Дипломна

работа

Иван Илков

Cloud as a Linux service

Table of contests :

1. Анализ на проблемната област.
2. Функционално описание на системата.
3. Софтуерно проектиране на системата.
4. Описание на програмната реализация и особеностите на нейното приложение.
5. Ръководство за потребителя.
6. Инсталиране на системата.

# Анализ на проблемната област.

Днешните голями ИТ предприятия черпят изчислителна мощ от многоядрени процесори и големия брой виртуализирани машини, решавайки по този начин , експоненциално нарастващата нужда от различни по тип и сложност сървъри.

В тези ИТ организации, създаването на сървърни профили е предизвикателство, тъй като ръчно създаваните такива, доведе администраторите до безброй конфигурационни машини. Спазването на добрите практики , добави още един слой на сложност към сървърната администрация, когато ИТ организациите се борят да се покрие нарастващия обем на вътрешни стандарти, стандарти за сигурност и правните директиви.

## 1.1.Съществуващи решения

PUPPET - <http://puppetlabs.com/>

Пъпет лаб са едни от лидерите в ИТ автоматизацията .Тяхното решение за големите ИТ организаци е моделно-базиран продукт , проектиран така че да решава ефективно сървърното администриране , което е най-критичния елемент на днешната дата център инфраструктура.

VAGRANT - <http://www.vagrantup.com/>

Vagrant – осигурява лесен за конфигуриране , възпроизвеждане и преносима среда .Създадена на базата на най-добрите технологични стандарти и контролирана от един последователен работен поток, за да увеличи производителността и гъвкавостта на мървърното администриране. За да постигне своята цел , програмата използва гигантите в тази област . Машините се провизират върху VirtualBox , VMware . След това , стандартни инструменти за провизиране могат да се използват, като например Шел скрипт . С помоща на тези инструменти , може да се инсталира и конфигурира различен софтуер върху дадена сървърна машина.

CHEF - <http://www.getchef.com/>

Chef е системна и облачно ориентирана автоматизационна инфраструктурна програма . Тя прави лесното провизиране на сървъри и приложения на всякакъв вид физическа, виртуална или облачно базирана локация, без значение от големината ан инфраструктурата. Chef – клиента разчина на абстракни дениции (cookbook / recipes – готварска книга / рецепта) който са писани на Руби и се управляват като сорс код. Всяка дефиниця указва как специфична част от ИТ инфраструктурата трябва да се управлява. След това chef клиента пирлага тези дефиниции върху сърварите и приложенията , както е отисано. Резултатът е напълно автоматизирана инграструктура. Когато нов сървър се стартира , единственното нещо което програмата прави е да приложи тези готварски книги и рецепти.

## 1.2.Сравнителен анализ

## 1.3. Извод

На базата на разгледаните съществуващи приложения могат да се направят следните изводи:...

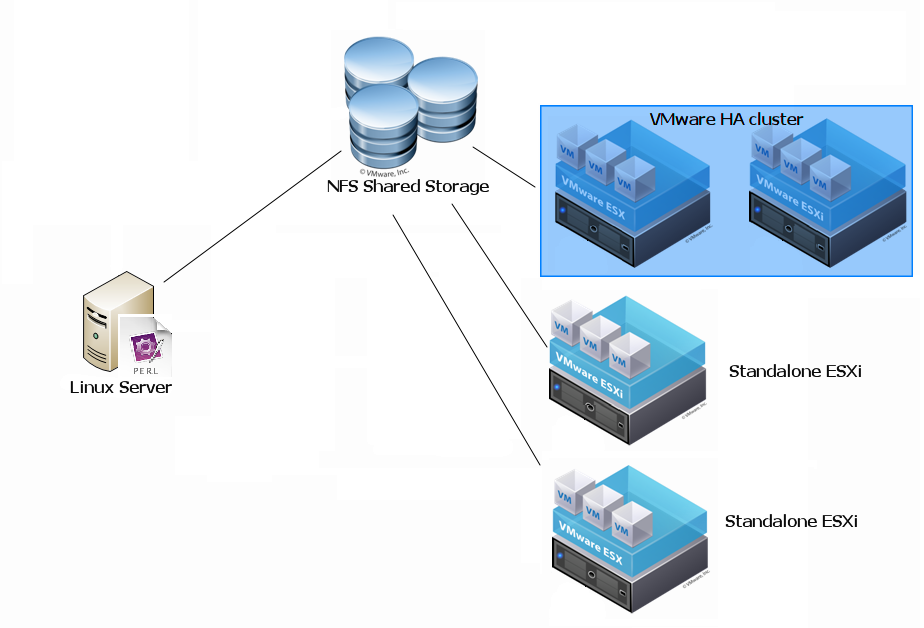
# 2.Функционално описание на системата.

## 2.1 Въведение

Програмата за услуги в облака , служи за атоматично провизиране на виртуални машини – сървъри. Чрез автоматизацията на процеса , се избягват редица проблеми , намалява се обема на работата която даден аминистратор трябва да извърши . Както се елиминира и възможноста от човешка грешка . Всичко това довежда до един праволинеен процес за провизиране на сървъри, в рамките на минути.

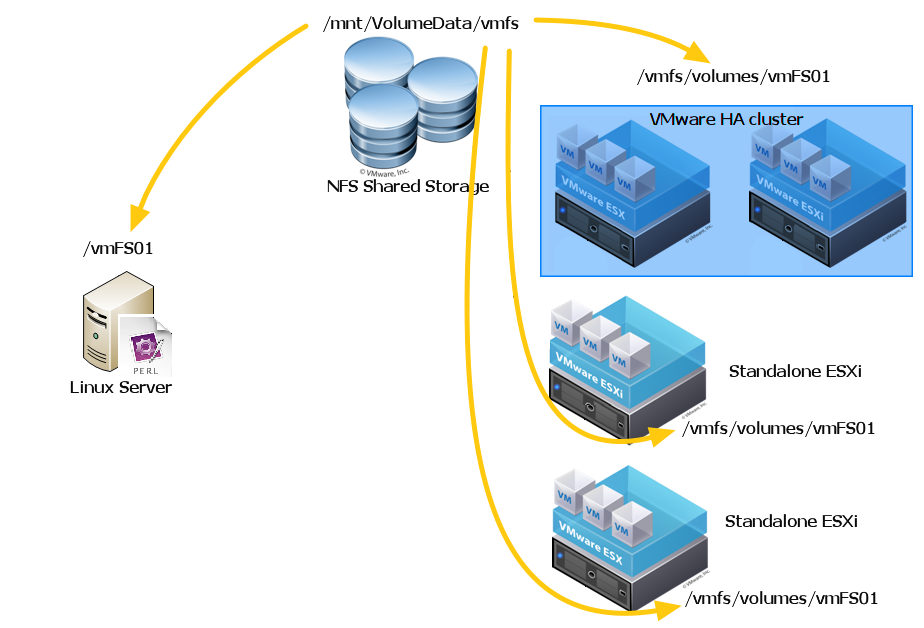
Програмата за услуги в облака се изпълнява ръчно ,т.е. изисква се ръчно стартиране и ръчно попълване на конфигурационен файл. Но има възможност , програмата да се доразвие и адаптира към мониторинг система . В следствие на което на базата на достигнати стойнотси на използваемост на ресурси (90% процесор, 90% оперативна памет) .Програмата ще може в рамките на няколко минути да провизира и конфигурира нови сървъри в зависимост от нуждите и по този начин да задоволи моментните изисквания от към ресурси.

## 2.2 Обобщена Аритектура



Фиг 1. Обощена аритекрура

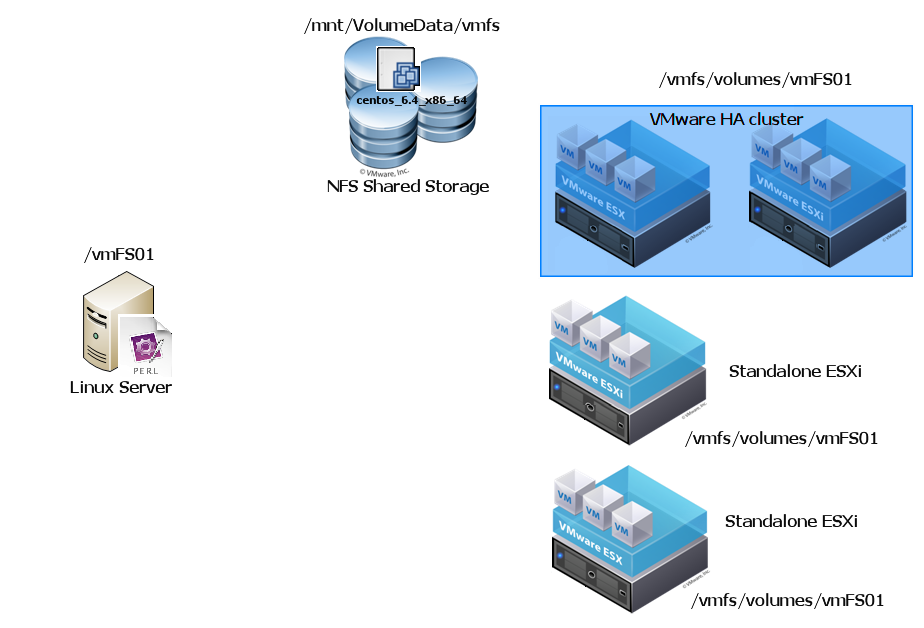
За реализацията на програмата , като постановка съм използвал конфигурация със споделен мрежови сторидж NFS. (Програмно , няма ограничение системата да се пренапише за FC сторидж) . Идеята тук е линукс сървъра , от който ще се изпълнява програмта да има достъп до същитото дата хранилище ,което се използва от vmware hypervisor-а за съхранение на виртуалните машини. Като краен резултат , трябва да имаме едно дата хранилище /mnt/VolumeData/vmfs01/vmFS01/ , намиращо се в мрежовия сторидж , монтирано едновременно върху линукс сървъра /vmFS01 и вървху всеки един от vmware hypervisior като /vmfs/volumes/vmFS01.



Фиф 2. Споделяне на общ мрежови дата ресурс

Следващото пред условие , което трябва да бъде изпълнено , преди да изпълним програмта е да поставим образец на операционната система (template OS). Tой е представлята базова операционна система ,с конфигурирани общи настройки за сигурност и ъпгрейдван с полседните пакети за сигурност . Доставя се заедно с програмата и се копира в общото дата хранилище.

Можем да имаме не ограничен брой такива образци , за различните операционни системи – 32 или 64 битова , както и различните линукс дистрибуции. Името на самия образец на следваща стъпка се описва в конфигурационния файл. (centos\_6.4\_x86\_64 примерно име на образец)



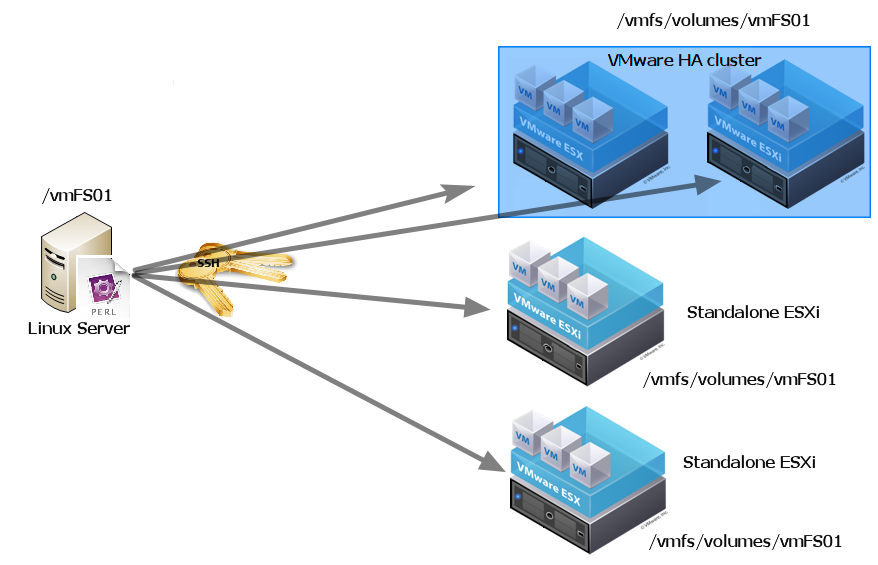
Фиг 3. Образец на операционната система , копиран

в общото мрежово хранилище

На следващата стъпка от конфигурирането на средата , ще е необходима безключово свързване между между линукс сървъра , на който се изпълнява програмата и всичките vmware hypervisor-и . Програмата прави проверка на базата на зададените в конфигурационния файл ESXis и в случай ,че няма предварително направена такава, пита потребителя дали желае това да стане автоматично. Тук става въпрос за SSH keys, който ще бъдат генерирани и копирани на съответните VMware операционни системи. Единственото ръчно действие което трябва да се предпиреме от потребителя е при съмото копиране да въведе паролата на отдалечения сървър.

Тези действия са необходими ,за да може програмата да изпраща съответните команди до сървъра за виртуализация.

Kъм този момент , програта вече е валидирала данните въведени в конфигурационния файл и знае ИП адресите на всички VMware хипервайзори с който ще си взаимодейства.



Фиг 4. Разпространение на SSH keys

Като обубщение , до тук можем да кажем , че средата е напълно конфигурирана. Т.е вече имаме :

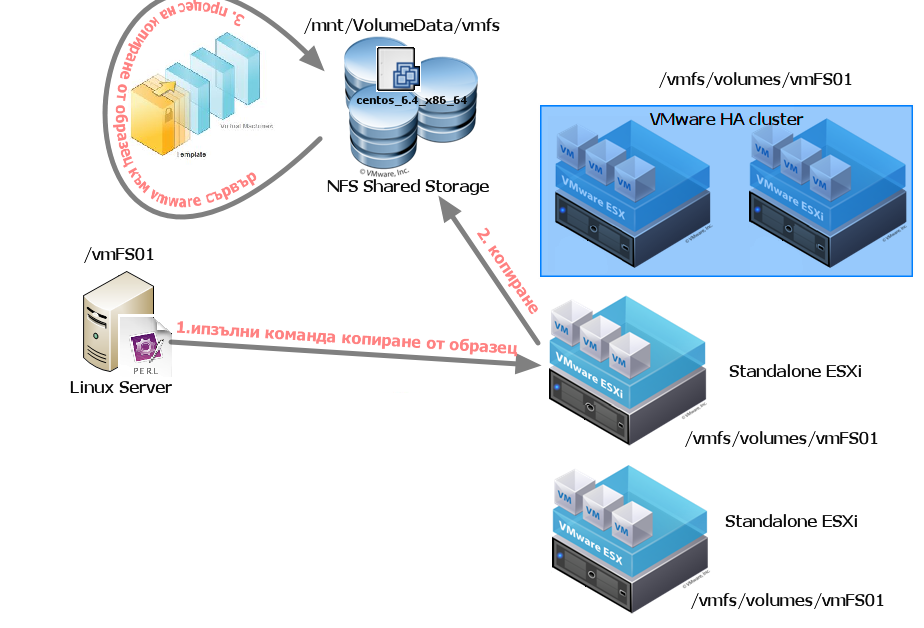
- Общо дата хранилище достъпно , както от VMware сървърите , така и от линукс сървъра от който изпълняваме програмата

- Имаме образец или темплейт на операционата система , който ще дистрибутираме . Намиращ се в общото хранилище

- Разполагаме и със свързаност без въвеждане на парола по протокол SSH , между линукс съврвъра и крайните VMware сървъри върху , който работи или ще работи нашите облачни услуги.

## 2.3 Същина на програмата

Следва същината на нашата програма за услуги в облака. На този етап , входните параметри са взети и се пристъпва към създаване на крайните сървъри и услуги. Пърият етап от тази фаза е , копирането на задения темплейт който ще се използва към нормална виртуална машина .Реализацията става като скрипта ипрати съответната команда към един от VMware хостовете (избран на принципа round-robin). Това е и най – времеотнемащата задача. Времето неогходимо за нея може да варира от 3-4 минути до 15-20 в зависимост от големината на дисковете на образеца , както и от скороста на мрежата използвана за споделяне на дата. Добрата новина е , че поради това че използваме VMware команда изпълнена на сътоветния VMware хост, не се копират целите файлове от информация , а само значещите данни.



Фиг. 5 копиране на образец

След приключване на задачата по копиране на образец. Ние разполагаме с една базова операционна система, която трябва да бъде конфигурирана според въведеното в конфигурационня файл. Програмата за услуги в облака , има възможност да конфигурира три типа съсръри. Те са сървър за уеб страници – хостинг , сървър за споделяне на файлове – шеър сървър и днс . Тези типове сървъри служат само за пример, няма ограничение програта да бъде пренаписана и добавена възможност за сървър на който върви някакво специфично клиентско приложение .

В този момент ние разполагаме с идентично копие на образеца , което трябва да бъде конфигурирано , включително инсталиране на допълнителни програми/пакети , настроки на тези програми , както и базови настроики по операционната система като ИП, мрежова маска , шлюз и днс сървъри.

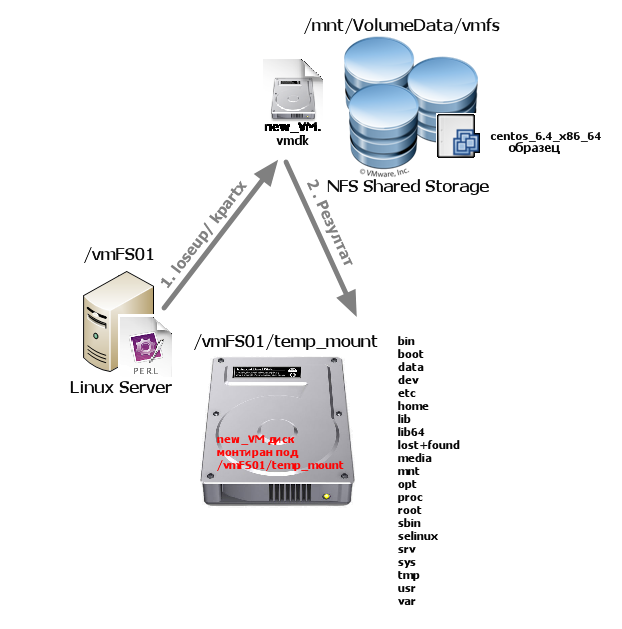
Това конфигурира става възможно благодарение на две линускс програмки - kpartx и losetup , който са инсталирани заедно с програмата за услуги в облака.

Losetup e линукс програма която служи за асоцииране на луп устройства /dev/loop0 към нормални файлови или блокови устройства, като например блок диска на новата виртуална машина която имаме.

Програмата прави точно това, търси първото свободно луп устройство на линукс машината от която изпълняваме програмта и го асоцира с диска на новата виртуална машина.

Kpartx е линукс програма , която може да чете служебната информация (partition table) от даден диск и след това монтира всеки един от дисковите дялове.

Това е следващата стъпка , която нашата програма предпирема , тя е конфигурирана така че автоматично да намери главния root дял на новата виртуална машина и да го монтира като папка в от операционната система .



Фиг 6. Монтиране на рут диска на новта виртуална машина

Следващите стъпки са промяна на конфигурационните файлове , на новата операционна система, както следва .

За ИП адрес и мрежова маска , създава се бекъп и се едитва следния файл , според въведените в конфигурационния файл настройки :

* /vmFS01/temp\_mount/etc/sysconfig/network-scripts/ifcfg-eth0

За шлюз , създава се копие и се порменя според настойките :

* /vmFS01/temp\_mount /etc/sysconfig/network

За днс адреси , създава се копие и добавят днс адресите от входния файл за програма към

* /vmFS01/temp\_mount/etc/resolv.conf

За създаване на сървър предоставящ спецефична услуга, се създава папка :

* /vmFS01/temp\_mount/root/updates

В нея се копират необходимите пакети , и конфигурационни файлове . Който се намират на линукс сървъра от който се изпълнява програмата в пака /root/cloud/ . Там има папка за всяка една от услугите , който програмта може да конфигурира – хостинг , днс и шеър :

dns

hosting

install\_scripts

share

Както и папка , наречена “install\_scripts” там се намира изпълнимия код , който по – късно , при стартиране на нашата виртуална машина , ще инсталира всички пакети , конфигурационни файлове .Както и ще направи , необходимите промени за това услугата да тръгва автоматично . Последната команда от този инсталационнен скрипт е рестарт.

Инсталационния скрипт ще бъде изпълнен само веднъж, а за неговото изпълнение ще отговаря самата операционна система на новата виртуална машина. Това се реализира , чрез редактирането на следния файл :

* /vmFS01/temp\_mount/etc/rc.local

В момента , когато пакетите са инсталирани и новият сървър е готов за рестартиране, точно преди това , нашата програма закоментира изпълнението на този скрипт.

## 2.4 Заключителни стъпки

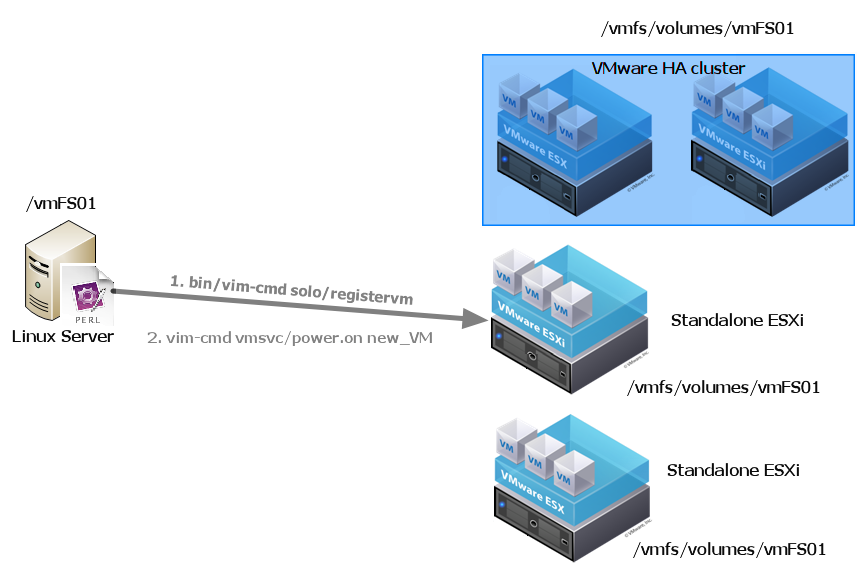
Към този момент , вече разполагаме с конфигурирана операционна система и приложение , чакащо инсталация. Остава да направим последните няколко заключителни стъпки .

Ената от тях е раз-монтирането на рут дяла на операционната система , която конфигурирахме в предходната стъпка . Това става отново , благодарение на същите две линукс програми – kpartx и losetup.

кpartx –d рут дяла, който сме монтирали

losetup –d на луп девайса , който сме използвали

Следващата и последна стъпка , която програмата за услуги в облака изпълнява е изпращането на команда към VMware хоста , да регистрира и стартира съответна виртуална машина.



Фиг. 7 регистриране и стартиране на новия сървър

Финалния етап , който се случва след стартирането на вируталния сървър , е изпълнение на install\_scrip - скрипта който сме копирали в същината на нашата програма. Тои се грижи , апликацията да бъда стартирана и конфигурирана. След изпълнението му виртуалната машина се рестартира .

В този момент вече имаме работеща услуга, периода необходим за конфигурирането и стартирнето средно около 6 минути. В зависимост от скороста на копиране на виртуалната машина от базовия образ. Тук нашата програма се връща отново в начално състояние и проверява за следващ съсрвър , който трябва да се предостави.

# 3.Софтуерно проектиране на системата.

## 3.1 Въведение

Проектирането на софтуерния продукт обхваща всички дейности свързани с реализирането на утвърдените изисквания посочени в документа Спецификация (технико-икономическо задание). Основните принципи при проектирането е изследване и разбиране на поставената цел, и декомпозирането й на подзадачи, който лесно могат да бъдат изпълнени.

Съществуват два основни подхода при проектирането на софтуер: функционален и обектно-ориентиран подход.

При функционалното проектиране софтуера се разглежда като централизирана система, състояща се от отделни компоненти, всеки от които реализира определена функция.

При обектно-ориентираното проектиране софтуера се разглежда като децентрализирана система, състояща се от отделни обекти, всеки от които управлява собственото си състояние. Обектите имат множество от атрибути, определящи състоянието им, и операции върху тези атрибути. Комуникацията между обектите се осъществява посредством набор от съобщения.

Етапите при функционалното проектиране биват два вида:

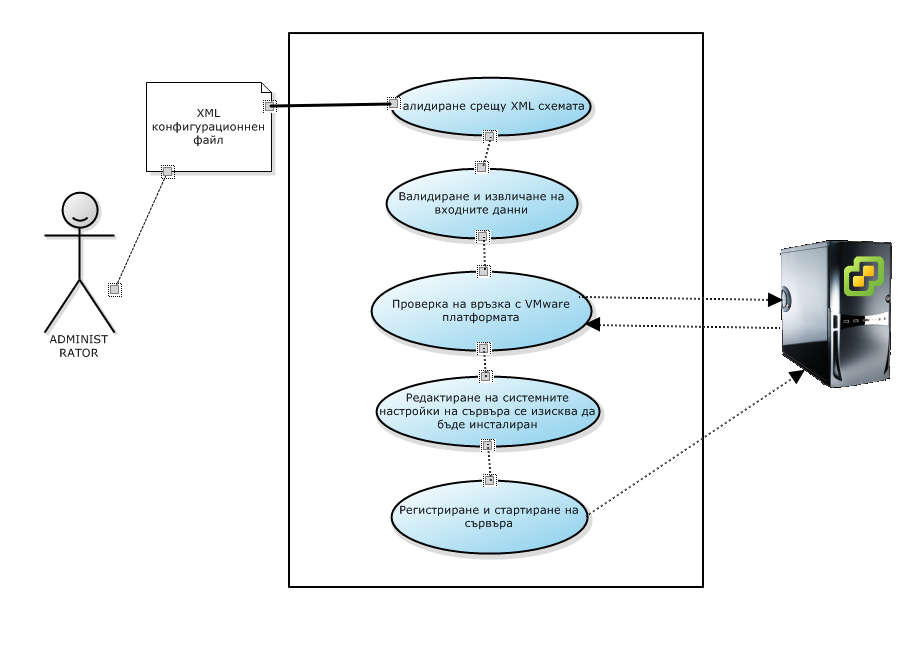
* предварително (външно) проектиране;
* детайлно (вътрешно) проектиране.

Като резултат от предварителното (външното) проектиране се създава външният (архитектурен) проект на софтуерния продукт, отразяващ цялостната му структура и начините на взаимодействие в нея, осигуряващи концептуална цялост на системата.

## 3.2Логическа организация на данните

Описание.

Ситемният администратор , попълва XML конифигурационния файл. Този файл съдържа две основни секции. Едната е VMware , в която се записват всички необходими настройки от гледна точка на VMware сървърите , който ще се използват . Втората секция е , тази съдържаща основните настройки на сървърите , който ще бъдат дистрибутирани. В примерния случай , ще кажем че администратора е въвел един VMware сървър и един сървър от тип DNS който иска да бъде инсталиран. Програмта взима като входен параметър този XML файл . Сверява конфигурационния файл срещу неговата схема . Втората стъпка , която програмата предпиема е да извлече данните , като същевременно прави детайлна проверка за вярност и ги записва в свой глобални променливи. Следващата стъпка е провка на връзката до сървъра предоставящ виртуализацията . В случай че нята така без акредитирана вързка , програмата предлага да направи. Следващата стъпка създаване на дисковете на новия сървър ( в нашия случай , под дискове се разбира файл който съдържа операционната система) , посредством клониране. Когато дискът на новия сървър е готов , почва редактиране на параметрите на опеарационната система. Последната сътпка е регистриране и стартиране на новосъздадия сървър върху VMware платформата. Описаният по горе потребителски случай може да се види на фигура по долу .



Фиг. UML user case

Необходими действия .

- Попълване на XML файла от администраотра

- Стартиране на програмата

Деиставащи лица

* Коммпютърен Администратор

Предусловия.

- ИП , системни акредитация до VMware сърър

- монтиране на споделен дисков масив , едновременно на VMware сървъра и на системата на която се изпълнява програмта

- инталиране на линукс инструментите kpartx и losetup на системата от която се изпълнява програмта

Цели

- автоматизирано **инсталиране и конфигуриране** на нов виртуален сървър

Условия за провал :

* XML конфигурационния файл не намерен в локацията описана в ръководстовто на потребителя. Файлът не се валидира успешно срещу XML схемата му.
* VMware ип-то не съдържа 4 октетека цифри , разделени с точка , който са в диапазона 0 – 255
* Споделения файлов сторидж не е монтиран според въведеното име в XML файла , във VMware сървъра
* Споделения файлов сторидж не е монтиран според въведеното име върху локалната система , от която се изпълнява програмата
* Името , типа и ИП настройките на виртуалния сървъра, който желаеам да бъде инсталиран и конфигуриран са не направилни
* При въведени грешни юзер и парола на VMware сървъра
* При не намиране на линукс инструментите kpartx и losetup

Стъпки на изпълнение :

1. Администратора въвежда ИП на VMware съръра , както и папка под която е монтиран споделения файлов сторидж . Въвежда също така име , тип , разновидноста на опеарационната система (темплейт ) и ип настройки на виртуалния съръвър който иска да бъде инсталиран .
2. Стартиран прорамата
3. Изпълнява се базова проверка срещу XML конфигурационния файл срещу XML схемата му

3.а XML схемата или самия XML файл не е намерен

3.б XML файла е попълнен неправилно или има променена от очакваното структора

4. XML файла е форматиран и структориран правилно

5. Извличане на данните за VMware сървара и локалната за системата папка от XML файла

6. Сверяване на данните за VMware сървъра

6.а ИП адреса на VMware сървъра не е валиден

6.б Името на директорията под която е монтиран споделения файлов сторидж за VMware сървъра не е валидно

6.в Валидно ИП и име на директорията

7. Сверяване на данните от XML файла за локалната папка под която е монтиран споделения файлов сторидж

7.а Локалната папка е навалидна

8. Локалната папка , под която е монтиран споделения файлов сторидж е наред

9. Извличане на данните за виртуания сървъра

10. Валидиране на Името , типа , ИП настройките на виртуалния сървър

10.а Не валидно име на виртуалния сървър

10.б Не валиден тип на виртуалния сървър

10.в Не валидни ИП настройки за виртуалния сървър

11. Коректни Име, Тип и ИП настройки за виртуалния сървър

12. Проверка дали има връзка до VMware сървъра

12.a Няма връзка до VMware сървъра

13. Има връзка до VMware сървъра

14. Има ли без акредитирана връзка до VMware сървъра (SSH keys)

14.а Не няма, програмата предлага да направи такава. Изисква се администратора да въведе еднократно юзер и парола.

15. Има без акредитирана връзкa (SSH keys)

16. Стартира се процес по клониране на виртуалния сървър от темплейт

17. Стартира се процес за конфигуриране на виртуалния сървър

18. Проверка за това дали са инсталирани линукс инструментите kpartx и losetup

18.a Линукс инструментите kpartx и losetup не са инсталирани

19. Линукс инструментите kpartx и losetup са инсталирани

20. Виртуалния сървър се регистрира на VMware сървъра

21. Виртуалния сървър се стартира

## 3.3 Структура на софтуерния продукт

Тук се разглеждат основните компоненти, външно-проявимите им свойства и отношенията между тях;

## 3.4 Интерфейс на софтуерния продукт

Детайлно се разглежда интерфейса между отделните части на софтуера; начина на взаимодействие между разработвания софтуер и други софтуерни системи, и не на последно място начина на взаимодействие между софтуера и потребителя.

При детайлното проектиране (component-level) софтуерната система се представя като йерархия от "черни кутии", които трябва да се реализират като обособени програмни части, наречени модули. Модулът е функционално обособен и стандартно оформен елемент, който е основна, самостоятелно разработена, тествана и документирана единица.

4.3. Архитектура на системата - отразява връзките и взаимодействието между програмните модули и данните.

4.4. Организация на данните - концептуален модел на БД.

4.5. Избор на език и среда за програмиране.

4.6. Реализация на програмната система:

4.6.1. Структура на данните - избор на ключови полета, тип и размер.

4.6.2. Описание на програмните модули - изпълнявана функция, интерфейс между отделните програмни модули, обобщен алгоритъм (блок-схеми или псевдокод).

4.6.3. Структура и организация на потребителския интерфейс.

4.7. Формат на входните документи - формален и логически контрол.

4.8. Формат на извежданите справки.

# 4.Описание на програмната реализация и особеностите на нейното приложение.

## 4.1 Структура на приложението

## 4.2 Реализация на функциите

### Функция validateConfig ($parsedXML)

### Функция extractGlobalVarible($parsedXML)

### Функция checkESXiConnection($esxi\_ip\_list, $esxi\_DS)

### Функция startCloning($server\_hash,$esxi\_ip\_list, $esxi\_DS, $l\_dir)

### Функция insertConfiguration($server\_hash, $l\_dir)

### Функция registerVMtoESXi($server\_hash, $esxi\_ip\_list, $esxi\_DS)

# 5.Ръководство за потребителя.

## 5.1 Локация на програмата

След инсталация всички необходими фаилове се намират в /root/cloud/ директорията. Тя съдържа самата програма и следните папки :

/conf/

/dns/

/hosting/

/install\_scripts/

/share/

/logs/

start\_cloud.pl

В първата папка „conf“ се намира конфигурациония файл и валидиращата схема. В папката “logs” се съдържат изходните логове от изпълнението на програмата.

Пакка “dns” съдържа , линукс инсталационните пакети за този тип сървър , както и конфигурационния му файл. Файлът съдържа , базови насторки , той се дописва автоматично от програмата на по – късен етап.

Пакка “hosting” съдържа , линукс инсталационните пакети за този тип сървър , както и конфигурационния му файл. Файлът съдържа , базови насторки , той се дописва автоматично от програмата на по – късен етап.

Папка „install\_scripts“ съдържа, скипт който се копира на ново създадената виртуална машина и прави първичната инсталация и конфигурация , при първото му стартиране.

Пакка “share” съдържа , линукс инсталационните пакети за този тип сървър , както и конфигурационния му файл. Файлът съдържа , базови насторки , той се дописва автоматично от програмата на по – късен етап.

Папка „logs“ съдържа , изходния код от изпълнението на програмата. Това е и първото място което трябва да се провери в случай на възникнала грешка .

Файлът „start\_cloud.pl“ е самата програма, с разширение .pl за да бъде ясно че става въпрос за езика на който е писана , а именно пърл .

Първата стъпка която потребителя трябва да предприеме е да премине в следната дирекетория : /root/cloud/conf

cd /root/cloud/conf

Вътре може да се видят два файла :

conf.xml

conf.xsd

Първият с разширение xml , е нашият конфигурационнен файл . Които ще трябва да редактираме , като втора стъпка.

Вторият файл е , валидационната схема. Той предварително попълнен и потребителя не трябва да се ангажира с него. Неговата е цел за валидация на пърия.

## 5.2 Конфигурационен файл

Редактиране на конифуграционния файл

vi conf.xml

Как изглежда една примерна конфигурация :



Потребителя трябва да попълни всички полета както е пояснено по-долу.

Следната таблица , показва елеменитите на конфигурационния файл :

|  |  |
| --- | --- |
| **Елемент** | **Описание** |
| VMware | Елемент определящ настройките за vmware hypervisor-ите . |
| VMware: ESXi\_list | Лист описващ всички ИП адреси на VMware сървърите , който ще се използват за вируализация. Самиме ИП-та трябва да бъдат разделени със запетая. |
| VMware: Datastore | Пълния път , до мястото от ESXi сървърите където е монтирано общото мрежово дата хранилище. |
| VMware: LocalDir | Това е директорията на линукс сървъра , която се използва за монтиране на общото мрежово дата хранилище. |

|  |  |
| --- | --- |
| **Елемент** | **Описание** |
| Server | Елемент oпределящ всички настройки за новия виртуален сървър , който програмата ще създаде. Имаите в предвид , че може да имаме не ограничен брой елементи ограничени с таг <Server> </Server> |
| Server: VMname | Име на виртуалната машина. Под това име ще се регистрира новия вървър на общотото дата хранилище , както и в VMware hypervisor-a |
| Server: Type | Тип на новия сървър. Може да избираме един от трите вида : **hosting** , **dns** или **share** |
| Server: Template\_name | Име на темплейта , който желаем да използваме за дистирибуцията на новия виртуален сървър. Към момента разполагаме само с **centos\_6.4\_x86\_64** |
| IP | ИП адрес , който ще използва новия вируталне сървър. Това е и адреса , на който ще ползваме желаната от нас услуга. |
| Netmask | Мрежова маска , на новия виртуален сървър |
| Gateway | Шлюз , за новия виртуален сървър |
| DNS | Днс ИП адрес. |
| DNS | Втори Днс ИП адрес. |

След като попълним , конфигурационния файл по желания от нас начин и съобразен с таблицата от горе. Можем да продължим със следващата стъпка , а именно изпълнение на програмта.

## 5.3 Изпълнение на програмата

За целта трябва да променим нашата текуща директория да бъде /root/cloud

cd /root/cloud/

След което да изпълним , програмата :

./start\_cloud.pl

Примерен изходен код , който се очаква да видим на екрана и във файл в лог директорията изглежда по този начин :

|  |
| --- |
| [root@linux-new script]# ./start\_cloud.pl |
| STARTED: ./start\_cloud.pl |
|  |
| Logs are written to:[ /home/script/logs/start\_cloud.log ] |
|  |
| [ /home/script/conf/conf.xml ] is well formed XML configuration file...proceeding. |
|  |
| STARTING FUNCTION: main::validateConfig |
|  |
| Defined schema file [ /home/script/conf/conf.xsd ] is well formed... |
|  |
| Validating XML against the Schema... |
|  |
| [ /home/script/conf/conf.xml ] validated successfully |
|  |
| STARTING FUNCTION: main::extractGlobalVarible |
|  |
| Validating the IP addresses for the ESXi . |
|  |
| ESXi IP address [ 192.168.0.152 ] is [ OK ]. |
|  |
| ESXi DataStore name is [ OK ]. |
|  |
| Local Mount point Directory is [ OK ]. |
|  |
| Validating the name BrainCoffee . Names must be unique. Program will now warn you about identical names ! |
|  |
| VM BrainCoffee is [ OK ]. |
|  |
| Validating the Type for and Template for BrainCoffee . |
|  |
| VM BrainCoffee Type [ share ] is [ OK ]. |
|  |
| Template [ centos\_6.4\_x86\_64 ] could be found in [ /vmFS01 ]. |
|  |
| Validating the IP addresses for the VM BrainCoffee . |
|  |
| VM BrainCoffee IP address [ 192.168.0.105 ] is [ OK ]. |
|  |
| VM BrainCoffee IP address [ 255.255.255.0 ] is [ OK ]. |
|  |
| VM BrainCoffee IP address [ 192.168.0.1 ] is [ OK ]. |
|  |
| VM BrainCoffee IP address [ 192.168.0.1 ] is [ OK ]. |
|  |
| VM BrainCoffee IP address [ 78.90.90.1 ] is [ OK ]. |
|  |
| Validating the name Linux\_Test . Names must be unique. Program will now warn you about identical names ! |
|  |
| VM Linux\_Test is [ OK ]. |
|  |
| Validating the Type for and Template for Linux\_Test . |
|  |
| VM Linux\_Test Type [ Hosting ] is [ OK ]. |
|  |
| Template [ centos\_6.4\_x86\_64 ] could be found in [ /vmFS01 ]. |
|  |
| Validating the IP addresses for the VM Linux\_Test . |
|  |
| VM Linux\_Test IP address [ 192.168.0.106 ] is [ OK ]. |
|  |
| VM Linux\_Test IP address [ 255.255.255.0 ] is [ OK ]. |
|  |
| VM Linux\_Test IP address [ 192.168.0.1 ] is [ OK ]. |
|  |
| VM Linux\_Test IP address [ 192.168.0.1 ] is [ OK ]. |
|  |
| VM Linux\_Test IP address [ 78.90.90.1 ] is [ OK ]. |
|  |
| STARTING FUNCTION: main::checkESXiConnection |
|  |
| Check ESXi for connection. |
|  |
| ESXi IP [ 192.168.0.152 ] is pingable. |
|  |
| Check ESXi IP [ 192.168.0.152 ] for installed SSH Keys. |
|  |
| Remote Host [ 192.168.0.152 ] is accessible with SSH keys. |
|  |
| Last check DataStore [ /vmfs/volumes/vmFS01 ] on host [ 192.168.0.152 ]. |
|  |
| DataStore [ vmFS01 ] is Presented on host [ 192.168.0.152 ]. |
|  |
| STARTING FUNCTION: main::startCloning |
|  |
| INFO: Cloning a virtual disk, this may take a while... |
|  |
|  |
| INFO: Cloning a virtual disk, this may take a while... |
|  |
|  |
| STARTING FUNCTION: main::insertConfiguration |
|  |
| Check if Losetup is installed |
|  |
| Check if Kpartx is installed |
|  |
| Mounting Virtual Disk of VM [ Linux\_Test ] |
|  |
| Find first free loop device [ /dev/loop0 ] |
|  |
| Start Mounting Process ... [ Linux\_Test ] |
|  |
| Read partition table for [ Linux\_Test ] |
|  |
| LVM staus for [ Linux\_Test ] . |
|  |
| Apply Network settings. |
|  |
| Clear Udev Rules |
|  |
| Appling Type [ hosting ] |
|  |
| Mounting Virtual Disk of VM [ BrainCoffee ] |
|  |
| Find first free loop device [ /dev/loop0 ] |
|  |
| Start Mounting Process ... [ BrainCoffee ] |
|  |
| Read partition table for [ BrainCoffee ] |
|  |
| LVM staus for [ BrainCoffee ] . |
|  |
| Apply Network settings. |
|  |
| Clear Udev Rules |
|  |
| Appling Type [ share ] |
|  |
| STARTING FUNCTION: main::registerVMtoESXi |
|  |
| Registering VM [ Linux\_Test ] to ESXi Host [ ESXi ] with vmid [ 518 ] . |
|  |
| Power on VM [ Linux\_Test ] |
|  |
| Registering VM [ BrainCoffee ] to ESXi Host [ ESXi ] with vmid [ 519 ] . |
|  |
| Power on VM [ BrainCoffee ] |
|  |
| Program Finished Succesfully !! |

## 5.4 Проверка на сървърите и услугите

Като проверка , на това дали програмта за услуги в облака е изпълнена правилно и резултата е този който очакваме , може да проверим по следния начин .

За успешното приключване на самата програма , трябва предходната стъпка да е завършила с този ред : „Program Finished Succesfully !!“ Както е и добра практика да разгледате подробно изходящия код , така че да няма грешки .

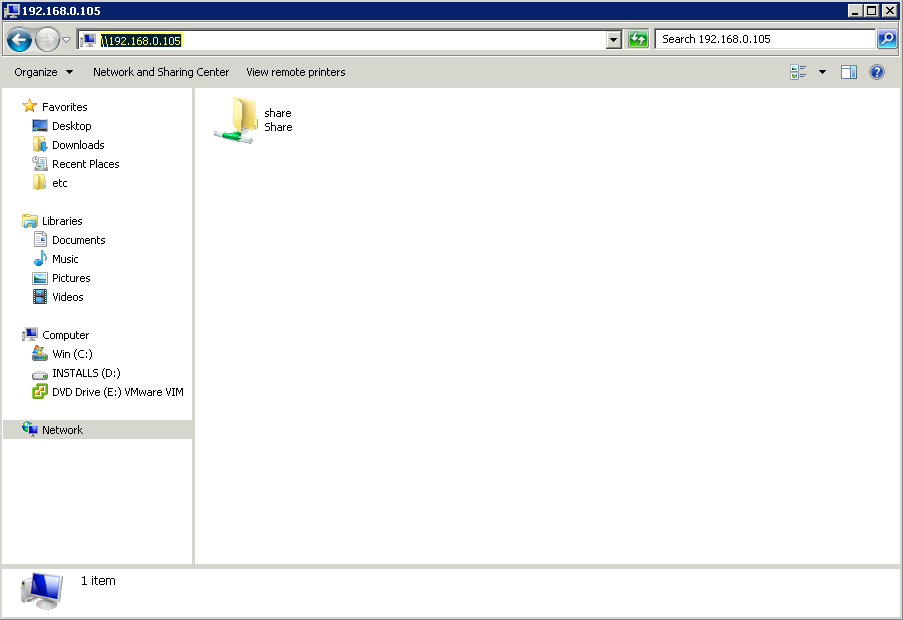
Проверката на новите сървъри става според услугата , за която са конфигурирани . Използва се ИП адреса на който сме задали за същия сървър в конфигурационния файл. Примено от изпълнената програма може да видим че сървър с ИП 192.168.0.105 се използва за файлово споделяне (share) а пък втория с ИП 192.168.0.160 се използва за хостване на уеб страници.

Проверката на сървъра за файлово споделяне става като се въведе , ИП адреса му в Уиндолс Експлорера , като точно преди това сме въвели , две обратно наклонени черти . Накратко това което трябва да въведем е [\\192.168.0.105](file:///\\192.168.0.105) . След оторизиране с базовия създен юзер и парола :

User : **sambatest**

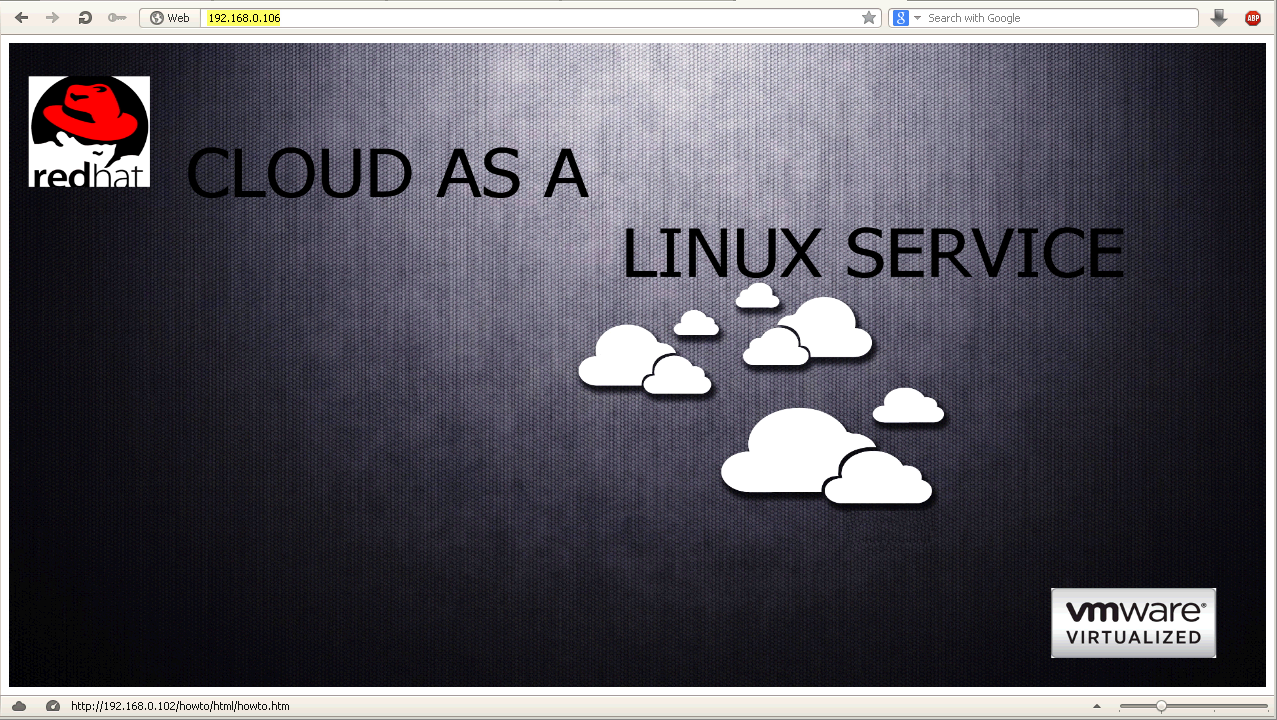
Password : **test123**

Може да видим следния екран. В който вече може да работим.



Фиг. 5.1 проверка на услугата фаилово споделяне

Проверката на сървъра за уеб споделяне става посредством браузер. За да се осъществи самата проверка е необходимо да въведем ИП адреса на сървъра в полето за уеб адреси:



Проверката на сървър от тип Днс , става последством използването на линукс пакета с днс инструменти – bind-utils. След инсталацията му , може да направим следната проверка :

В случай , че проверкта е от същия хост , на който сме направили инсталацията :



В случай , че проверката е от друг хост :



## 5.5 Потребителски идентификационни данни за достъп

За достъп до ново създадените сървъри използвайте следните данни :

**Потребителско име : root**

**Парола : parola123**

# 6.Инсталиране на системата.

## Изтегляне на архива.

Програмата се разпространява като архив с разширение tgz. Името на файла е **cloud.tgz** .

Поставете инсталационния диск , дистрибутиран с тази дипломна работа, в сд дисковото устройство от на сървъра , от който ще се изпълнява програмта. Логнете се сървъра от който ще се изпълнява програмта и копирайте архива в папка /root .

mount –t iso9660 –o ro /dev/cdrom /mnt

cd /mnt

cp cloud.tgz /root/



Уверете се че файла е копиран.

cd /root

pwd – изходния код от тази команда трябва да покаже текущата ни директория , в която се намираме а именно /root

ll – това кратък вариянт на командата ls –la . С нея проверяваме далипрорамата ни е копирана успешно.



## 6.2 Разархивирайте файла

За раз-архивиране на файла се използва следната команда :

tar –zxvf cloud.tgz



## 6.3 Инсталирайте на необходимите пакети

Всички необходими пакети , за изпълнението на програмата се намират в /root/cloud/prerequisite . Tова са пети за програмния език пърл, негови библиотеки , както няколко пакета който позволяват четене на XML файлове от пърл. Инсталацията се извършва по следния начин :

cd /root/cloud/prerequisite

rpm –Uvh \*





С това инсталацията на нашата програма е приключила успешно. Можем да продължим към нейното изпълнение. Моля прегледайте глава 5. Ръководство на потребителя.