Audit System at CESNET-CERTS

PAVEL VACHEK
CESNET-CERTS
CESNET, Association of Legal Entities
Zikova 4, 160 00 Prague 6
THE CZECH REPUBLIC
Pavel.Vachek@cesnet.cz http://www.ces.net

Abstract: CESNET-CERTS, the Computer Security Incident Response Team of the CESNET Association in Prague, Czech Republic, uses several security tools based on freely licensed programs. One of them is an enhanced system for host security auditing which is based on *Nessus* and runs on a PC server under Linux. An e-mail interface developed in-house allows users to perform basic host security audits simply and securely without having to study the extensive *Nessus* manuals and/or installing the *Nessus* server. The use of a similar open-source program *OpenVAS* within the Audit System is also considered.

Key-Words: Cesnet Association, Host security audit, *Nessus, OpenVas*, Pc, Linux, E-mail interface

1 Introduction

Most academic networks consist of a number of local area networks with computers running several operating systems, especially some versions of MS Windows, MacOS, Unix and Linux distributions. Keeping heterogenous networks safe requires specially trained staff and experience shows that removing infections from compromised hosts in large heterogenous networks can become a nightmare.

Fortunately, there are some tools available to the security-minded administrators and advanced users which can make their tasks easier. The CESNET-CERTS security team adapted some of them to their needs and has been using them successfully for several years.

One of them is the OTRS (Open-source Ticket Request System) which makes the handling of Incident Reports for our security staff much simpler. The OTRS as modified and used by CESNET-CERTS is described in [1].

Another useful security service developed by the CESNET-CERTS is the CESNET Intrusion Detection System [2] which is based on the *LaBrea* program [3]. *LaBrea* intercepts and tarpits all connection attempts caused by malware or by hackers. Should CESNET IDS find out that the attack originated from some part of the CESNET2 network, it sends appropriate e-mail notice to administrators of respective originating network. Connection attempts originating outside CESNET2 are reported to the DSHIELD distributed security project [4]. The CESNET IDS is quite easy to set up, gives no false positives and requires very little maintenance (if any).

Yet another security service offered by the CESNET-CERTS is the CESNET Audit system [5] which is the main topic of this paper.

Host security improves significantly if authorized administrators can check security of their machines whenever necessary, e.g., before and after applying security patches, and any time the machine seems to behave strangely. Making this security check available for almost everybody has been the goal of the CESNET Audit project.

Nessus is a widely known and highly valued host vulnerability scanner, originally distributed under the GPL license. *Nessus* ranks high in security tool surveys [6]. It is used by more than 75 thousand organizations worldwide. Its description can be found in [7], [8] and [9]. Renaud Deraison, the author of *Nessus*, is one of the founders of the Tenable Network Security company [10] which currently handles the program maintenance and development.

At the time of writing this paper, the following *Nessus* versions are available:

- *Nessus* version 2.2.11 is an old version which is still distributed in source code under the GPL license. Tenable does not support this version any more.
- *Nessus* 3.2.1 has been until recently the latest version. Otherwise, all features of the following version 4.0.1 apply.
- *Nessus* 4.0.1, currently the latest version, is being offered free of charge but its license has been closed and source code is not

available. *Nessus* versions 3 and 4 are distributed only in binary form for Linux, FreeBSD, Solaris, MacOS, and MS Windows/Vista.

Nessus proper needs external security plugins to perform security audits. Each plugin specifies one security test to be performed. Some 30 thousand test plugins exist in August 2009. Home users may run a part of these plugins free of charge using the Plugin HomeFeed. However, the full set of security plugins for any business use is available using the Plugin ProfessionalFeed [11] which costs 1200 USD/year.

Nessus operates using the client-server mode. Both command-line and GUI-based clients exist. Installing *Nessus* server and clients is not difficult but learning to fully master all their capabilities takes quite a lot of time. The e-mail interface described below allows users to run basic *Nessus* audits simply and securely without having to install server and/or clients and without having to learn all their features, simply by sending an appropriate PGP-signed e-mail message to the Audit server.

2 Running *Nessus* Audit Using E-mail Interface

Consider a user *test@example.cz* who needs to check security of his host 192.168.123.45 using the CESNET Audit e-mail interface. E-mail address of the *Nessus* auditing server and client is *audit@audit.example.net*. This user can choose from the following options:

- run only safe tests (some vulnerabilities may not be detected but neither the operating system nor running services will be affected negatively), or
- run all security tests including those potentially dangerous (more vulnerabilities may be detected but some services or the operating system might crash)

- check responses of default TCP/UDP ports only (test runs faster but some running services may not be detected), or
- check responses of all TCP ports (this test takes a longer time but it should discover all running services)

- results of security scan may be returned in various formats, e.g., text, html, xml

- security audit can start after a delay specified in hours and minutes.

Our user has chosen to run all security tests, check responses of all TCP ports, select a delay of 2 minutes and requires the results in HTML format. So he sends the following PGP-signed e-mail:

```
From: test@example.cz
Date: Fri, 21 Aug 2009 14:50:21 +0200
To: audit@audit.example.net
Subject: AUDIT 20090821

target: 192.168.123.45
config: full
format: html
tcpscan: all
delay: 2 m
end
```

Seven seconds later, the auditing server responds with the following PGP-signed reply:

```
From: audit@audit.example.net
Date: Fri, 21 Aug 2009 12:50:28 +0000
To: test@example.cz
Subject: Re: AUDIT 20090821

AUDIT v. 53 (4.3.2009) start: Fri Aug
21 12:50:22 2009 GMT

Audit configuration requested in
Qfile:
CONFIG:  full
FORMAT:  html
TCPSCAN: all
TARGET:  192.168.123.45
DELAY: 120 seconds (-> Fri Aug  21
12:52:26 2009 GMT)

Audit request
`sssssssss.ttttttttt.rrrr' queued.
```

The following important data can be found in this mail:

- security audit should start at 12:52:26 GMT
- File *Qfile* *`sssssssss.ttttttttt.rrrr'* containing appropriate CONFIG, FORMAT, TCPSCAN, and TARGET parameters has been queued.

This queue is inspected every minute. At appropriate time, *Nessus* is started using parameters found in the oldest queued file *Qfile*. After the security audit has terminated, a PGP-signed mail similar to the following is sent:

```
From: audit@audit.example.net
Date: Fri, 21 Aug 2009 13:00:03 +0000
To: test@example.cz
Subject: Re: AUDIT 20090821

Hello,
please find the results of your Nessus
security audit in the attached file.
```

```
Audit request queued on Fri Aug 21
12:50:26 2009 GMT.
Launch scheduled for Fri Aug 21
12:52:26 2009 GMT.
Launching audit on Fri Aug  21 12:52:01
2009 - 25 second(s) ahead of schedule.

Parameters supplied in
`sssssssss.ttttttttt.rrrr':
        CONFIG:  full
        FORMAT:  html
        TCPSCAN: all
        TARGET:  192.168.123.45

Best regards,
                the CESNET AUDIT robot.
```

Attached file *RESFILE.html* contains the audit results. The following Fig.1 shows an example summary of detected security holes (important security problems), security warnings (possible problems) and security notes (informational messages):

```
Address        Port/Service    Issue

192.168.123.45 domain (53/tcp)  Security hole

192.168.123.45 smtp (25/tcp)    Security notes

192.168.123.45 unknown (33/tcp) Security notes

192.168.123.45 general/tcp      Security notes

192.168.123.45 ntp (123/udp)    Security notes

192.168.123.45 domain (53/udp)  Security warning
```

Fig.1: First part of security audit result

Thanks to the "TCPSCAN: all" parameter, *Nessus* was able to detect as well as to classify correctly an SSH server running on a non-standard TCP port 33 – see Fig.2:

```
Informational unknown (33/tcp)

Synopsis : An SSH server is listening on this
port.

Description : It is possible to obtain
information about the remote SSH server by
sending an empty authentication request.

Risk factor : None

Plugin output : SSH version : SSH-2.0-
OpenSSH_5.2

SSH supported authentication :
publickey,password

Nessus ID : 10267
```

Fig.2: SSH server running on a non-standard port

The following Fig.3 shows another part of security audit result with detailed description of detected problem as well as its importance, link to further information if possible, *Nessus* ID and perhaps also a suggested problem solution:

```
Warning  domain (53/udp)

Synopsis: Remote DNS server is vulnerable to
cache snooping attacks.

Description: The remote DNS server answers to
queries for third-party domains which do not
have the recursion bit set. This may allow a
remote attacker to determine which domains have
recently been resolved via this name server,
and therefore which hosts have been recently
visited.  For instance, if an attacker was
interested in whether your company utilizes the
online services of a particular financial
institution, they would be able to use this
attack to build a statistical model regarding
company usage of aforementioned financial
institution. Of course, the attack can also be
used to find B2B partners, web-surfing
patterns, external mail servers, and more...

See also: For a much more detailed discussion
of the potential risks of allowing DNS cache
information to be queried anonymously, please
see:
http://www.rootsecure.net/content/downloads/pdf
/dns_cache_snooping.pdf

Risk factor: Medium / CVSS Base Score : 5.0
(CVSS2#AV:N/AC:L/Au:N/C:P/I:N/A:N)

Nessus ID : 12217
```

Fig.3: Another possible part of security audit result

The following Fig.4 summarizes important internal parameters of the auditing server:

```
Information about this scan :

Nessus version : 4.0.1
Plugin feed version : 200908210123
Type of plugin feed : ProfessionalFeed (Direct)
Scanner IP : 192.168.123.90
Port scanner(s) : nessus_tcp_scanner
Port range : 0-65535
Thorough tests : yes
Experimental tests : no
Paranoia level : 1
Report Verbosity : 1
Safe checks : no
Optimize the test : no
CGI scanning : enabled
Web application tests : disabled
Max hosts : 10
Max checks : 10
Recv timeout : 5
Backports : None
Scan Start Date : 2009/8/21 12:52
Scan duration : 410 sec
```

Fig.4: Summary of server internal parameters

3 Auditing server configuration

Linux on the auditing server runs in runlevel 3, i.e., without any graphic user interface. Both the *Nessus* server and command-line client are installed on this machine. However, the *Nessus* GUI client must be launched at least once on an accessible machine to generate the appropriate client configuration file; afterwards, this resulting configuration file is to be copied to the auditing server.

The auditing server will run as few network services as possible - e.g., SSH (*OpenSSH*), SMTP (*Postfix*) and NTP (*ntpd*) only. Firewall will let these services communicate with selected IP addresses only to protect the server from Denial of Service and spam attacks.

3.1 Installing Nessus

To access the up-to-date *Nessus* test plugins for commercial purposes, buying the ProfessionalFeed license for an enterprise network is necessary. Tenable sends us our activation code and URL to log into the Customer Support Portal. Mailed instructions should be followed to activate the ProfessionalFeed. We log into the auditing server using the *root* username. Using *lynx* or a similar browser, we download *Nessus x.y.z for Linux* and *MD5.asc* files from [12].

Note: The following paths/filenames are valid for the OpenSUSE 10.x. Some data may slightly differ for other Linux distributions.

Nessus-x.y.z-suse10.0.i586.rpm checksum should be checked against the *MD5.asc* file. Both *Nessus* server and command-line client are installed using

$ rpm -Uvh Nessus-x.y.z-suse10.0.i586.rpm

Nessus server configuration file is created at address */opt/nessus/etc/nessus/nessusd.conf* (hereinafter, *nessusd.conf*). The following will be displayed:

```
Create a nessusd certificate using
/opt/nessus/sbin/nessus-mkcert
Add a nessusd user use
/opt/nessus/sbin/nessus-adduser
Start the Nessus daemon (nessusd) use
by typing /etc/rc.d/nessusd start
Start the Nessus client (nessus) use
/opt/nessus/bin/nessus
Register your Nessus scanner at
http://www.nessus.org/register/ to
obtain all the newest plugins
```

3.2 Installing and configuring *nessusd*

X.509 certificate is created using

$ /opt/*nessus*/sbin/nessus-mkcert

Command

$ /opt/*nessus*/sbin/nessus-adduser

creates *nessusd* account for user *root* (authentication = *password*, password = e.g., *Secret*). For simplicity, rules may be empty (press *CTRL-D* only); this means that user *root* can audit any Internet machine. Individual *Nessus* users accessing *Nessus* through the E-mail interface will be limited using *PERM* directory as described later.

The following or similar data should be set in the *nessusd.conf* file:

max_hosts = 10
max_checks = 10
log_whole_attack = no
dumpfile = /dev/null
port_range = 0-65535
optimize_test = no
checks_read_timeout = 5
safe_checks = no
auto_enable_dependencies = yes
use_mac_addr = no
listen_address = 127.0.0.1 (if access from external *Nessus* clients is to be allowed, IP address of external interface should be given here)

where:

max_hosts ... maximum number of hosts to test at the same time
max_checks ... maximum numer of plugins to run concurrently on each host
log_whole_attack ... log data on each plugin launched
dumpfile ... plugin error messages will be logged here
port_range ... TCP and UDP ports to be tested (default: some 9000 ports according to */opt/nessus/var/nessus/services.txt*)
optimize_test ... banners of remote services are trustworthy
checks_read_timeout ... 5 seconds for LANs
safe_checks ... only safe tests should run
auto_enable_dependencies ... all necessary plugins are launched automatically
use_mac_addr ... tested machines are identified using Ethernet addresses
listen_address ... *nessusd* listens on this interface IP address

Using the following command,

$ /etc/rc.d/*nessusd* start

Nessusd is started and all plugins are downloaded

from the *Nessus* plugin server. Command

$ ps -ef | grep *nessusd*

should display the following string if *nesssusd* is listening:

```
nessusd: waiting for incoming
connections
```

3.3 Installing and configuring *Nessus* client

The following command

$ *nessus* -x -T text -q localhost 1241 root Secret *targets results* &

checks both *Nessus* client and server operation using the command-line interface. In addition, a "safe" security audit of our auditing server is performed from within.

Parameters are as follows:

-x ... X.509 certificate is not checked
-T text ... result file in ASCII text format
-q ... batch mode
localhost ... *nessusd* server address
1241 ... *nessusd* listens on this TCP port
root ... *nessus* username
Secret ... this user's password
targets ... file containing a list of machines to be tested *(localhost)*
results ... name of audit result file

The following notice will be displayed:

```
The plugins that have the ability to
crash remote services or hosts have
been disabled. You should activate them
if you want your security audit to be
complete
```

After the audit has terminated, the *results* file should be checked and any security problems of the auditing server should be fixed.

To allow all plugins including those potentially dangerous, a GUI *Nessus* client must run on some other accessible machine – probably on a workstation on which GUI is running. *NessusClient-x.y.z-suse10.3.i586.rpm* together with *MD5.asc* should be downloaded, checked and installed on this workstation. Command

$ *nessus* &

will display the *Nessus* GUI interface. All appropriate settings should be selected, especially the ENABLE ALL option and address of machine to be tested. Security audit is launched by clicking on the START THE SCAN button. Audit results should again be inspected and any problems concerning this workstation should be fixed. After the audit, *Nessus* client configuration file */root/.nessusrc* is created which allows running all test plugins. It is copied to file *.NESSUSRC* and moved to the auditing server.

To allow automatic plugin download, the following should be set in *nessusd.conf* on the auditing machine*:*

# Automatic plugins updates - if enabled and *nessus* is registered, then
# fetch the newest plugins from plugins.*nessus*.org automatically
auto_update = yes
# Number of hours to wait between two updates
auto_update_delay = 24
# Should we purge the plugin db at each update ? (slower)
purge_plugin_db = no

4 E-mail interface programs

There are two e-mail interface programs: *auditXX* and *procXX*. Both are written in the Perl language. Both the Perl interpreter and modules from [13] are needed.

AuditXX runs under the *audit* username. Its main tasks are:

- receiving e-mail from local SMTP server
- checking for PGP signature presence and validity
- checking for correct length of received messages (messages with excessive length are silently ignored but reported to the auditing server administrator; most likely cause for this may be spam or even our own bounced outgoing messages)
- sending unsigned e-mail containing a short summary of allowed commands and their possible parameters in response to received unsigned e-mail
- extracting required audit parameters (CONFIG, FORMAT, DELAY, etc.) from received valid mail
- building and queuing appropriate batch file *Qfile*
- sending PGP-signed response (especially positive acknowledgement message) to received signed e-mail

Program *procXX* runs under the *root* username. Its main tasks are:

- inspecting the queued batch files every minute

- checking for existence of lock files. These files indicate whether another security audit is running or whether the *Nessus* client could not connect to the *Nessus* server; the latter might happen, e.g., during *Nessus* plugin feed retrieval and this would be no cause for alarm
- launching security audit using appropriate parameters at proper time (i.e., if no other audit is running, security audit should start with a tolerance of +/- 30 seconds)
- returning the security audit results in a PGP-signed mail to the originating e-mail address.

Installation instructions of the e-mail interface on server *audit.example.net* follow. Username *audit* is created to communicate using e-mail address *audit@audit.example.net*. *Nessus* server and command-line client, *Postfix, sshd, ntpd*, and *procXX* run under the *root* username.

4.1 E-mail system configuration

Audit.example.net is the only host accepting e-mail for the auditing server. Zone file for the *example.net* domain contains a single MX record for the *audit* host (no secondary backup MX servers are used to avoid possible spam):

audit IN MX 10 audit

SMTP server *Postfix* is configured both for accepting and sending e-mail. E-mail delivered to *audit@audit.example.net* is forwarded to the *auditXX* standard input using the *.forward* file:

\audit, "|/home/audit/bin/auditXX"

4.2 PGP and E-mail system operation check

PGP public and private keys for *audit@audit.example.net* are generated using *gpg*. Resulting private key is also copied into the *root* keyring. PGP-signed test e-mail containing simple text and an attached file can be sent using our program *MailtestXX*. PGP KeyId, passphrase, sending and receiving e-mail address must be inserted into the source file:

```
$from = 'audit@audit.example.net';        # server e-mail address
$to =  'test@example.cz, test2@example.cz'; # receiving address(es)
$keyid = '037ADEADBEEF1256';        # for audit@audit.example.net
$pass = 'AuditPwd';      # Passphrase for PGP key
```

Launch the program as follows:

$ MailtestXX *file*

(*file* is any test file to be sent, e.g., in HTML format). After successful termination, the following should be displayed:

```
xx input lines, yyy characters
read ... sent.
```

Recipients should check for correct mail delivery and readability. *MailtestXX* proper might display some errors; system log file (e.g., */var/log/allmessages*) might contain some as well.

4.3 Installing and testing *auditXX*

AuditXX (currently, *audit53*) should be copied to */home/audit/bin*. Appropriate PGP KeyId, passphrase and server e-mail address must be inserted into the *auditXX* source file:

```
$from = 'audit@audit.example.net';        # server E-mail address
$keyid = '037ADEADBEEF1256';        # for audit@audit.example.net
$pass = 'AuditPwd';                 # Passphrase
```

Under OpenSUSE 10.x, the following was necessary to allow access to Perl modules:

$ chmod -R 755 /usr/lib/perl5/site_perl/5.8.8/*

To test the *auditXX* operation, a short (or possibly empty) unsigned e-mail should be sent to the auditing server. *AuditXX* should respond with an e-mail similar to the following Fig.5:

```
From: audit@audit.example.net
Date: Fri, 21 Mar 2009 14:21:58 +0000
To: test@example.cz
Subject: Re: TEST 1

AUDIT v. 53 (4.3.2009) start: Fri Aug 21 14:21:58 2009 GMT
The following UNSIGNED text has been received:
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
TEST 1
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

Please resend using a correct PGP key.
This may also help:

CESNET AUDIT v. 53 (4.3.2009) HELP:

CONFIG:  previous | full | safe
FORMAT:  previous|nbe|html|html_graph|text|xml|old-xml|tex|nsr
TARGET:   previous | list of IP or domain addresses to be tested
TCPSCAN: previous | all | default
```

```
DELAY:   hh H mm M | mm M hh H | hh H |
mm M
VERBOSE  (for debugging purposes)
END      (end of input data)
```

Fig.5: Response to a short unsigned e-mail

Details concerning any encountered errors and their causes can be found in file */home/audit/DEBUG* or in the *Postfix* error log.

In the next step, correct receipt of PGP-signed messages must be tested. Let us suppose that:

- user *test@example.cz* may run security audit of 10.1.2.3, 10.2.3.5
- user *user@example.com* may run security audit of 192.168.100.200..204

We shall call a PGP signing party with these users, sign their public keys and insert them into *audit* and *root* user keyrings. All data necessary for the operation of security audits will be found within the */ home/audit/AUDIT* directory:

$ ls -R /home/audit/AUDIT/

/home/audit/AUDIT/: (created on 1st *auditXX* run)
test@example.cz/ (created on 1st receipt of signed e-mail from *user*)
user@example.com/ (created on 1st receipt of signed e-mail from *test*)
PERM/ (created on 1st receipt of signed e-mail)
PROC/ (created manually by *root*)
QUEUE/ (created on 1st *auditXX* run)

/home/audit/AUDIT/test@example.cz:
CFG CFG.sav (created by *auditXX*)

/home/audit/AUDIT/user@example.com:
CFG CFG.sav (created by *auditXX*)

/home/audit/AUDIT/PERM:
test@example.cz user@example.com (files created manually by audit administrator)

/home/audit/AUDIT/PROC:
NESSUSFAIL (lock file created and deleted by *procXX* if necessary)
NESSUSLOCK (lock file created and deleted by *procXX* if necessary)
RESFILE (result file created by *procXX*)
TARGFILE (list of tested hosts created by *procXX*)

/home/audit/AUDIT/QUEUE:
Queued files are inserted by *auditXX* and removed by *procXX*. Top (the oldest) file is the first one to be tested. Filename format is:
ssssssssss.rrrr or *ssssssssss.ttttttttt.rrrr*

- sssssssss ... requested start time of security audit in Unix epoch time
- *ttttttttt* ... time of sending this request if delay specified in Unix epoch time
- rrrr ... pseudorandom number to make filenames unique.

Files *CFG* and *CFG.sav* contain the CONFIG, FORMAT, TARGET, and TCPSCAN parameters of the last successful security audit. For repeated audits, parameter values missing in e-mail are replaced by the saved values in *CFG*. However, at least one parameter must always be given so that an empty signed e-mail sent inadvertently does not launch scan by mistake.

Files in the */home/audit/AUDIT/PERM* directory contain a list of IP addresses which respective users are allowed to test. Their format is simple:

```
# test@example.cz may test 10.1.2.3, 10.2.3.5:
10.1.2.3
10.2.3.5

# user@example.com - 192.168.100.200..204
# Several addresses (space or TAB separated) may
# be given on each line
192.168.100.200 192.168.100.201 192.168.100.202
192.168.100.203 192.168.100.204

# CIDR notation may also be used:
# another user might be allowed to check whole
# `class B' network:
172.31.0.0/16
```

Administrator of the auditing server creates these files manually according to the users' requests. He knows these users personally and he can check if they have a right to audit the respective machines. This is also the reason why auditing requests from external users whom the audit administrator does not know personally are rejected; official network administrators of University and/or CESNET customer networks are allowed audits only of their own respective networks as stated in the CESNET Audit System FAQ [14].

Full operation of the e-mail interface can be tested easily as described in Chapter 2 above. Complete debugging information of audit request processing can be found in file */home/audit/DEBUG*; any important error message is sent to the user by e-mail in the format of his original e-mail, i.e., unsigned, inline (clearsigned) PGP, or PGP/MIME. Audit server administrator is also informed using SMS or e-mail message.

4.4 Installing and checking *procXX* operation

This is even simpler. *ProcXX* (current version = 66) should be copied into */root/bin*. As in the case of *auditXX*, the following must be inserted into the source file:

```
$from = 'audit@audit.example.net ' ;      # server e-mail address
$keyid = '037ADEADBEEF1256';              # audit@audit.example.net
$pass = 'AuditPwd';                       # Passphrase
```

ProcXX is launched by *cron* every minute; therefore, the following two lines should be put into *crontab*:

```
# Checking the nessus audit queue every minute:
* * * * * /root/bin/procXX
```

On finding any queued files within */home/audit/AUDIT/QUEUE/*, *procXX* checks the filename of the first, i.e., oldest file. Its leftmost part gives the Unix epoch time when the security audit is to be launched. At appropriate time and if no other instance of *Nessus* is running, a new *Nessus* client is launched with appropriate parameters taken from this *Qfile*. At the same time, *NESSUSLOCK* file is created as described in the following paragraph. After *Nessus* terminates, *procXX* e-mails the security audit results to the appropriate user address and deletes the *Qfile*. Afterwards, the first following *Qfile* (if any) is checked and the above process is repeated immediately; otherwise, *procXX* terminates and *NESSUSLOCK* is deleted.

Lock file */home/audit/AUDIT/PROC/NESSUSLOCK* is created when *Nessus* is launched. Existence of this file indicates that no other security audit may be started. Immediately after its start, *procXX* checks if file *NESSUSLOCK* exists. Should *procXX* learn that this is so and that *NESSUSLOCK* is older than some preselected value (several hours), it sends warning SMS and/or e-mail messages at regular intervals to the audit server administrator.

Another lock file */home/audit/AUDIT/PROC/NESSUSFAIL* is created if *Nessus* is launched but terminates with an error message. So far, this has been recorded in cases when download of *Nessus* Plugin Feed was underway and a new instance of *Nessus* was launched at the same time. This is taken as an error which may yet be recovered soon: *NESSUSFAIL* indicates that launch of *Nessus* using appropriate Qfile may be reattempted several times. To be able to distinguish the current Qfile which is to be reused, its filename is prefixed with character `C'. Server administrator will be informed about every such repeated attempt as well as about the final outcome.

Complete log of *procXX* operation can be found in */root/DEBUG*; again, any important error message is e-mailed both to the user and audit server administrator.

5 *OpenVAS*

As mentioned in Chapter 2 above, *Nessus* version 2 was released under the GPL license but the following versions were released using a closed license in binary form only. In reaction to this, by the end of 2005, a team of developers launched the *OpenVAS* Project [15] which is a fork of *Nessus* version 2.2.5. The *OpenVAS* (Open Vulnerability Assessment System) attempts to provide both the *OpenVAS* and its Plugin Feed free of charge using GNU GPL.

Rough comparison of *Nessus* and *OpenVAS* features (end of August 2009) is as follows:

Latest version:
- *Nessus:* 4.0.1
- *OpenVAS:* 2.0.4

Approximate number of test plugins:
- *Nessus:* 30000
- *OpenVAS:* 13000

Server communicates on default port:
- *Nessus:* TCP 1241
- *OpenVAS:* TCP 9390

Command-line client:
- *Nessus:* part of server
- *OpenVAS:* GUI klient + "--disable-gtk"

Plugin Feed Enhanced Security:
- *Nessus:* none
- *OpenVAS:* PGP-signed plugins

Cost for Professional Users:
- *Nessus:* 1200 USD/year
- *OpenVAS:* none

The above data shows that *OpenVAS* looks very interesting, especially but not only with regard to its zero cost. Its main disadvantage seems to be the significantly lower number of *OpenVAS* test plugins compared to that of *Nessus.*

While collecting further data on *Nessus* and *OpenVAS*, an interesting document [16] was found. Its main conclusions are:

- *Nessus* discovered significantly more of all known vulnerabilities (mainly older ones)
- *OpenVAS* found significantly more CVE

vulnerabilities which were discovered during last two years (2008 and 2009).

- Overall, *OpenVAS* is also a reliable and trustworthy tool for security audit.

A diagram summarizing the results of this document is shown in the following Fig.6:

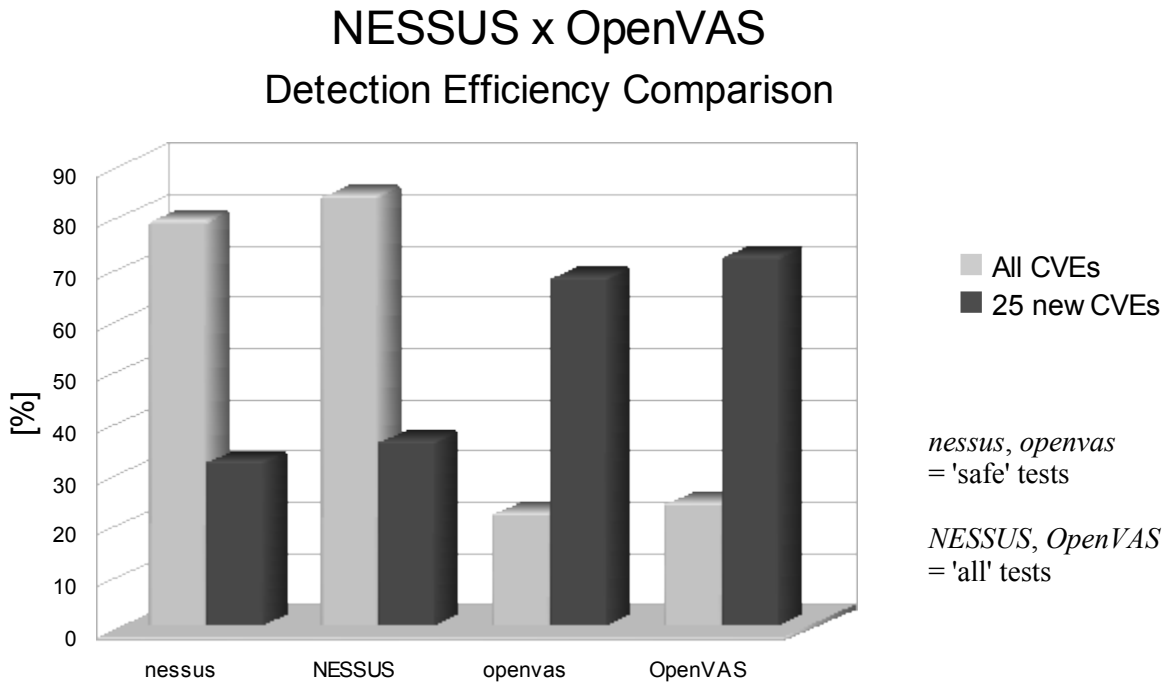

Fig.6: Comparison of *Nessus* and *OpenVAS* known vulnerability detection efficiency

6 Conclusion

CESNET Audit server in Prague has been operating successfully for several years on DELL PowerEdge SC1425 with CPU Intel Xeon 3.4 GHz, 1 GB RAM, 70 GB SCSI disc. Operating system Linux OpenSUSE 10.x.; system uses *Nessus* plugin ProfessionalFeed. System works best for auditing hosts in local area network where test packets do not cross any router interface; however, administrators of several Universities and CESNET2 customers use it successfully as well.

The e-mail interface described above does not use all *Nessus* features; however, it allows every host administrator to run basic security audit easily and without need for studying the extensive program documentation. Of course, installing this e-mail interface does not preclude advanced users from installing GUI clients and from using all advanced *Nessus* features.

To offer the CESNET Audit system users the advantages of both *Nessus* and *OpenVAS* security scanners, this author has decided to modify the Audit system to run also *OpenVAS* in addition to *Nessus* should the user wish so. So far, *OpenVAS* has been installed and tested under OpenSuSE 11.x without problems; security audits of both *Nessus* and *OpenVAS* on tested machines give identical results. In near future, CESNET Audit users should be able to run *OpenVAS* audit of their machines after the *Nessus* audit terminates simply by specifying "*OpenVAS*: yes" keyword in their first PGP-signed e-mail request sent..

7 Acknowledgement

I would like to thank the CESNET Association for supporting my work and also all beta testers of the CESNET Audit system for their valuable comments and suggestions. Special thanks go to my colleagues Andrea Kropáčová and Pavel Kácha.

This work has been supported by the research grant "Optical Network of National Research and Its New Applications" (MSM 6383917201) of the Ministry of Education of the Czech Republic.

References:

[1] Pavel Kácha, OTRS: Streamlining CSIRT Incident Management Workflow, Proceedings of the 13th WSEAS International Conference on COMPUTERS, Rodos (Rhodes) Island, Greece, July 23/25, 2009, pp. 121-126.

[2] Pavel Vachek, CESNET Intrusion Detection

System, Proceedings of the 6th WSEAS International Conference on INFORMATION SECURITY and PRIVACY (ISP '07), Puerto de la Cruz, Tenerife, Canary Islands, Spain, December 14-16, 2007, pp. 166-171.
[3] *http://labrea.sourceforge.net/*
[4] *http://www.dshield.org/*
[5] Pavel Vachek, CESNET Audit System, Proceedings of the 13th WSEAS International Conference on COMPUTERS, Rodos (Rhodes) Island, Greece, July 23/25, 2009, pp. 153-158.
[6] E.g., *http://sectools.org/vuln-scanners.html*
[7] *http://www.nessus.org/*
[8] Deraison, R. et al.: *Nessus* Network Auditing. Syngress Publishing, 2004, ISBN 1931836086
[9] Rogers, R. et al.: *Nessus* Network Auditing, Second Edition. Syngress Publishing, 2008, ISBN 1597492086
[10] *http://www.tenablesecurity.com/*
[11] *http://www.nessus.org/products/professional-feed/*
[12] *http://www.nessus.org/download/*
[13] *http://search.cpan.org*
[14] *https://www.cesnet.cz/skupiny/csirt/sluzby/audit.html*
[15] *http://www.openvas.org/*
[16] Laboratory for Systems and Signals, Faculty of Electrical Engineering and Computing, University of Zagreb: *Nessus/OpenVAS* Comparison Test. Version 1.02 2009-04-06. *http://security.lss.hr/novosti/Nessus_vs_OpenVAS_en.pdf*