Knowledge-Based Recognition of Security Attacks in Telecommunication Networks

Sebastian Abeck, Jodok Batlogg, Robert Scholderer
University of Karlsruhe
Institute for Telematics
Zirkel 2, D-76128 Karlsruhe
Phone: +49-721-608-{6391,6393,6394}, Fax: +49-721-608-4046
email:{abeck,batlogg,scholder}@telematik.informatik.uni-karlsruhe.de
http://wwwcm.telematik.informatik.uni-karlsruhe.de

ABSTRACT: Providers of telecommunication networks are faced with the problem of security attacks both from outside and inside. Means to prevent such attacks are security mechanisms running in the network components and organizational rules, such as security policies defined by the provider of the telecommunication network. To ensure that these security means work effectively, the provider has to observe the operation of the network. Therefore (security management) information is logged, e.g. by components which build the network. The analysis of such logging information can give hints if security mechanisms and/or security policies are circumvented.
In this paper we describe the problem of detecting security attacks based on logging information with the example of a real and complex scenario. In this scenario the network components are digital switching nodes, such as EWSD by Siemens and S12 by Alcatel. To cope with the mass of logging data which is generated by all these components - in our scenario about 3 Gbytes per day - an adequate log archive has to be designed and implemented. An efficient analysis of the logging data residing in the archive is provided by a management tool we implemented in cooperation with the network provider experts. We describe the tool, a kind of expert system to detect irregular operations in the telecommunication network automatically.

KEYWORDS: Security Logging, Data Analysis, Security Attacks, Telecommunication Networks

INTRODUCTION

This paper examines the knowledge-based recognition of security attacks in a networked system on the basis of a concrete example. The networked system used for the purposes of our study is a **telecommunications network** which is constructed from a partially intermeshed network of digital switching nodes (SN). The SNs are very complex systems which can only be effectively operated with the support of special management tools and highly-skilled personnel who use these tools. The EWSD systems from Siemens and the S12 systems from Alcatel are concrete examples of these SN systems.

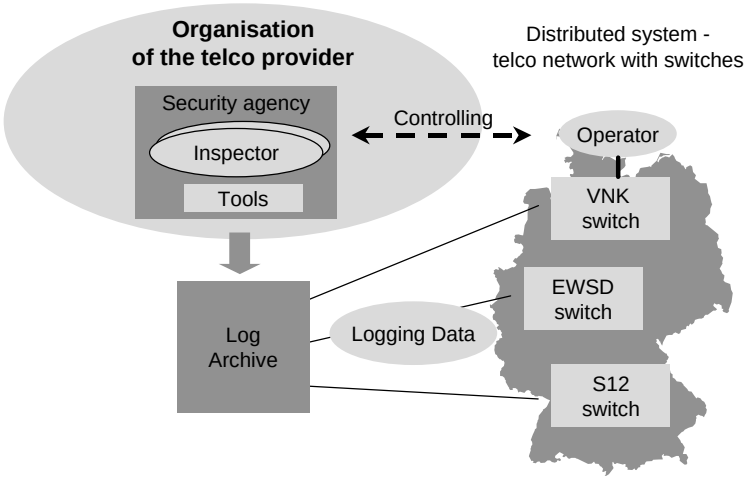

Figure 1: Szenario

We will focus on one important aspect of operation, the type of digital switching network illustrated in Figure 1 The provider has the responsibility of taking suitable measures to ensure that attacks which could pose a threat to the telecommunications network are detected and traced. An appropriate security department is set up for this purpose with so-called security inspectors. Using certain operational quality information, they look for conspicuous signs of anything which could threaten the security of the networked system being monitored.
Although it may be desirable for the provider to have complete protection against any threat of security, in practice it is not possible to provide complete guarantees because of new threat scenarios which are constantly being developed. For this reason, security mechanisms are integrated directly into the components of the digital switching networks (for example, for authentication and authorization) and a higher-ranking control (a special kind of intrusion detection) is needed which specifically checks the network for potential security attacks.
In the scenario examined here, the **log data** produced from the SNs is used to monitor the security quality of digital switching networks. The SN logs, described in more detail in Section 2, also document all the administrative operations carried out by the operating personnel of these components. The log data enables the identification of attacks to security or other security-critical wrong operations e.g. Carroll (1996). Therefore, we have to define **attack patterns** (see Section 3) which identify certain undesirable activities or sequences of activities at the switching nodes and allow the provider to detect and follow-up on potential attacks to security e.g. Hall (1996).
Sample provider-accepted tool solutions are indicated for all the means needed for knowledge-based recognition of security attacks. An integrated approach which deals with cooperation of the tool user was incorporated into the development of these tools. The interaction between all the tools produces a generic solution of **Cooperative IT Log Processing** (CILP) derived from the provider's requirements. The central CILP tools (log structure editor, log analyzer, log archive) have been designed so that they can be adapted for use with any operational problems which are processed using log data. The component in the CILP environment which coordinates the different CILP users (roles) is the log assistants. This paper closes with an outline of this type of an intelligent management tool.

A CLOSER LOOK TO THE LOGGING DATA

Now we'll take a closer look at the logging data. Logging data consists of a sequence of recordsets with different information fields. There are among other fields like username and –id, date, time, the command executed and a lot more. It is very important to know the structure of the recordsets, otherwise it is nearly impossible to analyze the logging data computer-assisted. Without information about the structure, it's only possible to search for an ASCII-string. If you want to check if a user executed a special command at a special time or with as special result, you need to know the structure of the recordsets. However it is quite difficult to get the structure of the logging data: It doesn't use any standard and varies from switch to switch. The manufacturers don't provide a specification of their logging data. Even the different software versions of one manufacturer can produce different logging data e.g. Abeck (1998). Only experts have the experience to interpret and explain the structure of the logging data. To construct a kind of "structure knowledge base" we developed our so called "structure editor". This tool allows us to construct the structure out of a plain logging file step by step and interactive. This knowledge is saved for other tools.

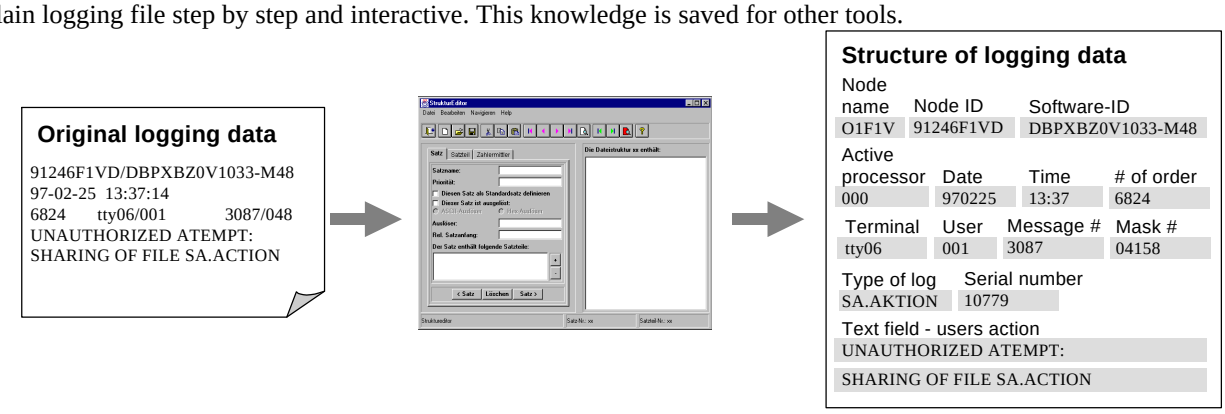

Figure 2: Structure of logging data

The log files consist of a sequence of data sets in which the following information fields appear (see Fig. 2):

- User name and unique user identifier
- Time and date
- Name of command executed
- Details about the switching node, other plain text

A knowledge of the structure of the data sets is essential for a computer-supported search for more complex attack patterns to be carried out in the log files. Without the structure information, a log file would be a pure (ASCII) character string which could only be search for conspicuous character patterns. For example an attack pattern used to check if a

user had initiated a command sequence (with specific parameter values / at a specific time interval/ with a certain output result /...) cannot be supported by a computer-based tool at the character level unless structural knowledge of the log data set exists. What we've got right now is the logging data and its structure. We have a lot of possibilities which can be searched. We may search for a few critical commands, e.g. setting up a new user or switching off the logging mechanism. Another possibility is a senseless combination of a "normal" command with a "normal" parameter. An example, is the forwarding of phone calls to dedicated phones. Some "bad guys" attract attention if there is an accumulation of a series of commands, e.g. the attempt to log in with a bad user/pass combination. Other commands attract attention if they are executed at the wrong time (e.g. weekend, public holiday). The search specification has to be done inside our tool – the logging detector. It is very flexible and allows the user to define all the desired search combinations with the aid of customised modules and user defined search patterns.

ANALYZING THE LOGGING DATA

The goal of our research is to examine the principles of attack patterns which can be detected in logging data. For analysing logging data, inspectors need information about attacks that might happen. For security requirements attack patterns must be defined. Presently providers don't have repositories for this purpose, due it's difficult to define attacks.

ATTACK PATTERNS

As our experiences showed, users are trying to use system resources for there own. The aim is to use the services of digital telecommunication networks without paying. Cracking passwords is the common way to get the administrator's or superuser's rights. The privileged status of a superuser enables the attacker to set up new users. So it is possible to hide the identity and to touch the services of the telecommunication net illegally e.g. Sienkiewicz (1994). When manipulations have been committed, attackers are trying to hide their traces by other manipulations. Audit functions will be modified, in a way that logging data will be deleted to avoid the attack to be discovered. All the actions described above occur in normal day-to-day operations, but in our case they have conspicuous circumstances:

- a lot of login attempts with a wrong user/password combination will be recognized,
- critical actions were executed out of the regular operation time
- someone uses audit functions without organizational permission.

It is easy to see, that there has to be independent security group inside the provider's organisation. If inspectors inside this group want to find these critical actions, they must first define what they want to search for. A good means to define attack patterns are regular expressions e.g. Friedl (1998).
In what follows the attack patterns will be separated into two different classes and specified with regular expressions.

1. `One command´ attack patterns:

This category contains all attack patterns, which deal with one command executed on a switching node. These attack patterns allow a sub-classification e.g. Abeck (1999):

- The first subclass contains the "simple" commands: Just one command and it is optional parameters. An example of an attack pattern based on one command is "deleting all logging data in the logfile op.action" executed on a switching node (Fig. 3).

```
(command, equal, DELETEFILE:FILE=OP.ACTION,ALL;)
```

Figure 3: `Simple One Command´ attack pattern

- More complex attack patterns constructed by combination of simple commands with additional parameters not related to the command e.g. Shieh (1991). A typical example is to set up a new user out of the regular operation time (Fig. 4).

```
(command, equal, SETUSERID:BENID=)AND(18:00,<,time,<,7:00)
```

Figure 4: `Complex One Command´ attack pattern

2. `Sequence of commands´ attack patterns:

Often it is not sufficient to search for just one command. It should be possible to model a sequence of commands executed in a certain order. Of course there may be normal commands inside the matching sequence. These "inserts" are on the one hand a result of commands executed in parallel and on the other hand critical commands that are often wide spread over several days. As an example the sequence of a "brute force cracking passwords attack" is shown in figure 5.

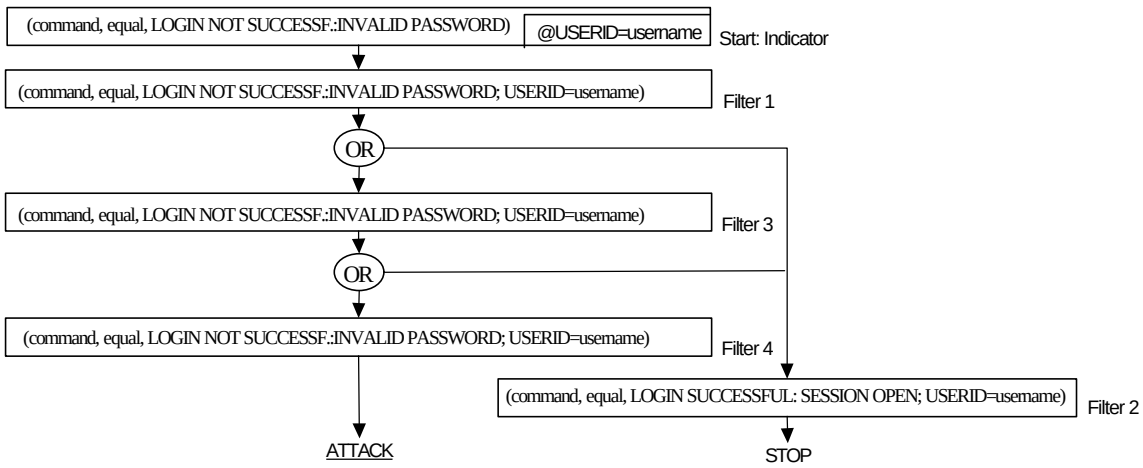

Figure 5: `Sequence of commands´ attack pattern

If the search itself is done with the aid of computers, the attack patterns have to be formulated with an editor suited to the "search program". The requirements of such an editor are:

- Simple creation of attack patterns:

A sophisticated user interface, developed in interaction with the inspectors offers the user an intuitive possibility to formulate the attack patterns as regular expressions. The difficulties of regular expressions, like terms, constants, formulas and atoms are avoided in this editor. The acceptance of this editor, called "Attack Pattern Editor" in the following text increased considerably.

- Secure and suitable storage for the created attack patterns:

Attack patterns created in the past by inspectors increase the provider's knowledge base. The attack patterns have to be saved for the systematic and suitable access by the inspectors. Providers define their own security policy, which grants the access to this sensitive data. This is important because the log analyser's use of attack patterns has to be regulated.

ANALYSIS

Having the logging data, its structure and matching attack patterns, the search itself can begin. As mentioned, the log analyser's toolkit contains functions to support the automatic and interactive analysis of logging data. These functions are integrated in the tool Log Analyser.
Sophisticated algorithms inside allow an automatic search for attacks by correlating structured logging data and attack patterns. The tool has to manage a variety of searches within logging data. The list shows basic examples for that:

- Attack patterns to all logging data
- Attack patterns to logging data in a period of time
- Attack patterns to logging data of a special component

The main modules of the algorithm inside the Log Analyser are consistency check, execution and result storage. The consistency check finds out if attack patterns can be applied to the structure of logging data. For instance, the fields of the logging data and the fields of the attack patterns have to be the same. In the execution module the logging data will be examined if the attack patterns are applicable. At last the results are collected in a storage where they can be separated in classes of different priority and then be reanalysed.
The tool - called Log Analyser - has to fulfil the following requirements:

- View to the logging data (Inspectors are used to see the original logging data)
- View to the recognised fields based on the previous defined structure e.g. time, date, user, ...
- Easy navigation through the record sets of logging data
- Available attack patterns have to be offered to the inspector

For interactive analysis of the logging data, inspectors need some other functions, which support the investigated evaluation. In this case, the log analyser offers possibilities to get an overview of parts of the logging data. For example the check of some foreign users having an account on the analysed component can be done by listing all users or other relevant information. If inspectors get results from the log analyser, they can save the hits in a special folder in an hierarchical order for reanalysing them later, or once again by concrete attack pattern.
The log analyser is implemented in JAVA and can be used on every platform of operating systems also it can read all logging data independent of its type and format. So the log analyser is a tool which supports the analysis of logging data and has the potential for detecting attacks automatically.

OUTLOOK: LOG PROCESSING ASSISTANT

Individual solutions which meet the provider's requirements were created for each aspect being processed through the derivation of tools needed for the scenario from process-oriented management means e.g. Mayerl (1998). The consolidation of these individual solutions to create a provider-accepted overall solution is the aim of the **log assistant** which is also used in the coordination and cooperation of the roles involved in log processing e.g. White (1996). Fig. 6 illustrates the different roles and the management means they require in order to carry out their activities.

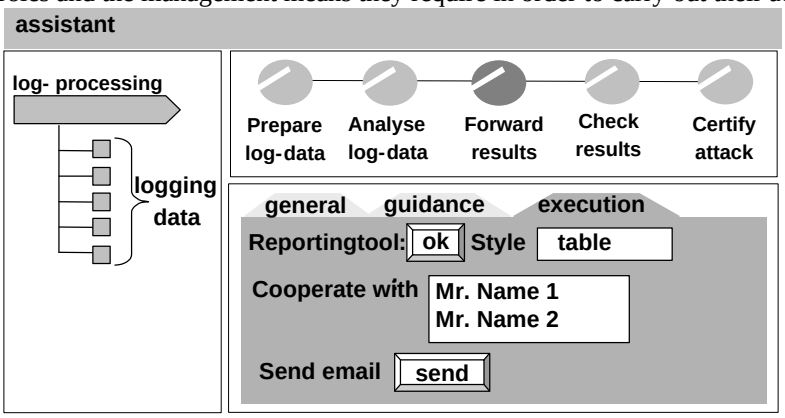

Figure 6: Log processing assistant

These results provide the basis for an examination of the effective and efficient use of process-oriented management means. This requires a coordination of the tools used so that the roles receive maximum support in the implementation of their tasks e.g. Scholderer (1998). We will not pursue how the automated processing of the tasks is controlled and instead focus on how the roles should be initiated so that they execute their tasks in accordance with the goals.

REFERENCES

Abeck, Sebastian.; Batlogg, Jodok.; Mayerl, Christian.; Scholderer, Robert., 1998, "Security logging in telecommunication networks" Second German- South African-Workshop, Gernsbach, Germany, pp. 14.

Abeck, Sebastian.; Hegering, Heinz.-Gerd; Neumair, Bernhard., 1999, "Integriertes Management vernetzter Systeme", Heidelberg, Germany.

Carroll, John, 1996, "Computer Security", Butterworth-Heinemann, Boston, USA.

Friedl, Jeffrey, 1998, "Reguläre Ausdrücke", Cambridge.

Hall, Jane., 1996, "Management of telecommunication systems and services: modeling and implementing TMN-based multi-domain management", Berlin, Springer.

Mayerl, Christian, 1998, "Process oriented approach to develop provider accepted management tools, Eunice '98 Open European Summer School on Network Management and Operation, Munich, Germany, pp. 131-138.

Scholderer, Robert, 1998, "Process oriented assistance to support the operation of a networked system", Eunice '98 Open European Summer School on Network Management and Operation, Munich, Germany, pp. 156-163.

Shieh, S. W.; Gligor V. D., 1991, "A Pattern-Oriented Intrusion-Detection Model and Ist Applications, Dpt. of Electrical Engineering", University of Maryland, College Park, MD 20742, 1991, Procedures of the IEEE Symposium on Research in Security and Privacy, pp. 327-342.

Sienkiewicz, Bodo, 1994, "Computer-Sicherheit", Addison-Wesley, Germany.

White, G.; Pooch, V., 1996, "Cooperating Security Managers: distributed intrusion detection systems", Computers & Security, Vol. 15, No. 5, pp. 441-450.