



## Project Documentation

### Project Hosting (Z13481)

**IT-factory**

Kenzo Caems – R0749796  
Jolien Calaerts – R0737987  
Brent De Vos – R0706989  
Guylian De Wit – R0716728  
Furkan Ozkan – R0744454  
Davy Van Roey – R0368936  
Leonardo Wolo – R0744052

Schooljaar: 2019 – 2020

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel





## Project Documentation

### Project Hosting (Z13481)

**IT-factory**

Kenzo Caems – R0749796  
Jolien Calaerts – R0737987  
Brent De Vos – R0706989  
Guylian De Wit – R0716728  
Furkan Ozkan – R0744454  
Davy Van Roey – R0368936  
Leonardo Wolo – R0744052

Schooljaar: 2019 – 2020

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel



## FOREWORD

We are seven students in the IT Factory of Thomas More Geel that commissioned to set up a hosting environment for our latest CCS project for students of APP and BIT. This project means that we create a hosting environment for end users where they can place their PHP projects. This environment has to support PHP and databases. We divided the tasks among our team so that everyone could continue to work at their own pace.

With the help of our teachers Bram Verbruggen, Filippo Bagnoli, Gunther van Landeghem and Nathalie Bosschaerts, we've learned a lot from this project. They were always there for us when we had a problem. Because of the Corona virus not everything went according to plan but that wasn't a problem for these amazing teachers who arranged a remote environment so we could realize our project. Finally, we wish to thank all of the above-mentioned people for being patient and helping us during our project!

# TABLE OF CONTENTS

<b>INTRODUCTION.....</b>	<b>10</b>
<b>1 LIST OF ABBREVIATIONS.....</b>	<b>11</b>
<b>2 HOSTING ENVIRONMENT ARCHITECTURE .....</b>	<b>12</b>
<b>2.1 Scheme .....</b>	<b>12</b>
<b>2.2 Explanation .....</b>	<b>12</b>
<b>3 COMPONENTS AND TECHNOLOGIES .....</b>	<b>13</b>
<b>3.1 Backup servers .....</b>	<b>13</b>
3.1.1 Operation of the backup servers .....	13
<b>3.2 Security server.....</b>	<b>14</b>
3.2.1 Operation of the FortiClient EMS .....	14
<b>3.3 Monitoring server.....</b>	<b>15</b>
3.3.1 Operation of the monitoring server.....	15
<b>3.4 Configuration management automation server .....</b>	<b>16</b>
3.4.1 Operation of the Puppet server and agents.....	16
<b>3.5 Bug Tracking .....</b>	<b>17</b>
3.5.1 Operation of the Mantis bug tracking system .....	17
<b>3.6 Ticket Solution .....</b>	<b>18</b>
3.6.1 Operation of OpenSupports .....	18
<b>3.7 Webserver.....</b>	<b>19</b>
3.7.1 Operation of the webserver .....	19
3.7.2 Example usage of our custom bash script.....	19
<b>3.8 Password safe (hosted by ourselves).....</b>	<b>20</b>
3.8.1 Operation of the password safe.....	20
<b>4 LIST OF IMPLEMENTED OPS REPORT CARDS .....</b>	<b>21</b>
<b>4.1 Ops Report Card Documents .....</b>	<b>23</b>
<b>5 TEST SCENARIOS .....</b>	<b>24</b>
<b>5.1 Malicious file upload by a ftp user .....</b>	<b>24</b>
<b>6 NEXT STEPS .....</b>	<b>25</b>
<b>6.1 Firewall .....</b>	<b>25</b>
<b>6.2 Hosting panel .....</b>	<b>25</b>
<b>7 CONCLUSION .....</b>	<b>26</b>
<b>SOURCES .....</b>	<b>27</b>



## INTRODUCTION

Our team had to set up a platform where the PHP projects could be managed. This was a tough challenge as we had to do everything remotely due to external factors. At first, we brainstormed how we were going to handle it. It made sense for us to start by drawing up a schedule (see first chapter), which gave us a better overview of the structure. We got access to a VSphere environment, the servers are located at the school, but the VM's were installed from our home by using the VSphere web client.

We were also instructed to update and upload every week our schedule that we had made on Trello. This was very useful to use so that we had a better overview of who was working on which part of the project. We had to complete at least 12 ops reports cards to show that we built a successful environment. The Ops Report Card is a list of 32 fundamental "best practices" or "capabilities" that high performance sysadmin teams do. We used it as a checklist to examine where our team needs improvement. We've completed more than 12 cards and have been looking for new challenges all the time. One of the most important things in our team was the division of tasks. We each implemented and worked out 2 cards so that we could work more efficiently and faster. We didn't leave everything until the last project week. We've done a lot of research's during the lessons. In addition, we first installed the software's on our local VM's on VirtualBox, so that we could host them on our apache server during the project week.

Security and redundancy was one of the main requirements for our system. That's why we chose to implement a password safe system called Securden. This shows that we have a mature way to manage our passwords. We also configured endpoint security on all our servers. Usability is of course also important in such a hosting platform. That's why we built a ticket system called OpenSupports. This ensures that the customers (in this case the students from APP and BI) can send a ticket to us. This keeps the contact between us and the them up to date. Not everything went so smoothly. It goes without saying that during the elaboration of our project there have been several conflicts. This is why we built a bug tracking system called MantisBT. This allows us to share and solve our problems between each other. Suppose something happens to our virtual machines or something like that, and everything breaks down... That's why we need backup for all of this. This led us to the idea that we put a backup in our own server as well as in the cloud (in case that the server explodes). This will make a backup of the system every night so it will not affect the operation of the servers during the day.

If we add all these things together, we can say that the famous quote '*great things never came from comfort zone*' got destroyed by our team. We can proudly announce that we've successfully hosted our platform from our comfort zone...



# 1 LIST OF ABBREVIATIONS

## Teammates

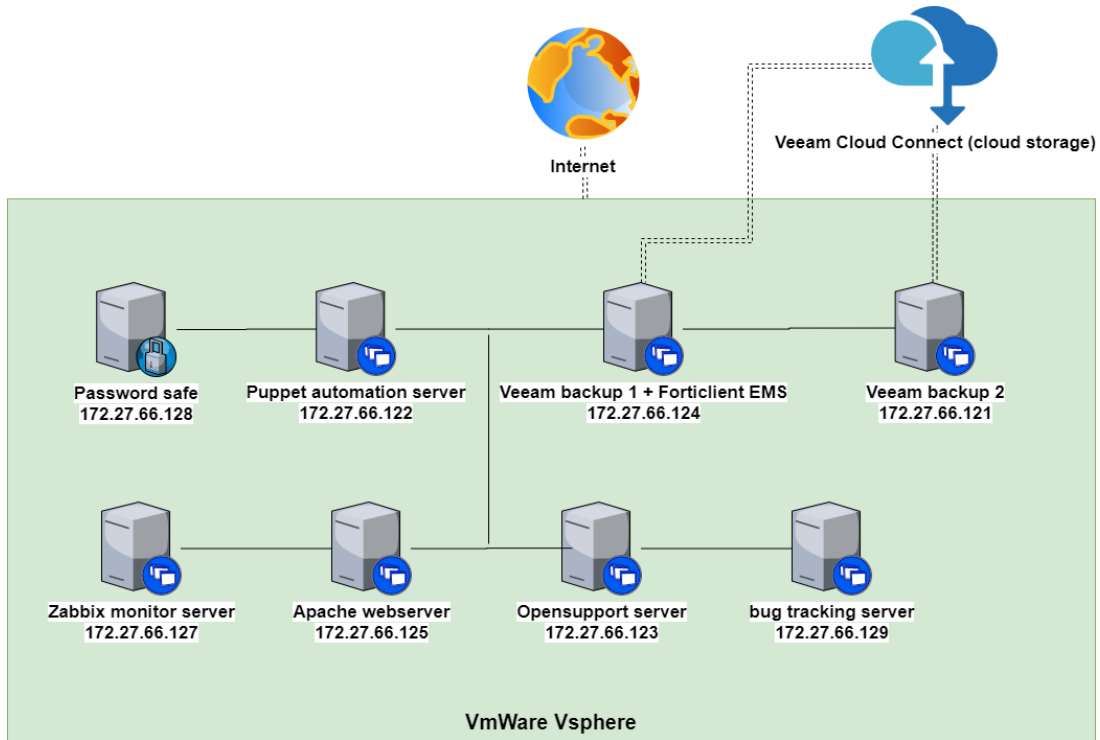
GDW	Guylian De Wit
FO	Furkan Ozkan
LW	Leonardo Wollo
DVR	Davy Van Roey
JC	Jolien Calaerts
BDV	Brent De Vos
KC	Kenzo Caems

## Technical terms

DNS	Domain name system
HTTP	Hypertext transfer protocol
IP	Internet protocol
SQL	Structured query language
DHCP	Dynamic host configuration protocol
SNMP	Simple network management protocol
PHP	Personal home page
HTML	Hypertext markup language
DB	Database
UTF	Unicode transformation format
URL	Uniform resource locator
TLS	Transport layer security
TCP	Transmission control protocol
ICMP	Internet control message protocol
ZBX	Zabbix
VM	Virtual machine
OS	Operating system
SMTP	Simple mail transfer protocol
BT	Bug Tracking
EMS	Endpoint management server
AD	Active Directory
SIEM	Security Information and Event Manager
SAML	Security Assertion Markup Language
FTP	File transfer protocol

## 2 HOSTING ENVIRONMENT ARCHITECTURE

### 2.1 Scheme



### 2.2 Explanation

For our architecture we have chosen for:

- Password safe to store all the logins for our accounts
- Puppet server to automate tasks on our servers
- Two Veeam backup servers and a backup repository in the cloud with Veeam Cloud Connect. If there is a failure we can always go back to an older backup
- Zabbix server to monitor all our servers and their services and alerting us if there is a failure
- Webserver to host our clients webspace. This server serves as an webserver, mysql and ftp server.
- Opensupport server, if a client has a problem he can always send in a support ticket asking for help.
- Bug tracking to view and manage all our current or fixed bugs.

## 3 COMPONENTS AND TECHNOLOGIES

### 3.1 Backup servers

#### Corresponding ops report card:

25. Are Your backups automated?



For the backup server software we chose to use the products of Veeam who has provided professional enterprise backup solutions for years. Both backup servers in our hosting environment are running the Veeam Backup & replication server software. With this software we are able to easily manage and configure multiple backup jobs for all our servers. We are also able to do different types of restores when needed when a server crashes or a file gets corrupted.

But we have also gone a step further. We were able to gain a Veeam Cloud Connect license from the Veeam platinum partner ReVirt Global. With this license are we able to store our backups in the cloud delivered by any.cloud. Thanks to this cloud connect license we are proudly following the backup 3-2-1 rule. This means:

- We have at least 3 copies of all our backups
- We are saving these backups on 2 different types of media (cloud and local storage)
- One of these 3 copies is stored on another location (in our setup in the cloud)

#### 3.1.1 Operation of the backup servers

Backup server 1 and 2 are both configured with automatic backup jobs that backups every server in our environment every 12 hours. These backups are locally stored on backup server 1 and 2. Once a day a backup of all servers is stored on the cloud storage. We choose for a once a day backup to the cloud (this happens at night) because backing up to the cloud request a lot of bandwidth and we want to save our bandwidth for our customers who make use of our webhosting services.

These backup jobs are configured with the 'enable guest file system indexing' option. Thanks to this option we can provide particular file restores from a server next to: instant vm recovery, entire vm restores and virtual disk restores.

To make our backup and restore operations faster we configured our backup repository on an extra hard disk that is formatted as ReFS volume with a 64 KB cluster size that makes fast cloning possible.

(fast cloning increases the speed of synthetic backup creating and transformation, reduces disk space requirements and decreases the load on storage devices)

The OPSDOC from these backup servers can be found in the OPSDOC bundle.

## 3.2 Security server

### Corresponding ops report card:

28. Do servers run self-updating, silent anti-malware software?



We want to protect our servers as well as possible. We took up the challenge and opted for the endpoint security solution from Fortinet called FortiClient, which is often used in both large and small companies. We requested and received a trial license for 10 endpoints. These endpoints are going to be centrally managed in FortiClient Endpoint management server which we installed on one of our windows servers.

We installed on all servers the FortiClient client software which can not be managed or disabled by the users but only by the administrator in the FortiClient Endpoints Management Server. This is important because cyber protection should be managed by someone who knows something about IT security and we don't want our users to create security gaps in our servers.

This FortiClient Endpoint Protection provides our servers with:

- Real time virus and malware protection
- Web Filter
- Application Firewall
- Real time reporting and patching of vulnerabilities
- Real time protection against botnets, zero-day malware, ...
- For more specific details visit: [www.forticlient.com/](http://www.forticlient.com/)

### 3.2.1 Operation of the FortiClient EMS

The EMS server is configured as follows:

The servers have real time protecting against malware, viruses, users trying to visit a malicious website,...

But twice a day there is a full system scan on all servers for vulnerabilities, viruses and malware. If the system finds any malicious things are these directly quarantined and deleted. The EMS server receives alerts and notifications with detailed information of these events.

Found vulnerabilities are being patched automatically (if possible) because we don't want to give hackers any chance. The EMS server again receives alerts and notifications from these events with detailed information which actions have been done to patch these vulnerabilities and which vulnerabilities had been found. The EMS server also gives an overview of all clients with real time information of maybe present security risks on certain clients.

The security policy and profile applied to all clients is centrally managed on the endpoint management server by the sysadmin. Users can't change any configurations of security policies or disable the real time anti-malware protection. Even cancelling a running anti malware scan is not possible.

The OPSDOC from this endpoint security service can be found in the OPSDOC bundle.

### 3.3 Monitoring server

#### Corresponding ops report card:

12. Does each service have appropriate monitoring?
3. Does the team record monthly metrics?



To monitor all our different servers and their services we chose to install the Zabbix monitoring software on a separate server. Our Zabbix monitoring server monitors the following from our servers:

- Network traffic of the network interfaces
- Bandwidth
- CPU jumps, usage, utilization
- Swap usage
- Disk utilization, queue, read/write rates
- MySQL connections, bandwidth, operations innoDB buffer pool, queries, threads,
- Apache requests per sec, worker states, connections
- Processes
- Memory usage, utilization
- System load
- Disk space usage

Next to monitoring provides our Zabbix servers also our monthly metrics of the past monitoring results.

#### 3.3.1 Operation of the monitoring server

Our Zabbix server is configured to monitor the above described things from our servers and in some cases some extra things depending on what services are running on that server. On every server runs a Zabbix agent which communicates with the Zabbix server. We also added and configured an alerting system to our monitoring server so that the right contact persons are contacted through email when a certain service is down. The alerting system and smtp service was integrated in the Zabbix software but the supported amount of emails that could be sent can be maxed out. We wanted to take no risk and opted for another smtp service called Mailjet. With Mailjet we are capable of sending 200 alert/notification emails a day.

The OPSDOC from this monitoring server can be found in the OPSDOC bundle.

### 3.4 Configuration management automation server

#### Corresponding ops report card:

16. Do you use configuration management tools like cfengine/puppet/chef?



To manage all of our Linux servers we installed the Puppet packages on a separate server and configured this server as a Puppet master server where all configurations on other servers can be arranged.

#### 3.4.1 Operation of the Puppet server and agents

As told before we configured a separate server as a Puppet master server where all configurations on other servers can be arranged. To configure this correctly we needed to install the Puppet agent software on all other Linux servers and perform a little bit of configuration. With this software installed all Linux servers are able to communicate with the Puppet master server and receive configuration updates from the master server.

Example: Our apache webserver is fully configured by pushing configurations from the Puppet master server.

Also updating or upgrading Linux versions on our servers can be done on all machines by pushing a configuration from our Puppet master server.

Without our Puppet server we had to do this manually on all servers which is very time consuming.

The OPSDOC from this puppet server can be found in the OPSDOC bundle.

### 3.5 Bug Tracking

#### Corresponding ops report card:

7. Does your team use a bug-tracking system for their own code?



Nobody's perfect, and that goes for software too. We came across a lot of problems and errors that all had to be solved during the implementation of the software in our VMs. This gave us the idea that we would best set up a bug tracking system and update this every time we received an issue.

We have chosen MantisBT as software. With this system we can easily report our bugs that we had and manage the data on bugs that occur in the system. The goal is to maintain high product quality and get rid of time-wasting things. We can keep our team updated with notifications on issue updates, resolution, or comments. In addition we can also customize the issue fields and notifications. It is clear and simple and easy to implement.

The main reason that we are using this, is the fact that bugs and software errors have damaged many reputations and are responsible for massive costs both in lost revenue and hours wasted digging through logs. Better bug tracking could have prevented a great deal of heartache all around the world.

#### 3.5.1 Operation of the Mantis bug tracking system

First of all we needed to build a LAMP server which includes:

- Apache 2
- PHP
- Database (MariaDB)

After the pre-installs were complete, we've created an admin for Mantis. This admin is entitled to everything and can create and delete new users. In case of conflicts he can also change the rights.

After we logged in to the system as admin with the default credits, we created a new project named 'Project Hosting' (We can also add different projects later if necessary). Under this project we created several users with the same rights so that everyone could access the system. This makes it easy to edit issues and get a proper view of the bugs. Then we started creating bugs. When a bug has been created, modified or removed, the users that are participating in the project receive an email of what has been created, along with a link that goes directly to the problem details.

After a bug was fixed, the person responsible for it could put it in the 'solved' category so it wouldn't be among the bugs anymore. This is of course to provide a better overview for the users. These bugs can be changed and removed by anyone since we, as the admin, gave everyone the same rights in the beginning. But this can be changed very easily and quickly.

### 3.6 Ticket Solution

#### Corresponding ops report card:

1. Are user requests tracked via a ticket system?



We want to be able to support our end users in a quick, and efficient way. We opted to use OpenSupports as our ticketing system.

OpenSupport is a ticketing system with login panel. Our end users can create a account on our support platform and open a ticket regarding various issues which will be prioritized by severity by our support team and answered accordingly. End users can upload screenshots to further document their issues and can create tickets under different departments. (Help, Security, Bugs...).

The ticket is then assigned to one of our support team employees and will help our end user out as soon as possible. This is done via their staff account(s), you can access the administrator panel through <https://url.com/admin> and can manage tickets, end users, staff and departments via this administrator panel.

Opensupports also keeps track of the created tickets, and thus offers us great insights. This way we can analyse user data and keep track of frequently asked questions and address these accordingly.

The Opensupports software offers the following:

- Account creation
- Ticket creation under different departments
- Assigning severity levels to tickets
- Creating different departments
- Statistics for analytical purposes
- Feature rich administrator panel
- Creation of articles

#### 3.6.1 Operation of OpenSupports

Opensupports is an opensource platform, it requires PHP, MySQL and Apache2. It is configured to store user accounts and the generated data into this database. Passwords are salted and hashed for optimal user security. When the installation is complete you can sign in to the administrator panel to add staff members, managed end users, manage tickets, create articles etc...

The web panel is very easy to navigate and allows us to work efficiently. It allows you to create different departments, assign staff members to each department and separate workforce into different categories.

Please refer to the Installation and Usage documentation for more details.



### 3.7 Webserver



Our webserver makes use of Apache2, PHP, MySQL, PhpMyAdmin, vsftpd, puppet and makes use of a custom written script in Bash by one of our team members.

Apache2 is configured by our puppet master server and is used to provide a hosting space for web applications and files. We use the webserver as our central MySQL server. The applications running in other virtual machines make use of this MySQL instance. Furthermore, we've installed phpMyAdmin to easily manage the database(s), and we configured virtual hosts with a Bash script written by one of our team members.

The webserver takes care of the following:

- Providing users with individual user accounts
- Separated virtualhosts
- Separated home and virtual host directories
- Separated FTP instances
- Centralized MySQL instance
- PhpMyAdmin to easily manage databases
- Custom bash script to automate the creation of virtual hosts

#### 3.7.1 Operation of the webserver

The only thing a staff member has to interact with is the custom bash script to add virtual host instances for new end users. It is executed by running the following command. `sudo bash vhost.sh <name> <password> <domain.com>`.

This will create a new user, with its corresponding home and virtual host directory. It will add the correct apache2 virtual host configuration file, and even create a default index.html page for this user.

Aside from that, you can easily manage the MySQL databases via phpMyAdmin and make changes where needed.

The custom bash script can be found on:

[https://github.com/Sp3nge/Project-Hosting-Team2CCS/tree/master/vhost\\_script](https://github.com/Sp3nge/Project-Hosting-Team2CCS/tree/master/vhost_script)

(this script is already cloned 29 times by people from the whole world. We are proud to contribute in the open source community).

#### 3.7.2 Example usage of our custom bash script

```
root@webserver:~# bash vhost.sh hello hello hello.com
** Script created by SpengeSec aka R0716728**
* Adding new user hello
* Password set for user hello
* Created Virtual host directory
* Changing ownership for vhost directory
* Added user hello to www-data usergroup
* Duplicating default apache2 vhost configuration
* Done writing vhost configuration
* Enabled new vhost in apache2
* Reloaded apache2 service
* Default index.html created
* Symbolic link from /var/vhosts/hello to /home/hello created!
* All done, website available!
* Add <ip> <www.website.com> <website.com> to /etc/hosts and/or C:\Windows\System32\drivers\etc\hosts
```

### 3.8 Password safe (hosted by ourselves)



To properly manage all our machines and tools, we will need at least 1 account for every piece of software. So we need as many passwords as accounts. To manage all these passwords, we will store them all in a password vault called Securden. With Securden, everyone in our team can store their passwords in a centralized database. In Securden, we can set up a safe infrastructure.

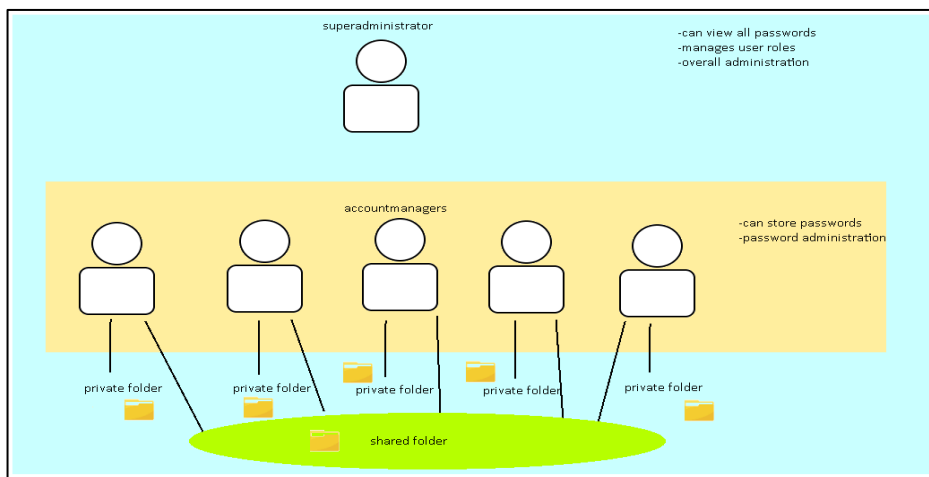
There are 4 main types of roles. In order of least to most powerful: user, account admin, admin and super admin.

- Normal User: can make his own accounts and share those with other users. They can also view the accounts that are shared by the administrators.
- Account Administrator: can do all the same things as users. They also have administration tools that involve those accounts.
- Administrator: can do all the above. Administrators can also manage all users. They can create new users, delete old, alter their credentials or change their role. Finally, administrators have control over all configurations of the app.
- Super Administrator: can do all of the above, but also has the ability to view all accounts of all users (both private as shared).

\*An account in Securden is a user/password combination and some additional info about this account (type, URL/IP address, ...)

#### 3.8.1 Operation of the password safe

In our infrastructure every member of our team will be an Account Admin. Each account admin will get 2 folders to store their accounts in: a private one for all personal accounts like Mantis and Github, and a shared folder with the logins for our machines. There also will be a Superadmin with the name PS\_SA, that will be managed by only 1 team member. This superadmin account will be used in case of emergency (to retrieve lost logins, disable possible hacked accounts...). Because PS\_SA is the only one that can alter the roles of others, there is only one way intruders can hijack Securden and lock out everyone else from the system.



## 4 LIST OF IMPLEMENTED OPS REPORT CARDS

### **OPS 1: Are user requests tracked via a ticket system?**

Yes they are, refer to the chapter [Ticket Solution](#) or the OPSDOC of Opensupports provided in our uploaded zip file for more information.

### **OPS 2: Are “the 3 empowering policies” defined and published?**

Yes they are. We have written and published these on our GitHub repo: <https://github.com/Sp3nge/Project-Hosting-Team2CCS/blob/master/the%203%20empowering%20policies%20uitgeschreven.pdf>

The also included this written file in our uploaden ZIP file. You can find the file in the ZIP archive under the name '3empoweringpolicies.pdf'.

### **OPS 3: Does your team record monthly metrics?**

Our Zabbix server provides us with monthly metrics from various information. Refer to the chapter [Monitoring server](#) or the OpsDoc of Zabbix provided in our uploaded zip file for more information.

### **OPS 5: Do you have a password safe?**

Yes we have, it's a must have these days! Refer to chapter [Password safe](#) or the OpsDoc of Securden password safe in our uploaded zip file for more information.

### **OPS 6: is your team's code kept in a source-code control system?**

Yes it is, all our documents and self-written scripts are kept in our own GitHub repository where all our team members are invited as collaborators. Our repository can be accessed through this link: <https://github.com/Sp3nge/Project-Hosting-Team2CCS>

### **OPS 7: Does your team use a bug-tracking system for their own code ?**

Yes we do! Please refer to chapter [Bug Tracking](#) or the OpsDoc of Mantis Bug tracking for more information.

### **OPS10: Do you have a “post-mortem” process ?**

Yes we have, Please refer to the post mortem file in our uploaded zip file.

### **OPS 11: Does each service have an OpsDoc?**

Yes, we have written an OpsDoc for every service in our hosting setup. You can find these OpsDocs in our uploaded zip file.

### **OPS 12: Does each service have appropriate monitoring?**

Yes they have! All our services are monitored by our Zabbix server. Please refer to chapter [Monitoring server](#) or the OpsDoc of Zabbix in our uploaded zip file for more information.

**OPS 16: Do you use configuration management tools like cfengine/puppet/chef?**

Yes, all our servers can be managed by our Puppet Master server (except the windows servers). Please refer to chapter Configuration management automation server or the OpsDoc of Puppet in our uploaded zip file for more information.

**OPS 19: Is there a database of all machines?**

Yes there is! Next to the machine database that Zabbix provides have we created an excel file with a list of all servers with some basic attributes like: OS, RAM, Disk size, IP address, primary contact person, ... . Please refer to 'ServerDatabase.xlsx' file in our uploaded ZIP file to see the self-made database.

**OPS 22: Do you have a PC refresh policy ?**

Yes we have. Please refer to the 'PCrefreshPolicy.pdf' file in our uploaded ZIP file to view our PC refresh policy.

**OPS 25: Are your backups automated?**

Yes they are! We have created a very good backup job configuration, thanks to our 2 Veeam backup servers and cloud storage. We are proudly following the 3-2-1 backup rule with our backup setup. Please refer to chapter [Backup servers](#) or the OpsDoc of Veeam backup server in our uploaded zip file for more information.

**OPS 27: Do Machines in your data center have remote power / console access?**

All our servers are running a ssh service so we can remotely access these servers. We also have remote access to our servers through our password safe from Securden. The ESXI hosts where our servers are hosted on can we access via an VPN service to the school network.

**OPS 28: Do desktops/laptops/servers run self-updating, silent, anti-malware software?**

Yes they do! We have configured a very strict security profile on all servers to give malware or hackers no chance. Please refer to chapter [Security server](#) or the OpsDoc of Forticlient Endpoint management server in our uploaded zip file for more information.

**OPS 29: Do you have a written security policy?**

Yes we have. Please refer to the 'SecurityPolicy.pdf' file in our uploaded ZIP file to view our Security policy document.

**Total implemented Ops Report Cards: 16**

## **4.1 Ops Report Card Documents**

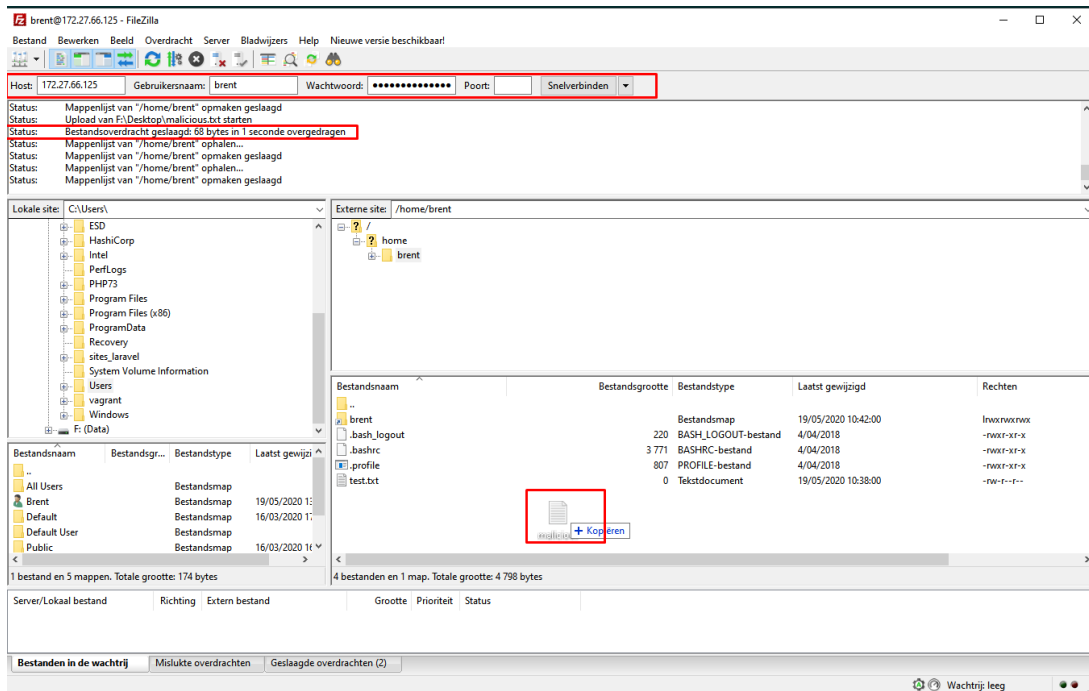
All of the required documents of an Ops Report Card can be found in the Zip File that we've uploaded.

## 5 TEST SCENARIOS

### 5.1 Malicious file upload by a ftp user

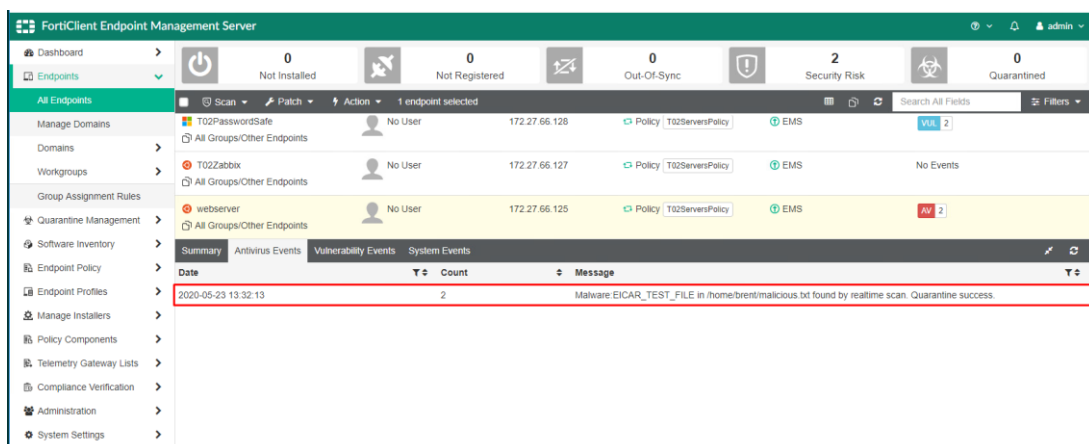
Every client that registers for a webspace on our servers has ftp access to his private folder and their public\_html folder for website hosting.

As test scenario we tried to upload an EICAR test file (virus test file) to test the security of our webserver.



We were able to upload the file but it immediately disappeared. The security of our webserver protects the server in real-time and successfully placed it in quarantine and deleted the malicious file.

All of these events were successfully reported to our Forticlient endpoint management server where the security of all our servers is being managed.



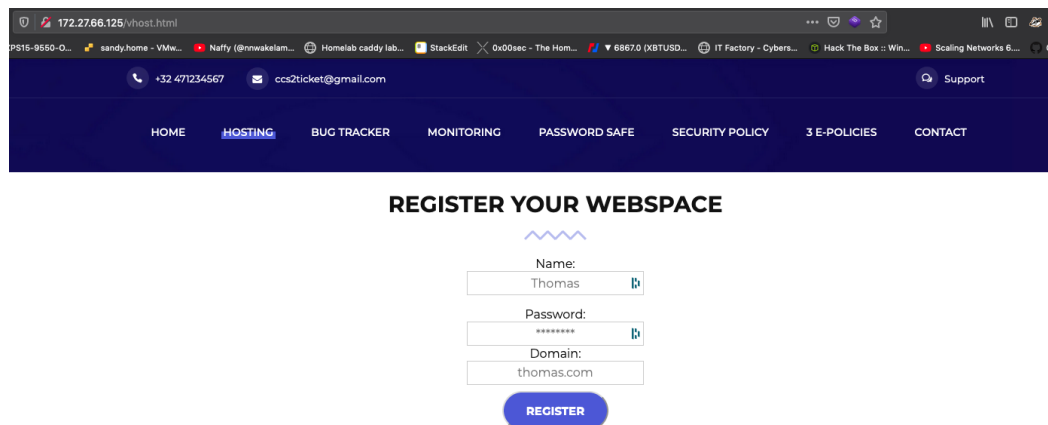
## 6 NEXT STEPS

### 6.1 Firewall

In a next version of this project we would install 2 pfSense firewalls running in high availability mode. Our servers are now very well secured with Fortinet Forticlient & EMS. But a firewall would be the cherry for our server security.

### 6.2 Hosting panel

At the moment any one is able to access the hosting panel on our website.



The screenshot shows a web browser window with the address bar displaying '172.27.66.125/vhost.html'. The page has a dark blue header with a navigation menu: HOME, HOSTING (active), BUG TRACKER, MONITORING, PASSWORD SAFE, SECURITY POLICY, 3 E-POLICIES, and CONTACT. Below the header, the main content area is titled 'REGISTER YOUR WEBSITE' with a blue wavy line separator. The registration form consists of three input fields: 'Name:' with the value 'Thomas', 'Password:' with the value '\*\*\*\*\*', and 'Domain:' with the value 'thomas.com'. A blue 'REGISTER' button is positioned below the domain field.

At the moment this allows anyone to create a 'webspaces' via the vhost.sh bash script Guylian has written. This script creates a user account on the webserver and a virtual host for the specified domain name.

This is not ideal, in a real world scenario you'd want to isolate this function.

We would be perfectly capable of implementing a login and registration panel with html and PHP, linked to our MySQL database. But due to lack of time we've opted to not implement this functionality in our Proof of Concept.

## 7 CONCLUSION

*"If everyone is moving forward together, then success takes care of itself."*

This quote was our guiding thread during the 2nd semester. We worked very hard to set up the ideal hosting environment for our fellow APP/BI students. If we hadn't had each other's help, we would never have come to a successful conclusion.

Of course, we've followed different stages. First there was the research stage where we investigated individually each software. It had to be safe and easy to implement for the ideal solution. After all that research, we entered the design phase in our local network. Here we tested every piece of software on our laptop. Documentation was crucial for us, so we decided to focus on documentation during this phase as well, so we didn't end up with just design. In the end we finally got access to VSphere where we could design our project for real.

We've had ups and downs like any team. But since we were strong in our shoes, we were able to counteract all the problems. Coming together is a beginning, staying together is progress, and working together is success!



## **SOURCES**

We provided a centralized source list in our zip file where all sources of all documents are listed.