	Æ66AJXAX	HW3PH2 /S7TZV7/Z7DVW75MPV Y2KH2
Dualism (IS/postings)	KSSJØSS	SESSION_MANAGER	W	RSRZLLG0_ACTUAL	D0	1	96	Æ66AJXAX	HW3PH2 HYHDMMFA_KT24KM Y2KH2WZ
Dualism (IS/postings)	KSSJØSS	PP_MY_REQUIREMENTS	3	SAPMSYST	D0	1	96	FPASNMCÅ	2HKÅYKT2CPÅ 33_0Æ_HWR4CHW0WÅ