

# **RetypingDante Application**

**for Z25.org**

**Confidential**

**Final**

Madison Gurkha B.V.

ing. Hans Van de Looy

Guido van Rooij, Msc.

Walter Belgers, Msc., CISSP, CISA

Version: 1.0 – 30th March 2009

## Statement of Confidentiality

This document contains sensitive and confidential information concerning vulnerabilities, as well as methods for exploiting these vulnerabilities. It is being provided to Z25.org as a deliverable of an consulting engagement. The sole purpose of this document is to provide Z25.org with the results of, and recommendations derived from, this consulting engagement.

Madison Gurkha B.V. recommends that special precautions be taken to protect the confidentiality of the information contained in this report. Each recipient agrees that, prior to reading this document, (s)he shall not redistribute or use the information contained herein and any other information regarding Madison Gurkha B.V. for any purpose other than those stated.

While Madison Gurkha B.V. is confident that the major security vulnerabilities of the target systems have been identified, there can be no assurance that an assessment of this nature will identify all possible security exposures. Additionally, the findings and recommendations presented in this document are based on the technologies and known threats as of the date of this report. As technologies and risks change over time, the vulnerabilities and recommendations associated with the targets may also change.

### Madison Gurkha B.V.

Vestdijk 9  
5611 CA Eindhoven

P.O. Box 2216  
5600 CE Eindhoven

**P:** +31 (0)402 377 990  
**F:** +31 (0)402 371 699  
**E:** [info@madison-gurkha.com](mailto:info@madison-gurkha.com)  
**I:** <http://www.madison-gurkha.com>

### Copyright © 2009 Madison Gurkha B.V.

All rights reserved. No part of this document may be reproduced, stored in a retrieval system, transmitted or utilized in any form or by any means, electronic, mechanical, photocopying, microfilm, internet or otherwise, without permission in writing from Madison Gurkha B.V. or Z25.org.

### Trademark

Madison Gurkha and the logo of Madison Gurkha are trademarks of Madison Gurkha B.V.. All other trademarks mentioned in this document are owned by their organisations.

# Document Management

## Reviewers

Name	Function	Date	Version
Walter Belgers	Principal Security Consultant	2009-03-13	0.1

Table 1: Reviewers

## Changes

Version	Date	Initials	Changes
0.1	2009-03-11	HL	Initiële versie
0.2	2009-03-16	HL	Na interne review
1.0	2009-03-23	HL	Finale versie

Table 2: History

# Contents

<b>1</b>	<b>Management summary</b>	<b>1</b>
1.1	Goal of the assessment	1
1.2	Security assessment	1
1.3	Most important findings	2
1.4	Most important recommendations	2
1.4.1	Strategical recommendations	2
1.4.2	Tactical recommendations	2
1.4.3	Operational recommendations	2
<b>2</b>	<b>Introduction</b>	<b>4</b>
2.1	Goal of the audit	4
2.2	Scope of the audit	4
2.3	Target systems and applications	5
2.4	Information in advance	5
2.5	Audit criteria	5
2.6	Classification of risks and points of interest	6
2.7	Dossier	6
<b>3</b>	<b>Reconnaissance</b>	<b>7</b>
3.1	Tools overview	7
3.2	WHOIS	7
3.3	Domain Name Service (DNS)	10
<b>4</b>	<b>Vulnerability scans</b>	<b>12</b>
4.1	SSH	12
4.2	SMTP	12
4.3	HTTP(S)	13
4.4	IMAP(S)	17
4.5	Conclusion	19
<b>5</b>	<b>Application audit</b>	<b>20</b>
5.1	Overview	20
5.2	Design, existence and working of the application	20
5.2.1	Initial source code checking	21
5.2.2	Update mechanism	21
5.2.3	Encryption of data	24
5.2.4	Container handling	25
5.2.5	Keylogging process	26
5.2.6	Open UDP port	26
<b>6</b>	<b>Security assessment</b>	<b>28</b>
<b>7</b>	<b>Overview of results</b>	<b>29</b>
7.1	High risks	29
7.2	Medium risks	29
7.3	Low risks	30
7.4	Points of interest	31
<b>A</b>	<b>Best practices</b>	<b>32</b>

---

A.1 Diversity . . . . .	32
A.2 Redundancy . . . . .	32
A.3 Isolation . . . . .	33
A.4 Safe defaults . . . . .	33
A.5 Input validation . . . . .	33
A.6 Completeness . . . . .	33
A.7 Restriction of privileges . . . . .	34
A.8 Function separation . . . . .	34
A.9 Compartments . . . . .	34
A.10 Open design . . . . .	34
A.11 Ergonomics . . . . .	34
A.12 Simplicity . . . . .	35
A.13 Protection of data . . . . .	35
A.14 Availability of data . . . . .	35
A.15 Protection of communication . . . . .	35
A.16 Additional protection of server systems . . . . .	35
<b>B Additional sources of information</b>	<b>37</b>
B.1 Websites . . . . .	37
B.2 Mailing-lists . . . . .	38
<b>C Used acronyms</b>	<b>39</b>

## List of Tables

1	Reviewers . . . . .	iii
2	History . . . . .	iii
2.1	Design, existence and working . . . . .	4
2.2	Target systems and applications . . . . .	5
2.3	Information in advance . . . . .	5
B.1	Overview interweb resources. . . . .	37
B.2	Overview mailinglist resources. . . . .	38

## List of listings

3.1	WHOIS retypingdante.com . . . . .	7
3.2	WHOIS 80.100.151.120 . . . . .	8
3.3	WHOIS Z25.org . . . . .	9
3.4	DNS information retypingdante.com . . . . .	10
3.5	DNS version information . . . . .	10
4.1	Apache webserver header . . . . .	13
4.2	SVN Entries . . . . .	13
4.3	HTTPS Certificate information . . . . .	14
4.4	HTTPS Ciphers . . . . .	15
4.5	HTTP protocol information . . . . .	16
4.6	TRACE method . . . . .	17
4.7	TRACE and TRACK filtering . . . . .	17
4.8	IMAPS Ciphers . . . . .	18
5.1	Update server response . . . . .	23

# Chapter 1

## Management summary

Z25.org has requested Madison Gurkha B.V. to assess the IT security of RetypingDante Application. This research started on March 9, and is performed by Guido van Rooij, Walter Belgers en Hans Van de Looy. This report is written directly after analysis of the results.

### 1.1 Goal of the assessment

In the offer with reference OFF-20090209.01-z25.org-CB-apps, the following description of the assignment (in Dutch) is included in the situations section:

*“Stichting z25.org (verder aangeduid als z25.org) is een ontwikkelinstelling van nieuwe media voor cultuur en innovatie. Zij wil nieuwe media innovatief toepassen binnen een culturele en kunstzinnige context. Z25.org wil experimenten uitvoeren waarbij nieuwe media bepalend zijn voor de betekenisvorming. Als ontwikkelinstelling richt zij zich vooral op de professionals en toptalenten welke werkzaam zijn in de nieuwe media. Z25.org stelt zich als doel om een bredere doelgroep aan te spreken dan het publiek met een interesse voor technologische toepassingen in een culturele context.*

*Retyping Dante is een door z25.org ontwikkeld media kunstwerk over internet cultuur en culturele ontwikkelingen rond web 2.0. Voor het Retyping Dante project heeft z25.org een applicatie geschreven die toetsaanslagen verzamelt en naar een centrale server stuurt. Dit brengt de nodige gevoeligheid met zich mee en daarom worden de verzamelde toetsaanslagen versleuteld met RSA naar de centrale server gestuurd. Nu wil z25.org graag, onder andere middels een source code review, weten of de Retyping Dante applicatie software juist beveiligd is. Z25.org heeft daarom behoefte aan het onafhankelijk vaststellen van de status van haar ICT beveiliging met betrekking tot de Retyping Dante applicatie.”*

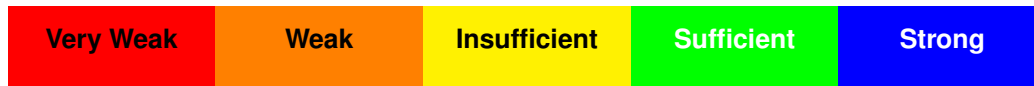
### 1.2 Security assessment

Based on the issues that were found during the investigation, the auditors have come to the following judgement of the security level:

Although there is one high level and one medium level risk identified in the `virgilius` software, we still consider the Retyping Dante application to be secure enough for its purpose.

The high risk is only a problem in combination with other, larger, problems in the clients IT environment. Since this environment is not completely under control of the creators of `virgilius`, we do urge the developers to implement a verification that the client is really communicating with the update server (possibly even using an encrypted channel), but we consider the risk of this problem by itself not high enough to set the security assessment to insufficient or below.

Most medium risks are found in other services running on another system in the z25.org environment (i.e. not the collecting server). These cause some threat to that server, but should have nothing to do with the Retyping Dante project or the communication between the clients and the server. The collecting server for the Retyping Dante project will only be used for this purpose (collecting typed characters), and is not shared. A Cisco router is used to forward port 9458/udp to the dedicated server for this project.



## 1.3 Most important findings

This report extensively describes the most important issues that were found during the investigation. In this section, we only describe the most important issues. A full overview of all issues that were found, is given in chapter 7 on page 29.

- The update procedure is vulnerable to an attack allowing code execution on the client;
- An attacker with access to the server can possibly reconstruct what is typed on specific clients;
- Several services on another server accessible via the provided target (speakeasy.z25.org) use the flawed SSLv2 protocol and allow weak ciphers for communication.

## 1.4 Most important recommendations

The auditors would like to bring the most important recommendations to the attention of the management, as well as the right level at which actions need to be taken to resolve the issues.

### 1.4.1 Strategical recommendations

Recommendations on a strategical level relate to the central information security policy of an organisation, the most important security goals and the relation with the environment. Such goals, in most cases, touch upon the center of the organisation, including the identity and which that, its main tasks.

- We have no strategical recommendations.

### 1.4.2 Tactical recommendations

The tactical information security policy tries to answers questions such as “what are we going to do and what means do we have, keeping in mind the goals?” This is an elaboration on the strategic policies.

- We advise to use a dedicated collecting server that runs no other services for this project; according to the remarks from Z25.org this is implemented in the current situation.

### 1.4.3 Operational recommendations

Operational information security policy deals with the question: “what are we actually doing?” It is not without reason this involves discussing procedures, guidelines, programs and actions. The operation policy relates to the execution of actual policies on short term, in most cases within the period of one year.

- We advise for the virgilius client to verify that it is talking to the correct update server (possibly using a secure channel for the complete information exchange);
- We advise for the virgilius client to only send full (containing 32 characters) containers to the collecting server.

# Chapter 2

## Introduction

Z25.org has given Madison Gurkha B.V. the assignment to investigate the IT security of Retyping-Dante Application. The audit started on March 9, and is performed by Guido van Rooij, Walter Belgers en Hans Van de Looy. Subsequently, this report was written.

### 2.1 Goal of the audit

The goal of a complete audit is to achieve a judgement on the *design*, confirming the *existence* and a check of the actual *working* of the system that is to be investigated. In short, we can give these terms the following definitions:

Term	Meaning	Example
Design	Things have been arranged	A design exists
Existence	Measures have been taken (meaning: “the prerequisites to implement the design are present”)	The design has been implemented
Working	Work is done in accordance with the agreement (design), using the available means (existence)	The measures (existence) work according to the design

Table 2.1: Design, existence and working

In many cases, not a complete investigation is requested, but only an investigation indicates the presence or lack of existence and working.

For this investigation, the goal was specified (translated from Dutch) as follows:

*“Z25.org wants to know if the Retyping Dante application software is securely implemented. To establish this trust level Madison Gurkha B.V. is asked to perform a source code review of the application, test the executable and perform a limited scan of the Information and Communication Technology (ICT) infrastructure used for the collection of data.”*

### 2.2 Scope of the audit

To achieve the goal mentioned above, Madison Gurkha B.V. has used the following approach:

*“The projectteam started by an investigation of the supplied targets (WHOIS information). We immediately discovered that the name of one of the targets (purgatorio.retypingdante.com) was (still) pointing to an IP-address that was out of scope. We discarded that name from the target list and performed the normal DNS related tests on the retypingdante.com DNS servers. We also performed a vulnerabilityscan of the supplied IP-address using nessus. Thereafter the team focussed on the*

sourcecode and tested some attackvectors, using the supplied windows executable that was installed on a virtual Windows XP system. We also visited the <http://www.retypingdante.com> website to get a better understanding of the project itself. After analysis of all collected data this report was written.”

While visiting the “Retyping Dante” website the following statement caught our special attention since it does reflect one of our visions:

*“Through web 2.0 an increasing number of people are communicating more freely through the internet. However this massive growth doesn’t encompass the initial design of the internet. To compensate a lot of effort has been put in securing the internet by building massive firewalls and technical workarounds. This approach will eventually break the internet and strain its progression. In order to communicate freely we need to bring down our firewalls and focus on securing the endpoints by using safe protocols between these endpoints.”*

Although “using safe protocols between these endpoints” is, in our opinion, not enough to keep the endpoints secure. The endpoints themselves must be hardened in such a way that a vulnerability in one provided service does not weaken other parts of the system. A more complete overview of this is provided in appendix A on page 32.

## 2.3 Target systems and applications

In the contract, the following list of target systems and applications is given:

Identification	Description
retypingdante.svn.sourceforge.net/svnroot/retypingdante	Sourcecode
purgatorio.retypingdante.com	Collecting server
80.100.151.120	Real collecting server
virgilius-installer.exe	Windows installer
virgilius_1.30_all.deb	Ubuntu release

Table 2.2: Target systems and applications

Obviously, systems and application that are not mentioned in the contract, are explicitly out of scope for this audit.

## 2.4 Information in advance

Z25.org has given Madison Gurkha B.V. the following information in advance:

Identification	Description
RTD_InstallationOverview.pdf	Installation guide
RTD_VirgiliusBeveiliging.pdf	Description of used crypto

Table 2.3: Information in advance

This information will, after the project has been completed, safely destroyed by Madison Gurkha B.V..

## 2.5 Audit criteria

As Madison Gurkha did not receive a security policy regarding the setup and management of the systems to be audited, these were audited using best practices, as applicable for the type of systems during the audit.

Some examples of security principles that are part of the audit criteria are:

- Diversity, making use of multiple layers of security, where the loss of one layer of security does not (yet) make the whole system vulnerable to attack;
- Implementation of redundancy, avoiding single points of failure;
- Safe defaults, also known as the deny all, except principle that says nothing is allowed, unless specifically allowed. This is a very important rule when setting up firewalls and hardening systems.
- Not leaking information. All information that leaks to an attacker, such as test files, documentation, error messages etc., make is easier for an attacker to penetrate the security;
- Compartments, when one part of the network or a system has become unsafe, this should not have an effect on the safety of the rest of the network or system;
- Limiting privileges, make sure that software and users have a minimal set of privileges, to make abuse harder to accomplish;
- Keep It Simple, Stupid (KISS), the less complex a system is, the less the chances are of making mistakes or configuration errors.

A more elaborate overview of the audit criteria that are used by Madison Gurkha B.V. are included in this report as appendix A on page 32.

## 2.6 Classification of risks and points of interest

All risks are being categorized as being high (**H**), medium (**M**) or low (**L**) level risks. High level risks need to be mitigated or, better yet, removed, as soon as possible. Medium level risks also need to be mitigated or removed, but this is less urgent. For low level risks, mitigation or removal would be nice, but not necessary.

Points of interest do not indicate risks but functional problems without risk. If these are to be fixed is being left to the customer

## 2.7 Dossier

During the audit, extensive logging was made. This information was used to create this report and is stored in the dossier belonging to this audit . This information will be kept on the systems of Madison Gurkha B.V. for a limited period of time.

When the final version of this report is given to the customer, the digitally stored data will be overwritten three (3) times (with the values 0xff, 0x00 and 0xff respectively) before they are being removed from the system (the end result can be compared with a secure erase as specified in "acs-DoD 5220"). Information on paper, CD-ROM and/or DVD-ROM will be destroyed using a shredder complying to DIN standard 32 757 1/1995 level 3 (which applies to the de-struction of confidential documents).

# Chapter 3

## Reconnaissance

In the first phase of the assessment general information of the investigated Information Technology (IT) infrastructure is collected. This information consists of the following parts:

- WHOIS related information;
- DNS related information;

### 3.1 Tools overview

During this phase of the assessment, amongst others, the following tools could be used:

BackTrack, dig, google, mtr, nslookup, paratrace, shazou<sup>1</sup>, traceroute, whois, Wikto.

### 3.2 WHOIS

While checking the WHOIS information of the specified server we found the first discrepancy, since the specified name of the target system (purgatorio.retypingdante.com) did not match the specified IP-address (80.100.151.120) but resolved to 62.193.248.118. This IP-address belonging to purgatorio.retypingdante.com is located on a French network. We therefor decided to disregard the purgatorio.retypingdante.com host.

The WHOIS-information of the domain retypingdante.com showed the information displayed in listing 3.1.

```
Domain Name: RETYPINGDANTE.COM
Registrar: KEY-SYSTEMS GMBH
Whois Server: whois.rrpproxy.net
Referral URL: http://www.key-systems.net
Name Server: NS1.DOMAINDISCOUNT24.NET
Name Server: NS2.DOMAINDISCOUNT24.NET
Name Server: NS3.DOMAINDISCOUNT24.NET
Status: ok
Updated Date: 06-sep-2008
Creation Date: 06-sep-2008
Expiration Date: 06-sep-2009
...
DOMAIN: RETYPINGDANTE.COM
```

<sup>1</sup>op dit moment te vinden via <https://addons.mozilla.org/en-US/firefox/addon/2993>

```
RSP: domaindiscount24.com
URL: http://www.dd24.net

owner-contact: P-ARL502
owner-organization: Stichting z25.org
owner-fname: Arnaud
owner-lname: Loonstra
owner-street: Concordiastraat 67A
owner-city: Utrecht
owner-zip: 3531 EM
owner-country: NL
owner-phone: +31628159323
owner-email: info@z25.org

admin-contact: P-ARL502
admin-organization: Stichting z25.org
admin-fname: Arnaud
admin-lname: Loonstra
admin-street: Concordiastraat 67A
admin-city: Utrecht
admin-zip: 3531 EM
admin-country: NL
admin-phone: +31628159323
admin-email: info@z25.org

tech-contact: P-ARL502
tech-organization: Stichting z25.org
tech-fname: Arnaud
tech-lname: Loonstra
tech-street: Concordiastraat 67A
tech-city: Utrecht
tech-zip: 3531 EM
tech-country: NL
tech-phone: +31628159323
tech-email: info@z25.org

billing-contact: P-ARL502
billing-organization: Stichting z25.org
billing-fname: Arnaud
billing-lname: Loonstra
billing-street: Concordiastraat 67A
billing-city: Utrecht
billing-zip: 3531 EM
billing-country: NL
billing-phone: +31628159323
billing-email: info@z25.org

nameserver: ns1.domaindiscount24.net
nameserver: ns2.domaindiscount24.net
nameserver: ns3.domaindiscount24.net
```

Listing 3.1: WHOIS retypingdante.com

The specified IP-address 80.100.151.120 is owned by XS4ALL for use on their ADSL network as determined from the contents of the WHOIS information displayed in listing 3.2.

```
inetnum:      80.100.144.0 - 80.100.155.255
netname:      XS4ALL
descr:        XS4ALL Internet BV
descr:        ADSL Static IP numbers
country:      NL
admin-c:      XS42-RIPE
tech-c:       XS42-RIPE
status:       ASSIGNED PA
```

```
remarks:      Please send email to "abuse@xs4all.nl" for complaints
remarks:      regarding portscans, DoS attacks and spam.
mnt-by:       XS4ALL-MNT
source:       RIPE # Filtered

role:         XS4ALL Internet NOC
address:      XS4ALL Internet BV
address:      Postbus 1848
address:      1000BV Amsterdam
address:      The Netherlands
phone:        +31 20 3987654
fax-no:       +31 20 3987604
abuse-mailbox: abuse@xs4all.nl
admin-c:      CB127
tech-c:       CB127
tech-c:       OD45
tech-c:       RZ2757-RIPE
tech-c:       KAI11-RIPE
nic-hdl:      XS42-RIPE
mnt-by:       XS4ALL-MNT
source:       RIPE # Filtered

% Information related to '80.100.0.0/15AS3265'

route:        80.100.0.0/15
descr:        XS4ALL Internet routing
origin:       AS3265
mnt-by:       XS4ALL-MNT
source:       RIPE # Filtered
```

Listing 3.2: WHOIS 80.100.151.120

Z25.org is mentioned in the WHOIS-record of the target domain and is based in the Netherlands, according to its WHOIS-record, displayed in listing 3.3.

```
Domain ID:D95840501-LROR
Domain Name:Z25.ORG
Created On:17-Mar-2003 14:06:37 UTC
Last Updated On:18-Mar-2008 01:22:01 UTC
Expiration Date:17-Mar-2009 14:06:37 UTC
Sponsoring Registrar:Key-Systems GmbH (R51-LROR)
Status:OK
Registrant ID:P-SLL23-7312
Registrant Name:Simonis Laurens
Registrant Street1:Laurens Reaalstraat 4bis
Registrant City:Utrecht
Registrant Postal Code:3531 GN
Registrant Country:NL
Registrant Phone:+31.614230858

Registrant Email:lmsimonis@wxs.nl
Admin ID:P-SLL23-7312
Admin Name:Simonis Laurens
Admin Street1:Laurens Reaalstraat 4bis

Admin City:Utrecht
Admin Postal Code:3531 GN
Admin Country:NL
Admin Phone:+31.614230858
Admin Email:lmsimonis@wxs.nl
Tech ID:P-DTD1-85
Tech Name:domaindiscount24.com technical department
Tech Organization:Key-Systems GmbH
Tech Street1:Prager Ring 4-12 Building 5
Tech City:Zweibruecken
```

```
Tech State/Province:Rheinland Pfalz
Tech Postal Code:66482
Tech Country:DE
Tech Phone:+49.6332791850
Tech FAX:+49.6332791851
Tech Email:tech@domaindiscount24.com
Name Server:NS1.DOMAINDISCOUNT24.NET
Name Server:NS3.DOMAINDISCOUNT24.NET
Name Server:NS2.DOMAINDISCOUNT24.NET
```

Listing 3.3: WHOIS Z25.org

### 3.3 Domain Name Service (DNS)

Dig supplied the following DNS information for the `retypingdante.com` domain:

```
; <<>> DiG 9.4.2-P2 <<>> retypingdante.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21807
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;retypingdante.com.      IN  A

;; ANSWER SECTION:
retypingdante.com.  600 IN  A   62.193.248.118

;; AUTHORITY SECTION:
retypingdante.com.  171691 IN  NS  ns3.domaindiscount24.net.
retypingdante.com.  171691 IN  NS  ns1.domaindiscount24.net.
retypingdante.com.  171691 IN  NS  ns2.domaindiscount24.net.

;; ADDITIONAL SECTION:
ns1.domaindiscount24.net. 76501 IN  A   88.198.17.120
ns2.domaindiscount24.net. 76501 IN  A   217.11.52.234
ns3.domaindiscount24.net. 76501 IN  A   217.188.246.105

;; Query time: 26 msec
;; SERVER: 88.159.10.2#53(88.159.10.2)
;; WHEN: Mon Mar  9 08:37:18 2009
;; MSG SIZE rcvd: 173
```

Listing 3.4: DNS information `retypingdante.com`

None of the nameservers specified in listing 3.4 allowed a zone transfer of the target domain and none allowed resolving external names. All three nameservers supplied the “version information” as shown in listing 3.5.

```
; <<>> DiG 9.4.2-P2 <<>> @ns1.domaindiscount24.net version.bind chaos txt
; (1 server found)
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47999
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;version.bind.          IN  TXT

;; ANSWER SECTION:
version.bind.          5   IN  TXT "Served by Key-Systems GmbH"
```

```
;; Query time: 17 msec
;; SERVER: 88.198.17.120#53(88.198.17.120)
;; WHEN: Mon Mar 9 08:44:11 2009
;; MSG SIZE rcvd: 69
```

Listing 3.5: DNS version information

The configuration of all three nameservers is good.

# Chapter 4

## Vulnerability scans

In this phase of the assessment we used `nessus` to check the collecting server, with the IP-address 80.100.151.120, for known vulnerabilities. This server is also known as `speakeasy.z25.org` and uses UDP port 9458 to accept data from all `virgilius` clients. This communication uses the Open-Sound Control (OSC) protocol which is specified on <http://opensoundcontrol.org/introduction-osc>.

Besides this function this IP-address also provides access to several other services:

- Secure Shell (SSH) on port 22/tcp
- Simple Mail Transfer Protocol (SMTP) on port 25/tcp
- HyperText Transfer Protocol (HTTP) on port 80/tcp
- Internet Message Access Protocol (IMAP) on port 143/tcp
- Secure HTTP (HTTPS) on port 443/tcp
- Secure IMAP (IMAPS) on port 993/tcp

### 4.1 SSH

An SSH server is listening on the remote port. This server only allows version 2.0 of the protocol which is considered safe:

```
$ telnet speakeasy.z25.org 22
Trying 80.100.151.120...
Connected to speakeasy.z25.org.
Escape character is '^]'.
SSH-2.0-OpenSSH_4.3p2 Debian-9etch3
```

### 4.2 SMTP

An SMTP server is listening on the remote port. Since SMTP servers are the targets of spammers, it is recommended to disable it if it is not used. The server is identified as:

```
220 speakeasy.cc.lan ESMTP Postfix (Debian/GNU)
```

## 4.3 HTTP(S)

Using the remote HTTP banner, it is possible to guess that the Linux distribution installed on the remote host is:

```
Linux Kernel 2.6 on Debian 4.0 (etch)
```

The web server is running:

```
Apache/2.2.3 (Debian) mod_python/3.2.10 Python/2.4.4 PHP/5.2.0-8+etch13 mod_ssl
/2.2.3 OpenSSL/0.9.8c
```

Listing 4.1: Apache webserver header

### Low Risk 1 - speakeasy.z25.org

#### Finding:

Too much information on the installed Apache configuration is displayed.

#### Confirmation:

See listing 4.1.

#### Risk:

The version information provided by the Apache webserver can be used to search for specific vulnerabilities in installed tools like PHP or Python.

#### Recommendation:

Specify `ServerToken ProductOnly` in the `httpd.conf`.

		Impact		
		L	M	H
Knowledge	H			
	M		■	
	L			

Requesting `http://speakeasy.z25.org/.svn/entries` shows the information as specified in listing 4.2.

```
8
dir
24589
http://svn.egroupware.org/egroupware/branches/1.4/aliases/default
http://svn.egroupware.org/egroupware

2007-09-25T11:16:20.496080Z
24466
ralfbecker
has-props

svn:special svn:externals svn:needs-lock
svn:externals

923e3c2e-ae13-0410-ba18-ced7211e33de
```

Listing 4.2: SVN Entries

## Low Risk 2 - speakeasy.z25.org

### Finding:

The web server on the remote host allows read access to `.svn/entries` files.

		Impact		
		L	M	H
Knowledge	H			
	M			
	L		■	

### Confirmation:

See listing 4.2 on the preceding page.

### Risk:

The webserver exposes all file names in the svn module on the website.

### Recommendation:

Configure permissions for the affected web server to deny access to the `.svn` directory.

While scanning the HTTPS service we noticed that this service uses a self-signed certificate as shown in listing 4.3.

```
Subject Name:

Country: NL
State/Province: Utrecht
Organization: Stichting z25.org
Common Name: egw.z25.org

Issuer Name:

Organization: Stichting z25.org
Email Address: support@z25.org
Locality: Utrecht
State/Province: Utrecht
Country: NL
Common Name: Z25 Root CA

Serial Number: 01

Version: 1

Signature Algorithm: MD5 With RSA Encryption

Not Valid Before: May 19 12:18:52 2008 GMT
Not Valid After: May 17 12:18:52 2018 GMT

Public Key Info:

Algorithm: RSA Encryption
Public Key: 00 D4 5E 3D C1 09 05 7B 2F 6B 3D 8D 7A 23 4D 55 BB CF 0E E1
88 2A 6F A1 07 36 36 37 D7 DE 02 0B 95 E1 4C 04 F5 98 1F 57
9D 70 59 ED 43 28 6D A9 F9 FA 90 7E 02 5B 1A 11 DD FF ED 4A
3F 34 FE A9 6D 0A 72 B3 01 A7 AB 8F 61 8F 58 94 ED B8 E3 8E
CE 91 19 A6 79 CC A6 46 FB FC 5F 6D A7 F9 34 C5 71 4B 1D A4
08 C7 3E 45 EF 19 EB 6B 8E ED E9 D2 AD DC 60 55 C3 D4 E2 98
06 BB 07 A5 94 B6 8F D9 21

Exponent: 01 00 01

Signature: 00 5B 1A 5E E1 F9 D2 20 31 B9 1C 54 1D A3 B1 E1 0C 53 D0 12
C4 65 94 33 AB DA DD E6 F9 7D AE 81 52 44 7F 47 99 0C FF 03
25 54 D1 93 42 64 C7 3E 0D 69 DB 50 23 A9 32 E7 97 CB 63 31
87 C1 93 DB 54 A9 1F C2 EC AF C4 DD 97 FC A0 D5 6F D6 11 9F
50 50 A1 6A 81 FF 6E 73 39 D9 D5 55 57 A6 FD E8 99 55 A5 07
0F AE 12 58 E5 95 1E 90 46 84 09 2E 15 56 E5 91 7D 89 26 9D
93 52 67 37 43 FA 83 A4 E3
```

Listing 4.3: HTTPS Certificate information

### Low Risk 3 - speakeasy.z25.org

#### Finding:

The remote HTTPS service uses an SSL certificate that has been signed using a cryptographically weak MD5 hashing algorithm. MD5 is known to be vulnerable to collision attacks.

#### Confirmation:

See listing 4.3 on the facing page.

#### Risk:

In theory, a determined attacker may be able to leverage this weakness to generate another certificate with the same digital signature, which could allow him to masquerade as the affected service.

See also: <http://www.phreedom.org/research/rogue-ca/>.

#### Recommendation:

Create a new certificate using SHA-1 as the signature algorithm.

		Impact		
		L	M	H
Knowledge	H		■	
	M			
	L			

### Medium Risk 1 - speakeasy.z25.org

#### Finding:

The remote HTTPS service accepts connections encrypted using SSLv2, which reportedly suffers from several cryptographic flaws and has been deprecated for several years.

#### Confirmation:

See listing 4.4.

#### Risk:

An attacker may be able to exploit SSLv2 related issues to conduct man-in-the-middle attacks or decrypt communications between the affected service and clients. See also: <http://www.schneier.com/paper-ssl.pdf>.

#### Recommendation:

We advise the use of SSLv3 or TLSv1 instead.

		Impact		
		L	M	H
Knowledge	H			■
	M			
	L			

The HTTPS server supports the ciphers as specified in listing 4.4.

Low Strength Ciphers (< 56-bit key)

SSLv2

**EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export**

**EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export**

SSLv3

**EXP-EDH-RSA-DES-CBC-SHA Kx=DH(512) Au=RSA Enc=DES(40) Mac=SHA1 export**

**EXP-DES-CBC-SHA Kx=RSA(512) Au=RSA Enc=DES(40) Mac=SHA1 export**

**EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export**

**EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export**

TLSv1

**EXP-EDH-RSA-DES-CBC-SHA Kx=DH(512) Au=RSA Enc=DES(40) Mac=SHA1 export**

**EXP-DES-CBC-SHA Kx=RSA(512) Au=RSA Enc=DES(40) Mac=SHA1 export**

**EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export**

**EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export**

Medium Strength Ciphers (>= 56-bit and < 112-bit key)

SSLv2

DES-CBC-MD5 Kx=RSA Au=RSA Enc=DES(56) Mac=MD5

SSLv3

EDH-RSA-DES-CBC-SHA Kx=DH Au=RSA Enc=DES(56) Mac=SHA1

DES-CBC-SHA Kx=RSA Au=RSA Enc=DES(56) Mac=SHA1

```

TLsv1
EDH-RSA-DES-CBC-SHA Kx=DH Au=RSA Enc=DES(56) Mac=SHA1
DES-CBC-SHA Kx=RSA Au=RSA Enc=DES(56) Mac=SHA1

High Strength Ciphers (>= 112-bit key)
SSLv2
DES-CBC3-MD5 Kx=RSA Au=RSA Enc=3DES(168) Mac=MD5
RC2-CBC-MD5 Kx=RSA Au=RSA Enc=RC2(128) Mac=MD5
RC4-MD5 Kx=RSA Au=RSA Enc=RC4(128) Mac=MD5
SSLv3
EDH-RSA-DES-CBC3-SHA Kx=DH Au=RSA Enc=3DES(168) Mac=SHA1
DES-CBC3-SHA Kx=RSA Au=RSA Enc=3DES(168) Mac=SHA1
RC4-MD5 Kx=RSA Au=RSA Enc=RC4(128) Mac=MD5
RC4-SHA Kx=RSA Au=RSA Enc=RC4(128) Mac=SHA1
TLsv1
EDH-RSA-DES-CBC3-SHA Kx=DH Au=RSA Enc=3DES(168) Mac=SHA1
DHE-RSA-AES128-SHA Kx=DH Au=RSA Enc=AES(128) Mac=SHA1
DHE-RSA-AES256-SHA Kx=DH Au=RSA Enc=AES(256) Mac=SHA1
DES-CBC3-SHA Kx=RSA Au=RSA Enc=3DES(168) Mac=SHA1
AES128-SHA Kx=RSA Au=RSA Enc=AES(128) Mac=SHA1
AES256-SHA Kx=RSA Au=RSA Enc=AES(256) Mac=SHA1
RC4-MD5 Kx=RSA Au=RSA Enc=RC4(128) Mac=MD5
RC4-SHA Kx=RSA Au=RSA Enc=RC4(128) Mac=SHA1

```

Listing 4.4: HTTPS Ciphers

## Low Risk 4 - speakeasy.z25.org

### Finding:

The remote HTTPS service may be forced to use weak encryption ciphers that are relatively easy to break.

		Impact		
		L	M	H
Knowledge	H			
	M			
	L			

### Confirmation:

See listing 4.4 on the previous page.

### Risk:

An attacker may be able to decode the communication between the client(s) and the HTTPS server when weak encryption ciphers are used.

### Recommendation:

We advise to only support high strength ciphers.

It is possible to enumerate web directories. The following directories were discovered:

```
/doc, /icons, /setup
```

While this is not, in and of itself, a bug, these directories should be manually inspected to ensure that they are in compliance with company security standards

Some information about the remote HTTP configuration can be extracted.

```

Protocol version : HTTP/1.1
SSL : yes
Pipelining : yes
Keep-Alive : yes
Options allowed : GET,HEAD,POST,OPTIONS,TRACE
Headers :

Date: Mon, 09 Mar 2009 09:46:29 GMT
Server: Apache/2.2.3 (Debian) mod_python/3.2.10 Python/2.4.4 PHP/5.2.0-8+etch13
      mod_ssl/2.2.3 OpenSSL/0.9.8c
X-Powered-By: PHP/5.2.0-8+etch13
Location: login.php

```

```
Content-Length: 0
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

Listing 4.5: HTTP protocol information

The server response from a TRACE request is shown in listing 4.6.

```
TRACE /xrh47h8x.html HTTP/1.1
Host: a80-100-151-120.adsl.xs4all.nl
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Date: Mon, 9 Mar 2009 09:46:44 GMT
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Accept-Charset: iso-8859-1,*,utf-8
Pragma: no-cache
Accept-Language: en
Connection: Keep-Alive
```

Listing 4.6: TRACE method

### Low Risk 5 - speakeasy.z25.org

#### Finding:

The remote webserver supports the TRACE method. TRACE is a HTTP method that is used to debug web server connections. In addition, it has been shown that servers supporting the TRACE method are subject to cross-site scripting attacks, dubbed XST for “Cross-Site Tracing”, when used in conjunction with various weaknesses in browsers.

		Impact		
		L	M	H
Knowledge	H			
	M			
	L		■	

#### Confirmation:

See listing 4.6.

#### Risk:

An attacker may use this flaw to trick legitimate web users to give him their credentials. See also: [http://www.cgisecurity.com/whitehat-mirror/WH-WhitePaper\\_XST\\_ebook.pdf](http://www.cgisecurity.com/whitehat-mirror/WH-WhitePaper_XST_ebook.pdf).

#### Recommendation:

Disable the TRACE method via the `TraceEnable` directive. In older versions of Apache one had to filter the methods using constructs like represented in listing 4.7.

```
RewriteEngine on
RewriteCond %{REQUEST_METHOD} ^ (TRACE|TRACK)
RewriteRule .* - [F]
```

Listing 4.7: TRACE and TRACK filtering

## 4.4 IMAP(S)

The IMAPS server uses a self-signed certificate, and allows the use of SSLv2 as a valid protocol.

## Medium Risk 2 - speakeasy.z25.org

### Finding:

The remote IMAPS service accepts connections encrypted using SSLv2, which reportedly suffers from several cryptographic flaws and has been deprecated for several years.

		Impact		
		L	M	H
Knowledge	H			■
	M			
	L			

### Confirmation:

See listing 4.8.

### Risk:

An attacker may be able to exploit SSLv2 related issues to conduct man-in-the-middle attacks or decrypt communications between the affected service and clients. See also: <http://www.schneier.com/paper-ssl.pdf>.

### Recommendation:

We advise the use of SSLv3 or TLSv1 instead.

This server supports the ciphers as specified in listing 4.8.

#### Low Strength Ciphers (< 56-bit key)

##### SSLv2

**EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export**

**EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export**

##### SSLv3

**EXP-ADH-DES-CBC-SHA Kx=DH(512) Au=None Enc=DES(40) Mac=SHA1 export**

**EXP-ADH-RC4-MD5 Kx=DH(512) Au=None Enc=RC4(40) Mac=MD5 export**

**EXP-EDH-RSA-DES-CBC-SHA Kx=DH(512) Au=RSA Enc=DES(40) Mac=SHA1 export**

**EXP-DES-CBC-SHA Kx=RSA(512) Au=RSA Enc=DES(40) Mac=SHA1 export**

**EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export**

**EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export**

##### TLSv1

**EXP-EDH-RSA-DES-CBC-SHA Kx=DH(512) Au=RSA Enc=DES(40) Mac=SHA1 export**

**EXP-ADH-DES-CBC-SHA Kx=DH(512) Au=None Enc=DES(40) Mac=SHA1 export**

**EXP-ADH-RC4-MD5 Kx=DH(512) Au=None Enc=RC4(40) Mac=MD5 export**

**EXP-DES-CBC-SHA Kx=RSA(512) Au=RSA Enc=DES(40) Mac=SHA1 export**

**EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export**

**EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export**

#### High Strength Ciphers (>= 112-bit key)

##### SSLv2

DES-CBC3-MD5 Kx=RSA Au=RSA Enc=3DES(168) Mac=MD5

RC2-CBC-MD5 Kx=RSA Au=RSA Enc=RC2(128) Mac=MD5

RC4-MD5 Kx=RSA Au=RSA Enc=RC4(128) Mac=MD5

##### SSLv3

**ADH-DES-CBC3-SHA Kx=DH Au=None Enc=3DES(168) Mac=SHA1**

**ADH-RC4-MD5 Kx=DH Au=None Enc=RC4(128) Mac=MD5**

EDH-RSA-DES-CBC3-SHA Kx=DH Au=RSA Enc=3DES(168) Mac=SHA1

DES-CBC3-SHA Kx=RSA Au=RSA Enc=3DES(168) Mac=SHA1

RC4-MD5 Kx=RSA Au=RSA Enc=RC4(128) Mac=MD5

RC4-SHA Kx=RSA Au=RSA Enc=RC4(128) Mac=SHA1

##### TLSv1

EDH-RSA-DES-CBC3-SHA Kx=DH Au=RSA Enc=3DES(168) Mac=SHA1

DHE-RSA-AES128-SHA Kx=DH Au=RSA Enc=AES(128) Mac=SHA1

DHE-RSA-AES256-SHA Kx=DH Au=RSA Enc=AES(256) Mac=SHA1

**ADH-DES-CBC3-SHA Kx=DH Au=None Enc=3DES(168) Mac=SHA1**

**ADH-AES128-SHA Kx=DH Au=None Enc=AES(128) Mac=SHA1**

**ADH-AES256-SHA Kx=DH Au=None Enc=AES(256) Mac=SHA1**

**ADH-RC4-MD5 Kx=DH Au=None Enc=RC4(128) Mac=MD5**

DES-CBC3-SHA Kx=RSA Au=RSA Enc=3DES(168) Mac=SHA1

AES128-SHA Kx=RSA Au=RSA Enc=AES(128) Mac=SHA1

AES256-SHA Kx=RSA Au=RSA Enc=AES(256) Mac=SHA1

```
RC4-MD5 Kx=RSA Au=RSA Enc=RC4 (128) Mac=MD5
RC4-SHA Kx=RSA Au=RSA Enc=RC4 (128) Mac=SHA1
```

Listing 4.8: IMAPS Ciphers

## Low Risk 6 - speakeasy.z25.org

### Finding:

The remote host supports the use of anonymous SSL ciphers and supports the use of SSL ciphers that offer either weak encryption or no encryption at all. While this enables an administrator to set up a service that encrypts traffic without having to generate and configure SSL certificates, it offers no way to verify the remote host's identity and renders the service vulnerable to a man-in-the-middle attack.

		Impact		
		L	M	H
Knowledge	H			
	M			
	L			

### Confirmation:

See listing 4.8 on the preceding page.

### Risk:

An attacker may be able to use a Man-in-the-Middle (MitM) attack to access the communication between the client(s) and the HTTPS server.

### Recommendation:

We advise to only support high strength ciphers.

## 4.5 Conclusion

Although no major vulnerabilities were discovered on this system and all investigated services are possibly not related to the specific retypingdante project, we still urge the developers to host the collecting service on a dedicated server, to ensure the maximum level of security for the Retyping Dante project.

After providing access to the preliminary version of this report Z25.org showed us the network diagram of the setup. According to this diagram all services mentioned above, with the exception of the UDP service that uses port 9458, are located on an internal system 192.168.12.1. The service using port 9458/udp is located on internal server 192.168.12.9 (purgatorio.retypingdante.com). This information is taken into account in this version of the report.

# Chapter 5

## Application audit

In addition to the investigation of standard software, as described in the previous chapter, we describe the investigation of the custom made applications in this chapter.

### 5.1 Overview

The `virgilius` packages comes as a Windows self-installing executable and as a Debian-package. We used the following files:

- `virgilius-1.26-installer.exe`
- `virgilius_1.30_all.deb`

We also used `svn co https://retyingdante.svn.sourceforge.net/svnroot/retyingdante/trunk/` to extract the complete source tree. But since 1.30 is the current version there were no differences found.

The Debian package is actually an archive containing `control.tar.gz` and `debian-binary`, which are there for Debian package management. The `data.tar.gz` file contains the actual program files.

The file `/usr/bin/virgilius.py` is the main program. In `/usr/share/virgilius`, we find several Python components and a subdirectory `gfx` holding icons. In `/usr/share/applications`, a file `virgilius.desktop` is present. It starts `virgilius` with the `-u` option, meaning it does not look for updates.

According to the developers, the updates of the Linux version of `virgilius` are distributed using a Debian repository. This makes the automatic update check unnecessary.

The file `/usr/share/python-support/virgilius.dirs` is needed to make Python find the correct files. In `/usr/share/doc/virgilius`, we find a `README`-file (containing very little information), a changelog and a copyright file.

### 5.2 Design, existence and working of the application

While reading through the Python source code we tried to identify possible attack vectors for this program. The potential candidates were:

- Python code problems
- Update mechanism
- Encryption of data
- Keylogging process

- Open UDP port

## 5.2.1 Initial source code checking

A quick Python checkscript called `pychecker` was run on all source code. This did not reveal any problems.

The file sizes are as follows:

```
2626 OSC.py
    20 rtd_Encrypt.py
    133 rtd_Packer.py
    139 rtd_Shipper.py
    46 rtd_UpdateCheck.py
    495 rtd_WxUi.py
    164 rtd_Xkeylog.py
    87 virgilius.py
3710 total
```

The total number of lines is 3710, but 2626 are in use by `OSC.py`, which is the OpenSound Control module which is not written by Z25.org. What remains are 1084 lines of code.

The file `rtd_WxUi.py` contains the main code. There are separate files that deal with encryption, packing, shipping, the actual logging and checking for updates. The `virgilius.py` is only a startup script.

## 5.2.2 Update mechanism

In the source code we identified the update mechanism, so we needed to check whether this process could be influenced to potentially harm the client. The first issue we noticed was a race condition in the code (`rtd_WxUi.py`):

```
def onUpdate(self, event):
    """
    Check for update
    """
    import rtd_UpdateCheck
    upd = rtd_UpdateCheck.rtd_UpdateCheck(VERSION)
```

In the last two lines, we see a race condition. First, `rtd_UpdateCheck.rtd_UpdateCheck` is called, after which `upd.onNewUpdate` is overridden by a new method.

Apart from possible loss of functionality, we see no harm done, as the original method does nothing more than a print, as we can see in `rtd_UpdateCheck.py`:

```
def onNewUpdate(self, msg):
    print msg
```

In the same module we see:

```
if self.__curVersion < float(ver):
    self.onNewUpdate(msg)
```

So the `onNewUpdate` method is called with `msg`. In `rtd_WxUi.py:onNewUpdate()`, we see:

```
def onNewUpdate(self, ver):
    """
    Callback function to be called by the updater thread
    """
    evt = rtd_Event(detail = ["Update", ver])
    wx.PostEvent(self, evt)
```

The name `ver` implies that it holds a version, while in fact it holds a message. We trace further what happens with the event and see, in `rtd_WxUi.py`:

```
#
# Custom event manager
#
def onRtdEvent(self, event):
    """
    onRtdEvent runs methods according to event.detail
    """
    if type(event.detail) == list:
        if event.detail[0] == "Update":
            print "RtdEvent: ", event.detail
            self.onMessage(event.detail[1])
        else:
            print "RtdEvent list event not matched: ", event.detail
```

So the message is passed to `onMessage`:

```
def onMessage(self, msg):
    """
    Show a message
    """
    dlg = wx.Dialog(None, -1, "Message from Virgilius",
                   style=wx.DEFAULT_DIALOG_STYLE|wx.RAISED_BORDER|wx.RESIZE_BORDER|
                   wx.TAB_TRAVERSAL)
    hwin = HtmlWindow(dlg, -1, size=(400,200))
    hwin.SetPage(msg)
    btn = hwin.FindWindowById(wx.ID_OK)
    irep = hwin.GetInternalRepresentation()
    hwin.SetSize((irep.GetWidth()+25, irep.GetHeight()+10))
    dlg.SetClientSize(hwin.GetSize())
    dlg.CentreOnParent(wx.BOTH)
    dlg.SetFocus()
    result = dlg.ShowModal()
    print result
    dlg.Destroy()
```

Thus, the message is rendered with an HTML-parser. Naturally, we try to see what happens with Javascript.

We install the windows client on a windows system and add `repos.retypingdante.com` to the host file (`C:\windows\system32\drivers\etc\hosts`) with an address pointing to one of our own systems. We then make the client check for an update.

On the host mimicing `repos.retypingdante.com`, we see a connection on port 80, requesting an update:

```
GET /win/version.py HTTP/1.1
Accept-Encoding: identity
Host: repos.retypingdante.com
Connection: close
User-Agent: Python-urllib/2.5
```

We feed the client the following answer:

```
HTTP/1.0 200 OK
Date: Mon, 09 Mar 2009 09:54:32 GMT

(1.29, '<body bgcolor=#444422 TEXT=#000000 LINK=#ff0000>\nA new version of
Virgilius is available. Please click on\n<a href="http://repos.retypingdante.
com/win/virgilius-1.26-installer.exe">\nthis link </a>\n\nto download the new
version\n</body>\n')
```

Note that our clients' version is 1.28, thus the client should believe there is an update. Indeed, we see a popup windows with the message we provided in the answer above.

In the popup window, a link is shown. When we click on it, Internet explorer is started and the following request is made:

```
GET /win/virgilius-1.26-installer.exe HTTP/1.1
Accept: */*
Accept-Language: en-us
UA-CPU: x86
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 1.1.4322;
.NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)
Host: repos.retypingdante.com
Connection: Keep-Alive
```

When we feed the same answer as previously:

```
HTTP/1.0 200 OK
Date: Mon, 09 Mar 2009 09:54:32 GMT

<body bgcolor=#444422 TEXT=#000000 LINK=#ff0000>\nA new version of Virgilius is
available. Please click on\n<a href="http://repos.retypingdante.com/win/
virgilius-1.26-installer.exe">\nthis link </a>\nto download the new version\n
</body>
```

we see that it is rendered.

Next we try to insert Javascript in the first request, with the following response:

```
HTTP/1.0 200 OK
Date: Mon, 09 Mar 2009 09:54:32 GMT

(1.29, ' <body bgcolor=#444422 TEXT=#000000 LINK=#ff0000>\nA new version of
Virgilius is available. Please click on\n<a href="http://repos.retypingdante.
com/win/virgilius-1.26-installer.exe">\nthis link </a>\nto download the new
version\n<script>alert ("Boe")</script></body>\n')
```

Listing 5.1: Update server response

The Javascript is not executed. We have not investigated this any further, since clicking the update link will launch a browser window that does a request to a link we control and where Javascript will be executed (depending on browser features and security settings).

## High Risk 1 - virgilius

### Finding:

When an attacker controls the host file or the DNS of the victim, or when there is another way of mimicing the update server (e.g. with a man in the middle attack) the update mechanism can be used to execute code on the client system.

		Impact		
		L	M	H
Knowledge	H			
	M			■
	L			

### Confirmation:

All necessary steps are described in section 5.2.2 on page 21.

### Risk:

The update mechanism is prone to attacks, enabling the execution of code on the client.

### Recommendation:

When either situation is true, the victim has a problem that is an order of magnitude bigger than the possibility of executing JavaScript. However, we still advice to have the client verify that it is talking to the correct update server.

## Remark 1 - virgilius

### Finding:

The update server is not up to date; still announcing version 1.26, while the Microsoft Windows version supplied already mentioned version 1.28.

### Confirmation:

See listing 5.1 on the previous page.

### Recommendation:

Update the update server.

## 5.2.3 Encryption of data

In `rtd_Encrypt.py`, we see that the package `M2Crypto` is being used. This Python package can be found at <http://chandlerproject.org/bin/view/Projects/MeTooCrypto>. The package offers a wrapper around (Open)SSL and offers several services, of which RSA-encryption is one.

It is good to make use of proven implementations of cryptographic protocols, instead of writing your own version.

For the actual RSA encryption, a public key is included in the package (`virgilius.pem`).

```
> openssl asn1parse -dump -in ./source/retypingdante/trunk/inferno/src/virgilius.
pem
 0:d=0  hl=3 l= 159 cons: SEQUENCE
 3:d=1  hl=2 l=  13 cons: SEQUENCE
 5:d=2  hl=2 l=   9 prim: OBJECT                :rsaEncryption
16:d=2  hl=2 l=   0 prim: NULL
18:d=1  hl=3 l= 141 prim: BIT STRING
0000 - 00 30 81 89 02 81 81 00-c1 15 61 2d b0 8f 0d 09  .0.....a-....
0010 - 11 af c4 13 62 38 9b ea-c6 a7 8b 7b bf a9 80 20  ....b8.....{...
0020 - f5 02 f8 dd c3 e0 6f 13-03 2e d9 60 54 3e 9b 2b  .....o....`T>.+
0030 - 6f f5 be 66 df c0 dd 40-43 03 95 97 05 a9 e4 31  o..f...@C.....1
0040 - 9c d2 dc e0 55 76 12 53-d2 51 1f 1c 53 fc d9 9d  ....Uv.S.Q...S...
0050 - b7 bf 27 45 5c c6 9b 60-09 43 c5 dc 7b 6c 29 30  ..'E...\`C...{1}0
0060 - 91 06 27 10 ab 24 e6 88-f0 a8 ee 16 29 0d ba af  ..'..$......)...
0070 - 4b bb e1 d8 20 79 a3 5b-96 e7 79 22 07 6d c0 bb  K... y.[..y".m..
0080 - d6 62 5b 9b e2 db 3f 6f-02 03 01 00 01          .b[...?o.....
```

Before the data is encrypted, it is shuffled randomly:

```
crypt = self.enc.encrypt(self.randomize(self.container))
```

and

```
def randomize(self, s):
    """
    shuffle a string. Takes a string as an argument and returns
    the shuffled string
    """
    list_s = list(s)
    random.shuffle(list_s)
    return "".join(list_s)
```

The maximum size of the container, before being sent, is 32 bytes by default but can be smaller, as containers are being sent when the `shipTimer` threshold has been reached as well. This is set to 4 seconds by default. For an average typist, about 8 characters will be typed in this timeframe.

We suspect that the randomisation function has been added to make sure that an attacker with access to the decrypted characters (on the server), cannot deduce what has originally been typed. When the characters are randomised in chunks of 32, this indeed becomes difficult. However, when the amount of characters per shipment drops to under 10, it becomes reasonably easy to do.

It would be safest to only send the maximum amount of data per shipment. The drawback is that it can take longer for data to come in, but it will never take more than 4 seconds. When there are many clients sending in data, the delays will not be noticeable anymore. An other advantage is that of speed. In the implementation we see that RSA is used exclusively. Public key encryption is slow and this could become a bottleneck on the server. In practical implementations, secret key encryption is normally used, where the secret key is the only data being sent using public key encryption.

### Medium Risk 3 - virgilius

#### Finding:

The amount of characters per shipment may be too small, due to the client sending data each 4 seconds maximum.

		Impact		
		L	M	H
Knowledge	H			■
	M			
	L			

#### Confirmation:

All necessary steps are described in section 5.2.3 on the facing page.

#### Risk:

An attacker with access to the server can possibly reconstruct what is typed on specific clients.

#### Recommendation:

Only ship containers that contain a full load of 32 characters.

In all cases, a 148-byte UDP packet is sent, irrespective of the amount of keystrokes contained in the packet. Two packets containing the same 1-octet keystroke, have different encryptions. This is due to M2Crypt being called using `pkcs1_padding`:

```
enc_data = self.pubkey.public_encrypt(data, M2Crypto.RSA.pkcs1_padding)
```

This type of padding is defined in RFC3447 (<http://www.ietf.org/rfc/rfc3447.txt>) and suggests the padding to be pseudorandom in the case of public key operations (which the above is). As M2crypt uses OpenSSL, we verified that OpenSSL does conform to RFC3447.

Indeed it does, as can be seen in `crypto/rsa/rsa_pk1.c` where `RSA_padding_add_PKCS1_type_2()` is specified. This results in the actual UDP-packet being sent having a fixed length, independent of the amount of data typed. This results in traffic analysis becoming harder to do, which is good.

## 5.2.4 Container handling

In `rtd_Packer.py`, we see how items (keys) are added to a container. When the container is full, or after a timeout of 4 seconds, the container is shipped. Due to the fact that there might be multiple threads involved when shipping the container, a lock is initialised:

```
#Lock to guard the container
self.__lck = threading.Lock()
```

The lock is used when shipping the container:

```
def ship(self):
    self.__lck.acquire()
    if self.__encrypt:
        crypt = self.enc.encrypt(self.randomize(self.container))
        #add the shuffled and encrypted string to the queue for
        shipping
        self.__shipQ.put((crypt, "b"))
    else:
```

```

        self.__shipQ.put((self.randomize(self.container), "s"))
#clear the string container
self.container = ""
self.__lck.release()
print "rtd_Packer: Package packed for shipment"

```

However, when a received keystroke is added to the container, no locking is done:

```

def run(self):
    while 1:
        try:
            item = self.__packQ.get(True)
        except Queue.Empty:
            #This should never happen!
            print "rtd_Packer: No work to perform?"
        else:
            if item is None:
                #If item is None it means we need to quit
                print "rtd_Packer: received quit item!"
                break
            self.shipTimer.runEvent.set()
            self.container = self.container + item
            if len(self.container) == self.
                __max_containersize:
                print "rtd_Packer: String containerfull!
                    container size:", len(self.container)
                self.ship()
                self.shipTimer.cancelEvent.set()

```

Depending on how Python internally implements the container update, this might result in the loss of collected keystrokes. But in this project this is probably not considered to be a major problem.

## Remark 2 - virgilius

### Finding:

The container handling code has not implemented locking correctly.

### Confirmation:

See section 5.2.4 on the previous page.

### Recommendation:

Implement correct locking when a received keystroke is added to the container.

## 5.2.5 Keylogging process

The keylogging is done in `rtd_Xkeylog.py`. In this file, we read that the code is based on `pyxhook`. We can find this software on SourceForge: <http://pykeylogger.cvs.sourceforge.net/>.

When we compare the code to the original code (`pyxhook.py`), we see that it is now simpler. Only normal keys need to be logged, other events such as mouse events can be ignored.

## 5.2.6 Open UDP port

The client opens a UDP port on the client ("random port number") to communicate with the server. We used the following code to write a string consisting of 20Mbyte worth of "A" characters to the open UDP port on the client, but nothing happened:

```
perl -e 'print "A"x20000000; print "\n"|nc -u 10.42.0.192 1197'
```

This is correct since the UDP port is only used to send out data to the server, nothing is ever read from it.

# Chapter 6

## Security assessment

The purpose of a Technical Security Assessment is to judge the current setup, existence and working (in whole or in part) of the infrastructure that has been identified as the target. This judgement is based on the results of the investigation (the current situation) in comparison with the best practices as included in appendix A on page 32 of this report.

Although there is one high level and one medium level risk identified in the `virgilius` software, we still consider the Retyping Dante application to be secure enough for its purpose.

The high risk is only a problem in combination with other, larger, problems in the clients IT environment. Since this environment is not completely under control of the creators of `virgilius`, we do urge the developers to implement a verification that the client is really communicating with the update server (possibly even using an encrypted channel), but we consider the risk of this problem by itself not high enough to set the security assessment to insufficient or below.

Most medium risks are found in other services running on another system in the `z25.org` environment (i.e. not the collecting server). These cause some threat to that server, but should have nothing to do with the Retyping Dante project or the communication between the clients and the server. The collecting server for the Retyping Dante project will only be used for this purpose (collecting typed characters), and is not shared. A Cisco router is used to forward port 9458/udp to the dedicated server for this project.

# Chapter 7

## Overview of results

In this chapter all results (risks and points of interest) are collected in a complete overview, sorted by risk-level. Every result also refers to the page where more information on the result, and the way it is obtained, can be read. This chapter should provide a clear overview of the results of the assessment including the mitigating actions that should be implemented.

### 7.1 High risks

**High Risk 1 - virgilius** . . . . . 23  
**Finding:** When an attacker controls the host file or the DNS of the victim, or when there is another way of mimicing the update server (e.g. with a man in the middle attack) the update mechanism can be used to execute code on the client system.  
**Risk:** The update mechanism is prone to attacks, enabling the execution of code on the client.  
**Recommendation:** When either situation is true, the victim has a problem that is an order of magnitude bigger than the possibility of executing JavaScript. However, we still advice to have the client verify that it is talking to the correct update server.

### 7.2 Medium risks

**Medium Risk 1 - speakeasy.z25.org** . . . . . 15  
**Finding:** The remote HTTPS service accepts connections encrypted using SSLv2, which reportedly suffers from several cryptographic flaws and has been deprecated for several years.  
**Risk:** An attacker may be able to exploit SSLv2 related issues to conduct man-in-the-middle attacks or decrypt communications between the affected service and clients. See also: <http://www.schneier.com/paper-ssl.pdf>.  
**Recommendation:** We advise the use of SSLv3 or TLSv1 instead.

**Medium Risk 2 - speakeasy.z25.org** . . . . . 18  
**Finding:** The remote IMAPS service accepts connections encrypted using SSLv2, which reportedly suffers from several cryptographic flaws and has been deprecated for several years.  
**Risk:** An attacker may be able to exploit SSLv2 related issues to conduct man-in-the-middle attacks or decrypt communications between the affected service and clients. See also: <http://www.schneier.com/paper-ssl.pdf>.  
**Recommendation:** We advise the use of SSLv3 or TLSv1 instead.

**Medium Risk 3 - virgilius** . . . . . 25

**Finding:** The amount of characters per shipment may be too small, due to the client sending data each 4 seconds maximum.

**Risk:** An attacker with access to the server can possibly reconstruct what is typed on specific clients.

**Recommendation:** Only ship containers that contain a full load of 32 characters.

## 7.3 Low risks

### Low Risk 1 - [speakeasy.z25.org](http://speakeasy.z25.org) . . . . . 13

**Finding:** Too much information on the installed Apache configuration is displayed.

**Risk:** The version information provided by the Apache webserver can be used to search for specific vulnerabilities in installed tools like PHP or Python.

**Recommendation:** Specify `ServerToken ProductOnly` in the `httpd.conf`.

### Low Risk 2 - [speakeasy.z25.org](http://speakeasy.z25.org) . . . . . 14

**Finding:** The web server on the remote host allows read access to `.svn/entries` files.

**Risk:** The webserver exposes all file names in the `svn` module on the website.

**Recommendation:** Configure permissions for the affected web server to deny access to the `.svn` directory.

### Low Risk 3 - [speakeasy.z25.org](http://speakeasy.z25.org) . . . . . 15

**Finding:** The remote HTTPS service uses an SSL certificate that has been signed using a cryptographically weak MD5 hashing algorithm. MD5 is known to be vulnerable to collision attacks.

**Risk:** In theory, a determined attacker may be able to leverage this weakness to generate another certificate with the same digital signature, which could allow him to masquerade as the affected service. See also: <http://www.phreedom.org/research/rogue-ca/>.

**Recommendation:** Create a new certificate using SHA-1 as the signature algorithm.

### Low Risk 4 - [speakeasy.z25.org](http://speakeasy.z25.org) . . . . . 16

**Finding:** The remote HTTPS service may be forced to use weak encryption ciphers that are relatively easy to break.

**Risk:** An attacker may be able to decode the communication between the client(s) and the HTTPS server when weak encryption ciphers are used.

**Recommendation:** We advise to only support high strength ciphers.

### Low Risk 5 - [speakeasy.z25.org](http://speakeasy.z25.org) . . . . . 17

**Finding:** The remote webserver supports the TRACE method. TRACE is a HTTP method that is used to debug web server connections. In addition, it has been shown that servers supporting the TRACE method are subject to cross-site scripting attacks, dubbed XST for "Cross-Site Tracing", when used in conjunction with various weaknesses in browsers.

**Risk:** An attacker may use this flaw to trick legitimate web users to give him their credentials. See also: [http://www.cgisecurity.com/whitehat-mirror/WH-WhitePaper\\_XST\\_ebook.pdf](http://www.cgisecurity.com/whitehat-mirror/WH-WhitePaper_XST_ebook.pdf).

**Recommendation:** Disable the TRACE method via the `TraceEnable` directive. In older versions of Apache one had to filter the methods using constructs like represented in listing 4.7 on page 17.

### Low Risk 6 - [speakeasy.z25.org](http://speakeasy.z25.org) . . . . . 19

**Finding:** The remote host supports the use of anonymous SSL ciphers and supports the use of SSL ciphers that offer either weak encryption or no encryption at all. While this enables an administrator to set up a service that encrypts traffic without having to generate and configure SSL certificates, it offers no way to verify the remote host's identity and renders the service vulnerable to a man-in-the-middle attack.

**Risk:** An attacker may be able to use a MitM attack to access the communication between the client(s) and the HTTPS server.

**Recommendation:** We advise to only support high strength ciphers.

## 7.4 Points of interest

**Remark 1 - virgilius** . . . . . 24

**Finding:** The update server is not up to date; still announcing version 1.26, while the Microsoft Windows version supplied already mentioned version 1.28.

**Recommendation:** Update the update server.

**Remark 2 - virgilius** . . . . . 26

**Finding:** The container handling code has not implemented locking correctly.

**Recommendation:** Implement correct locking when a received keystroke is added to the container.

# Appendix A

## Best practices

In this chapter, we describe the technical security criteria that Madison Gurkha B.V. uses for technical security assessments of IT applications and infrastructures for which no security policy has been made. These criteria are a result of combining *best practices* in the IT industry for comparable environments. In most cases, these criteria relate to measures that have to be taken on an operational level.

The criteria themselves are chosen in such a way that, when an IT environment is tested against these criteria, much security related problems would be detected. This reduces the risk of problems being present in the environment after all. Note that we do not want to indicate that such a test will find all security related problems in an IT environment. A test can only detect certain problems. As time goes on, knowledge of possible security related flaws increases. Also, new security related problems are found, and the IT environment itself will be subject to change. We therefore advise organisations to repeat the Technical Security Assessment at regular intervals and reviewing the criteria being used to test against.

### A.1 Diversity

The security architecture must take several measures that are each different. This way, failing of one security measure does not automatically lead to the failing of the complete system or application. This principle is also known as *defense in depth*.

The Dutch central bank (*De Nederlandsche Bank*) uses these criteria in her prescriptions on building IT infrastructures. The underlying idea is to have multiple layers of security defending the information. By having multiple layers, a single error will not lead to information becoming available to unauthorised persons.

We also like to point out, that at least two levels of security need to be implemented. Besides the installation and correct configuration of a firewall infrastructure, the servers should also be hardened. The application will need to be developed in such a way that abuse is prevented.

### A.2 Redundancy

Components within a system are often duplicated, causing the availability of the system being independent of the availability of a single component. This principle can also be used for security measures. The security architecture should consist of a combination of measures, making the overall security independent of a single measure. This principle thus avoids possible *Single Points of Failure (SPoF)*.

Having a *single point of failure* can lead to unavailability of a service because a part of the overall environment is not redundantly installed. This can be a piece of hardware, but also for example a database full of information for which a back up solution is lacking. Depending on the required availability, a hot standby system or having a spare system on stock can be chosen.

### A.3 Isolation

Isolation means that hardware and software that are relevant for security, i.e. the *Trusted Computing Base (TCB)* must be kept as small and compact as possible. The larger the TCB, the harder it gets to verify that the security of the TCB is guaranteed. This principle has to be implemented in firewalls at a minimum.

In this context, a firewall is a configuration of one or more systems that protect a networking segment against unauthorised access coming from the outside. If the configuration of firewall systems that are part of the IT infrastructure can fail, this can lead to the protected segment becoming fully available to the outside. The firewall systems that are protecting the production networks against unauthorised traffic coming from the outside, should be setup in such a way that a problem with these systems does not result in outsiders being able to get access to the protected network. Also, the firewall systems should be configured in such a way that only traffic that is explicitly allowed is let through. This principle is also known as the principle of safe defaults, or the *deny all, except* principle (see next section).

### A.4 Safe defaults

A system may only allow access after explicit permission has been given. Everything that is not explicitly allowed is denied by default. As described in the previous section, safe defaults can be implemented using the *deny all, except* principle. This principle must be used when configuring network-filtering devices like firewalls.

This principle should of course also be implemented for user-initiated access to systems, programs and data.

### A.5 Input validation

This criterion is especially valid for tailor-made applications. Because of lack of correct input validation, in many cases attackers can abuse the software. In some cases, this may lead to attackers being able to access data that should be inaccessible, or even alter or remove data from the system.

All interactive applications that can be accessed by end-users need to have enough input validation. Specifically, incorrect input should lead to error messages or dilution of stored data only. The Technical Security Assessment will inspect in how far erroneous input can lead to unauthorised access to protected data.

### A.6 Completeness

All forms of access can only take place after the system has authorised the access. Users and processes are required to prove their identity first.

The level of authorisation will of course depend on the value adjudged to the system, the processes and the data that is stored on the system.

## A.7 Restriction of privileges

The system must be set up in such a way that users and processes are unable to perform tasks or use resources other than those that are strictly needed. This principle is also known as *Least Privilege*, *Need To Know* and *Need To Use*.

This principle of least privileges must also be implemented for access to systems. Users should only get access to those parts of the system that they need to. Implementing such restrictions will not prevent break-ins but it will limit the effect of them. The restriction principle can be further implemented by using a high level of function separation.

An example of the *least privilege* principle at work in a back-office environment is implementing restricted permissions. The applications should for instance not communicate with backend systems using (database) administrator rights.

Another example of restriction can be found when trying to limit the amount of information leakage. All information that leaks to an attacker, such as test files, documentation, detailed error messages etc., will make it easier for an attacker to circumvent the security measures. This has to be avoided as much as possible.

## A.8 Function separation

Wherever possible, functions within a system should be split up. The new, smaller functions should be appointed to separate functionaries. A special case of function separation is the *four eye principle*, which means that access to sensitive information can only be given when multiple functionaries have given approval. (Note: Madison Gurkha B.V. always adheres to this principle when assessing systems as well).

## A.9 Compartments

The system must consist of separate compartments, segments or modules. The coupling between compartments must be kept as lean as possible, to improve auditability. This also results in an increase of robustness and hence security of a system.

## A.10 Open design

A well-made security structure is not based on keeping the internal mechanisms that are being used a secret (*Security through Obscurity*). Instead, it is based on an open design. When having a closed design, there is always a risk of the internal mechanisms becoming known, for instance by using *reverse engineering* techniques or the leakage of design specifications and architectural designs. The advantage of having an open design is that it is easier to test extensively and easier to improve.

## A.11 Ergonomics

The system must be designed in such a way that the chance of human mistake is as small as possible. The amount of end-user mistakes can be decreased by designing a user interface that is *intuitive*, by integrating security measures in process steps and performing automatic checks when using the system (such as input validation). Making the tools to do administration and the process of administration itself transparent enough can avoid administrator mistakes, resulting in all those that are involved have an insight in what is expected of them.

## A.12 Simplicity

Complexity is always a factor that works against security. When a system becomes more complex, the risk of making mistakes increases. Not only in the system itself, but also the use of it. Easy to understand systems will in general be safer, or will be easier to make more safe, than complex systems. This principle is also known as the KISS principle, which stands for *Keep It Simple, Stupid*.

## A.13 Protection of data

Data that is being stored on systems, e.g. in databases, are very valuable. Access to this data needs to be regulated with some form of access control (this is equally valid in case the data is stored on backup tapes). The access control will at a minimum consist of a form of identification (a user name for example) and a form of authentication (a password for example). To always have the correct data available, it is important to have a rollback mechanism. Decommissioning hardware and software may never lead to data being available to a third party. Examples are decommissioned hardware that still has data on the hard drive, but also decommissioned backup tapes.

The logging of both successful and failed authentication attempts is an important tool for detection and tracing of unauthorised authentication requests. A clear login message may be of help in case an unauthorised login attempt is brought before justice.

During a Technical Security Assessment, the security of data that is available on the operational systems will be reviewed. The other aspects we just mentioned are part of the procedures at the organisation. Protected (personal) data may not be accessible to third parties. Only authorised users must be able to change their own data.

## A.14 Availability of data

Data has to be available as much as possible. The availability will be improved if all systems involved behave in a fault-tolerant way, and are able to withstand *Denial of Service (DoS)* attacks.

During a Technical Security Assessment, the ways in which the availability can be negatively influenced will be investigated.

With these results, the investigators can create a clear view of places that problems can arise, and in what way these problems can be avoided and/or solved.

## A.15 Protection of communication

Depending on the type of data that is being sent, it can be important to take measures to find out if the sender is the one she claims to be (*anti-spoofing* measures – authentication) and/or to find out if the receiver is the one she claims to be. There might also be the need to protect the communications channel itself against eavesdropping and modification by a third party. Securing communications against eavesdropping can be important to the party providing the data, but also for the party receiving the data, for instance because of privacy considerations.

## A.16 Additional protection of server systems

This principle follows from the aforementioned *Defense In Depth* principle. Every server must be set up to deliver one or more specific services. Another name for this principle is *hardening*. A standard installation of a server is inadequate because in most cases this will enable more services than necessary, which can lead to security risks.

Also, care must be taken to set up servers that provide the same services the same way. On a related note, security *patches* should be installed in time.

A Technical Security Assessment will investigate the current configurations of the server systems and make recommendations on necessary changes.

## Appendix

## B

## Additional sources of information

For Madison Gurkha B.V., it is important to keep its knowledge up to date. Besides study and internal projects, our professionals regularly use internet sources for new and interesting information. Some of the sources we regularly use are listed in this appendix. With this list, we can help employees in your organisation to continually keep their knowledge that is so important for IT security, up to date.

Of course, such a list can never be complete. It must be seen as a basic list of information sources. Madison Gurkha B.V. tries to update this list regularly. The list is presented in a random order.

### B.1 Websites

Besides the main website of Madison Gurkha B.V., on which new articles and podcast regularly appear (<http://www.madison-gurkha.com/>) and the secure website, from which customers can securely download reports (<https://cfe.madison-gurkha.com/>), we regularly make use of information from, amongst others, the following sources:

URL	Korte omschrijving
<a href="http://www.owasp.org/index.php/Category:OWASP_Project">http://www.owasp.org/index.php/Category:OWASP_Project</a>	Open Web Application Security Project(s)
<a href="http://www.oisssg.org/content/view/7/71">http://www.oisssg.org/content/view/7/71</a>	Information Systems Security Assessment Framework
<a href="http://www.isecom.org/osstmm/">http://www.isecom.org/osstmm/</a>	Open Source Security Testing Methodology Manual
<a href="https://www.pcisecuritystandards.org/">https://www.pcisecuritystandards.org/</a>	PCI Data Security Standard
<a href="http://www.nsa.gov/snac/index.cfm?MenuID=scg10.3.1">http://www.nsa.gov/snac/index.cfm?MenuID=scg10.3.1</a>	Security Configuration Guides
<a href="http://iase.disa.mil/stigs/stig/index.html">http://iase.disa.mil/stigs/stig/index.html</a>	Security Technical Implementation Guides
<a href="http://www.sans.org/free_resources.php">http://www.sans.org/free_resources.php</a>	SysAdmin, Audit, Network, Security Institute – Resources
<a href="http://csrc.nist.gov/publications/">http://csrc.nist.gov/publications/</a>	National Institute of Standards and Technology – Computer Security Resource Center
<a href="http://osvdb.org/">http://osvdb.org/</a>	The Open Source Vulnerability Database
<a href="http://www.govcert.nl/">http://www.govcert.nl/</a>	Computer Emergency Response Team of the Dutch government

Table B.1: Overview interweb resources.

## B.2 Mailing-lists

Every knowledgeable security professional will divulge that breaking news is not published on websites. The latest information will, on the internet, be published in mailing-lists and can be located using up-to-date archives thereof. This is the reason we have included the following enumeration of lists that are regularly read by our consultants:

<b>Mailinglist address</b>	<b>Korte omschrijving</b>
security-basics@securityfocus.com	Security Basics
bugtraq@securityfocus.com	Bug Tracking
cert-advisory@cert.org	US-CERT Advisories
forensics@securityfocus.com	Forensics
framework-hackers@spool.metasploit.com	Metasploit framework development
full-disclosure@lists.grok.org.uk	Full disclosure security
incidents@securityfocus.com	Incidents
pen-test@securityfocus.com	Penetration testing
NewsBites@sans.org	News bites from SANS
webappsec@securityfocus.com	Web application security

Table B.2: Overview mailinglist resources.

# Appendix C

## Used acronyms

In our profession acronyms are often used. Because the meaning may not always be directly clear, this appendix tries to provide an overview of all acronyms used in this document.

<b>dig</b>	Domain Name Gopher
<b>DIN</b>	Deutsches Institut für Normung e.V.
<b>DNS</b>	Domain Name Service
<b>DoS</b>	Denial of Service
<b>HTTP</b>	HyperText Transfer Protocol
<b>HTTPS</b>	Secure HTTP
<b>ICT</b>	Information and Communication Technology
<b>IMAP</b>	Internet Message Access Protocol
<b>IMAPS</b>	Secure IMAP
<b>IT</b>	Information Technology
<b>KISS</b>	Keep It Simple, Stupid
<b>MitM</b>	Man-in-the-Middle
<b>mtr</b>	My Trace Route
<b>OSC</b>	OpenSound Control
<b>SMTP</b>	Simple Mail Transfer Protocol
<b>SPoF</b>	Single Point of Failure
<b>SSH</b>	Secure Shell
<b>TCB</b>	Trusted Computing Base