[Document title]

[Document subtitle]

VASCO Data Security

[Course title]

# Preface

# Summary

# Table of contents

[Preface 1](#_Toc511635377)

[Summary 1](#_Toc511635378)

[Table of contents 2](#_Toc511635379)

[List of figures, graphs 3](#_Toc511635380)

[Introduction 4](#_Toc511635381)

[Problem statement 5](#_Toc511635382)

[Current situation 5](#_Toc511635383)

[What could be improved 5](#_Toc511635384)

[Goals 6](#_Toc511635385)

[Methodology (practical goals) 6](#_Toc511635386)

[Research (documentation, discussion and quantification of the gains) 6](#_Toc511635387)

[Implementation 7](#_Toc511635388)

[Proposed architecture 7](#_Toc511635389)

[Changes needed 7](#_Toc511635390)

[Discuss technologies 8](#_Toc511635391)

[Selenium grid 8](#_Toc511635392)

[Docker 8](#_Toc511635393)

[Why containers? 8](#_Toc511635394)

[IE in a container? 8](#_Toc511635395)

[Issues with the IE node 9](#_Toc511635396)

[Performance difference 10](#_Toc511635397)

[Conclusion 11](#_Toc511635398)

# List of figures, graphs

# Introduction

Introduce Vasco

# Problem statement

## Current situation

\\srv-be-file.vasco.com\UserProfiles$\willaar1\Documents\Stuff voor thesis\thesisDocs\Presentaties\prentjes\Poging2000.emf

## What could be improved

# Goals

## Methodology (practical goals)

## Research (documentation, discussion and quantification of the gains)

Document the advantages and disadvantages of the proposed deployment architecture

Include a discussion on the difficulties encountered with the solution/technologies

Quantify the proposed solution’s impact in terms of saved resources and execution time

# Implementation

## Proposed architecture

Needed: a picture that explains Selenium (grid)

\\srv-be-file.vasco.com\UserProfiles$\willaar1\Documents\Stuff voor thesis\thesisDocs\Presentaties\prentjes\Poging1000.emf

## Changes needed

Changes needed to the test suite code to make it run on the new architecture

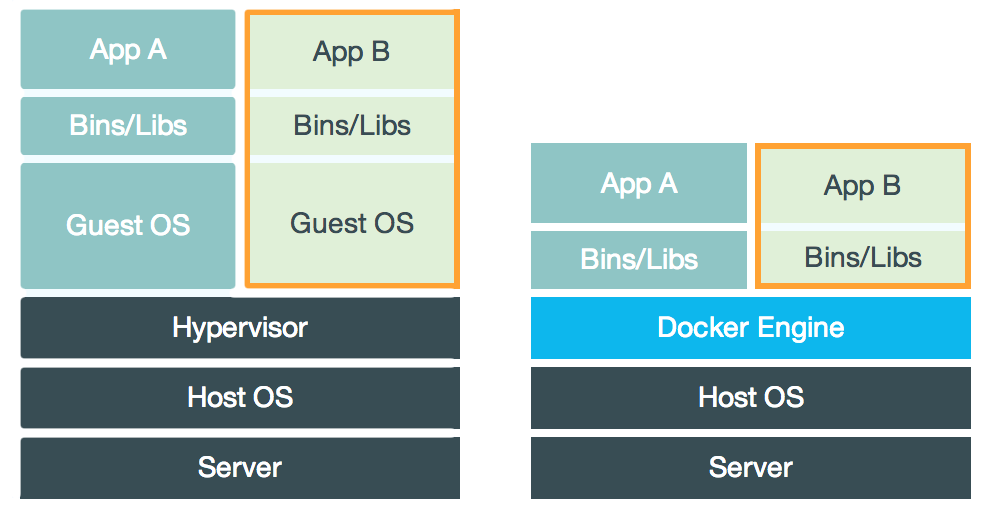
# Discuss technologies

## Selenium grid

## Docker

### What is a container?

Maybe make an original diagram comparing a vm and a container



The best way to explain what a container is, is probably comparing it to a virtual machine (VM for short). While containers and VMs are two different technologies, they can be used for a similar purpose: the isolation of an application and the relevant dependencies in a self-contained unit.

Furthermore, utilizing VMs or containers this way can remove the need for physical hardware, and can result in a more efficient use of resources

#### Virtual Machines

At its core, a virtual machine is an emulation of a full-fledged computer that runs an operating system, executes commands and runs applications, as if it would be an actual, real computer, instead of just a simulation. A VM runs on a physical machine through a hypervisor. A hypervisor can run either on the operating system of host machine (a hosted hypervisor) or directly on the hardware (native or bare-metal hypervisor)

The hypervisor itself runs on a physical system called a “host machine”. The host machine provides the VMs with the necessary resources, like RAM and CPU. These resources can be divided amongst the VMs to best suite your use-case.

A defining feature of a VM is that they come with a complete virtualized hardware environment, which includes a CPU, RAM, storage, network adapters, graphical interfaces, and more.

#### Containers

Unlike a VM which provides hardware virtualization, a container provides virtualization on the operating system level. It does this by abstracting the user space.

A container is similar to a VM in quite a couple of ways. For instance, they have a private process tree, their own network interface and IP address, can execute commands as root and mount filesystems.

The big difference with virtual machines is that containers package up just the user space, and not the kernel or virtual hardware like a VM does. A container shares the host system’s kernel with other containers. But, because every container gets its own user space, it is possible to run multiple containers on a single host.

### Why containers?

### IE in a container?

For the execution of web based test suites, there is a need for the tests to be run on Internet Explorer to ensure a good coverage of use cases. There are, however, a few issues with the proposed deployment architecture related to Internet Explorer. The first being: running internet explorer in a container

The SeleniumHQ repository offers Docker images for a Selenium Hub and for nodes with either Firefox or Google Chrome, with Internet Explorer being notable absent.

There is, however, a NodeBase dockerfile provided to serve as a basis from where to build your own dockerfile to create a non-standard image. The NodeBase is equipped with only the install of selenium, and does not include a browser.

However, the issue here is that the NodeBase image is based on the Base image in the repository that in turn is based on Ubuntu 16.04. In addition, Internet explorer was never ported to Ubuntu, or any other Linux distribution for that matter, so installing it on the NodeBase image is practically impossible.

There is always the possibility of installing wine (a compatibility layer capable of running windows applications on other operating systems such as Linux and macOS), but that would be a bad idea as running in on a compatibility layer in a container seems like more trouble than it’s worth. Especially considering that Internet Explorer is already known for running slowly and being generally unstable on wine.

Another problem is that if you were to build a Docker image based on Windows and run an Internet Explorer browser on it that way, then you wouldn’t be able to run that image on a Linux host. This is because a container image always needs to be based on the same OS as the host you are running it on.

The workaround here is just to run a selenium node running Internet Explorer on a windows machine and adding it to the grid. This way we lose the scalability of running it as a container, but it is necessary to get it working.

(do I need to put the steps to configure a machine to be a selenium node here?)

## Issues with the IE node

Discuss the IE Webriver and why it makes it difficult/impossible to run the suite

<https://stackoverflow.com/questions/19662045/selenium-hover-elements-with-ie>

<http://jimevansmusic.blogspot.be/2013/01/revisiting-native-events-in-ie-driver.html>

<https://github.com/SeleniumHQ/selenium/wiki/InternetExplorerDriver>

<https://github.com/seleniumhq/selenium-google-code-issue-archive/issues>

# Performance difference

# Conclusion

Should at least answer the following questions:

* Which parts of the proposed deployment architecture were actually implemented?
* Recap which other parts were scrapped (maybe shortly touch on the why), and which parts were changed due to technological limitations or other constraints?
* What didn’t work at all?
* Is there a gain in efficiency?
  + Why, or why not?
* Is the gain worth the trouble?

Include suggestions for future research

NO NEW INFORMATION HERE!!!