Development of Honeypot System Emulating Functions of Database Server

Antanas Čenys[a], Darius Rainys[a,b], Lukas Radvilavičius[c] & Andrej Bielko[b]
[a]Information System Laboratory
Semiconductor Physics Institute
A.Goštauto 11, Vilnius Lithuania
[b]UAB "BlueBridge"
J.Jasinskio 16, Vilnius, Lithuania
[c]Vilnius Gediminas Technical University
Saulėtekio 11, Vilnius Lithuania

cenys@uj.pfi.lt/darius@mokslas.lt/nso@xxx.lt/andrej.bielko@bluebridge.lt

SUMMARY

Possibilities to develop the honeypot type intrusion detection systems (IDS) for databases are discussed. Two types of concept honeypot systems are suggested. Network level system is based on the emulation of the database connections and is aimed to detect intruders searching for database servers and attempting to read basic database listeners information. Honeypot type database level IDS module is aimed to react to enquiries of database tables not used by real applications but loaded to the database to attract intruders.

1.0 INTRODUCTION

Importance of network security is rapidly growing all around the world and Lithuania is not an exception. Public awareness, however, is mainly concentrated on the damage related with the spread viruses and worms, or loss of productivity due to spam. Up to now it were no highly publicized incidents of other type in Lithuania. Public organizations responsible for the information security and computer security community are, however, well aware that situation is not that simple. Reliable data statistical data on illegal activities in computer network is very important both for the analysis and as a tool to raise public awareness.

Intrusion detection systems (IDS) are the main tool to detect illegal activities in the network. Most of attacks to computer networks are based on known vulnerabilities and methods. As a result malicious software can be detected from known "signatures" and illegal intrusion can detected recording untypical or suspicious activities in the system. In our days, however, hackers improve their tools, methods and skills very fast. Nowadays skilled hacker can compromise the system without owner even noticing that, since all log files or even server itself can be modified. As a result IDS should adapt to the new threats very fast. One of very promising methods receiving wide attention recently is deployment of the traps to intruders named honeypots. The main idea of the method was introduced at the beginning of 90's [Stoll 1990, Cheswick 1991]. The honeypot is a service or a system in the network without any real use. According to L. Spitzner a *honeypot is an information system resource whose value lies in unauthorized of illicit use of that recourse* [Spitzner 2003]. The mains goal of honeypot systems is to detect the illegal activities and to gather information about intruder's methods, tools, and habits remaining invisible as well as to protect organization's productive servers. According to these goals two types of honeypot systems can be distinguished: *production* honeypots and *research* honeypots. The purpose of production honeypot systems is to minimize the danger of an intrusion into organization's servers. The research honeypot systems are used for the information gathering.

Paper presented at the RTO IST Symposium on "Adaptive Defence in Unclassified Networks", held in Toulouse, France, 19 - 20 April 2004, and published in RTO-MP-IST-041.

Honeypot technology is very well suited for the security situation monitoring due to very simple analysis of collected data and absence of false alarms. To test the advantages of the honeypot technology we have deployed research purpose high-interaction honeypot in the external segment of the local network. Using the system we were monitoring security situation in the computer network during two months. Significant number of scans, probes and attempts to compromise the system was detected. Most of the attempts were simple scans, so called "script kiddies", trying to hack the system using automated tools. Rate of attacks was more or less flat during a day as well as during a week. One of the most popular attempts was to implement "*code red*" virus and other worms. Collected statistical data presented elsewhere [Rainys 2003] allows the conclusion that most of attacks come from viruses, scanning activities viruses, and low level hackers. Serious and really dangerous attacks occurred not very often.
High interaction honeypots, in the external segment is a very useful to detect and to analyze the external threats. Most of security incidents including most dangerous ones, however, arise from the inside, i.e. organisation's full time employees, temporary workers as well as the workers from the services providers. Moreover according to the expert of the network security Bruce Schneier the only one incident deserving definitely to be defined as a cyber-terrorism attack in Australia was coming from the person with the deep inside knowledge [Schneier 2000]. High interaction honeypots inside the local area network can provide very interesting information about the local activity but such deployment adds an additional risk. Most of the network and servers IDS's are also oriented to detect the security incidents originating from the external network. They are also mainly oriented towards standard TCP/IP services: e-mail, WWW, FTP servers, SSH and other distant connection tools. Most valuable classified information, however, usually is stored in the databases of the organization and security of the database server is crucial.

2.0 ACTIVE TOOLS FOR DATABASE SECURITY

Access control in the database level is first and very important line of defense. Very often databases design and programming faults leads to the situation when the user without special authorization can get access to the confidential information which should not be accessible for this particular user. Most of the databases operate using client/server scheme. Applications residing on the remote servers connect to the database server using standard network protocols as TCP/IP, IPX/SPX, etc and special database server's protocol for connections functioning at the higher level. Often not only organization's applications servers but also workstations use direct connections to the database server. As a result it is very complicated or even impossible to ensure network segmentation minimizing possibilities to connect to the database server without limiting functional capabilities of the whole system. Due to this the measures to ensure database security should take into account also threats originating from local as well as external networks.

Network segmentation, network traffic filtration, restrictions of database tables access are passive tools of security. These tools often fail to detect unauthorized access attempts early. To achieve the higher level of security enabling to detect intrusion at the initial stage and to identify the intruder the active security tools such as database IDS and honeypots are to be developed and deployed. Two types of the active database security tools can be introduced depending on the level at which they are operating:

- Honeypot system emulating database server functions at the network level.

- Honeypot type database level IDS module. The aim of such module is to detect any suspicious activities of any user and to respond by calling specifically defined external procedures.

The two types of database IDS systems are aimed at detection of the different type of intruders. In the first case of network level database IDS the intruder would be detected during the search for database servers in the organization's network and attempts to read basic database listeners information such as database server version, databases names, etc.

Database level IDS module is aimed to detect the intruders who already have user name and password for access to the database. The module is aimed to react to enquires of database tables not used by real applications but loaded to the database to attract intruders.

In both cases realization of the active database security tools should include the following subsystems:

- Analysis subsystem. This is subsystem responsible for the detection of any illegal activities in the database.

- Reaction subsystem. This subsystem is responsible for all actions undertaken after the detection of such activities. The action can include reporting to the network operating or central IDS system via SNMP traps, etc, or to security administrator via e-mail, pager, etc.

Below we analyze the simplest ways to develop the active type tools to be used with the one of the most widespread database systems namely Oracle database.

3.0 DATABASE EMULATION IN NETWORK LEVEL

An effective tool to monitor illegal activities would be a honeypot type system emulating database functions at the network level. A system of this type is oriented for monitoring of the external attacks. One possibility to develop such system is to create artificial Oracle database “Listener” enabling detection of all unauthorized activities and defining IP addresses and user names used in these connection attempts. As a response the system administrator can be informed or further connections from particular IP addresses can be denied automatically.

The system should include two subsystems:

- Subsystem imitating connections to database

- Subsystem responsible for monitoring and response.

The first subsystem is the most important and can be created using various means. The direct way is to write the software module emulating real connections according to particular database protocol. However this way can be very time and recourses consuming if the protocol is complicated or proprietary and/or encoded. A simple way around this problem is to use freely available software tools developed for protocol testing and emulation. In this case emulation is not completely corresponding to the original protocol, but it can be sufficient if full scale dialog with the database is not needed and the purpose is only to detect an attempt of connection.

As an example we have exploited the software package SPIKE. Using this package it is possible to create communication rules emulating real connections. The corresponding example script can be of the following form:

```
s_block_start("firstpacket");
s_binary_block_size_halfword_bigendian_variable("firstpacket");
s_int_variable(0x0000,5);
s_binary("01 00 00 00 01 38 01 2c 00 00 08 00");
s_int_variable(0x7fff,5);
s_binary(" 86 0e 00 00 01 00 ");
s_binary_block_size_halfword_bigendian_variable("connectpacket");
s_binary(" 00 3a 00 00");
```

```
s_int_variable(0x0200,5);
s_binary("01 01 00 00 00 00 00 00  00 00 00 00 0b 2c 00 00 ");
s_binary(" 3b fb 00 00 00 00 00 00  00 00");
s_block_start("connectpacket");
s_string_variable("");
s_string("(");
s_string_variable("DESCRIPTION");
s_string_variable("=");
s_string("(");
s_string_variable("CONNECT_DATA");
s_string_variable("=");
s_string("(");
s_string_variable("SID");
s_string_variable("=");
s_string("*");
s_string(")");
s_string_variable("");
……..
```

Having script of this or more sophisticated type the SPIKE component “generic_listen_tep” is to be exploited to create a tool listening for the particular port and acting according to the rules established by the script.

The second subsystems can be realized using simple sniffers running on the same machine. Software packages tepdump, ettercap or others can be exploited for monitoring and recording. The standard tools are to be used for collection and analysis of the recorded data. The simplest way is to inform the system administration via e-mail about all connection attempts. Automated response tools also can be easily created.

4.0 HONEYPOT TYPE IDS MODULE IN DATABASE LEVEL

The simplest case of the honeypot type security tools for databases are honeytokens described by Lance Spitzner [Spitzner 2003b]. Bogus information of any type can be loaded into the database just to attract the attention of the intruders. Another type of the trap for intruders can be table without any real information but with attracting name. The aim of the honeypot type IDS module in the database level is to create a system reacting in real time to any attempt to access such table or tables. The simple way to detect access attempts is to use sniffers with select settings like in the case of network level system. However, in the case of encoded connections this way is impossible and it is necessary to use internal capabilities of the particular database. The realization in this case depends significantly on the particular database and can vary from version to version. As an example below we analyze schematically the possibility to create such system for Oracle database server shown in Fig. 1.

In the Oracle database audit is turned on for writing to the database by adding the following line to the file *init.ora*:

Audit_trail = db

Every time a user attempts anything in the database with the enabled audit the Oracle kernel checks to find out if an audit record should be created or updated (in the case or a session record) and generates the

record in a table owned by the SYS user called AUD$. This table is, by default, located in the SYSTEM tablespace. Writing to this table can cause problems with potential denial of service attacks. If the SYSTEM tablespace fills up, the database will hang.

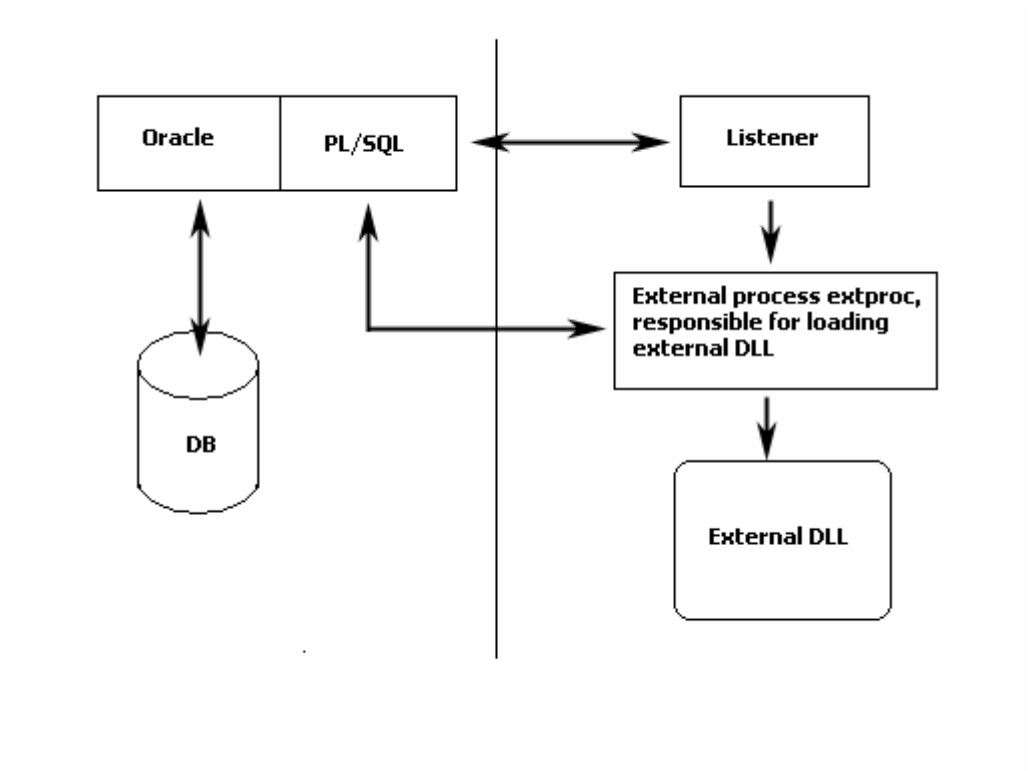

Figure 1: Scheme of the Oracle database server's subsystems

The following command will activate an audit of the access attempts for the table honeypot_table_name:

SQL> audit select on honeypot_table_name;
Audit succeeded.

Audit records can be checked in a different ways: by selecting from SYS.AUD$ where the raw audit trail is recorded or by selecting from dba_audit_trail giving a DBA view of the raw audit trail. So to realize the analysis subsystem for the honeypot type database IDS system PL/SQL code should be written checking the SYS.AUD$ table and analyzing it for the attempts to read data from the honeypot table in the database.

In the recent version 9i of the Oracle database there are possibilities for the fine grained auditing. In this case it is possible to realize honeypot type database IDS system without checking accesses to the honeypot table, but only that providing values of the particular columns from the table:

DBMS_FGA.ADD_POLICY(
object_schema => 'cs',
object_name => 'account',
policy_name => 'chk_account',

audit_condition => 'dept = "CS" ',
audit_column => 'honeypot_column');

Either of the following SQL statements will cause the database to log an audit event record:

SELECT name, honeypot_column FROM cs.account WHERE dept = 'CS';

or

*SELECT * FROM cs.account;*

In order to realize reaction subsystem, it is possible to use an event handler to generate an alert of an administrator that a honeypot system is activated, or to create external procedure saved in DLL file and registered PL/SQL. During the operation PL/SQL loads dynamically DLL-library and calls in external procedures as PL/SQL subprograms.

5.0 CONCLUSIONS

It is demonstrated that honeypot type IDS systems for enhancement of the database security can be developed using freely available software tools. Two suggested concept systems are based on the basic honeypot technology principle to exploit information system recourse without any real value except for attracting and detecting illegal activities. For the databases with large number of records and users it can be a very effective way for early intrusion detection. It should be pointed out, however, that honeypot type tools having many advantages such as simplicity of the concept, ability to detect local threats could not replace other security tools.

6.0 REFERENCES

[Cheswick 1991] B.Cheswick, “An evening with Berferd in which a Cracker is Lured, Endured, and Studied”, http://www.tracking-hackers.com/papers/berferd.pdf , 1991.

[Rainys 2003] D. Rainys, A. Bielko, A. Čenys, **“**Enhancing IDS - Honeypot Systems”, 4th Conference on Informatics and Information Technology, Macedonia: 2003 in press.

[Schneier 2000] B. Schneier, J. Wiley, “Secrets And Lies: Digital Security in a Networked World”, John Wiley & Sons, 2000.

[Spitzner 2003a] L. Spitzner, “Honeypots: Definitions and Value of Honeypots”, http://www.tracking-hackers.com/papers/honeypots.html, 2003.

[Spitzner 2003b] L. Spitzner, “Honeytokens: The Other Honeypot”, http://www.securityfocus.com/infocus/1713, 2003.

[Stoll 1990] C. Stoll, “Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage”, Pocket Books, New York, 1990.