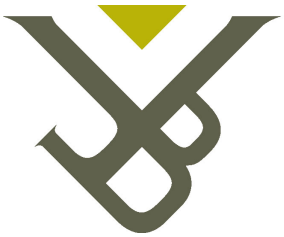

Vrije Universiteit Brussel

Faculty of Science and Bio-engineering Sciences

Computer Sience Department

Dean: Prof. dr. P. Geerlings

Security aspects in virtual networks

by

Laurent De Wilde

Promotor: Prof. dr. ir. Martin Timmerman

A thesis submitted in partial fulfillment for the degree of

Master Of Science in the Applied Sciences and Engineering: Computer Science

Academic year 2014 - 2015

Preface

This is where the preface will come. . . .

Laurent De Wilde, June 2015

Copyright declaration

Laurent De Wilde, June 2015

Security aspects in virtual networks

by

Laurent De Wilde

A thesis submitted in partial fulfillment for the degree of
Master Of Science in the Applied Sciences and Engineering: Computer Science

Academic year 2014 - 2015

Promotor: Prof. dr. ir. Martin Timmerman
Faculty of Sience and Bio-engineering Sciences
VRIJE UNIVERSITEIT BRUSSEL

Computer Science Department
Dean: Prof. dr. P. Geerlings

Summary

Here comes the summary of the thesis. . . .

Keywords

Security, virtual machines, virtual networks, Hyper-V, Windows, virtualization.

Security aspects in virtual networks

Laurent De Wilde

Supervisor(s): Prof. dr. ir. Martin Timmerman

Abstract—**Here comes a short abstract**

Keywords—**Security, virtual machines, virtual networks, Hyper-V, Windows, virtualization.**

I. Introduction

HERE comes the extended abstract.

II. Virtual networks

Some text ....

III. Conclusion

References

[1] Andrew Stuart Tanenbaum and Todd Austin *Structured Computer Organization*, vol. 6, pp. 2-7, New Jersey, U.S.A., 2013

Contents

Preface i

Copyright declaration ii

Overview iii

Extended abstract iv

Contents v

List of Figures viii

List of Tables xvi

List of Abbreviations xvii

1 Introduction 1
- 1.1 Virtualization . . . 1
  - 1.1.1 Abstractions . . . 1
  - 1.1.2 Abstraction and virtual machines . . . 3
- 1.2 Why use virtualization . . . 4
  - 1.2.1 Benefits of using machine virtualization . . . 4
  - 1.2.2 Challenges and disadvantages of using machine virtualization . . . 5
- 1.3 Security issues regarding virtualization . . . 6

2 Virtualization 7
- 2.1 Machine virtualization - Hypervisors . . . 7
  - 2.1.1 Bare-metal hypervisors . . . 8
  - 2.1.2 Hosted hypervisors . . . 14
- 2.2 Network virtualization . . . 14
  - 2.2.1 External virtual network . . . 15
  - 2.2.2 Internal virtual network . . . 16

2.2.3 Private virtual network . . . 16
2.2.4 External network virtualization . . . 17
2.3 Storage virtualization . . . 19
2.3.1 Host-based storage virtualization . . . 19
2.3.2 Virtual hard disks in Hyper-V . . . 20

3 Security aspects **21**
3.1 Known non-network related security issues . . . 21
3.1.1 Virusses and malware problems . . . 21
3.2 Known network related security issues . . . 23
3.3 Possible solutions . . . 23
3.4 Sniffing on virtual networks . . . 24
3.4.1 First research question: sniffing a virtual NIC . . . 24
3.4.2 Penetration testing on virtual networks . . . 32
3.4.3 Actual penetration testing . . . 44
3.5 Security of dual-boot systems . . . 73
3.5.1 Second research question: hacking of dual-boot systems . . . 73
3.5.2 Preventing access to a (virtual) hard disk . . . 82
3.5.3 Downsides of using BitLocker . . . 88

4 Installation of a private cloud **94**
4.1 Installation of Microsoft System Center 2012 R2 Virtual Machine Manager 94
4.1.1 Installation requirements . . . 94
4.1.2 Generatl installation overview . . . 95
4.1.3 Pre-installation configuration . . . 95
4.1.4 Installation of MSSQL Server 2012 SP2 . . . 98
4.1.5 Configuring Distributed Key Management in Active Directory . . . 104
4.1.6 Post configuration of MSSQL Server . . . 110
4.1.7 Installation of VMM . . . 113
4.2 Installation of Windows Azure Pack . . . 122
4.2.1 Prerequisites . . . 123
4.2.2 Post Installation of Service Provider Foundation . . . 137
4.2.3 Installation of the Windows Azure Pack . . . 141

5 Conclusions and recommendations **149**

A Test lab **150**
A.1 General description . . . 150
A.2 Technical description . . . 151
A.2.1 Hardware overview . . . 151

A.2.2 Networking overview 151
A.3 Visualization of the test lab 153

Bibliography **154**

List of Figures

1.1 Abstraction applied to disk storage 2
1.2 The OSI model 3
1.3 Virtualization of hard disks 4

2.1 Bare-metal hypervisor 8
2.2 Hyper-V Architecture 10
2.3 Xen architecture 12
2.4 VMware architecture 13
2.5 VMware simplified architecture 14
2.6 Microsoft Virtual Switch concept 16
2.7 External Virtual Network 17
2.8 Internal Virtual Network 18
2.9 Private Virtual Network 18
2.10 Logical Volume Management 19

3.1 VM based rootkit 22
3.2 The network setup used to investigate packet sniffing. 25
3.3 PowerShell commands to configure port monitoring. The vNICs of `Server 1` and `Server 2` are set as 'source', whereas the vNIC of monitoringVM is set as 'destination'. 26
3.4 Traffic captured on the Hyper-V virtual network prior to executing the ICMP Ping requests. 27
3.5 ICMP Ping traffic captured by the monitoring VM. Besides this, some DNS traffic can be observed as well. 28
3.6 ICMP Ping traffic from `Server 1` (192.168.1.51) to the Hyper-V host (192.168.1.6). Note that other 'external' traffic is captured as well. In this case, traffic from `Server1` to a wireless access point. 29

3.7 ICMP Ping traffic captured from a Xen VM to a Hyper-V VM, in this case `Server1`. 30
3.8 ICMP Ping traffic captured from a Xen VM to the Hyper-V host. 31
3.9 Confirmation of the correct installation of Snort. 36
3.10 The sniffing / capturing interface `eth1` runs in promiscious mode and captures all the packets on the network. No IP address has been set (as it should be). 36
3.11 Confirmation of the network settings. 37
3.12 We are working on the 192.168.1.0/24 network, but by means of testing, we have setup Snort to listen on the /16 subnet (255.255.0.0). As it will turn out, this will **not** affect the correct working of Snort. So one could randomly choose a value (8, 16 or 24). 38
3.13 After PulledPort has run, one can notice that 19 new rules have been downloaded and added to the ruleset of Snort. 40
3.14 Everything is running fine. 40
3.15 Ping traffic gets picked up by Snort. 41
3.16 `tcpdump` confirms that the Snort VM is receiving traffic other than the traffic destinated for the VM. This proves the fact that the Snort VM is actually sniffing traffic. 42
3.17 Traffic originating from a Xen VM to the Xen host is captured by the Snort VM as well as traffic originating from a Xen VM to a Hyper-V VM. 43
3.18 Traffic from the Hyper-V host (the hypervisor where the Snort VM is installed on) to a Xen VM is captured by Snort. 43
3.19 Traffic between the two hypervisors is picked up as well. 43
3.20 Traffic originating from outside either virtual network to any virtual network and traffic destinated to a target outside any virtual network is detected by Snort. 44
3.21 Internal traffic between Xen VM's is intercepted by Snort. 44
3.22 The NMAP command as executed on the webserver (atlas, 192.168.1.11). . 46
3.23 Basic, unfragmented NMAP scanning of a Hyper-V VM (192.168.1.51). The SnortVM has the IP address of 192.168.1.50. Snort reports each attempt to scan a particular port number. 47
3.24 The NMAP stealth, SYN packet command as executed on the webserver (atlas, 192.168.1.11). 47
3.25 The scan for running services from the stealth scan is detected by Snort. . 48

3.26 However, this Snort alert indicates that a host reassembling a fragment datagram cannot complete the reassembly due to missing fragments whitin the time limit (60s by default). However, I'm not sure whether this is Snort warning for a fragmented / stealth scan. 48
3.27 ICMP ping with large packet size is detected by Snort. 48
3.28 OS detection from a NMAP scan is also detected by Snort. 49
3.29 The vulnerable webpage before the attack. It displays the string value that is provided in the "name" parameter in the URL. 50
3.30 The script in action. 51
3.31 The script is successfully injected into the PHP page. 51
3.32 Fortunately, Snort detects the XSS attack without the need for adding additional rules. 52
3.33 To prevent XSS attacks from happening, one can change "alert" to "drop" in the rules that triggers the alert. 52
3.34 Persons table in the test database running on MySQL 5.6 populated with 3 records. 53
3.35 Persons table in the test database running on MySQL 5.6 populated with 3 records. 54
3.36 Persons table in the test database running on MySQL 5.6 populated with 3 records. 54
3.37 SQL injection attack in action with all the results displayed on the webpage. 55
3.38 Snort alerts for a possible SQL injection. 55
3.39 The triggering rule. Once again, one could change "alert" to "drop" in order to prevent the SQL injection from happening. 55
3.40 The source code of the webpage. Note the "shell_exec" statement. 56
3.41 A command injection attack has just been launched and the results of executing the `ipconfig /all` command are displayed on the webpage. 57
3.42 Output of the `netstat -a` command displayed on the web page. 57
3.43 Snort alerting for command injections. 58
3.44 First, I created a rule to actually detect FTP traffic as I plan to DOS attack the FTP server is a later stage. The starting of the FTP service and some FTP traffic are detected by Snort. 59
3.45 Successful FTP logins are also detected by Snort (however, this is not a thread and can be disabled by simply comment the rule that triggered the alert. 59
3.46 Attempting to login as root. 59

3.47 FTP root access is successfully detected. 60
3.48 The command to attack the FTP server as seen in Metasploit. 60
3.49 Snort reported the various attacks. 61
3.50 Appearantly, someone tried to SSH scan my Xen server.... This was fortunately detected by Snort. 61
3.51 An example of `hosts.allow`. In this example, only computer with IP address of 192.168.1.2 and 192.168.1.2 are allowed to connect to the server through SSH. 62
3.52 An example of `hosts.deny`. This file is used in combination with `hosts.allow` and indicates that only IP addresses of 192.168.1.2 and 192.168.1.40 are allowed to connect to the server through SSH and that all other IP addresses have their access denied. 62
3.53 Metasploit is scanning the network for databases... 63
3.54 ... and this is detected by Snort. 63
3.55 This is captured by Snort. 64
3.56 Also logging in a root is detected by Snort. 64
3.57 The plugin to create the malicious payload. 65
3.58 The actual creation of the malicious payload. The "LHOST" stands for Local HOST and indicates that the trojan makes a connection with my (attacking) computer via port 4444. 65
3.59 Preparing the listener for when an unsuspicious user clicks on the file. . . . 65
3.60 A user clicks on the file and a connection between my computer and the victim is established. 66
3.61 Now I can for example browse the hard disk drive of the victim's computer... 67
3.62 ...or obtain some network information to prepare for subsequent attacks. . 68
3.63 Fortunately, this is detected by Snort. 68
3.64 The FTP server receives a lot of login attemps per second. This way, we hope to flood it and eventually make it go offline. 69
3.65 Snort reacts. 70
3.66 The DOS attack on the webserver in action... 70
3.67 Fortunately, this is detected by Snort. 71
3.68 The apt updating process is seen as a thread by Snort. 72
3.69 Metasploit's updating process is known by Snort... 72
3.70 Downloading an .exe file from the Internet is also seen and reported by Snort. 72
3.71 Network infrastructure as of the beginning of research question number two 74

3.72 Mounting the virtual hard disk in Windows Server 2012 R2 using Windows Explorer . . . 75
3.73 . . . after which the partitions become visible (browseable). . . . 76
3.74 Creation of the malicious Trojan. . . . 77
3.75 Transferred the Trojan to the host by means of a shared folder. . . . 77
3.76 Once the VM has started, it connects automatically to my computer and I can browse the files, even when the hard drive is in use and not mountable anymore in Windows Explorer. . . . 78
3.77 An example of the directory listing of the compromised VM. . . . 78
3.78 The two disks of the two OS's visible in Windows Explorer. . . . 79
3.79 The trojan is inserted in the other OS of the dualboot system. . . . 79
3.80 The Trojan has connected to our computer. . . . 80
3.81 The parameters are set up correctly and the module is runned. . . . 80
3.82 . . . resulting in a BSOD. . . . 81
3.83 The BitLocker settings in the group policy of Windows Server 2012 R2. . . 83
3.84 A 256 bit encryption is chosen for maximum protection. . . . 83
3.85 Also non-system drives can be secured. . . . 84
3.86 This setting forces the requirement of entering a password when the VM is started. . . . 84
3.87 When the settings are set, the VM is rebooted and "BitLocker drive encryption" is selected in the Configuration Panel. Since we want to encrypt the entire VM (and therefore preventing the disk from mounting), the "C:" drive is selected. . . . 85
3.88 When the encryption of the system drive has finished, the VM is rebooted and a password is prompted when one wants to boot the VM. . . . 86
3.89 Without knowing the password, it is impossible to login. . . . Of course, this prompting for a password at boot time can be disabled in the Group Policy Editor. . . . 87
3.90 Now we try to mount the virtual hard drive, but a password is required to do so. . . . 87
3.91 Only when the correct password is supplied, one is able to access the files on the virtual disk. . . . 88
3.92 Read - and write speeds according to "CrystalDiskMark" before the drive has been encrypted by BitLocker. Note that I use an actual system (virtual) disk. . . . 89
3.93 Here I use HDTune Pro to perform the benchmarking. . . . 90

3.94 These are the results after the system drive has been encrypted using BitLocker. The sequential read speed drops from 117 MB/s to 88 MB/s and the sequential write speed drops from 34 MB/s to 30 MB/s. 91
3.95 Also HDTune confirms the speed drop: from 100 MB/s on average to 91 MB/s on average. 92

4.1 configuring static IP settings. 96
4.2 Setting an appropriate server name and joining the server to the AD domain. 97
4.3 Confirmation of successfully joining the AD domain. 97
4.4 The overview screen of the local server with a proper server name set, the server being joined to the domain, the firewall and remote desktop both being enabled. Also, a static IP has been set and the IE Enhanced Security Configuration has been disabled. Now we are ready to install Microsoft SQL Server 2012 SP2. 98
4.5 `netfx3` has been installed successfully. 98
4.6 Extract the image using WinRAR and run setup.exe. 99
4.7 In the left pane, select “Installation” and subsequently in the right pane, select “New SQL Server stand-alone configuration”. 99
4.8 Setup files are being installed. 100
4.9 Checking additional prerequisites. 101
4.10 Selection of the features. 102
4.11 Windows Authentication Mode is chosen and the SQL administrators are added. 103
4.12 Setup has completed successfully. 104
4.13 Making the service account for DKM. 105
4.14 The properties of the service account. 106
4.15 Create a new Object. 107
4.16 Creation of the container. 107
4.17 Creation of the container. 108
4.18 Configuration of the permissions for VMMDKM. 109
4.19 Adding the VMMService account to the local administrators group. 109
4.20 Login into SQL Server Management Studio using Windows Authentication. 110
4.21 Creation of a new login. 111
4.22 Selecting the VMMService account. 112
4.23 Confirmation. 112
4.24 Download the Windows ADK for Windows 8.1. 113

4.25 Install the ADK on the current compute, since VMM will also be installed on this local computer. . . . 114
4.26 Install only the Deployment Tools and Windows PE. . . . 115
4.27 Extract the files and double click on `setup.exe`. . . . 116
4.28 The main screen of Virtual Machine Manager 2012 R2 is shown. . . . 116
4.29 Select the VMM management server. . . . 117
4.30 Choose the installation location. . . . 118
4.31 Configuration of the database. . . . 119
4.32 120
4.33 Specify the location of the shared libraries. . . . 121
4.34 Overview of the configuration settings. Click `Install`. . . . 122
4.35 Overview screen of the local server showing the begin situation. . . . 123
4.36 Authentication Mode: Mixed Mode. . . . 124
4.37 Install only the VMM console. . . . 125
4.38 Select the `Web Server (IIS)` Server Role. . . . 126
4.39 Select `HTTP Activation`. . . . 127
4.40 Select `Management OData IIS Extension`. . . . 127
4.41 128
4.42 Download and install WCF Data Services for OData V3. . . . 129
4.43 Download and install ASP.NET MVC 4. . . . 129
4.44 The properties of the domain account `SPFService`. . . . 130
4.45 The four global security groups have been made. . . . 131
4.46 Create a self-singed certificate. . . . 131
4.47 SPF can be found on the SC Orchestrator installation media. . . . 132
4.48 The prerequisites are checked. When everything is installed, setup can be continued. . . . 133
4.49 Configuration of the database. A newly created instance can be used, as well as the instance created earlier when setting up the VMM virtual machine. 134
4.50 Certificate selection. Choose the self-singed certificate created earlier. . . . 134
4.51 Configuration of the `Admin` virtual directory of IIS. Choose the `SPFAdmin` and the `SPFService` service account, both created earlier in the pre-setup process. . . . 135
4.52 Configuration of the `Provider` virtual directory of IIS. Choose the `SPFProvider` and the `SPFService` service account, both created earlier in the pre-setup process. . . . 136

4.53 Add the SPFService account to the local groups created by the Service Provider Foundation installation. . . . 137
4.54 The SPFService account needs administrative permissions in Virtual Machine Manager. . . . 138
4.55 Adding administrative permissions to the SPFService account. . . . 139
4.56 Verifying the correct settings of the Application Pools. . . . 139
4.57 Creation of the local SPF_REG account. . . . 140
4.58 The account has been added to the four local groups created by the SPF installer. . . . 140
4.59 Selection of the WAP Portal and API Express. . . . 141
4.60 Click Next. . . . 142
4.61 The installation has completed successfully. Click Finish. . . . 142
4.62 Configuration of the database access for use with WAP. . . . 143
4.63 Setup has completed successfully. . . . 144
4.64 Login using the local administrator account. . . . 145
4.65 The main screen of WAP. . . . 145
4.66 Before setting up a VM cloud, the SPF Endpoint needs to be registered. . . . 146
4.67 In case of failure, reset the password for the SPF_REG account. . . . 147
4.68 Restart the Web Server 147
4.69 . . . and everything should work fine now. . . . 147
4.70 Connect to the VMM server. . . . 148
4.71 The cloud is now visible in VM clouds. . . . 148

A.1 The network setup 153

List of Tables

A.1 Overview of the hardware used in the test lab 151

List of Abbreviations

DKM	**D**istributed **K**ey **M**anagement
DMA	**D**irect **M**emory **A**ccess
ESXi	**E**lastic **S**ky **X** **i**ntegrated
HIDS	**H**ost **I**ntrusion **D**etection **S**ystem
ICMP	**I**nternet **C**ontrol **M**essage **P**rotocol
OS	**O**perating **S**ystem
OSI	**O**pen **S**ystems **I**nterconnection
NIC	**N**etwork **I**nterface **C**ard
NIDS	**N**etwork **I**ntrusion **D**etection **S**ystem
VM	**V**irtual **M**achine
VMBR	**V**irtual **M**achine **B**ased **R**ootkit
VMM	**V**irtual **M**achine **M**onitor
VMX	**V**irtual **M**achine e**X**tensions

Chapter 1

Introduction

In the introductory chapter, some basic aspects of virtualization will pass the revue.

1.1 Virtualization

1.1.1 Abstractions

Computer systems are built as hierarchies with interfaces that separate levels of abstraction. Those levels of abstraction hide lower-level implementation details, which allows for independant development of each seperate layer and thus simplifying the development and maintanance process [Smith and Nair, 2008].

To clearify this layered build of computer systems, two examples are provided: one applies to application design, whereas the second example applies to disk storage.

First example Imaging a higher-level programming language, called **Language1**. This language consists of classes, methods, variables, etc.... This Language5 is very convenient and understandable for people. However, since computer systems only understand machine language, let us call this **Language1**, one cannot simply feed this language to the low-level digital circuits and hoping for the program to execute.

In fact, the gap between what is conventient for people and what is convenient for computers tend to be very large. So the need to translate the higher-level code (Language5) to low-level bytecode (Language1) is needed. This is where abstractation layers come in.

Imaging a virtual machine **Machine1** that accepts Language5. The programs written in Language5 can then be translated of interpreted by a program written in Language1, which can be directly executed on the computer hardware [Tanenbaum and Austin, 2013].

However, for practical reasons, those two languages Language5 and Language1 cannot be too different. This means that intermediate Languages and thus virtual machines must exist to interpret of translate Language5 into a more lower-level language **Language4**. This process of continually and gradually translating higher-level languages into lower-level languages can further be executed until the final, lowest-level language (Language0) is reached [Tanenbaum and Austin, 2013].

Each virtual machine is a layer. So in the example given, 5 layers exist to translate a higher-level language into a lower-level one. The bottom-most language is the simplest one that computers understand (i.e.: bytecode) whereas the top-most language is the most sophisticated one.

Second example In the case of hard disk abstraction, the operating systems hides (i.e.: abstracts) the addressing details, that is, sectors and tracks, for the application software. This means that, from the application point-of-view, the disk appears as a set of files.

Programmers create, read, modify and delete files without knowing those low-level tracks and sectors. The figure below clearyfies the process.

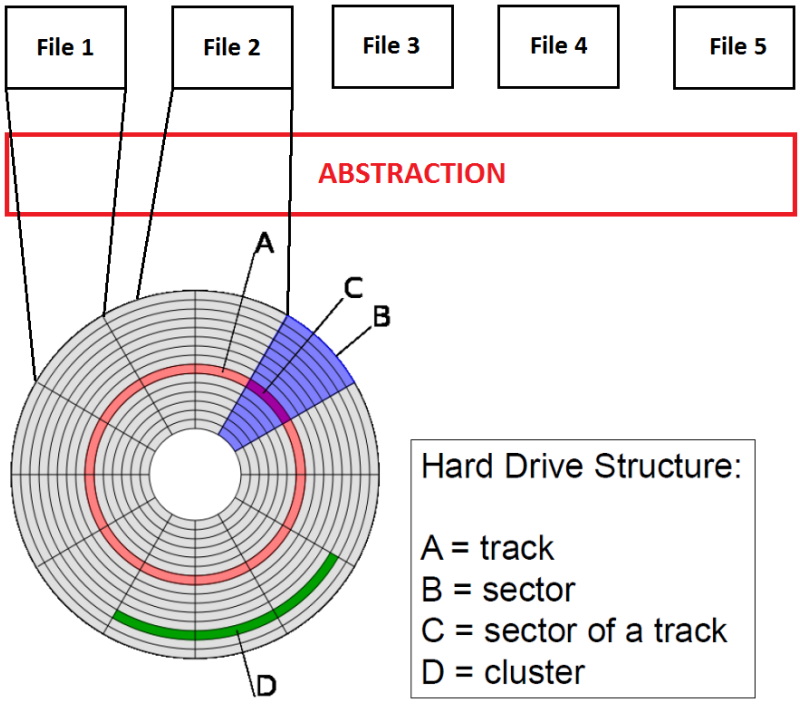

Figure 1.1: Abstraction applied to disk storage. The abstraction layer is in fact the operating system.

In fact, not only computer systems are built as hierarchies: also networking systems use several abstractation layers to communicate with each other. What follows is a brief explanation of the OSI model, that describes how network applications may communicate with each other [Briscoe, 2008].

The OSI model consists of 7 layers as illustrated in the figure below: Top-level applica-

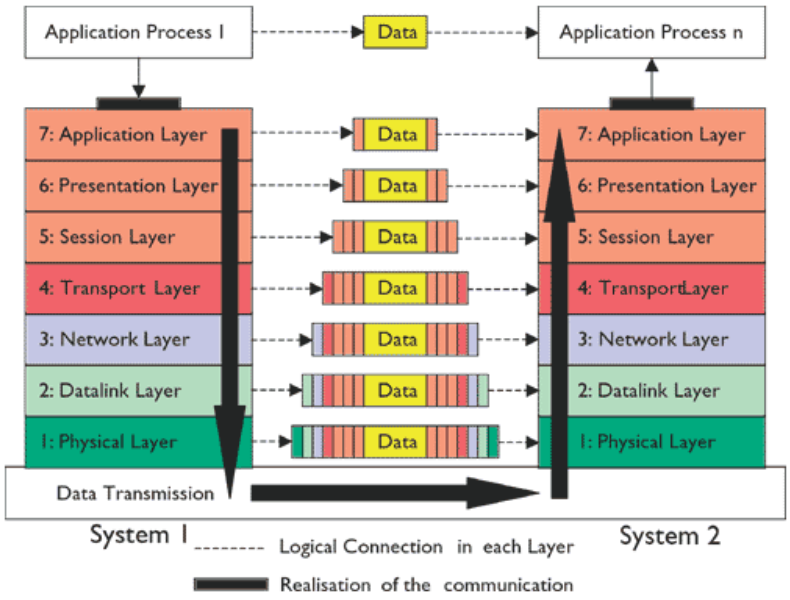

Figure 1.2: The OSI model is another example of abstraction. This time not in computer systems, but in networking.

tions do not communicate directly with each other. Instead, data is passed from one layer to another, starting at the application layer and proceeding to the bottom layer. There, the data is sent over the communication channel to the other host where the whole process takes place in reverse order [Beal, 2015].

The seven layers can be seen as abstraction layers. Each layer hides details of the level directly beneith it.

1.1.2 Abstraction and virtual machines

Virtualization exists in many forms. Not only there exist storage (disk) virtualization, but also network virtualization, virtualized applications and hypervisors. In the first case, virtualization does not necessarily aim to hide details [Smith and Nair, 2008].

Consider the figure below. Virtualization transforms a physical disk into two smaller disks. Each of those disks appears to have its own tracks, sectors and clusters. Furthermore,

the virtualization software uses the file abstraction described in the previous section to map a virtual disk onto the real physical disk.

Data that is to be written onto the virtual disk, is converted to a file write. Since the file resides on the real physical disk, data is actually written to the physical disk.

This is an example of abstraction and virtualization applied to disk storage. Obviously,

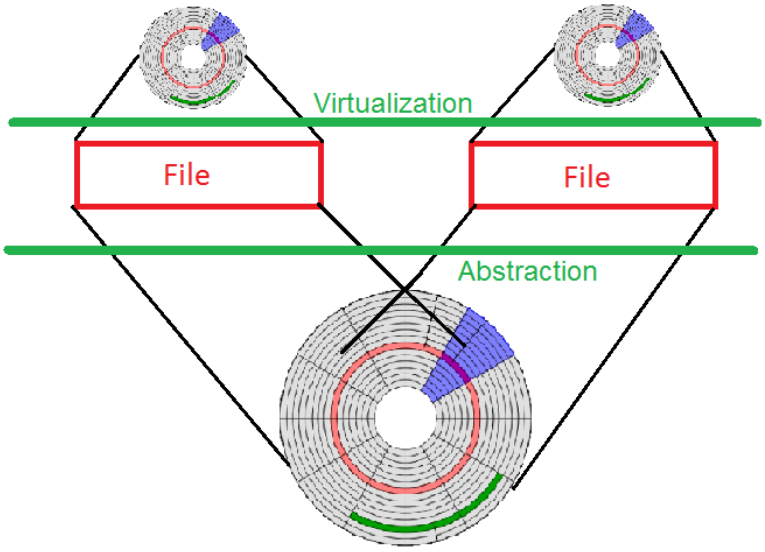

Figure 1.3: Virtualization applied to disk storage

the whole concept of (disk) virtualization can also be applied to physical machines. This is where hypervisors make their entry.

A hypervisor will abstract the physical resources, that is, CPU, memory, disk and network, from systems running on top of it [Vanover, 2013]. This means that those physical resources are shared between the virtual machines and thus allows that multiple virtual machines can run on a single physical machine.

Chapter 2 provides more details about virtualization and hypervisors in particular.

1.2 Why use virtualization

1.2.1 Benefits of using machine virtualization

- **Server consolidation** The most prominent advantage of using machine virtualization (where a hypervisor is used to abstract the physical machine resources) is server consolidation. In the case of a typical non-virtualized application server for example,

about 5% to 10% of the server's hardware is utilized. However, when a server hosts multiple virtual machines, utilization can reach values of 50% to 80% [Bigelow, 2009].

The whole point of machine virtualization is that the hardware of the physical machine is used more efficiently. It permits to get more out of the existing, physical hardware, because multiple virtual machines that act as real servers can run on top of one physical machine. This has an important consequence: fewer physical servers can be used to achieve the same goals. This means lower operating costs, e.g.: less power and air conditioning are needed [VMware, 2015b].

- **Easy cloning** In contrast to a physical server, which consists of a mixture of application files, OS files, driver files and user files, a VM exists as a single file [Bigelow, 2009]. This file can be duplicated, which leads to easy creation of exactly the same machine (server). Obviously, these images can be modified for each application [Vogel, 2014].

- **High availability** When running multiple virtual machines, the load can be distributed amongst them. When a VM fails, another VM can just be started with ensures minimal downtime or data loss [Vogel, 2014].

- **Scalability** Scalability can greatly be improved using virtual machines. Additional resources can quickly be allocated from the host to the guest [Microsoft, 2015a]: when a certain task requires more RAM, adding RAM to the virtual machine consist of editing a parameter on the hypervisor, whereas in the case of a physical machine, it can take minutes to add more RAM [Vogel, 2014].

1.2.2 Challenges and disadvantages of using machine virtualization

- **Longer recovery in case of physical hardware failure** and therefore longer downtime. When the physical machine breaks down because of a catastrophic hardware failure, all the virtual machines need to wait until the physical host is brung up online, after which the VM's can start booting. This means increased downtime [Vogel, 2014].

1.3 Security issues regarding virtualization

Consider the VMware ESXi Hypervisor. All virtual machines are isolated from each other [VMware, 2015a]. The advantages of this practise are for example the fact that if one VM fails, the remaining VM's remain accessible or the fact that if one VM gets infected with, let's say, a Trojan Horse, this will not affect the other VM's [Prowse, 2014].

However, this is not entirely true: cases have been reported in where virusses are able to break out VM's [Wang, 2009]. Additionaly, since VM are networked - either internally using a virtual switch or bridged with the physical network (or both), a new thread arises. Malware with a network component (e.g.: worms), travels where their routing tells - or allows - them to go [Marcin, 2011]. Therefore , it is perfectly possible that an infected VM might infect other VM's while the network administrator believed this could never happen.

The whole point is that one may not assume that VM's are really isolated from each other and that an infected VM cannot infect another one - even if they are not networked. Chapter 3 will cover security in more detail.

In this thesis, virtual networks will be tested against known and unknown security problems concerning virtual networks. Furthermore, securtity aspects will be highlighted in virtual networks.

This chapter briefly described some basic concepts of virtualization, especially machine virtualization. Advantages and disadvantages of virtualization have been discussed as well as a brief mention about security issues related to virtualization.
In the next chapter,)

Chapter 2

Virtualization

In this chapter, (the) three types of virtualization will be explained in detail: machine virtualization, network virtualization and storage virtualization.

Before digging deeper into security aspects in virtual networks, some types of virtualization must first be defined.

2.1 Machine virtualization - Hypervisors

In order to be able to run virtual machines, a hypervisor is needed. Chapter 1 already described the function of a hypervisor, also called a virtual machine manager [House, 2006]: a hypervisor abstracts the physical resources - CPU, memory, disks, network adapters, ...- from systems running on top of it. So it allows for a physical machine (that we will call a host) to share their hardware resources amongst virtual machines (that we will call guests) running as guests on top of the host (the physical hardware). Note that a hypervisor is nothing more than a piece of software [Kleyman, 2012] that ensures that VM's do not interrupt each other [House, 2006].

Each VM appears to have it's own processor, memory and disk. However, the hypervisor is actually controlling the VM's resources and allocating the resources to their needs [House, 2006].

There exist two types of hypervisors: **Type 1** and **Type 2** hypervisors [Siebert, 2006]. Type 1 hypervisors are the so-called **Bare-metal hypervisors** whereas type 2 hypervisors are known as **hosted hypervisors**. Both will be discussed in the upcoming sections.

From know on, the physical machine will be called the **host** and the virtual machine that runs on top of the host (through the hypervisor), will be called the **guest**.

2.1.1 Bare-metal hypervisors

The name 'bare-metal' ("without an operating system") comes from the fact that this type of hypervisor is deployed as a bare-metal installation. This implies that it is not required to first install a server operating system: the hypervisor is the first thing to be installed on the host [Siebert, 2006]. To be precise: the hypervisor is installed as the operating system of the host [Kleyman, 2012].

A bare-metal hypervisor runs directly on the host's hardware and therefore allow for direct access to the hardware recourses, which results in greater performance compared to hosted hypervisors [Siebert, 2006] as illustrated in the picture below. One point of remark: the VM's running on top of the hypervisor do not have direct access to the hardware recourses. Instead, they have a virtual view of for example the processor and run in a private memory address region that is unique for each guest [MSDN, 2015]. Only the hypervisor has direct access to the hardware.

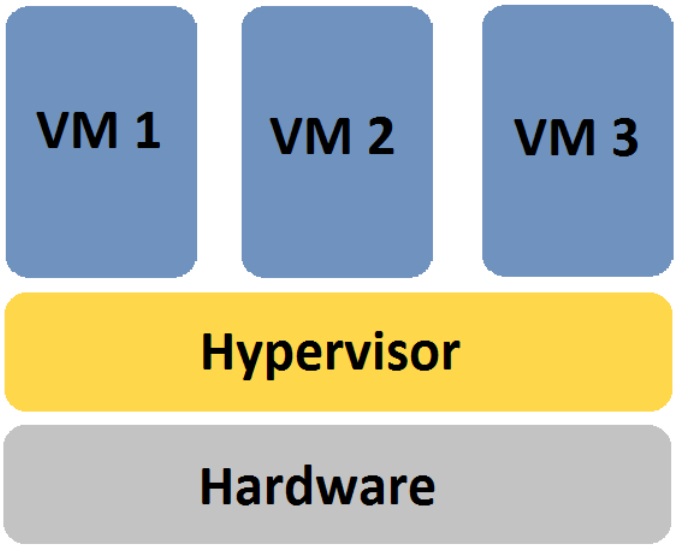

Figure 2.1: A schematic architecture of a typical bare-metal hyervisor. As one can see, the hypervisor runs directly on the host's hardware and therefore has direct access to the hardware.

2.1.1.1 Hyper-V

Hyper-V is a bare-metal hypervisor developed by Microsoft [Soper, 2012]. Hyper-V is the replacor of Microsoft Virtual PC in newer versions of Windows[Microsoft, 2015b] and was first included in Windows Server 2008, where system administrators could enable the

Hyper-V role [Shinder, 2008]. Hyper-V only supports x64 versions of Windows Server 2008 and Windows Server 2012 [MSDN, 2015].

The Hyper-V role provides the software infrastructure and management tools to create and manage a virtualized computer environment. As previously explained in section 1.3, each VM runs in a isolated computing environment [Microsoft, 2010]. In Hyper-V, this isolation is achieved by means of partitions [MSDN, 2015].

Architecture of Hyper-V In order for Hyper-V to work, a root partition (also known as parent partition) must be present on top of the hypervisor running a 64 bit version of Windows Server 2008 or Windows Server 2012. [MSDN, 2015]. It also runs the virtualization stack, contains the device drivers and therefore has direct access to the hardware resources as well [Smyth, 2009].

In essential, this root partition is nothing more than a virtual machine running Windows Server 2008 or Windows Server 2012 used to control the other guest VM's. The machine that installed the Hyper-V role, becomes the root partition. The root partition can be compared to the Dom0 VM of the Xen Project hypervisor explained in section 2.1.1.2.

The picture below visualizes the Hyper-V's architecture.

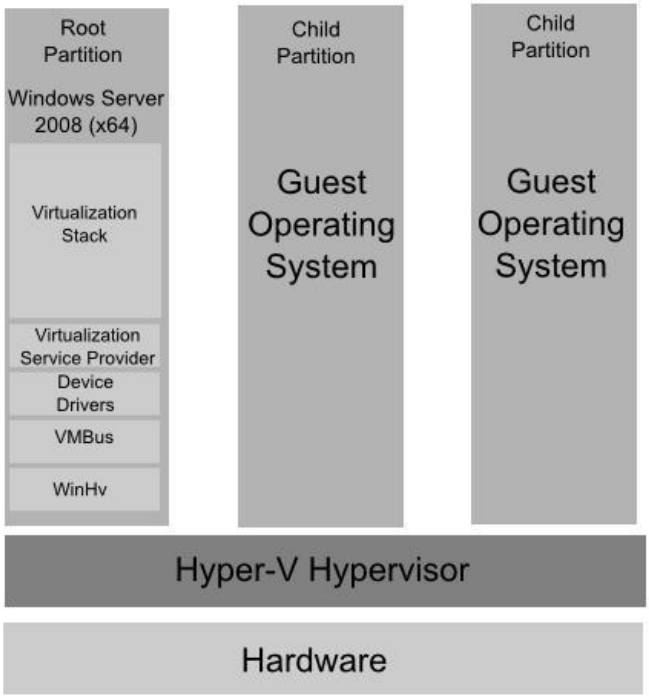

Figure 2.2: The architecture of the Hyper-V hypervisor. The root partion contains the Virtualization Stack which, in his turn, contains a collection of tools that provide the Hyper-V functionality, for example the virtual devices and the device drivers.

2.1.1.2 Xen

The Xen Project hypervisor is the only open source bare-metal hypervisor available at the time of writing. It consists of four main components being the Xen Project hypervisor, the guest VM's, the Control Domain and the Toolstack [Pavlicek, 2012].

Architecture of the Xen Project hypervisor The Xen hypervisor runs directly on the hardware and contains the scheduler. It is responsible for handling interrupts, CPU time and memory usage. The hypervisor is the first program to run after exiting the boot-loader [Pavlicek, 2012].

The Control Domain (Dom0) is a special type of virtual machine. Compared to the other guest VM's, this Dom0 has special privileges. E.g.: it is able to access the hardware directly and interacts with the other VM's. Therefore, the Dom0 contains the drivers for the hardware and a toolstack [Pavlicek, 2012].

One could compare this Dom0 with the root partition of Hyper-V.

The Dom0 is the first VM to be started by the hypervisor.

The Toolstack (xl is the prefered toolstack at the time of writing [Kurth, 2012]) allows users to manage their VM's [Pavlicek, 2012].

The guest VM's run on top of the Xen hypervisor and run their own operating system and application software. These are the actual VM's.
The picture below visualizes the Xen Project hypervisor's architecture [Kurth, 2012].

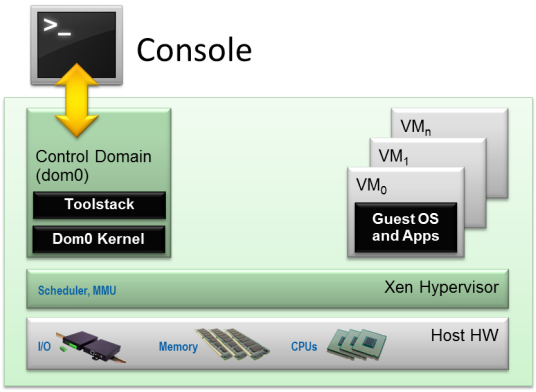

Figure 2.3: The architecture of the Xen Project hypervisor. It consists of four main components: the hypervisor itself, the guest VM's, the Control Domain and the Toolstack for managing the VM's.

2.1.1.3 VMware ESXi

VMware ESXi the bare-metal hypervisor developed by VMware. It is the successor of the ESX hypervisor [Grehl, 2015].

Architecture of the ESXi hypervisor The VMware ESXi hypervisor consists of the VMkernel, which is the underlying operating system and processes that run on top of the VMkernel.

The VMkernel contains the drivers, the management agents and applications and hardware monitoring components. The processes run on top of it are the Direct Console User Interface (DCUI), the virtual machine monitor (VMM) together with the helper processes VMX. Each VM has its own VMM and VMX process. On top of the VMkernel run the guest VM's.

The DCUI is responsible for low-level configuration and management. It is accessible through the server console and is used for basic, initial configuration.

The VMM is a process that provides an execution environment for the virtual machines, together with some helper processes knows as VMX processes.

Note that - in contrast to Hyper-V and Xen, there does not exist a fully functional console OS that manages the VM's. Instead, only a small POSIX kernel is included. This implies that the footprint of the hypervisor is very small - not more than 32MB.

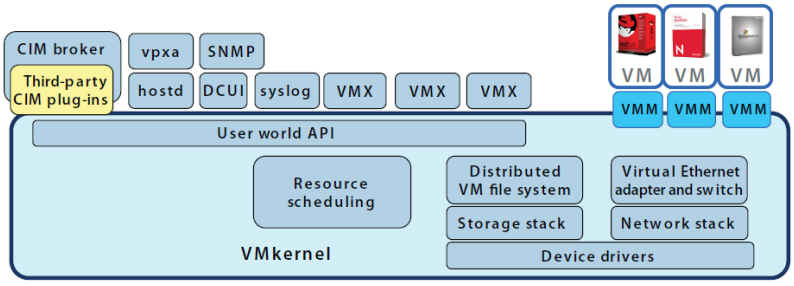

Figure 2.4: ...

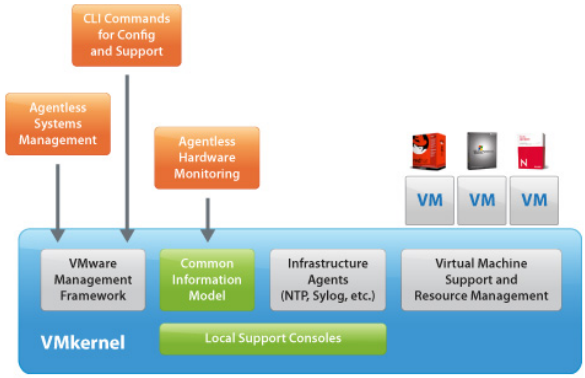

Figure 2.5: ...

The architectures of these 3 hypervisors are roughly equal .... In this thesis, Hyper-V will be used.

2.1.2 Hosted hypervisors

As described in section 2.1.1, bare-metal hypervisors run directly on top of the hardware. However, another type of hypervisor exist that runs on top of an existing operating system: the hosted hypervisors or type-2 hypervisors.

So basically, this means that an operating system is installed inside another operating system in contrast to the bare-metal hypervisor where each OS has direct access (through the hypervisor) to the hardware.

An example of such a hosted hypervisor is VirtualBox [Oracle, 2015].

2.2 Network virtualization

After having defined virtual machines and the hypervisors, it is now time to focus how a virtual machine is able to communicate with the outside world and to have a closer look how internal (i.e.: inside the hypervisor) network virtualization works.

When one wants to provide a VM with network capabilities, at lease one virtual network interface card (vNIC) has to be assigned to this VM - just as a physical computer requires a physical NIC as well [Technet, 2015a].

As Hyper-V will be used in this thesis, the focus of internal virtual networking will be stressed on Hyper-V.

One must not forget that a virtual network is actually just a software logic that is part of Hyper-V and sents and receives packets in OSI layer 2.

There exist three different types of virtual networks in Hyper-V: external virtual networks, internal virtual networks and private virtual networks [Technet, 2015b]. Each of them will be explained in the following section.

2.2.1 External virtual network

Virtualization is all about abstracting existing hardware as explained in the first chapter. The same concept is used for virtual networking. Consider a host with an arbitrary number of cores and an arbitrary amount of RAM which hosts 50 VM's, but with only one physical NIC.

The requests performed by the virtual NIC's of the VM's can overhelm the physical NIC of the host. To overcome this problem, Microsoft developed an abstraction layer that uses the Virtual Switch concept [Kennedy, 2011].

This abstraction layer sits in between the the physical NIC and the vNIC's of the different VM's. It therefore abstracts the physical NIC. Microsoft calls this *External Networking* [Technet, 2015b] and is illustrated in the figure below [Kennedy, 2011].

When creating an external virtual network in Hyper-V, the following changes take place:

- The host creates a new virtual NIC that is used to connect to the physical network. So a minimum of two network adapters exist on the host.
- The Microsoft Virtual Switch Protocol is bound to the physical NIC.

Once created, the virtual network will work exactly the same as a phyiscal network, with that difference that the switch is a software-matic; additional ports (NICs) can be added dynamically. Following figure visualizes the external network concept of Hyper-V. The external network type has the following capabilities [Howard, 2008]:

- It allows for communication between a VM and and an external network so that all VM's are visible as seperate hosts on the external network as if they would be dedicated, physical hosts. One can compare this with Xen bridged networking [Jackson, 2012].

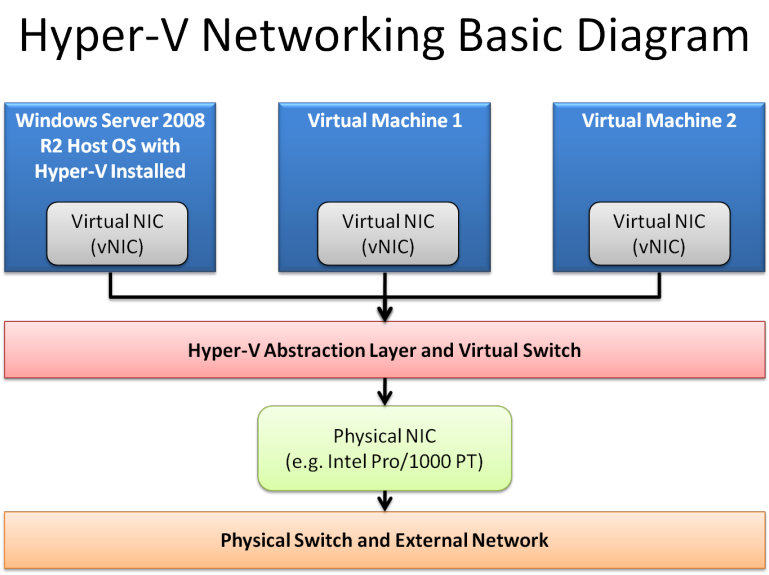

Figure 2.6: The concept of virtual networking as an abstraction layer on top of the physical NIC. The purple layer represents the virtual switch. Note that this is only occurs with external networking - internal and private networking do not use virtual switches.

- It allows for communication between VM's on the same host.
- It allows for communication between the VM's and the host.

2.2.2 Internal virtual network

In an internal virtual network, the phyiscal adapter plays no role in the network and is therefore not bounded. Thus access and communication to external networks (including the LAN outside the virtual machine) is impossible [Kennedy, 2011; Howard, 2008]. However, communication with the host is possible.
Some characteristics of internal networks are [Technet, 2015b]:

- It allows for communication between VM's on the same host.
- It allows for communication between VM's and the host.

The figure below visualizes this concept [Technet, 2015b]:

2.2.3 Private virtual network

When a private virtual network is used, VM's are only able to communicate with each other. No communication with the host exist, however. That is the difference with an

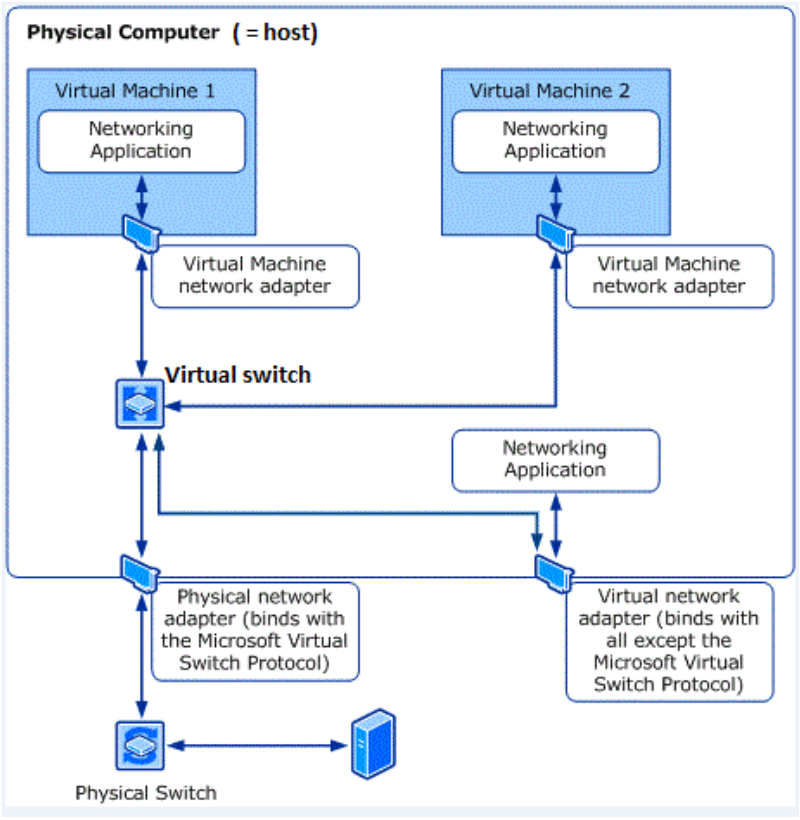

Figure 2.7: A diagram of the different components of external networking. The physical NIC binds with the virtual switch. The networking applications of the host communicate with the external network (the physical switch) through the newly created virtual NIC. Note that each VM is assigned a virtual NIC.

internal private network.
Some characteristics of internal networks are [Technet, 2015b]:

- Communication between VM's only.

he figure below visualizes this concept [Technet, 2015b]:

2.2.4 External network virtualization

VLAN's, ... Moet aan de prof vragen of dit ook moet.

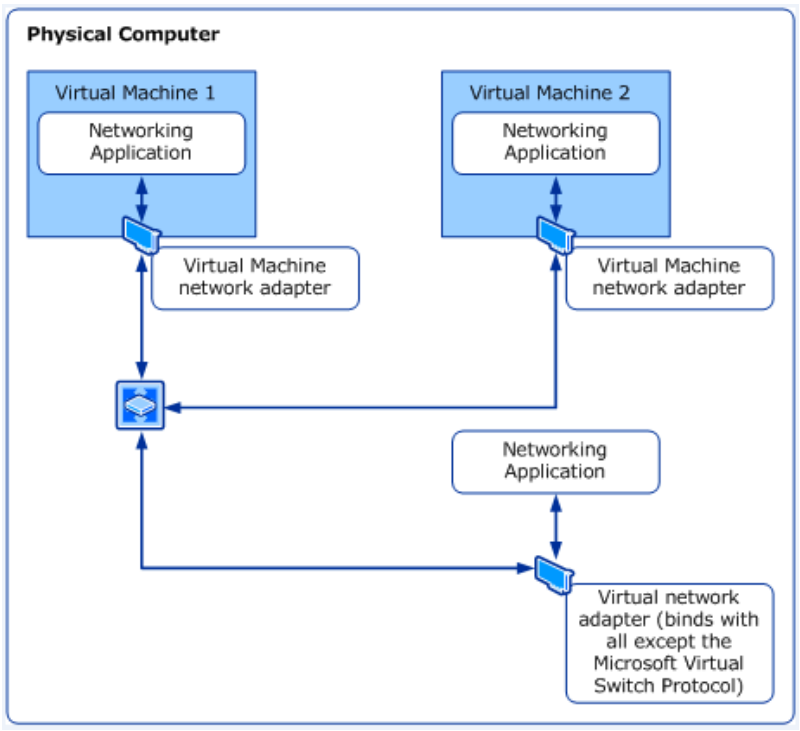

Figure 2.8: ...

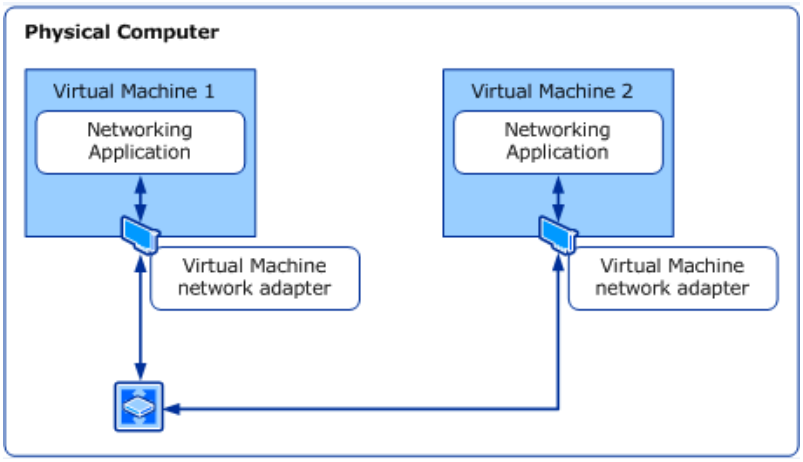

Figure 2.9: ...

2.3 Storage virtualization

2.3.1 Host-based storage virtualization

Host-based storage virtualization adds, as any virtualization technique (see chapter 1), a layer of abstraction between the host OS and the physical disks. Each mayor OS provides a piece of software called a logical volume manager to provide such a layer of abstraction [Garrison, 2011].
Whereas in traditional disk management, the OS looks for physical disks and partitions that reside on those physical disks, with a logical volume manager disks can be abstracted. This way, a logical disk can span multiple physical disk and appear in the OS as one disk.

A logical volume manager adds more flexibility in a way that logical volumes can be resized and moved around. This is not a straightforward job when physical disks and partitions are used [TLDP, 2002].
Microsoft's implementation of logical volume management is called Logical Disk Manager. In the next section, virtual hard disks in Hyper-V will be discussed.
The next figure illustrates the concept of Logical Volume Management.

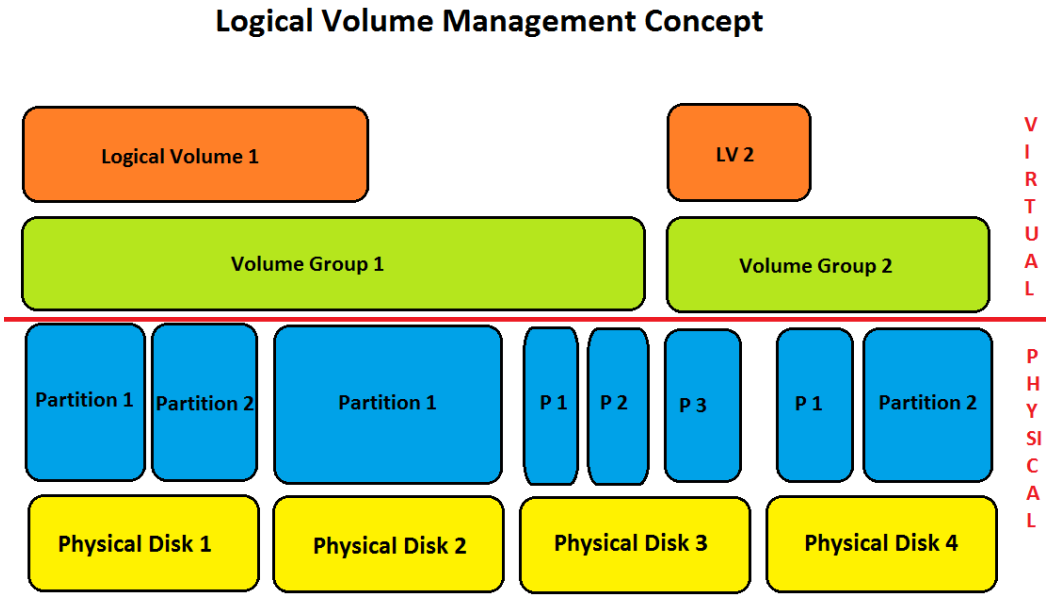

Figure 2.10: LVM adds a layer of abstraction between the OS and the phyical disks / partitions. In the figure, a volume group spans 5 physical partitions and disks. On top of the volume group lies a logical volume (partition). As one can see, there is space left on the volume group and thus expanding the logical volume (partition) is possible. With a phyiscal partition, this would not be possible without losing data.

2.3.2 Virtual hard disks in Hyper-V

A virtual hard disk (with extension .vhd) enables one to create a virtual disk which resides on the physical disk as a single file [MSDN, 2012]. A VHD has the same capabilities as a physical disk and thus are used the same way. They are able to host file systems and support standard disk operations [MSDN, 2012].

Chapter 3

Security aspects

In this chapter, some security issues concerning virtual machines and virtualization in general are covered, which will be the basis for further research in this MA thesis.

3.1 Known non-network related security issues

3.1.1 Virusses and malware problems

The main question we ask ourself here is the fact that if it is possible that virusses and rootkits can be transmitted from a guest VM to the host without the VM having networking capabilities?

As seen in the first section, virtual machines are all isolated from each other. Thus, it might seem odd that breakouks are likely to occur. However, with recent advances in technology as x86 Virtualization and I/O MMU Virtualization it turns out this is in fact possible.

I/O MMU Virtualization and DMA With I/O MMU Virtualization, for example Intel VT-d [Ott, 2009] or AMD-Vi [AMD, 2009] it becomes possible to directly assign an I/O device such as a graphical adapter to a VM [Burger, 2012]. But it also enables Direct Memory Access (DMA). One does not have to be a security expert to see that an infected VM with Direct Memory Access can infect the memory of the host.

x86 Virtualization and rootkits Noticable rootkits [1] are the **Blue Pill** rootkit developed by Joanna Rutkowska and the **SubVirt rootkit**, developed by Microsoft. These rootkits are two examples of VM-based rootkits (VMBR), which install an additional VMM between the host hardware and an existing operating system [Rutkowska, 2006; King et al., 2006], after which this new VMM can be used to host malicious software as illustrated in figure 3.1.1. Additionally and in contrast to usual rootkits, hypervisor-level rootkits remain undetectable, because they cannot be accessed by software [Rutkowska, 2006; Ben-Yehuda, 2013].

One possible solution to minimize (the bluepill attack becomes impracticable) this problem from happening is to disable x86 virtualization [2], despite the loss in performance. Also bear in mind that an anti-virus scanner on the host does not protect a guest VM against virusses [Avis, 2013].

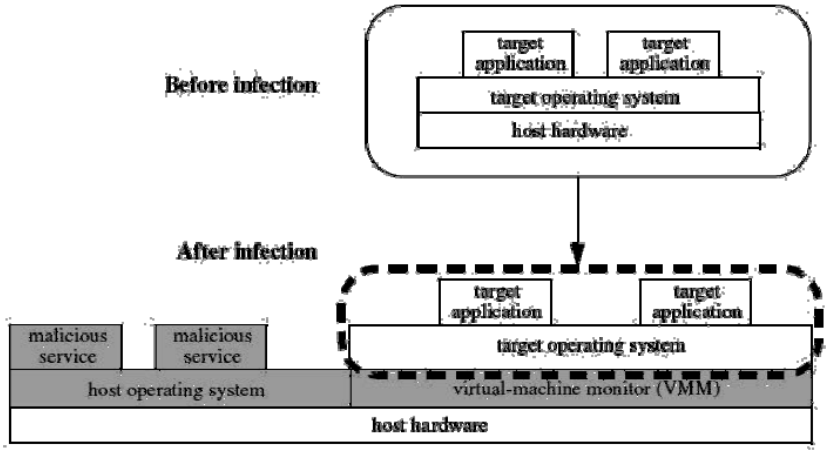

Figure 3.1: A schematic view of a virtual machine based rootkit (VMBR). A VMM, for example Hyper-V, is installed underneath an existing OS. This means that the rootkit is now the hypervisor.

[1] **Rootkit:** a mostly malicious program that is designed to hide itself for detection and thus for the fact that an OS has been compromised. It is used to gain *root* access (hence the name) to the compromised OS to perform, for example, eaves-dropping [Ben-Yehuda, 2013; Kassner, 2008].

[2] **x86 virtualization:** also known as hardware-assisted virtualization [Vangie, 2007], improves the performance when full virtualization is used. It adds hardware support to run VM's more efficiently. Normally, the machine code of the guest OS is translated to machine code of the host OS, by means of binary translation. With x86 virtualization - or hardware-assisted virtualization, the need for such binary translation disappears. Hardware-assisted virtualization must be supported by the CPU [Jeong, 2013]. Therefore, for example Intel and AMD developed respectively Intel-VT and AMD-V for that purpose [Intel, 2015]

3.2 Known network related security issues

When VM's are directly connected to an external network (e.g.: using Hyper-V's external network mode), then the VM is threated as any other physical machine regarding the transmission of virusses. Just a virusses spread between physical machines using files (e.g.: via mail attachments or software downloads) that are transferred through the network [Avis, 2013], the same can also happen with VM's who are connected to an external network, since when a virtual network is connected to a physical NIC, it is exposed to the same threads and security risks as that physical NIC and thus as a normal physical network[Technet, 2015c].

3.3 Possible solutions

3.3.0.1 Use of firewalls

The built-in Windows Firewall in Hyper-V does not interfere - and thus does not protect guests - with guest traffic in any way. Packets will just pass the Windows Firewall without being analyzed, because of the physical adapter bound to the virtual switch is unbound from anything that Windows Firewall has access to.

However, extensions to the Hyper-V Virtual Switch are available that take care of these problems. A network packet filter and an intrusion detection or firewall are two out of four examples of such extensions [Remde, 2012].

Obviously, one can always completely isolate virtual networks from each other - and from physical networks - in order to protect the host. This can be done using VLAN's - or IP subnets - : the hypervisor (host) does not have to be on the same VLAN as the guests and thus placing it in a seperate VLAN is perfectly possible [Siron, 2014].

3.3.0.2 Securing the guests

As already described in the first section, virtual machines are isolated from each other, in such a way they cannot access each other's physical resources such as RAM.

However, in 2012, a security issue was found in 64 bit virtualization software running on Intel CPU's. With this vulnerability, when a system exception occured, it became possible to escape from the local guest OS into the host OS with elevated privileges, with all the consequences.

Therefore, a guest has to be secured and securing a guest is just like securing a phyiscal machine.

3.3.0.3 Antimalware solution

Attention has to be payed when installing antimalware solutions on the host OS. Either antimalware solution poses a threat to the host: most of the antimalware software dislike XML files, however, these files defines the VM's, so deleting them will cause the VM's to disappear.

This means that one has to be very careful when installing antimalware solutions and in particulary take care of exclusions.

3.4 Sniffing on virtual networks

Another potential security hazard is the possibility to snif a virtual network interface to run a network packet analyzer such as Wireshark to capture all the traffic intented for that host and reconstruct the protocols.

An even bigger security risk is that an intruder could possibly hack a virtual switch to set up a trunk port that is able to see the network traffic on the other ports and redirect this traffic to the virtual network interface the intruder is listening on.

Actually, the fact that both Hyper-V and Xen are using virtual **switches** rather than virtual **hubs** is a first layer of protection.

Indeed, a switch only forwards the traffic to the intented host - since only one host per segment is present, whereas a hub sends an incoming packet to all ports - since multiple hosts can reside on one network segment.

In the upcoming sections, the possibilities of packet sniffing on Hyper-V and Xen virtual networks will be examined as well as measures to prevent packet sniffing, thus enhancing the security.

3.4.1 First research question: sniffing a virtual NIC

Following the above mentioned security risks, the first research question is thus ***if it is possible to snif a virtual network interface and run a network packet analyzer on it?*** and additionally ***is it possible to hack a virtual switch to capture all the***

traffic?

For the purpose of investigating this research question, a test lab has been created. The reader is invited to take a look at figure A.1 for a schematic logical overview of the network used in the investigation. Appendix A contains a detailed description of the hardware used.

Before hacking the virtual switch, we must first determine if it is possible at all to snif a virtual network interface - that is, running a network packet analyzer on it?

3.4.1.1 Setting up the environment and testing

Consider the following environment: one Hyper-V host running three VM's. Two of these VM's are ordinary VM's and one is a so-called `Monitoring VM`, which runs a packet analyser - or any other sniffing / capturing software.
Each VM has one virtual NIC and are connected to the external switch of Hyper-V. This means that each VM has access to the lab network and is thus acting as a normal physical device. The picture below represents the reader with a visual representation of the network.

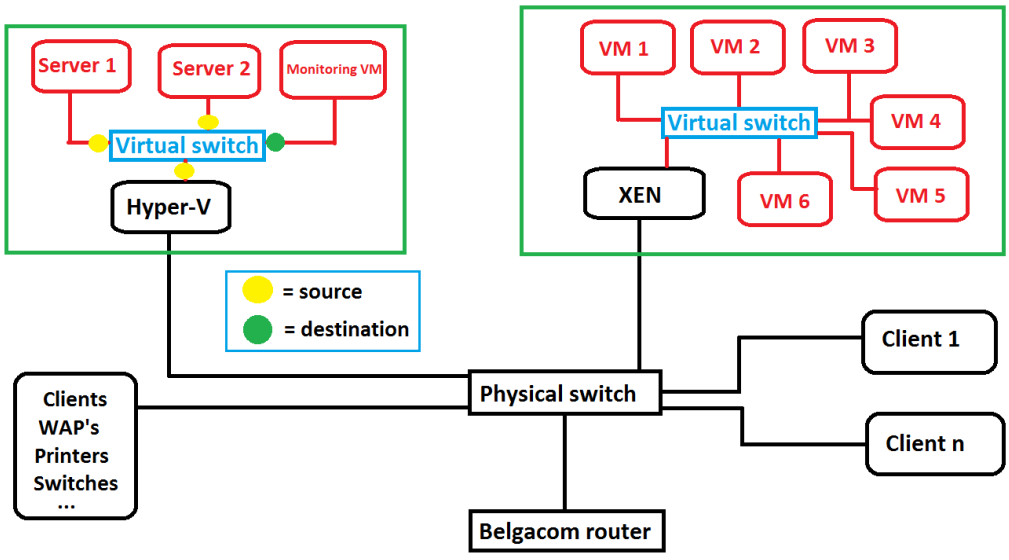

Figure 3.2: The network setup used to investigate packet sniffing.

After doing some research, it turns out that Hyper-V 2012 has built-in port mirroring. Using this feature, one can set a source interface and a destination interface. All traffic coming

from and to the source interface is then copied to the destination interface. (NOG ONDERZOEKEN OF DIT KAN GEWIJZIGD WORDEN BIJ EEN REEDS DRAAIENDE VM)
Any person with Administrator privileges is able to change the monitoring mode of a VM (that is, a vNIC). This way, when a hacker has gained Administrator access, he could create a new VM with port mirroring configured, or configure port mirroring on an existing VM and install a packet sniffer on it to reconstruct protocols.
The figure below illustrates the necessary commands to setup port mirroring on Hyper-V. The first three commands are straightforward, but the last two commands are neccesary to forward traffic that arrives at the Hyper-V external switch interface to the virtual switch. Without these rules, any traffic arriving at the Hyper-V host will 'stop' at the host and thus won't be picked up by the `Monitoring VM`.

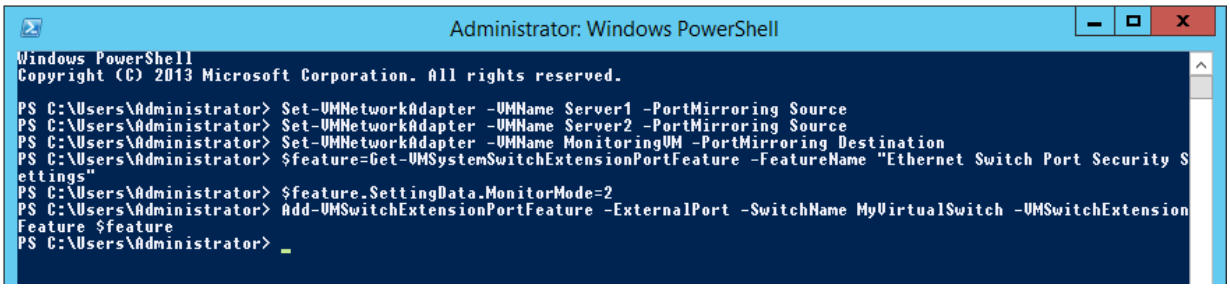

Figure 3.3: PowerShell commands to configure port monitoring. The vNICs of `Server 1` and `Server 2` are set as 'source', whereas the vNIC of monitoringVM is set as 'destination'.

First stage: inter-VM testing After executing the commands, the setup was tested. Wireshark was installed on the `Monitoring VM` and I pinged from `Server 1` to `Server 2` and watched how Wireshark captured all the ICMP Ping requests and replies - not destinated to the `Monitoring VM`. So, **yes**, the vNIC on the `Monitoring VM` captured traffic between VM's (inter - VM) on the same physical host and thus act as a sniffer.

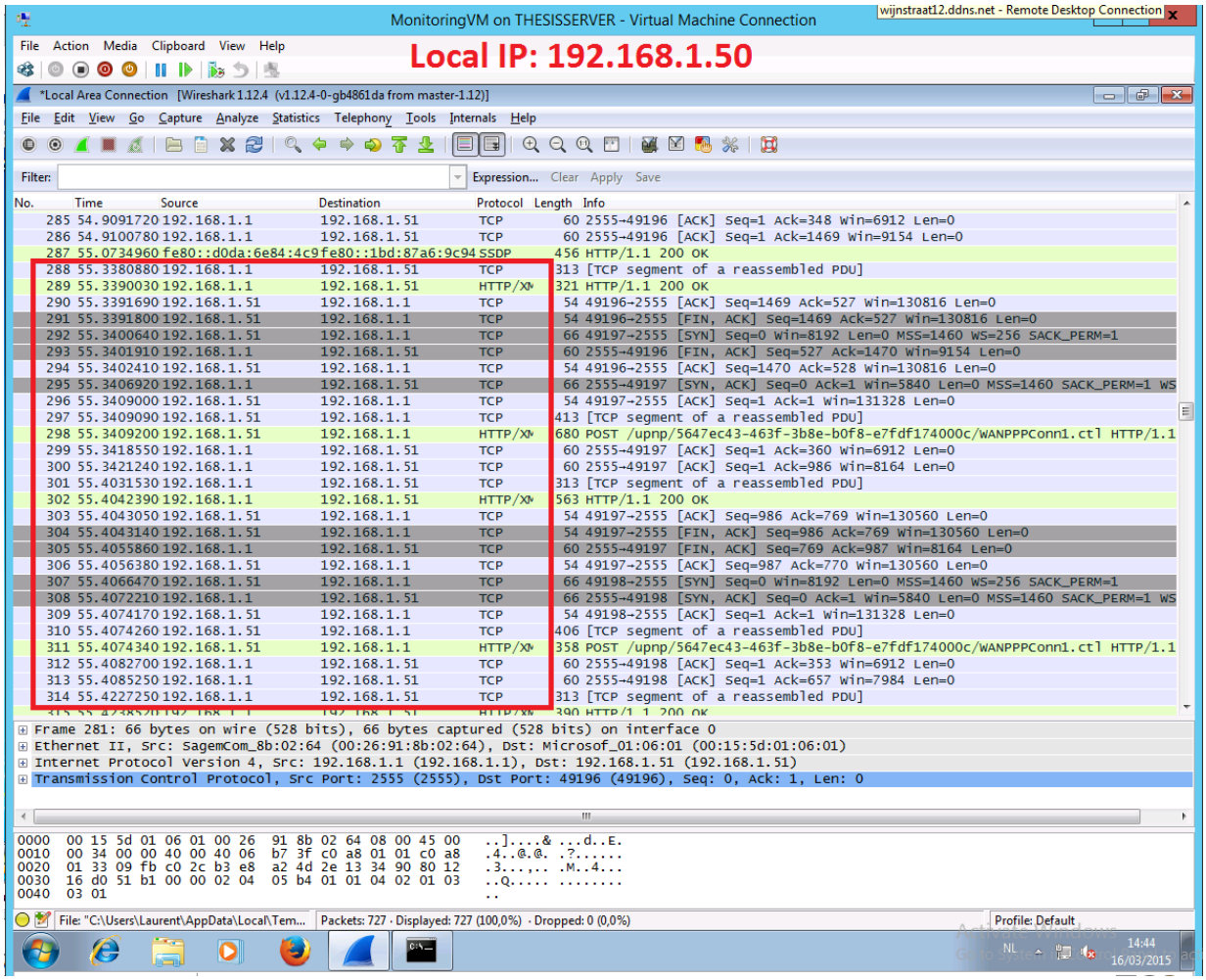

Figure 3.4: Traffic captured on the Hyper-V virtual network prior to executing the ICMP Ping requests.

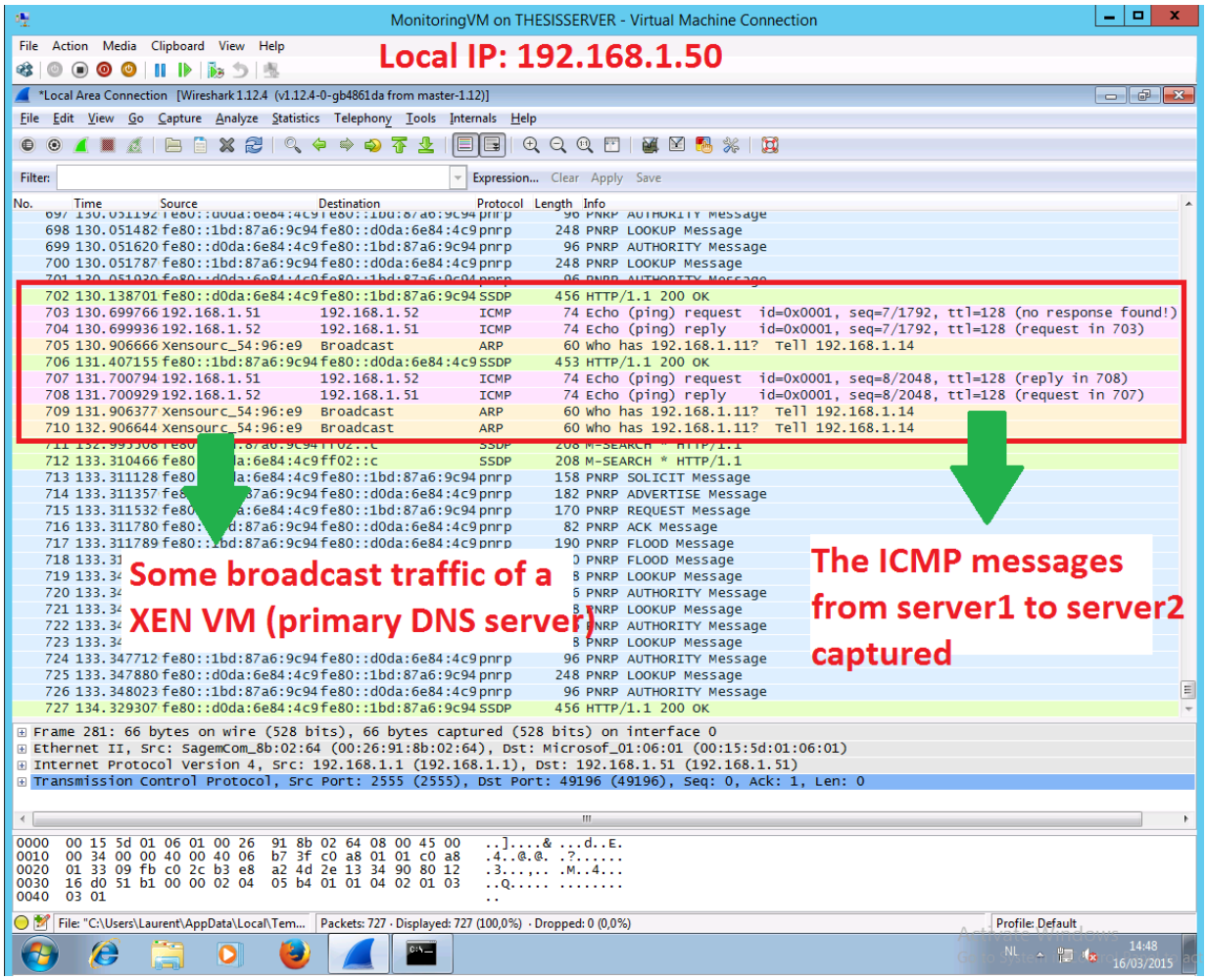

Figure 3.5: ICMP Ping traffic captured by the monitoring VM. Besides this, some DNS traffic can be observed as well.

Second stage: external network testing So far, only internal traffic between VM's has been tested for sniffing. I also wanted to test if this `Monitoring VM` is able to capture traffic originating from the Xen server or any other client on the network. Therefore, extra ICMP Ping commands have been executed from a random Xen VM to the Hyper-V host as well as to either VM running on this Hyper-V host as illustrated in the following figures.

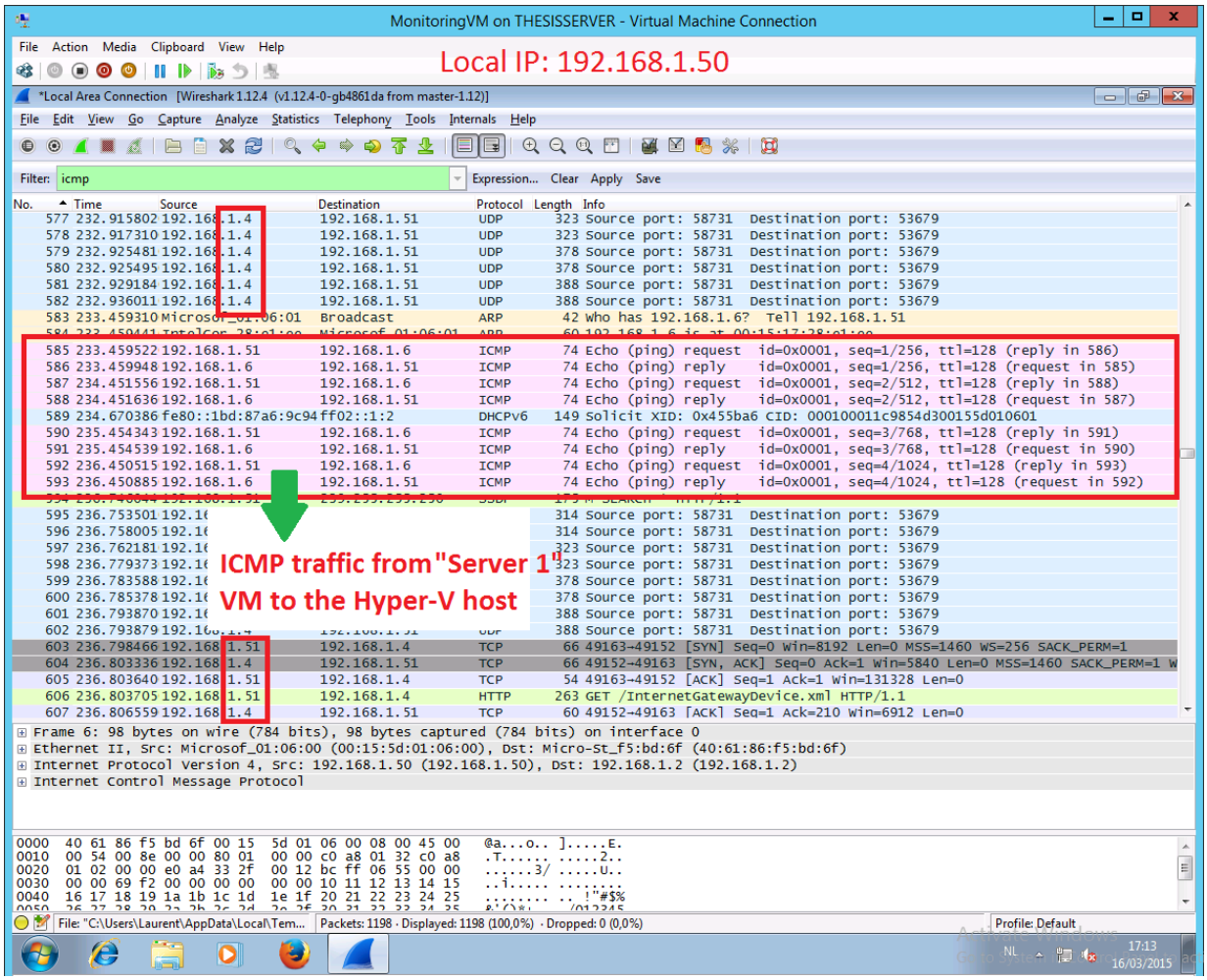

Figure 3.6: ICMP Ping traffic from `Server 1` (192.168.1.51) to the Hyper-V host (192.168.1.6). Note that other 'external' traffic is captured as well. In this case, traffic from `Server1` to a wireless access point.

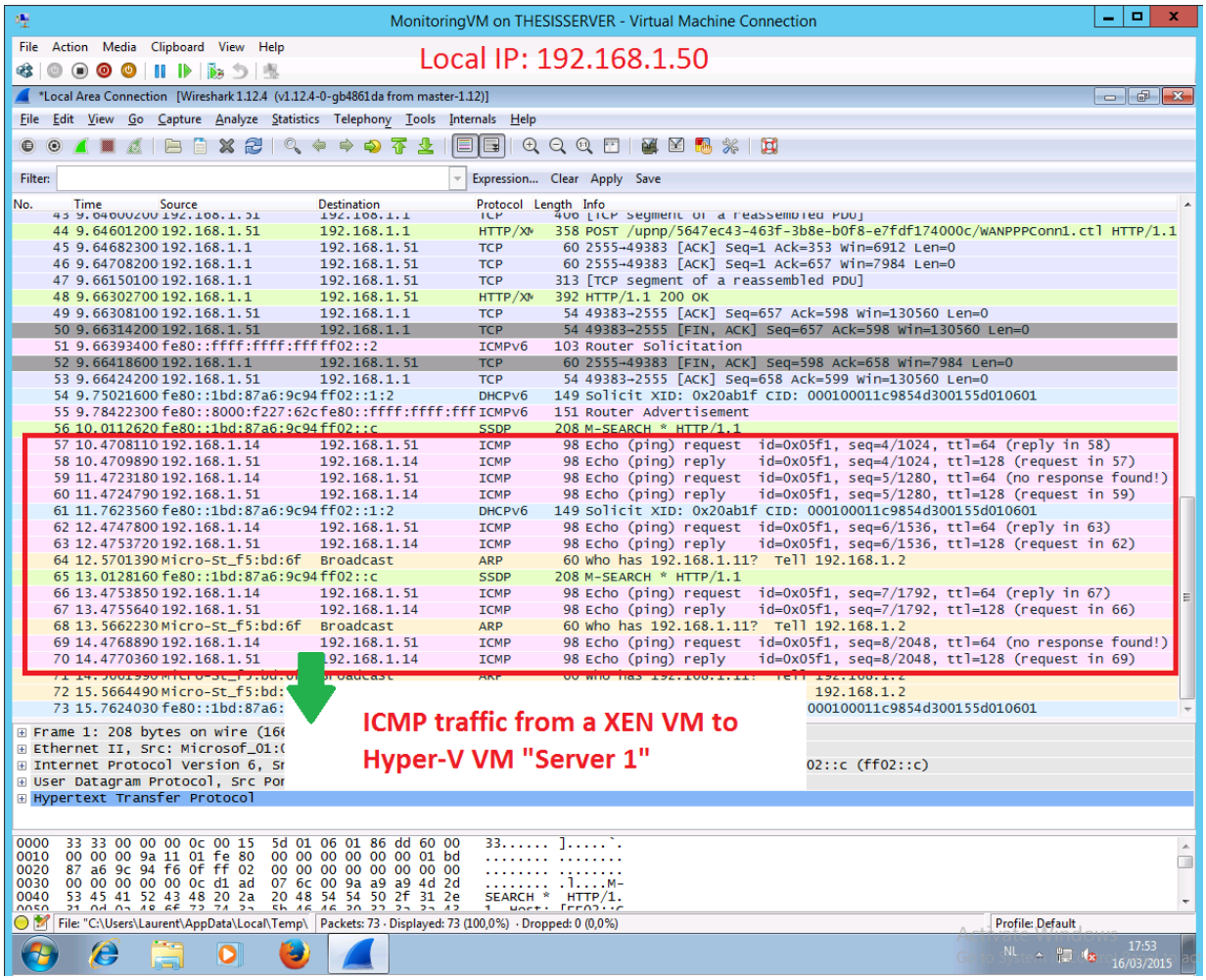

Figure 3.7: ICMP Ping traffic captured from a Xen VM to a Hyper-V VM, in this case `Server1`.

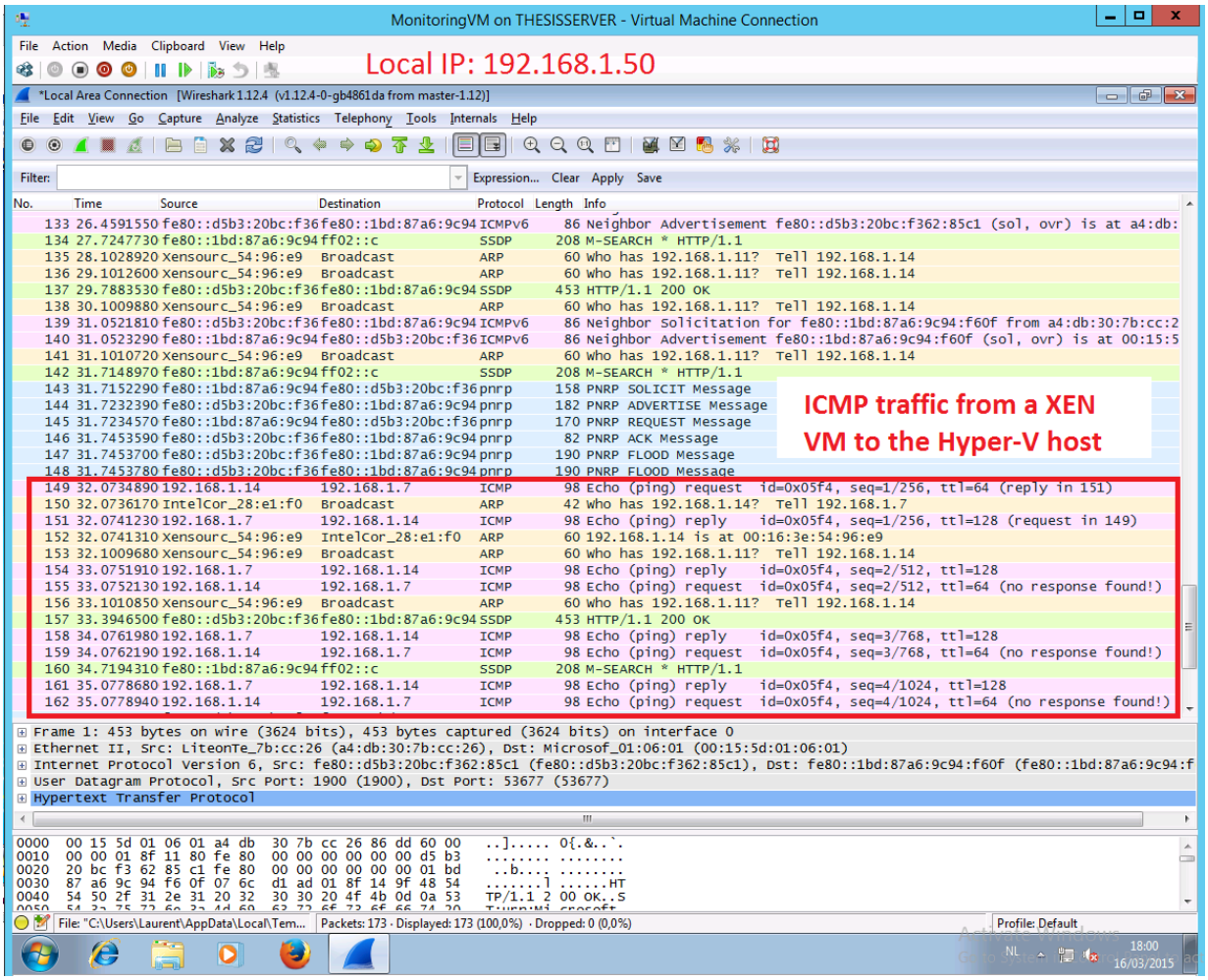

Figure 3.8: ICMP Ping traffic captured from a Xen VM to the Hyper-V host.

Traffic capturing capabilities has been demonstrated with various configurations:

- Inter-VM between two Hyper-V VM's.
- From a Xen VM to the Hyper-V host.
- From a Xen VM to a Hyper-V VM.
- From a Hyper-V VM to the Hyper-V host.

As illustrated, it is perfectly possible for an intruder to snif network traffic between VM's. Now that has the possibilities of packet sniffing / capturing has been shown, an important question raises: how can one prevent packet sniffing on virtual NIC's?

3.4.2 Penetration testing on virtual networks

Now that the possibilities of packet sniffing on virtual networks have been proven, it is time to perform some penetration testing on these virtual networks. Since packet sniffing is possible, we should be able to run an intrusion detection system (IDS) on either virtual network to intercept and prevent those penetration attacks. For this purpose, the Snort IDS will be used.

The upcoming sections will provide the reader a brief description of the Snort IDS and its components as well as an overview of installing Snort. Next, the actual penetration testing, countermeasures and techniques to prevent certain types of network attacks will be discussed.

3.4.2.1 Snort

Based on the different types of classifications of intrusion detection systems, one could classify Snort as a "Signature-based Network Intrusion Detection and Prevention System".

Additionally, Snort is open-source and capable of performing real-time network traffic analysis, as well as packet logging on IP networks. It consists of two major parts: the detection engine and the rules that are used to describe traffic that has to be collected or blocked. Furthermore, there exist two types of rules: rules that are available to all users (the so-called "Community Rules") and private, proprietary (paid) rules that are developed by SourceFire, the company behind Snort [Cisco, 2015c]. Addionally, so-called "Registered rules" are also available and are situated between the GPL rules and the proprietary rules.

I will make use of the freely available community rules combined with the "Registered rules". As one will notice, it will turn out that these rules are rather basic and need to be extende as well as additional rules have to be added in order to make Snort a viable IDS. This is allowed, since Snort uses the GPL license and we will make extensive use of this privilege to freely modify the rules. As previously mentioned, "Registered rules" have also been included in addition to the basic GPL rules. In order to do so, an account on the Snort website has been made and the rules were downloaded using `wget`.

As mentioned in the above paragraphs, Snort uses pre-defined rules to detect malicious activity on the network. One could compare it with the way antivirus programs work: any traffic that Snort picks up is matched against the database of rules and when a match has

been found, an alert is raised.

3.4.2.2 Preprocessors

Preprocessors extend the functionality of Snort by examining packets or by modifying them so that Snort can properly interpret the packets. [Cisco, 2015a].

Some attacks cannot be detected by normal signatures (rules), so "examine" preprocessors detect suspicious behaviour. So one could say this type of preprocessor is used to detect non-signature-based attacks [Cisco, 2015a].

The othe type of preprocessor is used to normalize traffic, so that Snort can match signatures in an accurate way.

Preprocessor code is run before the detection engine is called. Additionally, each packet captured by Snort is cycled through every preprocessor, in order to discover even more attacks [Koziol, 2003].

3.4.2.3 PulledPork

PulledPork (PP) is Perl script that will automatically download new Snort rules (signatures) in the background. Of course, one can always run PP directly at the command line to force the downloading of new rules [Cummin, 2010].

The downloaded rules are stored in a file called "download.rules".

3.4.2.4 Barnyard2

Barnyard2 is an interpreter for Snort binary output files. It is available under the GPL license and therefore, free to use [Firnsy, 2010].

Barnyard2 allows Snort to write data to the disk in an efficient way. The parsing of binary data into different formats is handed over to a seperate process. Because Barnyard2 takes away some work, Snort will not miss any network traffic [Northrop, 2013].

3.4.2.5 Packet capturing

Packet capturing is accomplished by a program called `netsniff-ng`. Via a network interface set to promiscious mode, netsniff-ng captures all the traffic the sensors (explained later) of Ubuntu see and store as much of the information as possible. I.e., until the hard drive is full. Of course, a build-in mechanism to purge old data when the amount of data reaches a pre-defined level existss.

One could compare packet capturing with a video camera, that precisely sees and registers who, when and where was. The video camera is netsniff-ng and the persons walking around are the packets. All this is registered in a database.

Additionally, a video camera is also capable of registering of what people took with them. This is also the case with packet capturing: the payload of the packets can be examined as well as the destination address of the packets.

3.4.2.6 NIDS and HIDS

Network-based (NIDS) and host-based intrusion detection systems (HIDS) analyse the traffic that netsniff-ng captures and will log any malicious packets as well as sending alerts. There exist multiple IDSs:

NIDS

- Signature-based NIDS: Snort or Suricata. In this paper, Snort will be used.
- Anomaly-based NIDS: Bro IDS. The definition of anomaly-based IDSs has been covered in the previous chapters and since Snort will be used, we will not dive deeper into Bro.

HIDS

- OSSEC: an open-source HIDS for Windows, Linux and Mac OS X. Instead of monitoring an entire network, OSSEC monitores only one specific host on suspicious activity [Sid, 2014].

3.4.2.7 Analysis tools

With Snort / Suricata data and the packet capturing of `netsniff-ng`, there is a vast amount of data available for the analist. To help managing the alerts generated by Snort or Suricata, some handy tools have been installed, including:

- **Sguil**: a network security analysis tool providing an intuitive GUI that provides access to realtime events and raw packet capturing data. The client is written in tcl/tk. In the client, one can view alerts of Snort, Bro, Suricata and OSSEC [Visscher, 2014].

  The alerts are stored in a seperate MySQL database and this allows a user to query alerts by type, IP address or port, for example.

With Sguil, it is also possible to categorize alerts. This can be done either manually or automatically. The next chapter will provide more details of how this can be achieved [Visscher, 2014].

- **Squert**: a web application that is used to view and query event data for the Sguil database. One could see it as a web interface for the Sguil database. It is neither ment to be a real-time interface, nor a replacement for Sguil, but more to bring additional visualization options to Sguil [Halliday, 2012].
- **Snorby**: a RoR (Ruby On Rails) web application that allows one to visualize Snort and Suricata alerts as well as perform queries on them. For example, listing the most active IDS signatures, most active sensors, ... While all this can also be done with the Sguil database, Snorby offers a web interface instead of manually querying the Sguil database. In contrast to Sguil and Squert, Snorby uses his own, seperate database [Webber, 2015].

3.4.2.8 Running Snort IDS on a virtual network

With Snort, Snorby and Sguil installed and configured, we can start using it. Of course, in order to be sure whether Snort indeed picks up all the traffic that flows through the network, we must first test it. The figure below illustrates the correct installation of Snort.

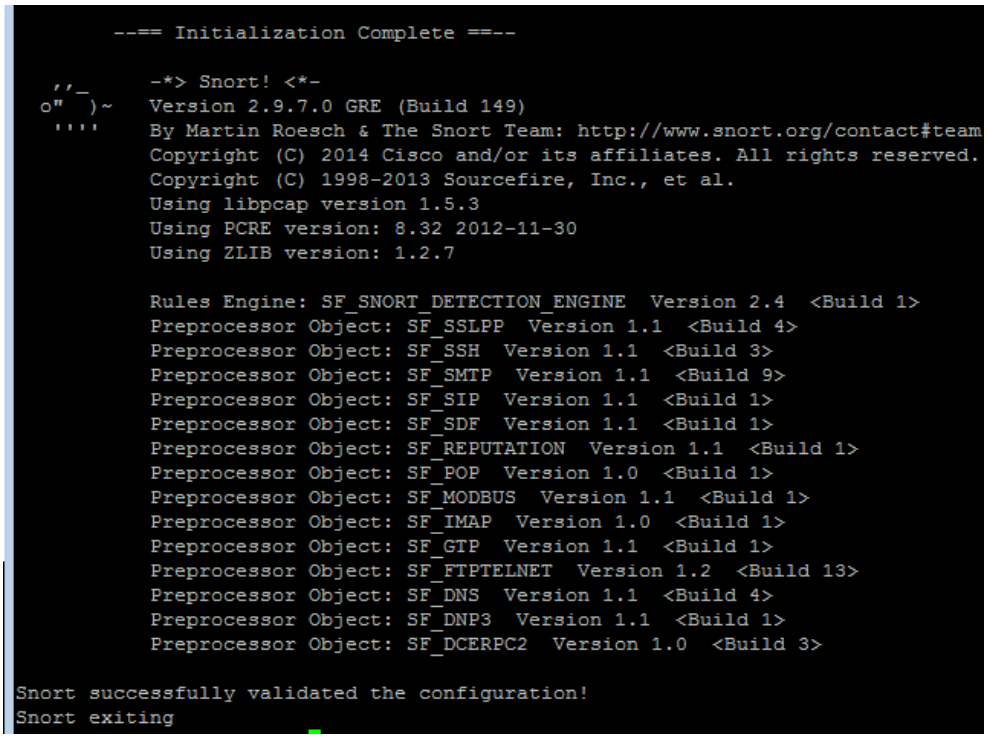

Figure 3.9: Confirmation of the correct installation of Snort.

3.4.2.9 First things first: basic configuration of Snort

First of all, we need to verify the network settings. Therefore, the `ifconfig` command is run. In addition, we also have a look at `/etc/network/interfaces` to confirm the settings.

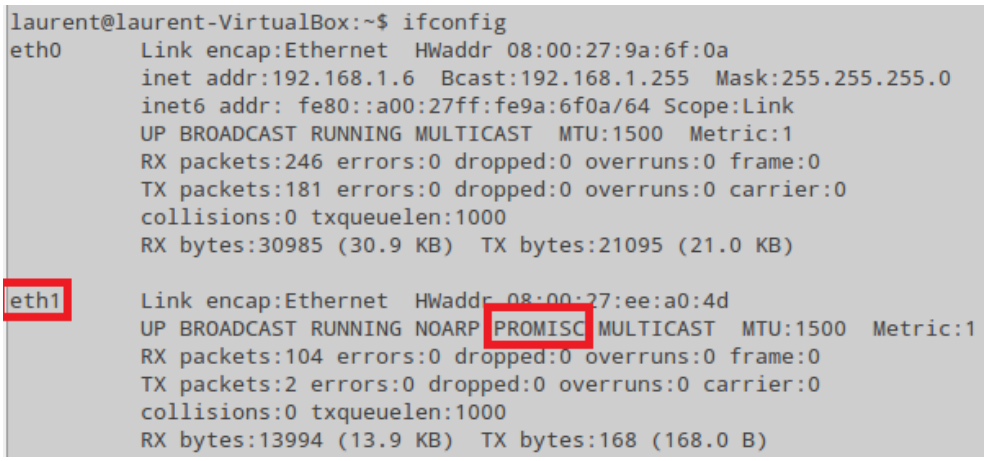

Figure 3.10: The sniffing / capturing interface `eth1` runs in promiscious mode and captures all the packets on the network. No IP address has been set (as it should be).

```
# loopback network interface
auto lo
iface lo inet loopback

# Management network interface
auto eth0
iface eth0 inet dhcp

# Sniffing network interface
auto eth1
iface eth1 inet manual
  up ip link set $IFACE promisc on arp off up
  down ip link set $IFACE promisc off down
  post-up ethtool -G $IFACE rx 4096; for i in rx tx sg tso ufo gso gro lro; do ethtool -K $IFACE $i off; done
  post-up echo 1 > /proc/sys/net/ipv6/conf/$IFACE/disable_ipv6
```

Figure 3.11: Confirmation of the network settings.

After having verified that the network settings are correct, we can configure some of the basic options of Snort. This includes network settings and is performed in "snort.conf". For example, the network segments on which Snort has to listen have to be configured.

```
# Setup the network addresses you are protecting
ipvar HOME_NET [192.168.1.0/16,10.0.0.0/8,172.16.0.0/12]

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET

# List of sip servers on your network
ipvar SIP_SERVERS $HOME_NET
```

Figure 3.12: We are working on the 192.168.1.0/24 network, but by means of testing, we have setup Snort to listen on the /16 subnet (255.255.0.0). As it will turn out, this will **not** affect the correct working of Snort. So one could randomly choose a value (8, 16 or 24).

3.4.2.10 Configuring Snort rules

Throughout this paper, we will frequently modify and add Snort rules (signatures), so an explanation of how a Snort rule is composed is given first.

Most Snort rules are written in a single line. When a rules needs to span multiple lines, this can always be performed by adding a backslash (\) at the end of the line.

Snort rules consist of two main parts: the rule header and multiple rule options. The header contains the rule's action method, protocol, source IP, source port, destination IP and destination port.

The rule options contains alert messages, signature ids, revisions and many more options.

Below is the general layout of a Snort rule.

action protocol sourceIP sourcePort → destinationIP destinationPort (OPTIONS);

To clear things up, an example is provided.
Consider following rule:

alert icmp any any → any any (msg:"ICMP ping"; sid:100002;)

This could be read as follows: "alert all ICMP traffic from any source IP, from any port to any destination IP and to any destination port. Alert as "ICMP ping" and the unique id of the rule is 100002.".

Action can be one the following [Cisco, 2015b]:

- alert: an alert is generated after which the packet is logged.
- log: just log the packet, do not generate any alerts.
- pass: ignore the packet.
- drop: log and block the packet.
- reject: log and block the packet and sent a TCP reset or ICMP port unreachable when TCP or UDP is used, respectively.

Protocol can be one the following [Cisco, 2015b]:

- TCP
- UDP
- ICMP
- IP

The IP address can be an ordinary IP address or an address/CIDR combination. For example, 192.168.1.0/24 would mean the block of addresses from 192.168.1.1 till 192.168.1.254.

3.4.2.11 Updating the rules using PulledPork

After installing Snort, it is a good practise to update the ruleset, just as one may do when one has just installed an antivirus program. This is done using the `update-rules` command.

One could make a cron-job of this command to run it, for example, every day at 3 am.

```
Rule Stats...
        New:-------19
        Deleted:---6
        Enabled Rules:----17195
        Dropped Rules:----0
        Disabled Rules:---3871
        Total Rules:------21066
```

Figure 3.13: After PulledPort has run, one can notice that 19 new rules have been downloaded and added to the ruleset of Snort.

3.4.2.12 Confirming that Snort is actually working

To confirm that all the agents (sensors) and the servers are running, the following command is executed: `sostat -quick`. Which yields following output:

```
Starting: OSSEC-eth1
  * starting: netsniff-ng (full packet data)                    [  OK  ]
  * starting: pcap_agent (sguil)                                [  OK  ]
  * starting: snort_agent-1 (sguil)                             [  OK  ]
  * starting: snort_agent-2 (sguil)                             [  OK  ]
  * starting: snort-1 (alert data)                              [  OK  ]
  * starting: snort-2 (alert data)                              [  OK  ]
  * starting: barnyard2-1 (spooler, unified2 format)            [  OK  ]
  * starting: barnyard2-2 (spooler, unified2 format)            [  OK  ]
  * starting: prads (sessions/assets)                           [  OK  ]
  * starting: pads_agent (sguil)                                [  OK  ]
  * starting: sancp_agent (sguil)                               [  OK  ]
  * starting: argus                                             [  OK  ]
  * starting: http_agent (sguil)                                [  OK  ]
laurent@OSSEC:~$
```

Figure 3.14: Everything is running fine.

However, we cannot assume that, just by installing and configuring Snort - where everything **seems** to be working, that everything **is** actually working. To convince myself, a simple Snort rule has been made to detect and alert for any ICMP Ping traffic on the

network:

alert icmp any any → any any (msg:"ICMP Ping traffic"; sid:100002;)

When pinging from the attack machine to the victim, Snort indeed picks up the ICMP traffic and fires a corresponding alert, as well as storing the information in the database. This can be seen in the figure below.

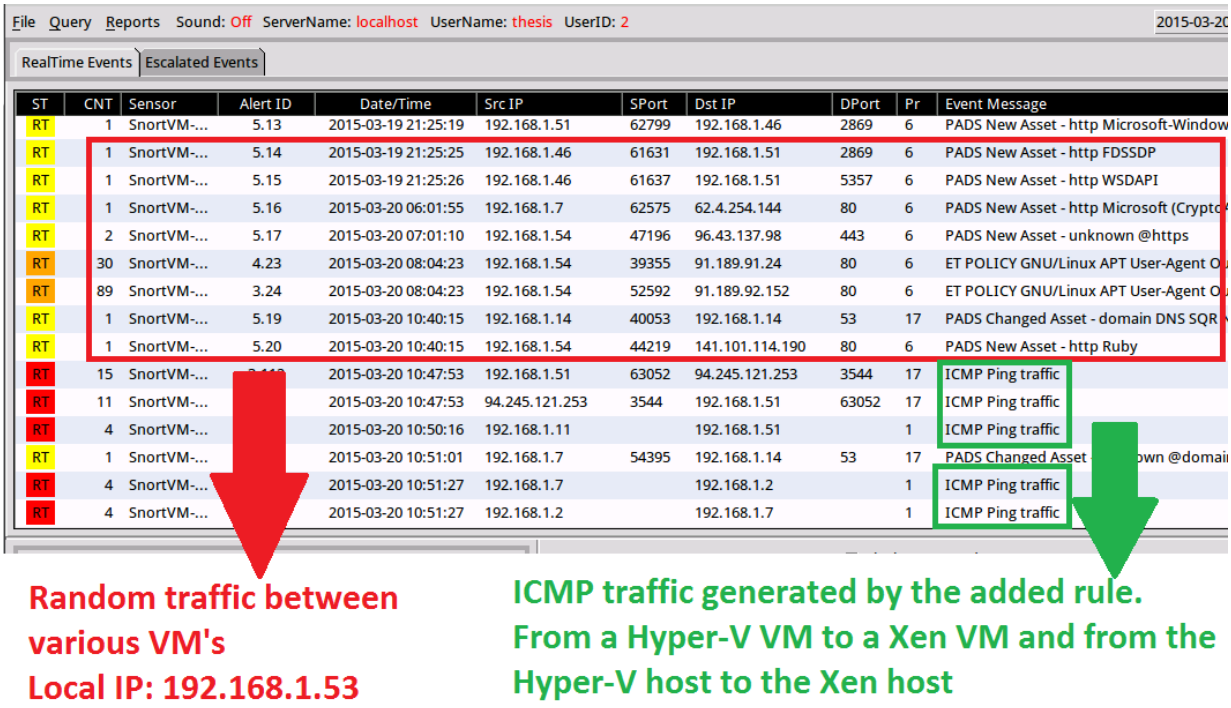

Figure 3.15: Ping traffic gets picked up by Snort.

This can also be confirmed by executing `tcpdump` on the Snort VM:

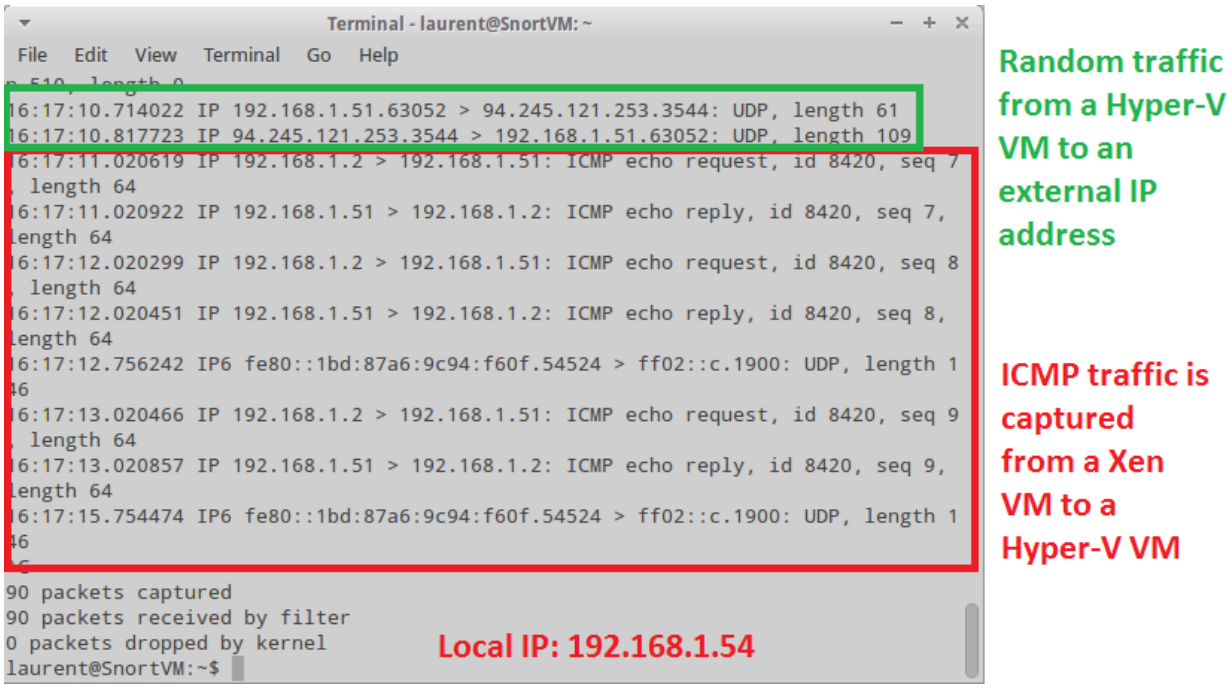

Figure 3.16: `tcpdump` confirms that the Snort VM is receiving traffic other than the traffic destinated for the VM. This proves the fact that the Snort VM is actually sniffing traffic.

However, Snort is only picking up intrusions on the Hyper-V network including the Hyper-V host. Thus, intrusions on the Xen virtual network are not detected, because those VM's resides on a seperate virtual switch.

So I wanted a way to also detect malicious activity on the Hyper-V network with the SnortVM running on a seperate virtual network.

Since I'm using in-kernel bridging for the virtual Xen switch, my initial idea was to snif the bridged interface (since all traffic - also between VM's - passes through this interface) and copy all the packets to another, physical interface set in promiscious mode. From there, I could then forward the packets to the Hyper-V network.

However, it is not possible to forward packets on an interface that is in promiscious mode. So instead of making a copy of the bridged interface, the only solution was to make a copy of the packets on each virtual interface connected to the vSwitch.

This solution works fine as illustrated in the following screenshots.

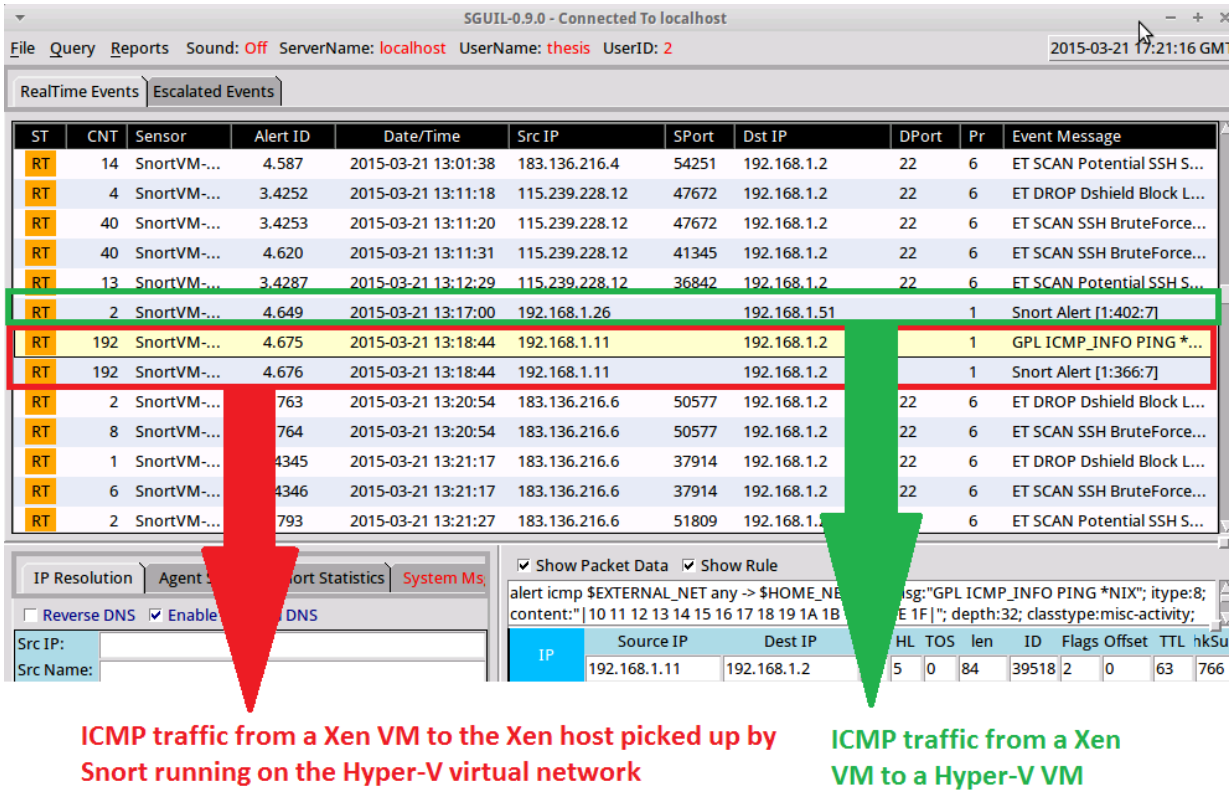

Figure 3.17: Traffic originating from a Xen VM to the Xen host is captured by the Snort VM as well as traffic originating from a Xen VM to a Hyper-V VM.

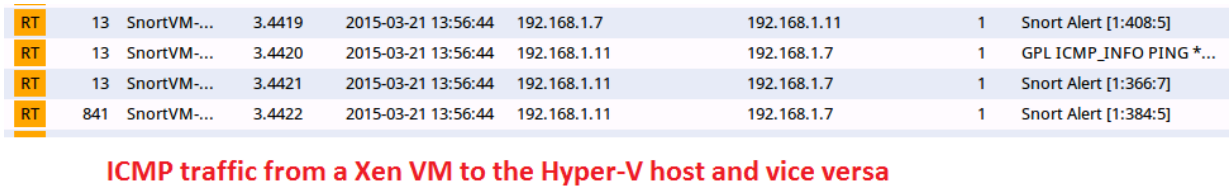

Figure 3.18: Traffic from the Hyper-V host (the hypervisor where the Snort VM is installed on) to a Xen VM is captured by Snort.

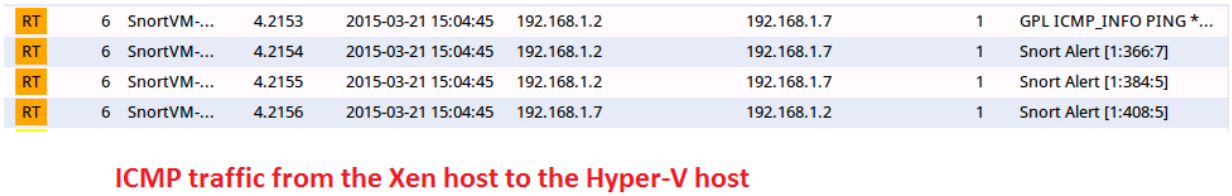

Figure 3.19: Traffic between the two hypervisors is picked up as well.

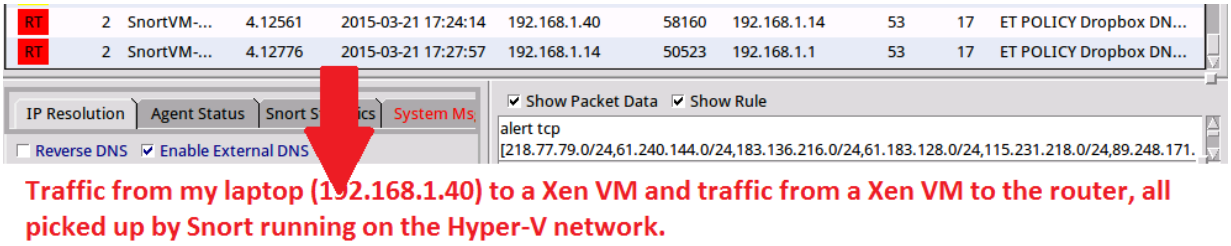

Figure 3.20: Traffic originating from outside either virtual network to any virtual network and traffic destinated to a target outside any virtual network is detected by Snort.

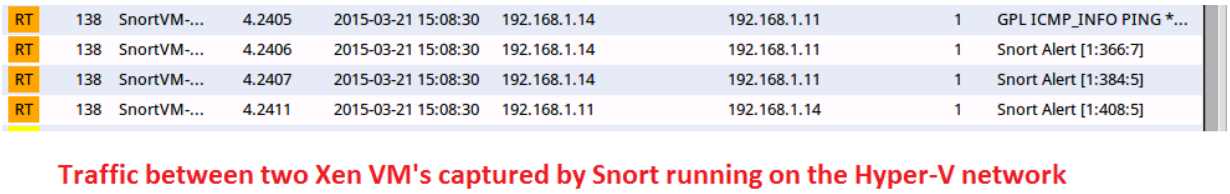

Figure 3.21: Internal traffic between Xen VM's is intercepted by Snort.

So to summarize, the following is possible / is achieved regarding intrusions:

- Detecting intrusions between two VM's running on the Hyper-V virtual network.
- Detecting intrusions between a VM running on the Hyper-V virtual network and the Xen virtual network and vice versa.
- Detecting intrusions between two VM's running on the Xen virtual network.
- Detecting intrusions between the Xen host and Hyper-V host and vice versa.

3.4.3 Actual penetration testing

In order to make sure that Snort is actually protecting our network against numerous and various anomalies, it is essential to test Snort against various attacks. Based on how Snort reacts on the attacks, we will configure Snort, that is, modifying the rules (signatures), add rules if no alerts are raised or removing some of the rules that are triggering false alerts. Not only can one adjust the rules, also thresholds and limitations of how many alerts each rule is allowed to generate can be set.

Of course, false positives will also be generated and the art of configuring Snort is minimizing the amount of false positives and maximizing the generation of alerts of real threads.

3.4.3.1 What types of attacks will be performed?

Various types of network and host attacks will be executed on the network in order to test Snort's reaction. These types include:

- Port scans
  - Basic port scan
  - Advanced port scan
- Webserver attacks
  - XSS
  - SQL Injection
  - Command Injection
- FTP server attacks
  - FTP root access
  - FTP malicious payloads
  - Various other FTP attacks
- SSH attacks
- Database attacks
  - Database scanning
  - Login attempts (including root access)
  - Bruteforce attempts
- Trojan and virus injection / infection
- DOS attacks

3.4.3.2 Motivation for the choice of the attacks

Why were these nine types of attacks chosen? The attacks are based on the actual services provided by some servers in my own home network. I have for example a web server running, an FTP server, a Samba server and a MySQL server. All those servers are running on Linux and to remotely login on those servers, SSH is frequently used.

So that is why webserver, FTP and SMB attacks will be performed. In addition, each hacker starts by scanning the network for running computers and once a computer has been found, a port scan is executed to scan the host for open ports which the hacker can exploit.

3.4.3.3 Testing methods

In order to avoid a too high level of artificial server setup, all the attacks are performed on production servers. Before doing so, the correct functioning of those services is controlled and when it is confirmed that the service / server that is about to be attacked functioning properly, the actual attack is performed.

After the attack has been performed, Sguil is checked for realtime events. Remember that Snort writes its "findings" to the Sguil database. When the attack is listed, it means that Snort has successfully picked up / recognized the attack

Also the DOS attack is performed on two production servers being the FTP server and the webserver. However, the Trojan infection is performed on my own laptop, not a real production server. So I will be infecting my own computer from a VM acting as the attacker.

3.4.3.4 NMAP scanning

Once the attacker knows which hosts are online, he can start querying a host to determine what services are running on the host or what types of protocols the host supports. This is the first phase in a network attack: the reconnaissance phase.

That is why first of all, some NMAP scanning was performed for reconnaisance of open ports and running services of the target hosts. I started with an unfragmented scan on a Hyper-V VM from the webserver, which has NMAP installed.

Unfragmented packets First ....

```
laurent@atlas:~$ nmap -v -A -Pn 192.168.1.51
```

Figure 3.22: The NMAP command as executed on the webserver (atlas, 192.168.1.11).

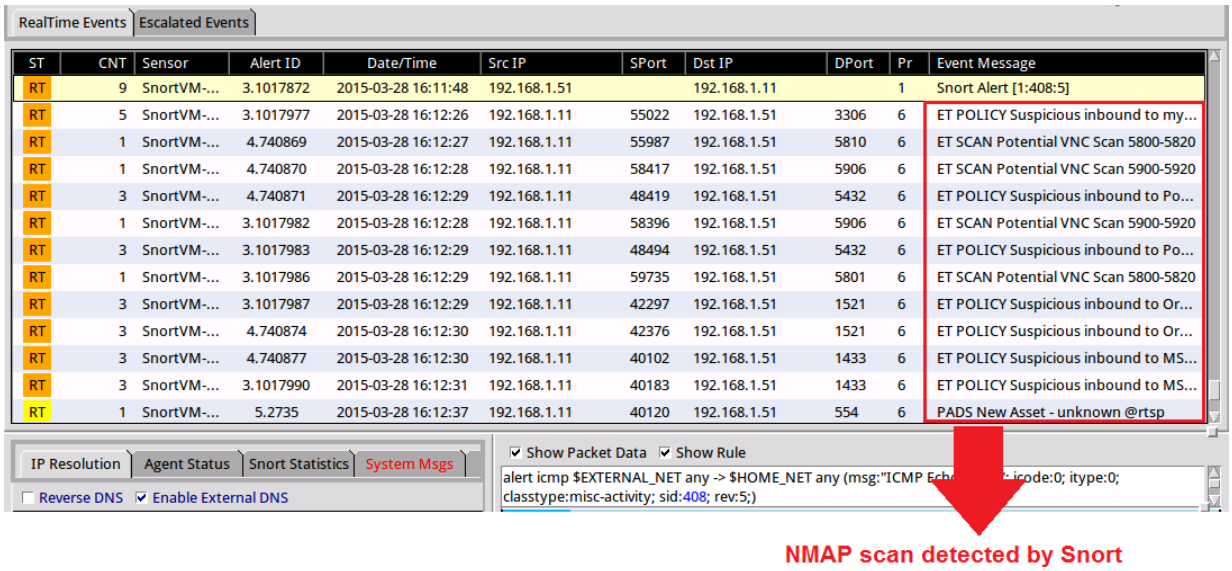

Figure 3.23: Basic, unfragmented NMAP scanning of a Hyper-V VM (192.168.1.51). The SnortVM has the IP address of 192.168.1.50. Snort reports each attempt to scan a particular port number.

Fragmented SYN packets Next, I performed an NMAP scan with fragmented packets, which splits up the TCP header over several tiny packets to trick / fool IDSs and firewalls. This way, an attacker hopes to evade packet filters.

The NMAP command executed is the following:

```
^Claurent@atlas:~$ sudo nmap -T4 -v -A -Pn -sS -f 192.168.1.51

Starting Nmap 6.40 ( http://nmap.org ) at 2015-03-28 17:19 CET
NSE: Loaded 110 scripts for scanning.
NSE: Script Pre-scanning.
Initiating ARP Ping Scan at 17:19
Scanning 192.168.1.51 [1 port]
Completed ARP Ping Scan at 17:19, 0.22s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 17:19
Completed Parallel DNS resolution of 1 host. at 17:19, 0.00s elapsed
Initiating SYN Stealth Scan at 17:19
```

Figure 3.24: The NMAP stealth, SYN packet command as executed on the webserver (atlas, 192.168.1.11).

The stealth, fragmented ICMP ping scan is not detected by Snort. However, Snort did detect the scan for running services as can be seen in the following figure.

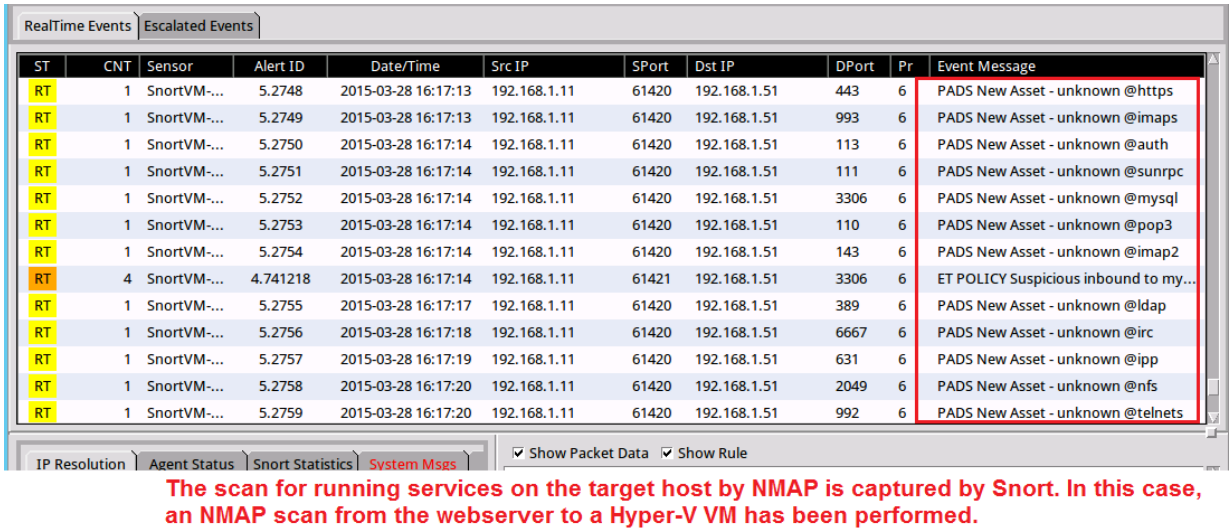

Figure 3.25: The scan for running services from the stealth scan is detected by Snort.

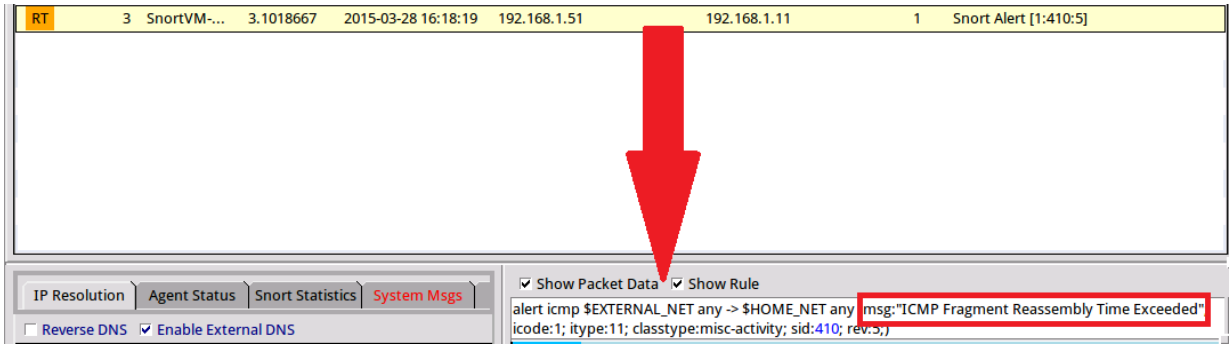

Figure 3.26: However, this Snort alert indicates that a host reassembling a fragment datagram cannot complete the reassembly due to missing fragments whitin the time limit (60s by default). However, I'm not sure whether this is Snort warning for a fragmented / stealth scan.

Then I performed an ICMP ping to the Hyper-V VM (192.168.1.51) with a size of 1000 bytes. This gets detected by Snort right away.

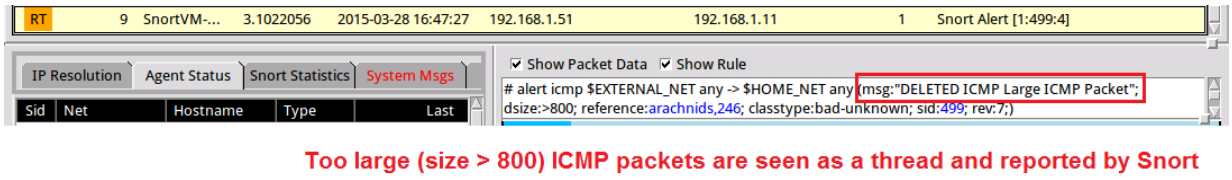

Figure 3.27: ICMP ping with large packet size is detected by Snort.

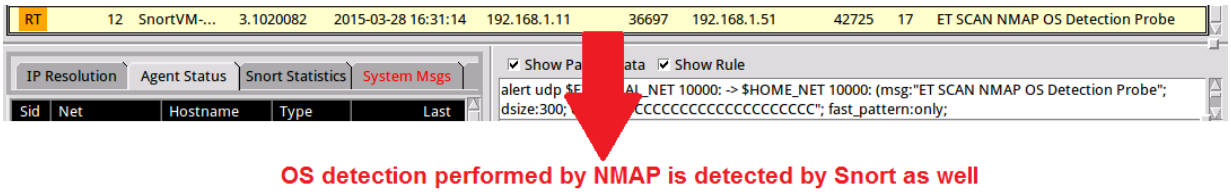

Figure 3.28: OS detection from a NMAP scan is also detected by Snort.

3.4.3.5 Preventing port scanning with Snort

In section 3.4.2.10, an overview of the different action types of Snort rules has been supplied. One of these actions is `drop`, which causes the matched packets to be blocked. Therefore, different network attacks including port scanning can be prevented.

As an example, consider the following rule:

alert icmp any any → any any (msg:"ICMP ping"; sid:100002;)

This rule will generate alerts when ICMP Ping packets are intercepted. However, when changing `alert` to `drop`, ICMP ping packets are filtered out of (from) the network and thus preventing port scanning.

This process can be repeated for every rule that generated alerts. Of course, in order to improve security, one can also configure the firewall to restrict the effectivity of port scans. Most effective way to do so is denying all traffic and only allowing the traffic needed to access the internal services. Also, tracking malicious activity over time and configuring rules to cut off attacks when a treshold is reached. For example, when 50 consecutive ICMP Ping requests are detected.

3.4.3.6 Webserver attacks

This section will cover some common webserver attacks, such as XSS attacks, SQL injections and command injections.

3.4.3.7 Cross-site scripting (XSS) attack

Sometimes, one can abuse an URL of a webpage to inject JavaScript code into the page. This is called a cross-site scripting (XSS) vulnerability. If a hacker gives the modified URL to someone else and he can get this person to click on the modified links, a hacker / attacker can achieve the following: change the content of the page, steal cookie values,

gain access to the user's history, ... This is why we want to detect this.

Therefore, I made a vulnerable PHP webpage myself, to exploit an XSS atack. The page takes the "name" parameter from the URL and displays it on the page. But this makes is ideal for an XSS attack.

Figure 3.29: The vulnerable webpage before the attack. It displays the string value that is provided in the "name" parameter in the URL.

Then, the following script is injected into the page:
<script>alert('XSS vulnerability')</script>.
Which yields following output:

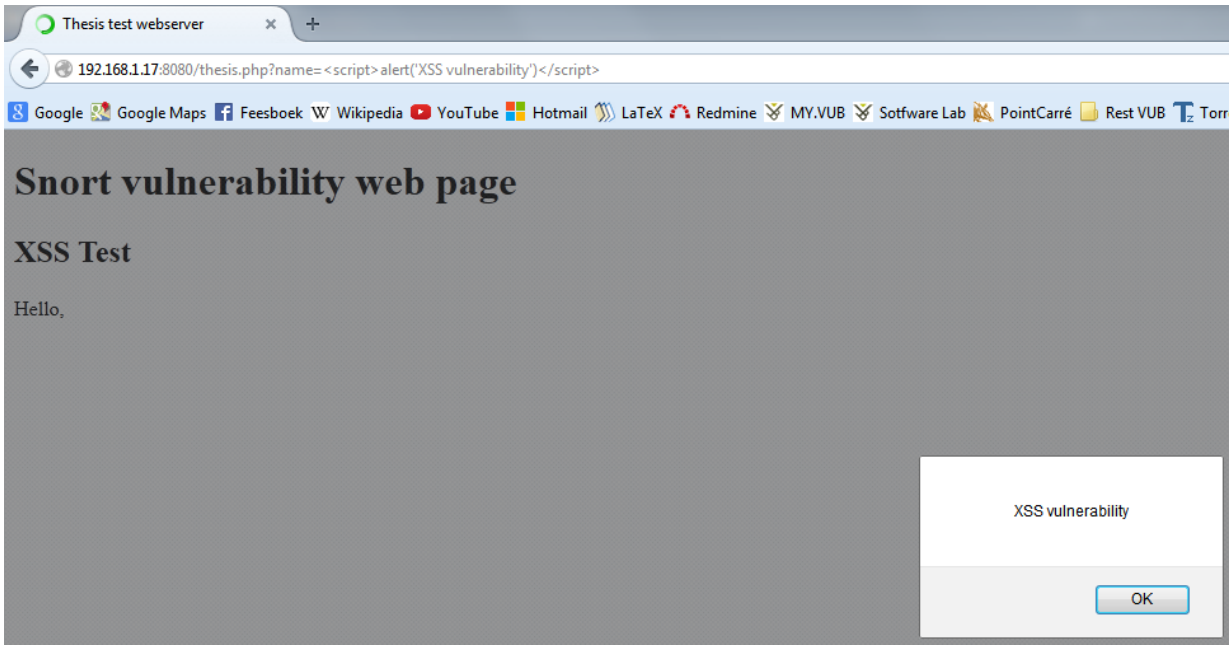

Figure 3.30: The script in action....

As one can observe, the script is indeed successfully injected into the webpage.

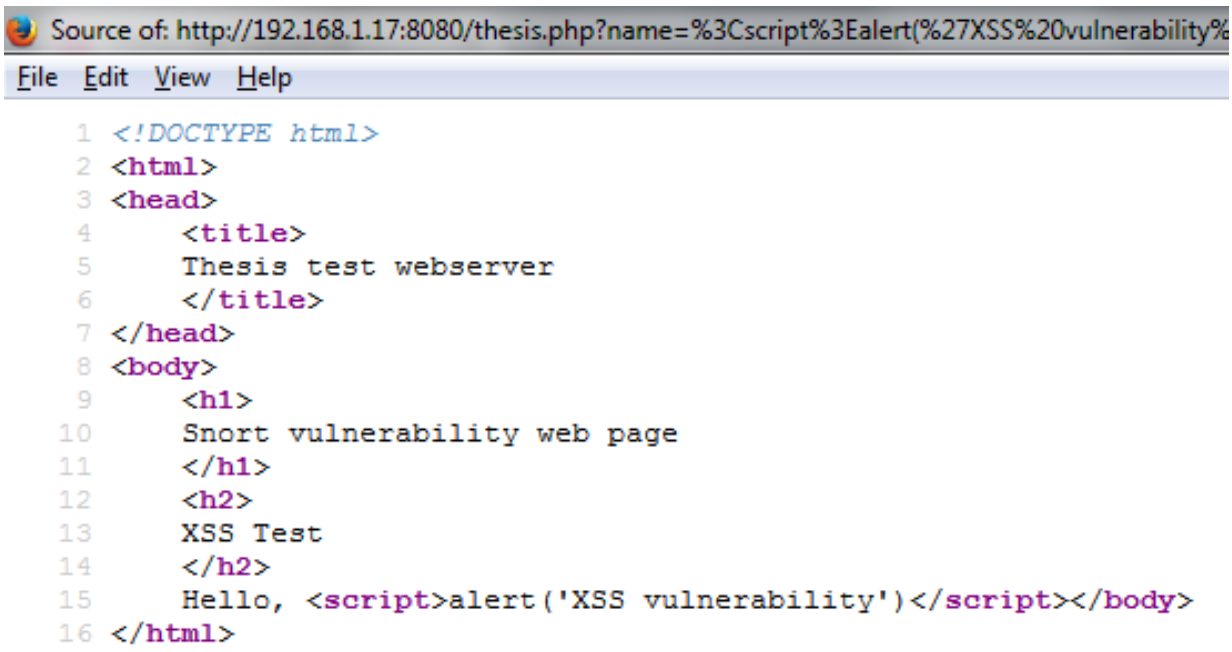

Figure 3.31: The script is successfully injected into the PHP page.

How did Snort react? Snort detects this anomaly right away, without the need for adding extra rules, as can be seen in the following screenshot. However, one can also avoid this attack from happening by changing the "action" method in the rule from "alert" to "drop".

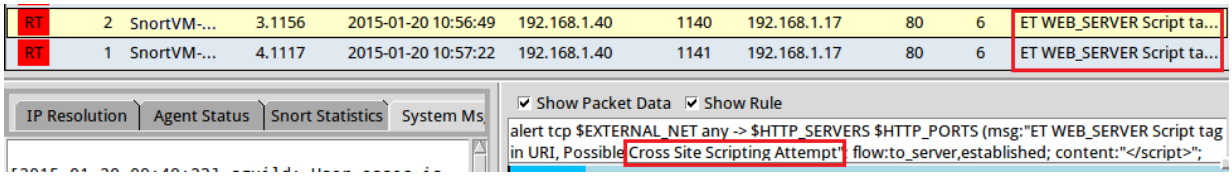

Figure 3.32: Fortunately, Snort detects the XSS attack without the need for adding additional rules.

```
drop tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"ET WEB_SERVER Script tag in URI, Possible Cross Site Scripting
Attempt"; flow:to_server,established; content:"</script>"; fast_pattern:only; nocase; http_uri; reference:url,ha.ckers.org/xs
s.html; reference:url,doc.emergingthreats.net/2009714; classtype:web-application-attack; sid:2009714; rev:6;)
```

Figure 3.33: To prevent XSS attacks from happening, one can change "alert" to "drop" in the rules that triggers the alert.

3.4.3.8 SQL Injection

SQL Injection is an attack technique in which users can insert SQL commands as strings that are passed to an SQL server, via a webpage input [**?**]. It can be used to read sensitive data from a database or to perform administrative tasks to the database, for example: shutting down the DBMS.

To detect such attacks, a custom PHP webpage has been created that is vulnerable to an SQL injection attack.

To provide the reader a general overview, the entire content of the MySQL "Persons" table is displayed on the webpage. Obviously, in reality, nothing is displayed when a user first loads the page. But the content is displayed for information purposes.

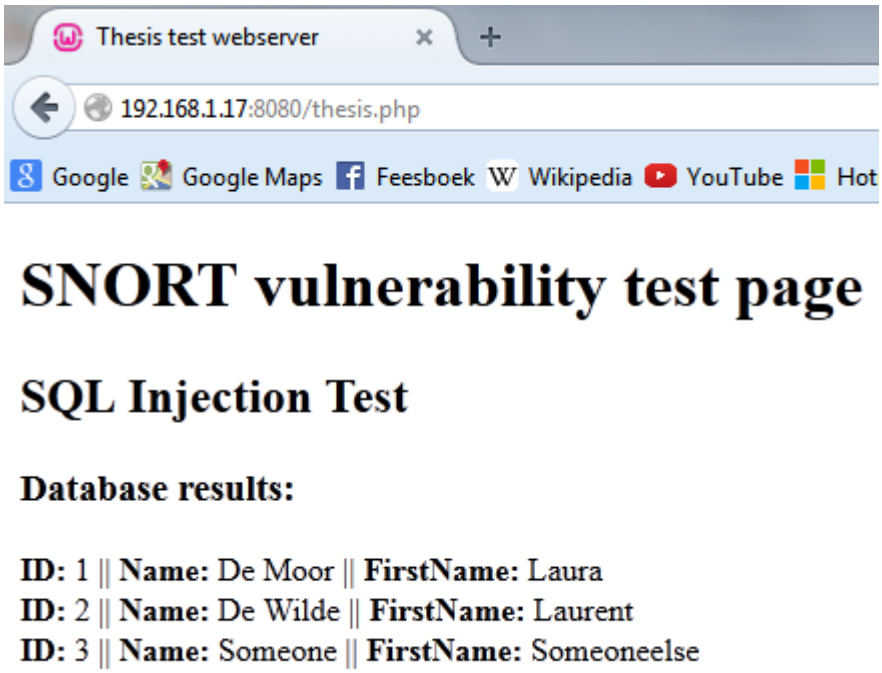

Figure 3.34: Persons table in the test database running on MySQL 5.6 populated with 3 records.

Then, an input textfield is created where the user can enter the firstname of a person he is looking for. The source code is listed below:

```php
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test";


$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

if($_POST['s']){
    $user = $_POST['user'];
    $txt_sql = "SELECT * FROM ossec WHERE FirstName = '$user'";

$result = mysqli_query($conn, $txt_sql);


if (mysqli_num_rows($result) > 0) {

echo "<h3>Database results:</h3>";
while($row = mysqli_fetch_assoc($result)) {
    echo "<b>ID:</b> " . $row["PersonID"]. "  || <b>Name:</b> " . $row["Name"]. "  || <b>FirstName:</b> " . $row["FirstName"]. "<br>";
}
} else {
echo "0 results";
}
}
```

Figure 3.35: Persons table in the test database running on MySQL 5.6 populated with 3 records.

The next screenshot shows an example of how a POST request can look like. In this example, the user has entered the search string "Laurent".

Source of: http://192.168.1.17:8080/thesis.php - Mozilla Firefox

File Edit View Help

```
 1 <!DOCTYPE html>
 2 <html>
 3   <head>
 4     <title>OSSEC Test webserver</title>
 5   </head>
 6
 7   <body>
 8     <h1>
 9     SNORT vulnerability test page
10     </h1>
11     <h2>
12     SQL Injection Test
13     </h2>
14     <form action="" method="post">
15     <table width="20%">
16     <tr>
17         <td>FirstName</td>
18         <td><input type="text" name="user"></td>
19     </tr>
20     </table>
21     <input type="submit" value="OK" name="s">
22     </form>
23     <h3>Database results:</h3><b>ID:</b> 2  || <b>Name:</b> De Wilde  || <b>FirstName:</b> Laurent<br>  </body>
24 </html>
```

Figure 3.36: Persons table in the test database running on MySQL 5.6 populated with 3 records.

Of course, when the textfield is empty and the query is submitted, nothing is displayed. But when an attacker enters the string "OR 1 = 1", he gets to see all the information

stored in the Persons table. This is called SQL injection based on $1 = 1$ is always true. It is an attempt to make a query succeed no matter what.

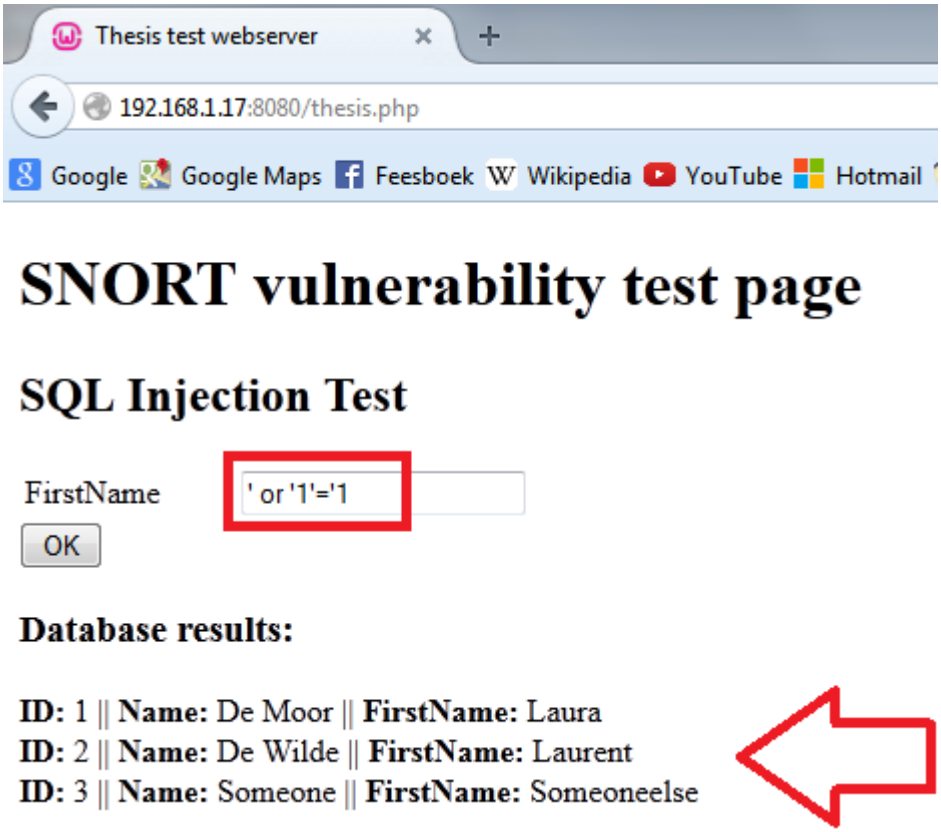

Figure 3.37: SQL injection attack in action with all the results displayed on the webpage.

And did Snort react on this type of webserver attack? Yes it did, as can be seen in the following screenshots:

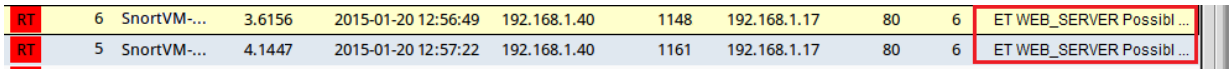

Figure 3.38: Snort alerts for a possible SQL injection.

```
drop tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"ET WEB_SERVER Possible SQL Injection Attempt SELECT FROM";flow:
established,to_server; content:"SELECT"; nocase; http_uri; content:"FROM"; nocase; http_uri; pcre:"/SELECT.+FROM/Ui"; referen
ce:url,en.wikipedia.org/wiki/SQL_injection; reference:url,doc.emergingthreats.net/2006445; classtype:web-application-attack;
sid:2006445; rev:10;)
~
```

Figure 3.39: The triggering rule. Once again, one could change “alert” to “drop” in order to prevent the SQL injection from happening.

3.4.3.9 Command injection

A command injection attack is an attack in which a hacker tries to execute commands on the host operating system using a vulnerable (web)application.

To simulate this attack, I created a vulnerable PHP webpage that returns the content of a file that the user requested. It does this by executing the "type" command on the host machine. But as one may have guessed, an attacker can also execute a command this way.

Following screenshot displays the source code of the webpage.

```
    <h2>
    Command Injection Test
    </h2>
<?php
    $name = "nobody";
    if(isset($_GET['filename']) && $_GET['filename'] != ""){
        $name = $_GET['filename'];
    }
    echo shell_exec('type '.$_GET['filename']);
?>
  </body>
```

Figure 3.40: The source code of the webpage. Note the "shell_exec" statement.

An actual example of command injection is listed below. Here, the "ipconfig" command is executed on the target host and all information about the network interfaces is returned.

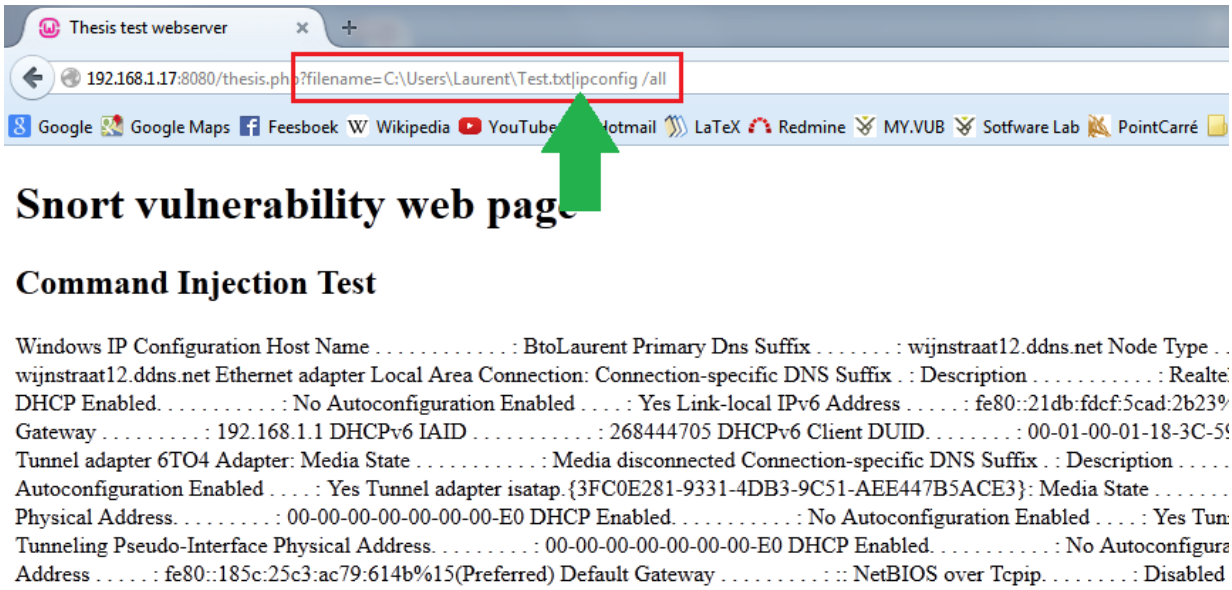

Figure 3.41: A command injection attack has just been launched and the results of executing the `ipconfig /all` command are displayed on the webpage.

Another example is provided below: when supplying the `netstat` command to the URL, an intruder can also obtain information about which ports are currently opened.

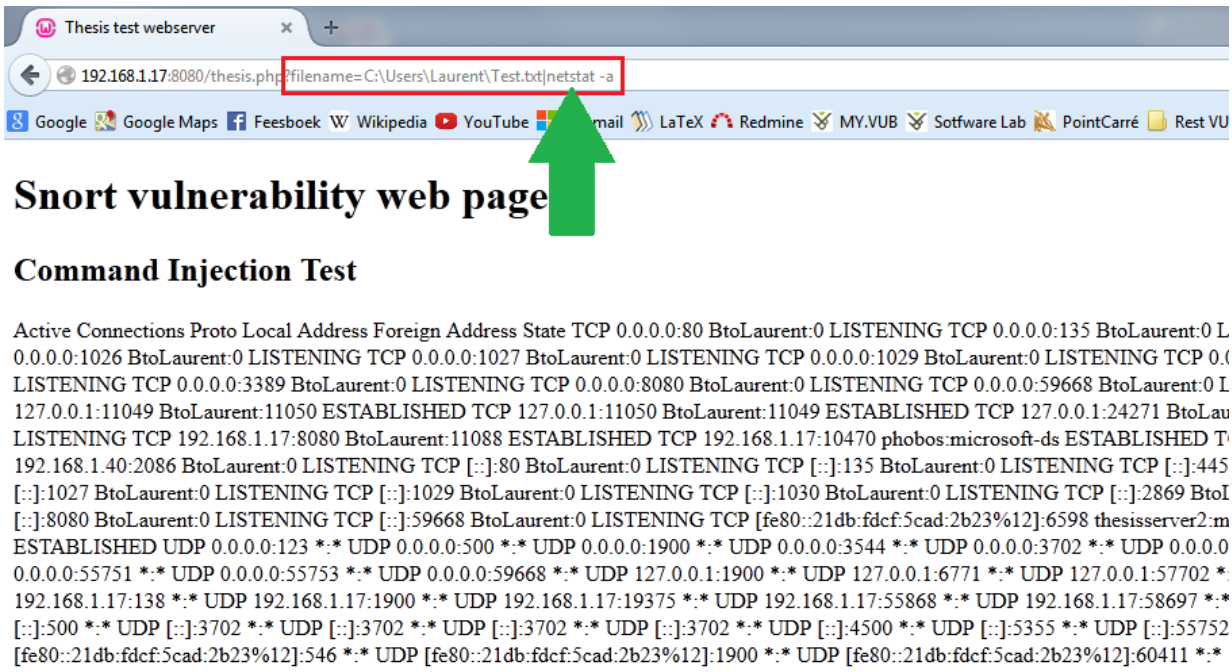

Figure 3.42: Output of the `netstat -a` command displayed on the web page.

How did Snort react on this thread? Well, at first, i did NOT. Meaning that there are some extra rules needed. Especially, we want to look for executable files that are passed in the URL, because an attackers usually wants to execute programs (commands) on the host machine. However, it turned out that Snort actually already had a rule that alerts for this kind of attack, but it turned out this rule had been disabled. So enabling the rule was sufficient to make Snort alert on command injection attacks.

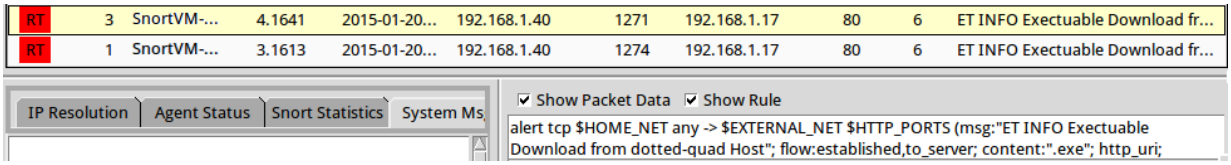

Figure 3.43: Snort alerting for command injections.

Again, changing the action from `alert` to `drop` will cause Snort to block these packets and thus preventing this command injection from happening.

3.4.3.10 FTP server attacks

Next, some attacks on the FTP server `vsftpd` running on a Xen virtual network are executed to see how Snort reacts on this. The FTP server has IP address 192.168.1.16 and runs on a Xen VM called "farbauti". The client computer has the IP address 192.168.1.40 and runs a Windows 7 Professional 64 edition. All FTP attacks were launched using FileZilla Client 3.10.
The following will be tested: attempting to login as root user and attempting to attack the FTP server with various attack methods, e.g.: command overflow, format string, ...
With a **command overflow**, a cracker sents overly long commands to the FTP server (that is, with a high amout of buffer content). This leaves the hacker the opportunity to execute code on the FTP server.
When the `printf-style` **format specifier** is sent as argument to some FTP commands, it causes data on the stack to be overwritten. This is because the FTP server trying to process data containing such format strings, eventually resulting in the possibility to execute code on the FTP server.
Remember that the Snort VM runs on the Hyper-V network and has IP address of 192.168.1.50.

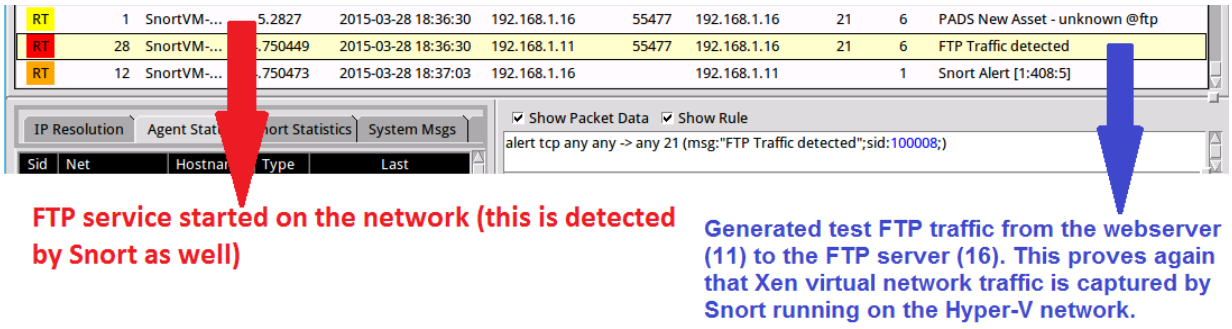

Figure 3.44: First, I created a rule to actually detect FTP traffic as I plan to DOS attack the FTP server is a later stage. The starting of the FTP service and some FTP traffic are detected by Snort.

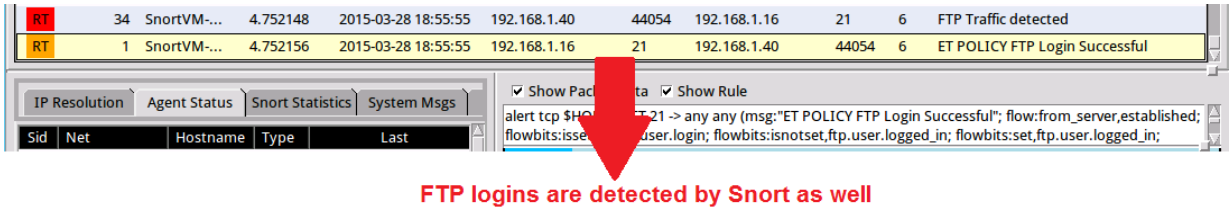

Figure 3.45: Successful FTP logins are also detected by Snort (however, this is not a thread and can be disabled by simply comment the rule that triggered the alert.

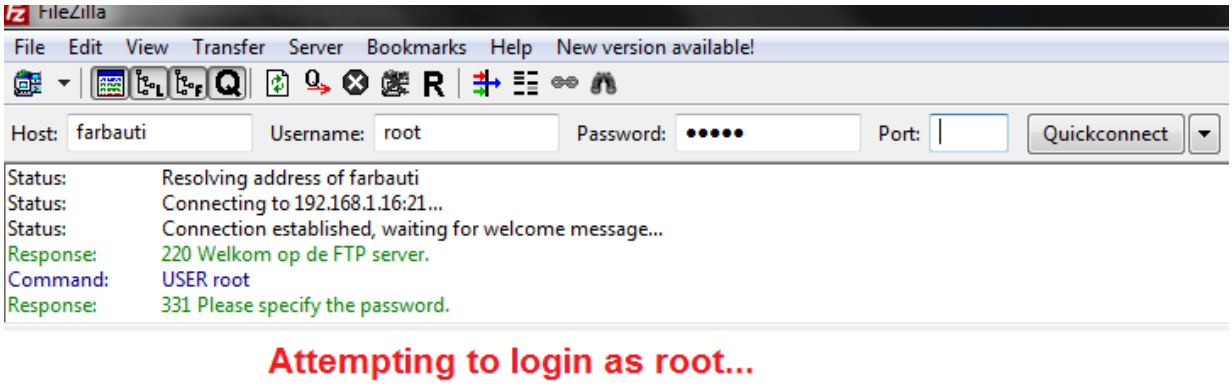

Figure 3.46: Attempting to login as root.

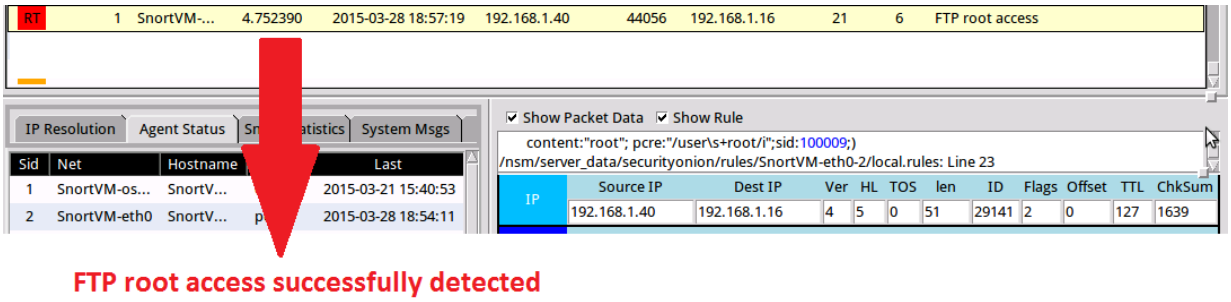

Figure 3.47: FTP root access is successfully detected.

For the actual FTP server attacks, I used Metasploit's db_autopwn command on port 21 on target host 192.168.1.16.

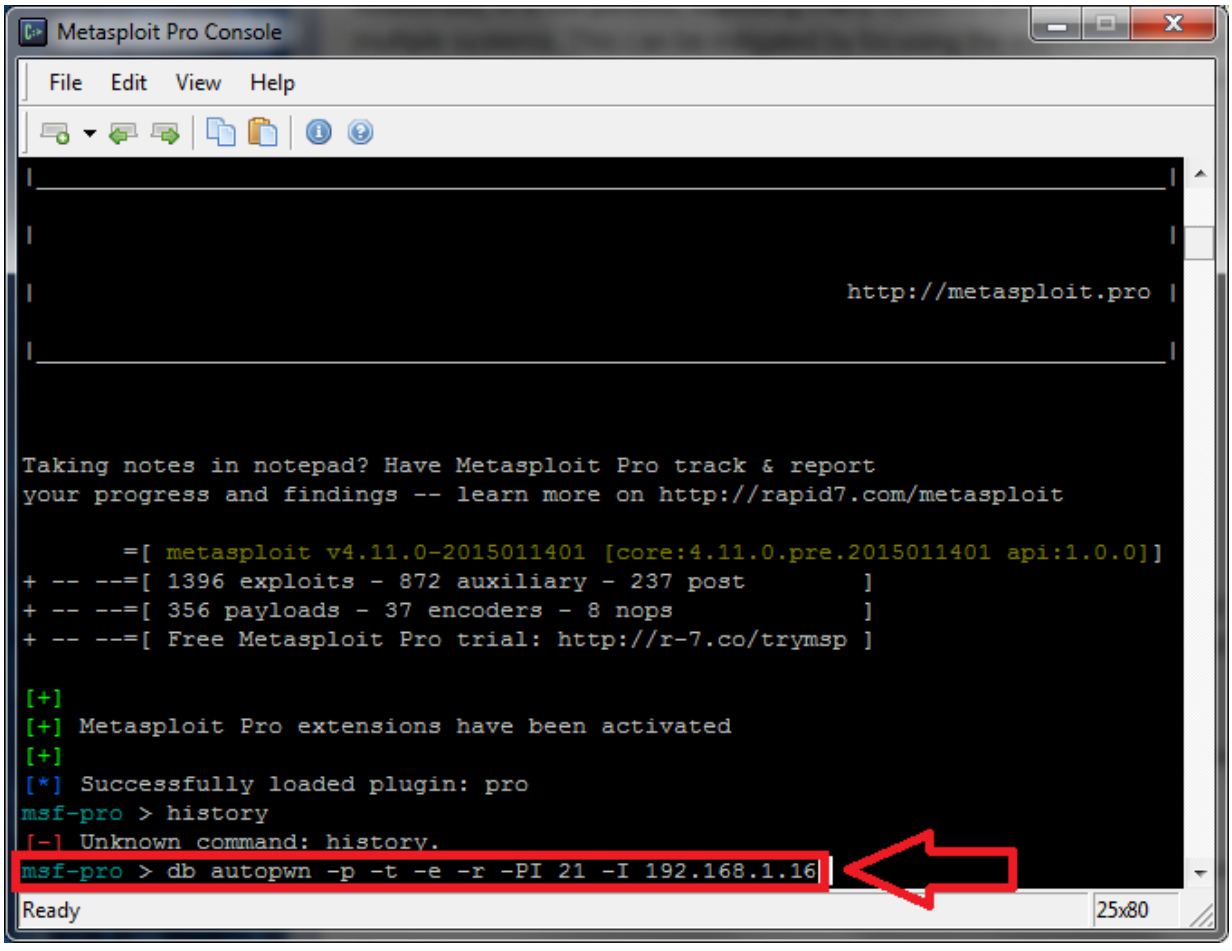

Figure 3.48: The command to attack the FTP server as seen in Metasploit.

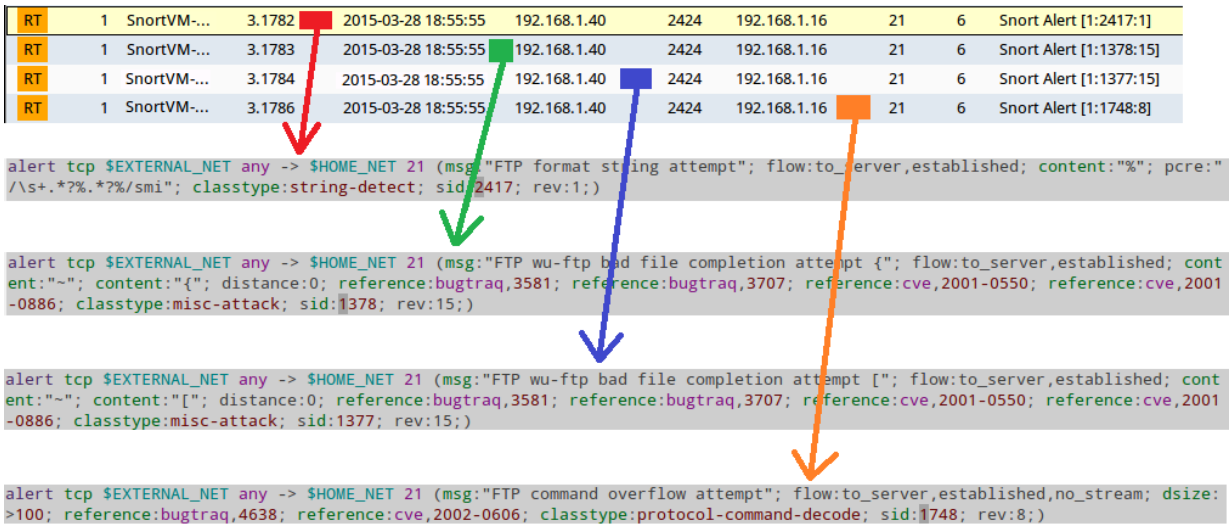

Figure 3.49: Snort reported the various attacks.

3.4.3.11 Preventing FTP server attacks

Some early versions of ProFTPd are vulnerable to command overflow attempts, so installing the latest software - and security updates of the installed FTP server is the first step in denying hackers taking over control of the system. One might also consider moving on onto more secure FTP servers such as `vsftpd`. While peforming the attack using Metasploit, the `vsftpd` server did not go offline, nor did the penetration attempts such as command overflow attempts succeeded.

3.4.3.12 SSH attacks

There was no need to simulate an SSH attack, as the next screen capture reveals:

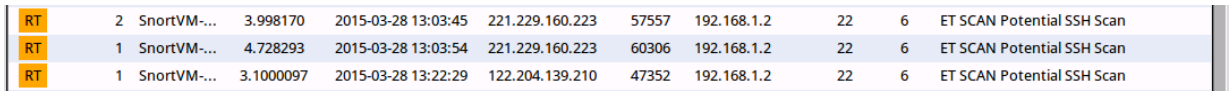

Figure 3.50: Appearantly, someone tried to SSH scan my Xen server.... This was fortunately detected by Snort.

How can SSH attacks be prevented?

- Allow only one IP address to be connected to the SSH server.
- Run the SSH daemon on a non-standard port.

- Make use of `hosts.allow` and `hosts.deny` to allow only certain IP addresses to connect to the server via SSH.

```
# /etc/hosts.allow: list of hosts that are allowed to access the system.
#                   See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: LOCAL @some_netgroup
#             ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#
sshd: 192.168.1.2
sshd: 192.168.1.40
```

Figure 3.51: An example of `hosts.allow`. In this example, only computer with IP address of 192.168.1.2 and 192.168.1.2 are allowed to connect to the server through SSH.

```
# /etc/hosts.deny: list of hosts that are _not_ allowed to access the system.
#                  See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: some.host.name, .some.domain
#             ALL EXCEPT in.fingerd: other.host.name, .other.domain
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#
# The PARANOID wildcard matches any host whose name does not match its
# address.
#
# You may wish to enable this to ensure any programs that don't
# validate looked up hostnames still leave understandable logs. In past
# versions of Debian this has been the default.
# ALL: PARANOID
sshd: ALL: DENY
```

Figure 3.52: An example of `hosts.deny`. This file is used in combination with `hosts.allow` and indicates that only IP addresses of 192.168.1.2 and 192.168.1.40 are allowed to connect to the server through SSH and that all other IP addresses have their access denied.

- Use `hashlimit` in `iptables` to, for example, allow only one SSH connection per IP address per minute.

3.4.3.13 Database server attacks

A scan looking for MySQL databases on the network has been executed, as well as commands to show the available databases on the server and root login.

For the database scan, Metasploit has been used again. The IP address of the MySQL server is 192.168.1.23 and the MySQL database runs on a Xen VM called `mimas`.

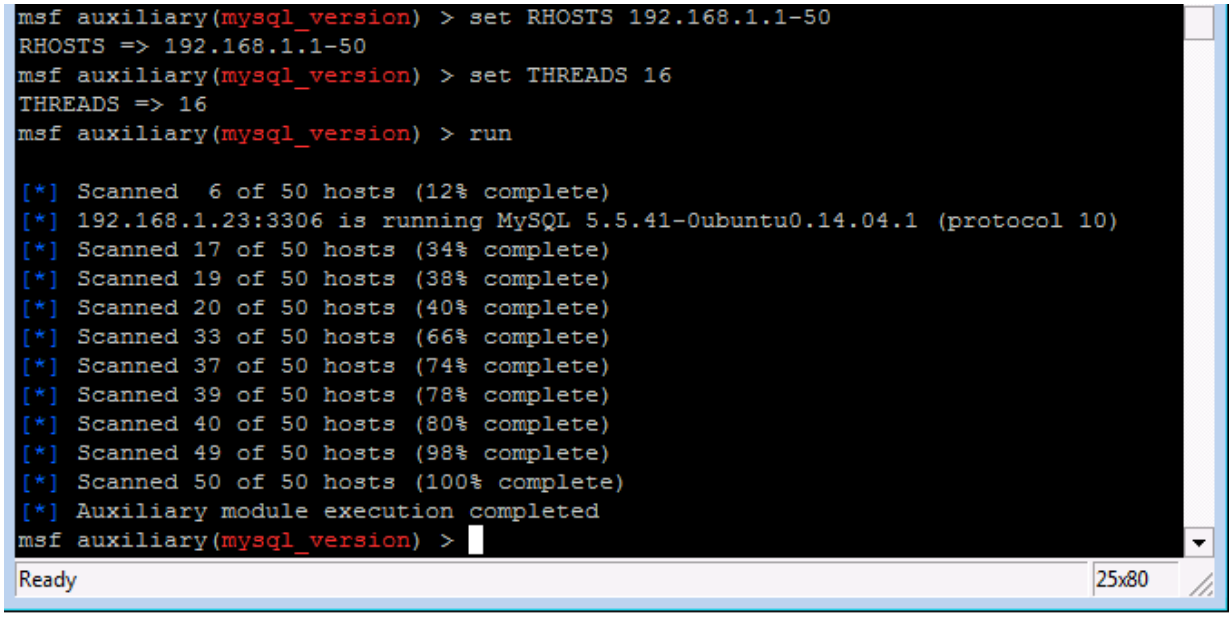

Figure 3.53: Metasploit is scanning the network for databases...

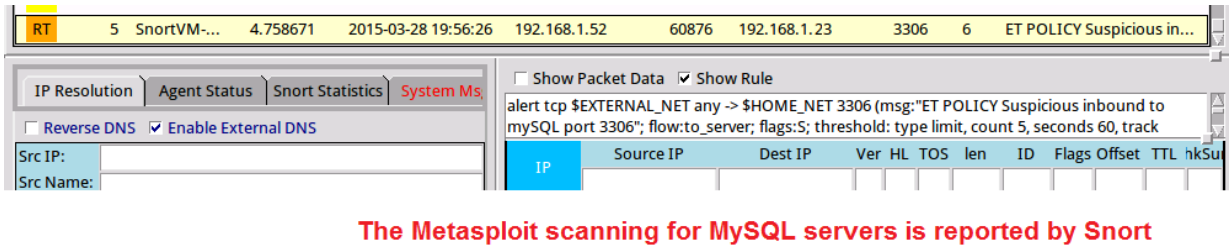

Figure 3.54: ... and this is detected by Snort.

Then I executed the "show databases" on the terminal of one of the Xen VM's.

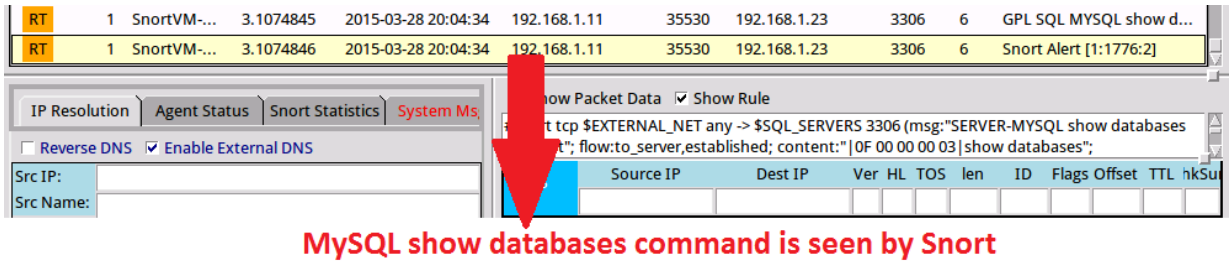

Figure 3.55: This is captured by Snort.

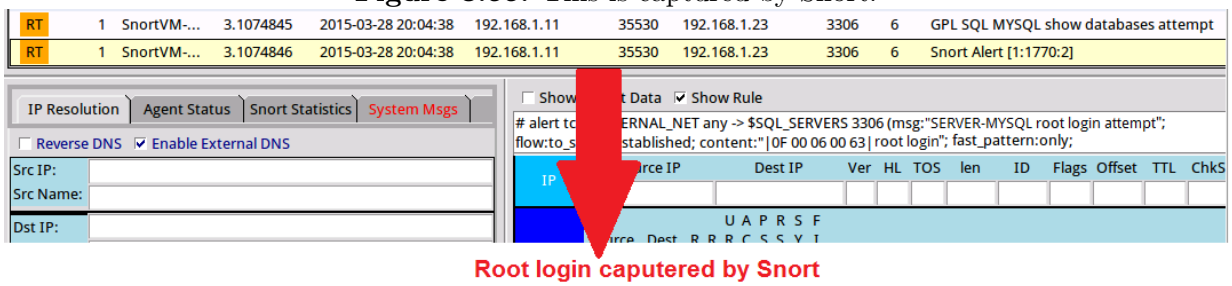

Figure 3.56: Also logging in a root is detected by Snort.

3.4.3.14 Trojan Infections

I created a Trojan Horse to test Snort against Trojan infections and to prove that the default settings of Windows Firewall are not secure enough. The Trojan is a program with a malicious payload that is created on my computer (the attacker), is transfered to the victim and executed by an ordinary user who thinks the program is harmless.

I misuse the fact that the default setting of Windows Firewall allows all outbound connections: I make use of reverse TCP, which means that the victim establishes the connection to the attacker, instead of the other way around (because incoming access is blocked by Windows Firewall).

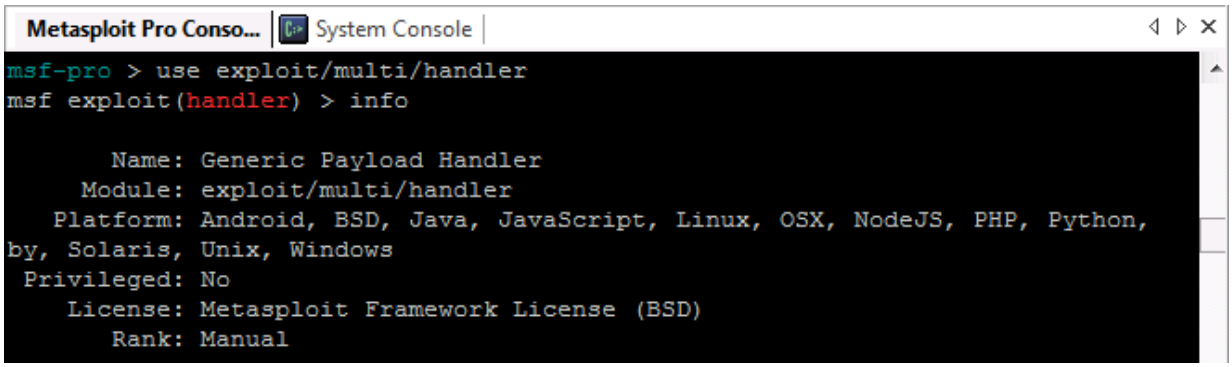

Figure 3.57: The plugin to create the malicious payload.

Figure 3.58: The actual creation of the malicious payload. The "LHOST" stands for Local HOST and indicates that the trojan makes a connection with my (attacking) computer via port 4444.

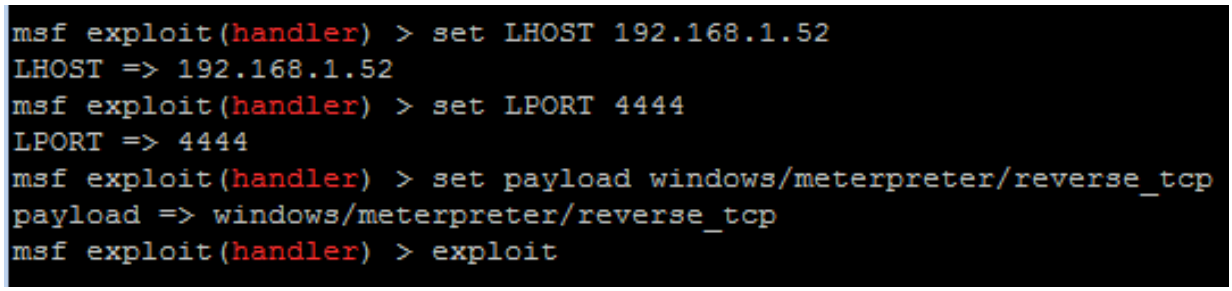

Figure 3.59: Preparing the listener for when an unsuspicious user clicks on the file.

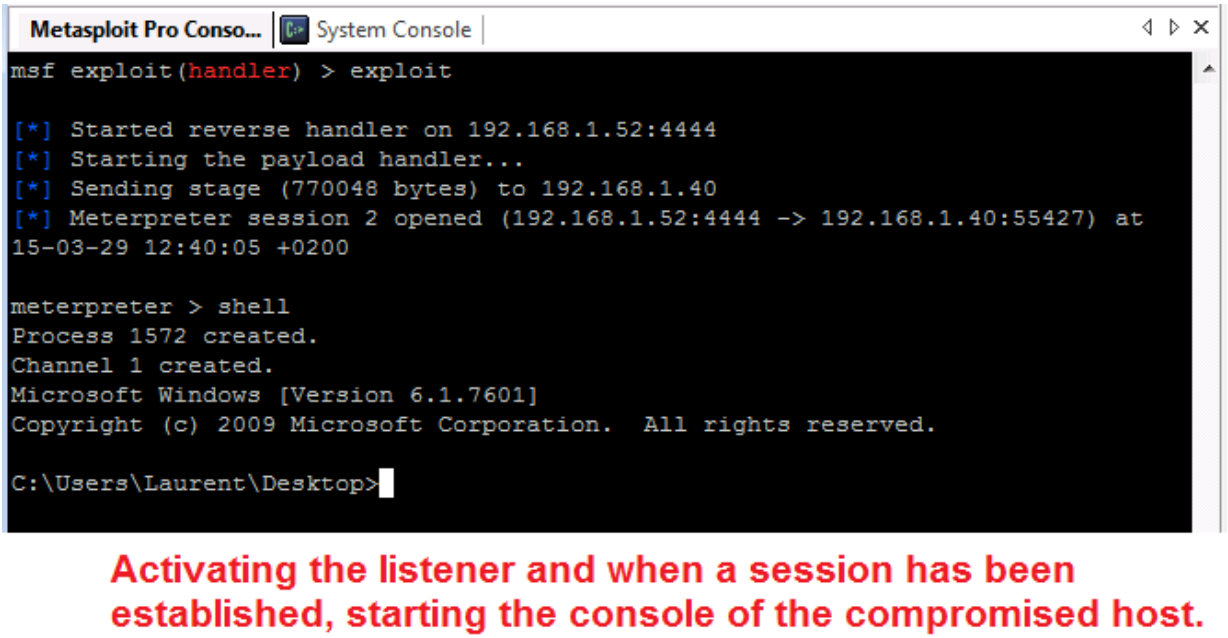

Figure 3.60: A user clicks on the file and a connection between my computer and the victim is established.

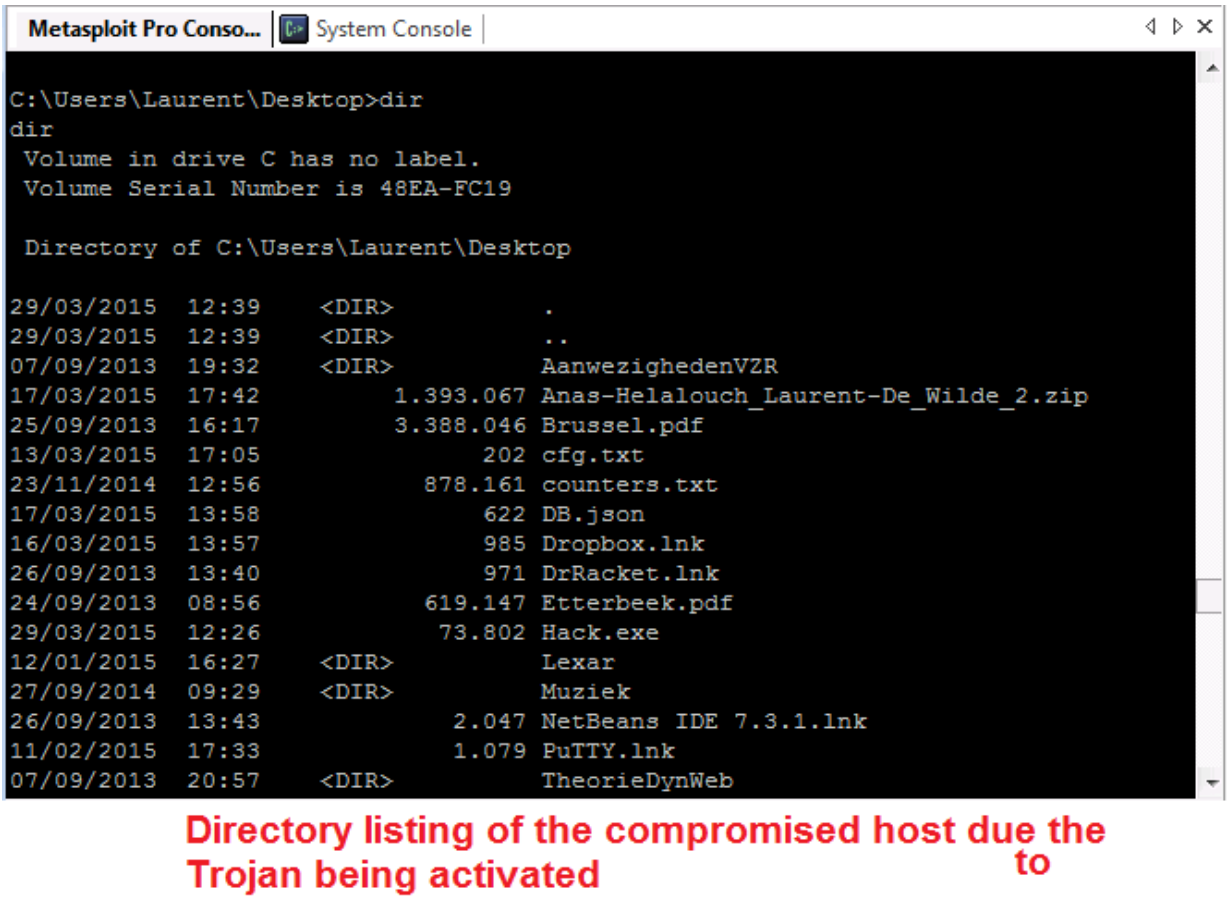

Figure 3.61: Now I can for example browse the hard disk drive of the victim's computer...

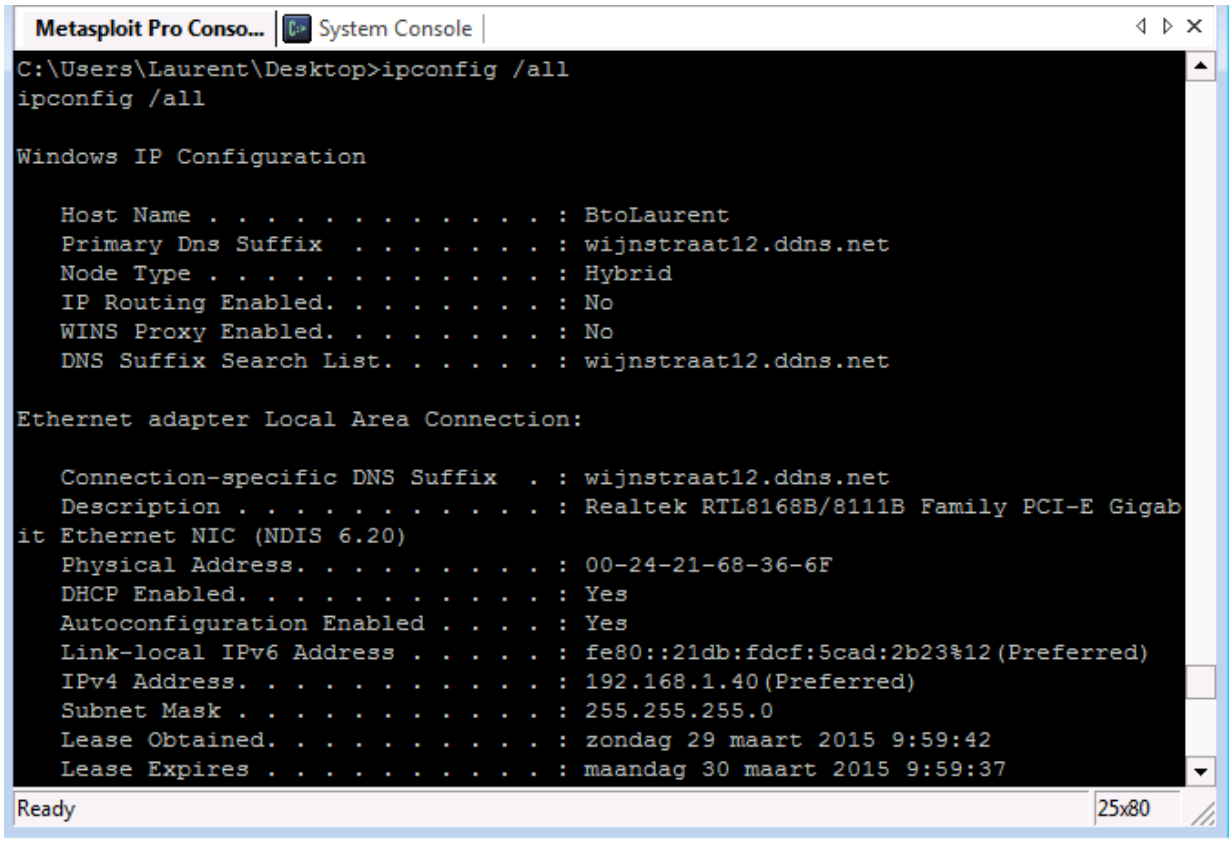

Figure 3.62: . . . or obtain some network information to prepare for subsequent attacks.

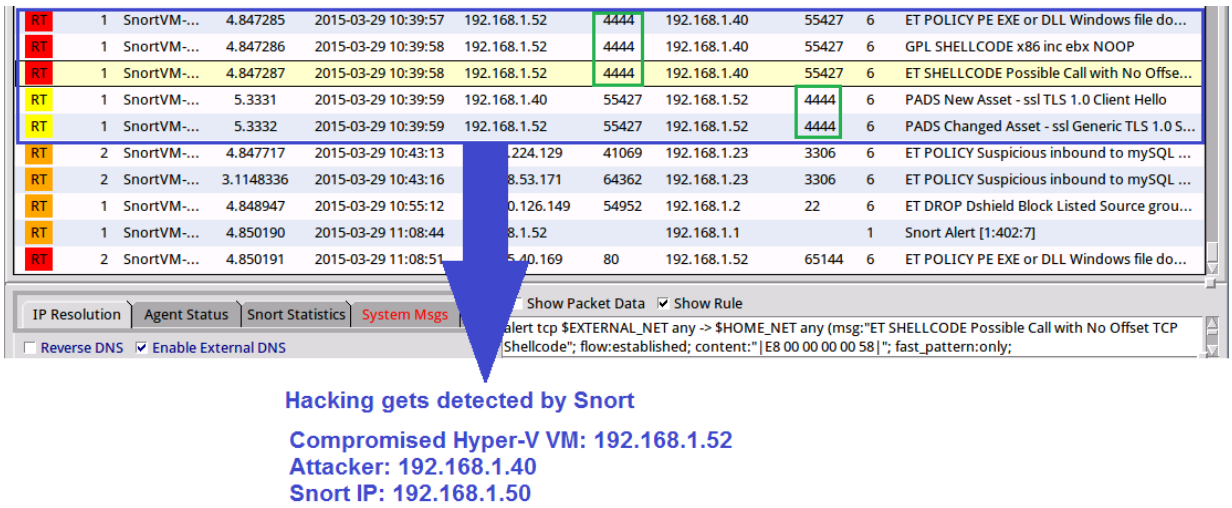

Figure 3.63: Fortunately, this is detected by Snort.

Creating this Trojan, I proved that it possible the get around the Windows Firewall and that also the outbound connections must be restricted.

3.4.3.15 DOS attacks

Using LOIC (Low Orbit Cannon), I performed a DOS attack on an FTP - and HTTP server.

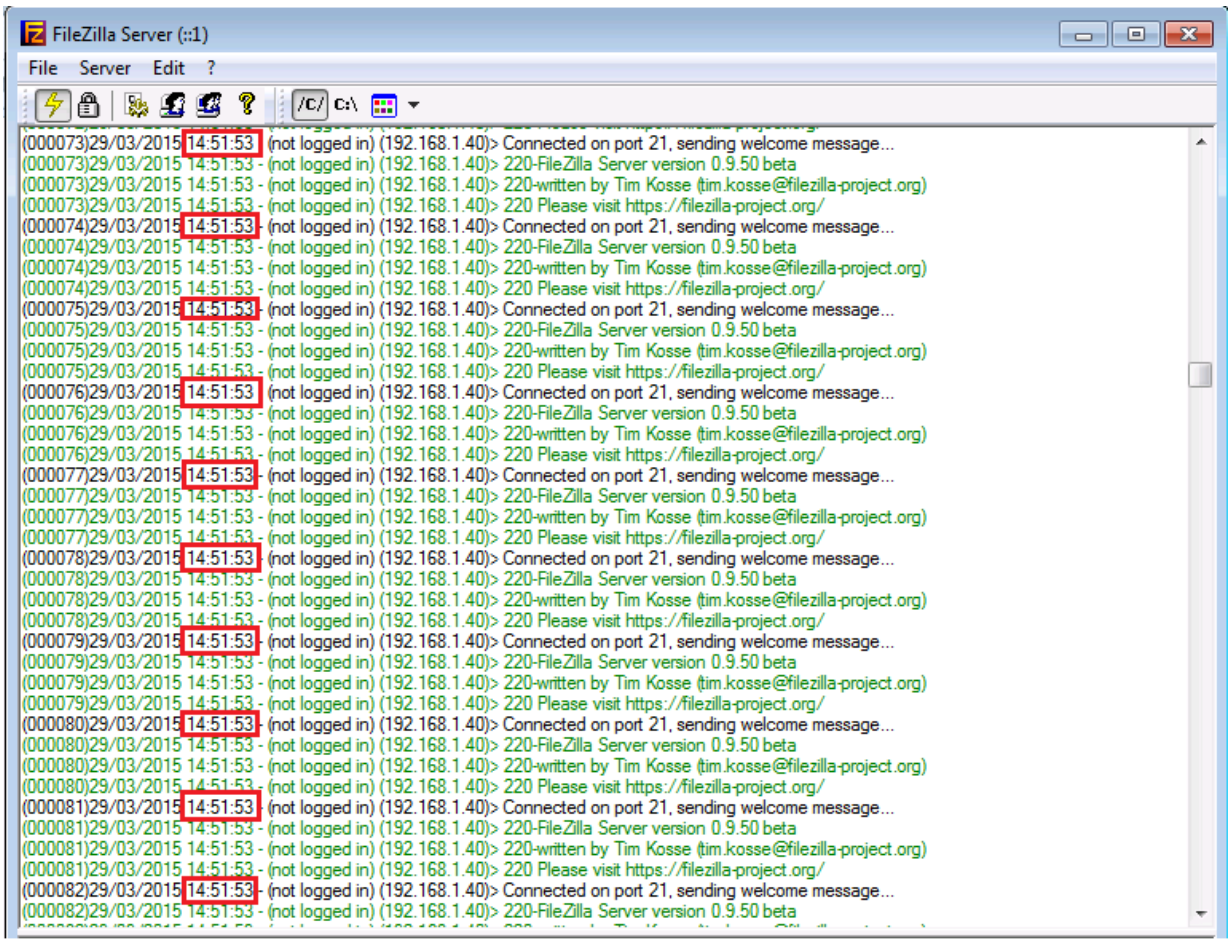

Figure 3.64: The FTP server receives a lot of login attemps per second. This way, we hope to flood it and eventually make it go offline.

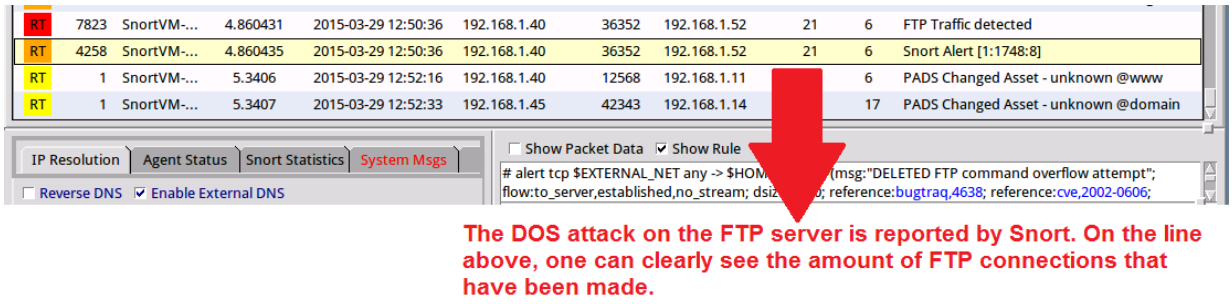

Figure 3.65: Snort reacts.

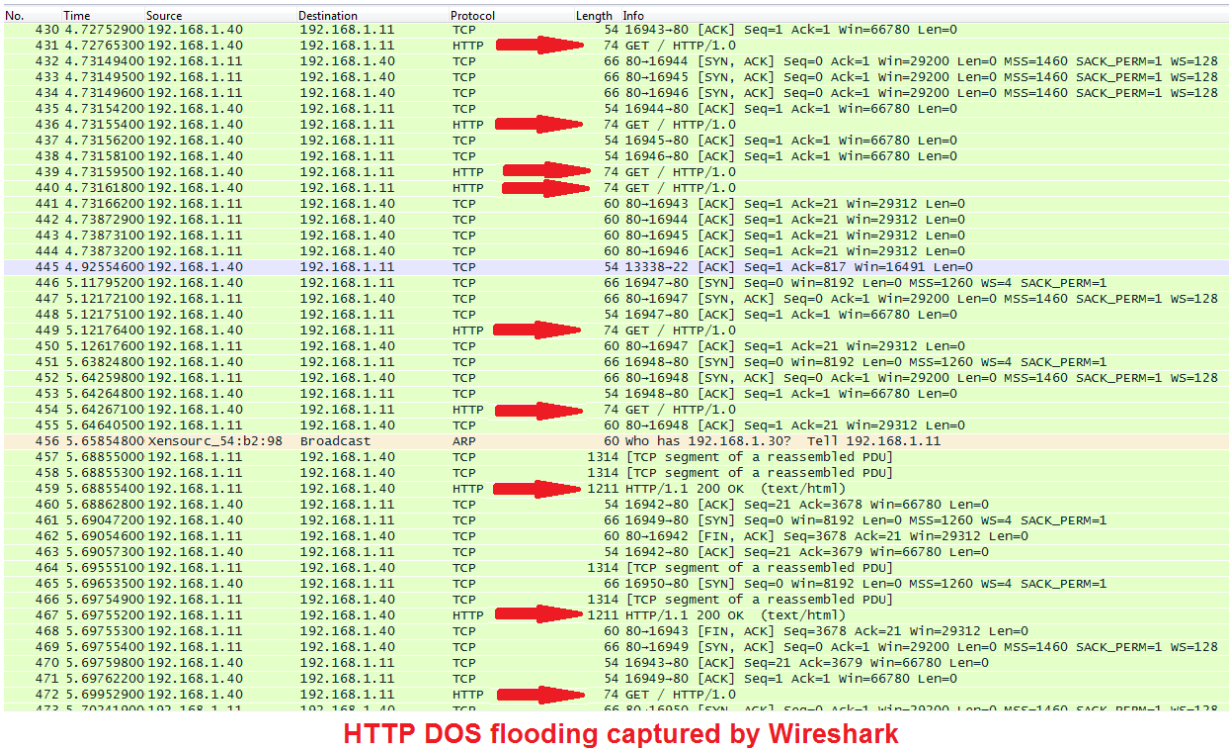

Figure 3.66: The DOS attack on the webserver in action...

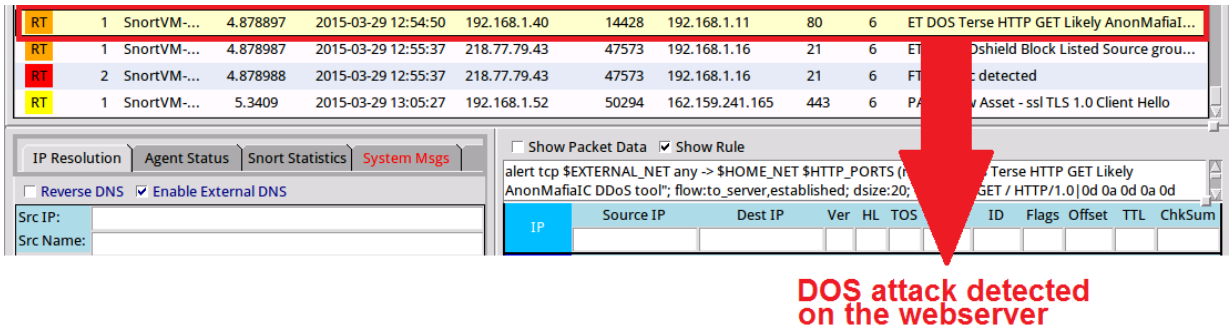

Figure 3.67: Fortunately, this is detected by Snort.

3.4.3.16 Protecting against DOS attacks

It is very difficult to protect a network / computer against a DOS attack. A common way to execute a (D)DOS attack is called a SYN flood or half-open attack. To understand how a SYN flood attack can be prevented, one must first understand the basics of TCP handshaking. A TCP handshake is a method to create a TCP connection over an IP network such as the Internet. This consists of three major parts:

1. First, a SYN (synchronize) packet is sent from host A to host B.
2. Host B returns a SYN-ACK (synchronize-acknowledgement) packet to host A.
3. Host A sents an ACK (acknowledgement) packet to host B.

When host B sents a SYN-ACK packet to host A, but has not yet received the ACK packet from host A, a half-open connection is established which can be abused. Hence the name "half-open attack". Host B has a list in its system memory describing the pending connections. This list is of finite size, so an overflow can be achieved by creating too many half-open connections, causing a memory overflow and disrupting the hosts's normal service. Also, when the memory is full, connections to legitimate hots cannot be established anymore and therefore deny service to them.
Creating such half-open connections can be achieved using spoofed IP addresses. An attackers sents SYN messages to the victim, say host B. They appear to be legitimate (due to the fact that it's IP address has been spoofed), but they reference to a host (the attacker's host) that is unable to respond to the SYN-ACK packet. Therefore, the final ACK packet will never be sent to the victim server system.

A SYN flood attack consists of
Eerst iets over SYN floods en daarna SYN cookies als manier om SYN floods te stoppen.

3.4.3.17 Random stuff

In this section, some Snort activity that occured regardless of the testing purposes is reported.

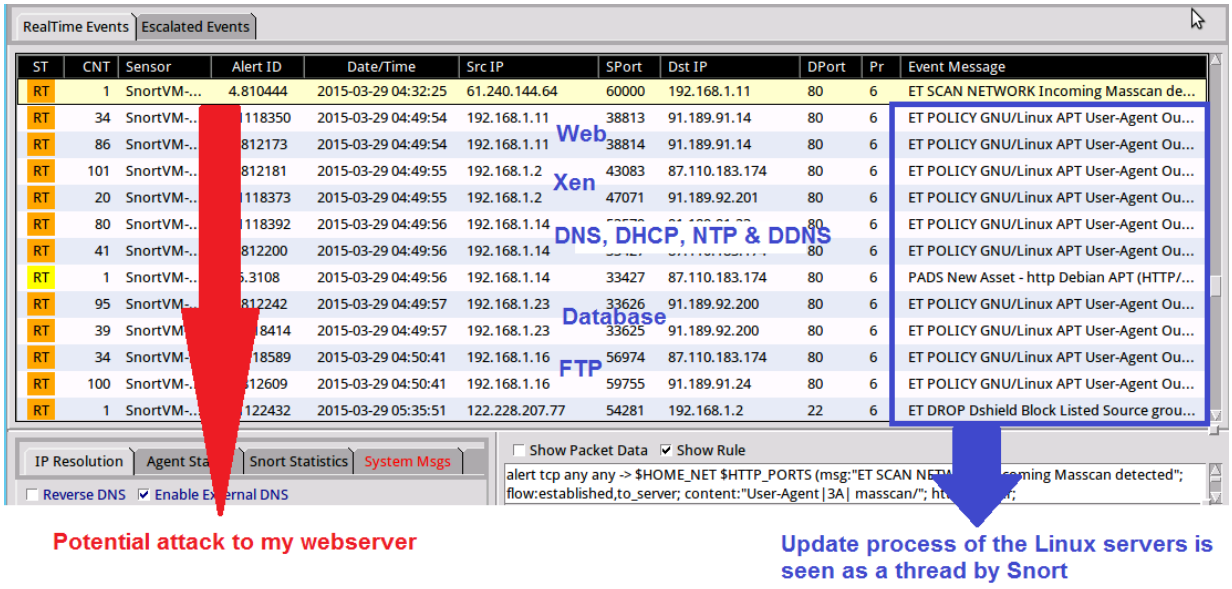

Figure 3.68: The apt updating process is seen as a thread by Snort.

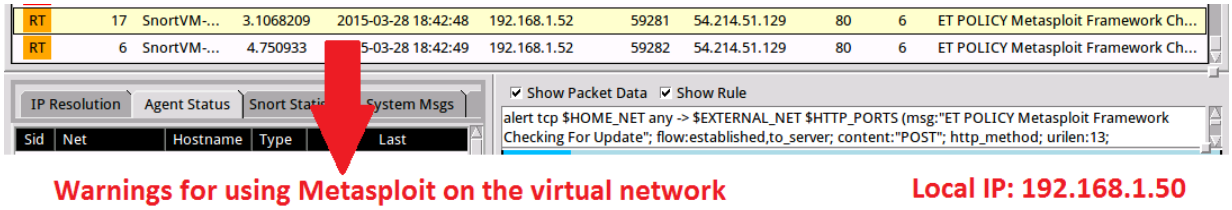

Figure 3.69: Metasploit's updating process is known by Snort...

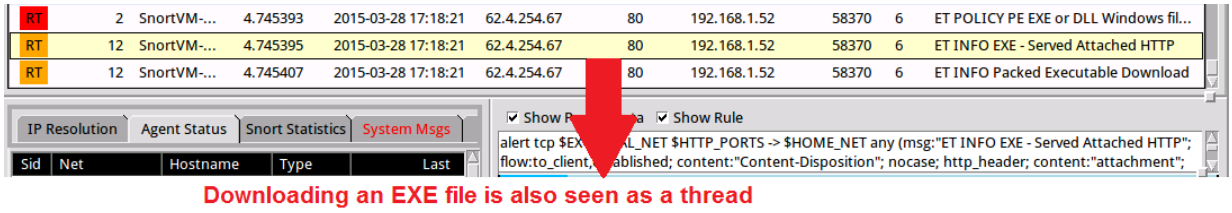

Figure 3.70: Downloading an .exe file from the Internet is also seen and reported by Snort.

Having performed those tests, I have proven that Snort works perfectly on a mixed environment with physical Windows machines, Linux machines, a Xen virtual network and a Hyper-V virtual network.

3.5 Security of dual-boot systems

Nowadays, most 'performance users' make use of a multiple-boot system to, for example, switch between a primary operating system and a secondary, less frequently used operating system.

Multi-boot systems can also be used to test a particular OS, rather then make use of a host-based hypervisor, which does not have direct access to the hardware and is therefore slower.

Finally, in my case, a dual-boot system is used as fail-over mechanism. That is, if one OS is completely infected with virusses or Trojan Horses, or a configuration fault makes the OS unusable, or any other reason that makes the OS unusable, one can always boot in the second, clean OS and resume normal operations.

But what will happen if a cracker hacks such a dual-boot system? Will he be able to access the partition of the other OS as well? If so, will he be able to place a virus or a Trojan Horse on the compromised partition that is activated when the second OS is booted?

3.5.1 Second research question: hacking of dual-boot systems

The above quetions are the motivation for the second research question:
Is it possible to hack a the other OS in a dual-boot configuration, placing a virus or any other type of malware on it and executing this malware at system startup? Is it also possible to mount a .vhdx file and place a virus on it? And if so, ***how can one prevent such malicious operations?***

To answer these questions, a second physical server will be configured as a dual-boot system with a RAID 5 drive array consisting of three drives. Two partitions will be created on this RAID 5 array and each one of them will be formatted with the NTFS file system and house a Windows 7 Professional 64 bit installation on each of them. Thus effectively creating a dual-boot system with two Windows 7 operating systems.
The figure below visualizes the modified network configuration.

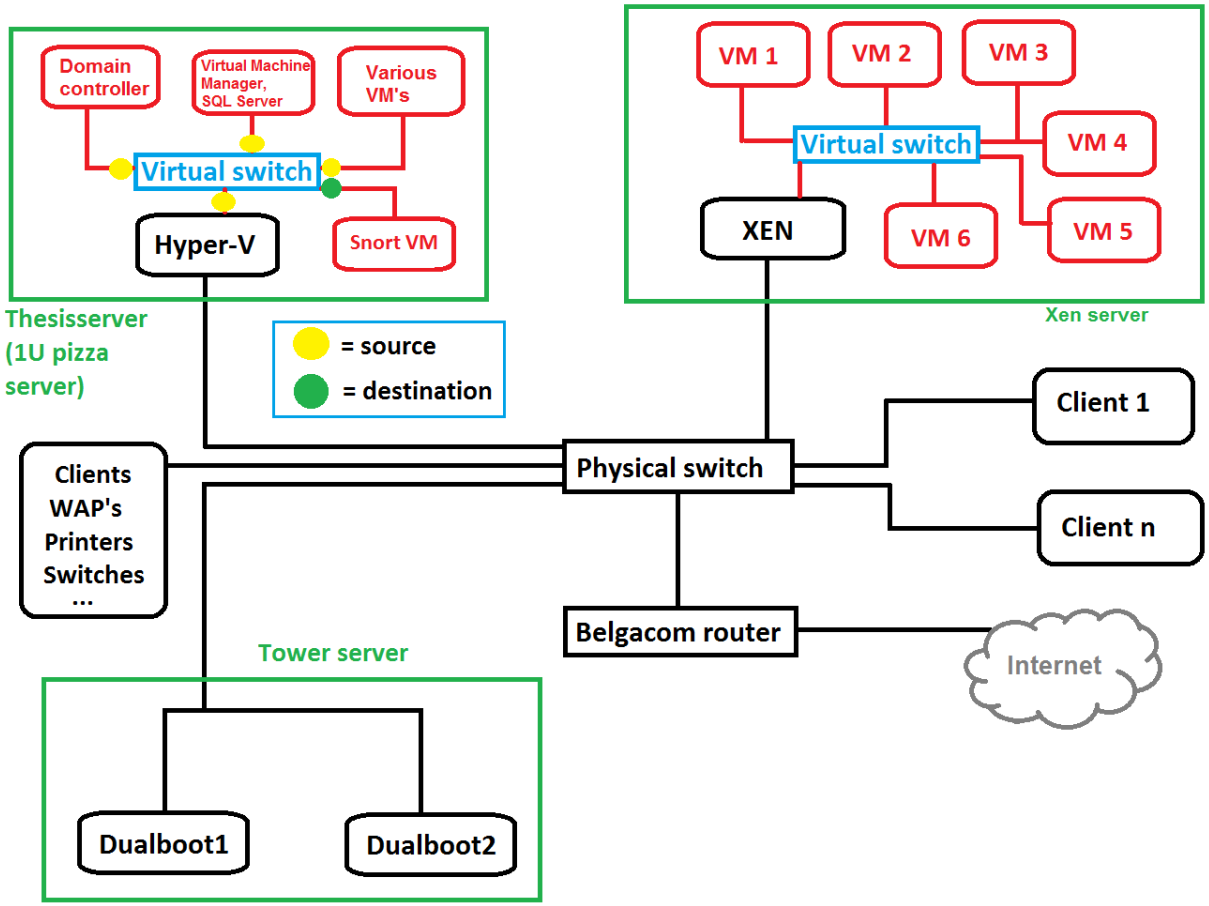

Figure 3.71: Network infrastructure as of the beginning of research question number two

First of all, it appears that one can just mount a virtual hard disk with Windows Explorer, after which the disk partitions (including the "System Reserved" and the normal data partitions) appear in the Explorer window. Access is possible just by browsing the directories. We assume that a hacker has attacked the Hyper-V host and has root access to the system.

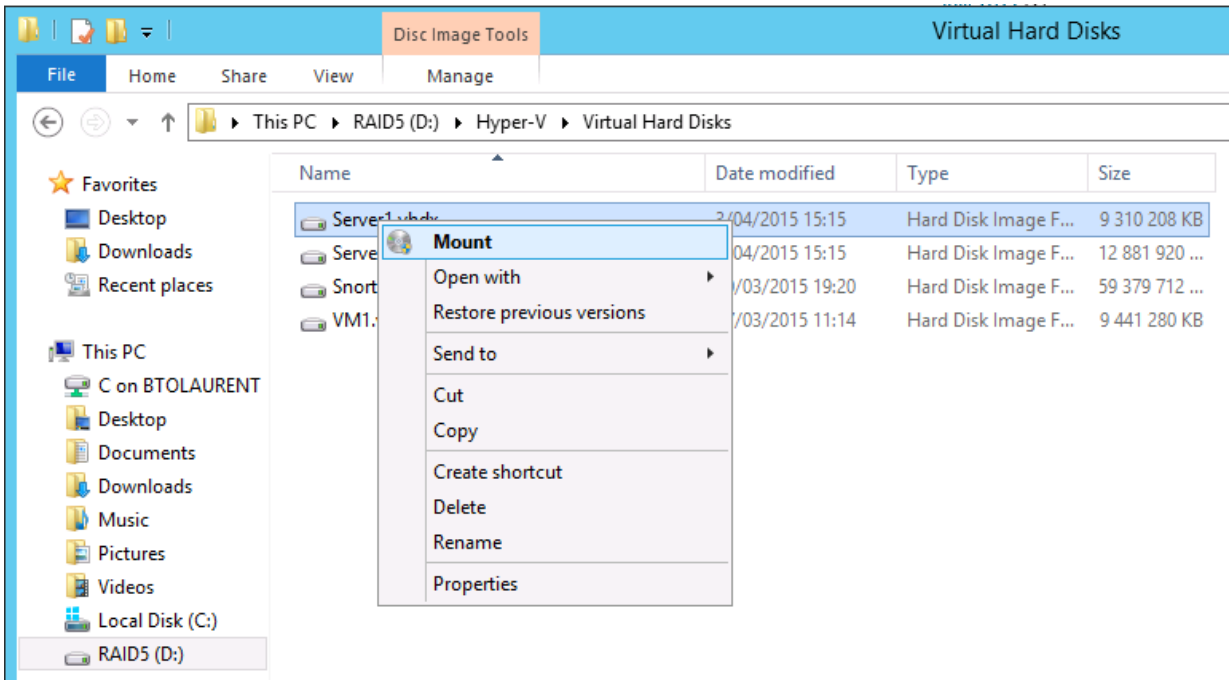

Figure 3.72: Mounting the virtual hard disk in Windows Server 2012 R2 using Windows Explorer ...

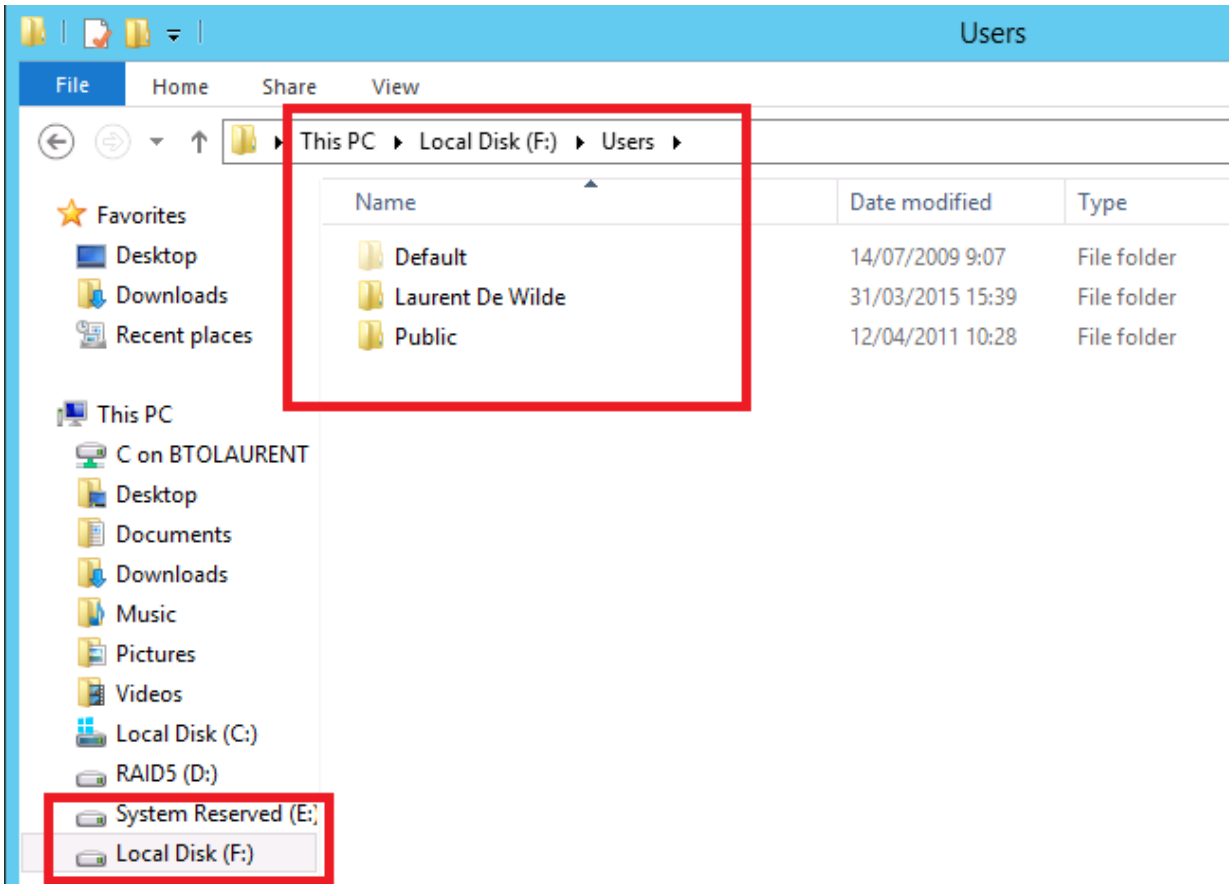

Figure 3.73: ... after which the partitions become visible (browseable).

However, I discovered that once the VM has been started again, no access to the hard drive in Windows Explorer is possible anymore. So I created a Trojan Horse on my computer that I inserted into the hard drive of the compromised VM and that automatically connects to the computer of the attacker (my computer) once the VM has booted without the user being aware of it, after which I can browse files, see the network configuration etc etc ....
This way, permanent access to the compromised VM is possible.

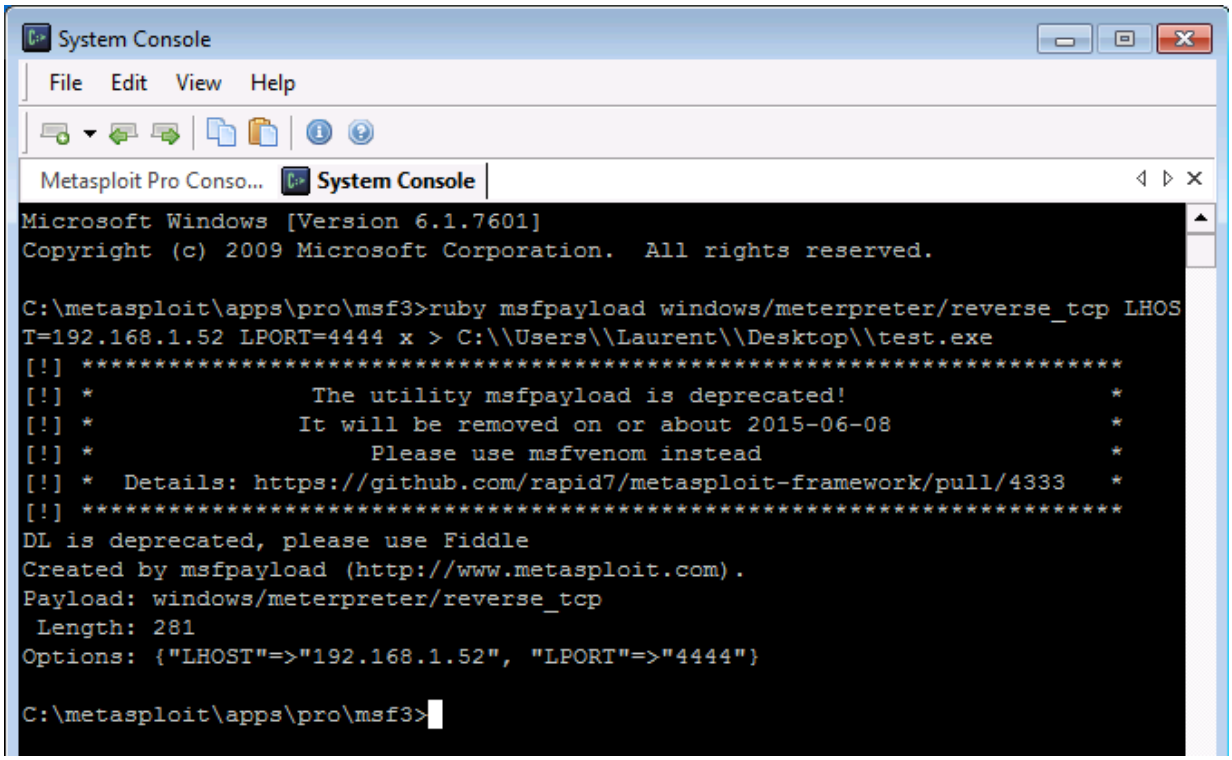

Figure 3.74: Creation of the malicious Trojan.

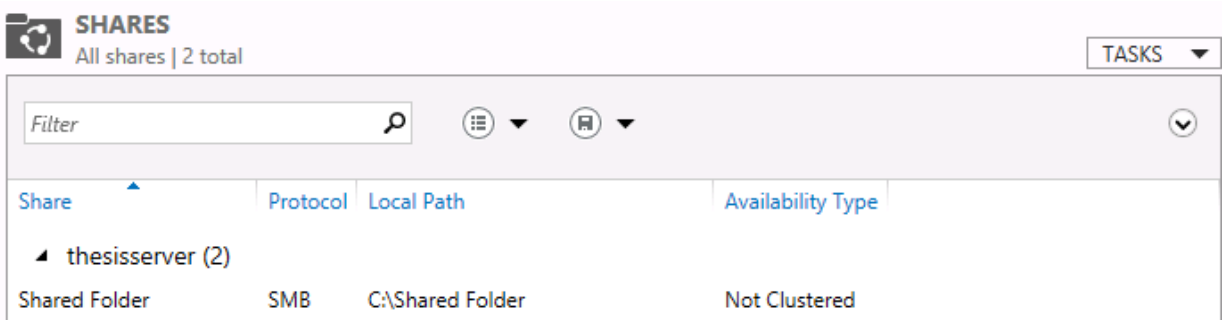

Figure 3.75: Transferred the Trojan to the host by means of a shared folder.

```
meterpreter > shell
Process 1780 created.
Channel 2 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>
```

Figure 3.76: Once the VM has started, it connects automatically to my computer and I can browse the files, even when the hard drive is in use and not mountable anymore in Windows Explorer.

```
Metasploit Pro Conso... | System Console
C:\Windows\system32>cd ..
cd ..

C:\Windows>cd ..
cd ..

C:\>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 709D-5699

 Directory of C:\

14/07/2009  05:20    <DIR>          PerfLogs
12/04/2011  10:28    <DIR>          Program Files
14/07/2009  06:57    <DIR>          Program Files (x86)
31/03/2015  15:39    <DIR>          Users
31/03/2015  15:12    <DIR>          Windows
               0 File(s)              0 bytes
               5 Dir(s)  58.723.999.744 bytes free

C:\>cd Users
cd Users

C:\Users>dir
```

Figure 3.77: An example of the directory listing of the compromised VM.

I showed that it is possible to break into a virtual hard disk and insert some virusses or trojans to infect a virtual machine trough the host.

Place a virus on the other dualboot system

Using “Dualboot2”, I managed to place a Trojan on “Dualboot1”. When “Dualboot1” was started again, the Trojan was started as well and connected to the attacker’s computer

(my computer) as shown in the figures. The two dualboot systems are both Windows 7 Professional x64 editions. This is just an example, any other virus could be used. It is just to show that this is indeed possible.

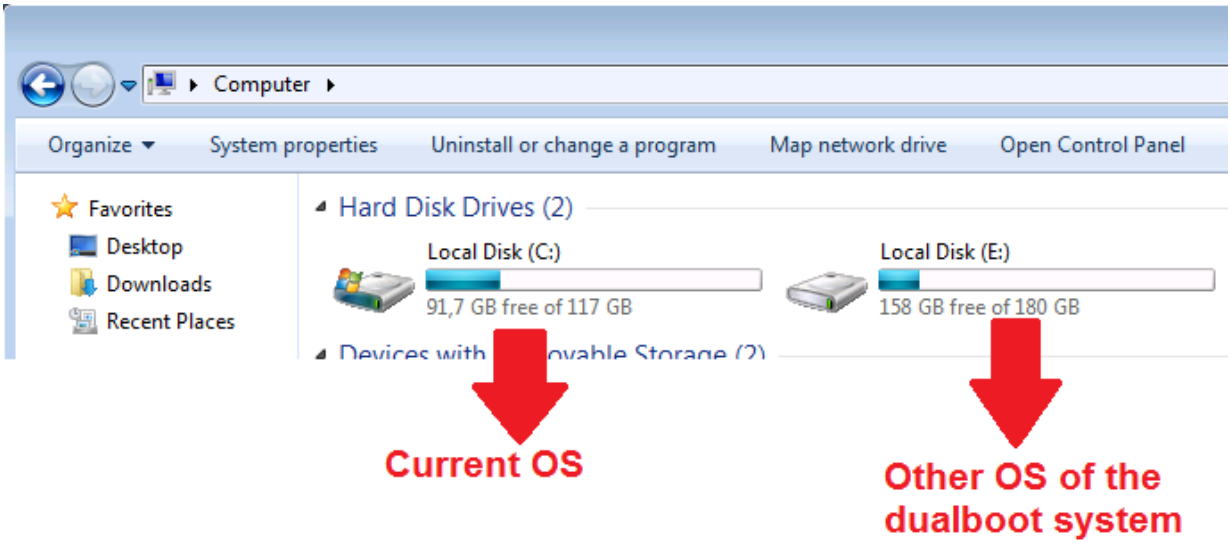

Figure 3.78: The two disks of the two OS's visible in Windows Explorer.

The Trojan is placed in the Startup folder and will be executed when the OS boots.

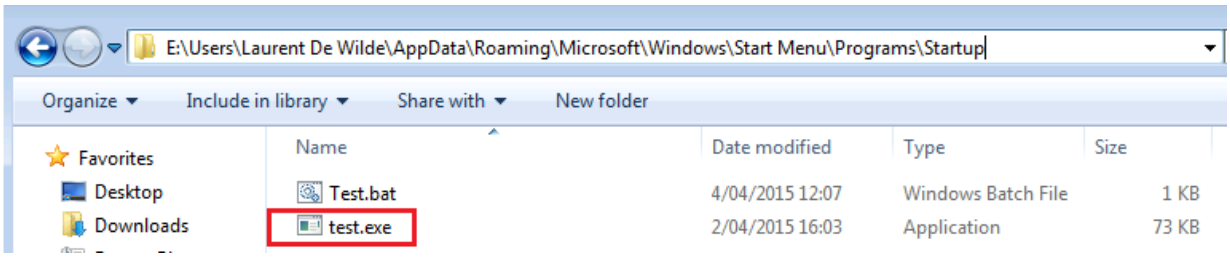

Figure 3.79: The trojan is inserted in the other OS of the dualboot system.

With the Trojan connected to our computer, we can now browse files, etc etc ....

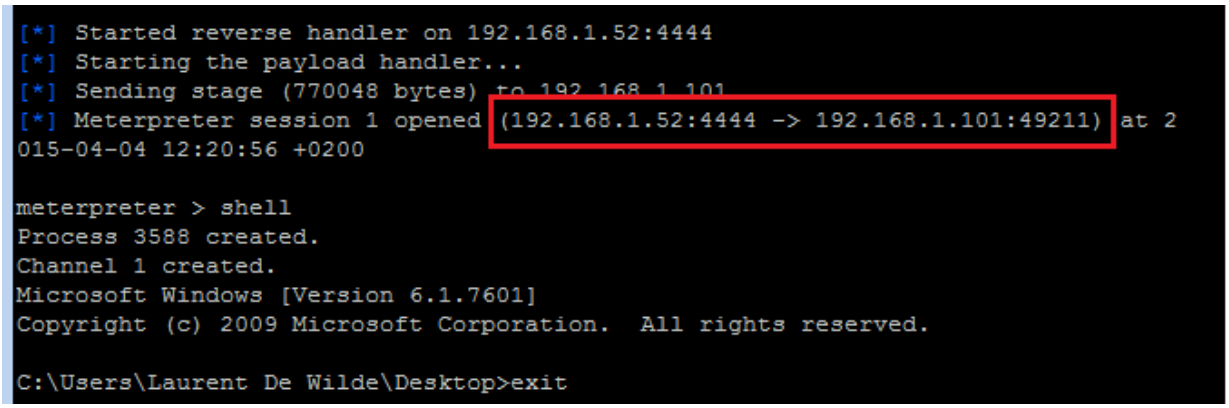

Figure 3.80: The Trojan has connected to our computer.

Not only is it possible to retrieve the directory listings and browse files stored on the hard disk, but one can also perform a DOS on the compromised host. To accomplish this, I make use of a known vulnerability in the Windows 7 operating system, called MS12-020. This means that this vulnerability allows a cracker to execute remote code.

In this example, a DOS will be executed. This is how it works: if the way the `ConnectMCSPDU` packet is handled is in the `maxChannelIDs` field, is altered, an invalid pointer will be used and thus resulting in a DOS condition.

To insert this remote malicious code, I make use of Metasploit.

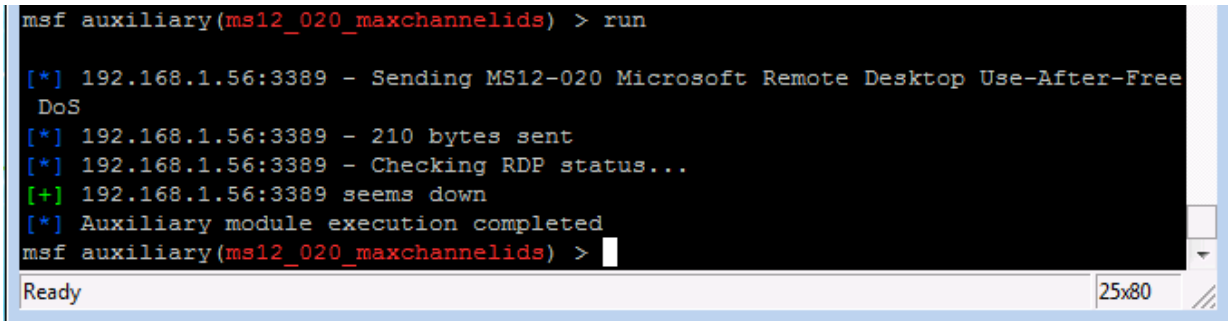

Figure 3.81: The parameters are set up correctly and the module is runned...

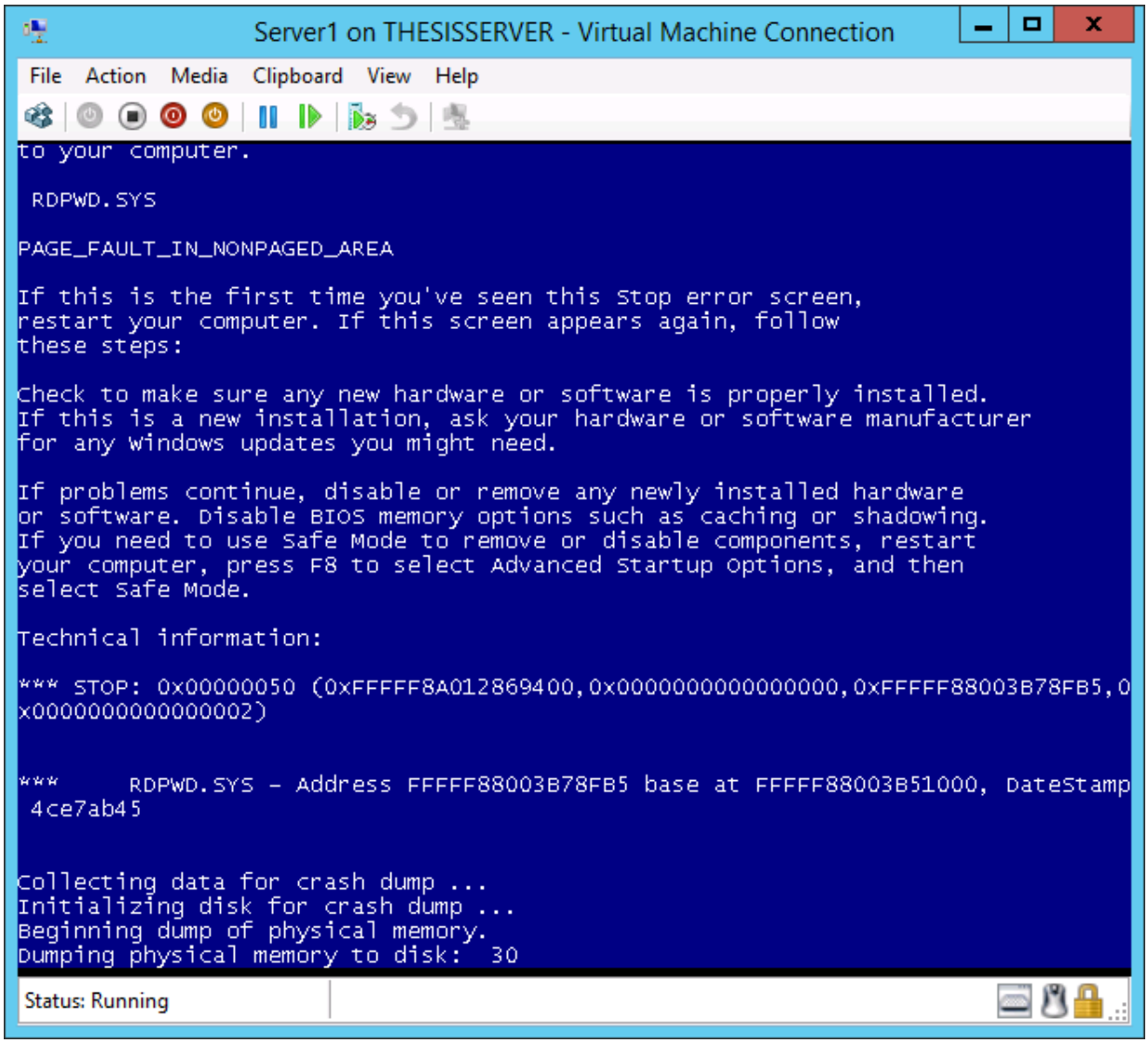

Figure 3.82: ... resulting in a BSOD.

Solutions to prevent this include

- Disable Remote Desktop Connection if not needed.
- Change the default port value of 3389.
- Use Network Level Authentication (NLA).
- Apply the latest patches.

3.5.2 Preventing access to a (virtual) hard disk

First of all, when a virtual hard disk is in use, that is, when the VM is already started, it is impossible to mount it. Thus the disk gets locked automatically when the VM is started. However, by default, when the VM is not started, it is possible to mount the disk and steal the files on it, as I showed in the previous SITREP.

The solution to this is (in my opinion), not to lock but to encrypt the disk using BitLocker. This tool is buillt-in by default in WS2012 R2 (but not enabled by default of course).
BitLocker encrypts the hard drive and makes it password-protected so that when an attacker wants to retrieve the data from the disk, (wants to mount it), he first needs to type in a password. When AES 256 bit encryption is used, the process of cracking the password becomes extremely difficult.
Also, when the disk is transfered to another Hyper-V server, it is still impossible to mount the disk without knowing the password. Obviously, another password than the Administrator password must be used - since the hacker already knows the (compromised) password of the system.

For this purpose, I will demonstrate this using a Windows Server 2012 R2 VM, since BitLocker is only available on the Enterprise or Ultimate versions of Windows 7 and I only have a Professional version of Windows 7 available.

First, I have enabled the BitLocker feature and configured it to use 256 bit encryption using group policy.

NOG EXTRA UITLEG OVER BITLOCKER NEEDED! : WHAT IS IT? POSSIBILITIES ETC ETC.

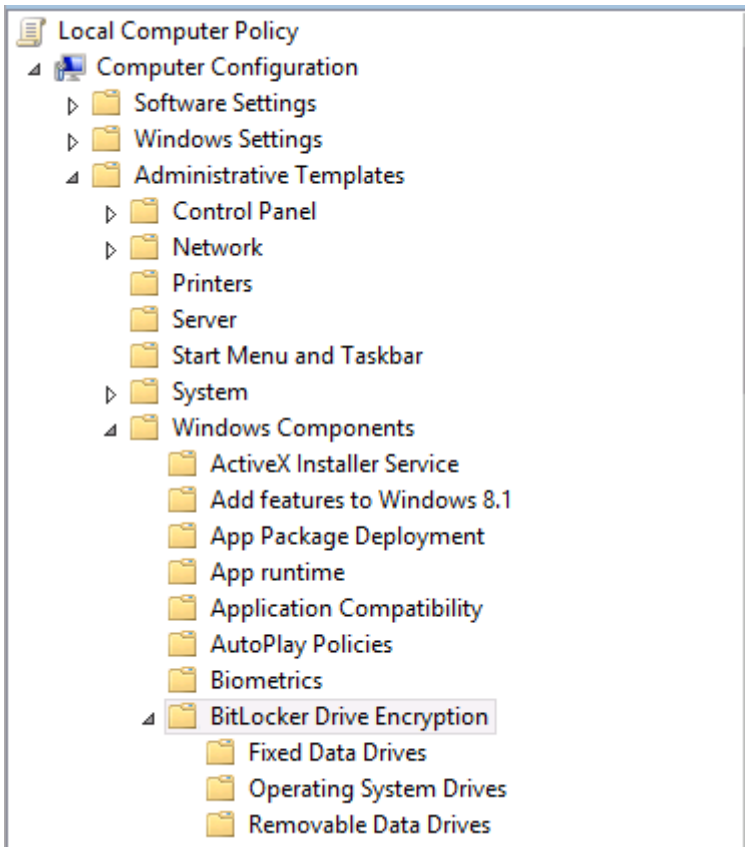

Figure 3.83: The BitLocker settings in the group policy of Windows Server 2012 R2.

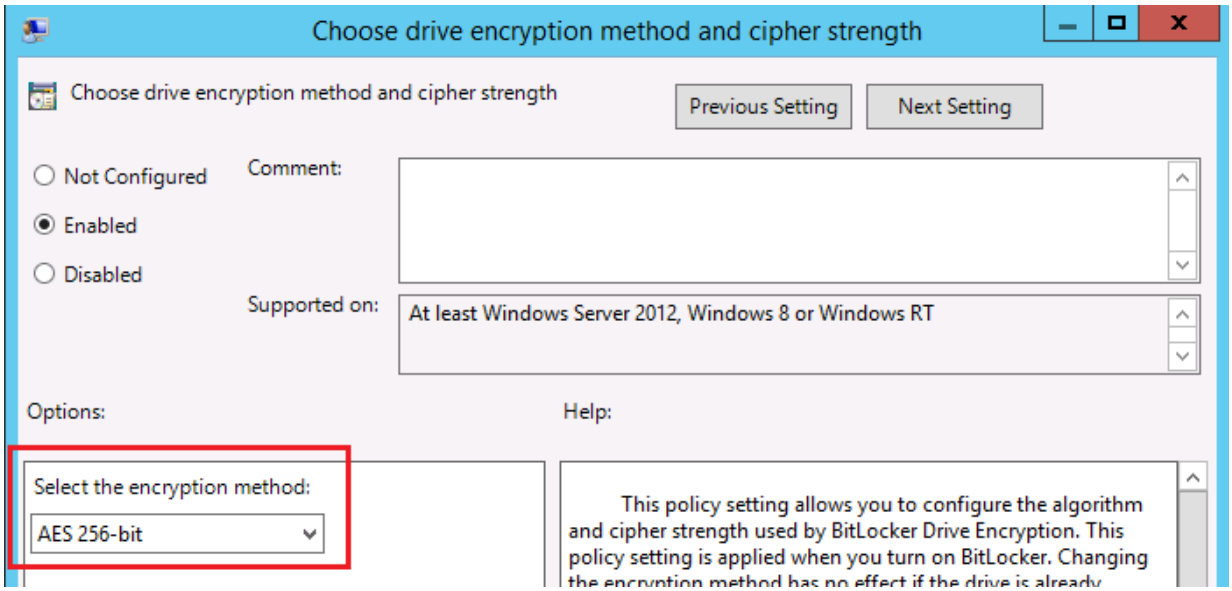

Figure 3.84: A 256 bit encryption is chosen for maximum protection.

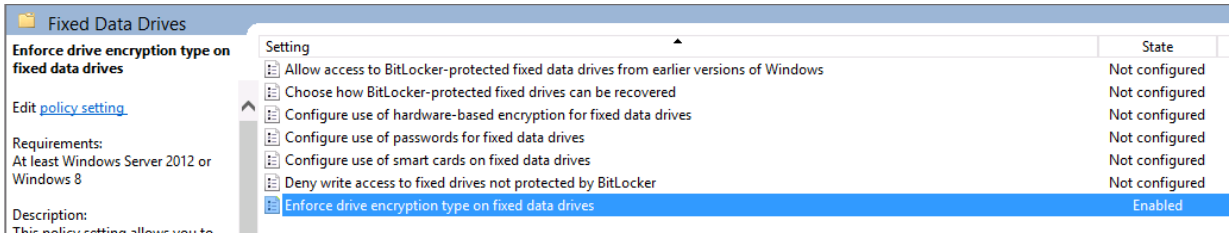

Figure 3.85: Also non-system drives can be secured.

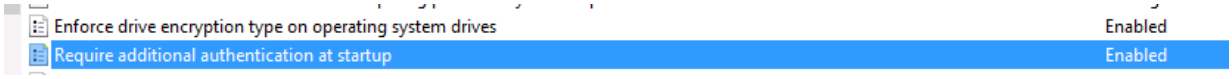

Figure 3.86: This setting forces the requirement of entering a password when the VM is started.

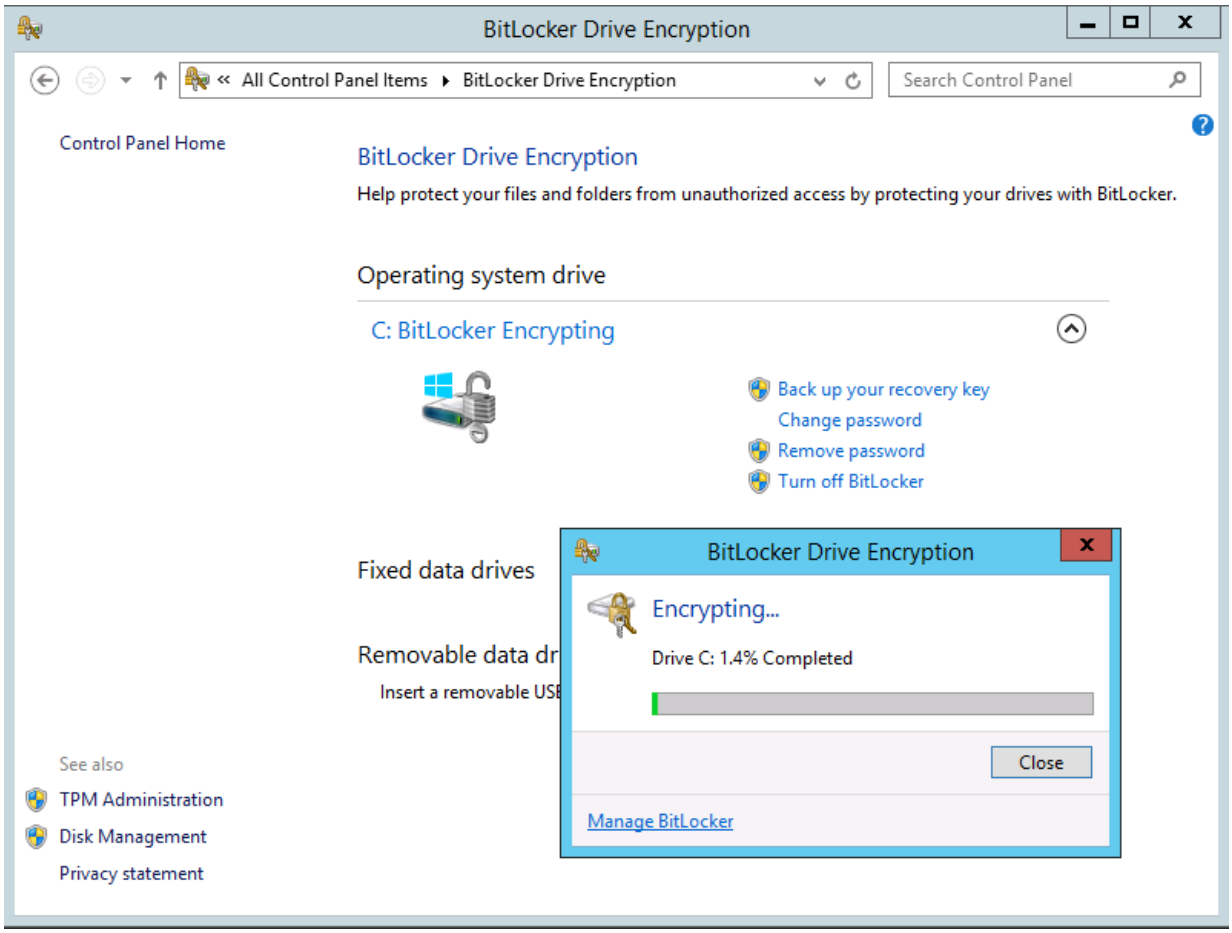

Figure 3.87: When the settings are set, the VM is rebooted and "BitLocker drive encryption" is selected in the Configuration Panel. Since we want to encrypt the entire VM (and therefore preventing the disk from mounting), the "C:" drive is selected.

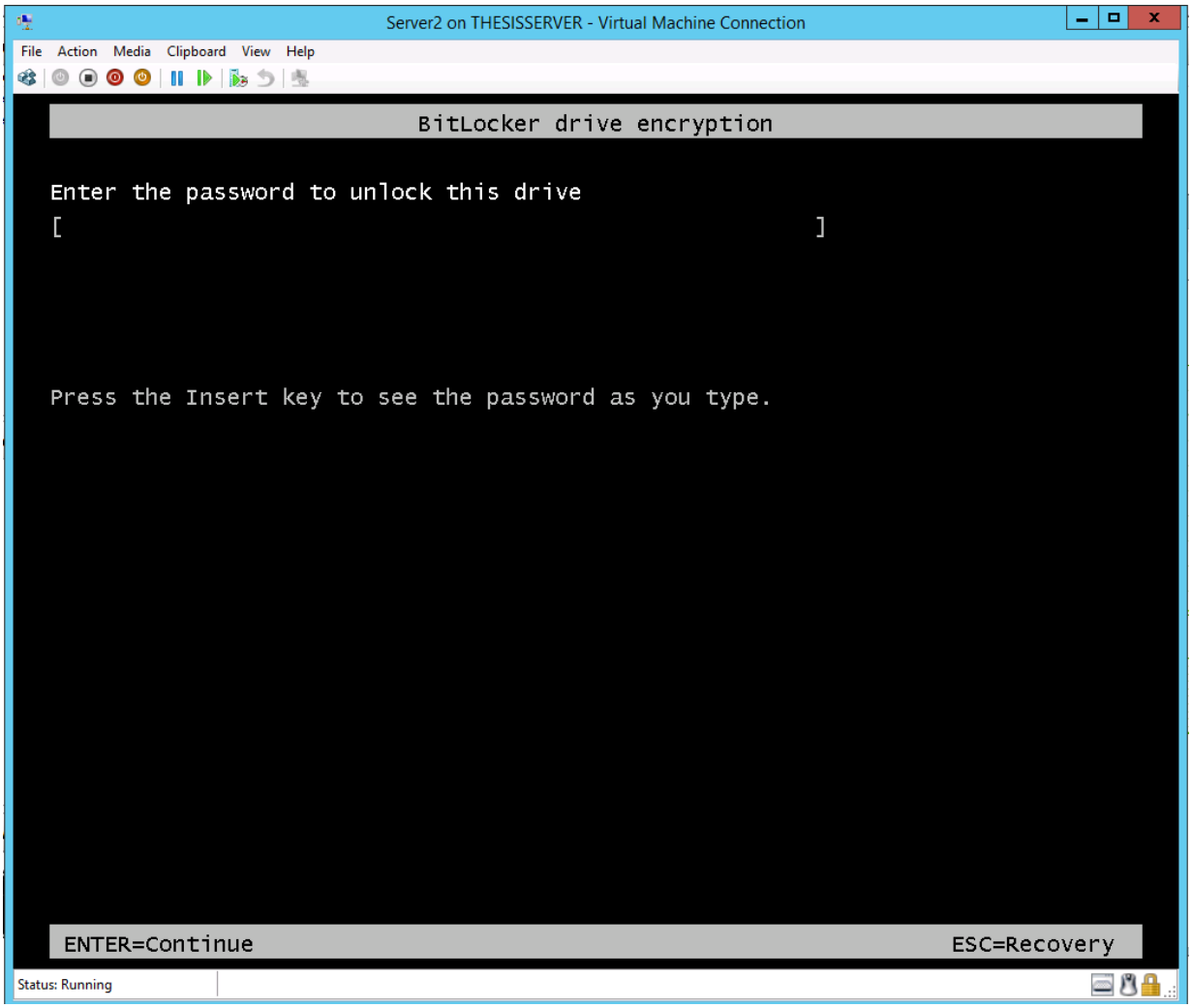

Figure 3.88: When the encryption of the system drive has finished, the VM is rebooted and a password is prompted when one wants to boot the VM.

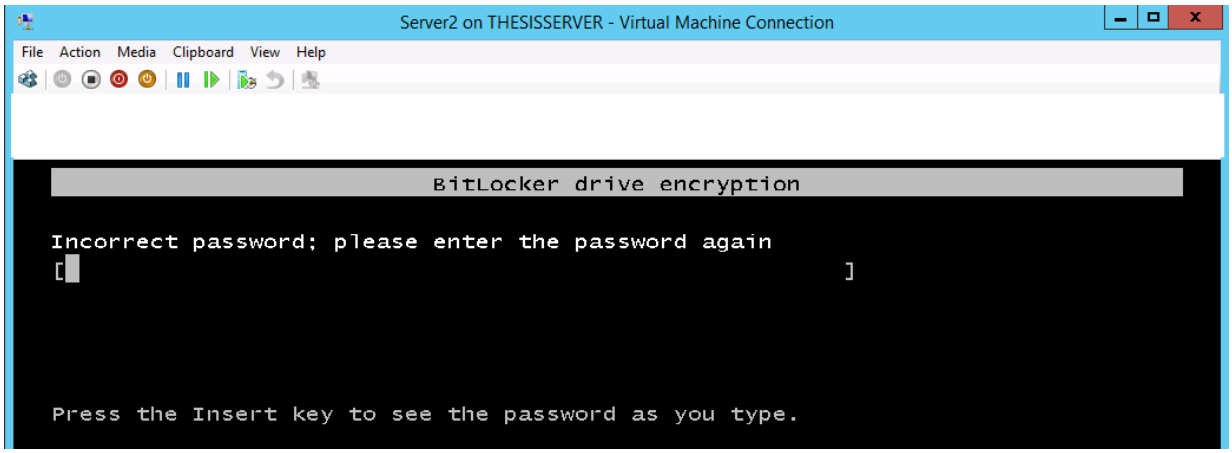

Figure 3.89: Without knowing the password, it is impossible to login. . . . Of course, this prompting for a password at boot time can be disabled in the Group Policy Editor.

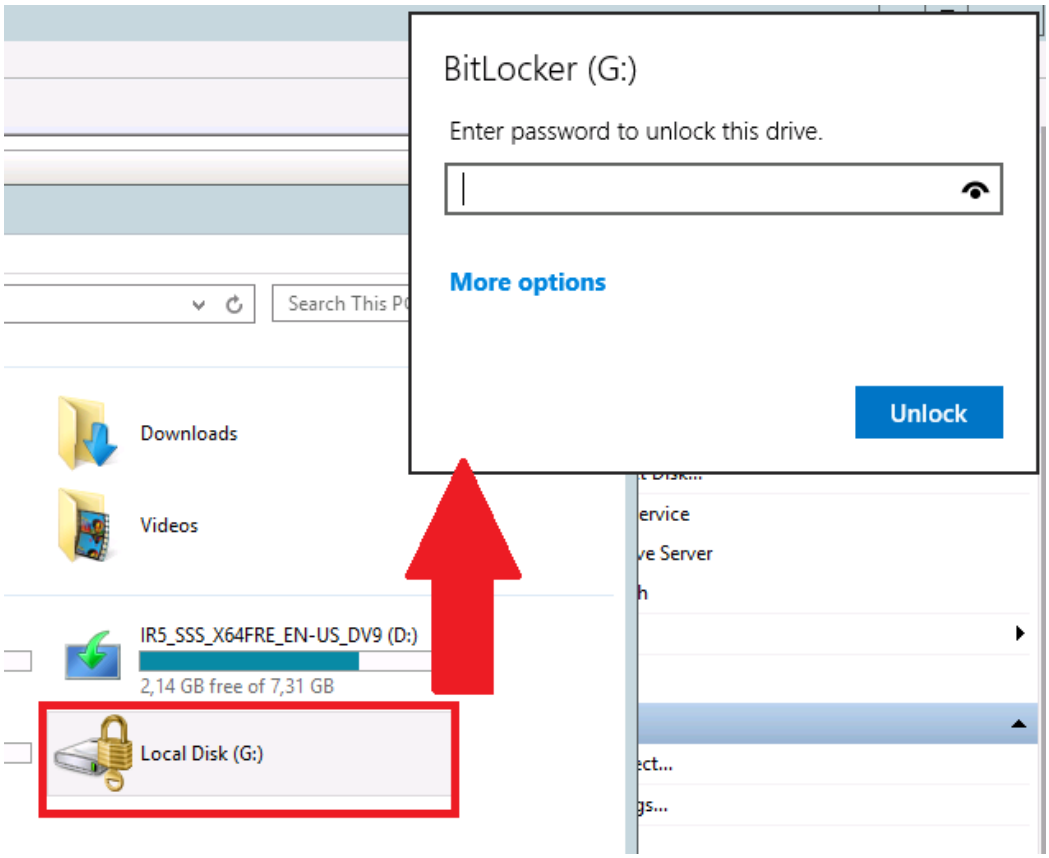

Figure 3.90: Now we try to mount the virtual hard drive, but a password is required to do so.

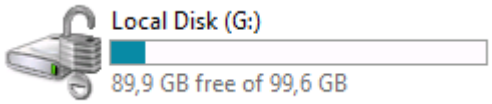

Figure 3.91: Only when the correct password is supplied, one is able to access the files on the virtual disk.

Not only does the encryption with BitLocker ensures full disk encryption of local disks and virtual disks (.vhdx files), but when transferring such virtual drive to another computer, one is able to use (i.e.: unlock) the drive on the destination computer. Thus providing an extra layer of security when transferring the drive over the network as a possible cracker that is eavesdropping the network connection will not be able to read the content of it.

3.5.3 Downsides of using BitLocker

In the previous section, an effective way of protecting a hard drive from unwanted access has been shown. However, with an added layer of security (i.e.: encrypion), performance may be affected in a negative way. After all, when the data is written to or read from the disk, an AES encryption algorithm must be applied and thus slowing down the whole read and write process.
To prove so, some benchmarking has been performed. NOG UITLEGGEN WHAT CRYSTAL DISK MARK IS!!

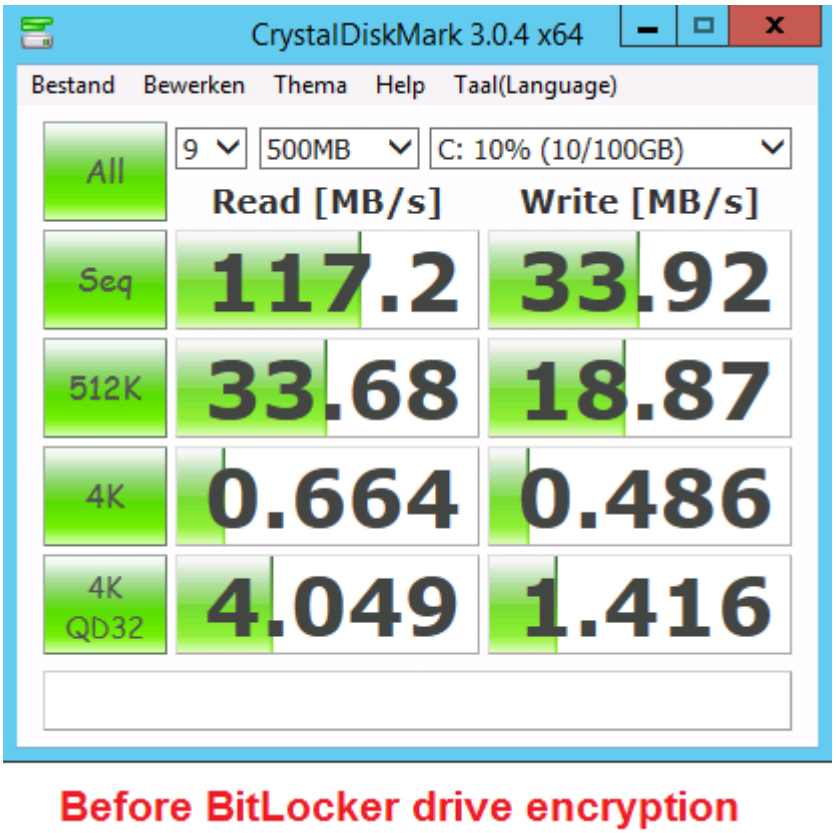

Figure 3.92: Read - and write speeds according to "CrystalDiskMark" before the drive has been encrypted by BitLocker. Note that I use an actual system (virtual) disk.

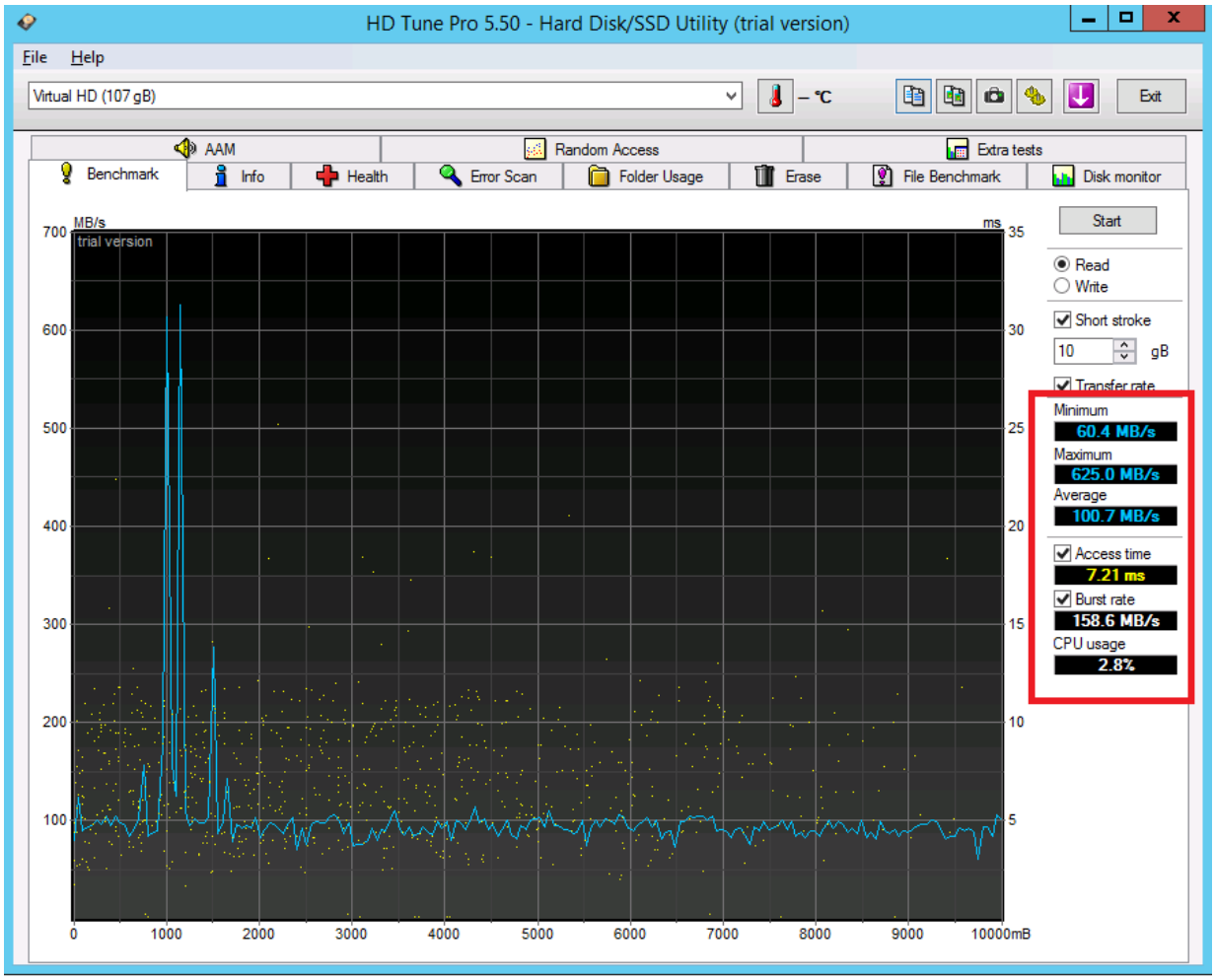

Figure 3.93: Here I use HDTune Pro to perform the benchmarking.

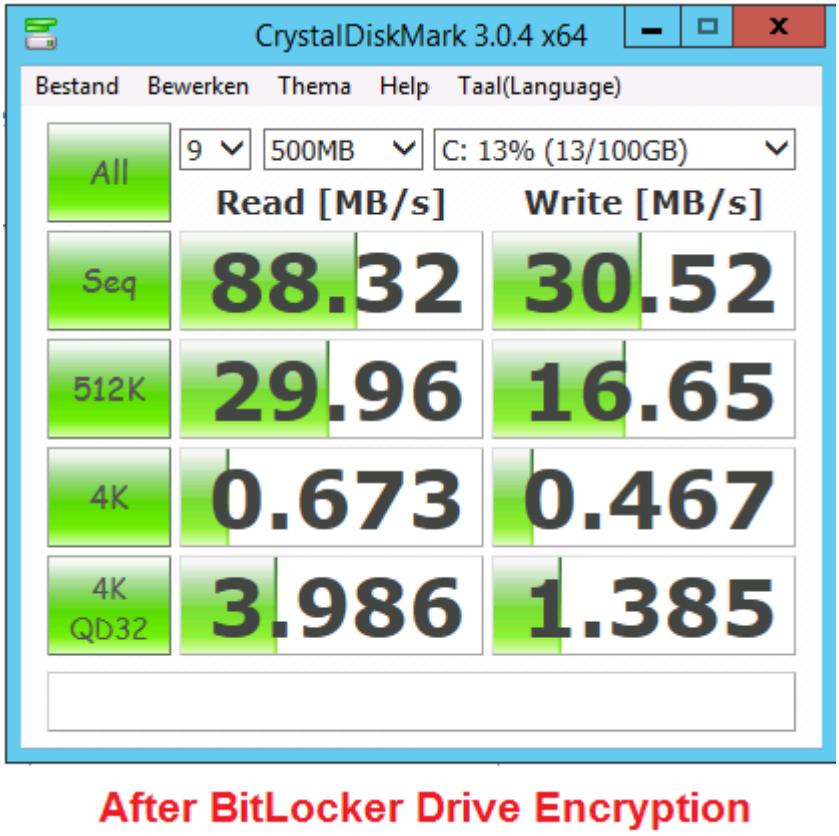

Figure 3.94: These are the results after the system drive has been encrypted using BitLocker. The sequential read speed drops from 117 MB/s to 88 MB/s and the sequential write speed drops from 34 MB/s to 30 MB/s.

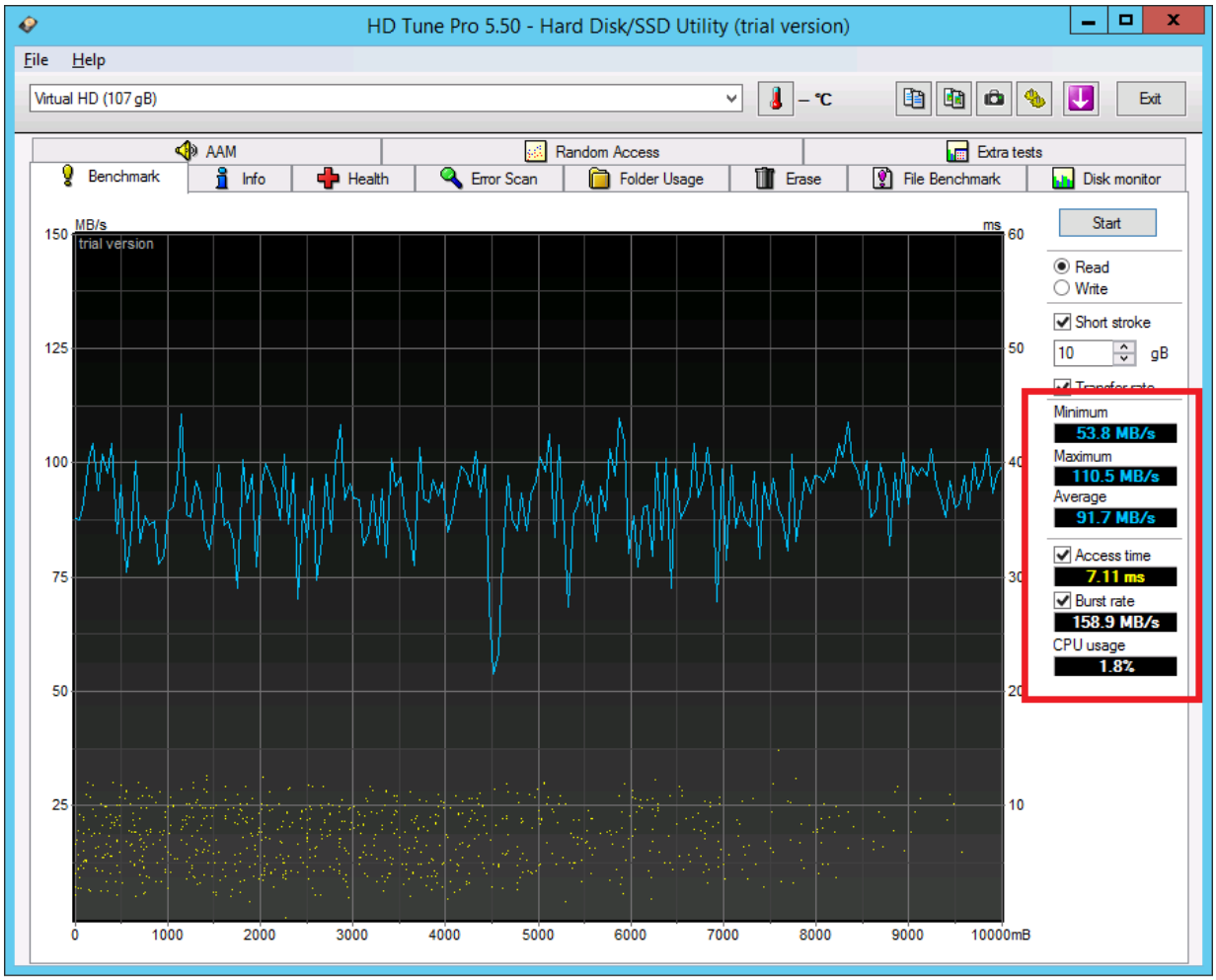

Figure 3.95: Also HDTune confirms the speed drop: from 100 MB/s on average to 91 MB/s on average.

Let us summarize the performance differences:

	Before	After	Difference in %
Read 1	117	88	24 % slower
Read 2	100	92	8 % slower
Write	34	30	12 % slower

So in conclusion, if we take the average of the two reading speed differences, we can state that the reading speed is 16% slower and the writing speed is 12% slower.

Note that the disk access time **remains the same** according to HDTune. The question now is: do we choose for performance but less security, or do we choose for security with a performance decrease of approximately 15%?

Chapter 4

Installation of a private cloud

In addition to network attacks against stand-alone network computers, some attacks against private clouds will be performed. In this chapter, the installation of a cloud environment using Windows Server 2012 R2 will be covered in detail.

4.1 Installation of Microsoft System Center 2012 R2 Virtual Machine Manager

4.1.1 Installation requirements

Before installing System Center 2012 R2 VMM, some hardware - and software requirements must be met.

Processor	Dual-core of 2,8 GHz
RAM	4 GB
Hard disk space without a local VMM database	40 GB
Hard disk space with a local VMM database	150 GB

Hardware requirements (up to managing 150 hosts)
When managing more than 150 hosts, it is recommended to use a dedicated computer for MSSQL Server. That is, to store the VMM database on a dedicated computer.

Software requirements

- Microsoft .NET Framework 4.5 or higher

- Windows Deployment and installation kit for Windows 8.1
- Windows Server 2012 R2
- Microsoft SQL Server 2012 (with or without SP1 and SP2)

Other requirements

- The server where the VMM will be installed, must be a member of an Active Directory domain.
- The server name cannot exceed 15 characters

4.1.2 Generatl installation overview

...

4.1.3 Pre-installation configuration

Prior to installing VMM, some settings will be configured such as a static IP address and joining the server to an AD domain. The complete installation has been performed on a freshly installed Windows Server 2012 R2 as VM on a Windows Server 2012 R2 running Hyper-V. Make sure you are logged in as a Domain Admin.

In order to be able to join the server to a domain, a static IP must be set. Make sure the DNS servers point to the DNS servers of the AD domain.

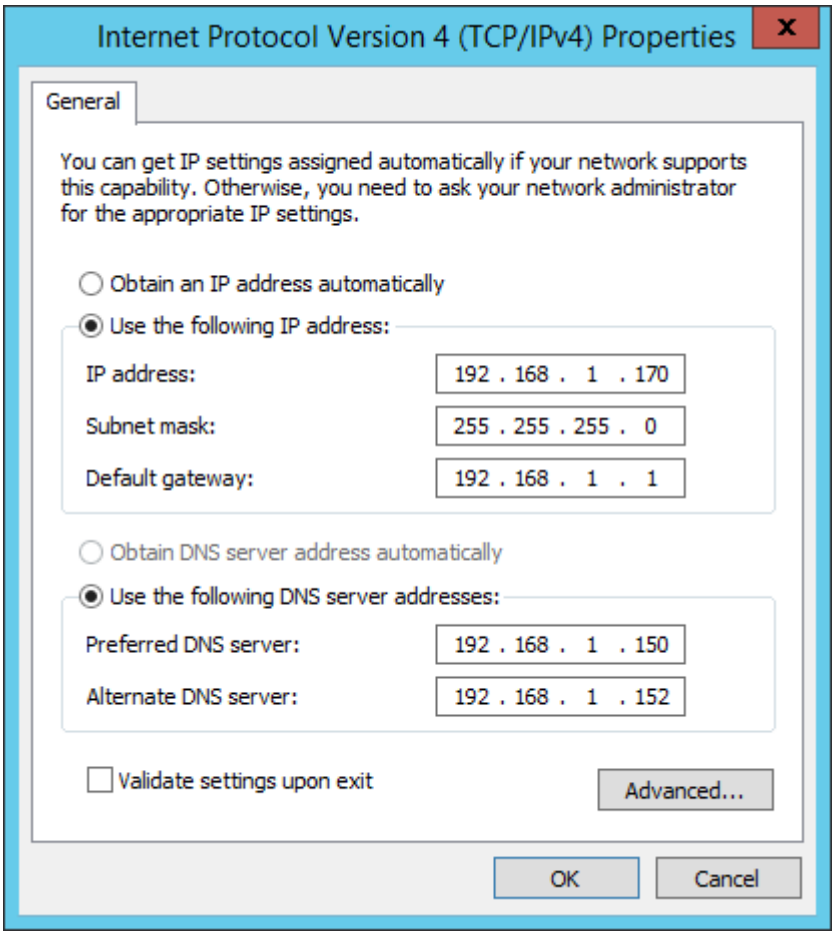

Figure 4.1: configuring static IP settings.

After this, the server can be joined to the AD domain.

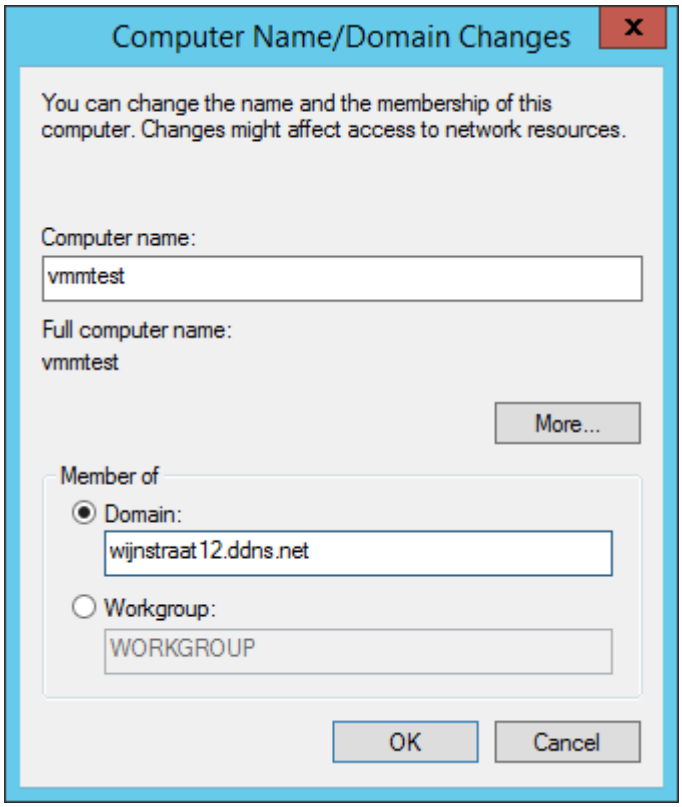

Figure 4.2: Setting an appropriate server name and joining the server to the AD domain.

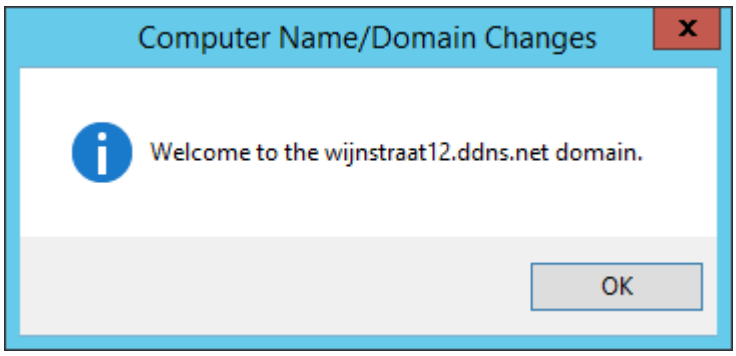

Figure 4.3: Confirmation of successfully joining the AD domain.

Next, the server has to be rebooted and the overview screen of the local server must be something as the figure below.

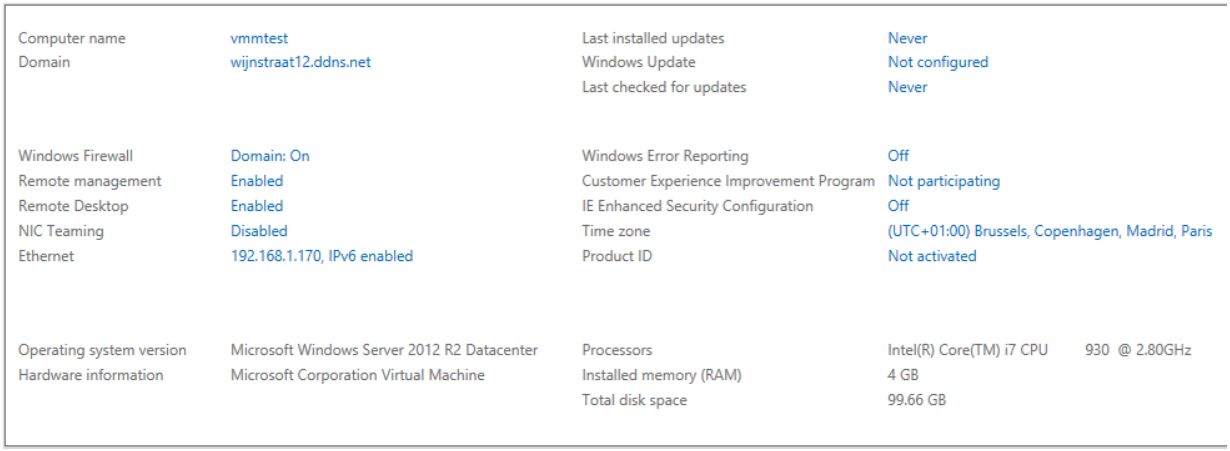

Figure 4.4: The overview screen of the local server with a proper server name set, the server being joined to the domain, the firewall and remote desktop both being enabled. Also, a static IP has been set and the IE Enhanced Security Configuration has been disabled. Now we are ready to install Microsoft SQL Server 2012 SP2.

4.1.4 Installation of MSSQL Server 2012 SP2

In this tutorial, SQL Server will be installed on the same server as the Virtual Machine Manager will be installed on. Before installing SQL Server 2012, the `netfx3` package will have to be installed, otherwise the installation process will fail. This is achieve by executing following command in PowerShell:

```
dism /online /enable-feature /featurename:netfx3 /all /source:d:\sources\sxs
```

Before executing this command, make sure that the installation media (that is, the .ISO image of WS2012R2) is inserted.

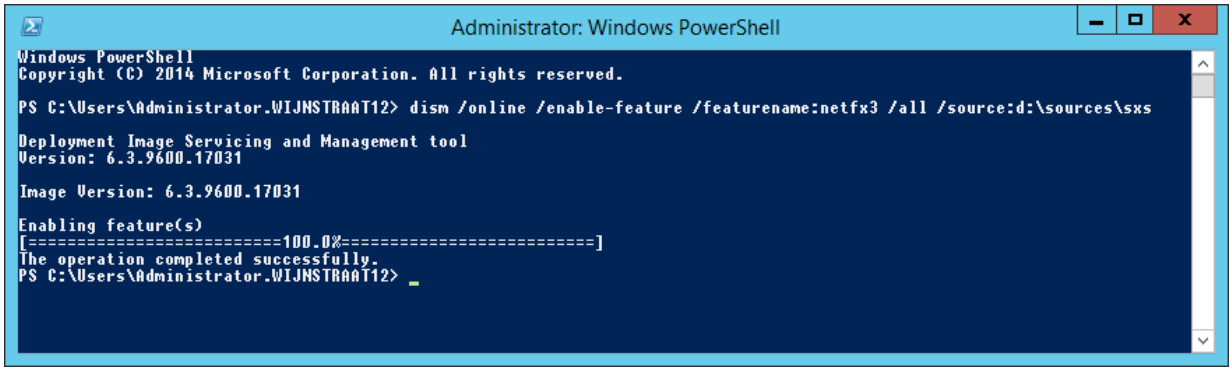

Figure 4.5: `netfx3` has been installed successfully.

Now, the actual installation of SQL Server can begin. When mounting the SQL Server .ISO image and trying to run setup.exe, I expected some error messages. However, when first extracting the .ISO file with WinRAR and then executing setup.exe, everything went fine.

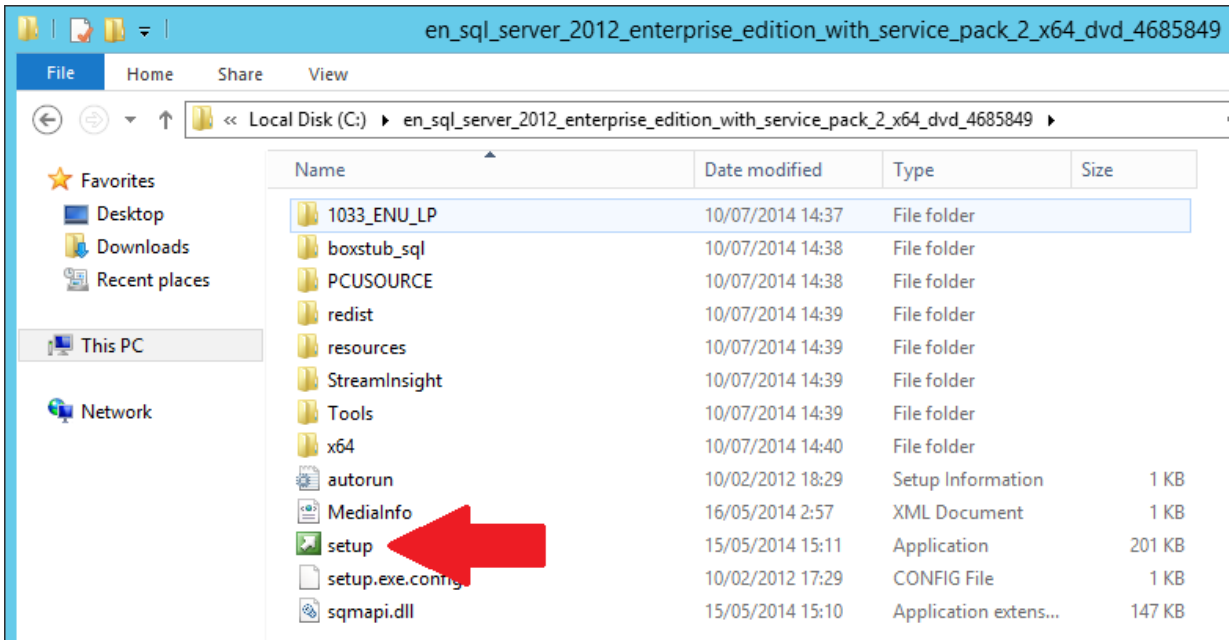

Figure 4.6: Extract the image using WinRAR and run setup.exe.

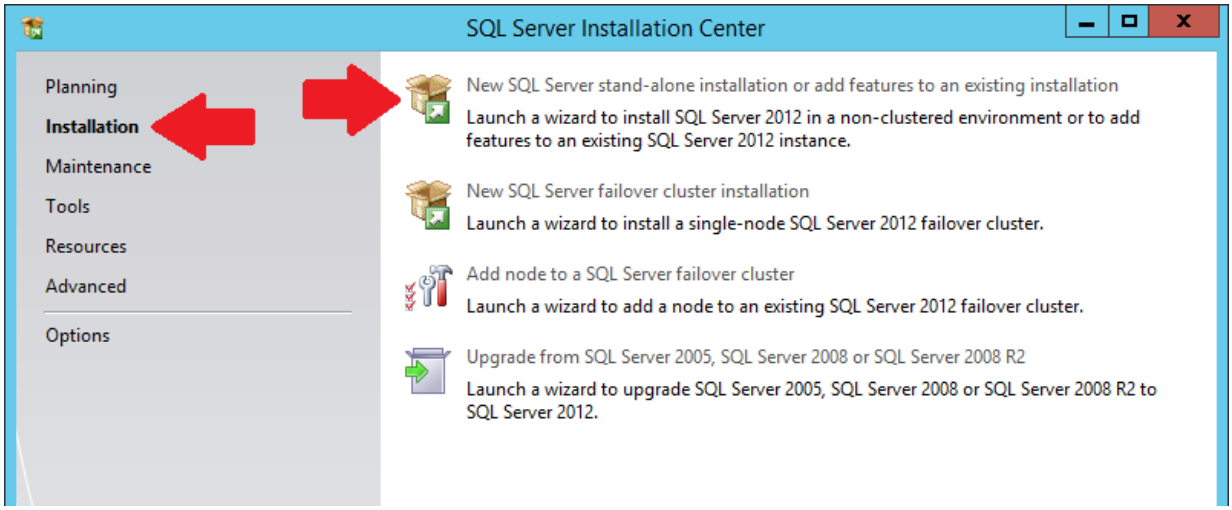

Figure 4.7: In the left pane, select "Installation" and subsequently in the right pane, select "New SQL Server stand-alone configuration".

The installation will begin and the prerequisites will be checked. When every prerequisite is fulfilled, all marks will be green. Next, enter the product key or choose the evaluation version. After this, the license terms will have to be accepted and the installation will check for product updates.

When everything has been passed successfully, one should be seeing the following figure.

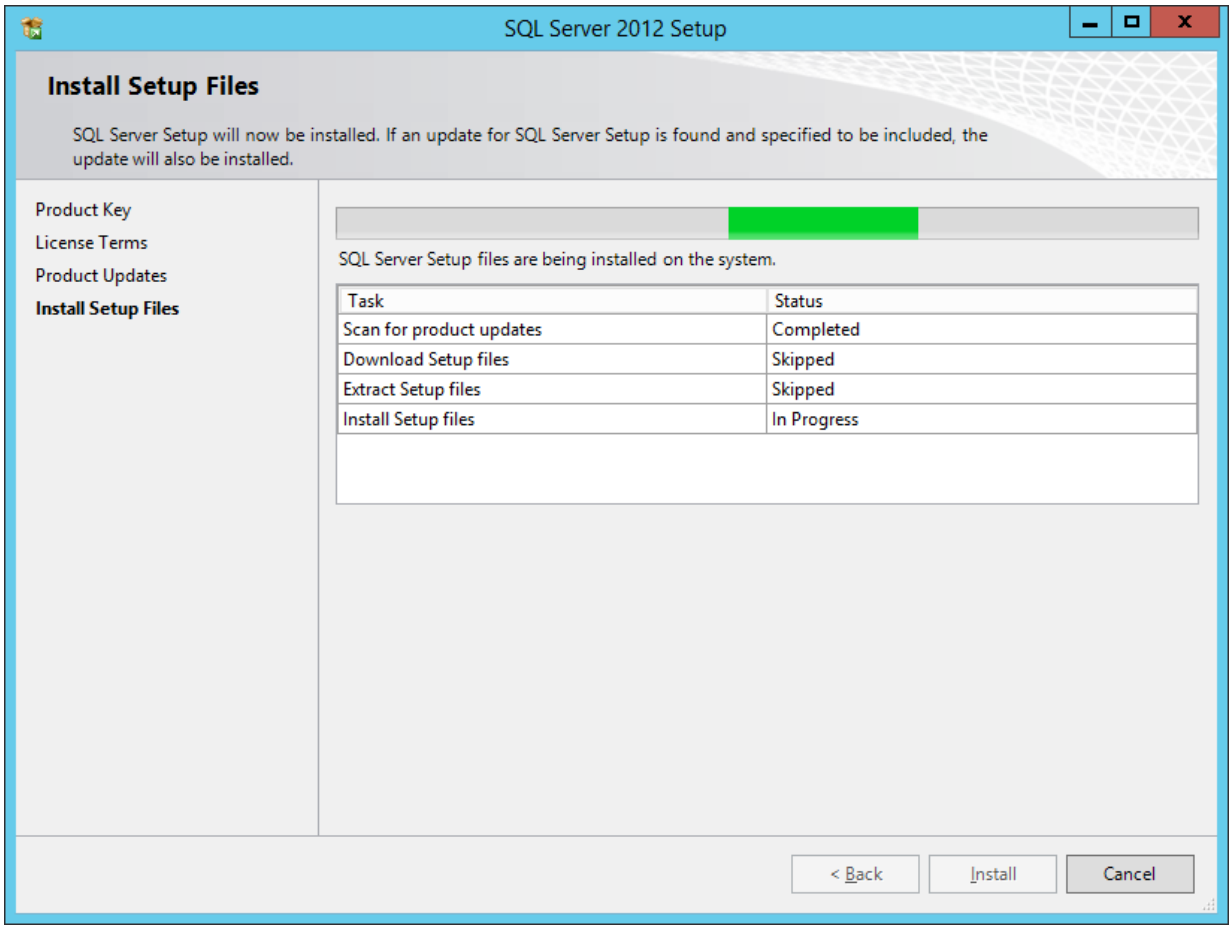

Figure 4.8: Setup files are being installed.

Some additional prerequisites are checked. When everything is passed, setup can be continued.

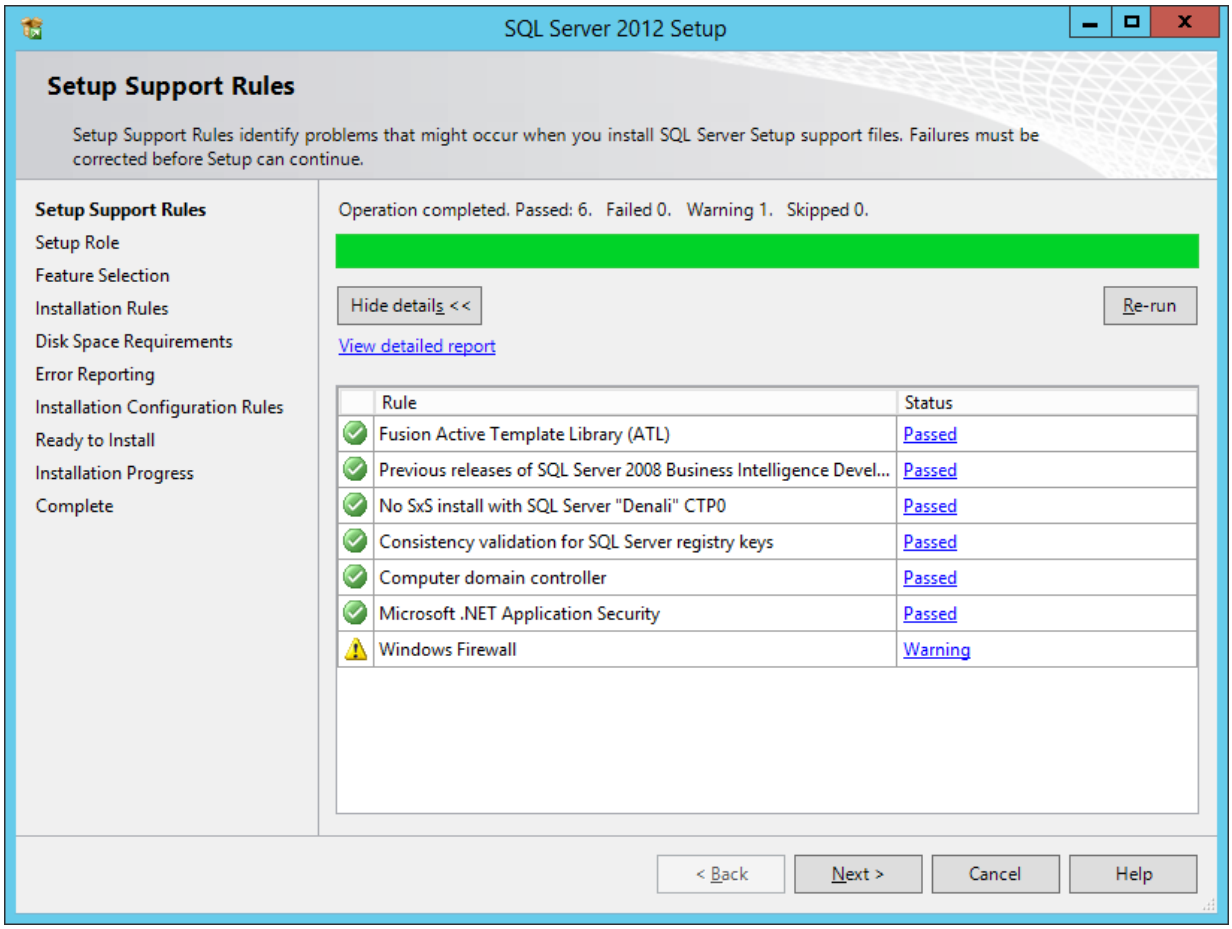

Figure 4.9: Checking additional prerequisites.

On the feature selecting screen, make sure the following features are checked:

- Database Engine Services
- Management Tools - Basic
- Management Tools - Advanced
- SQL Client connectivity SDK

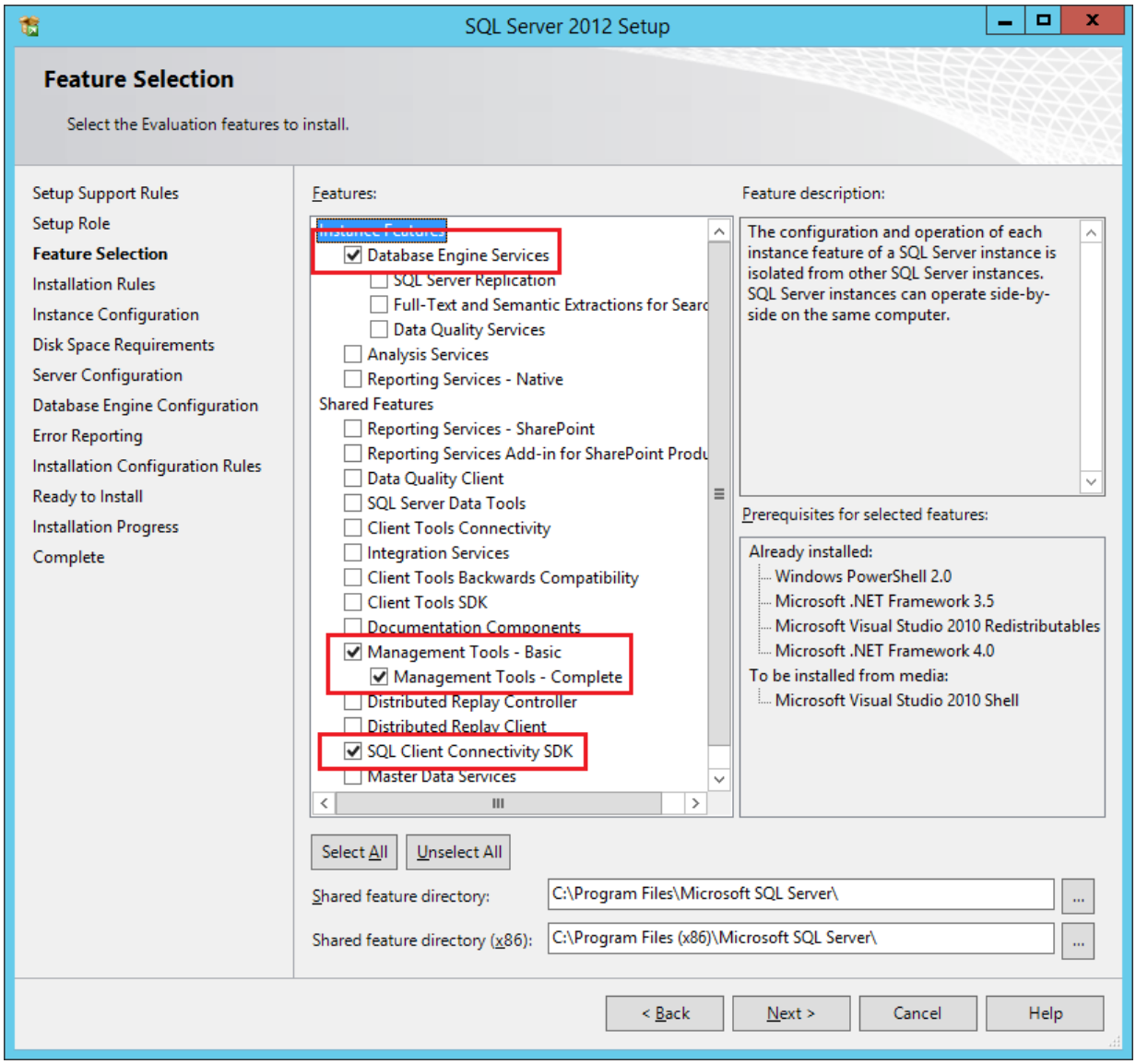

Figure 4.10: Selection of the features.

On the next screen, choose "Windows Authentication Mode" and specify the administrators. In this case, both Domain Admins as well as Local Admins have been chosen.

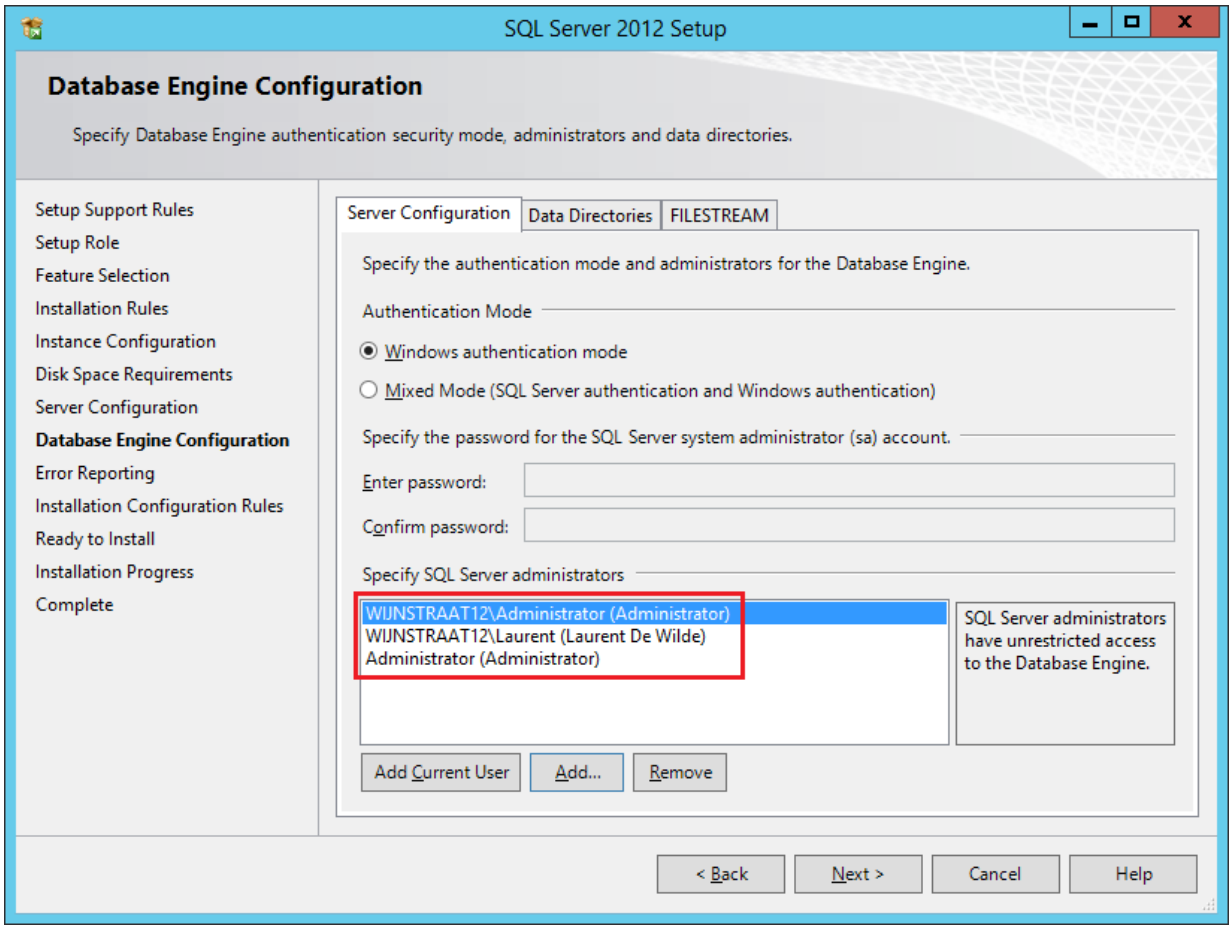

Figure 4.11: Windows Authentication Mode is chosen and the SQL administrators are added.

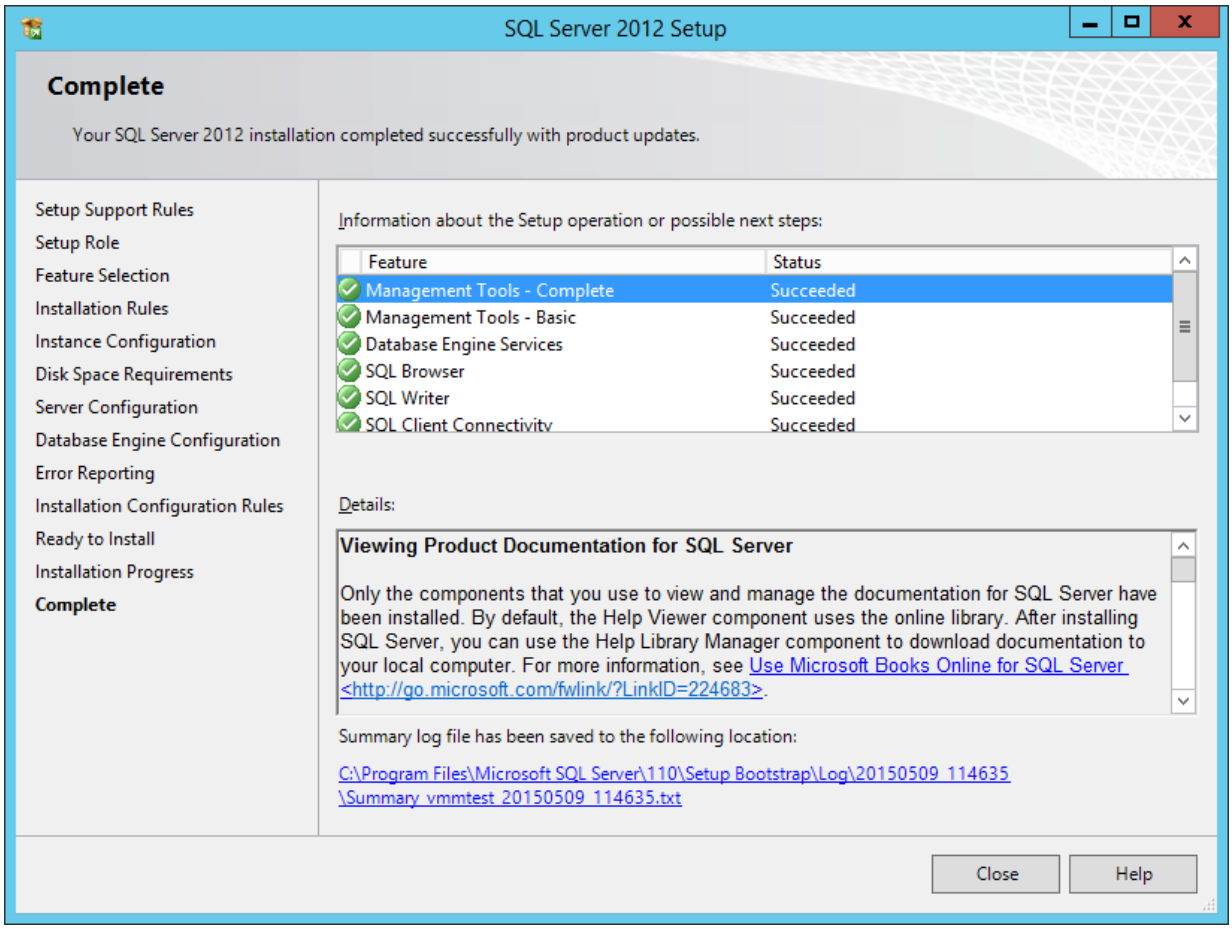

Figure 4.12: Setup has completed successfully.

4.1.5 Configuring Distributed Key Management in Active Directory

Distributed key management is used when a VMM cluster is used. The database of VMM is encrypted because it contains sensitive data. Consider the situation where two servers are used to form a clustered VMM environment. If the decryption keys are stored locally on server 1 and this particular server goes offline, there is no way to access the decryption keys anymore. This is why the keys are stored in a special container in Active Directory. This way, anytime access to the decryption keys is guaranteed.
In combination with DKM, a service account needs to be as well. This account is used to, for example, share .ISO images in the shared libraries of VMM.

Let us configure Distributed Key Management (DKM) and the service account.

To do so, a new OU named "ServiceAccounts" is made under the root domain in the "Active Directory Users and Groups" snap-in. In this newly created OU, a (domain)user "VMMService" is made. This user serves as the service account for VMM.

Make sure that the password never expires.

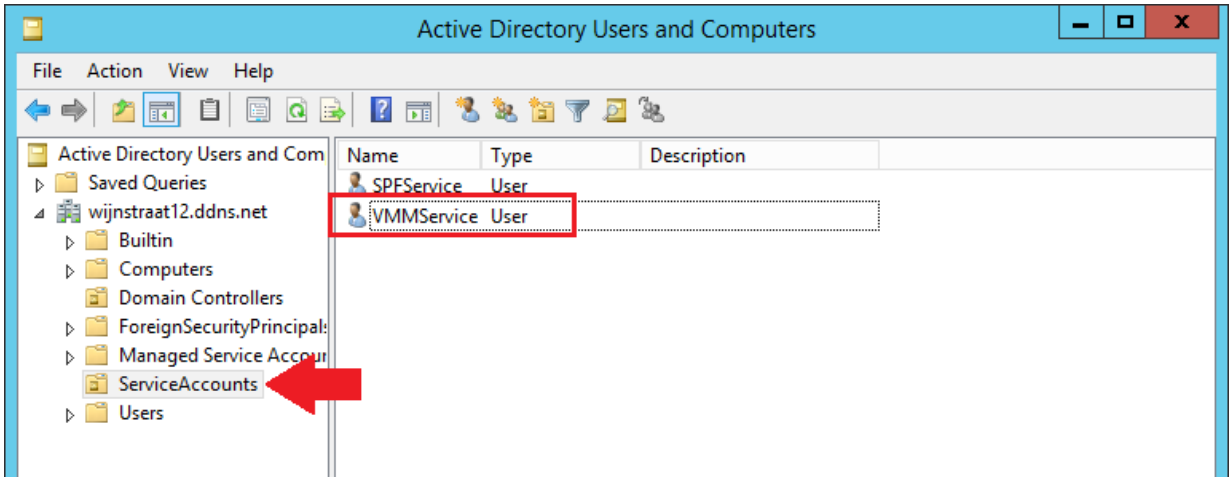

Figure 4.13: Making the service account for DKM.

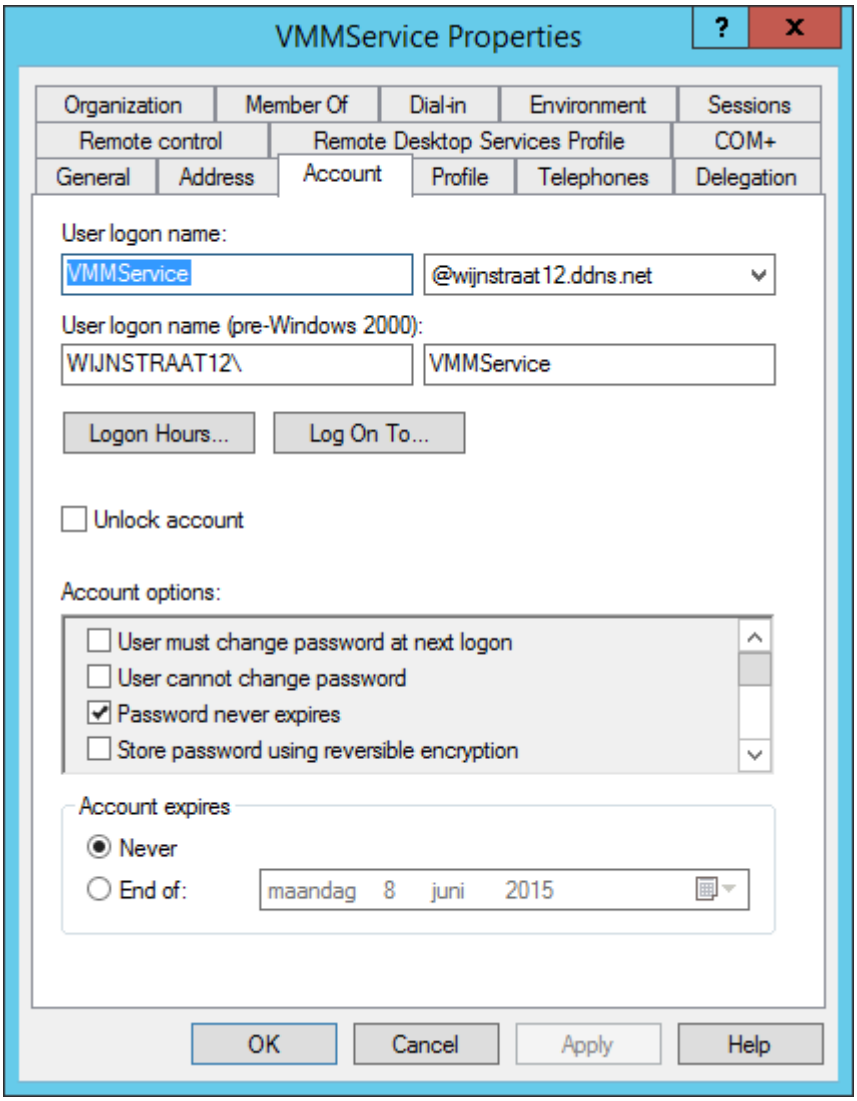

Figure 4.14: The properties of the service account.

Now, the actual container for the keys can be made in Active Directory. To do so, open "ADSI Edit" from the local server overview of the server manager: `Server Manager` → `Local Server` → `Tools` → `ADSI Edit`.
Accept the default naming context (click `OK`) and expand the `Default naming context` node. Right click on the root OU and choose `New` → `Object`.
Select `container` and name it for example "VMMDKM". Click `Finish`.

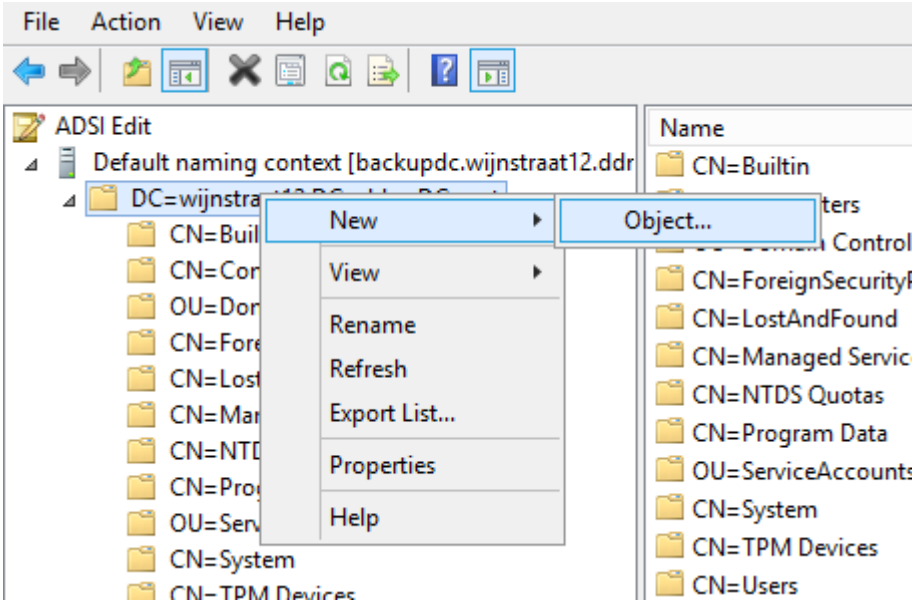

Figure 4.15: Create a new Object.

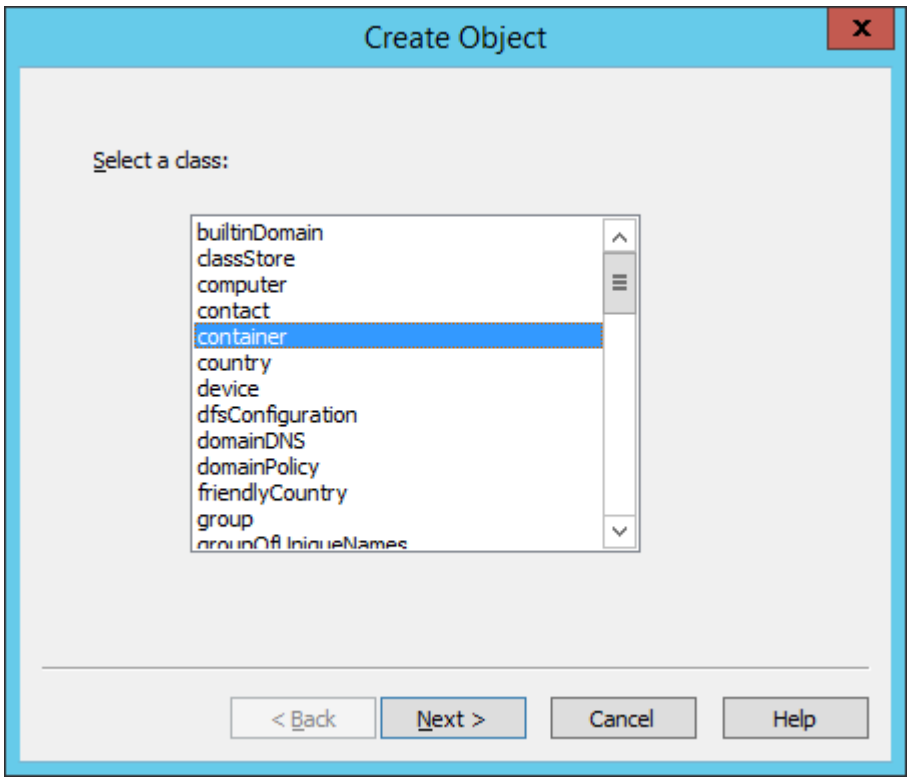

Figure 4.16: Creation of the container.

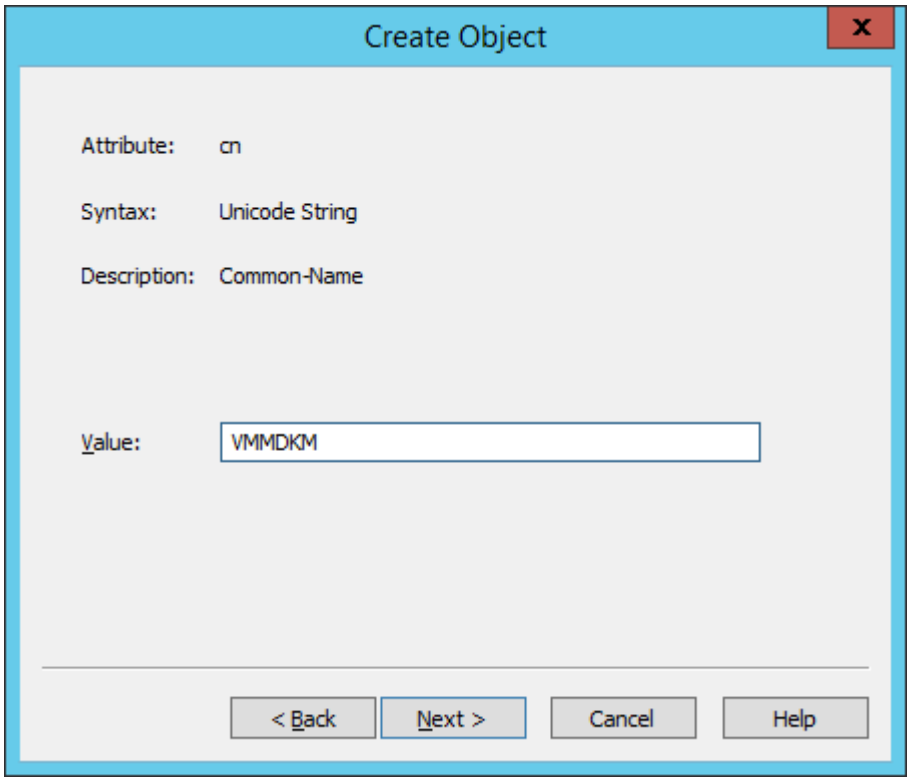

Figure 4.17: Creation of the container.

Next, some permissions for the VMMDKM container must be set. To do so, right click on the newly created container and select `Properties`. On the `Security` tab, click advanced. Click `Edit`.
Make sure the principal is "VMMService". and change `Applies to:` to "This object and all descendant objects".

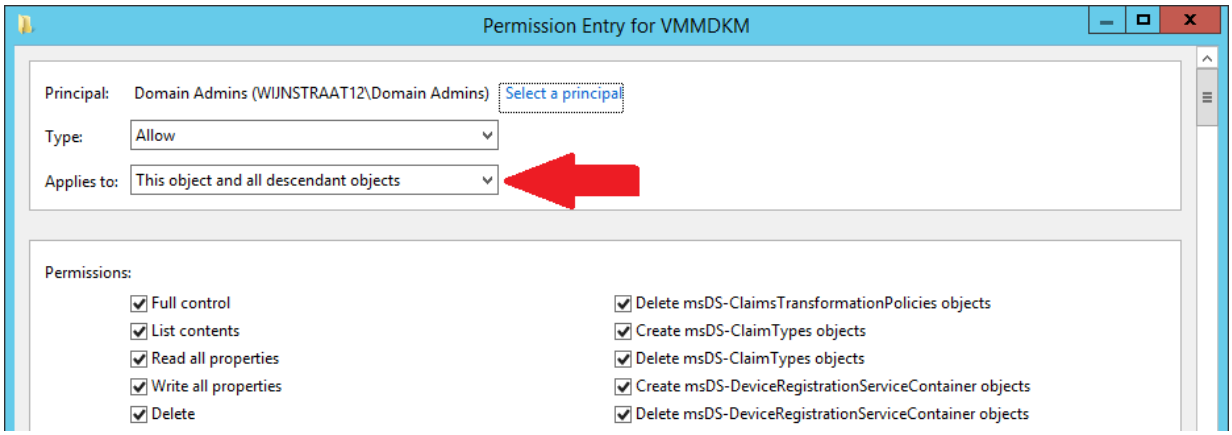

Figure 4.18: Configuration of the permissions for VMMDKM.

After this, the VMMService account needs to be added to the **local** administrators group. To do so, open computer management and select `Local Users and Groups` → `Groups`. Right click on the `Administrators` group and choose `Properties` and add the "VMM-Service" account to the group. Click `OK`.

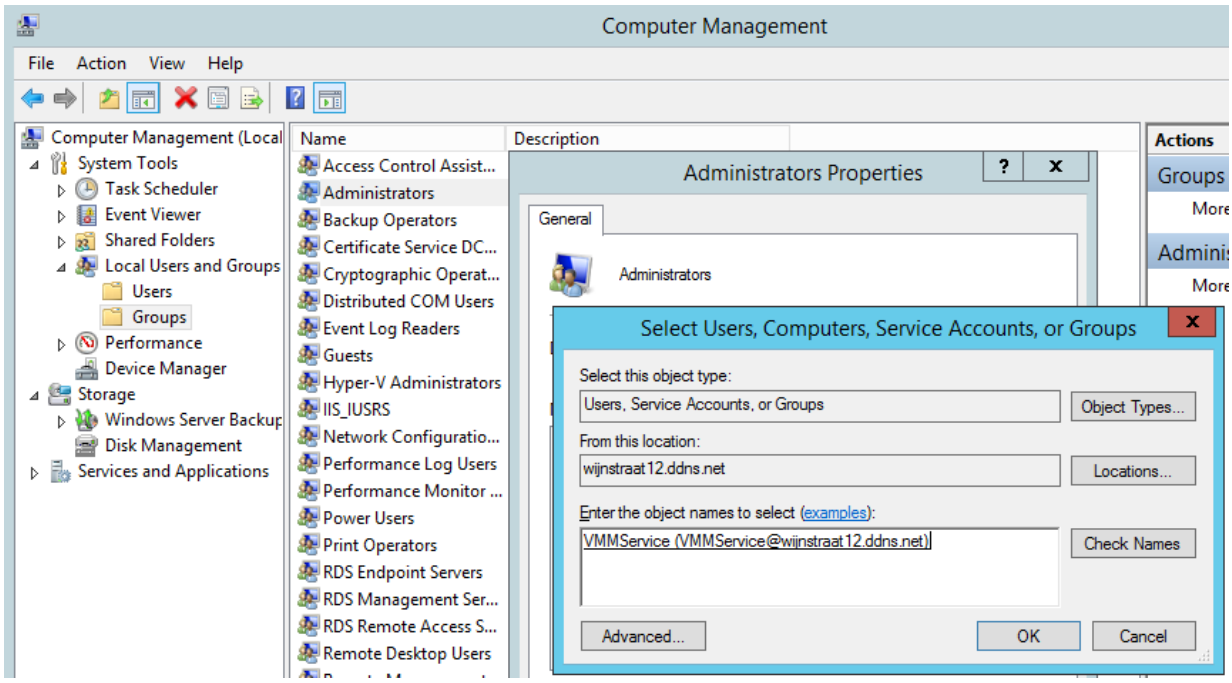

Figure 4.19: Adding the VMMService account to the local administrators group.

4.1.6 Post configuration of MSSQL Server

The VMMService account has to be given database acccess. Therefore, open "SQL Server Management Studio", login using Windows Authentication and expand the `Security` folder. Right click on `Logins` and create a new login. This login account is the VMMService account.

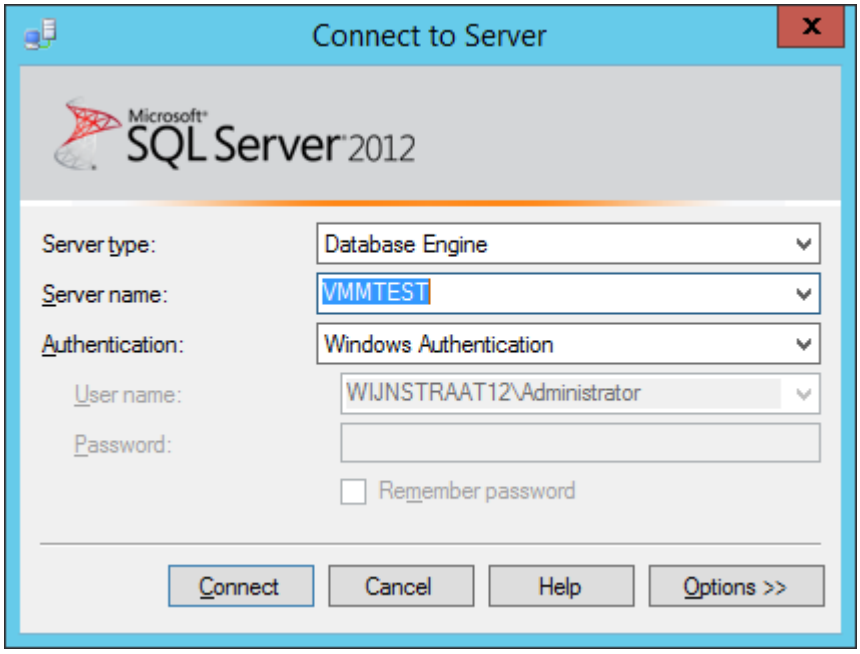

Figure 4.20: Login into SQL Server Management Studio using Windows Authentication.

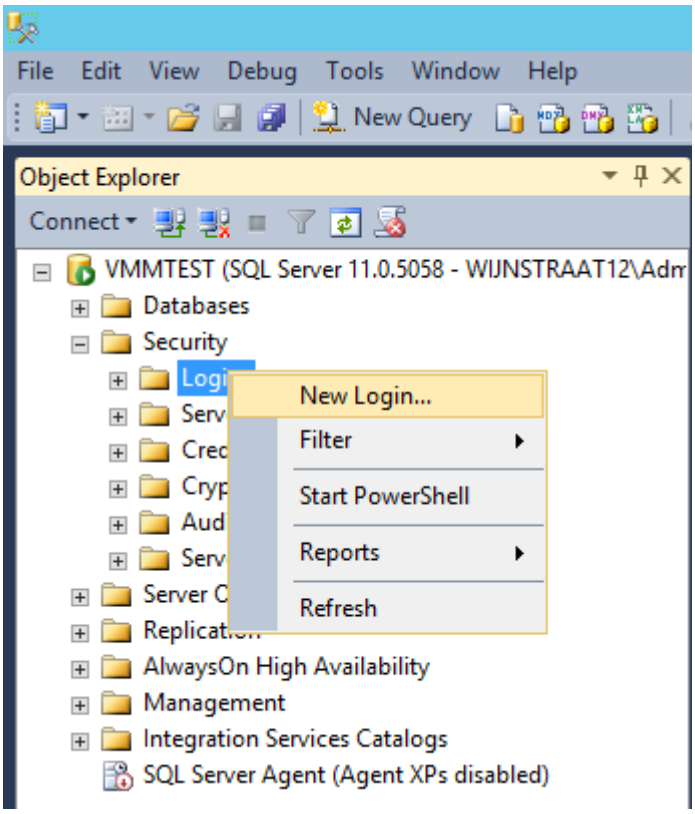

Figure 4.21: Creation of a new login.

On the `Server role` page of this new login, make sure to check the "dbcreator", "process admin", "public" and "security admin" roles. After this is completed, exit Management Studio.

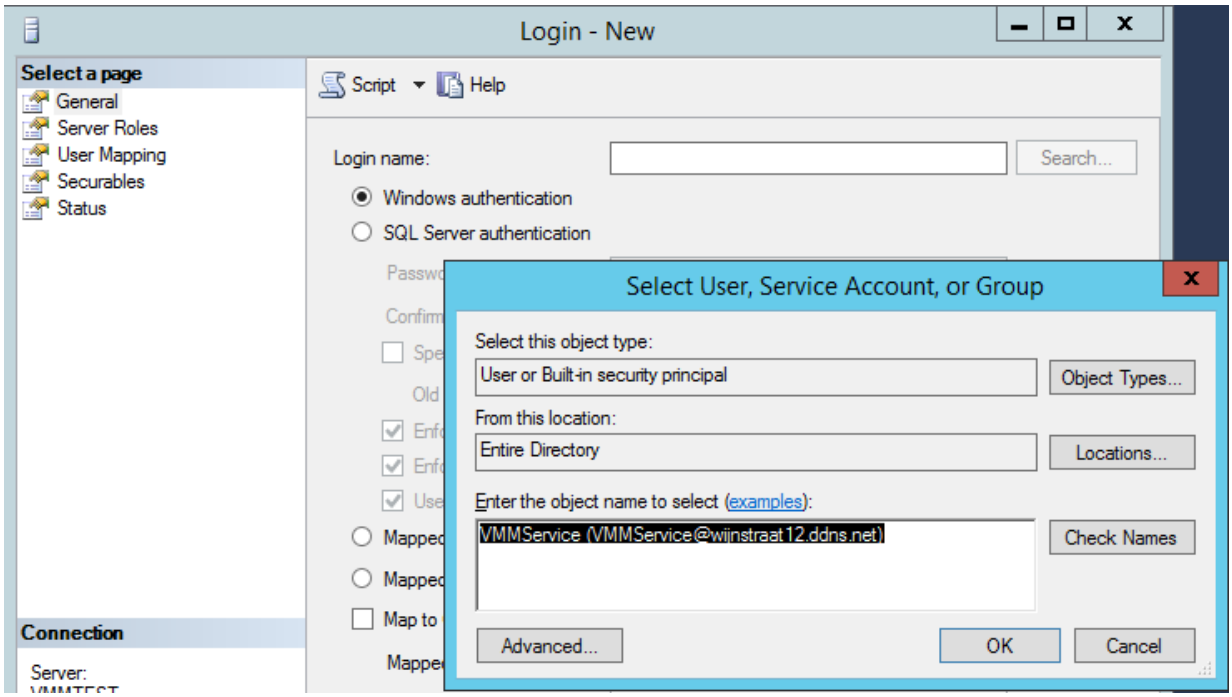

Figure 4.22: Selecting the VMMService account.

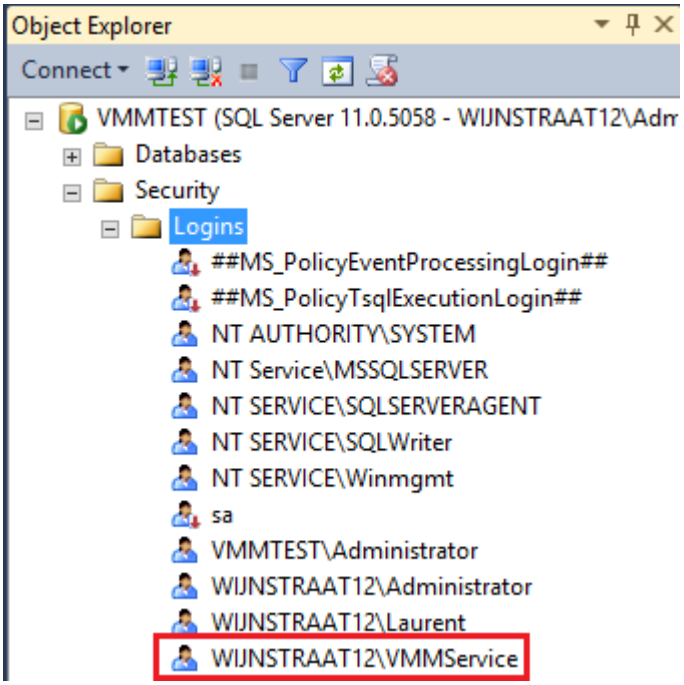

Figure 4.23: Confirmation.

4.1.7 Installation of VMM

There is one prerequisite that is not fulfilled yet: the installation of the Windows Assessment and Deployment Kit. Since Windows Server 2012 R2 is used, Windows ADK for Windows 8.1 must be installed. The Windows ADK can be downloaded from the Microsoft website: `https://www.microsoft.com/en-us/download/details.aspx?id=39982`

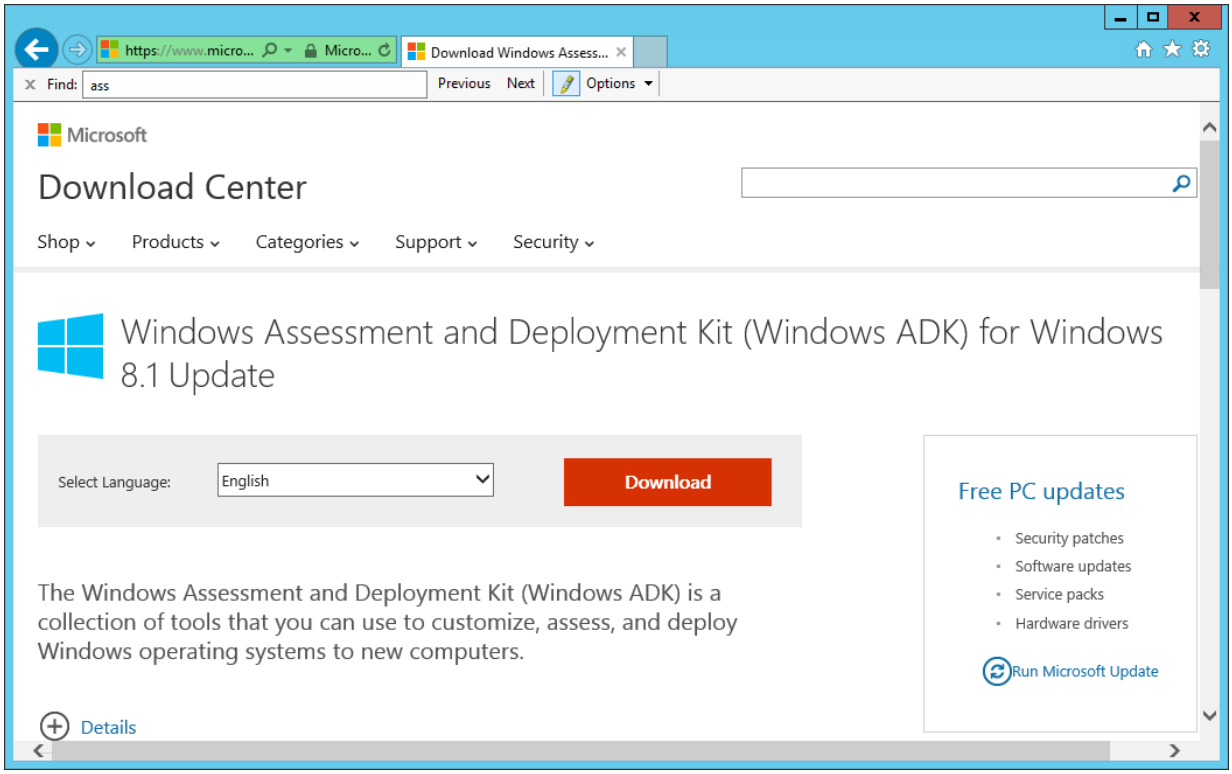

Figure 4.24: Download the Windows ADK for Windows 8.1.

Since we are installing VMM on the computer we are currently logged into, choose the option "Install the Windows Assessment and Deployment Kit for Windows 8.1 to this computer".

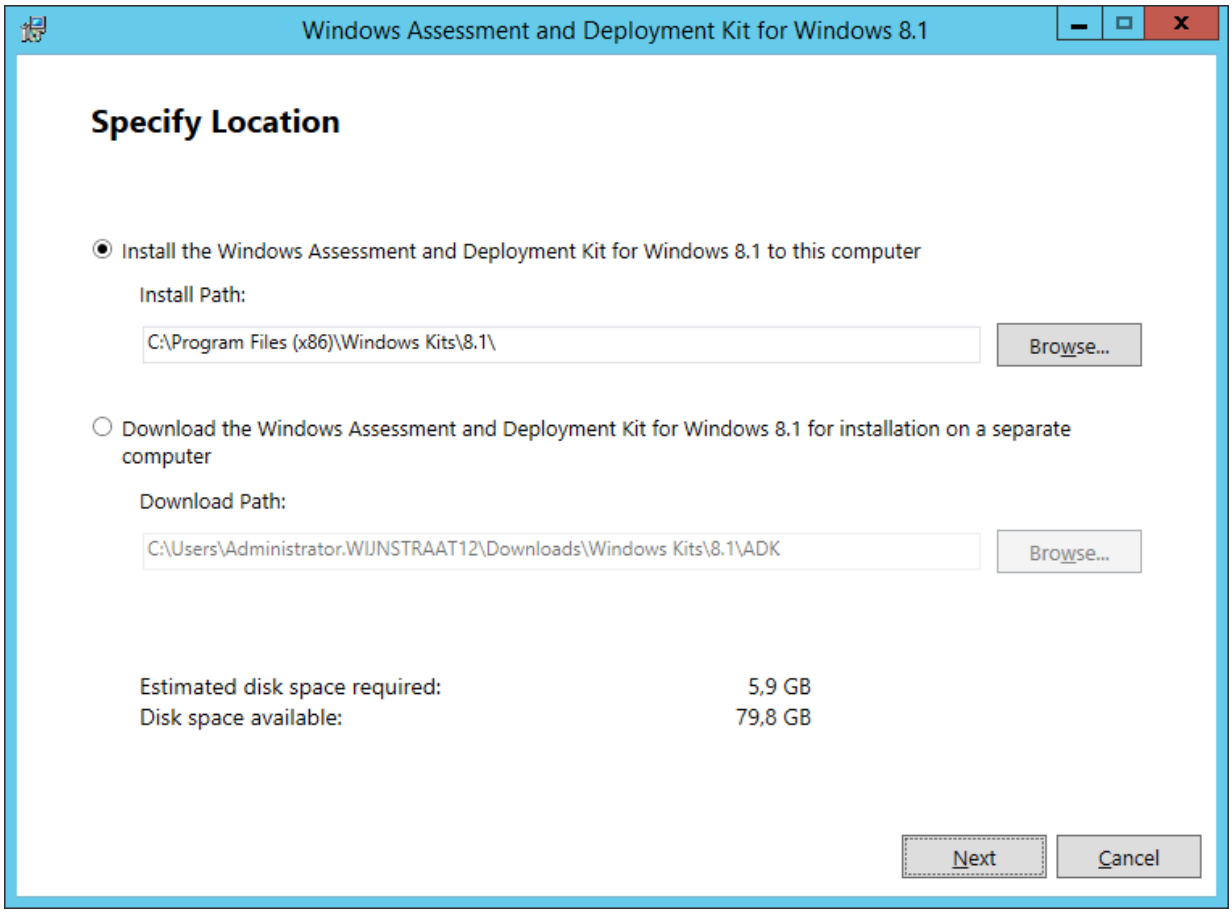

Figure 4.25: Install the ADK on the current compute, since VMM will also be installed on this local computer.

Choose whether or not you want to participate with the Customer Experience Improvement Program and acccept the License Agreement.
On the "Select features screen", make sure you only select "Deployment Tools" and "Windows Preinstallation Environment".

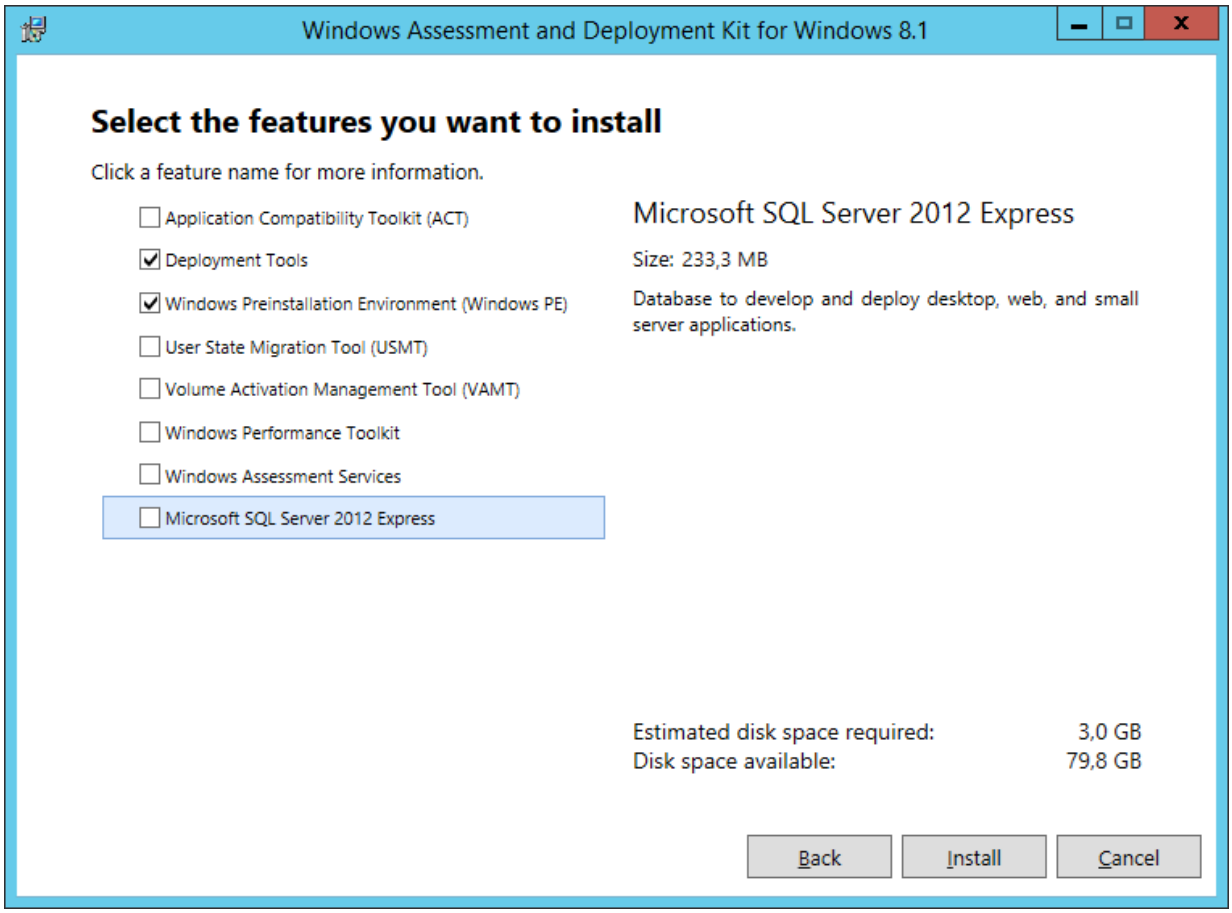

Figure 4.26: Install only the Deployment Tools and Windows PE.

Now the actual installation of SCVMM can be started. To do so, double click on the .msi file that has been downloaded from the Microsoft site. After extracting the files, open the folder and double click on `setup.exe`. The main screen of SCVMM is shown.

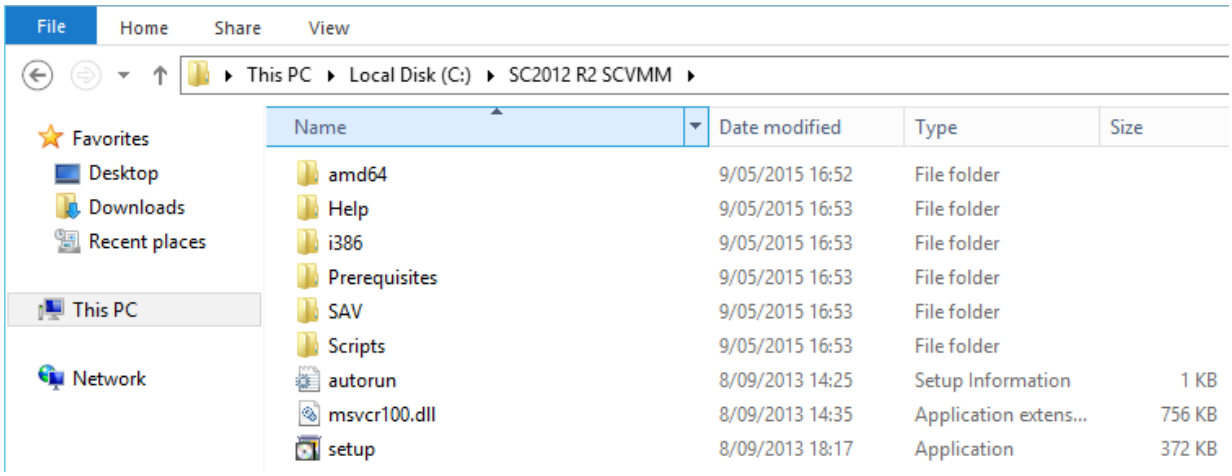

Figure 4.27: Extract the files and double click on `setup.exe`.

Since we want to install VMM, choose for `Install`.

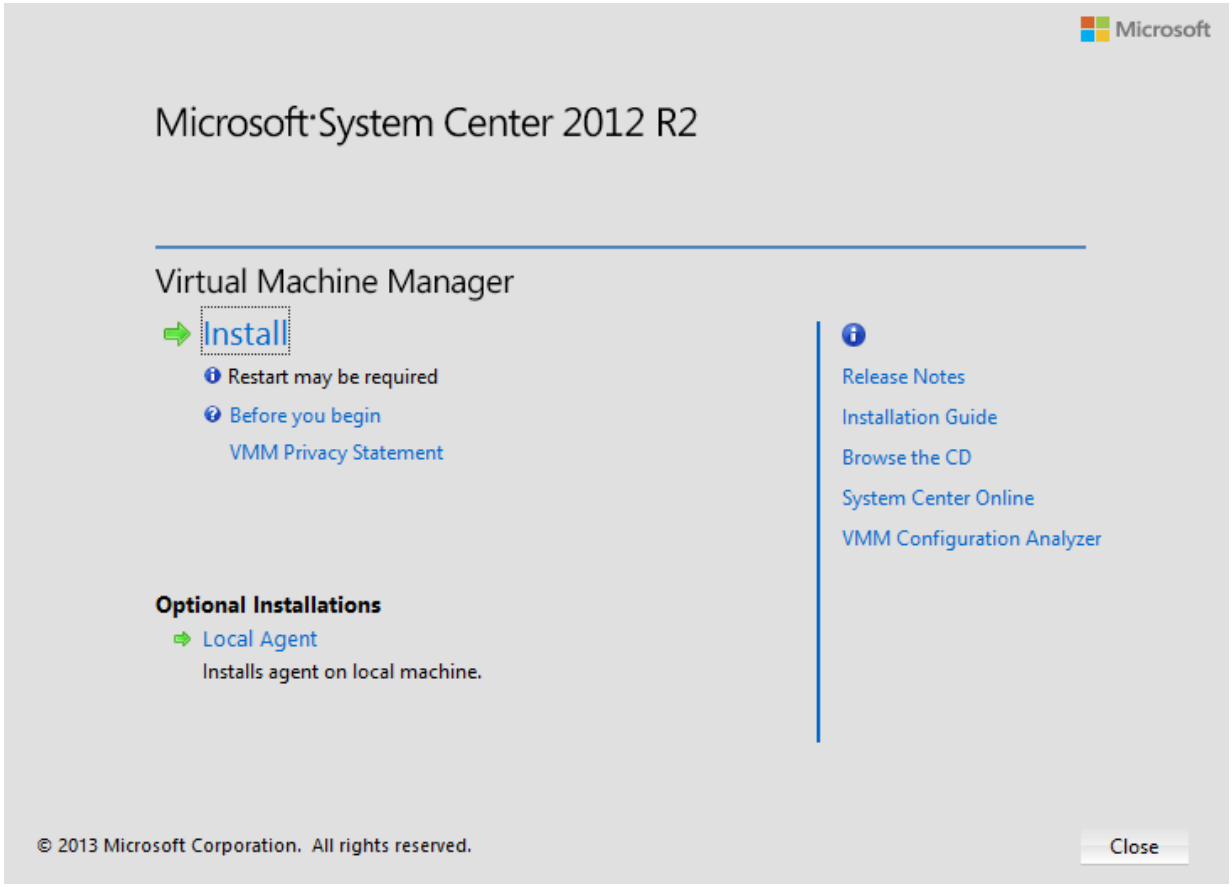

Figure 4.28: The main screen of Virtual Machine Manager 2012 R2 is shown.

Select `VMM management server`, the `VMM console` is automatically selected as well. The console is the Graphical User Interface of Virtual Machine Manager that allows one to connect to the VMM management server.

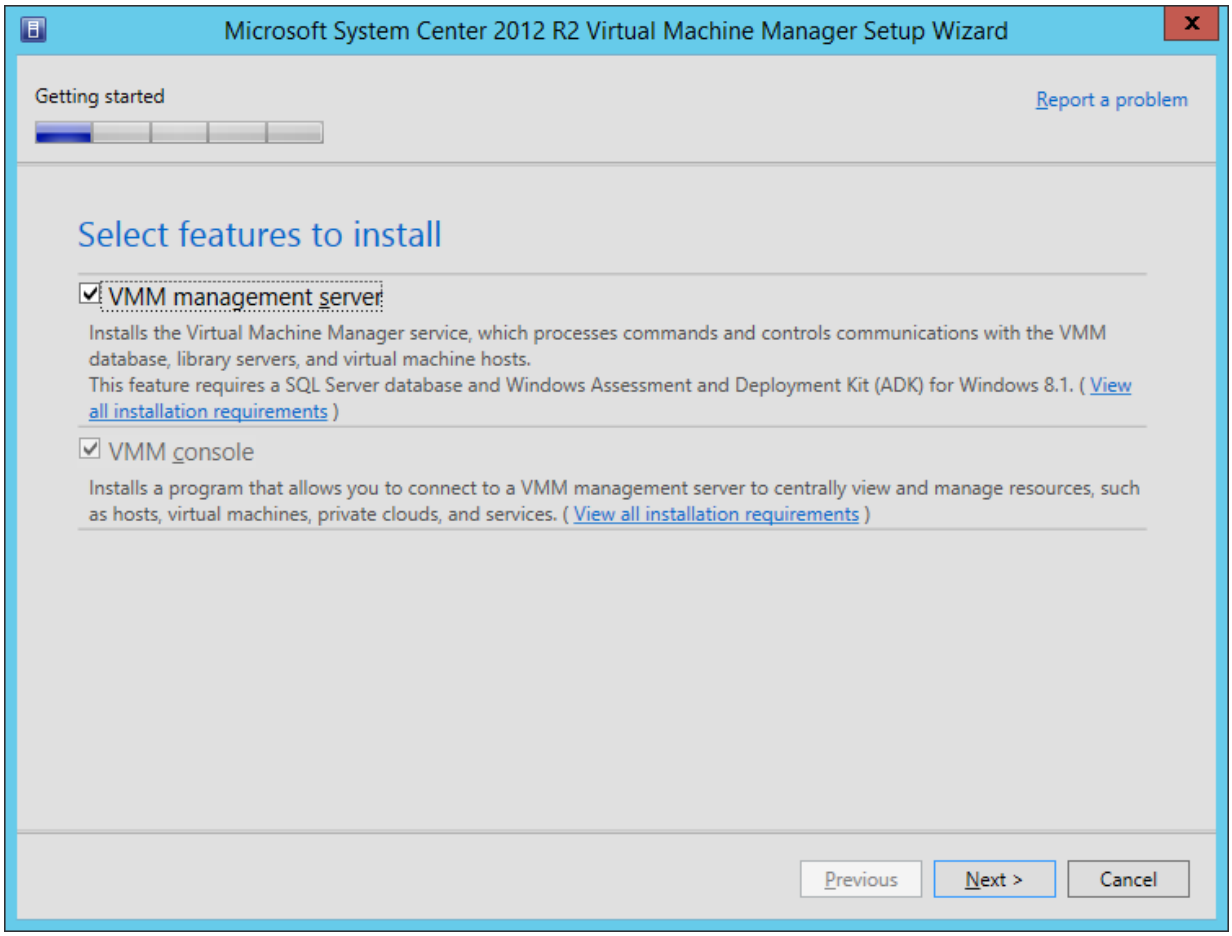

Figure 4.29: Select the VMM management server.

Next, the installation location has to be chosen.

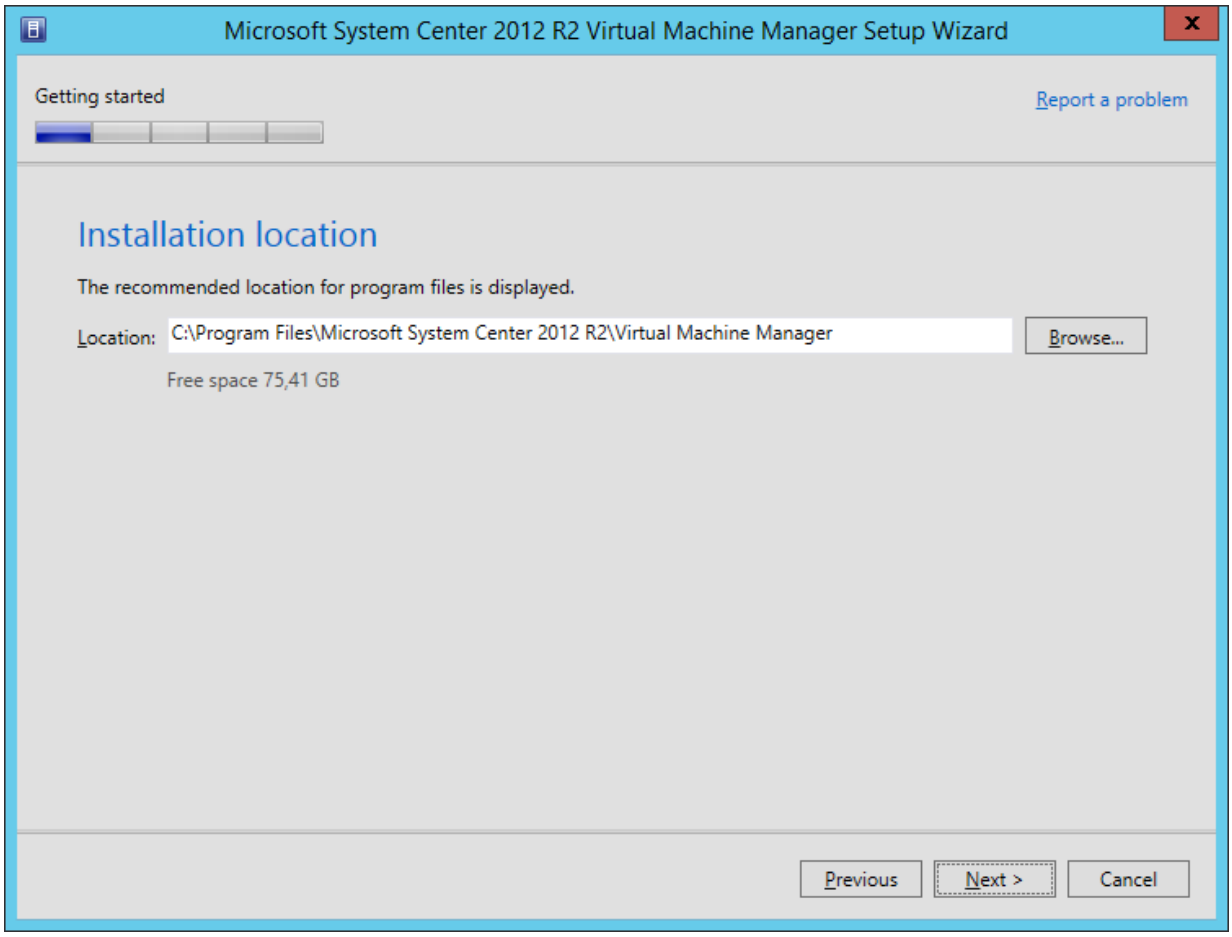

Figure 4.30: Choose the installation location.

On the next screen, the information regarding the database connection and configuration has to be provided. On the `Server name` field, "VMMTEST" or "localhost" are both valid names. Leave the `Port` field empty. The default port is used.
We want to login using the VMMService account. This is why we created the login in the previous section. Enter its name and password. Make sure to also provide the NETBIOS domain name of the account in capital letters, followed by a backslash (\) before the account name. An example could be: `WIJNSTRAAT12\VMMService`.
Specify the `Instance name` ("MSSQLSERVER" by default) and select `New database`.

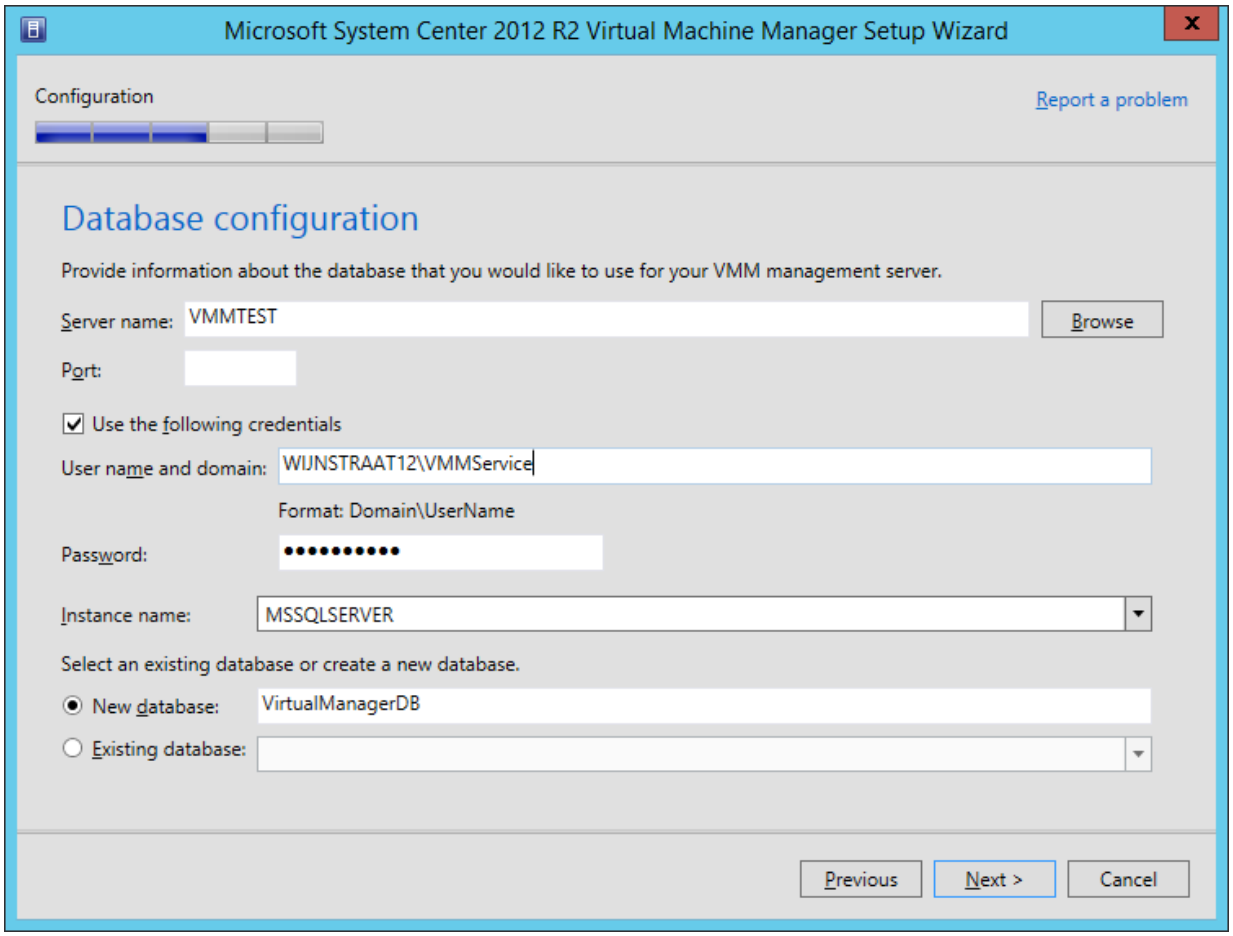

Figure 4.31: Configuration of the database.

Now, Distributed Key Management has to be configured. Another reason why a service account has been configured in the Active Directory. Note that DKM is optional, but if we want to configure a high availability VMM cluster in the future, the prerequisite work is already performed.
Provide the same VMMService account as used in the previous step(s). Don't forget to also specify the NETBIOS domain name.
Check `Store my keys in Active Directory` and specify the DN (Distinguished Name) of the VMMDKM container created earlier.

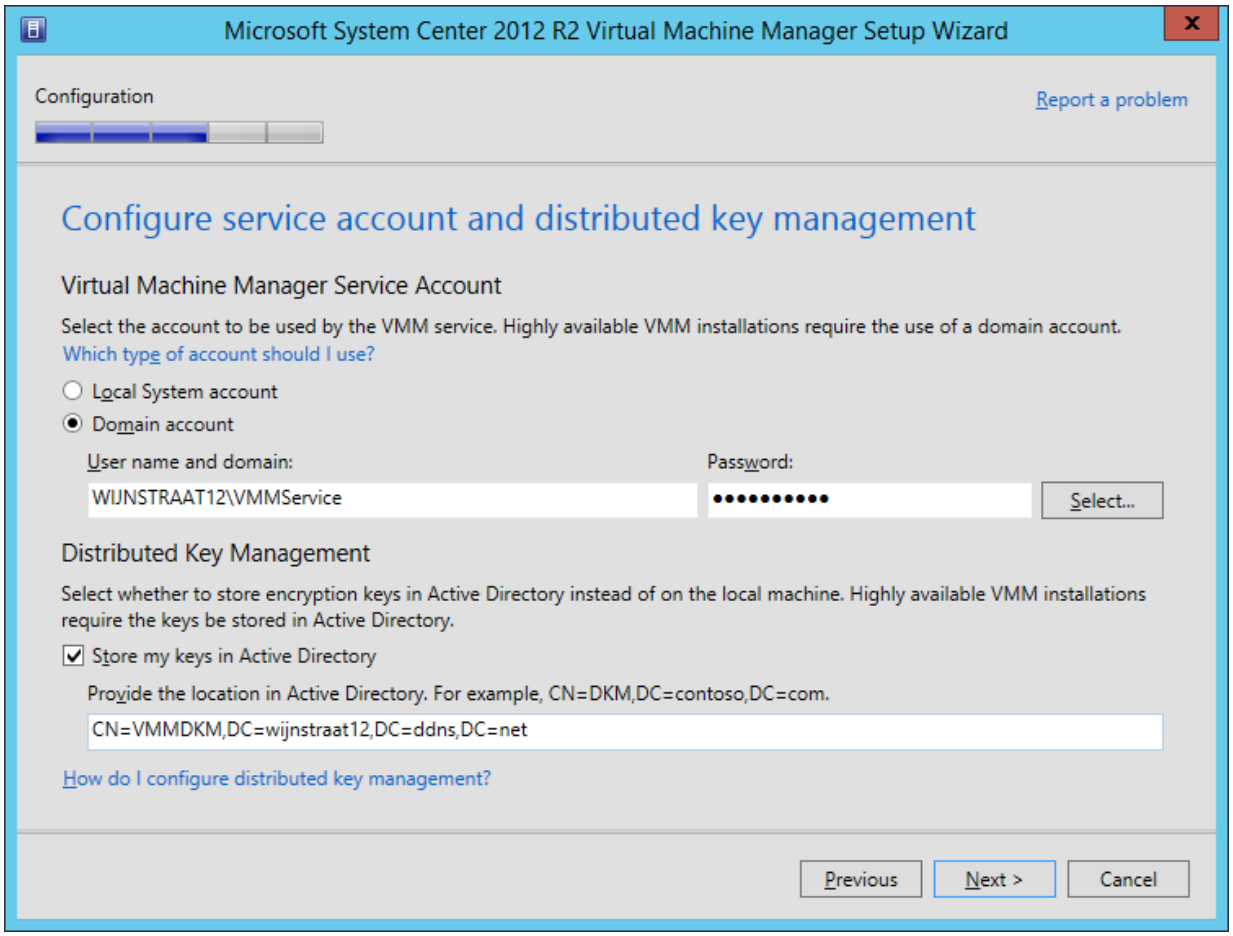

Figure 4.32: .

In the shared library, all your shared files such as .ISO images will be stored. Therefore, it might be a good idea to specify another destination of with more space than the default location. In this example, the default location is used.

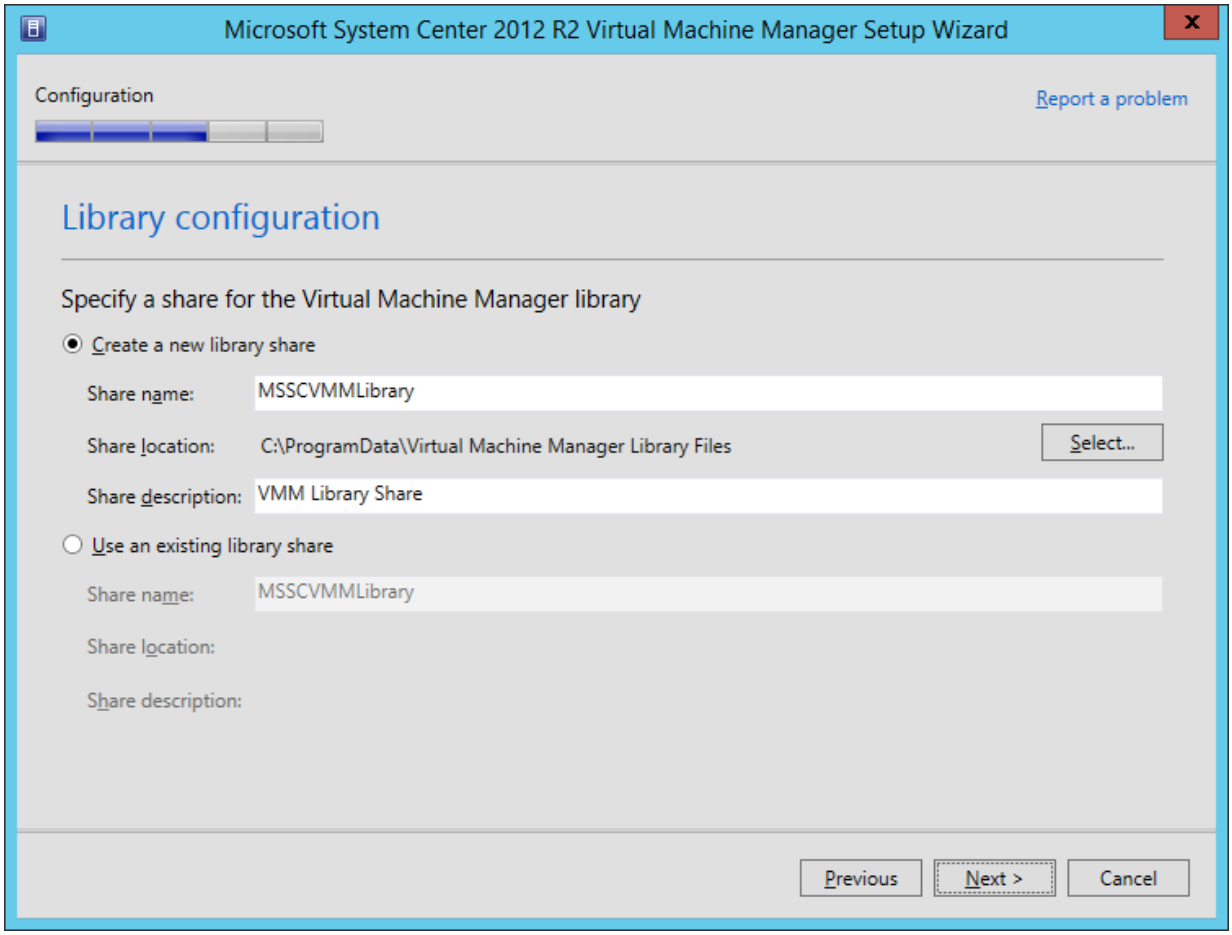

Figure 4.33: Specify the location of the shared libraries.

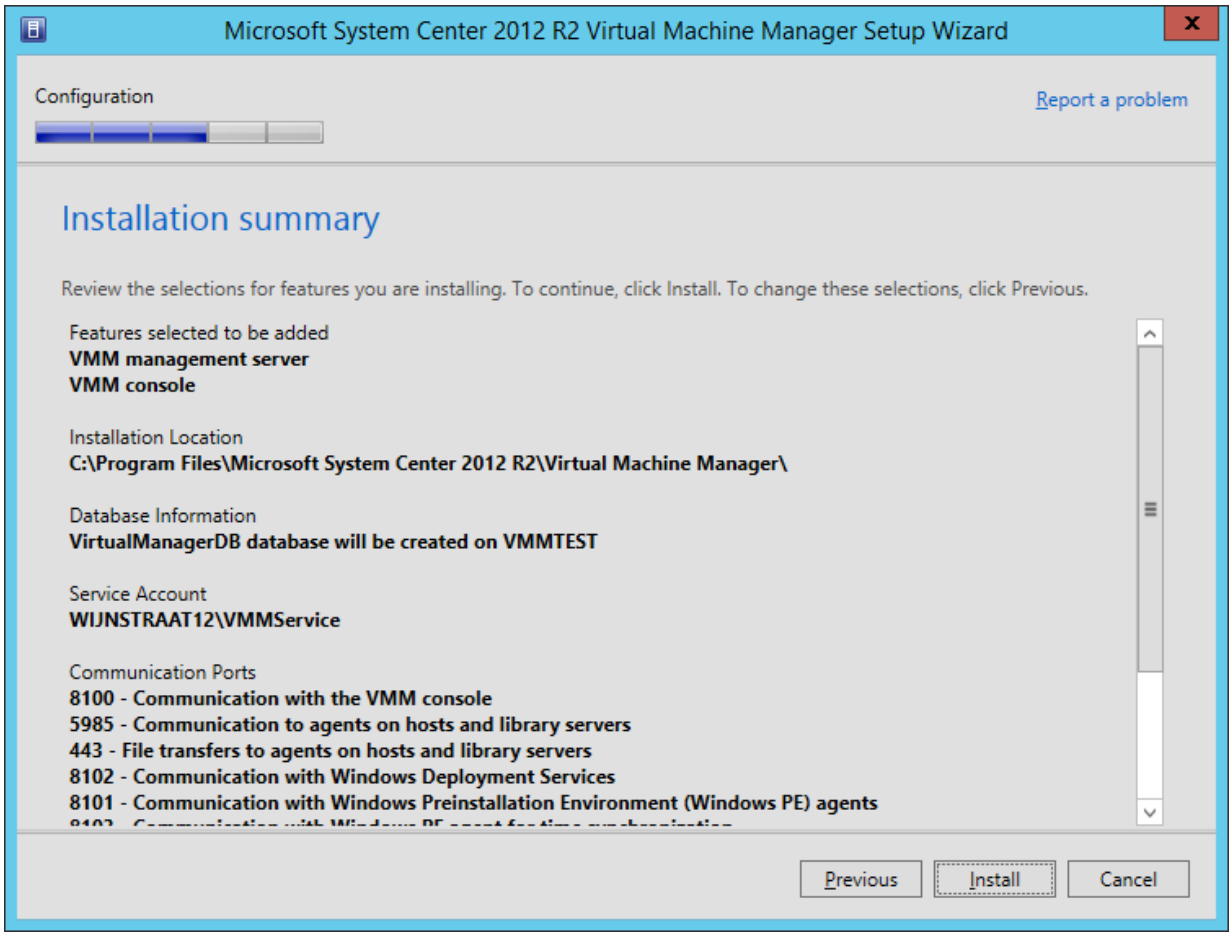

Figure 4.34: Overview of the configuration settings. Click `Install`.

4.2 Installation of Windows Azure Pack

Windows Azure Pack (formerly known as Windows Azure Services for Windows Server [Vredevoort, 2013]), is a collection of Windows Azure technologies that allows one to build its own private cloud and runs on top of SC Virtual Machine Manager 2012 R2 which, in his turn, runs on top of Windows Server 2012 R2. It basically brings Windows Azure to one's own datacenter [Maurer, 2014b,a].
Prior to installing Windows Azure Pack, the Service Profider Foundation (SPF) needs to be installed. It offers Infrastructure as a Service (IaaS) and is installed on top of VMM. So basically, the layered infrastructure is like this: `Windows Server 2012 R2` → `Virtual Machine Manager 2012 R2` → `Service Provider Foundation` → `Windows Azure Pack`.

Windows Aure Pack is installed on a dedicated machine called `windowsazure`. Windows

Aure Pack (or WAP for short) requires Microsoft SQL Server 2012 or higher. The SQL Server installed on the VMM virtual machine can be used, or a new, local, instance can be created.

4.2.1 Prerequisites

The installation of Windows Service Provider Foundation requires some additional steps to be performed prior to its installation. The following requirements must be met:

- Operating System: Windows Server 2012 or higher with PowerShell 3.0.
- System Center Virtual Machine Manager console SP1 or higher.
- Web Server IIS Server role with Scripts and Tools, Basic Security, Windows Authentication and ASP.NET 4.5.
- .NET Framework 4.5 or higher + HTTP Activation.
- Management OData IIS Extension.
- WCF Data Services 5.0 for OData V3.
- ASP.NET MVC 4

The installation of all the prerequisites will be covered in this manual. As previously said, WAP is installed on a dedicated VM with a fresh install of Windows Server 2012 R2. The server is given a static IP address and is made member of the same AD domain that the VMM server is running in. That is, `wijnstraat12.ddns.net`.

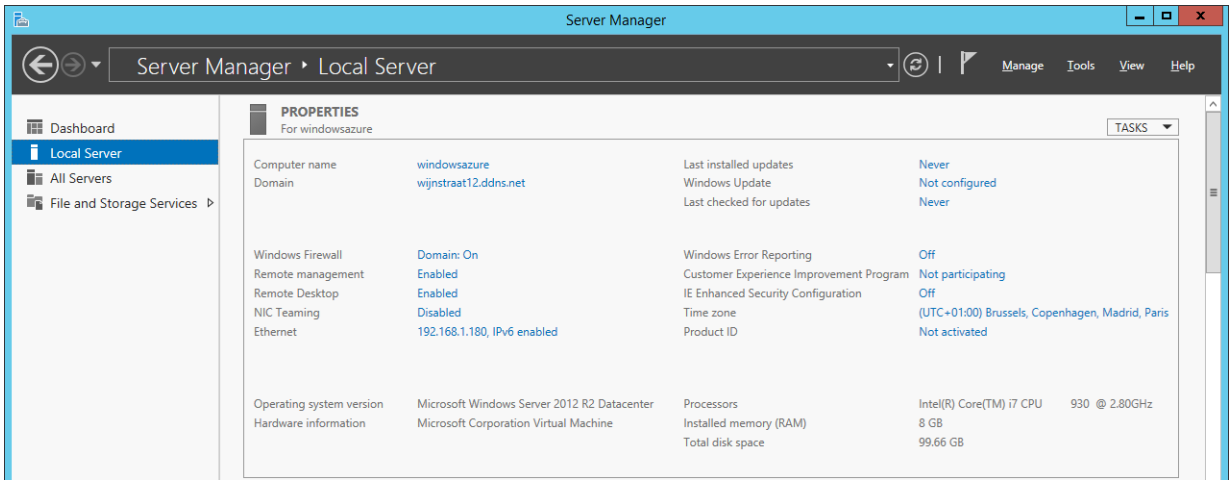

Figure 4.35: Overview screen of the local server showing the begin situation.

When choosing an installation of a new instance of SQL Server, make sure to select `Mixed Mode` as authentication mode. When reusing the existant instance of SQL Server, additional configuration will have to be performed in order to be able to install the WAP.

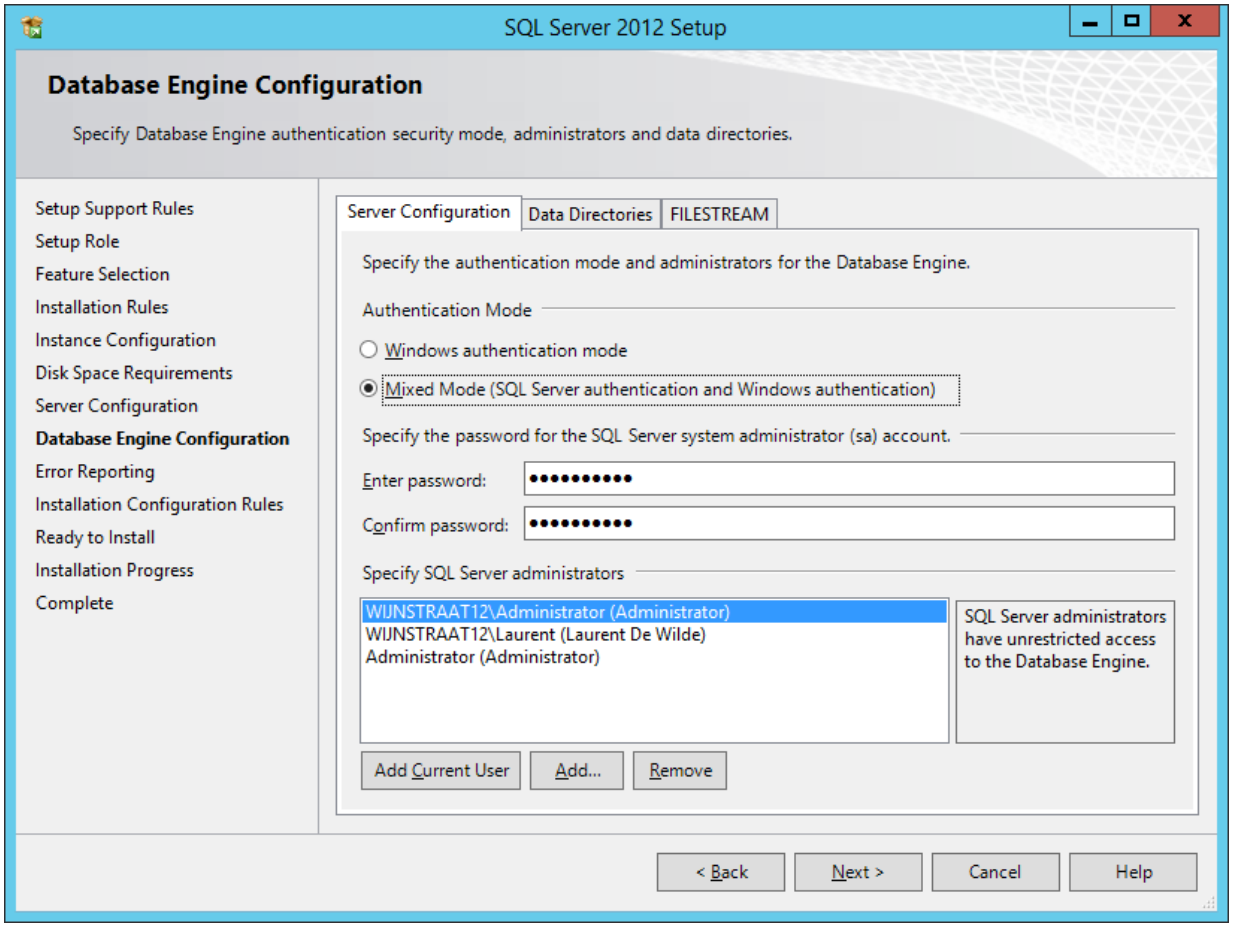

Figure 4.36: Authentication Mode: Mixed Mode.

Now the server is ready to install the prerequisites. The installation of the VMM console is the first requirement and can be installed from the VMM installation media. Extract the files of the .msi file and lauch `setup.exe`. Choose `Install` and check `VMM console`. Click `Install` on the confirmation screen.

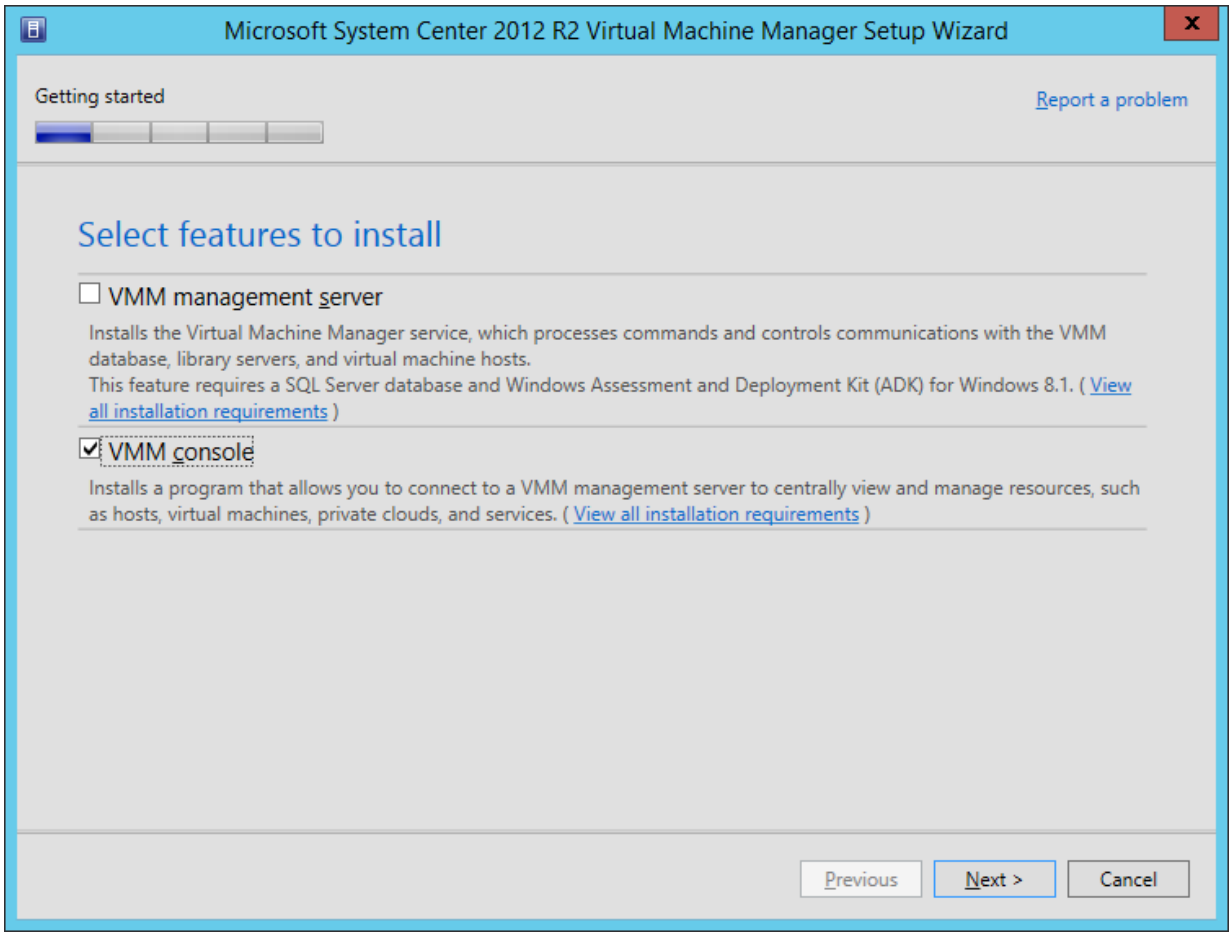

Figure 4.37: Install only the VMM console.

Next step is the installation of the Web Server (IIS). Therefore, open `Add roles and services` from the Server Manager and check `Web Server (IIS)`. Click `Next`.

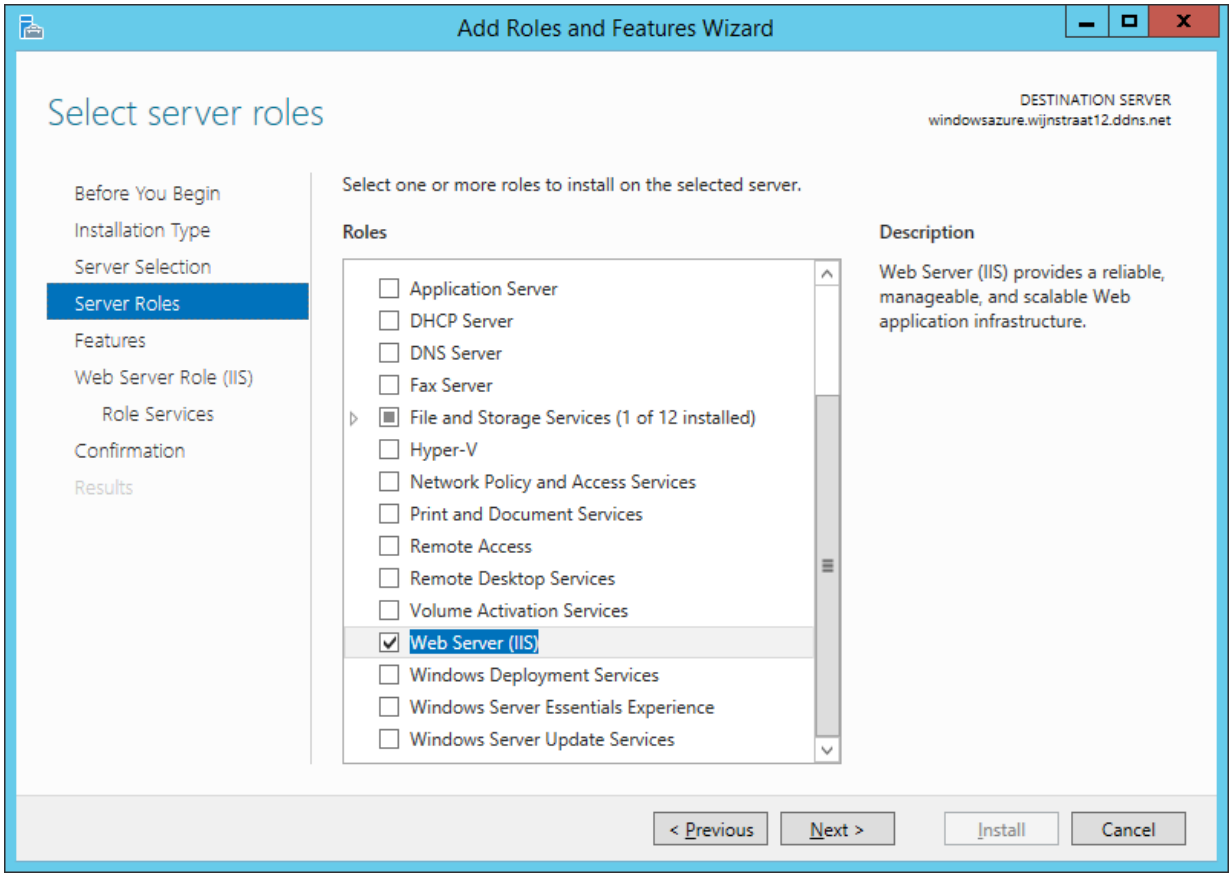

Figure 4.38: Select the `Web Server (IIS)` Server Role.

On the `Features` screen, select the following features:

- `.NET Framework 4.5 Features` → `WCP Services` → `HTTP Activation`
- `Management OData IIS Extension` and its associated features.

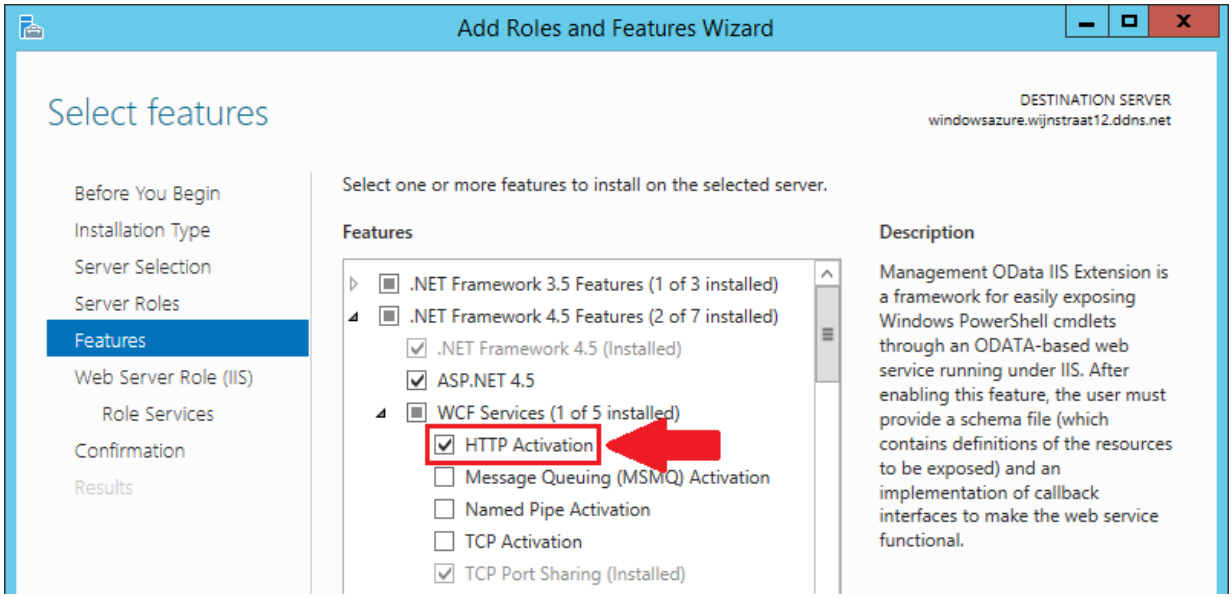

Figure 4.39: Select `HTTP Activation`.

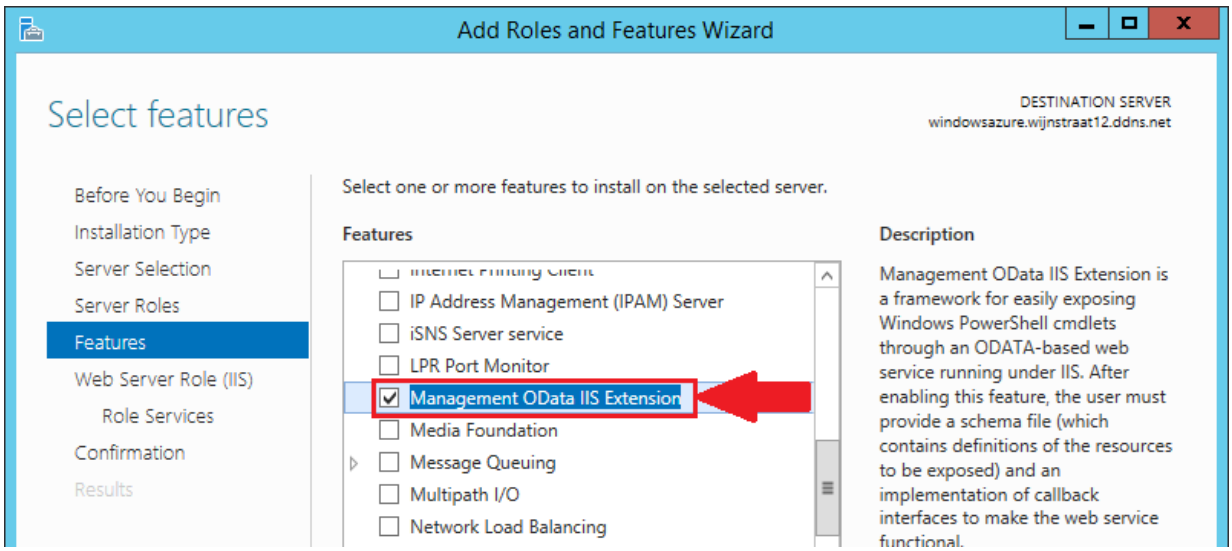

Figure 4.40: Select `Management OData IIS Extension`.

On the `Web Server Role (IIS) - Role Services` screen, select following role services:

- `IIS Management` → `Scripts and Tools`
- `Security` → `Basic Authentication`
- `Security` → `Windows Authentication`
- `Application Development` → `.NET 4.5`

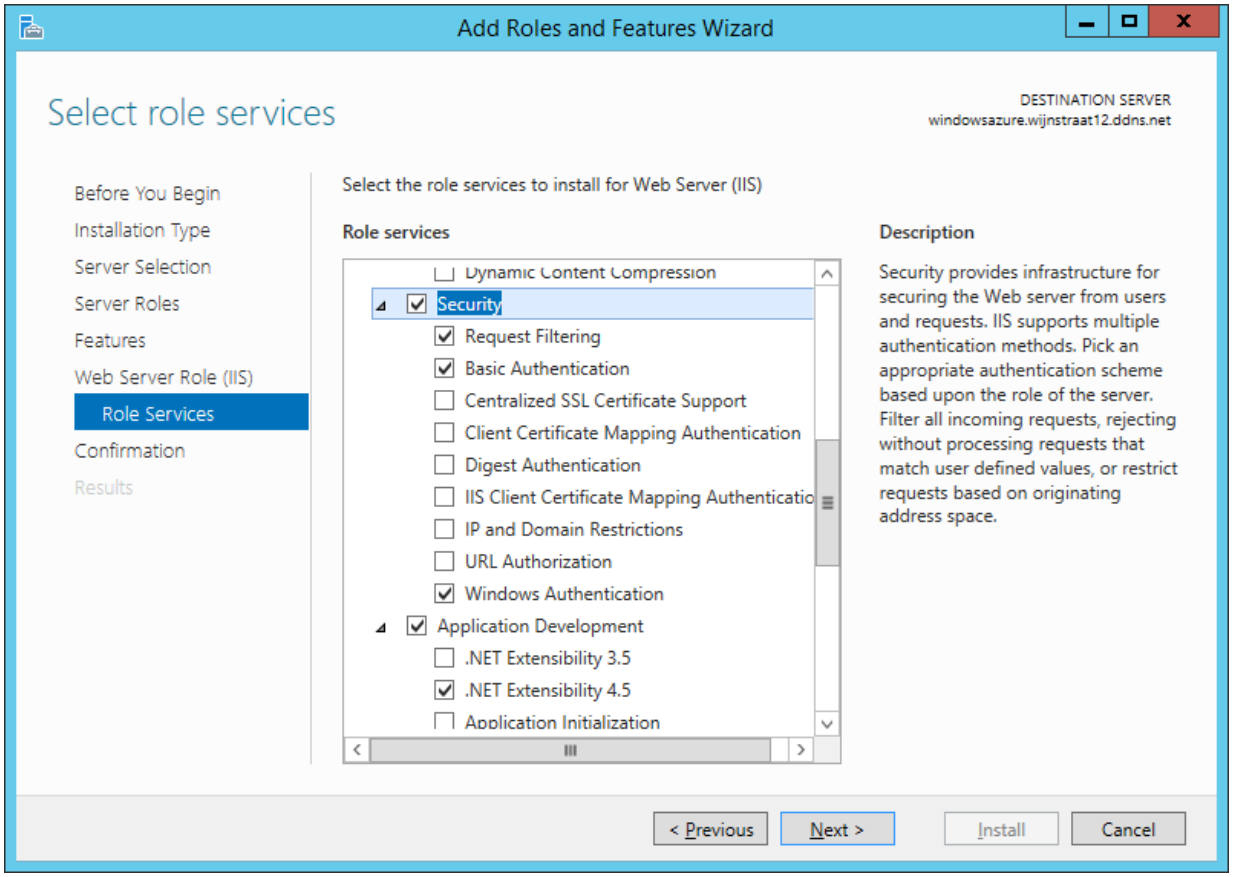

Figure 4.41: .

Some additional software needs to be installed being `WCF Data Services 5.0 for OData V3` and `ASP.NET MVC 4`.
The `WCF Data Services 5.0 for OData V3` can be downloaded from this location: `https://www.microsoft.com/en-us/download/details.aspx?id=29306`. The installtion is straightforward and will not be covered. Just execute the installer, accept the License Agreement and click `Install`.
`ASP.NET MVC 4` can be download from the following location: `https://www.microsoft.com/en-us/download/details.aspx?id=30683`. The installation is roughly the same as the WCF Data Services.

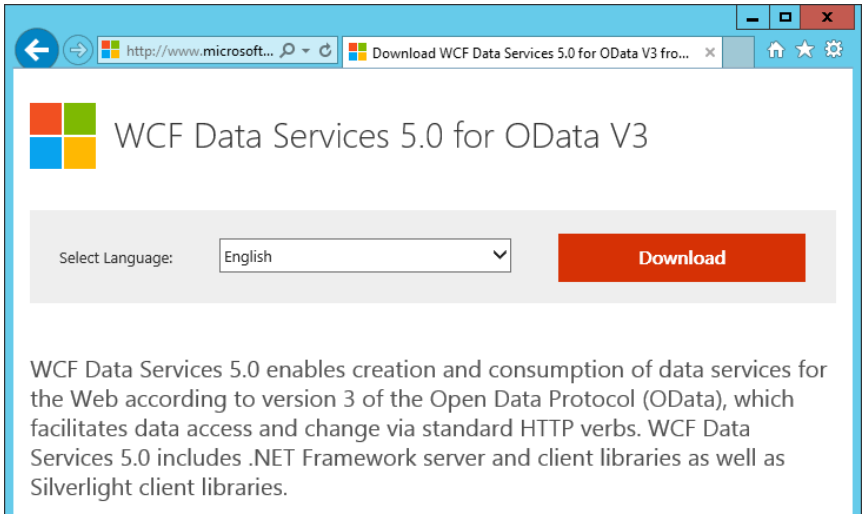

Figure 4.42: Download and install WCF Data Services for OData V3.

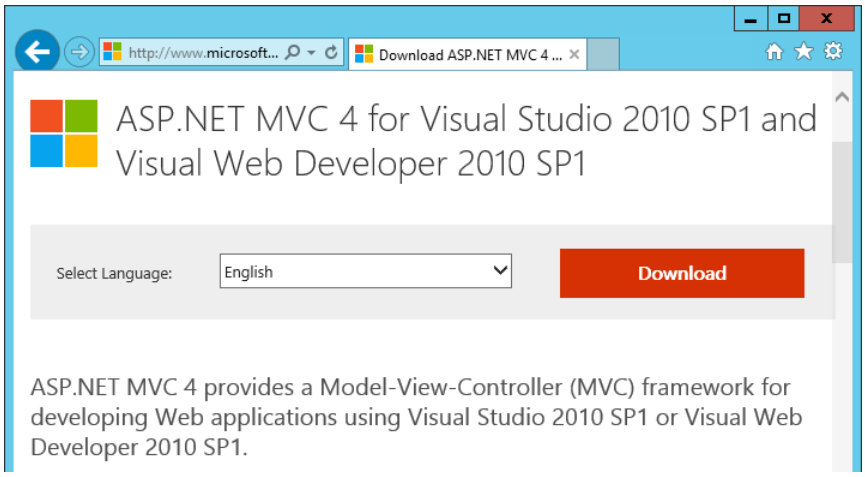

Figure 4.43: Download and install ASP.NET MVC 4.

Also for the Service Provider Foundation, a service accounts needs to be made. Therefore, add a domain user in the Active Directory called `SPFService`. Make sure the password never expires.
Four additional domain groups need to be made for setting permissions on the directories created by the installer. Those four groups are: `SPFAdmins`, `SPFProvider`, `SPFUsage` and `SPFVMM`.

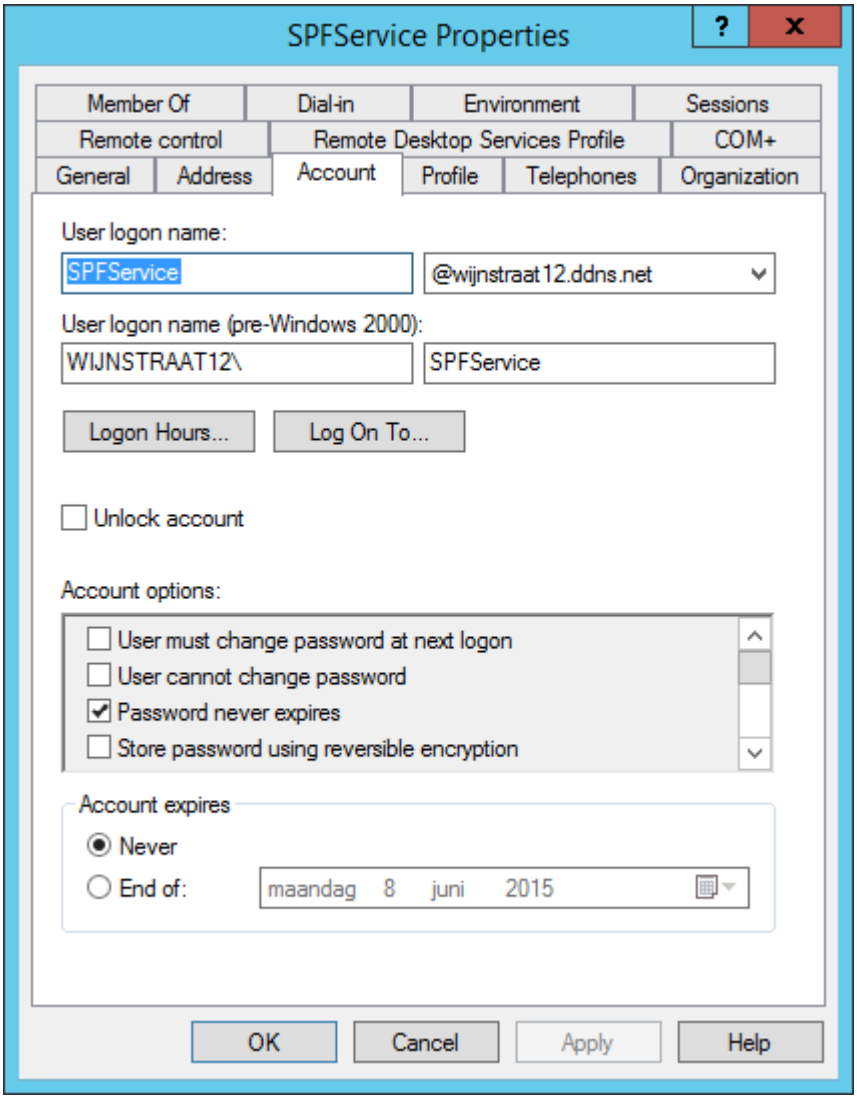

Figure 4.44: The properties of the domain account `SPFService`.

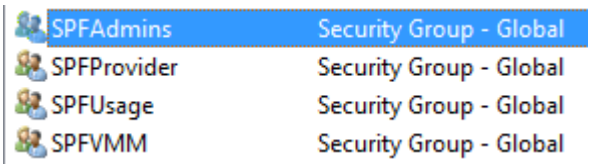

Figure 4.45: The four global security groups have been made.

Communications to and from the webserver should be secured / encrypted by SSL, which requires certificates. Two types of certificates exist: self-signed certificates and certificartes issued by a standalone certification authority.
In our case, the Windows Azure Pack will be installed on the same domain as the Service Provider Foundation and thus we are not required to use a public certificate issued by a standalone certification authority. So self-singed certificates will be used.
Installing such a certificate is done by using IIS Manager. Open it and select the webserver in the `Connections` pane. Next, click `Server Certificates` in the main window. On the `Actions` pane, click `Create Self-Singed Certificate`.
Specify the common name. Keep in mind that the common name must match the URL that is used when connecting to the Service Provider Foundation.

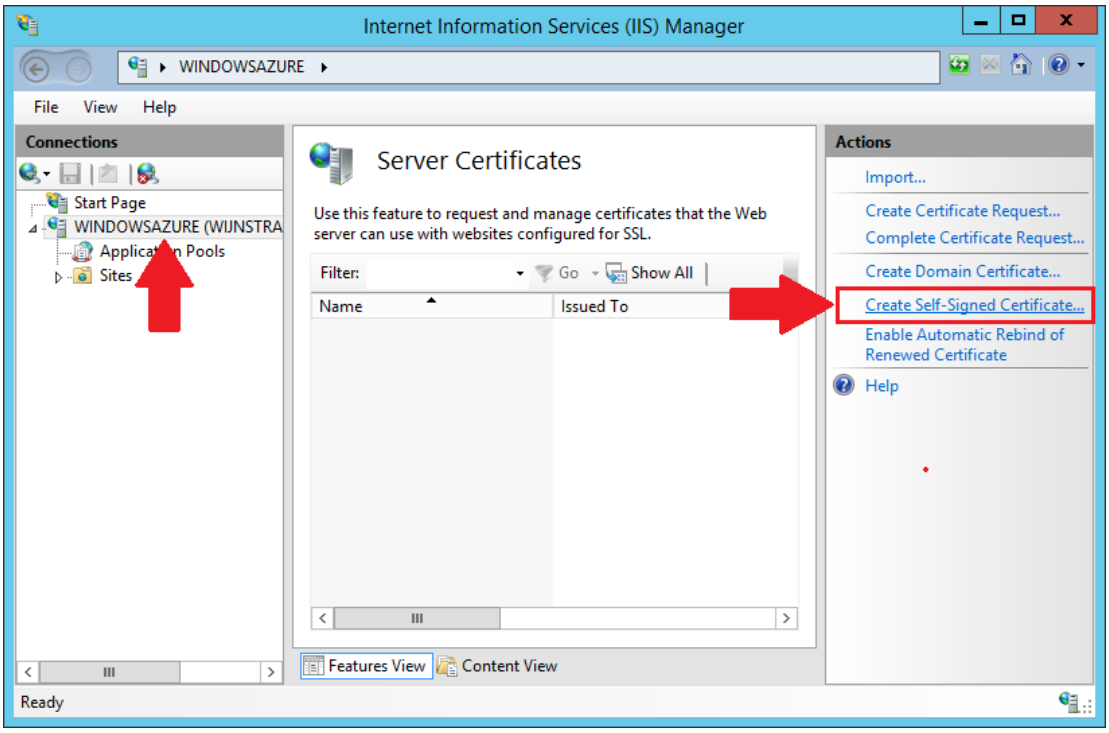

Figure 4.46: Create a self-singed certificate.

Now the installation of Service Provider Foundation can be started. The SPF can be found as a standalone installation on the System Center Orchestrator installation media. Extract the files of System Center Orchestrator, double click on `setup.exe` and select `Service Provider Foundation`.

Figure 4.47: SPF can be found on the SC Orchestrator installation media.

The installer will check if al the prerequisites have been met.

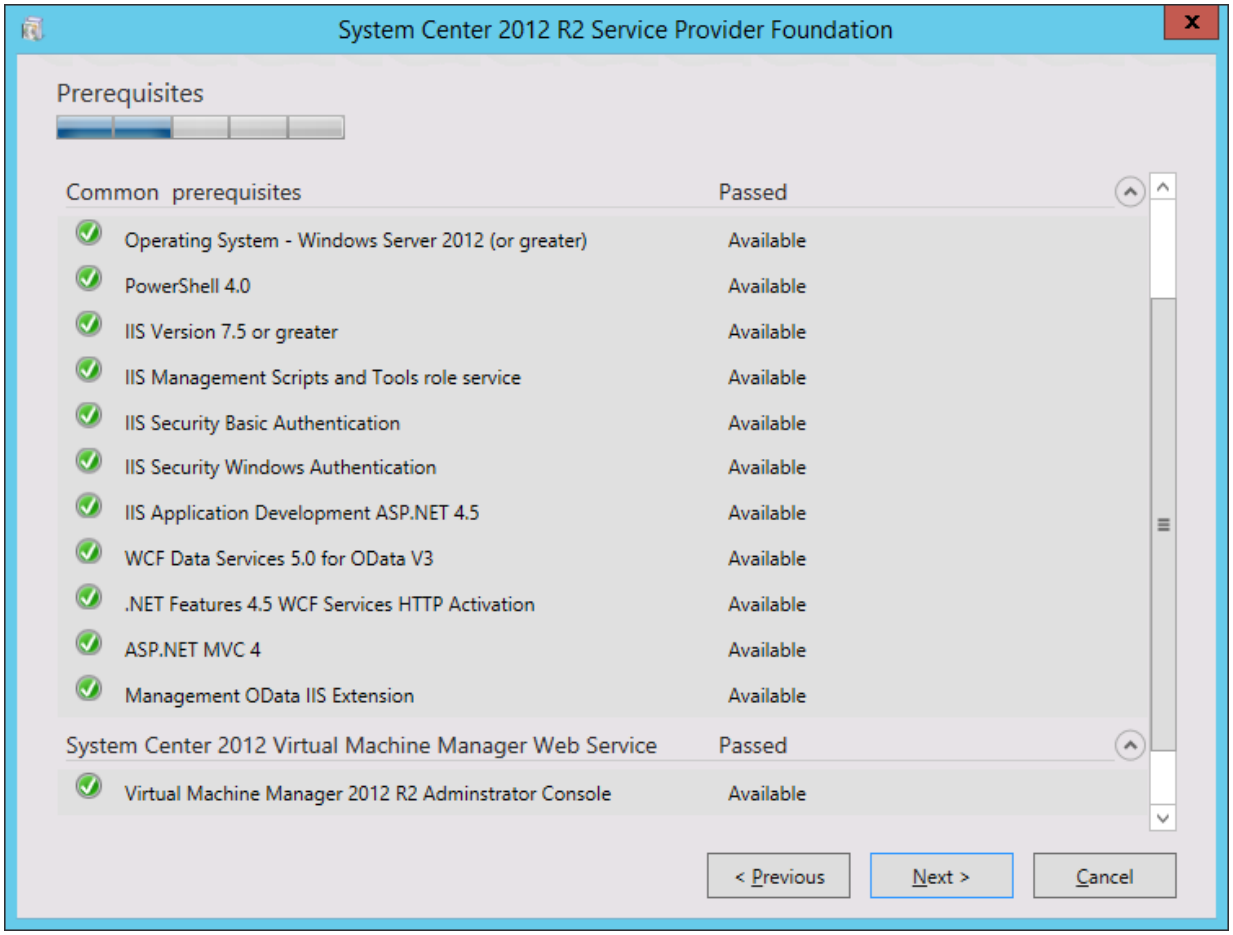

Figure 4.48: The prerequisites are checked. When everything is installed, setup can be continued.

Next, the database server location where the SPF database will be created, has to be configured. Either use the newly created, local SQL instance or use the SQL Server created earlier when setting up the VMM. In both cases, make sure the firewall allows both incoming and outgoing traffic on port 1433.

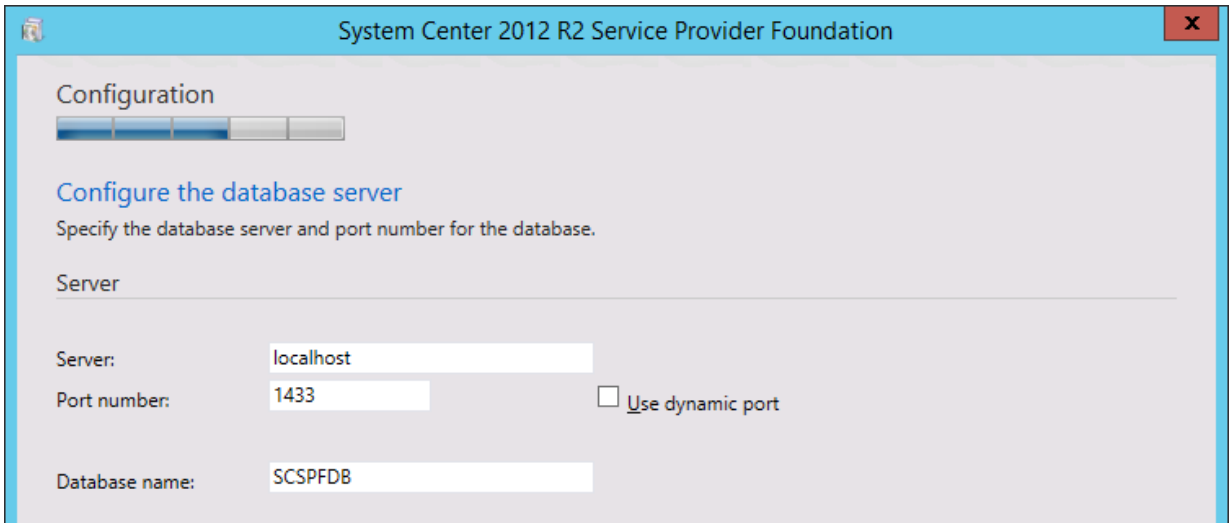

Figure 4.49: Configuration of the database. A newly created instance can be used, as well as the instance created earlier when setting up the VMM virtual machine.

Then the certificate for the web service has to be selected. Since we already created a certificate, select `Use existing certificate:` and select the correct certificate.

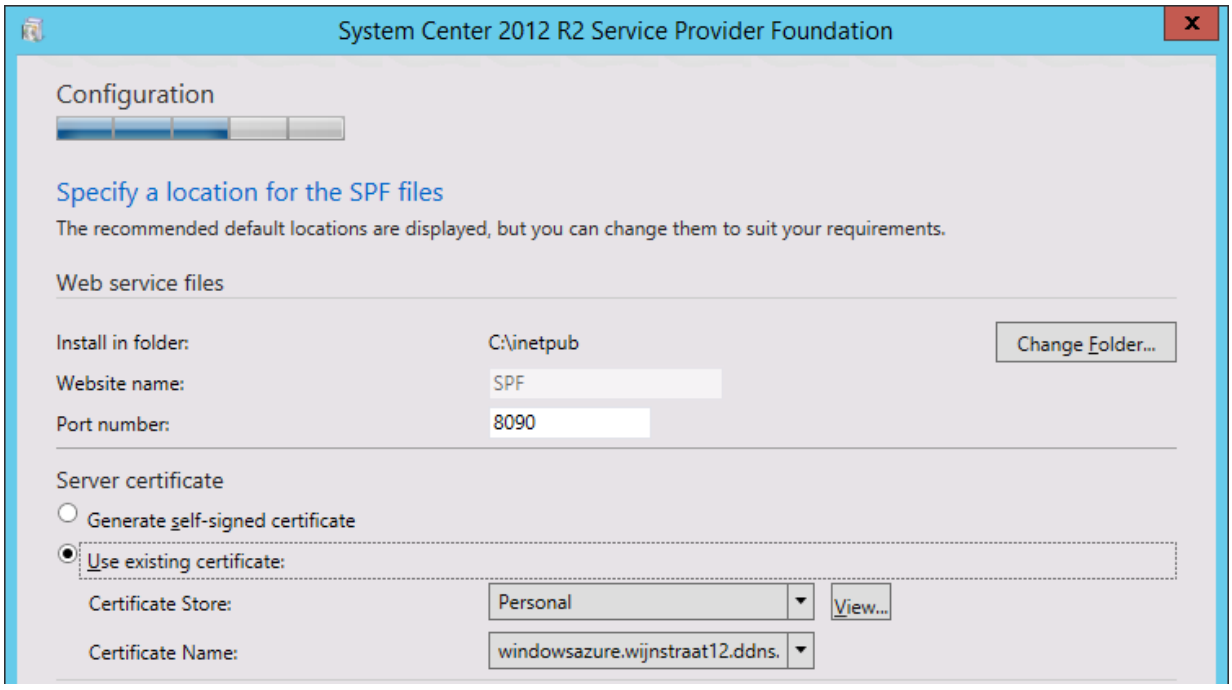

Figure 4.50: Certificate selection. Choose the self-singed certificate created earlier.

Now, the configuration of the virtual directories of IIS, permissions and App Pool Identities have to be configured. In the next four steps, the four domain groups created earlier need to be specified in each step. The service account `SPFService` is used in all steps.

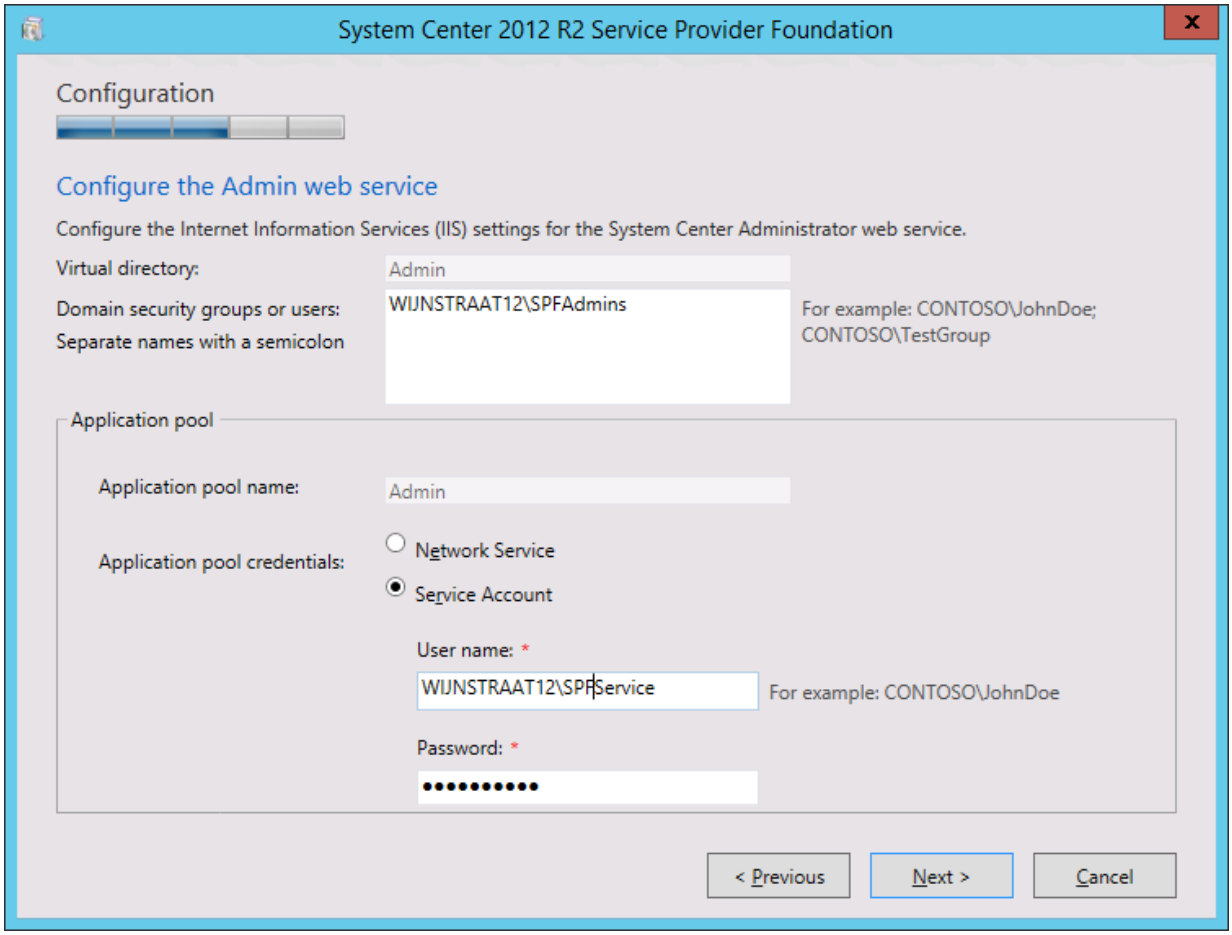

Figure 4.51: Configuration of the `Admin` virtual directory of IIS. Choose the `SPFAdmin` and the `SPFService` service account, both created earlier in the pre-setup process.

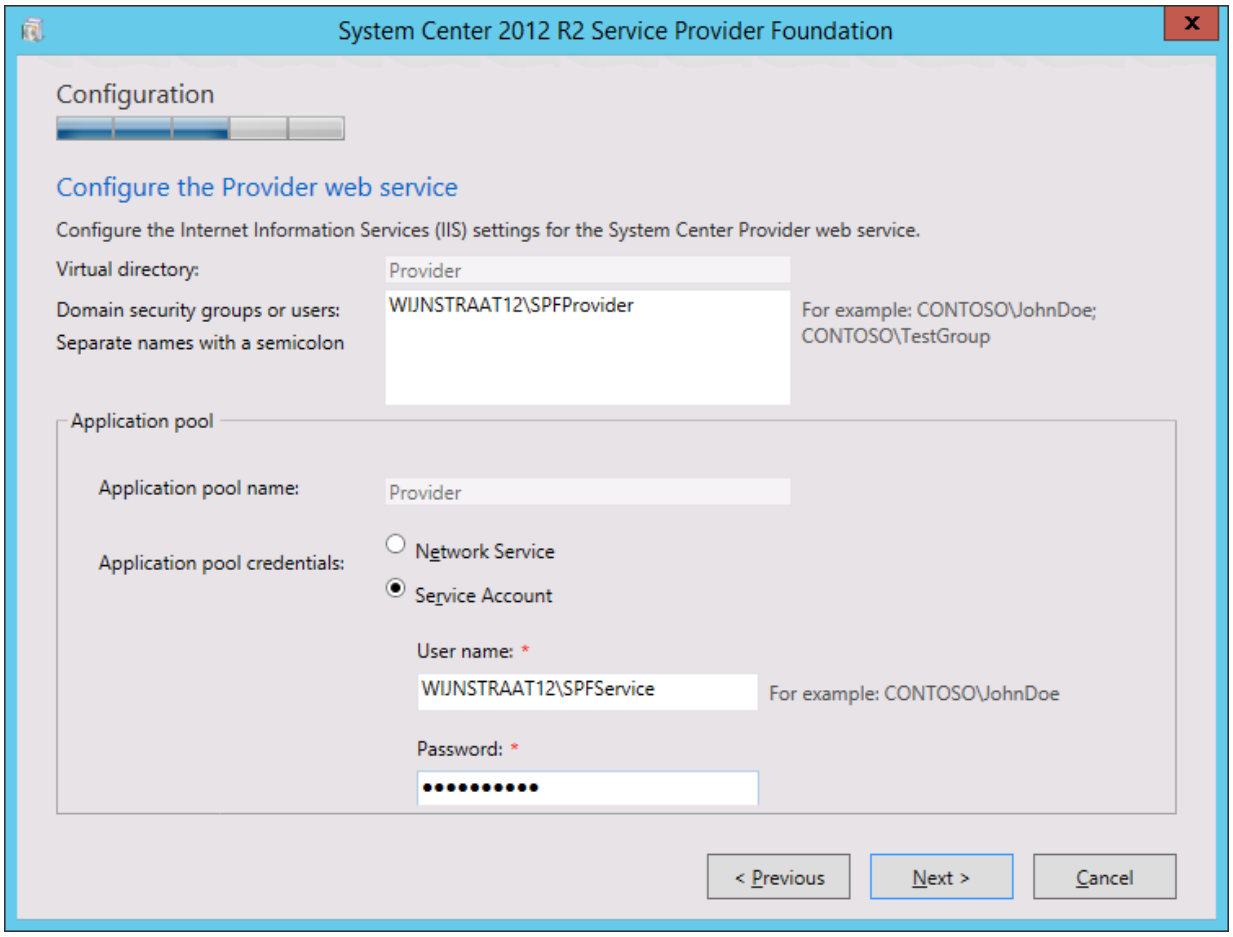

Figure 4.52: Configuration of the `Provider` virtual directory of IIS. Choose the `SPFProvider` and the `SPFService` service account, both created earlier in the pre-setup process.

Then the VMM web service and the Usage web service have to be specified. Use `SPFVMM` and `SPFUsage` respectively for this purposes. The service account remains the same: `SPFService`.

4.2.2 Post Installation of Service Provider Foundation

In order to use Windows Azure Pack correctly, some additional configuration must be performed. The `SPFService` account must be given additional permissions. It needs to be added as a member of the following **local** groups:

- SPF_Admin
- SPF_Provider
- SPF_Usage
- SPF_VMM

Double click on either one of them, click `Add...` and search for the `SPFService` account. Click `OK`. Repeat this process for the other groups.

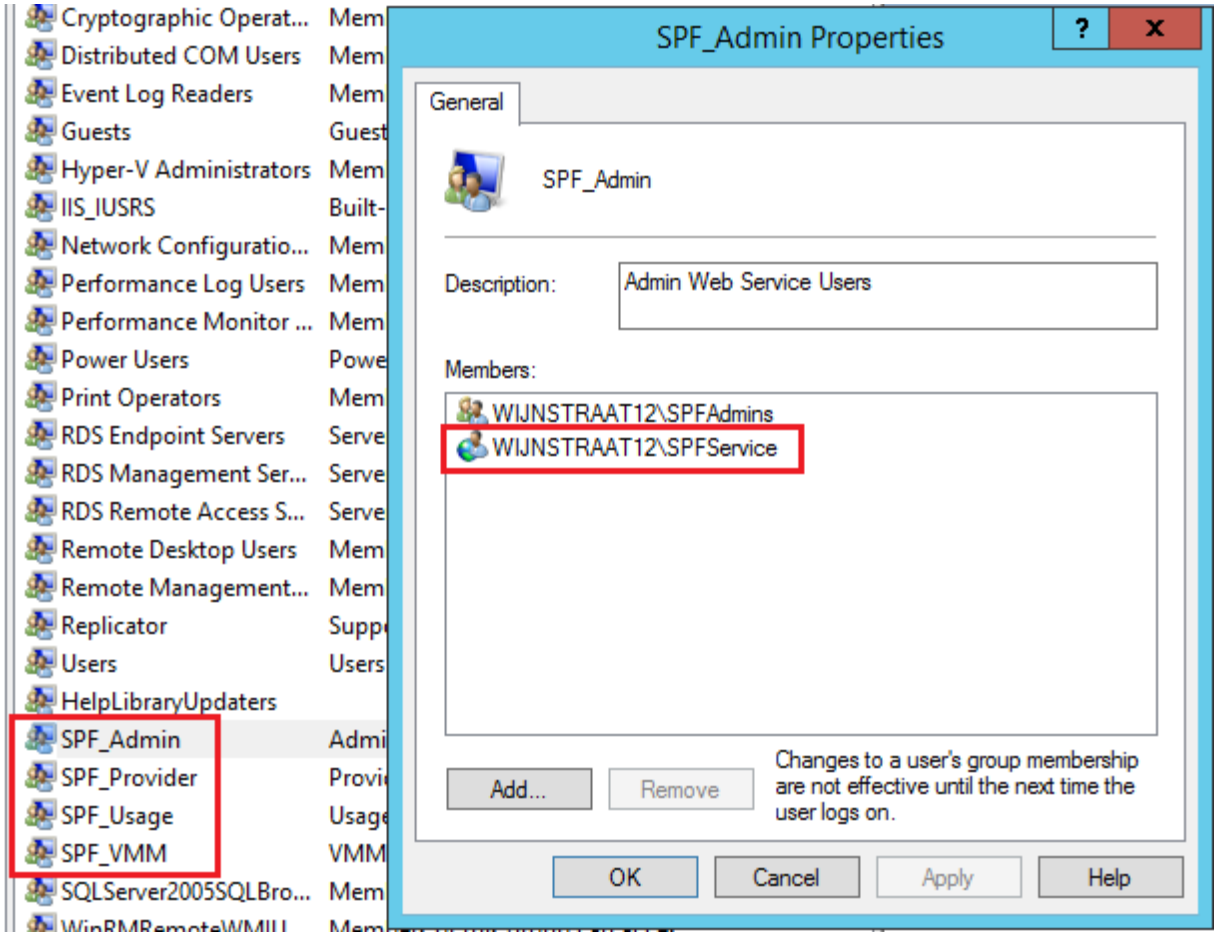

Figure 4.53: Add the `SPFService` account to the local groups created by the Service Provider Foundation installation.

The `SPFService` account also needs to be added to the Administrator user role in VMM. To do so, open VMM (either the console installed on this VM or on the VMM virtual machine itself). Click on `Settings` in the bottom-left corner and click on `Security` → `User Roles`. Double click on the `Administrator` user role and on the `Members` tab, click `Add...` to add the `SPFService` account to the user role.

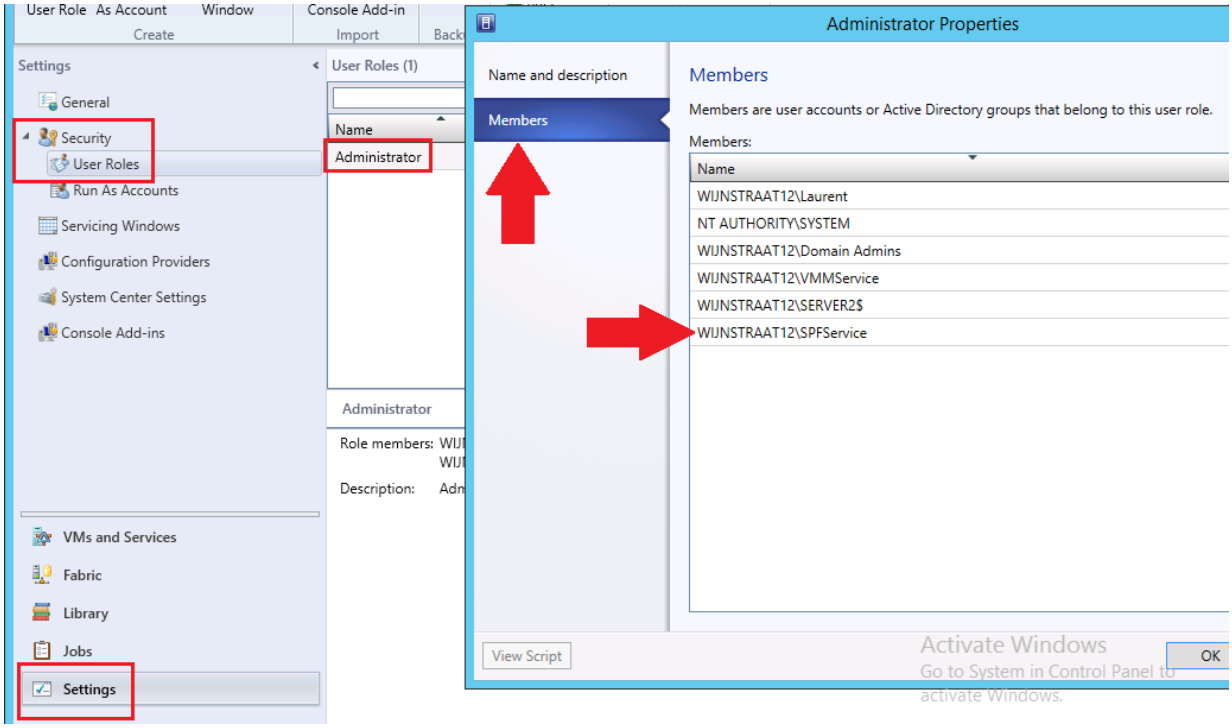

Figure 4.54: The `SPFService` account needs administrative permissions in Virtual Machine Manager.

The `SPFService` account needs permissions in the SQL Server as well. To set those permissions, open SQL Server Management Studio and navigate to `Security` → `Logins` and double click on `WIJNSTRAAT12\SPFService`. Select the `Server Roles` tab and check `sysadmin`.

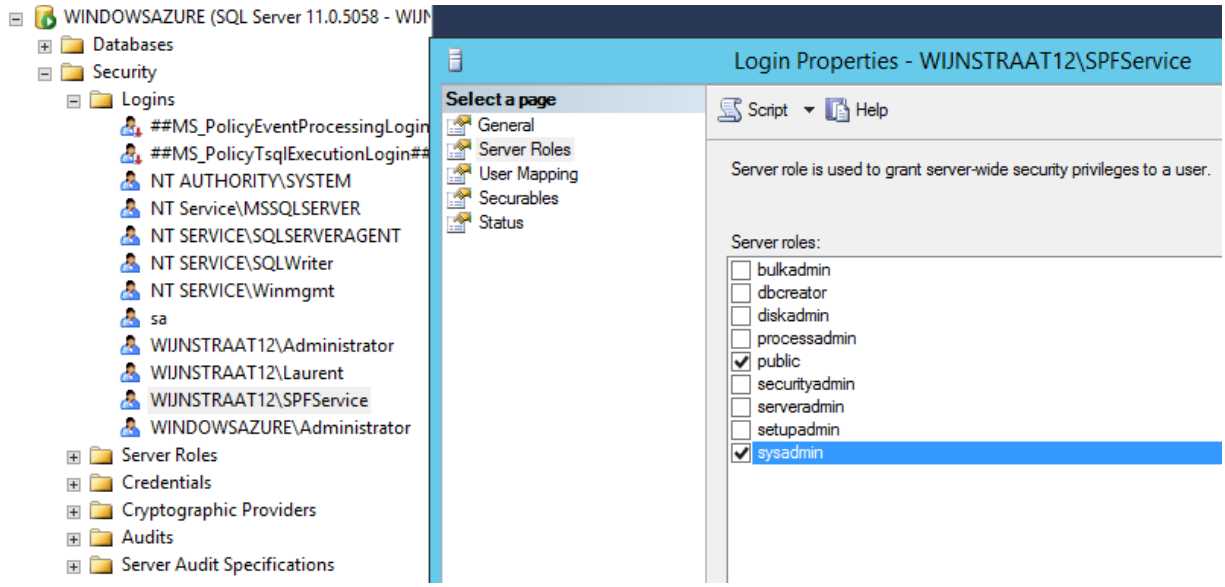

Figure 4.55: Adding administrative permissions to the `SPFService` account.

To verify the correct Application Pool settings of the web service, make sure that the identity of the Application Pools `Admin`, `Provider`, `Usage` and `VMM` is set to `WIJNSTRAAT12\SPFService`. To do so, open IIS Manager and select the `WINDOWSAZURE` web server. In the `Actions` pane on the right, click on `View Application Pools`.

Name	Status	.NET CLR V...	Managed Pipel...	Identity
.NET v4.5	Started	v4.0	Integrated	ApplicationPoolIdentity
.NET v4.5 Classic	Started	v4.0	Classic	ApplicationPoolIdentity
Admin	Started	v4.0	Integrated	WIJNSTRAAT12\SPFService
DefaultAppPool	Started	v4.0	Integrated	ApplicationPoolIdentity
Provider	Started	v4.0	Integrated	WIJNSTRAAT12\SPFService
Usage	Started	v4.0	Integrated	WIJNSTRAAT12\SPFService
VMM	Started	v4.0	Integrated	WIJNSTRAAT12\SPFService

Figure 4.56: Verifying the correct settings of the Application Pools.

To register the Service Provider Foundation in Windows Azure Pack, a **local** account has to be made on the same computer on where WAP will be installed. In our case, this is the current virtual machine we are working on.
Therefore, create a **local** user account named `SPF_REG` and make it a member of the following groups:

- SPF_Admin
- SPF_Provider
- SPF_VMM
- SPF_Usage

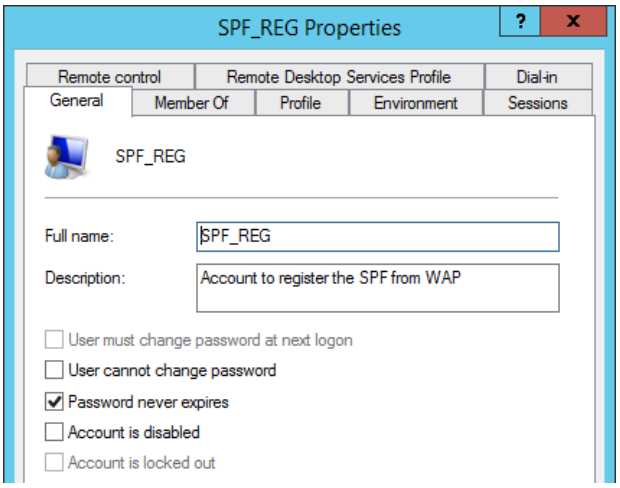

Figure 4.57: Creation of the local `SPF_REG` account.

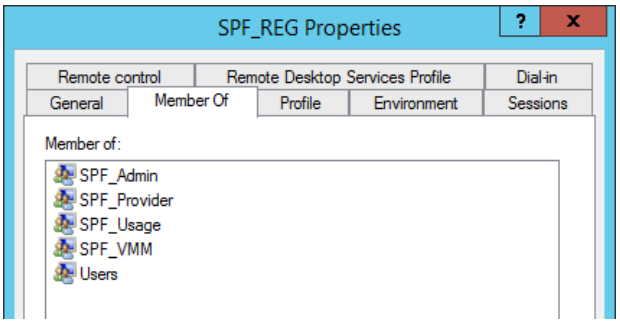

Figure 4.58: The account has been added to the four local groups created by the SPF installer.

4.2.3 Installation of the Windows Azure Pack

Two possibilities exist to install WAP: single server installation or distributed installation. The single server installation has been chosen, so this manual will focus on the installation of all the components of WAP on one server.

To install the Windows Azure Pack on a single server, download and run the Web Platform Installer 5.0, which can be downloaded from the Microsoft website: `http://www.microsoft.com/web/downloads/platform.aspx`.

On the `Products` tab, on the left pane select `Windows Azure` and add `Windows Azure Pack: Portal and API Express` with all its dependencies. Click `Next`.

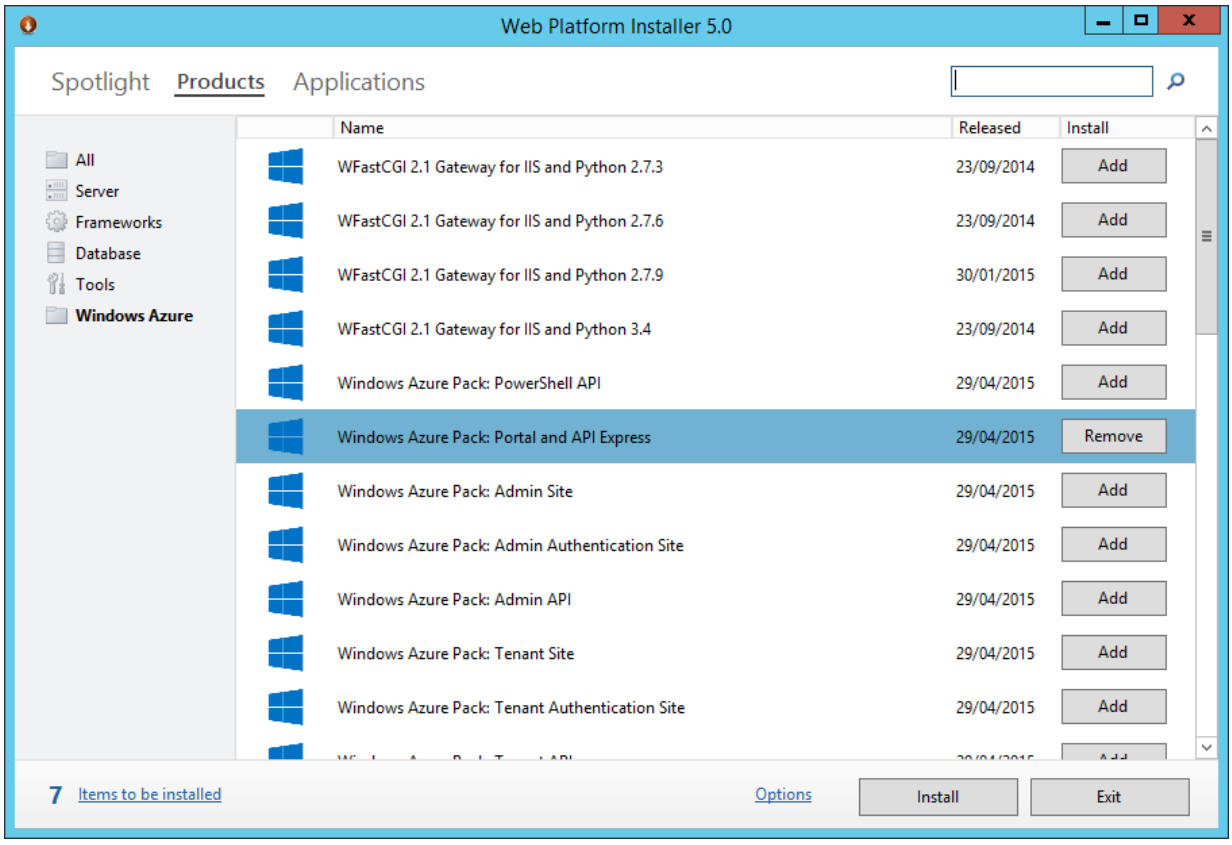

Figure 4.59: Selection of the WAP Portal and API Express.

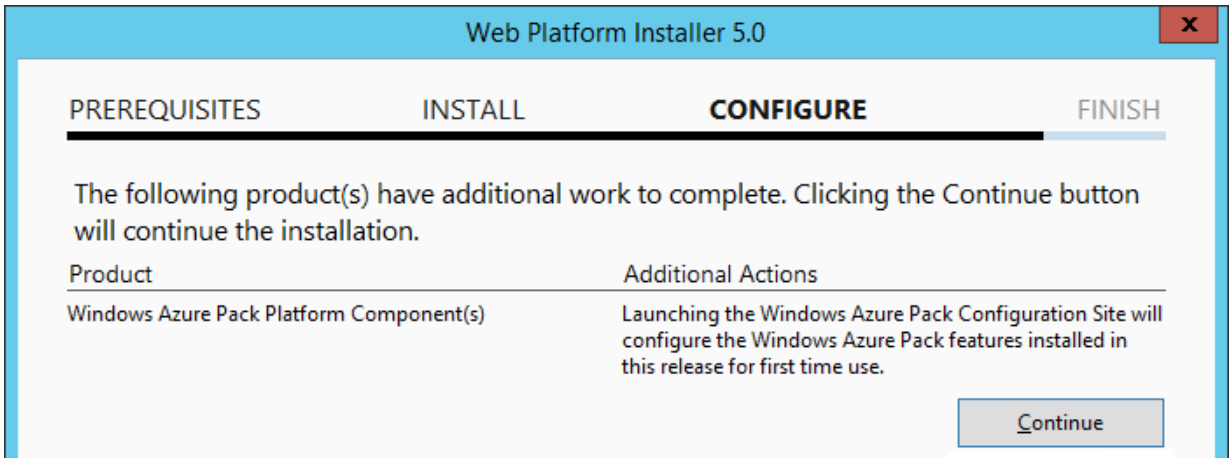

Figure 4.60: Click `Next`.

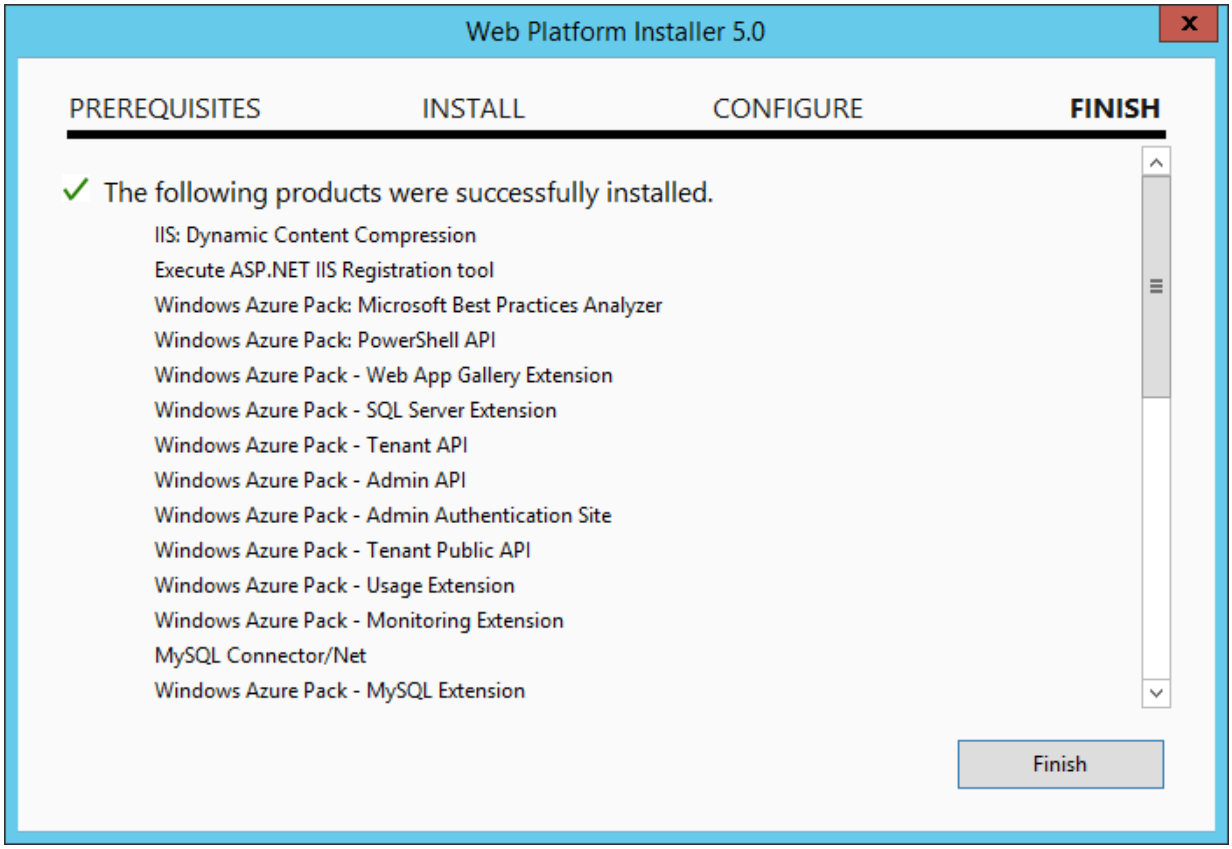

Figure 4.61: The installation has completed successfully. Click `Finish`.

After the installation has finished, go to `https://localhost:30101` from where the setup will be continued.
The first step is configuring the database connection which WAP will use. Fill in the appropriate server name, for example `localhost` or the name of the virtual machine that runs VMM and thus also runs an instance of SQL Server.
Choose `SQL Server Authentication` and let the database server admin name be `sa`. Fill in the password.
Then, choose a catchphrase. The use of a catchphrase requires SQL Authentication. When using an existing SQL Server instance using Windows Authentication (for example, the one configured on the VMM), setting up a catchphrase will not work. The security authentication mode needs to be changed to SQL Server and Windows Authentication mode. A guide to do so, can be found here: `https://technet.microsoft.com/en-us/library/ms188670.aspx`.
When a local instance has been created and Mixed Authentication Mode has been selected during installation, further configuration is not required.

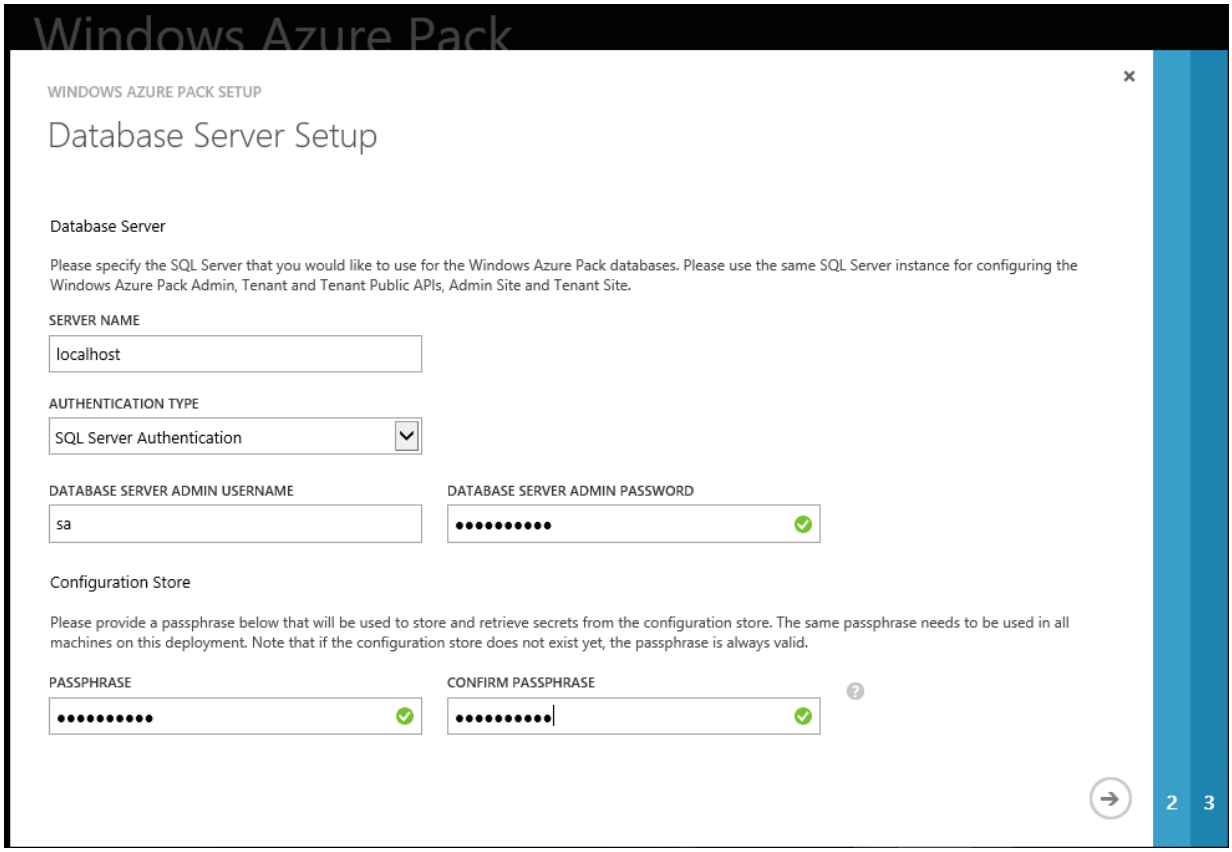

Figure 4.62: Configuration of the database access for use with WAP.

Choose whether or not you want to participate to the Customer Experience Improvement Program and go to the final page (3).
The `Features Setup` page shows which features will be configured. When clicking on the checkmark at the right bottom of the page, the configuration will start. When everything is configured successfully, green checkmarks will appear in before the configuration role as illustrated in the figure below.

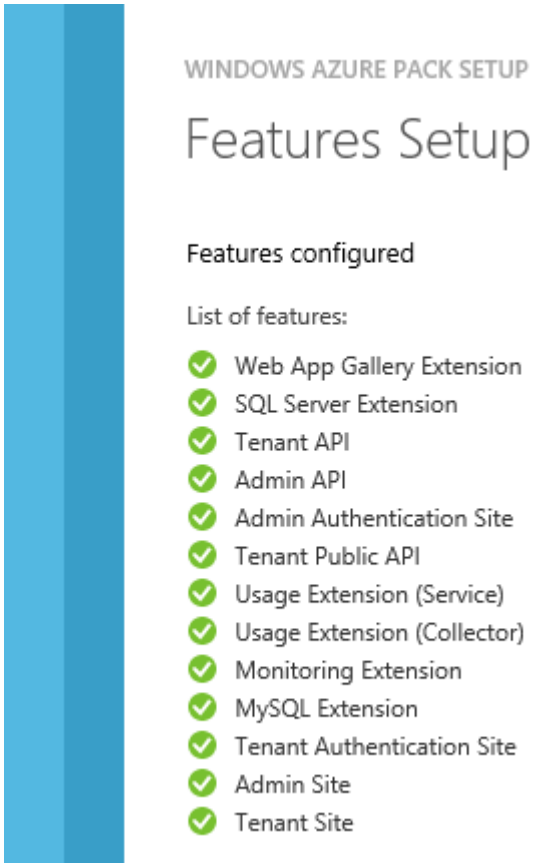

Figure 4.63: Setup has completed successfully.

The installation and configuration of WAP has been completed successfully! Now we have to login into WAP and configure a cloud.
Therefore, go to `https://localhost:30091` and log in using the **local** administrator account. To force Windows using the the local account, use a "`.\`" before the account name. An example would be: `.\Administrator`.

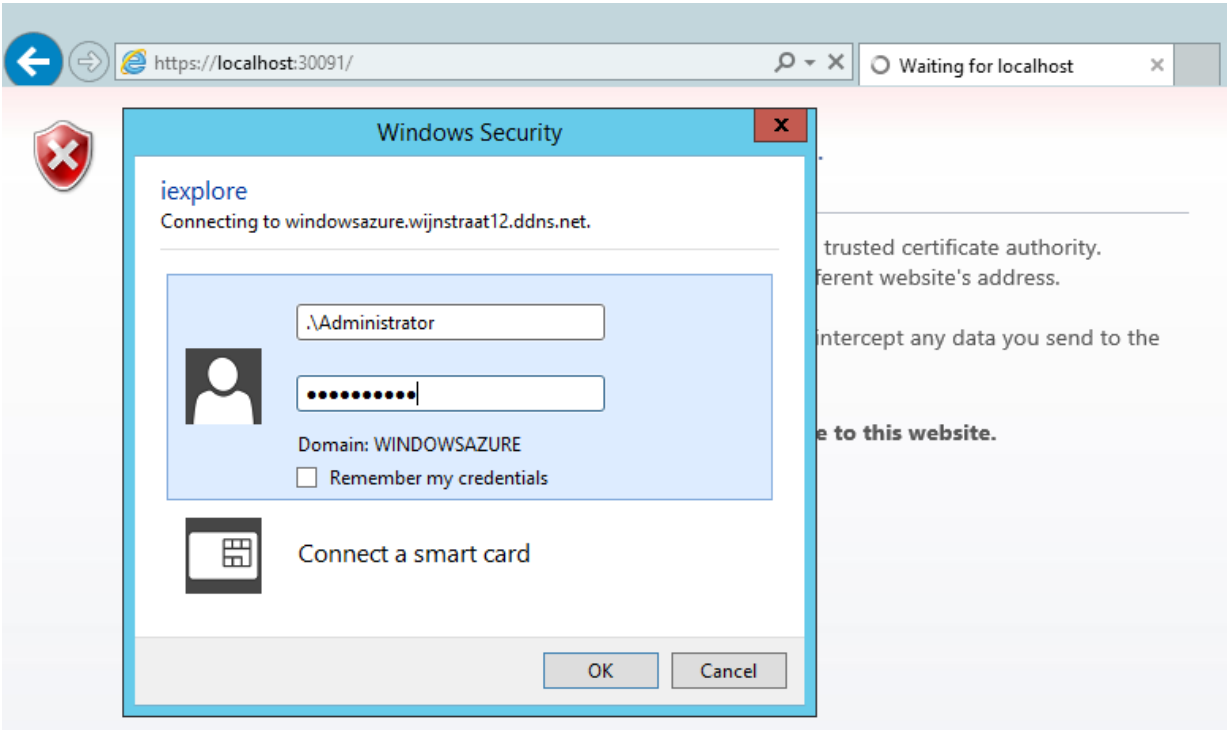

Figure 4.64: Login using the local administrator account.

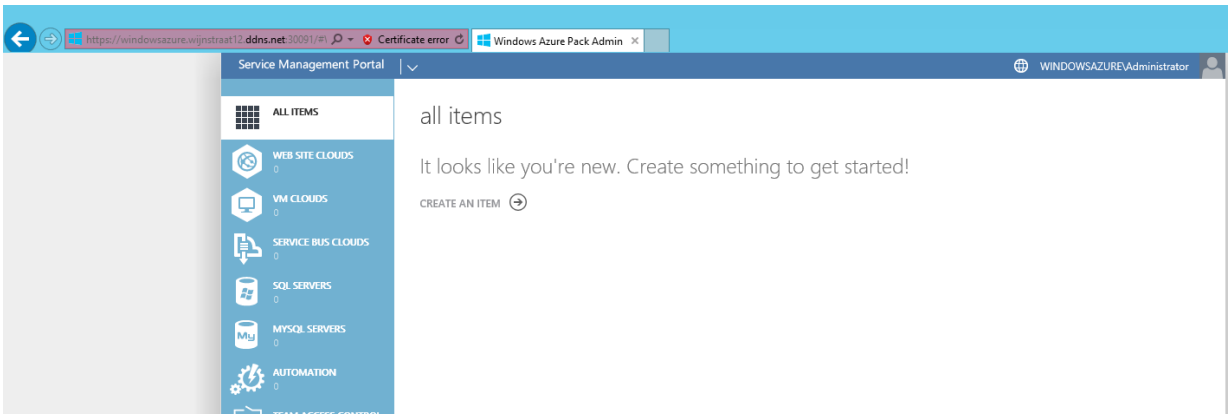

Figure 4.65: The main screen of WAP.

Now, the configuration of a VM cloud can be started. Select `VM Clouds` in the left pane. First, the Service Provider Foundation Endpoint must be registered using the local account `SPF_REG` created earlier.
However, in the past I received error messages saying that the registration of the SPF Endpoint could not be completed. The solution is as follows: reset the password of the `SPF_REG` account using the `Local Users and Groups` snap-in. Right click on the `SPF_REG` account and choose `Set password...`.
After that, the IIS Web Server needs to be reset. To do so, open PowerShell and execute following command: `iisreset.exe`.

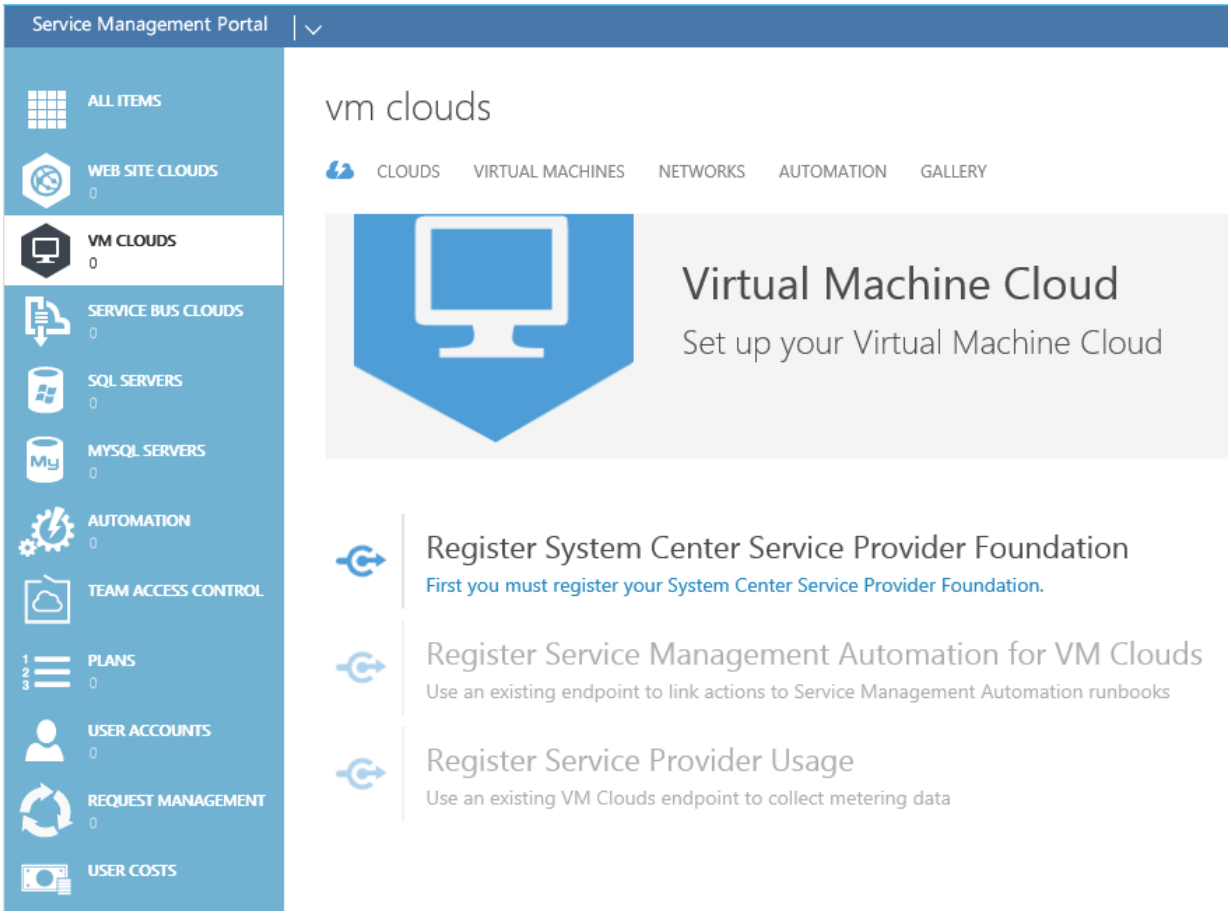

Figure 4.66: Before setting up a VM cloud, the SPF Endpoint needs to be registered.

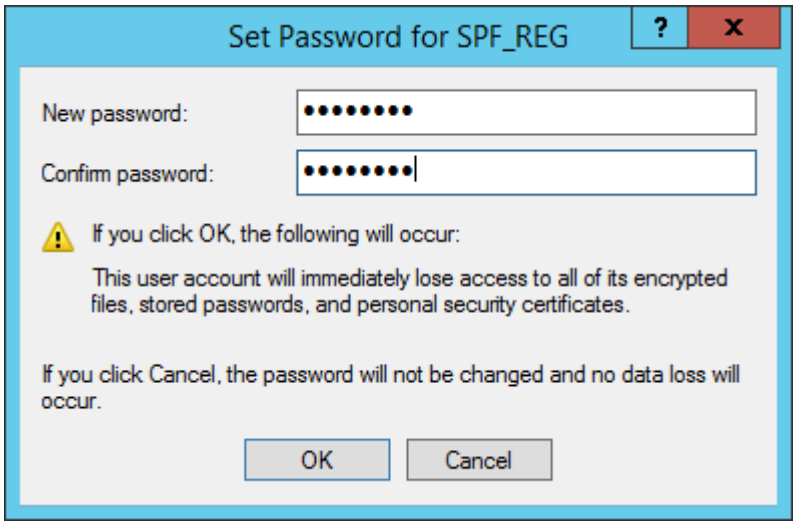

Figure 4.67: In case of failure, reset the password for the `SPF_REG` account.

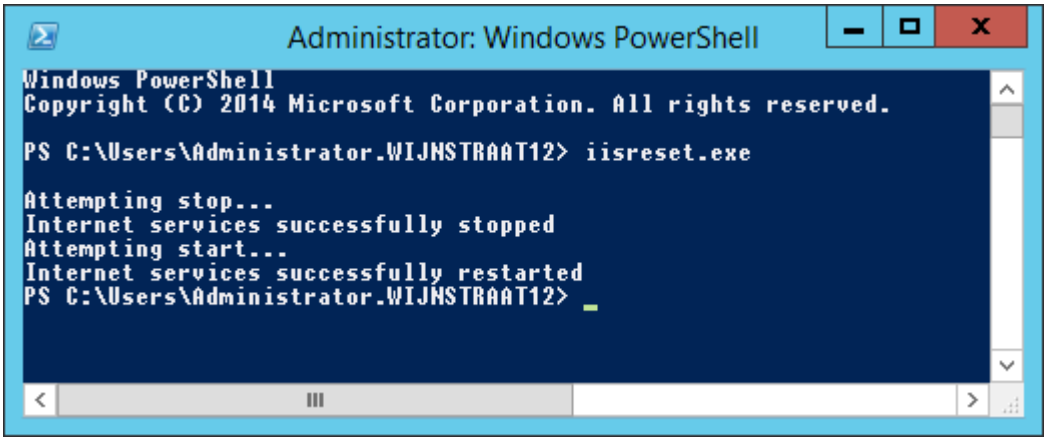

Figure 4.68: Restart the Web Server ...

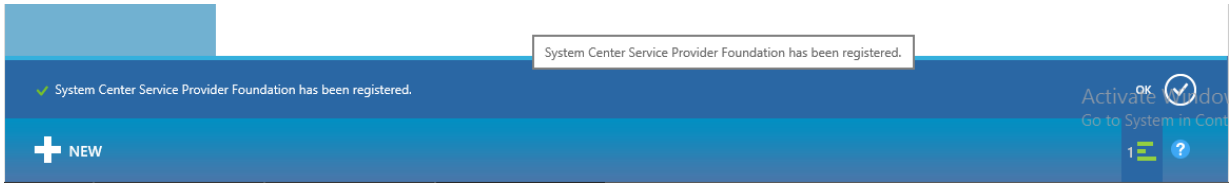

Figure 4.69: ... and everything should work fine now.

To manage a VM cloud, we must first connect to the VMM server. The existing cloud in VMM will be displayed in the `VM Clouds`.
Therefore, fill in the Virtual Machine Manager server FQDN: `vmmtest.wijnstraat12.ddns.net`.

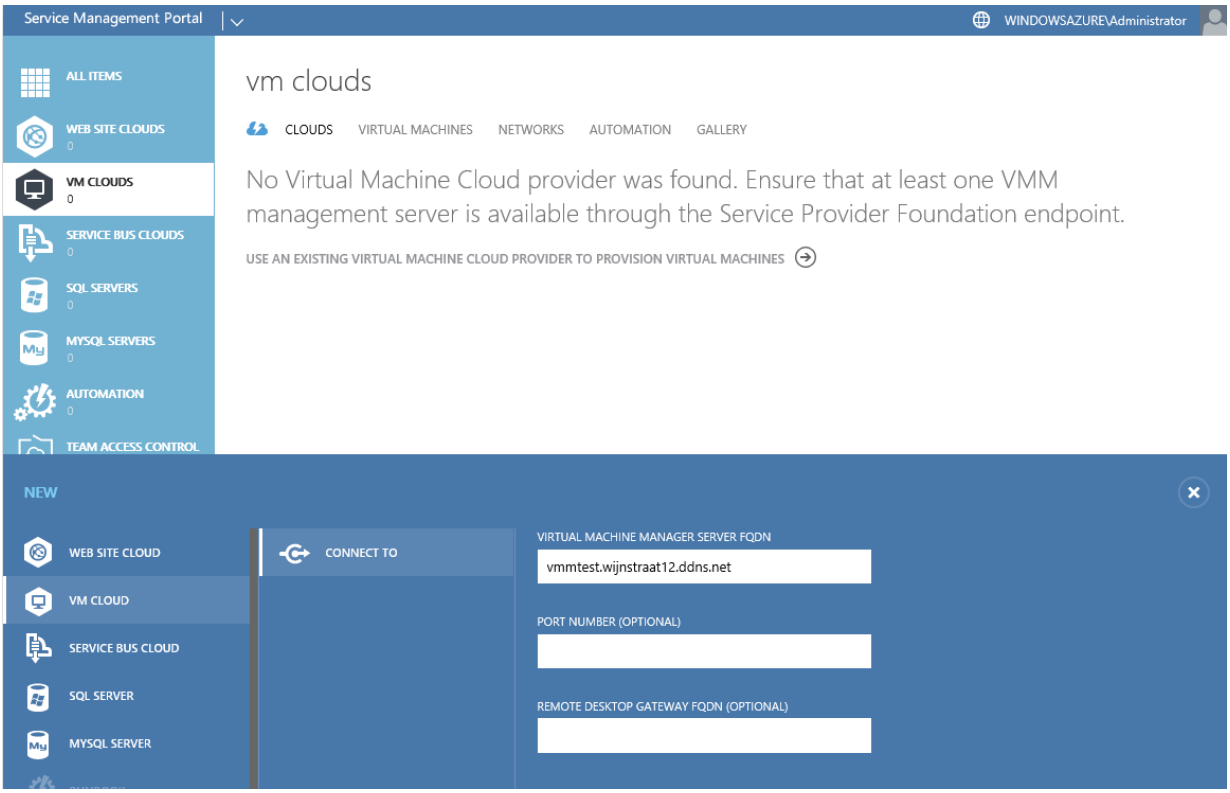

Figure 4.70: Connect to the VMM server.

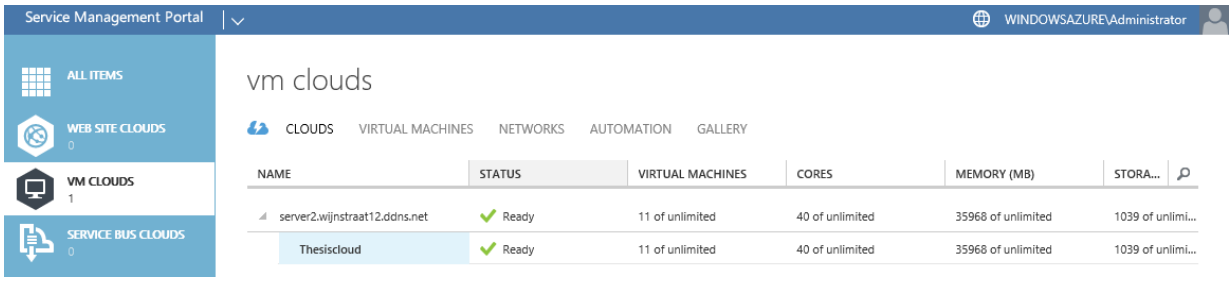

Figure 4.71: The cloud is now visible in `VM clouds`.

Chapter 5

Conclusions and recommendations

Here comes the conclusion.

Appendix A

Test lab

This appendix describes and visualizes the test lab that was used to perform the tests.

A.1 General description

In order to perform the required tests, a test lab is of course needed. The lab is an expansion of a production environment used in a Small Office Home Office (SOHO). The existing hardware equipment consists of a Belgacom router, a central switch, a physical tower server running the Xen hypervisor and some connections leading to the clients, other switches and network printers. Figure A.1 provides the reader with a visual representation of the network.

This network has been expanded with a 1U rack server and another tower server. On the 1U rack server and the tower server, Microsoft Windows Server 2012 R2 was installed as well as the Hyper-V role. So both servers run the Hyper-V hypervisor. In fact, due to [Osborne, 2013] and [Microsoft, 2011], it is recommended not to install any other roles besides the Hyper-V role. This is because the hypervisor partition is placed in between the parent partition and the hardware. Also, it will keep the management system (the Hyper-V host) clean as no updates are required but for the Hyper-V role.

Thus there exist three virtual networks: a Xen - based virtual network and two Hyper-V based networks. The reason for combining two different virtual networks is that prior to starting this master thesis, I had quite a lot of experience with the Xen hypervisor, but almost none with Microsoft's counterpart.

Another reason for the mixed environment is that I wanted to check whether the two different virtual networks are able to cooperate with each other as they should.

A.2 Technical description

A.2.1 Hardware overview

SSH is used to remotely access the virtual machines running on the Xen hypervisor, wheras Remote Desktop Connection (RDP) is used to connect to the Hyper-V hypervisor as well as to access the dual-boot server.

Since only one external IP address is available, different ports were used for RDP, ranging from 3389 to 3395.
The Xen host, the first Hyper-V host and second Hyper-V host run 4, 5 and 3 VM's, respectively.
Below is a table specifying the hardware of the different servers used in the test lab.

System	Xen server	Hyper-V server	Dual-boot server
Processor brand	AMD	Intel	Intel
Processor type	Athlon II x2 240e	Core 2 Quad Q9550	Core i7 930
Core speed	2.8 GHz	2.83 GHz	2.80 GHz
Number of cores	2	4	4
Number of sockets	1	1	1
Number of virtual cores	2	4	8
Memory amount and type	8 GB DDR3	8 GB DDR2	8 / 16 GB DDR3
# Harddisk drives	4	4	3
RAID type	LVM	5	5
Hardware RAID?	NO	YES	YES
Ethernet speed	1 Gbps	1 Gbps	1 Gpbs

Table A.1: Overview of the hardware used in the test lab

A.2.2 Networking overview

The table below provides an overview of the IP addressing scheme.

Device	**IP address**
Belgacom router	192.168.1.1
Xen host	192.168.1.2
Hyper-V host 1 (management)	192.168.1.7
Hyper-V host 1 (external switch)	192.168.1.6
Hyper-V host 2 (external switch)	192.168.1.8
Hyper-V VM 1	192.168.1.50
Hyper-V VM 2	192.168.1.51
Hyper-V VM 3	192.168.1.12
Hyper-V domain controller VM	192.168.1.150
Hyper-V backup domain controller VM	192.168.1.152
Hyper-V Virtual Machine Manager VM	192.168.1.151
Hyper-V Windows Azure Pack VM	192.168.1.160
Xen VM 1	192.168.1.11
Xen VM 2	192.168.1.14
Xen VM 3	192.168.1.16

A.3 Visualization of the test lab

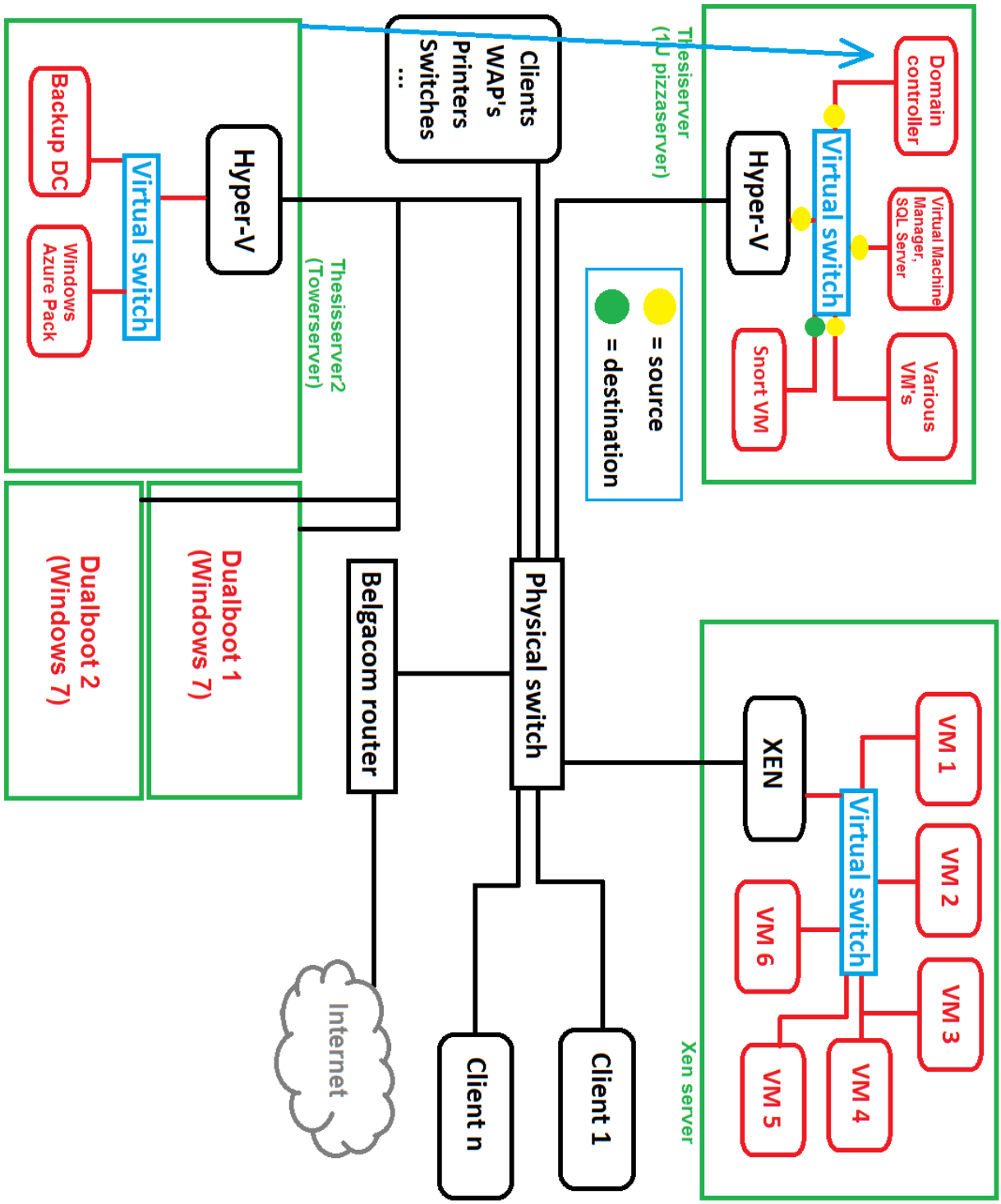

Figure A.1: The network setup

Bibliography

AMD (2009). *Advanced Micro Devices, Inc. AMD I/O Virtualization Technology (IOMMU) Specification License Agreement*. AMD. Retrieved on March 7, 2015.

Avis, C. E. (2013). Vmware or microsoft: Protecting vm's - agents or agentless? *http://blogs.technet.com/b/chrisavis/archive/2013/08/22/vmware-or-microsoft-protecting-vm-s-agents-or-agentless.aspx*. Retrieved on March 8, 2015.

Beal, V. (2015). The 7 layers of the osi model. *http://www.webopedia.com/quick_ref/OSI_Layers.asp*. Retrieved on February 24, 2015.

Ben-Yehuda, M. (2013). Machine virtualization, efficient hypervisors and stealthy malware. *http://www.mulix.org/lectures/vmsecurity/vmsec-cyberday13.pdf*. Retrieved on March 8, 2015.

Bigelow, S. J. (2009). Understanding the befefits of a virtual machine. *http://searchservervirtualization.techtarget.com/tip/Understanding-the-benefits-of-a-virtual-machine*. Retrieved on February 25, 2015.

Briscoe, N. (2008). Understanding the osi 7-layer model. *http://memberfiles.freewebs.com/61/55/58745561/documents/OSI.pdf*. Retrieved on February 24, 2015.

Burger, T. (2012). IntelÂ® virtualization technology for directed i/o (vt-d): Enhancing intel platforms for efficient virtualization of i/o devices. *https://software.intel.com/en-us/articles/intel-virtualization-technology-for-directed-io-vt-d-enhancing-intel-platforms-for-efficient-virtualization-of-io-devices*. Retrieved on March 7, 2015.

Cisco (2015a). Preprocessors. *http://manual.snort.org/node17.html*. Retrieved on January 23, 2015.

Cisco (2015b). Rules headers. *http://manual.snort.org/node29.html*. Retrieved on January 23, 2015.

Cisco (2015c). Snort license. *https://www.snort.org/license1*. Retrieved on January 23, 2015.

Cummin (2010). Pulledpork - what is it about? *https://code.google.com/p/pulledpork/wiki/PulledPork*. Retrieved on January 23, 2015.

Firnsy (2010). Barnyard2. *https://github.com/firnsy/barnyard2*. Retrieved on January 23, 2015.

Garrison, J. (2011). What is logical volume management and how do you enable it in ubuntu? *http://www.howtogeek.com/howto/36568/what-is-logical-volume-management-and-how-do-you-enable-it-in-ubuntu/*. Retrieved on March 3, 2015.

Grehl, F. (2015). Vmware esxi release and build number history. *http://www.virten.net/vmware/esxi-release-build-number-history/#esxi3.5*. Retrieved on February 27, 2015.

Halliday, P. (2012). the squertproject. *http://www.squertproject.org/*. Retrieved on January 23, 2015.

House, M. (2006). Hypervisor. *http://searchservervirtualization.techtarget.com/definition/hypervisor*. Retrieved on February 26, 2015.

Howard, J. (2008). Hyper-v: What are the uses for different types of virtual networks? *http://blogs.technet.com/b/jhoward/archive/2008/06/17/hyper-v-what-are-the-uses-for-different-types-of-virtual-networks.aspx*. Retrieved on March 2, 2015.

Intel (2015). Intel virtualization technology (intel vt). *http://www.intel.com/content/www/us/en/virtualization/virtualization-technology/intel-virtualization-technology.html*. Retrieved on March 7, 2015.

Jackson, I. (2012). Xen networking. *http://wiki.xenproject.org/wiki/Xen_Networking#Bridging*. Retrieved on March 2, 2015.

Jeong, S. (2013). In-depth overview of x86 server virtualization technology. *http://www.cubrid.org/blog/dev-platform/x86-server-virtualization-technology/*. Retrieved on March 7, 2015.

Kassner, M. (2008). 10+ things you should know about rootkits. *http://www.techrepublic.com/blog/10-things/10-plus-things-you-should-know-about-rootkits*. Retrieved on March 8, 2015.

Kennedy, P. (2011). Hyper-v networking and virtual switches overview. *http://www.servethehome.com/hyperv-networking-virtual-switches-overview/*. Retrieved on March 2, 2015.

King, S. T., Chen, P. M., Wang, Y.-M., Verbowski, C., Wang, H. J., and Lorch, J. R. (2006). Subvirt: Implementing malware with virtual machines. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 314–327, Oakland, CA. Institute of Electrical and Electronics Engineers, Inc.

Kleyman, B. (2012). Hypervisor 101: Understanding the virtualization market. *http://www.datacenterknowledge.com/archives/2012/08/01/hypervisor-101-a-look-hypervisor-market/*. Retrieved on February 26, 2015.

Koziol, J. (2003). Dissecting snort - preprocessors. *http://www.informit.com/articles/article.aspx?p=101148&seqNum=2*. Retrieved on January 23, 2015.

Kurth, L. (2012). Choice of toolstacks. *http://wiki.xen.org/wiki/Choice_of_Toolstacks#Default_.2F_XL*. Retrieved on February 26, 2015.

Marcin (2011). How secure are virtual machines really? *http://security.stackexchange.com/questions/3056/how-secure-are-virtual-machines-really-false-sense-of-security*. Retrieved on February 26, 2015.

Maurer, T. (2014a). Windows azure for your datacenter. *http://www.thomasmaurer.ch/2014/01/windows-azure-for-your-datacenter/*. Retrieved on May 12, 2015.

Maurer, T. (2014b). Windows azure pack architecture. *http://www.thomasmaurer.ch/2014/03/windows-azure-pack-architecture/*. Retrieved on May 12, 2015.

Microsoft (2010). Overview of hyper-v. *https://technet.microsoft.com/en-us/library/cc816638%28WS.10%29.aspx*. Retrieved on February 27, 2015.

Microsoft (2011). Hyper-v: Hyper-v should be the only enabled role. *https://technet.microsoft.com/en-us/library/ee941145(v=ws.10).aspx*. Retrieved on April 27, 2015.

Microsoft (2015a). Advantages of hyper-v. *https://msdn.microsoft.com/en-US/library/cc768521%28v=bts.10%29.aspx*. Retrieved on February 27, 2015.

Microsoft (2015b). Virtuele machines uitvoeren onder windows 8.1 met client hyper-v. *http://windows.microsoft.com/nl-be/windows-8/hyper-v-run-virtual-machines*. Retrieved on February 27, 2015.

MSDN, M. (2012). Virtual disk. *https://msdn.microsoft.com/en-us/library/windows/desktop/dd323684%28v=vs.85%29.aspx*. Retrieved on March 3, 2015.

MSDN, M. (2015). Hyper-v architecture. *https://msdn.microsoft.com/en-us/library/cc768520%28v=bts.10%29.aspx*. Retrieved on February 27, 2015.

Northrop, E. (2013). Barnyard2. *http://www.forensicswiki.org/wiki/Barnyard2*. Retrieved on January 23, 2015.

Oracle (2015). *VirtualBox User Manual*, volume 6. Oracle.

Osborne, R. (2013). Windows server 2012 hyper-v best practices. *http://blogs.technet.com/b/askpfeplat/archive/2013/03/10/windows-server-2012-hyper-v-best-practices-in-easy-checklist-form.aspx/*. Retrieved on April 27, 2015.

Ott, D. (2009). Understanding vt-d: Intel virtualization technology for directed i/o. *https://software.intel.com/en-us/blogs/2009/06/25/understanding-vt-d-intel-virtualization-technology-for-directed-io/*. Retrieved on March 7, 2015.

Pavlicek, R. (2012). Understanding hosted and bare-metal virtualization hypervisor types. *http://wiki.xen.org/wiki/Xen_Overview*. Retrieved on February 26, 2015.

Prowse, D. L. (2014). *CompTIA Security+ SYO-201 Cert Guide*, volume 3. Prentice Hall, New Jersey.

Remde, K. (2012). Extend your hyper-v virtual switch in windows server 2012. *http://blogs.technet.com/b/kevinremde/archive/2012/10/20/31-days-of-our-favorite-things-extend-your-hyper-v-virtual-switch-in-windows-server-2012-part-20-of-31.aspx*. Retrieved on March 9, 2015.

Rutkowska, J. (2006). Introducing blue pill. *http://theinvisiblethings.blogspot.be/2006/06/introducing-blue-pill.html*. Retrieved on March 8, 2015.

Shinder, D. (2008). 10 things you should know about hyper-v. *http://www.techrepublic.com/blog/10-things/10-things-you-should-know-about-hyper-v/*. Retrieved on February 27, 2015.

Sid, D. B. (2014). Ossec - home. *http://www.ossec.net*. Retrieved on January 23, 2015.

Siebert, E. (2006). Understanding hosted and bare-metal virtualization hypervisor types. *http://searchservervirtualization.techtarget.com/tip/Understanding-hosted-and-bare-metal-virtualization-hypervisor-types*. Retrieved on February 26, 2015.

Siron, E. (2014). 7 keys to hyper-v security. *http://www.altaro.com/hyper-v/7-keys-to-hyper-v-security/*. Retrieved on March 9, 2015.

Smith, J. E. and Nair, R. (2008). The architecture of virtual machines. *http://www.ittc.ku.edu/∼kulkarni/teaching/archieve/EECS800-Spring-2008/smith_nair.pdf*. Retrieved on February 24, 2015.

Smyth, N. (2009). An overview of the hyper-v architecture. *http://www.virtuatopia.com/index.php/An_Overview_of_the_Hyper-V_Architecture*. Retrieved on February 27, 2015.

Soper, T. (2012). Getting to know hyper-v: A walkthrough from initial setup to common scenarios. *https://technet.microsoft.com/library/ee256064%28WS.10%29.aspx*. Retrieved on February 27, 2015.

Tanenbaum, A. S. and Austin, T. (2013). *Structured Computer Organization*, volume 6. Prentice Hall, New Jersey.

Technet, M. (2015a). Configure networking. *https://technet.microsoft.com/en-us/library/cc770380.aspx*. Retrieved on March 2, 2015.

Technet, M. (2015b). Configuring virtual networks. *https://technet.microsoft.com/nl-be/library/cc816585%28v=WS.10%29.aspx*. Retrieved on March 2, 2015.

Technet, M. (2015c). Virtual network security. *https://technet.microsoft.com/en-us/library/cc720393(v=ws.10).aspx*. Retrieved on March 9, 2015.

TLDP (2002). What is logical volume management? *http://www.tldp.org/HOWTO/LVM-HOWTO/whatisvolman.html*. Retrieved on March 3, 2015.

Vangie, B. (2007). Understanding hardware-assisted virtualization. *http://www.webopedia.com/DidYouKnow/Computer_Science/hardware_assisted_virtualization.asp*. Retrieved on March 7, 2015.

Vanover, R. (2013). Introduction to virtualization, abstraction is key. *http://www.techrepublic.com/blog/data-center/introduction-to-virtualization-abstraction-is-key/*. Retrieved on February 24, 2015.

Visscher, B. (2014). Sguil: The analyst console for network security monitoring. *http://bammv.github.io/sguil/index.html*. Retrieved on January 23, 2015.

VMware (2015a). Security and virtual machines. *https://pubs.vmware.com/vsphere-4-esx-vcenter/index.jsp#security_for_esx_systems/c_security_and_virtual_machines.html*. Retrieved on February 25, 2015.

VMware (2015b). Virtualization advantages. *http://www.vmware.com/virtualization/virtualization-basics/virtualization-benefits.html*. Retrieved on February 25, 2015.

Vogel, D. (2014). The benefits and challenges of virtual machine hosting. *http://www.datapipe.com/blog/2014/04/23/benefits-challenges-virtual-machine-hosting/*. Retrieved on February 25, 2015.

Vredevoort, H. (2013). Windows azure pack wiki. *http://www.hyper-v.nu/archives/hvredevoort/2013/11/windows-azure-pack-wiki/*. Retrieved on May 12, 2015.

Wang, Z. J. (2009). How to recover virtualized x86 instructions. *https://www.virusbtn.com/conference/vb2009/abstracts/Wang.xml*. Retrieved on February 26, 2015.

Webber, D. (2015). Snorby, all about simplicity. *https://www.snorby.org/*. Retrieved on January 23, 2015.