**BLADELOGIC DEPLOYMENT**

1. Take the Backup’s
2. Ensure that the required amount of disk space is available(manually)
3. Manual DB deployment
4. Login to the bladelogic tool using the url
5. Login to the bladelogic console and select specific environment (Stage/Production) using the credentials
6. Under the deploy folder we can see the packages which contain the necessary tar’s for each type of deployment i.e.(App, Web, Merch)
7. We will add jobs like and create a batch job

eShop\_undeploy

merch\_undeploy

eShop\_release

eShop\_deploy

start\_eShop\_servers etc.,

1. Select the targets i.e, server names(app servers, web servers etc)
2. Right click on the batch job and select “Execute”
3. Check the logs for errors or successful deployment.

**JMS**

1. We have configured an internal JMS server using weblogic console
2. We have configured Queues for communication(Sendemail, processOrder,preauthAccepted)
3. The middleware team also has one JMS server configured at their end
4. The communication between us and them is basically for three TIL calls namely

* Match Address
* CheckBankAccount
* SendEmail

1. Correlation ID’s will be sent to the Middleware team if we encounter any time outs

**JDBC**

1. We have created several datasources in order to connect to the database.
2. We have configured the datasources by specifying the datasource name, IP address, port number, SID, username, password and the target.
3. We have set the initial capacity of the datasource to the maximum value which is 25 for optimal performance.

**Environment Details**

1. We have two load balancers
2. Three web servers which are clustered
3. Four Unix boxes, each containing multiple server instances
4. Altogether there are 13 servers out of which

* One Admin server
* 8 servers serve as managed servers
* 4 servers serve as utility servers(Lock Manager, Backup Lock Manager, Scenario Manager and Reporting server)

1. The 8 managed servers are clustered
2. We have two DB servers which are also clustered

**Softwares Used**

* Weblogic 10
* Oracle 10g
* ATG 9.1
* Apache Web Server 2.2.8
* Java Version - 1.6.0-25-b06
* JVM Version - R27.6.0

**Configuration Settings**

* Total RAM 16GB per Unix Box
* Initial thread pool size = 25
* Maximum thread pool size = 25
* To process an order we make use of “Process order” web service which is located at the DSL end
* For processing the order the DSL team uses “UpdateOrderStatus” and “UpdateOrder”

**JVM Tuning**

* GC strategy employed is “**Parallel**” and the mode is “**Throughput”** mode(default)
* -XX:+HeapDumpOnOutOfMemoryError is set to take the heap dump as and when outofmemory errors are encountered in the logs
* -Xms = 2048m (min = 256MB)
* -Xmx = 2048m (min = 512MB)
* -Xns = 40m
* -Xss = 1024KB
* -XXtlaSize:min=4KB
* -XXtlaSize:preferred=256KB (Preferred size = min to the power of min TLA size i.e 4^4)

**Deployment Activity Using Scripts**

**Pre-Release Activities:**

* We copy the tar files into a server(Gateway server)
* We’ll check for the free space, if available in sufficient amount then we take the Application, Web and DB server backup’s
* If not, we’ll do housekeeping activity and remove the old tar files to free the space
* As part of application server backup, we take the backup of “eShop” folder
* As part of DB server backup, we take the backup of the DB schemas(2 Catalogue/Content,

2 Publishing and 1 UP/User schema, 1 Merch and 1 CA schema)

* As part of the Web server backup, we take the backup of “images and docs”
* Now, we copy the application tar file into one of the app server boxes(aukesaws)
* Then, we copy the web tar file into each web server(3 web servers)
* And finally, we copy the DB zip file to DB server and unzip it
* With this the pre-release activities are completed
* We will wait till the scheduled release time for starting the release activity

**Post-Release Activites:**

* We put up the holding page by using a script located in the web server boxes

(cd /opt/apache-2.2/conf/eShopprod >. ./holdingpageupmain.sh)

**DB Release**

* Now, we log onto each of the DB schemas and execute the scripts provided by the development team

**Application Release**

* Set the environment
* Undeploy
* Release
* Secure Copy
* Start all managed servers and check logs
* Deploy ear and check logs

**Web Release**

* Untar the tar files
* Restart the web servers
* We’ll remove the Holding Page

(cd /opt/apache-2.2/conf/eShopprod>. ./holdingpagedownmain.sh)

**Managing WebLogic's EJB Container**

WebLogic's **EJB container** controls the life cycle of an EJB object and provides the runtime environment for deployed EJB components. The EJB container allows a client to obtain the EJB's home object, either through a JNDI lookup, via an EJB reference defined during deployment, or by using the EJB home handle. It provides EJBs with access to a wide range of container-managed services such as the local ENC, transactions, security, persistence, concurrency, locking, caching, clustering, and session-state replication. These services can be configured for a particular EJB component through the deployment descriptors associated with it.

**Web Container**

Weblogic’s **Web container** (also known as a Servlet container) is the component of a [web server](http://en.wikipedia.org/wiki/Web_server) that interacts with [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) [servlets](http://en.wikipedia.org/wiki/Servlet). A web container is responsible for managing the lifecycle of servlets, mapping a [URL](http://en.wikipedia.org/wiki/URL) to a particular servlet and ensuring that the URL requester has the correct access rights. A web container implements the web component contract of the [Java EE](http://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition) architecture, specifying a [runtime environment](http://en.wikipedia.org/wiki/Runtime_environment) for web components that includes [security](http://en.wikipedia.org/wiki/Computer_security), [concurrency](http://en.wikipedia.org/wiki/Concurrency_(computer_science)), [lifecycle management](http://en.wikipedia.org/wiki/Product_lifecycle_management), [transaction](http://en.wikipedia.org/wiki/Transaction_processing), deployment, and other services. A web container provides the same services as a [JSP](http://en.wikipedia.org/wiki/JavaServer_Pages) container as well as a federated view of the Java EE platform APIs.

**About IBM**

* International Business Machines (**IBM**)
* Founded in June 16, 1911
* Founder Charles Flint
* In 2012, Fortune ranked IBM as
  + - Second largest firm in US in terms of number of employees
    - Fourth largest firm in terms of Market Capitalization
    - Number 1 company for leaders(Fortune)
    - Best Global Brand
    - Most Innovative Company
    - It’s employees have won five Noble Prizes
* It offers Computer hardware, Computer software, IT services and IT consulting services
* Famous inventions by IBM include the [**automated teller machine (ATM)**](http://en.wikipedia.org/wiki/Automated_teller_machine)**, the** [**floppy disk**](http://en.wikipedia.org/wiki/Floppy_disk)**, the** [**hard disk drive**](http://en.wikipedia.org/wiki/Hard_disk_drive)**, the** [**magnetic stripe card**](http://en.wikipedia.org/wiki/Magnetic_stripe_card)**, the** [**relational database**](http://en.wikipedia.org/wiki/Relational_model)

**About Oracle**

* **Oracle Corporation** is an American [multinational](http://en.wikipedia.org/wiki/Multinational_corporation) computer technology corporation that specializes in developing and marketing [computer hardware](http://en.wikipedia.org/wiki/Computer_hardware) systems and [enterprise software](http://en.wikipedia.org/wiki/Enterprise_software) products – particularly [database management systems](http://en.wikipedia.org/wiki/Database_management_system).
* Founded in June 16, 1977
* Founders:[Larry Ellison](http://en.wikipedia.org/wiki/Larry_Ellison), [Bob Miner](http://en.wikipedia.org/wiki/Bob_Miner), [Ed Oates](http://en.wikipedia.org/wiki/Ed_Oates)
* Headquarters: California, US
* Oracle is currently employing approximately 115,166 people worldwide as of 30 June 2012 (2012 -06-30)[[update]](http://en.wikipedia.org/w/index.php?title=Oracle_Corporation&action=edit)
* Oracle is the [third-largest software maker](http://en.wikipedia.org/wiki/List_of_the_largest_software_companies) by revenue, after [Microsoft](http://en.wikipedia.org/wiki/Microsoft) and [IBM](http://en.wikipedia.org/wiki/IBM).[[4]](http://en.wikipedia.org/wiki/Oracle_Corporation#cite_note-3)
* The company also builds tools for [database](http://en.wikipedia.org/wiki/Database) development and systems of middle-tier software
* Oracle acquired companies like
  + - BEA Systems
    - Sun Microsystems
* Products such as
  + - Oracle SQL Developer
    - MySQL
    - PL/SQL
    - RDBMS

**JMS Tuning**

**Flow Control Threshold Parameters**

* **Bytes/Messages Threshold High:**

* When the number of bytes/messages exceeds this threshold, the JMS server/destination becomes armed and instructs producers to limit their message flow.
* **Bytes/Messages Threshold Low:**

* When the number of bytes/messages falls below this threshold, the JMS server/destination becomes unarmed and instructs producers to begin increasing their message flow.
* Flow control is still in effect for producers that are below their message flow maximum. Producers can move their rate upward until they reach their flow maximum, at which point they are no longer flow controlled.

**What Types of Objects Can Be Clustered?**

* + - Servlets
    - JSPs
    - EJBs
    - Remote Method Invocation (RMI) objects
    - Java Messaging Service (JMS) destinations
    - Java Database Connectivity (JDBC) connections
* **Data Sources**—In a cluster, the clients must obtain connections through a JDBC data source on the JNDI tree. The data source uses the WebLogic Server RMI driver to acquire a connection. The cluster-aware nature of WebLogic data sources in external client applications allows a client to request another connection if the server instance hosting the previous connection fails. Although not strictly required, BEA recommends that server-side clients also obtain connections via a data source on the JNDI tree.
* **Connection Pools**—Connection pools are a collection of ready-to-use database connections. When a connection pool starts up, it creates a specified number of identical physical database connections. By establishing connections at start-up, the connection pool eliminates the overhead of creating a database connection for each application. BEA recommends that both client and server-side applications obtain connections from a connection pool through a data source on the JNDI tree. When finished with a connection, applications return the connection to the connection pool.
* **Multipools**—Multipools are multiplexers for basic connection pools. To the application they appear exactly as basic pools, but multipools allow you to establish a pool of connection pools, in which the connection attributes vary from connection pool to connection pool. All of the connections in a given connection pool are identical, but the connections in each connection pool in a multipool should vary in some significant way such that an expected failure of one pool will not invalidate another pool in the multipool.

**What Types of Objects Cannot Be Clustered?**

The following APIs and internal services cannot be clustered in WebLogic Server:

* + - File services
    - Time services
    - WebLogic Events (deprecated in WebLogic Server 6.0)
    - Workspaces (deprecated in WebLogic Server 6.0)

**Differences between Weblogic and Websphere**

* Weblogic was introduced by BEA and Websphere was introduced by IBM
* Functionally these two products are fairly close except for some minor differences in supported standards
* WebLogic is much easier to use than Websphere.
* Weblogic outperforms Websphere in terms of
  + - Transactions per seconds
    - Memory usage
    - Database usage
    - Thread management and
    - Monitoring
* Weblogic is well documented where as Websphere is very poorly documented and very difficult to understand.
* In WebLogic to configure a simple database connection factory it took less than 5 minutes whereas in case of WebSphere app server, it took more than 2 days to do the same (For a new user)
* WebSphere tends to focus more on integration, connectivity, and web services
* WebLogic focuses more on emerging J2EE standards and ease-of-use
* In Weblogic there is hard deployment where as in Websphere there is n’ t.
* Websphere is slow, big and buggy. The releases cause side effects, bad performance, horrible maintenance. The web interface is absolutely slow.
* In Websphere the server startup time is extremely faster (<5 mins)
* Though there are migrations from WLS to WAS and vice versa, the majority of the users tend to say that Weblogic is better than Websphere.

**Differences between JRockit and Sun JDK JVM’s**

* JRockit JVM is divided into two generations namely
* Young Generation and
* Old Generation
* JRockit is used in **Production mode**
* Sun JDK JVM is divided into
* Young/Eden Generation
* Survivor Space
* Tenure/Old Generation
* Sun JDK is used in **Development mode**
* **Sun JDK uses interpreter (Interpreter and JIT in previous releases) –** In this mechanism, the byte code is read and then translated into machine language, but these results are not saved in the memory. So every time even if the same method is run again and again, the JVM has to translate the code into machine language. This means machine code will not be reusable as it is not saved anywhere in the memory.
* **Oracle JRockit uses only JIT compiler (Just In Time) –** JIT mechanism means, once a method is run, the byte code is translated into machine language and this is saved in the memory. This means if the method is run again, there is no need for translation and the machine code is reused.
* Because of the interpreter mechanism used by sun JDK, the startup time for the server is faster because it does not have to save the machine code in memory. Once the translation is done for a method, it moves to the other one. Where as oracle JRockit saves the code, which is why start up takes longer. For the same reason, oracle JRockit uses more memory than sun JDK.
* In the long run, JRockit gives a slightly better performance as compared to sun jdk.
* Oracle JRockit optimizes the code. It identifies the HOT SPOTS which means the methods that are being run more often. These methods are then queued up for optimization. This code is then optimized which improves performance. Many issues are seen because of the code optimization mechanism because it is a complex procedure. Optimization can be disabled.
* JIT is also used by Sun JDK, but that was in the earlier versions. The Java Hotspot VM removes the need for a JIT compiler in most cases.

**Memory spaces in JDKs:**

* Sun JDK has the following memory spaces: Eden space, survivior space, tenured generation. The objects move from one space to another according to its age and survival from garbage collection.
* JRockit has 2 spaces, young generation and old generation, it uses the same mechanism of garbage collection. There is nothing called as permanent generation in JRockit.

**Memory and other JVM tunings:**

* JRockit gives advanced JVM tunings. From the release R26 and above, JRockit takes care of few tunings by itself. For example if there is an outofmemory occuring on the native TLA in previous releases due to insufficient TLA size which is 2k by default, in later releases the JRockit tunes these settings as per the requirement of the application. This has to be done and taken care of by the user in case of sun jdk. But then it is always better to be in a safer side it is recommended to have the tunings done by self.

**JVM Crashes:**

* When JRockit crashes, a JRockit dump is produced which basically has the reason for the crash. JRockit uses native libraries by default. This can be disabled by disabling the NativeIO from the admin console. The most common reason for the JRockit crash is the conflict between native libraries. For example, the jdbc type 2 drivers which use native libs. It is recommended to use type 4 pure java drivers when using oracle JRockit. Another reason for the crash can be code optimization because of its complexity. The stack trace in the JRockit dump will show the exact cause. When the JVM crashes, it is important to test it again by disabling code optimization and check if the issue still persists.
* A sun jdk crash produces hs\_err\_pid file which has the root cause of the crash. There can be several reasons for sun jdk crash, due to bugs in them (defects in the code of the jdk). These issues need to be reported to the sun team.
* **Tools for performance tracking:**
* Sun jdk that comes bundled with weblogic server gives tools like JConsole which can be used for performance tracking and monitoring the memory in use of the JVM. This tool is very much necessary so that each and every detail about the memory being used by the application, cpu usage, memory leaks can be identified.
* Oracle JRockit has a much more advanced tool JRMC (JRockit mission Control) which gives advanced tracking features. JRA recordings can be taken which gives each detail about the JVM arguments, garbage collection details, methods using the maximum memory etc. The memory leak detector tool in JRMC is also one important and very helpful tool. These make it easy for the user and administrators to maintain a record and identify the issues with the application and the JVM.

**How to configure Horizontal Clustering?**

* Let us take the case where we have to configure **Horizontal Clustering** on two machines
* I will install weblogic on the first machine and create a domain
* The domain will contain an Admin Server by default
* Start the NodeManager utility
* In the Weblogic Console, navigate to **Environment-->Servers--> Admin Server-->SSL Configuration Tab-->Advanced >set “hostname verification” attribute to “none”**.
* To make the changes to take effect we will restart the Admin Server.
* I will create two managed servers and group them into a cluster
* I will add two more managed servers to the above cluster using the second machine’s hostname/IP address in combination with distinct port numbers
* I will configure two “machines” namely machine1 and machine2 for the two machines respectively
* I’ll configure machine1 and machine2 by setting the IP address/hostname and the SSL port of the first machine and second machine respectively
* I will use the **pack** command to create a subset of the domain present in Machine1 which can be used to create a managed server domain directory in Machine2 using the below syntax
* Open command prompt and navigate to the following path and execute the pack command

C: \bea\wlserver\_10.3\common\bin

**Syntax**

pack.cmd -domain=domain-path -template=template-path -template\_name=”template name” -managed=”true”

* A jar file will be created in the template path provided
* We’ll copy the tar file into the remote machine using SCP (in UNIX) and store it in any location
* Open command prompt and navigate to the following path and execute the unpack command

C: \bea\wlserver\_10.3\common\bin

**Syntax**

Unpack.cmd -template=template-path -domain=domain-path [-username=username]

[-password=password] [-server\_start\_mode=dev|prod]

* Start the Admin server and managed servers using Node Manager

**Note:**

By default, WebLogic is installed and configured with the DemoIdentity.jks keystore. This certificate and key are created by utils.CertGen with the default options of containing only the host name in the common name field (cn), and not the fully-qualified DNS name. As a result, attempts to establish SSL connections may fail in some situations due to a host name verification exception. When using the demo certificates in a multi-server domain, managed server instances will fail to boot if they cannot establish an SSL connection with the Admin Server and the a BAD\_CERTIFICATE error is generated. This error occurs because the host name verifier, which is enabled by default in all WebLogic domains and which is used during the SSL handshake, compares the value of the cn field in the certificate with the fully-qualified DNS name of the SSL server that accepts the SSL connection. If these names do not match, the SSL connection is dropped.

**SUDO Command**

* The **sudo** command is frequently used to execute a command that requires root privileges. For example, if you want to run a shell script setup.sh that installs a program into a directory that only root can modify you can use sudo as follows:

**sudo -u root ./setup.sh**

* You may be required to enter the root password before the command is executed. After you have logged in, you can continue to execute commands through sudo for a few minutes without having to specify the login (-u root) with every command. If possible, it is better to do your regular work using an account with restricted privileges to avoid causing serious damage to the system by accident.

su = switch user

sudo = do single command as root

* su gives the whole environment as you would log in as root. It has no restrictions whatsoever and needs the root password to function.
* sudo on the other hand uses the currently logged user's password and obeys the restrictions put in place in the file etc/sudoers. It can also log the commands and if the commands were allowed or denied.

**In how many ways can we take Heap dump and Thread Dump?**

**Thread Dump:**

1. ps – ef | grep <managed name> to get the PID

KILL -3 <PID>

1. Weblogic.Admin -Adminurl =adminurl -username= username -password= password Thread\_Dump
2. Using WLST
3. Using Admin Console (**Environment->Servers->Monitoring->Threads->Dump Thread Stacks** ) and (**Environment->Servers->Monitoring->Performance->Dump Thread Stacks** )
4. Jstack –l <PID>
5. Using jvisualvm
6. Using jrcmd

C:\bea\jrockit\_160\_22\_D1.1.1-3\bin>jrcmd <pid> print\_threads > c:\temp\mythreaddump

C:\bea\jrockit\_160\_22\_D1.1.1-3\bin>jrcmd <pid> print\_threads

**Heap dump:**

1. Using jps, we’ll get the PID then

Jmap –dump file=filename.bin <PID>

1. Using jvisualvm
2. Jstat
3. Using JRCMD ->Jrockit\_home/bin/jrcmd <PID> hprofdump filename=heapdump.hprof
4. –XX:+HeapDumpOnOutOfMemoryError can be added to the startup script to generate the heap dump

**JRCMD Command**

* It is a JROCKIT command line tool used to monitor and manage JVM’s run-time behavior.
* It can be used to generate the thread dump as well as the heap dump.
* 1. The below command will redirect the output to an output file named mythreaddump

C:\bea\jrockit\_160\_22\_D1.1.1-3\bin>jrcmd <pid> print\_threads > c:\temp\mythreaddump

* 2. The below command will display the output on the console itself

C:\bea\jrockit\_160\_22\_D1.1.1-3\bin>jrcmd <pid> print\_threads

**weblogic.Deploy Utility**

C:\bea\user\_projects\Domains\Shop>java weblogic.Deployer -url <http://localhost:7001> -username weblogic -password infy123 -stage -deploy deployment\_name -targets AdminServer/servername/JMS servername/VirtualHost <name path of the ear>

**weblogic.Admin Utility**

C:\bea\user\_projects\Domains\Shop>java weblogic.Admin -url <http://localhost:7001>-username weblogic -password infy123 VERSION/THREAD\_DUMP/LICENCES/SHUTDOWN/GETSTATE -target server\_name

**Differences between Production Mode and Development mode**

**Development Mode:**

* Default JDK is Sun HotSpot in this mode
* Server starup time is faster when compared to JROCKIT
* Admin Server uses an automatically created boot.properties during startup
* Log file rotation is based on server restart
* Default connection pool size is 15
* Log files roll over at 5KB
* Log file will be started in a new file
* Auto-Deploy feature is enabled by default
* In development mode, the demo certificates can be used for SSL
* Fast swap deployment feature is available

**Production Mode:**

* Default JDK is Oracle JRockit
* Server startup time is not as fast when compared to Sun JDK
* Log file rotation is based on the log file size
* Default connection pool size is 25
* Log files roll over at 5MB
* Log file will not be started in a new file
* Auto-Deploy feature is disabled
* If you use the demo certificates for SSL a warning is displayed
* Fast swap deployment feature is not available

**Fast Swap Feature**

* Fast Swap feature is available only in the development mode.
* It speeds up the development cycle
* Over the course of development, developers will be continuously modifying, deploying and testing the applications
* It allows new methods to be added to the existing classes at run time and test the result immediately
* Weblogic server makes this possible by dynamically redefining classes by the class loader i.e , the application will be recompiled on the fly without redeploying it. (Set **compile-on-save** in the IDE)
* Deployment can be done only in the exploded directory format
* The feature only picks up the changes that are made to the class files in the exploded directory

**Migratable Target**

A Migratable Target is nothing but a weblogic server onto which the failed JMS Server along with it’s services or only the pinned services will be migrated to.

**Server Migration/Whole Server Migration**

Whenever a weblogic server hosting the JMS server goes down, the JMS server in it’ s entirety will move to one of the migratable servers available in the cluster. If migratable servers are not configured then the migration will take place in any one of the available servers in the cluster automatically

**Service Migration/Pinned Service Migration**

WebLogic Server allows the administrator to migrate pinned services from one server to another in the cluster, either in response to a server failure or as part of regularly-scheduled maintenance. This capability improves the availability of pinned services in a cluster, because those services can be quickly restarted on a redundant server should the host server fail.

In this release, migration is supported only for JMS servers and the JTA transaction recovery services. The JMS and JTA services are referred as migratable services, because you can move them from one server to another within a cluster.

Automatic migration is not allowed in the case of Service Migration

**How will you test connection pool?**

Services->DataSources->Monitoring->Select the DataSource->Testing->Select the target server->Click “Test Data Source”

**Capacity planning**

The process of determining what type of hardware and software configuration is required to meet application needs adequately is called capacity planning.

As part of the capacity planning we collect several metrics like

* CPU Usage and
* Memory Usage for all the managed, web and DB servers
* Number of hits during a particular time period (8AM to 10PM BST) during the first week of every month

## JRE (Java Runtime Environment)

Java Runtime Environment contains JVM, class libraries, and other supporting files. It does not contain any development tools such as compiler, debugger, etc. Actually JVM runs the program, and it uses the class libraries, and other supporting files provided in JRE. If you want to run any java program, you need to have JRE installed in the system

## What is the difference between JDK and JRE?

* If you need to execute any java program we need JRE ("Runtime Environment").
* If you need to compile some java code we need JDK ("Java Development Kit").
* One more thing to be noted is JRE will always be shipped with JDK because JDK compiles the code, so if we want to execute the compiled code we need JRE to do so.
* The "JDK" is the Java Development Kit. i.e., the JDK is bundle of software that we can use to develop Java based software.
* The "JRE" is the Java Runtime Environment. i.e., JRE is an implementation of the Java Virtual Machine which actually executes Java programs.
* Typically, each JDK contains one (or more) JRE's along with the various development tools like the Java source compilers, bundling and deployment tools, debuggers, development libraries, etc.

## Two way SSL Authentication

Two way SSL is used for the communication between eShop and DSL as part of the order fulfillment process. SSL or Secured Socket layer is used for the communication between the web server and any browser or client. As both the entities acts as both server and client, two way SSL has been used to ensure security.

## Certificate Creation

* DSL creates its private key using Open SSL utility of Apache web server by firing the following command in command prompt of OpenSSL.

Open a command prompt.

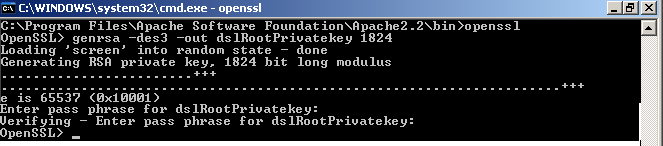
* Navigate to the directory where OpenSSL utility is present.
* Type OpenSSL and press enter.
* Type the following command and press enter.

**genrsa –des3 –out dslRootPrivatekey 1824**

where

dslRootprivatekey is the name of the private key

User has to provide the password for the key



* Using OpenSSL, DSL creates root certificate

For this, in the same command prompt of OpenSSL, type the following command and press enter.

**genrsa -des3 -out dslRootPrivate.key 1024**

**req -new -x509 -days 365 -key dslRootPrivate.key -out dslRootCA.crt**

Where dslRootCA.crt is the root certificate created.

It will create the root certificate of DSL.

* At eShop end, we create the keystore with the private key in it using keytool with the following command.

**keytool -genkey -keyalg RSA -alias mykey -keystore client\_keystore.jks**

where

mykey : the generated private key

client\_keystore.jks : the generates keystore

User should enter the passwords for both keystore and private key.

* With the keystore, eShop creates CSR ( client signing request ) and that will be sent to DSL. The command to create CSR is as follows.

**keytool -v -certreq -alias mykey -file client.csr -keypass vodafone -keystore client\_keystore.jks -storepass Vodafone**

where

mykey : name of the private key

client.csr : CSR file created

keypass : password of the private key

storepass : password of the keystore

client\_keystore.jks : keystore file

* DSL will sign the CSR and with the root certificate, creates the client certificate and sends it back to eShop. The command is

**x509 -req -days 365 -in client.csr -CA dslRootCA.crt -CAkey dslRootPrivate.key -set\_serial 01 -out shop.crt**

where

client.csr : CSR file sent by eShop to DSL

dslRootCA.crt : root certificate of DSL

dslRootPrivate.key : private key of DSL

shop.crt : client certificate of eShop signed by DSL

* eShop, at its end, adds the root certificate to the key store of eShop.

**keytool -import -alias rootserver -file dslRootCA.crt -keystore client\_keystore.jks -storepass Vodafone**

where

rootserver : alias name for the root certificate

dslRootCA.crt : root certificate of DSL

client\_keystore.jks : keystore file of Eshop

storepass : password of the keystore

* eShop also adds the client certificate to the keystore.

**keytool -import -alias mykey -file shop.crt -keystore client\_keystore.jks -storepass Vodafone**

where

mykey : name of the private key of eShop

shop.crt : client certificate of eShop signed by DSL

client\_keystore.jks : keystore file of Eshop

storepass : password of the keystore

* Similarly, DSL at it’s end adds the public key of eShop and client certificate signed by eShop to the keystore of DSL and root certificate of eShop to the trust store.

## Monitoring Tools

* **BAC (Business Availability Centre)**
* This is a third party tool which has in built scripts to monitor the site availability and response time of the site.
* We had different transactions that were configured such as
* Checkout
* Shop Basket
* eShop Home
* Upgrade etc.,
* It displays the transactions in two different formats
* Graph format
* Tabular format
* **HPOV (HP Open View)**
* This is also a third party tool which has in built scripts to monitor the
* Disk Space Utilization
* Network Connectivity and
* Security Issues
* Generates tickets corresponding to each of the above in BMC Remedy Tool
* **Cacti Tool**
* A third party tool used for monitoring the Network traffic for different servers over time
* Application Servers
* Web Servers
* DB Servers
* **Automated Scripts**
* We have some automated scripts for monitoring the below:
* Stuck threads (stdout logs)
* Unknown Exceptions(stdout logs)
* Deadlock exceptions(stdout logs)
* DSL Timeouts (serviceresponse.log) for
* Preauth\_declined
* Submit\_Order
* Middleware Timeouts/TIBCO Timeouts(JTC Logs – Correlation ID’s) for
* CheckBankAccount
* MatchAddress
* SendEmail

## Swap Space

Swap space in Linux is used when the amount of physical memory (RAM) is full. If the system needs more memory resources and the RAM is full, inactive pages in memory are moved to the swap space. While swap space can help machines with a small amount of RAM, it should not be considered a replacement for more RAM. Swap space is located on hard drives, which have a slower access time than physical memory.

Swap space can be a dedicated swap partition (recommended), a swap file, or a combination of swap partitions and swap files.

Swap should equal 2x physical RAM for up to 2 GB of physical RAM, and then an additional 1x physical RAM for any amount above 2 GB, but never less than 32 MB.

So, if:

**M = Amount of RAM in GB, and S = Amount of swap in GB, then**

**If M < 2**

**S = M \*2**

**Else**

**S = M + 2**

Using this formula, a system with 2 GB of physical RAM would have 4 GB of swap, while one with 3 GB of physical RAM would have 5 GB of swap. Creating a large swap space partition can be especially helpful if you plan to upgrade your RAM at a later time.

For systems with really large amounts of RAM (more than 32 GB) you can likely get away with a smaller swap partition (around 1x, or less, of physical RAM).

Linux is a demand-paged virtual memory system: all memory is broken up into pages—small equal-size chunks of a few kilobytes—and most of these chunks can be swapped (or “paged”) in or out of RAM as demand dictates (some pages are locked and can't be swapped). When a running process requires more RAM than is available, one or more pages of RAM that have not been used recently are “swapped out” to make RAM available. Similarly, if a running process requires access to RAM that previously has been “swapped out”, one or more pages of RAM are swapped out and the previously swapped-out RAM is swapped in. All of this happens behind the scenes without the programmer having to worry about it.

## How will you copy a file from one location to another?

* Using SCP command

**Syntax**:

In SSH window

Step1. scp <path of the source file> . Copies the file onto the SSH terminal

Step2. scp filename <destination path>

**Eg:**

scp dfadm@195.232.195.164:/opt/backup/Release11.09.01/Release\_eShop\_<<>>.tar .

scp Release\_eShop\_<<>>.tar [weloadm@195.232.195.196:/var/weblogic/eShop/eShopTarDist](mailto:weloadm@195.232.195.196:/var/weblogic/eShop/eShopTarDist)

## Thread Dump Location:

ps –ef | grep java

qProcessbox **20650** 1   0   Mar 26 ?         104:51 /opt/app/qProcessbox/java/jdk1.5.0\_22/bin/java

kill -3 20650

The Output of the Thread Dump will be generated in the Server STDOUT.log

Thread Dumps are one of the very important JVM reports which we can use to analyze Server/JVM Hang Situations and the Root Cause of it… Here are few very common techniques which we can use to take Thread Dumps…

Taking only one or two Thread Dumps is not very useful. Always make sure that you collect atleast 5-6 Thread Dumps taken in the interval of 10-12 Seconds….Which will be really helpful to investigate the Hang Situation.

## AWstats:

A powerful and featureful web server log analyzer Advanced Web Statistics (AWStats) is a powerful web server logfile analyzer written in perl that shows you all your web statistics including visits, unique visitors, pages, hits, rush hours, search engines, keywords used to find your site, robots, broken links and more. Gives more detailed information and better graphical charts than webalizer, and is easier to use. Works with several web server log format as a CGI and/or from command line. Supports multiple languages (English, French, Dutch, Spanish, Italian, German, Polish, Greek...).

## DB related Issues and their Resolution

**ISSUE 1.**

|  |  |  |
| --- | --- | --- |
| <BEA-149205> <Failed to initialize the application ‘TestDataSource’ due to error weblogic.application.ModuleException: . | | |
| The Above Exception: java.io.IOException: The Network Adapter could not establish the connection  It simply suggest that the Database URL may not be correct in the DataSource XML file locatied inside “<DOMAIN\_HOME>\config\jdbc” directoryor may be the TNS name OR some n/w issue between WLS BOX and the DB Box. Please try the following to Double Check it.  Step1). Add JDBC Driver also in the Classpath or Better run “. ./setWLSEnv.sh”  (NOTE: While running the above script please use two DOTs like mentioned above. The first DOT represents that set the Environment in the Current Shell and the second DOT (./) Slash represents that pick up the Script from the current Location. Both DOTs are separated by a single space.  Once u run “. ./setWLSEnv.sh” after that try to echo the values of $CLASSPATH and $PATH to make sure that the env is set properly)  Step2). Use WLS   dbping utility to test the Database Network Connectivity from the WebLogic Server Box: Syntax: java -classpath /bea103/wl\_server103/server/lib/weblogic.jar utils.dbping ORACLE\_THIN <dbUserName> <dbPasswoes> <dbURL>  Example: java -classpath /bea103/wl\_server103/server/lib/weblogic.jar utils.dbping ORACLE\_THIN scott tiger databaseHostName:1521:P15215h  Step 3). Try doing a telnet to connect to the Database Box on the Database Listen Port like following Just to make sure that the Database has started listening on the Mentioned Listen Port or Not …..:  telnet   databaseHostName   1521  **ISSUE 2.**  The error “Caused by: java.sql.SQLException: Io exception: Got minus one from a read call” clearly indicates the root cause of this issue. “We get the following error “Got minus one from a read call” if the Database goes under Maintenance…..or Database is in inconsistent State (Due to any reason like Database Maintenance or Database unmounting…etc ).  WLS is asking a JDBC driver for a connection and getting that exception, which means the DBMS or network dropped the socket during the driver-DBMS handshake.  The above error ensures that there is Nothing Wrong from WebLogic Side …we need to contact the Database Administrator only.  Step1). Open the Server Log and check the timestamp of the first occurance of that Error in the Server log.  Step2). Confirm with the Database Administrator…. What was the activity happening on the Database side at that time (90% cases u will find that the Database might be under Maintenance that time)  **ISSUE3.**  If you see that the Database is going into DataSource is moving to SUSPENDED/Disabled State….The first of all check the database connectivity. If the dfatabase connectivity is OK and WLS is able to establish the Connectivity with the database Successfully then Please try to use the “weblogic.Admin” utility to Restore (Resume) the DataSource.  You can use “weblogic.Admin” utility to Enable and Disable the Pool (DataSource) Just to Confirm whether the dataSource is actually active or not:  Step1). run “. ./setWLSEnv.sh” first in the same Shell prompt….then do the following:  Step2). To suspend: java weblogic.Admin -url t3://localhost:7001 -username weblogic -password weblogic SUSPEND\_POOL YourDataSourceName  Step3). To re-enable: java weblogic.Admin -url t3://localhost:7001 -username weblogic -password weblogic RESUME\_POOL YourDataSourceName  To test whether you are getting any Exception or Error while doing this….If yes u are getting any Error or exception then It means there May be some Database connectivity issue… If a datasource gets destroyed then even if u are having DataSource Configuration Entries in the “config.xml” and the “<DOMAIN\_HOME>/config/jdbc/\*.xml” files …still u wont be able to see that datasource in the Monitoring Tab in the AdminConsole. Because as soon as the datasource gets destroyed (Due to N/W disconnect or someother reason) the MBean Object gets destroyed.   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | |  | | |  |  | | | | | |  |  | | --- | --- | |  | | |  | [Tuning the Weblogic Server Performance](http://middlewaremagic.com/weblogic/?p=6384) Once applications are deployed, the problems start: Multiple users connect to the application and complain about the speed. Where do you start to locate the problem? Is it the operating system? What about garbage collection? Is the architecture of the application a problem? In this post we give a starting point in how to analyze a problem and act accordingly.  We outline best practices for solving production problems in three parts:   * System performance * Performance in general * Solve performance problems   Before we start with tuning the performance, it is important to have an overview of the internal architecture of the WebLogic Server. The processing of requests is based on the following components: listen threads, the socket muxer and the execute queue. When the server boots, the server mounts a listen thread to each configured port. The socket muxer detects an incoming request and places it in the execute queue. A free execute thread picks up the request from the queue and executes it. The execute thread processes the request in whole, in other words, the call to a servlet, the servlet call to an EJB and the EJB call to JDBC to query a database, are handled by one execute thread.  An important observation is that if application code blocks an execute thread for a long time, the server can not use this execute thread to process other requests. If the application is in a state where each thread is blocked for an indefinite period, the server stops responding or additional threads are created. Creation of additional threads that eventually will block does not help the situation. Threads that are not processed in the foreseeable future, does not benefit the performance. System performance A good understanding of the operating system, network, JVM and server resources (such as connection pools, JMS servers) and the associated tuning options help enormously in applying best practices. In general it is not enough to know what to do, we should also know why it works.  When we design an application, we must first understand the application itself and how users interact with it. We should investigate all system components to identify and understand the interaction between these components. By determining the workload over all layers, we get an understanding which components are most affected by the activity of users. A good understanding of the system allows us to choose the right system architecture. Once the system architecture is set, we can begin to focus on the architecture of the application. Some very wise man once said: “Architecture matters, and in systems of scale and systems that require availability, architecture matters absolutely! Failure to achieve a solid architecture will doom in advance any hope of significant scalability, and will leave the effects of failure within the system to pure chance.”  If after well-considered decisions, there are still performance problems in production and we want the maximum from the server, we need to tune. The tuning is done at different layers within the production environment. In the following discussion we start at the bottom and then work our way steadily upwards. Tuning the operating system In general, Java EE applications have some kind of Web interface. Usually, this type of applications have a few thousand concurrent users. As a result that a high number of connections between the browser and the server are opened and closed. These connections are nothing more than TCP sockets on operating system level. Most operating systems handle sockets as a form of file access and use file descriptors to keep track of which sockets are open. To contain the resources per process, the operating system restricts the number of file descriptors per process.  A TCP connection that is properly closed by an application is in the TIME\_WAIT state before it is returned to the operating system. While the connection is in TIME\_WAIT, all the used resources (including the file descriptor) stay allocated to the process. The result is that the file descriptor table can fill up. This means that we have to tune the operating system so that a scalable application does not run against an operating system restriction. Tuning basically means following recommendations from the hardware vendor.  A number of useful tools which can be used are netstat (to determine the number of sockets in TIME\_WAIT state, for example by using, netstat –a | grep TIME\_WAIT | wc –l) or iostat (to determine disk I/O on the operating system, for example, iostat 5 5). Note that iostat is part of the sysstat package, which can be installed by using: yum -y install sysstat. Tuning the Java Virtual Machine (JVM) The JVM used to run the WebLogic Server and the deployed applications is of great importance in the final performance of the WebLogic Server. Keep in mind that performance is nothing if we do not have stability – fast applications do not do the users any good if they are not running. So when choosing a JVM look for stability first, then the performance.  The garbage collection is the most important factor in tuning a JVM. Poorly tuned garbage collectors or applications that create an unnecessary numbers of objects, can have dramatic effects on the performance of the application. A proper tuning of the garbage collector, greatly reduces the processing time resulting in a significant improvement in application performance. More information on JVM tuning is presented in this [post](http://middlewaremagic.com/weblogic/?p=6388). Tuning the server During performance tests it is important to enlarge the listen queue. WebLogic Server uses the Accept Backlog parameter to specify the size of the queue. If client requests are rejected it may be that the size of the queue is too small. The login timeout parameter is used to specify the maximum time to set up a connection. By default, this value is 5 seconds (25 seconds for SSL), which may be too small for systems with a heavy load.   * **Thread management optimization** is handled by the WebLogic Server itself. By collecting performance information, the number of execute threads are adapted to the workload of the application. By default every deployed application is assigned to the default WorkManager. This gives each application the same priority and prevents applications to use more than their fair share of server resources. In general, the default WorkManager suffices. A number point are of interrest in this respect: * **Database Connection Pool** – If the application depends on database connections, it is important to note that no more concurrent requests can be processed than there are connections in the pool. In this case we can override the default WorkManager by imposing a maximum thread constraint. This constraint is set to be equal to the number of connections in the connection pool. * **Server Deadlock** – Some applications call resources in a different server instance, which in turn calls resources in the calling server. In this case, we must take extra care so that a server deadlock does not occur. A deadlock occurs when all threads are waiting for requests from the remote application and no thread is available to handle the callback process. In this case we can configure a minimum thread constraint so that a number of execute threads are available at all times to the process the callback. * **Server Overload** – If we tune the server to achieve a certain response time, it is important to note that at some point the server must no longer accept new requests. If a request is in the queue, it takes some time before it can be processed (the request queued is waiting for an execute thread). To ensure that the queue is not cluttered with long waiting requests, we setup a capacity constraint. If the capacity is exceeded, a 503 response is send back to the user. * WebLogic Server uses **resource pooling** to optimize performance. For example, stateless session beans and message-driven beans are pooled. If all resources are used, the server will increase the pool to suit the requests of the application’s requirements. Increasing the pool adds overhead to process a request. This overhead depends on the type of resource. When the pool reaches its maximum, the server can not increase the pool. If this situation occurs, requests must wait until a resource is released before the request can be processed. An obvious rule of thumb is to ensure that the pool is large enough to process the number of simultaneous requests.   Once optimum settings are found, there usually can be achieved some improvements in garbage collection.Before an application is put into production, it is important to conduct long-term load tests. These tests generally reveal problems, that only surface in production after a long time, such as memory leaks. Performance in general An appropriate architecture (the right patterns) can improve performance. Understanding where the application runs, such as in a Web and/or EJB container, and what these containers have to offer for is very important. Design Good application performace starts with a good design. Too complex an application will always perform badly, and tuning will not help. Proper use of design patterns could lead to significant advantages, such as standardizing default design problems. A number of design patterns also offer a performance enhancement, examples of these are the session facade and the command pattern:   * Session facades improve performance by offering high level business operations. Especially, when calling EJB by using remote interfaces (also when calling EJBs locally, because the EJB container provides services such as security, lifecycle management and transaction management). * The command pattern uses one command object to process requests. By using this pattern with the session facade the number of remote calls can be reduced, and as such we are also reducing remote network calls.  Web container Many applications complement the HTTP protocol by storing objects in the HTTP session. In the same browser session, the session data is available to subsequent requests. In a high-available application, the session data must be available even when the original server fails. This means that the session data must be replicated to other servers within a cluster. Keep in mind that there is a cost attached to replicating session data.  If a web application changes the HTTP session object, the server must store these changes at the end of each request. The amount of data that the server is to store depends on the data structure of the HTTP session object. If the web application changes a small amount of data in a large object structure, the server replicates this large object structure not just the changes, so it is important to divide large structures into small pieces. An HTTP session object also uses resources, so it is important to clean it up when you are done. EJB container EJB components can have a dramatic effect on performance, because they generally do more work than a web component. We can optimize calls to EJBs. If the caller and the EJB are in the same JVM and loaded by the same classloader, we can ensure that a pass-by-reference optimization occurs. This optimization must be turned on by using the deployment override **weblogic-ejb-jar.xml**, i.e., set the **enable-call-by-reference** element of a particular EJB component to true. Database access Efficient database access is vital to obtain scalability and a high throughput. If database access is slow all other tuning, such as JVM, EJB container etcetare, will be futile. First, we need an effective database design. A classic in this area is “An Introduction to Database Systems” written by CJ Tate. Many database designers have a preference for normalizing their design. With the result that multi-way joins are needed to retrieve data for a business object. A design that looks good on paper, is generally a disaster in production. If performance is the issue at hand, it is time to sit down with the database designer and denormalize critical tables.  The next step is that the physical database must meet the performance requirements. A DBA must use all the optimization options in order to achieve the best possible performance. Here it is important that the DBA has a complete list of all queries within the application so that the appropriate indexes can be made.  WebLogic Server supports both local and global transactions. Global operations include multiple resources. Global transactions have additional logging and extra network I/O is needed, which makes it slower than local transactions. If possible, use local transactions. Solve performance problems Now that the application and the environment are tuned to perfection, users are happy and the system shamelessly handles hundreds of requests per second, or not? As mentioned above a requirement for successful troubleshooting we need a good understanding of the system and its components. Each system is different and every performance problem is probably different. There are, however, a number of best practices that can help pinpoint problems. Preparation Solving problems is difficult and time-consuming. If users are unhappy, we need the right infrastructure, processes and people to solve the problem.  First, the application must be thoroughly tested. It is important to find out how the application behaved in the test, in order to know if the problem we have to find a solution for is normal during peak loads. Test results also gives us insight in the resource usage. Testing is half the work for solving production problems.  Additionally, all performance monitoring mechanisms must be in place in order to provide information on system performance and activity. Unfortunately, performance problems are not there on request, so we must have some form of logging in order to reconstruct the resource usage and activity during a given period.  Last be not least; it is not a bad idea to have a multi-disciplinary team available before the problems arise. Identifying and correcting bottlenecks A bottleneck is a resource within a system, which reduces the throughput of a system or affects the response time significantly. To fix bottlenecks in a distributed system is not one of the easiest tasks in IT. In general, there are multi-disciplinary teams required. The advent of performance monitoring tools makes fixing problems somewhat easier. Bottlenecks can occur in the web server, application code, application server, database, network, hardware or operating systems. Differences between Weblogic 9X and 10X  * Fast Swap feature is available in 10X but not in 9X * Sun JDK 1.6 is used in 10X, JDK 1.5 in 9X * Summary of servers page has the below in 10X * Configuration and * Control tabs * Supports EJB3.0 in 10X, EJB2.0 in 9X * JDBC 4.0 Support, JDBC 3.0 in 9X * Automatic Migration of Messaging/JMS Related Services allowed in 10X * Asynchronous HTTP Session Replication introduced in 10X * New Look and Feel (Colors, borders and etc.,) * Data sources can be restarted from WLS 10x, in 9X we have to remove the data source and reconfigure it in order to refresh  What is a Distributed Destination? A *distributed* [destination](http://docs.oracle.com/cd/E13222_01/wls/docs90/jms/fund.html#destination) is a set of destinations (queues or topics) that are accessible as a single, logical destination to a client. A distributed destination has the following characteristics:   * It is referenced by its own JNDI name. * Members of the set are usually distributed across multiple servers within a cluster, with each destination member belonging to a separate JMS server.  Why Use a Distributed Destination Applications that use distributed destinations are more highly available than applications that use simple destinations because WebLogic JMS provides load balancing and failover for member destinations of a distributed destination within a cluster. Once properly configured, your producers and consumers are able to send and receive messages through the distributed destination. WebLogic JMS then balances the messaging load across all available members of the distributed destination. When one member becomes unavailable due a server failure, traffic is then redirected toward other available destination members in the set.  **Common Name (CN) in SSL Certificate:**  The Common Name is the Fully Qualified Domain Name (FQDN) for which you are requesting the ssl certificate.  Before you can enroll for a SSL Server Certificate, you must generate a CSR from your webserver software. During the creation of the CSR, the following fields must be entered: Organization (O), Organizational Unit (OU), Country (C), State (S), Locality (L), and Common Name (CN). The Common Name field is often misunderstood and is filled out incorrectly.  The **Common Name** is typically composed of **Host + Domain Name** and will look like **"www.yoursite.com"** or **"yoursite.com".** SSL Server Certificates are specific to the Common Name that they have been issued to at the Host level. The Common Name must be the same as the Web address you will be accessing when connecting to a secure site. For example, a SSL Server Certificate for the domain "domain.com" will receive a warning if accessing a site named **"www.domain.com"** or **"secure.domain.com"**, as **"www.domain.com"** and **"secure.domain.com"** are different from **"domain.com"**. You would need to create a CSR for the correct Common Name. When the Certificate will be used on an Intranet (or internal network), the Common Name may be one word, and it can also be the name of the server. What is SSL and what are Certificates? The Secure Socket Layer protocol was created by Netscape to ensure secure transactions between web servers and browsers. The protocol uses a third party, a Certificate Authority (CA), to identify one end or both end of the transactions. This is in short how it works.   1. A browser requests a secure page (usually https://). 2. The web server sends its public key with its certificate. 3. The browser checks that the certificate was issued by a trusted party (usually a trusted root CA), that the certificate is still valid and that the certificate is related to the site contacted. 4. The browser then uses the public key, to encrypt a random symmetric encryption key and sends it to the server with the encrypted URL required as well as other encrypted http data. 5. The web server decrypts the symmetric encryption key using its private key and uses the symmetric key to decrypt the URL and http data. 6. The web server sends back the requested html document and http data encrypted with the symmetric key. 7. The browser decrypts the http data and html document using the symmetric key and displays the information.   Several concepts have to be understood here. 1.2.1. Private Key/Public Key: The encryption using a private key/public key pair ensures that the data can be encrypted by one key but can only be decrypted by the other key pair. This is sometime hard to understand, but believe me it works. The keys are similar in nature and can be used alternatively: what one key encrypts, the other key pair can decrypt. The key pair is based on prime numbers and their length in terms of bits ensures the difficulty of being able to decrypt the message without the key pairs. The trick in a key pair is to keep one key secret (the private key) and to distribute the other key (the public key) to everybody. Anybody can send you an encrypted message, that only you will be able to decrypt. You are the only one to have the other key pair, right? In the opposite, you can certify that a message is only coming from you, because you have encrypted it with you private key, and only the associated public key will decrypt it correctly. Beware, in this case the message is not secured you have only signed it. Everybody has the public key, remember!  One of the problem left is to know the public key of your correspondent. Usually you will ask him to send you a non confidential signed message that will contains his public key as well as a certificate.   |  | | --- | | Message-->[Public Key]-->Encrypted Message-->[Private Key]-->Message |  The Certificate: How do you know that you are dealing with the right person or rather the right web site. Well, someone has taken great length (if they are serious) to ensure that the web site owners are who they claim to be. This someone, you have to implicitly trust: you have his/her certificate loaded in your browser (a root Certificate). A certificate, contains information about the owner of the certificate, like e-mail address, owner's name, certificate usage, duration of validity, resource location or Distinguished Name (DN) which includes the Common Name (CN) (web site address or e-mail address depending of the usage) and the certificate ID of the person who certifies (signs) this information. It contains also the public key and finally a hash to ensure that the certificate has not been tampered with. As you made the choice to trust the person who signs this certificate, therefore you also trust this certificate. This is a certificate trust tree or certificate path. Usually your browser or application has already loaded the root certificate of well known Certification Authorities (CA) or root CA Certificates. The CA maintains a list of all signed certificates as well as a list of revoked certificates. A certificate is insecure until it is signed, as only a signed certificate cannot be modified. You can sign a certificate using itself, it is called a self signed certificate. All root CA certificates are self signed.  The certificate contains the reference to the issuer, the public key of the owner of this certificate, the dates of validity of this certificate and the signature of the certificate to ensure this certificate hasen't been tampered with. The certificate does not contain the private key as it should never be transmitted in any form whatsoever. This certificate has all the elements to send an encrypted message to the owner (using the public key) or to verify a message signed by the author of this certificate. The Symmetric key: Well, Private Key/Public Key encryption algorithms are great, but they are not usually practical. It is asymmetric because you need the other key pair to decrypt. You can't use the same key to encrypt and decrypt. An algorithm using the same key to decrypt and encrypt is deemed to have a symmetric key. A symmetric algorithm is much faster in doing its job than an asymmetric algorithm. But a symmetric key is potentially highly insecure. If the enemy gets hold of the key then you have no more secret information. You must therefore transmit the key to the other party without the enemy getting its hands on it. As you know, nothing is secure on the Internet. The solution is to encapsulate the symmetric key inside a message encrypted with an asymmetric algorithm. You have never transmitted your private key to anybody, then the message encrypted with the public key is secure (relatively secure, nothing is certain except death and taxes). The symmetric key is also chosen randomly, so that if the symmetric secret key is discovered then the next transaction will be totally different. The Hash: A hash is a number given by a hash function from a message. This is a one way function, it means that it is impossible to get the original message knowing the hash. However the hash will drastically change even for the slightest modification in the message. It is therefore extremely difficult to modify a message while keeping its original hash. It is also called a message digest. Hash functions are used in password mechanisms, in certifying that applications are original (MD5 sum), and in general in ensuring that any message has not been tampered with. It seems that the Internet Enginering Task Force (IETF) prefers SHA1 over MD5 for a number of technical reasons (Cf RFC2459 7.1.2 and 7.1.3). Signing: Signing a message, means authentifying that you have yourself assured the authenticity of the message (most of the time it means you are the author, but not neccesarily). The message can be a text message, or someone else's certificate. To sign a message, you create its hash, and then encrypt the hash with your private key, you then add the encrypted hash and your signed certificate with the message. The recipient will recreate the message hash, decrypts the encrypted hash using your well known public key stored in your signed certificate, check that both hashes are equal and finally check the certificate.  The other advantage of signing your messages is that you transmit your public key and certificate automatically to all your recipients.  There are usually 2 ways to sign**, encapsulating the text message inside the signature (with delimiters), or encoding the message altogether with the signature**. This later form is a very simple encryption form as any software can decrypt it if it can read the embedded public key. The advantage of the first form is that the message is human readable allowing any non complaint client to pass the message as is for the user to read, while the second form does not even allow to read part of the message if it has been tampered with. How to start a Web Server  * There's probably a shell script in **/etc/init.d** called apache. Simply type: /etc/init.d/apache start * If there is no startup script in the **/etc/init.d** * We can look into the installation-bin-dir of apache * ( e.g. **/opt/apache/bin or /usr/local/apache/bin**) and there should be a "apachectl"-program.  ./apachectl start ./apachectl stop ./apachectl restart * In our eShop   cd /opt/apache-2.2/scripts  /usr/local/bin/sudo ./stop\_apache.sh eShopprod  /usr/local/bin/sudo ./start\_apache.sh eShopprod Comparison Between Multicast Messaging and Unicast Messaging Mode When servers are in a cluster, these member servers communicate with each other by sending heartbeats and indicating that they are alive. For this communication between the servers, either unicast or multicast messaging is used. This is chosen from the admin console in **Cluster -> Configuration -> Messaging -> Messaging Mode.**   * To use multicast messaging, hardware configuration and support for multicast packets is required. * Unicast does not have this requirement, which is why using unicast in latest versions is recommended.   When multicast messaging is used, it is a one-to-many communication, every server sends the notification/heartbeat/multicast packet to each other. This causes a heavy load on the application’s multicast buffer, so if the buffer is full, new multicast messages cannot be written to the buffer and the application is not notified when messages are dropped. So there is a possibility that the server instances miss the messages. This might lead in the cluster throwing out the server instances out of the cluster.  Unicast configuration is much easier because it does not require cross network configuration that multicast requires. Additionally, it reduces potential network errors that can occur from multicast address conflicts.  **Mode of Communication:** **Multicast:** A multicast address and multicast port is used for listening to the messages.  **Unicast**: A network channel is used for communication between the servers. If no channel is specified, default network channel is used.  **Method of communication between servers: Multicast:** Each server communicates with every member server in the cluster. Which means heartbeats are sent to every server.  **Unicast:** For the member servers in the cluster, group leaders are chosen and only those group leaders communicate with the servers among the group and these leaders notify each other about the availability of all the other servers. For example : Suppose there are 6 servers in the cluster. 2 groups are made and there are 2 group leaders. Other 2 servers of the group will notify their leader that they are alive and the group leader will send this information to the leader of other group.  The frequency of communication in unicast mode is similar to the frequency of sending messages on multicast port.  For new server versions, using unicast is recommended because it is a simplified communication mode. But for backward compatibility with the previous versions, you will need to use multicast if there is a communication requirement between clusters of versions prior to WLS 10.0.  Whenever the messaging mode of the cluster is changed, all the servers in the cluster need a restart because the changes are not dynamic.  **What is OutOfMemory?**  An **OutOfMemory** is a condition in which there is not enough space left for allocating required space for the new objects or libraries or native codes. OutOfMemory can be divided in tow main categories:  **1). OutOfMemory in Java Heap:**  This happens when the JVM is not able to allocate the required memory space for a Java Object. There may be many reasons behind this…like  **1)** Very Less Heap Size allocation. Means setting the MaxHeapSize (-Xmx) parameter to a very less value.  **2)** The Leaking of Objects. Either the Application is not unreferencing the unused Objects or the Third part frameworks (Hibernate/Spring/Seam…etc) might not be releasing the references of the objects due to some inaccurate configurations.  **3)** In Many cases it may be the reason that Application codes are getting the JDBC connections objects from the DataSource are not being released back to the Connection Pool.  **4)** Garbage Collection strategy may be in correct according to the environmental/application requirements.  **5)** Inaccurate setting of Application/Frameworks Cache.   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Example:**Exception in thread "Thread-10" java.lang.OutOfMemoryError: Java heap space  **2). Native OutOfMemory:**  Native OutOfMemory is a scenario when the JVM is not able to allocate the required Native Libraries and JNI Codes in the memory.  Native Memory is an area which is usually used by the JVM for it’s internal operations and to execute the JNI codes. The JVM Uses Native Memory for Code Optimization and for loading the classes and libraries along with the intermediate code generation.  The Size of the Native Memory depends on the Architecture of the Operating System and the amount of memory which is already commited to the Java Heap. Native memory is an Process Area where the JNI codes gets loaded or JVM Libraries gets loaded or the native Performance packs and the Proxy Modules gets loaded…  Native OutOfMemory can happen due to the following main reasons:  **1)** Setting very small StackSize (-Xss). StackSize is a memory area which is allocated to individual threads where they can place their thread local objects/variables..  **2)** Usually it may be seen because of Tuxedos incorrect setting. WebLogic Tuxedo Connectors allows the interoperability between the Java Applications deployed on WebLogic Server and the Native Services deployed on Tuxedo Servers. Because Tuxedos uses JNI code intensively.  **3)** Less RAM or Swap Space.  **4)** Usually it may occur is our Application is using a very large number of JSPs in our application. The JSPs need to be converted into the Java Code and then need to be compiled. Which reqires DTD and Custom Tag Library resolution as well. Which usually consumes more native memory.  **Example:**   |  |  |  | | --- | --- | --- | | Exception in thread "main" java.lang.OutOfMemoryError: unable to create new native thread |  | | | **3).  OutOfMemory in PermGen Space:**  Permanent Generation is a Non-Heap Memory Area inside the JVM Space. Manytimes we see OutOfMemory in this Area. PermGen Area is NOT present in JRockit JVMs.  The PermGen Area is measured independently from the other generations because this is the place where the JVM allocates Classes, Class Structures, Methods and Reflection Objects. PermGen is a Non-Heap Area.It means we DO NOT count the PermGen Area as part of Java Heap.  The OutOfMemory in PermGen Area can be seen because of the following main reasons:  **1**) Deploying and Redeploying a very Large Application which has many Classes inside it.  **2)** If an Application is getting deployed/Updated/redeployed repeatedly using the Auto Deployment feature of the Containers. In that case the Classes belonging to the application stays un cleaned and remains in the PermGen Area without Class Garbage Collection.  **3)** If  ”**-noclassgc**” Java Option is added while starting the Server. In that case the Classes instances which are not required will not be Garbage collected.  **4)** Very Less Space for allocated the “=XX:MaxPermGen”  Example: you can see following kind of Trace in the Server/Stdout Logs:   |  |  | | --- | --- | |  |  | |  | | | | |  |  | | | |  |  | | | |  |  | | --- | --- | |  |  | | |  |  |  | | --- | --- | |  |  |   **List of port numbers used for different protocols**  **Port Number** **Protocol**  20 [FTP](http://en.wikipedia.org/wiki/FTP)—data transfer  22 [Secure Shell](http://en.wikipedia.org/wiki/Secure_Shell) (SSH)-used for secure logins, [file transfers](http://en.wikipedia.org/wiki/File_transfer)  ([scp](http://en.wikipedia.org/wiki/Secure_copy), [sftp](http://en.wikipedia.org/wiki/SSH_file_transfer_protocol)) and port forwarding  23 [Telnet](http://en.wikipedia.org/wiki/Telnet) protocol—unencrypted text communications  25 [Simple Mail Transfer Protocol](http://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol) (SMTP)—used for e-mail  routing between mail servers  53 [Domain Name System](http://en.wikipedia.org/wiki/Domain_Name_System) (DNS)  57 [Mail Transfer Protocol](http://en.wikipedia.org/wiki/Mail_Transfer_Protocol)  79 [Finger protocol](http://en.wikipedia.org/wiki/Finger_protocol)  80 [Hypertext Transfer Protocol](http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (HTTP)  115 [Simple File Transfer Protocol](http://en.wikipedia.org/wiki/Simple_File_Transfer_Protocol) (SFTP)  161 [Simple Network Management Protocol](http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol) (SNMP)  389 [Lightweight Directory Access Protocol](http://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol) (LDAP)  443 [Hypertext Transfer Protocol](http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (HTTPS)  514 [Shell](http://en.wikipedia.org/wiki/Remote_Shell)-used to execute non-interactive commands on a  remote system (Remote Shell, rsh, remsh)  530 Routing Information Protocol  989 [FTPS](http://en.wikipedia.org/wiki/FTPS) Protocol (data): [FTP](http://en.wikipedia.org/wiki/FTP) over [TLS](http://en.wikipedia.org/wiki/Transport_Layer_Security)/[SSL](http://en.wikipedia.org/wiki/Secure_Sockets_Layer)  990 [FTPS](http://en.wikipedia.org/wiki/FTPS) Protocol (control): [FTP](http://en.wikipedia.org/wiki/FTP) over [TLS](http://en.wikipedia.org/wiki/Transport_Layer_Security)/[SSL](http://en.wikipedia.org/wiki/Secure_Sockets_Layer)  **Why Virtualize?**  The most common use for virtualization is consolidation: combining multiple workloads on one physical computer. This allows people to run a lot of virtual machines on fewer physical computers.  There are many reasons for adopting server virtualization. A popular one is better resource utilization. It is not uncommon to see servers running at 10 percent or less of their capacity, at different points in the day. By letting several virtual servers share a single set of hardware, a much higher average utilization rate is achieved, and hardware and support costs are lowered.    **Load Balancer:**  A load balancer is a device that acts as a [reverse proxy](http://www.f5.com/glossary/reverse-proxy/) and distributes network or application traffic across a number of servers. Load balancers are used to increase capacity (concurrent users) and reliability of applications. They improve the overall performance of applications by decreasing the burden on servers associated with managing and maintaining application and network sessions, as well as by performing application-specific tasks.  Load balancers are generally grouped into two categories: Layer 4 and Layer 7. Layer 4 load balancers act upon data found in network and transport layer protocols (IP, TCP, FTP, UDP). Layer 7 load balancers distribute requests based upon data found in application layer protocols such as HTTP.  Requests are received by both types of load balancers and they are distributed to a particular server based on a configured algorithm. Some industry standard algorithms are:   * Round robin * Weighted round robin * Least connections * Least response time   Layer 7 load balancers can further distribute requests based on application specific data such as HTTP headers, cookies, or data within the application message itself, such as the value of a specific parameter.  Load balancers ensure reliability and availability by monitoring the "health" of applications and only sending requests to servers and applications that can respond in a timely manner.  A pair of BIG-IP load balancers (redundant BIG-IP controllers) supports automatic failover and helps insure the reliability of the request routing. The load balancing device pair is configured in the Active/Passive mode. Each F5 BIG-IP controller has two interfaces, exp0 (with public addresses) and exp1 (with private addresses).  **Note:** Exp1 is the administrative interface. IP addresses for the BIG-IP controller are placed on exp1.  **Load-balancing techniques**  Load balancing can be done through hardware- or software-based techniques. One technique, called **DNS load balancing**, involves maintaining identical copies of the site on physically separate servers. The DNS entry for the site is then set to return multiple IP addresses, each corresponding to the different copies of the site. The DNS server then returns a different IP address for each request it receives, cycling through the multiple IP addresses. This method gives you a very basic implementation of load balancing. However, since DNS entries are cached by clients and other DNS servers, a client continues to use the same copy during a session. This can be a serious drawback, as heavy website users may get the particular IP address that is cached on their client or DNS server, while less-frequent users get another. So, heavy users could experience a performance slowdown, even though the server’s resources may be available in abundance.  http://pcquest.com/images/server17_2k1.jpgAnother load-balancing technique involves **mapping the site name to a single IP address**, which belongs to a machine that is set up to intercept HTTP requests and distribute them among multiple copies of the Web server. This can be done using both hardware and software. hardware solutions, even though expensive, are preferred for their stability. This method is preferred over the DNS approach, as better load balancing can be achieved. Also, these load balancers can see if a particular machine is down, and accordingly divert the traffic to another address dynamically. This is in contrast to the DNS method, where a client is stuck with the address of the dead machine, until it can request a new one.  Another technique, **Reverse Proxying**, involves setting up a reverse proxy, that receives requests from the clients, proxies them to the Web server and caches the response onto itself on its way back to the client. This means that the proxy server can provide static content from its cache itself, when the request is repeated. This in turn ensures that the server itself can focus its energies on delivering dynamic content. Dynamic content cannot generally be cached, as it is generated real time. Reverse proxying can be used in conjunction with the simple load- balancing techniques discussed  earlier—static and dynamic contents can be split across different servers and reverse proxying used for the static content Web server only.  Load balancing allows Web servers to be scaleable, that is, a server can be scaled up or scaled down. For example if the load on your website is balanced across three identical servers and it experiences a sudden flood of users http://pcquest.com/images/server37_2k1.gifbeyond its current capacity, you can set up additional servers identical to the ones already running and modify the DNS entry of your website to include pointers to these additional hosts. This is called scaling up. Conversely, if you know that your website is particularly low on traffic during certain times, on Sundays, for example, you can remove one or more servers and put their resources to better use. This is scaling down. This involves prior knowledge of access patterns and a little bit of foresight.  These techniques apart, it is important to have the server’s basics right too—fast CPUs, plenty of RAM, faster (and fatter) disks, etc. Check to ensure that your operating system supports symmetric multiprocessing (SMP), else having multiple processors is a waste. Turning off reverse DNS (looking up the names of addresses accessing your site), if not needed, also provides a considerable increase in performance.  **Advantages:**   * To achieve optimal resource utilization * Maximize throughput * Minimize Response Time and * Avoid Overload * [**HTTP Compression**](http://en.wikipedia.org/wiki/HTTP_compression)**:** reduces amount of data to be transferred for HTTP objects by utilizing gzip compression available in all modern web browsers. The larger the response and the further away the client is the more this feature can improve response times. The tradeoff is that this feature puts additional CPU demand on the Load Balancer and it is a feature which could be done by web servers instead. * **HT**[**TP Caching**](http://en.wikipedia.org/wiki/HTTP_caching)**:** the load balancer can store static content so that some requests can be handled without contacting the web servers. * **HTTP Security:** some load balancers can hide HTTP error pages, remove server identification headers from HTTP responses, and encrypt cookies so end users can't manipulate them. * [**Priority Queuing**](http://en.wikipedia.org/wiki/Priority_queuing)**:** also known as [rate shaping](http://en.wikipedia.org/wiki/Rate_shaping), the ability to give different priority to different traffic. * **Client Authentication:** authenticate users against a variety of authentication sources before allowing them access to a website. * [**Firewall**](http://en.wikipedia.org/wiki/Firewall_(networking))**:** direct connections to backend servers are prevented, for network security reasons Firewall is a set of rules that decide whether the traffic may pass through an interface or not.   **Reverse proxy:**  In [computer networks](http://en.wikipedia.org/wiki/Computer_network), a reverse proxy is a type of [proxy server](http://en.wikipedia.org/wiki/Proxy_server) that retrieves resources on behalf of a [client](http://en.wikipedia.org/wiki/Client_(computing)) from one or more [servers](http://en.wikipedia.org/wiki/Server_(computing)). These resources are then returned to the client as though they originated from the reverse proxy itself.[[1]](http://en.wikipedia.org/wiki/Reverse_proxy#cite_note-apache-forward-reverse-0) While a [forward proxy](http://en.wikipedia.org/wiki/Proxy_server#Forward_proxies) acts as an intermediary for its (usually nearby) associated client(s) and returns to them resources accessible on the Internet, a reverse proxy acts as an intermediary for its (usually nearby) associated server(s) and only returns resources provided by those associated server(s).  **Uses of reverse proxies**   * Reverse proxies can hide the existence and characteristics of the origin server(s). * [Application firewall](http://en.wikipedia.org/wiki/Application_firewall) features can protect against common web-based attacks. Without a reverse proxy, removing malware or initiating takedowns, for example, can become difficult. * In the case of [secure websites](http://en.wikipedia.org/wiki/Secure_website), the [SSL](http://en.wikipedia.org/wiki/Secure_Sockets_Layer) [encryption](http://en.wikipedia.org/wiki/Encryption) is sometimes not performed by the web server itself, but is instead offloaded to a reverse proxy that may be equipped with [SSL acceleration](http://en.wikipedia.org/wiki/SSL_acceleration) hardware. * A reverse proxy can [distribute the load](http://en.wikipedia.org/wiki/Load_balancer) from incoming requests to several servers, with each server serving its own application area. In the case of reverse proxying in the neighborhood of [web servers](http://en.wikipedia.org/wiki/Web_server), the reverse proxy may have to rewrite the [URL](http://en.wikipedia.org/wiki/URL) in each incoming request in order to match the relevant internal location of the requested resource. * A reverse proxy can reduce load on its origin servers by [caching](http://en.wikipedia.org/wiki/Web_cache) [static content](http://en.wikipedia.org/wiki/Static_web_page), as well as [dynamic content](http://en.wikipedia.org/wiki/Dynamic_web_page). Proxy caches of this sort can often satisfy a considerable amount of website requests, greatly reducing the load on the origin server(s). Another term for this is [web accelerator](http://en.wikipedia.org/wiki/Web_accelerator). * A reverse proxy can optimize content by [compressing](http://en.wikipedia.org/wiki/Data_compression) it in order to speed up loading times. * In a technique known as "spoon feeding", a dynamically generated page can be produced all at once and served to the reverse-proxy, which can then return it to the client a little bit at a time. The program that generates the page is not forced to remain open and tying up server resources during the possibly extended time the client requires to complete the transfer. * Reverse proxies can be used whenever multiple web servers must be accessible via a single public IP address. The web servers listen on different ports in the same machine, with the same local IP address or, possibly, on different machines and different local IP addresses altogether. The reverse proxy analyzes each incoming call and delivers it to the right server within the local area network.   **Web Server:**  Web servers serve pages to clients as and when a request is made. Whenever a server receives a request, it creates a child process, which handles that particular request. As a result, most Web servers run in multi-threading and multi-processing environments. However, even such an environment puts a limit on the number of Web pages that can be served concurrently, largely because of two factors: the bandwidth available and the Web server itself. Assuming that you have sufficient bandwidth, the performance of your Web server becomes the critical factor.  Your Web server’s performance is determined to a large extent by the underlying hardware resources available to it. This limit is higher when the content delivered is static like images or text, but considerably lower when dealing with dynamic content. Load balancing involves spreading the ‘load’ among multiple machines, or sometimes even among multiple sites, thereby increasing the resources available. Load balancing in its crudest form would, for example, involve placing all HTML files on one host, all images on another and all CGI scripts on the third. Real-life load balancing, however, involves carefully examining access patterns of various files on the website and keeping identical copies of the same Web server and distributing the load amongst them.  **Connecting to Weblogic Server Embedded LDAP using LDAP Browser**  Weblogic Server Embedded LDAP Server runs on the same port as the Server. The Master LDAP Server runs on the same port on which the Admin Server runs. Any changes is first made on the Master LDAP Server and it gets propagated to the Embedded LDAP Server on the Mananged Servers. If we do any changes to the LDAP Server of the Managed Servers, it doesn’t get reflected on the Master LDAP Server so we should be careful with our application design when the application interacts with the Embedded LDAP Server.  **How to connect to the Embedded LDAP Server of Weblogic?**   * First we need to go to Domain > Security > Embedded LDAP and set the credential as Weblogic. * Then we need to restart the Weblogic Server. This step is very important!   **General Weblogic Server Interview Questions**  I often get mails from people asking what are the questions generally asked in Weblogic by interviewers. Generally for an Administrator’s positions, in addition to Weblogic questions, there are a lot of Unix questions and a few questions related to process and case handling.  These are the important questions that I cover while interviewing.  **1. How do you differentiate between a server hang and server crash issue?**  When a Server crashes, the JAVA process no longer exists. When the Server is hung, it stops responding. We can use the weblogic.ADMIN utility to ping the server. In case of a hang situation we can take multiple thread dumps and analyze the cause of hang.  **2. What can be the various reasons for a server crash?**   * Native IO * SSL Native Libraries * JVM * Supported Configuration * JDBC Driver Issue   **3. How do you troubleshoot a crash?**   * JVM crash generates a hs\_err\_pid file. We need to look into the stack trace of the hs\_err\_pid file * If the thread is from a Native IO, we need to disable Native IO. * If the stack trace is from the driver, we need to get in touch with the driver team. Quite possibly it’s a problem with driver. Changing the type of driver can be a workaround. * If the thread shows it coming from an optimized code, we can turn of optimization. * If the stack is from native calls of application, it’s a bug with the application and it has to be modified.   **4. How do you troubleshoot Server Hang?**   * We can use java weblogic.Admin PING to check if we get a normal response. * We need to take multiple thread dumps with kill -3 PID on UNIX and CTLR Break on Windows. Analyze the thread dump to find the root cause.   **5. What can be the reasons of Server hang?**   * Memory leak * Database query taking a long time to return * Deadlock.   **6. What is memory leak?**  Memory leak is when objects are not removed from the heap even when they are not required.  **7. What are the various causes for OUT OF MEMORY?**   * Insufficient heap size, not able to match the extra load. * Objects living too long, like HTTP Sessions. * Memory leak in application code. * Full GC not happening due to JVM Bug.   **8. How to troubleshoot and overcome such issues?**   * Gather memory data by enabling GC verbose. * If it’s due to Http Session, timing out http session after certain interval might help. * Look into the code for JDBC connection handling. * Optimizing the heap size according to the load.   **9. When does High CPU Usage occur?**  It occurs when one process or one thread utilizes high proportion of CPU unexpectedly.  **10. How to troubleshoot it?**  In Solaris environment, we need to take pstack and prstack and see what the threads are doing. In Windows we need to use pslist and process explorer.  **11. What is Clustering and what is achieved through it?**  Clustering is the grouping together of servers for the purpose of high availability and scalability. Load balancing and Failover is achieved.  **12. How does Cluster Communication Happen?**  Members of the Cluster communicate over the Cluster Multicast IP and Port by sending periodic heart beat messages.  **13. What is the difference between the Sun JVM and BEA JRockit JVM?**  The most well known JVM is the implementation from Sun. The Sun JVM is called HotSpot. The Sun JVM is shipped in the Java Developer’s Kit (JDK) and Java Runtime Environment (JRE) from Sun.  The BEA JRockit JVM from BEA systems is optimized for reliability and performance for server side applications. To achieve this, BEA JRockit JVM uses technologies such as code generation, hot spot detection, code optimization, advanced garbage collection algorithms and tight operating system integration.  **14. Tuning JVM Parameters.**   * If you have a single processor, single thread machine then you should use the serial collector (default for some configurations, can be enabled explicitly for with -XX:+UseSerialGC). * For multiprocessor machines where your workload is basically CPU bound, use the parallel collector. This is enabled by default if you use the -server flag, or you can enable it explicitly with -XX:+UseParallelGC. * If you’d rather keep the GC pauses shorter at the expense of using more total CPU time for GC, and you have more than one CPU, you can use the concurrent collector (-XX:+UseConcMarkSweepGC). * Note that the concurrent collector tends to require more RAM allocated to the JVM than the serial or parallel collectors for a given workload because some memory fragmentation can occur.   **15. How do you do performance tuning of WLS?**  It can be categorized into 4 parts.  **Application Tuning:**   * JSP pre-compilation and * EJB pool size cache   **OS Tuning:**   * Setting tcp ip parameter. * tcp\_time\_wait\_interval * tcp\_conn\_req\_max\_q   **Core Server Tuning:**   * Tuning work manager * Tuning chuck size and chunk pool size * Using performance packs and * Connection backlog buffering   **JVM Tuning:**   * Tuning the GC strategy * Monitoring Garbage Collection   **WHAT IS VIRTUAL MACHINE?**  **Virtual Machine** is a completely separate individual operating system installation on your usual operating system. It is implemented by software emulation and hardware virtualization.   Virtual machine is a software implementation of a physical machine - computer - that works and executes analogically to it. Virtual machines are divided in two categories based on their use and correspondence to real machine: system virtual machines and process virtual machines. First category provides a complete system platform that executes complete operating system, second one will run a single program.   Frequently multiple virtual machines with their own OS's are used in server consolidation, where different services are run in separate virtual environments, but on the same physical machine.  **Advantages:**   * Multiple OS environments can exist simultaneously on the same machine, isolated from each other. * Virtual machine can offer an instruction set architecture that differs from real computer's; * Easy maintenance, application provisioning, availability and convenient recovery. * Increase the hardware utilization. * Decrease the capital and operating cost by sharing in number of VM. * High availability and Secure. * You can run programs in other operating systems from what is running on the machine.   **Disadvantages:**   * Virtual machine is not that efficient as a real one when accessing the hardware. * When multiple virtual machines are simultaneously running on a host computer, each virtual machine may introduce an unstable performance, which depends on the workload on the system by other running virtual machines. * If server is shutdown, we cannot access the VM. * Increased memory usage and processor overhead.   **What is virtualization in Linux**  **Linux virtualization** refers to running one or more virtual machines on a physical computer that's operated by the [Linux](http://www.webopedia.com/TERM/L/Linux.htm) [open source](http://www.webopedia.com/TERM/O/open_source.html) [operating system](http://www.webopedia.com/TERM/O/operating_system.html).  Linux [virtualization](http://www.webopedia.com/TERM/V/virtualization.html) can be used for isolating specific apps, programming code or even an operating system itself, as well as for security and performance testing purposes.  Today's more powerful computers and hardware have made virtualization more practical and feasible for both desktop and server environments, helping to save power by consolidating several workspaces on one system as well as maximizing the workload that the compute can handle.  Popular Linux virtualization solutions include Xen, KVM, QEMU, VirtualBox and VMware. |  |

**Secure Socket Layer (SSL)**

**What SSL Provides:**

**SSL** secures communication by providing message encryption, integrity, and authentication. The SSL standard allows the involved components (such as browsers and HTTP servers) to negotiate which encryption, authentication, and integrity mechanisms to use.

* **Encryption** provides confidentiality by allowing only the intended recipient to read the message. SSL can use different encryption algorithms to encrypt messages. During the SSL handshake that occurs at the start of each SSL session, the client and the server negotiate which algorithm to use. Examples of encryption algorithms supported by SSL include AES, RC4, and 3DES.
* **Integrity** ensures that a message sent by a client is received intact by the server, untampered. To ensure message integrity, the client hashes the message into a digest using a hash function and sends this message digest to the server. The server also hashes the message into a digest and compares the digests. Because SSL uses hash functions that make it computationally infeasible to produce the same digest from two different messages, the server can tell that if the digests do not match, then someone had tampered with the message. An example of a hash function supported by SSL is SHA1.
* **Authentication** enables the server and client to check that the other party is who it claims to be. When a client initiates an SSL session, the server typically sends its certificate to the client. Certificates are digital identities that are issued by trusted certificate authorities, such as Verisign.

The client verifies that the server is authentic and not an imposter by validating the certificate chain in the server certificate. The server certificate is guaranteed by the certificate authority (CA) who signed the server certificate.

The server can also require the client to have a certificate, if the server needs to authenticate the identity of the client.

**About Private and Public Key Cryptography**

To provide message integrity, authentication, and encryption, SSL uses both private and public key cryptography.

* **Secret Key Cryptography**

Private, or symmetric, key cryptography requires a single, secret key shared by two or more parties to secure communication. This key is used to encrypt and decrypt secure messages sent between the parties. This requires prior and secure distribution of the key to each party. The problem with this method is that it is difficult to securely transmit and store the key.

In SSL, each party calculates the secret key individually using random values known to each side. The parties then send messages encrypted using the secret key.

* **Public Key Cryptography**

Public key cryptography solves this problem by employing public and private key pairs and a secure method for key distribution. The freely available public key is used to encrypt messages that can *only* be decrypted by the holder of the associated private key. The private key is securely stored, together with other security credentials, in an encrypted container such as an Oracle wallet.

Public key algorithms can guarantee the secrecy of a message, but they do not necessarily guarantee secure communication because they do not verify the identities of the communicating parties. To establish secure communication, it is important to verify that the public key used to encrypt a message does in fact belong to the target recipient. Otherwise, a third party can potentially eavesdrop on the communication and intercept public key requests, substituting its own public key for a legitimate key (the man-in-the-middle attack).

To avoid such an attack, it is necessary to verify the owner of the public key, a process called authentication. Authentication can be accomplished through a certificate authority (CA), which is a third party trusted by both of the communicating parties.

The CA issues public key certificates that contain an entity's name, public key, and certain other security credentials. Such credentials typically include the CA name, the CA signature, and the certificate effective dates (From Date, To Date).

The CA uses its private key to encrypt a message, while the public key is used to decrypt it, thus verifying that the message was encrypted by the CA. The CA public key is well known, and does not have to be authenticated each time it is accessed. Such CA public keys are stored in wallets.

**Keystores**

In Oracle Fusion Middleware, Oracle Virtual Directory uses a JKS keystore to store keys and certificates. Configuring SSL for Oracle Virtual Directory thus requires setting up and using JKS keystores.

**How SSL Sessions Are Conducted**

The SSL protocol has two phases: the **handshake phase** and the **data transfer phase**. The handshake phase authenticates the server and optionally the client, and establishes the cryptographic keys that will be used to protect the data to be transmitted in the data transfer phase.

When a client requests an SSL connection to a server, the client and server first exchange messages in the handshake phase. (A common scenario is a browser requesting a page using the https:// instead of http:// protocol from a server. The HTTPS protocol indicates the usage of SSL with HTTP.)

[Figure 6-1](http://docs.oracle.com/cd/E17904_01/core.1111/e10105/sslconfig.htm#CBDCACAA) shows the handshake messages for a typical SSL connection between a Web server and a browser. The following steps are shown in the figure:

1. The client sends a Hello message to the server.

* The message includes a list of algorithms supported by the client and a random number that will be used to generate the keys.

1. The server responds by sending a Hello message to the client. This message includes:

* The algorithm to use. The server selected this from the list sent by the client.
* A random number, which will be used to generate the keys.

1. The server sends its certificate to the client.
2. The client authenticates the server using the server's certificate.
3. The client generates a random value ("pre-master secret"), encrypts it using the server's public key, and sends it to the server.
4. The server uses its private key to decrypt the message to retrieve the pre-master secret.
5. The client and server separately calculate the keys that will be used in the SSL session.

These keys are not sent to each other because the keys are calculated based on the pre-master secret and the random numbers, which are known to each side. The keys include:

* Encryption key that the client uses to encrypt data before sending it to the server
* Encryption key that the server uses to encrypt data before sending it to the client
* Key that the client uses to create a message digest of the data
* Key that the server uses to create a message digest of the data

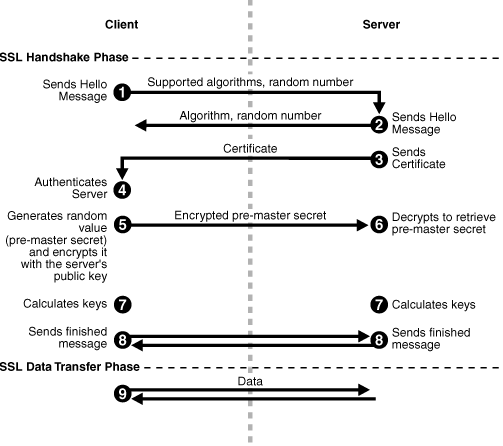
The encryption keys are symmetric, that is, the same key is used to encrypt and decrypt the data.

1. The client and server send a Finished message to each other. These are the first messages that are sent using the keys generated in the previous step (the first "secure" messages).

The Finished message includes all the previous handshake messages that each side sent. Each side verifies that the previous messages that it received match the messages included in the finished message. This checks that the handshake messages were not tampered with.

1. The client and server now transfer data using the encryption and hashing keys and algorithms.

Figure 6-1 SSL Handshake



About Cognizant:

* Cognizant Technology Solutions Corp. is an American [multinational](http://en.wikipedia.org/wiki/Multinational_corporation) provider of custom [information technology](http://en.wikipedia.org/wiki/Information_technology), consulting and [business process outsourcing](http://en.wikipedia.org/wiki/Business_process_outsourcing) services.
* Founder – Kumar Mahadeva
* Founded in 1994
* It is headquartered in [New Jersey](http://en.wikipedia.org/wiki/New_Jersey), USA.
* Cognizant saw a period of fast growth during the 2000s, becoming a Fortune 500 company in 2011.
* In 2011, the [Fortune magazine](http://en.wikipedia.org/wiki/Fortune_(magazine)) named it as the world's third most admired IT services company after [Accenture](http://en.wikipedia.org/wiki/Accenture) and [IBM](http://en.wikipedia.org/wiki/IBM).[[3]](http://en.wikipedia.org/wiki/Cognizant_Technology_Solutions#cite_note-3)
* The company has more than 150,000 employees globally, of which over 100,000 are in [India](http://en.wikipedia.org/wiki/India)Total Assets 5.50 Billion USD
* Total Revenue 6.12 Billion USD
* Cognizant provides information technology, consulting and BPO services. These include business & technology consulting, [systems integration](http://en.wikipedia.org/wiki/Systems_integration), application development & maintenance, IT infrastructure services, analytics, [business intelligence](http://en.wikipedia.org/wiki/Business_intelligence), [data warehousing](http://en.wikipedia.org/wiki/Data_warehousing), [CRM](http://en.wikipedia.org/wiki/Customer_relationship_management), [supply chain management](http://en.wikipedia.org/wiki/Supply_chain_management), engineering & manufacturing Solutions, [ERP](http://en.wikipedia.org/wiki/Enterprise_resource_planning), [R&D](http://en.wikipedia.org/wiki/Research_and_development) outsourcing, and [testing](http://en.wikipedia.org/wiki/Software_testing) solutions.
* The company has more than 150,000 employees globally, of which over 100,000 are in [India](http://en.wikipedia.org/wiki/India).
* The company has centers in Chennai, [Bangalore](http://en.wikipedia.org/wiki/Bangalore), [Coimbatore](http://en.wikipedia.org/wiki/Coimbatore), [Gurgaon](http://en.wikipedia.org/wiki/Gurgaon), [Hyderabad](http://en.wikipedia.org/wiki/Hyderabad,_India), [Kochi](http://en.wikipedia.org/wiki/Kochi), [Kolkata](http://en.wikipedia.org/wiki/Kolkata),[[39]](http://en.wikipedia.org/wiki/Cognizant_Technology_Solutions#cite_note-39) [Mangalore](http://en.wikipedia.org/wiki/Mangalore) (CoreLogic), [Mumbai](http://en.wikipedia.org/wiki/Mumbai) and [Pune](http://en.wikipedia.org/wiki/Pune).
* The company is operational in countries like UK, [Hungary](http://en.wikipedia.org/wiki/Hungary), [China](http://en.wikipedia.org/wiki/China), The Philippines, [Canada](http://en.wikipedia.org/wiki/Canada), Argentina, and Mexico.
* Acquisitions include

1. CoreLogic India
2. Galileo Performance
3. AimNet
4. Infopulse
5. Aces International
6. Active Intelligence
7. C1 Group

**UNIX UTILITIES**

When you work in a distributed environment then you need to communicate with remote users and you also need to access remote Unix machines.

There are several Unix utilities which are especially useful for users computing in a networked, distributed environment. This tutorial lists few of them:

**The Ping Utility:**

The ping command sends an echo request to a host available on the network. Using this command you can check if your remote host is responding well or not.

The ping command is useful for the following:

Tracking and isolating hardware and software problems.

Determining the status of the network and various foreign hosts.

Testing, measuring, and managing networks.

**Syntax:**

Following is the simple syntax to use ping command:

|  |
| --- |
| $ping hostname or ip-address |

Above command would start printing a response after every second. To come out of the command you can terminate it by pressing CNTRL + C keys.

**Example:**

Following is the example to check the availability of a host available on the network:

|  |
| --- |
| [amrood]$ping google.com  PING google.com (74.125.67.100) 56(84) bytes of data.  64 bytes from 74.125.67.100: icmp\_seq=1 ttl=54 time=39.4 ms  64 bytes from 74.125.67.100: icmp\_seq=2 ttl=54 time=39.9 ms  64 bytes from 74.125.67.100: icmp\_seq=3 ttl=54 time=39.3 ms  64 bytes from 74.125.67.100: icmp\_seq=4 ttl=54 time=39.1 ms  64 bytes from 74.125.67.100: icmp\_seq=5 ttl=54 time=38.8 ms  --- google.com ping statistics ---  22 packets transmitted, 22 received, 0% packet loss, time 21017ms  rtt min/avg/max/mdev = 38.867/39.334/39.900/0.396 ms  [amrood]$ |

If a host does not exist then it would behave something like this:

|  |
| --- |
| [amrood]$ping giiiiiigle.com  ping: unknown host giiiiigle.com  [amrood]$ |

**The ftp Utility:**

Here ftp stands for File Transfer Protocol. This utility helps you to upload and download your file from one computer to another computer.

The ftp utility has its own set of UNIX like commands which allow you to perform tasks such as:

Connect and login to a remote host.

Navigate directories.

List directory contents

Put and get files

Transfer files as ascii, ebcdic or binary

**Syntax:**

Following is the simple syntax to use ping command:

|  |
| --- |
| $ftp hostname or ip-address |

Above command would prompt you for login ID and password. Once you are authenticated, you would have access on the home directory of the login account and you would be able to perform various commands.

Few of the useful commands are listed below:

|  |  |
| --- | --- |
| Command | Description |
| put filename | Upload filename from local machine to remote machine. |
| get filename | Download filename from remote machine to local machine. |
| mput file list | Upload more than one files from local machine to remove machine. |
| mget file list | Download more than one files from remote machine to local machine. |
| prompt off | Turns prompt off, by default you would be prompted to upload or download movies using mput or mget commands. |
| prompt on | Turns prompt on. |
| dir | List all the files available in the current directory of remote machine. |
| cd dirname | Change directory to dirname on remote machine. |
| lcd dirname | Change directory to dirname on local machine. |
| quit | Logout from the current login. |

It should be noted that all the files would be downloaded or uploaded to or from current directories. If you want to upload your files in a particular directory then first you change to that directory and then upload required files.

**Example:**

Following is the example to show few commands:

|  |
| --- |
| [amrood]$ftp amrood.com  Connected to amrood.com.  220 amrood.com FTP server (Ver 4.9 Thu Sep 2 20:35:07 CDT 2009)  Name (amrood.com:amrood): amrood  331 Password required for amrood.  Password:  230 User amrood logged in.  ftp> dir  200 PORT command successful.  150 Opening data connection for /bin/ls.  total 1464  drwxr-sr-x 3 amrood group 1024 Mar 11 20:04 Mail  drwxr-sr-x 2 amrood group 1536 Mar 3 18:07 Misc  drwxr-sr-x 5 amrood group 512 Dec 7 10:59 OldStuff  drwxr-sr-x 2 amrood group 1024 Mar 11 15:24 bin  drwxr-sr-x 5 amrood group 3072 Mar 13 16:10 mpl  -rw-r--r-- 1 amrood group 209671 Mar 15 10:57 myfile.out  drwxr-sr-x 3 amrood group 512 Jan 5 13:32 public  drwxr-sr-x 3 amrood group 512 Feb 10 10:17 pvm3  226 Transfer complete.  ftp> cd mpl  250 CWD command successful.  ftp> dir  200 PORT command successful.  150 Opening data connection for /bin/ls.  total 7320  -rw-r--r-- 1 amrood group 1630 Aug 8 1994 dboard.f  -rw-r----- 1 amrood group 4340 Jul 17 1994 vttest.c  -rwxr-xr-x 1 amrood group 525574 Feb 15 11:52 wave\_shift  -rw-r--r-- 1 amrood group 1648 Aug 5 1994 wide.list  -rwxr-xr-x 1 amrood group 4019 Feb 14 16:26 fix.c  226 Transfer complete.  ftp> get wave\_shift  200 PORT command successful.  150 Opening data connection for wave\_shift (525574 bytes).  226 Transfer complete.  528454 bytes received in 1.296 seconds (398.1 Kbytes/s)  ftp> quit  221 Goodbye.  [amrood]$ |

**The telnet Utility:**

Many times you would be in need to connect to a remote Unix machine and work on that machine remotely. Telnet is a utility that allows a computer user at one site to make a connection, login and then conduct work on a computer at another site.

Once you are login using telnet, you can perform all the activities on your remotely connect machine. Here is example telnet session:

|  |
| --- |
| C:>telnet amrood.com  Trying...  Connected to amrood.com.  Escape character is '^]'.  login: amrood  amrood's Password:  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* \*  \* \*  \* WELCOME TO AMROOD.COM \*  \* \*  \* \*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Last unsuccessful login: Fri Mar 3 12:01:09 IST 2009  Last login: Wed Mar 8 18:33:27 IST 2009 on pts/10  { do your work }  [amrood]$ logout  Connection closed.  C:> |

**The finger Utility:**

The finger command displays information about users on a given host. The host can be either local or remote.

Finger may be disabled on other systems for security reasons.

Following are the simple syntax to use finger command:

Check all the logged in users on local machine as follows:

|  |
| --- |
| [amrood]$ finger  Login Name Tty Idle Login Time Office  amrood pts/0 Jun 25 08:03 (62.61.164.115) |

Get information about a specific user available on local machine:

|  |
| --- |
| [amrood]$ finger amrood  Login: amrood Name: (null)  Directory: /home/amrood Shell: /bin/bash  On since Thu Jun 25 08:03 (MST) on pts/0 from 62.61.164.115  No mail.  No Plan. |

Check all the logged in users on remote machine as follows:

|  |
| --- |
| [amrood]$ finger @avtar.com  Login Name Tty Idle Login Time Office  amrood pts/0 Jun 25 08:03 (62.61.164.115) |

Get information about a specific user available on remote machine:

|  |
| --- |
| [amrood]$ finger amrood@avtar.com  Login: amrood Name: (null)  Directory: /home/amrood Shell: /bin/bash  On since Thu Jun 25 08:03 (MST) on pts/0 from 62.61.164.115  No mail.  No Plan. |

* **To find out the folder/file size in UNIX**

**du –sh file/folder name**

**du –sh \*** size of all the contents in the current directory

* **Script to send an email to the support mailbox when new exceptions are found in log file**

vi new\_exception.sh

#!/bin/bash (Shebang)

#Script to send an email when new exceptions are found in the log file

curr\_date=$(date +”%Y-%m-%d”)

logfile=stdout.log.curr\_date.txt

curr\_time=$(date +”%H”) - 1:??:

grep “$curr\_time” | grep -v ‘KnownException’ | grep -v ‘KnownException1’ | grep -v ‘KnownException2’ | … grep -v ‘knownExceptionn’ /var/applogs/eShop/weblogic/managed?/$logfile **>** <outputfile>

if [ wc -l <outputfile> ]

then

mailx -s Unknown Exceptions Alert [sh\_vf\_eshop@vodafone.com](mailto:sh_vf_eshop@vodafone.com) **<**  <outputfile>

fi

:wq (Save the file)

Chmod +x new\_exception.sh

**Set a Cron Job for the above script**

Crontab -e

00 \* \* \* \* ./ new\_exception.sh

Press :wq (Save the cron job)

**Dos2Unix and Unix2Dos:**

* Sometimes we may need to move files between Windows and Unix systems.
* When a file which is moved from Windows to Unix is opened in an editor, “^M” characters will be displayed at the end of each line as follows:

This is Unix^M

This is Windows^M

* For this reason, some UNIX systems feature two utilities for converting these types of files
* Dos2unix
* Unix2dos

dos2unix foo foo.dos

**Script to monitor the number of Stuck Threads**

vi Stuck\_Thread.sh

#!/bin/bash

#Script to send an email when the number of Stuck Threads exceed the threshold value in the log file

curr\_date=$(date +”%Y-%m-%d”)

logfile=”stdout.log.$curr\_date.txt “

curr\_time=$(date +”%H”) -1:??:

grep -i “$curr\_time” | grep ‘STUCK’ /var/applogs/eShop/weblogic/managed?/$logfile **>** <outputfile>

if [ wc -l <outputfile> **>** 50]

then

mailx -s Stuck Threads Alert [sh\_vf\_eshop@vodafone.com](mailto:sh_vf_eshop@vodafone.com) **<**  <outputfile>

fi

:wq (Save the file)

Chmod +x Stuck\_Thread.sh

**Set a Cron Job for the above script**

Crontab -e

00 \* \* \* \* ./ Stuck\_Thread.sh

Press :wq (Save the cron job)

**UNIX - The vi Editor Tutorial**

There are many ways to edit files in Unix and for me one of the best ways is using screen-oriented text editor vi. This editor enable you to edit lines in context with other lines in the file.

Now a days you would find an improved version of vi editor which is called VIM. Here VIM stands for Vi IMproved.

The vi is generally considered the de facto standard in Unix editors because:

It's usually available on all the flavors of Unix system.

Its implementations are very similar across the board.

It requires very few resources.

It is more user friendly than any other editors like ed or ex.

You can use vi editor to edit an existing file or to create a new file from scratch. You can also use this editor to just read a text file.

**Starting the vi Editor:**

There are following way you can start using vi editor:

|  |  |
| --- | --- |
| Command | Description |
| vi filename | Creates a new file if it already does not exist, otherwise opens existing file. |
| vi -R filename | Opens an existing file in read only mode. |
| view filename | Opens an existing file in read only mode. |

Following is the example to create a new file testfile if it already does not exist in the current working directory:

|  |
| --- |
| [amrood]$vi testfile |

As a result you would see a screen something like as follows:

|  |
| --- |
| |  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~  "testfile" [New File] |

You will notice a tilde (~) on each line following the cursor. A tilde represents an unused line. If a line does not begin with a tilde and appears to be blank, there is a space, tab, newline, or some other nonviewable character present.

So now you have opened one file to start with. Before proceeding further let us understanding few minor but important concepts explained below.

**Operation Modes:**

While working with vi editor you would come across following two modes:

Command mode: This mode enables you to perform administrative tasks such as saving files, executing commands, moving the cursor, cutting (yanking) and pasting lines or words, and finding and replacing. In this mode, whatever you type is interpreted as a command.

Insert mode: This mode enables you to insert text into the file. Everything that's typed in this mode is interpreted as input and finally it is put in the file .

The vi always starts in command mode. To enter text, you must be in insert mode. To come in insert mode you simply type i. To get out of insert mode, press the Esc key, which will put you back into command mode.

Hint: If you are not sure which mode you are in, press the Esc key twice, and then you'll be in command mode. You open a file using vi editor and start type some characters and then come in command mode to understand the difference.

**Getting Out of vi:**

The command to quit out of vi is :q. Once in command mode, type colon, and 'q', followed by return. If your file has been modified in any way, the editor will warn you of this, and not let you quit. To ignore this message, the command to quit out of vi without saving is :q!. This lets you exit vi without saving any of the changes.

The command to save the contents of the editor is :w. You can combine the above command with the quit command, or :wq and return.

The easiest way to save your changes and exit out of vi is the hkl command. When you are in command mode, type ZZ and it will do the equivalent of :wq.

You can specify a different file name to save to by specifying the name after the :w. For example, if you wanted to save the file you were working as another filename called filename2, you would type :w filename2 and return. Try it once.

**Moving within a File:**

To move around within a file without affecting your text, you must be in command mode (press Esc twice). Here are some of the commands you can use to move around one character at a time:

|  |  |
| --- | --- |
| Command | Description |
| k | Moves the cursor up one line. |
| j | Moves the cursor down one line. |
| h | Moves the cursor to the left one character position. |
| l | Moves the cursor to the right one character position. |

There are following two important points to be noted:

The vi is case-sensitive, so you need to pay special attention to capitalization when using commands.

Most commands in vi can be prefaced by the number of times you want the action to occur. For example, 2j moves cursor two lines down the cursor location.

There are many other ways to move within a file in vi. Remember that you must be in command mode (press Esc twice). Here are some more commands you can use to move around the file:

|  |  |
| --- | --- |
| Command | Description |
| 0 or | | Positions cursor at beginning of line. |
| $ | Positions cursor at end of line. |
| w | Positions cursor to the next word. |
| b | Positions cursor to previous word. |
| ( | Positions cursor to beginning of current sentence. |
| ) | Positions cursor to beginning of next sentence. |
| E | Move to the end of Blank delimited word |
| { | Move a paragraph back |
| } | Move a paragraph forward |
| [[ | Move a section back |
| ]] | Move a section forward |
| n| | Moves to the column n in the current line |
| 1G | Move to the first line of the file |
| G | Move to the last line of the file |
| nG | Move to nth line of the file |
| :n | Move to nth line of the file |
| fc | Move forward to c |
| Fc | Move back to c |
| H | Move to top of screen |
| nH | Moves to nth line from the top of the screen |
| M | Move to middle of screen |
| L | Move to botton of screen |
| nL | Moves to nth line from the bottom of the screen |
| :x | Colon followed by a number would position the cursor on line number represented by x |

**Control Commands:**

There are following useful command which you can use along with Control Key:

|  |  |
| --- | --- |
| Command | Description |
| CTRL+d | Move forward 1/2 screen |
| CTRL+d | Move forward 1/2 screen |
| CTRL+f | Move forward one full screen |
| CTRL+u | Move backward 1/2 screen |
| CTRL+b | Move backward one full screen |
| CTRL+e | Moves screen up one line |
| CTRL+y | Moves screen down one line |
| CTRL+u | Moves screen up 1/2 page |
| CTRL+d | Moves screen down 1/2 page |
| CTRL+b | Moves screen up one page |
| CTRL+f | Moves screen down one page |
| CTRL+I | Redraws screen |

**Editing Files:**

To edit the file, you need to be in the insert mode. There are many ways to enter insert mode from the command mode:

|  |  |
| --- | --- |
| Command | Description |
| i | Inserts text before current cursor location. |
| I | Inserts text at beginning of current line. |
| a | Inserts text after current cursor location. |
| A | Inserts text at end of current line. |
| o | Creates a new line for text entry below cursor location. |
| O | Creates a new line for text entry above cursor location. |

**Deleting Characters:**

Here is the list of important commands which can be used to delete characters and lines in an opened file:

|  |  |
| --- | --- |
| Command | Description |
| x | Deletes the character under the cursor location. |
| X | Deletes the character before the cursor location. |
| dw | Deletes from the current cursor location to the next word. |
| d^ | Deletes from current cursor position to the beginning of the line. |
| d$ | Deletes from current cursor position to the end of the line. |
| D | Deletes from the cursor position to the end of the current line. |
| dd | Deletes the line the cursor is on. |

As mentioned above, most commands in vi can be prefaced by the number of times you want the action to occur. For example, 2x deletes two character under the cursor location and 2dd deletes two lines the cursor is on.

I would highly recommend to exercise all the above commands properly before proceeding further.

**Change Commands:**

You also have the capability to change characters, words, or lines in vi without deleting them. Here are the relevant commands:

|  |  |
| --- | --- |
| Command | Description |
| cc | Removes contents of the line, leaving you in insert mode. |
| cw | Changes the word the cursor is on from the cursor to the lowercase w end of the word. |
| r | Replaces the character under the cursor. vi returns to command mode after the replacement is entered. |
| R | Overwrites multiple characters beginning with the character currently under the cursor. You must use Esc to stop the overwriting. |
| s | Replaces the current character with the character you type. Afterward, you are left in insert mode. |
| S | Deletes the line the cursor is on and replaces with new text. After the new text is entered, vi remains in insert mode. |

**Copy and Paste Commands:**

You can copy lines or words from one place and then you can past them at another place using following commands:

|  |  |
| --- | --- |
| Command | Description |
| yy | Copies the current line. |
| yw | Copies the current word from the character the lowercase w cursor is on until the end of the word. |
| p | Puts the copied text after the cursor. |
| P | Puts the yanked text before the cursor. |

**Advanced Commands:**

There are some advanced commands that simplify day-to-day editing and allow for more efficient use of vi:

|  |  |
| --- | --- |
| Command | Description |
| J | Join the current line with the next one. A count joins that many lines. |
| << | Shifts the current line to the left by one shift width. |
| >> | Shifts the current line to the right by one shift width. |
| ~ | Switch the case of the character under the cursor. |
| ^G | Press CNTRL and G keys at the same time to show the current filename and the status. |
| U | Restore the current line to the state it was in before the cursor entered the line. |
| u | Undo the last change to the file. Typing 'u' again will re-do the change. |
| J | Join the current line with the next one. A count joins that many lines. |
| :f | Displays current position in the file in % and file name, total number of file. |
| :f filename | Renames current file to filename. |
| :w filename | Write to file filename. |
| :e filename | Opens another file with filename. |
| :cd dirname | Changes current working directory to dirname. |
| :e # | Use to toggle between two opened files. |
| :n | In case you open multiple files using vi, use :n to go to next file in the series. |
| :p | In case you open multiple files using vi, use :p to go to previous file in the series. |
| :N | In case you open multiple files using vi, use :N to go to previous file in the series. |
| :r file | Reads file and inserts it after current line |
| :nr file | Reads file and inserts it after line n. |

**Word and Character Searching:**

The vi editor has two kinds of searches: string and character. For a string search, the / and ? commands are used. When you start these commands, the command just typed will be shown on the bottom line, where you type the particular string to look for.

These two commands differ only in the direction where the search takes place:

The / command searches forwards (downwards) in the file.

The ? command searches backwards (upwards) in the file.

The n and N commands repeat the previous search command in the same or opposite direction, respectively. Some characters have special meanings while using in search command and preceded by a backslash (\) to be included as part of the search expression.

|  |  |
| --- | --- |
| Character | Description |
| ^ | Search at the beginning of the line. (Use at the beginning of a search expression.) |
| . | Matches a single character. |
| \* | Matches zero or more of the previous character. |
| $ | End of the line (Use at the end of the search expression.) |
| [ | Starts a set of matching, or non-matching expressions. |
| < | Put in an expression escaped with the backslash to find the ending or beginning of a word. |
| > | See the '<' character description above. |

The character search searches within one line to find a character entered after the command. The f and F commands search for a character on the current line only. f searches forwards and F searches backwards and the cursor moves to the position of the found character.

The t and T commands search for a character on the current line only, but for t, the cursor moves to the position before the character, and T searches the line backwards to the position after the character.

**Set Commands:**

You can change the look and feel of your vi screen using the following :set commands. To use these commands you have to come in command mode then type :set followed by any of the following options:

|  |  |
| --- | --- |
| Command | Description |
| :set ic | Ignores case when searching |
| :set ai | Sets autoindent |
| :set noai | To unset autoindent. |
| :set nu | Displays lines with line numbers on the left side. |
| :set sw | Sets the width of a software tabstop. For example you would set a shift width of 4 with this command: :set sw=4 |
| :set ws | If wrapscan is set, if the word is not found at the bottom of the file, it will try to search for it at the beginning. |
| :set wm | If this option has a value greater than zero, the editor will automatically "word wrap". For example, to set the wrap margin to two characters, you would type this: :set wm=2 |
| :set ro | Changes file type to "read only" |
| :set term | Prints terminal type |
| :set bf | Discards control characters from input |

**Running Commands:**

The vi has the capability to run commands from within the editor. To run a command, you only need to go into command mode and type :! command.

For example, if you want to check whether a file exists before you try to save your file to that filename, you can type :! ls and you will see the output of ls on the screen.

When you press any key (or the command's escape sequence), you are returned to your vi session.

**Replacing Text:**

The substitution command (:s/) enables you to quickly replace words or groups of words within your files. Here is the simple syntax:

|  |
| --- |
| :s/search/replace/g |

The g stands for globally. The result of this command is that all occurrences on the cursor's line are changed.

**IMPORTANT:**

Here are the key points to your success with vi:

You must be in command mode to use commands. (Press Esc twice at any time to ensure that you are in command mode.)

You must be careful to use the proper case (capitalization) for all commands.

You must be in insert mode to enter text.

**IMPORTANT COMMANDS:**

**Netstat:** Netstat command displays various network related information such as network connections, routing tables, interface statistics, masquerade connections, multicast memberships etc.,

Netstat provides statistics for the following:

* Proto - The name of the protocol ([TCP](http://en.wikipedia.org/wiki/Transmission_Control_Protocol) or [UDP](http://en.wikipedia.org/wiki/User_Datagram_Protocol)).
* Local Address - The [IP](http://en.wikipedia.org/wiki/Internet_Protocol) address of the local computer and the port number being used. The name of the local computer that corresponds to the [IP](http://en.wikipedia.org/wiki/Internet_Protocol) address and the name of the port is shown unless the **-n** parameter is specified. If the port is not yet established, the port number is shown as an asterisk (\*).
* Foreign Address - The [IP](http://en.wikipedia.org/wiki/Internet_Protocol) address and port number of the remote computer to which the socket is connected. The names that corresponds to the [IP](http://en.wikipedia.org/wiki/Internet_Protocol) address and the port are shown unless the **-n** parameter is specified. If the port is not yet established, the port number is shown as an asterisk (\*).
* State - Indicates the state of a [TCP](http://en.wikipedia.org/wiki/Transmission_Control_Protocol) connection. The possible states are as follows: CLOSE\_WAIT, CLOSED, ESTABLISHED, FIN\_WAIT\_1, FIN\_WAIT\_2, LAST\_ACK, LISTEN, SYN\_RECEIVED, SYN\_SEND, and TIME\_WAIT.

**Variants:**

**netstat -i**  Displays information for all configured interfaces

**netstat -I interface**  Displays current status of the specified interface

**netstat -A**  Displays socket information

**netstat -a**  Displays active and inactive sockets

**netstat -m**  Displays memory usage

**netstat -r**  Displays all routes

**netstat -s**  Displays statistics for all protocols

**netstat -i -p protocol\_id** Displays protocol statistics for a particular protocol

**Examples:**

**1. List All Ports (both listening and non listening ports)**

List all ports using **netstat –a**

# **netstat -a | more**

Active Internet connections (servers and established)

Proto Recv-Q Send-Q Local Address Foreign Address State

tcp 0 0 localhost:30037 \*:\* LISTEN

udp 0 0 \*:bootpc \*:\*

List all tcp ports using **netstat -at**

# **netstat -at**

Active Internet connections (servers and established)

Proto Recv-Q Send-Q Local Address Foreign Address State

tcp 0 0 localhost:30037 \*:\* LISTEN

tcp 0 0 localhost:ipp \*:\* LISTEN

tcp 0 0 \*:smtp \*:\* LISTEN

tcp6 0 0 localhost:ipp [::]:\* LISTEN

List all udp ports using **netstat -au**

# **netstat -au**

Proto Recv-Q Send-Q Local Address Foreign Address State

udp 0 0 \*:bootpc \*:\*

udp 0 0 \*:49119 \*:\*

**2. List Sockets which are in Listening State**

List only listening ports using **netstat -l**

List only listening TCP Ports using **netstat -lt**

List only listening UDP Ports using **netstat –lu**

List only the listening UNIX Ports using **netstat –lx**

**3. Show the statistics for each protocol**

Show statistics for all ports using **netstat -s**

# **netstat -s**

Ip:

11150 total packets received

1 with invalid addresses

0 forwarded

0 incoming packets discarded

11149 incoming packets delivered

11635 requests sent out

Icmp:

0 ICMP messages received

0 input ICMP message failed.

Tcp:

582 active connections openings

2 failed connection attempts

25 connection resets received

Udp:

1183 packets received

4 packets to unknown port received.

.....

**Show statistics for TCP (or) UDP ports using netstat -st (or) -su**

**# netstat -st**

**# netstat -su**

**4**. Display **PID** and **program** **names** in **netstat** output using **netstat -p**

**netstat -p** option can be combined with any other **netstat** option. This will add the “PID/Program Name” to the netstat output. This is very useful while debugging to identify which program is running on a particular port.

# **netstat -pt**

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name

tcp 1 0 ramesh-laptop.loc:47212 192.168.185.75:www CLOSE\_WAIT 2109/firefox

tcp 0 0 ramesh-laptop.loc:52750 lax:www ESTABLISHED 2109/firefox

**5. Don’t resolve host, port and user name in netstat output**

When you don’t want the name of the host, port or user to be displayed, use netstat -n option. This will display in numbers, instead of resolving the host name, port name, user name.

This also speeds up the output, as netstat is not performing any look-up.

# **netstat -an**

If you don’t want only any one of those three items ( ports, or hosts, or users ) to be resolved, use following commands.

**# netsat -a --numeric-ports**

**# netsat -a --numeric-hosts**

**# netsat -a --numeric-users**

**6. Print netstat information continuously**

netstat will print information continuously every few seconds.

# **netstat -c**

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address Foreign Address State

tcp 0 0 ramesh-laptop.loc:36130 101-101-181-225.ama:www ESTABLISHED

tcp 1 1 ramesh-laptop.loc:52564 101.11.169.230:www CLOSING

tcp 0 0 ramesh-laptop.loc:43758 server-101-101-43-2:www ESTABLISHED

tcp 1 1 ramesh-laptop.loc:42367 101.101.34.101:www CLOSING

^C

**7. Find out on which port a program is running**

# **netstat -ap | grep ssh**

(Not all processes could be identified, non-owned process info

will not be shown, you would have to be root to see it all.)

tcp 1 0 dev-db:ssh 101.174.100.22:39213 CLOSE\_WAIT -

tcp 1 0 dev-db:ssh 101.174.100.22:57643 CLOSE\_WAIT -

Find out which process is using a particular port:

# **netstat -an | grep ':80'**

Sample Output:

cbp031.904 wdcprodhome.nfsd 98304 0 49640 0 ESTABLISHED

**cbp031 - local system  
wdcprodhome - remote system**

**904: - peer port number,   
98304: - send window  
0: - send queue  
49640: - receive window  
0: - receive queue  
ESTABLISHED: - connection status**

**netstat**

Displays generic net statistics of the host you are currently connected to.

**netstat -an**

Shows all connections to the server including the source and destination ips and ports if you have proper permissions.

**netstat -rn**

Displays routing table for all ips bound to the server.

**netstat -an |grep :80 |wc -l**

Display the amount of active connections on [port](http://www.computerhope.com/jargon/p/port.htm) 80. Removing the [pipe](http://www.computerhope.com/jargon/p/pipe.htm) and [wc command](http://www.computerhope.com/unix/uwc.htm) would display each connection.

**netstat -natp**

Display active Internet connections.

## ps: displays current process status

To list all your processes, enter:

**ps**

To list all processes, enter:

**ps aux**

## uname: displays information about the operating system

**uname [-amnrsv]**

-a Displays all information specified with the -m, -n, -r, -s, and -v

options.

-m Displays the type of hardware running the system.

-n Displays the name of the node (this may be a name that the system is

known by to a communications network).

-r Displays the release number of the operating system.

-s Displays the name of the implementation of the operating system. (This

option is on by default.)

-v Displays the operating system version.

## kill: sends a signal to a running process

To kill a process with process id 21234

$ **kill -KILL 21234**

To stop a process with process id 21234

$ **kill -STOP 21234**

To resume a stopped process with process id 21234

$ **kill -CONT 21234**

## file: determine the kind of a file

file filename

Determines what kind of file filename is, if binary, text, if object file, if source

in Java ..

## nslookup: queries Internet name servers interactively

To find the dotted decimal IP address of host name snowhite.cis.temple.edu

$ **nslookup snowhite.cis.temple.edu**

To find the hostname of the dotted decimal IP address 155.247.190.207

$ **nslookup 155.247.190.207**

## ping: Sends ICMP ECHO\_REQUEST packets to network hosts

To determine the status of host www.cs.stanford.edu:

$ **ping www.cs.stanford.edu**

## traceroute: Print the route that packets take to the network host

To find the route to host rtfm.mit.edu, enter:

$ **traceroute rtfm.mit.edu**

## lsof: list information about files opened by processes

This is a complex command not available on all unix (you can obtain a copy from <ftp://vic.cc.purdue.edu/pub/tools/lsof/>).

To list all open files, use:

**lsof**

To list all open Internet, x.25 (HP-UX), and UNIX domain files, use:

**lsof -i -U**

To list all files using any protocol on any port of snowhite.cis.temple.edu, use:

**lsof -i @snowhite.cis.temple.edu**

To find the process that has /u/abe/foo open, use:

**lsof /u/abe/foo**

To find any open file, including an open UNIX domain socket file, with the

name /dev/log, use:

**lsof /dev/log**

To obtain PID and command name field output for each process, file descrip-

tor, file device number, and file inode number for each file of each pro-

cess, use:

**lsof –FpcfDi**

## vmstat: displays virtual memory statistics

**vmstat interval [count]**

**vmstat [-f|-M|-P|-s]**

where:

-f Displays only statistics about the number of forks since system startup

(see the fork() call).

-P Displays accumulated statistics about physical memory use.

-s Displays accumulated statistics along with the page size.

## df: Displays statistics on free disk space

## du:Displays a summary of disk usage

To display the disk usage of a directory tree and each of its sub-

trees, enter:

$ **du /u/fran**

To display the disk usage of each file, enter:

$ **du -a /u/fran**

To display only the total disk usage of a directory tree, enter:

$ **du -rs /u/fran**

**Ex:**

The following example shows the vmstat display of statistics that were gathered at five-second intervals:

$ **vmstat 5**

kthr memory page disk faults cpu

r b w swap free re mf pi po fr de sr dd f0 s1 -- in sy cs us sy id

0 0 0 863160 365680 0 3 1 0 0 0 0 0 0 0 0 406 378 209 1 0 99

0 0 0 765640 208568 0 36 0 0 0 0 0 0 0 0 0 479 4445 1378 3 3 94

0 0 0 765640 208568 0 0 0 0 0 0 0 0 0 0 0 423 214 235 0 0 100

0 0 0 765712 208640 0 0 0 0 0 0 0 3 0 0 0 412 158 181 0 0 100

0 0 0 765832 208760 0 0 0 0 0 0 0 0 0 0 0 402 157 179 0 0 100

0 0 0 765832 208760 0 0 0 0 0 0 0 0 0 0 0 403 153 182 0 0 100

0 0 0 765832 208760 0 0 0 0 0 0 0 0 0 0 0 402 168 177 0 0 100

0 0 0 765832 208760 0 0 0 0 0 0 0 0 0 0 0 402 153 178 0 0 100

0 0 0 765832 208760 0 18 0 0 0 0 0 0 0 0 0 407 165 186 0 0 100

## du

- display the number of disk blocks used per directory or file

## Description

The du utility gives the number of kilobytes contained in all files and, recursively, directories within each specified directory or file *filename*. If *filename* is missing, `.' (the current directory) is used.

A file that has multiple links to it is only counted once.

## Options

The following options are supported:

**-a**

Generates an entry for each file.

**-d**

Does not cross file system boundaries. For example, du -d / reports usage only on the root partition.

**-k**

Writes the files sizes in units of 1024 bytes, rather than the default 512-byte units.

**-L**

Processes symbolic links by using the file or directory that the symbolic link references, rather than the link itself.

**-o**

Does not add child directories' usage to a parent's total. Without this option, the usage listed for a particular directory is the space taken by the files in that directory, as well as the files in all directories beneath it. This option does nothing if the -s option is used.

**-r**

Generates messages about directories that cannot be read, files that cannot be opened, and so forth, rather than being silent (the default).

**-s**

Only displays the grand total for each of the specified *filename*s.

Entries are generated only for each directory in the absence of options.

## df

- display status of disk space on file systems

## Description

The df utility displays the amount of disk space occupied by currently mounted file systems, the amount of used and available space, and how much of the file system's total capacity has been used.

If arguments to df are path names, df produces a report on the file system containing the named file. Thus `df .' shows the amount of space on the file system containing the current directory.

## Options

The following options are supported:

**-a**

Report on all filesystems including the uninteresting ones which have zero total blocks (that is, auto-mounter).

**-i**

Report the number of used and free inodes. Print ` \* ' if no information is available.

**-t** *type*

Report on filesystems of a given type (for example, nfs or ufs).

Ex:

Filesystem kbytes used avail capacity Mounted on

sparky:/ 7445 4714 1986 70% /

sparky:/usr 42277 35291 2758 93% /usr

## Example 1 Showing usage of all subdirectories in a directory

example% **pwd**

/usr/ralph/misc

example% **du**

5 ./jokes

33 ./squash

44 ./tech.papers/lpr.document

217 ./tech.papers/new.manager

401 ./tech.papers

388 ./window

93 ./messages

15 ./useful.news

1211 .

## mesg

- permit or deny messages

Usage: mesg [-n | -y | n | y]

## Description

The mesg utility will control whether other users are allowed to send messages via [write(1)](http://docs.oracle.com/cd/E23824_01/html/821-1461/write-1.html#scrolltoc), [talk(1)](http://docs.oracle.com/cd/E23824_01/html/821-1461/talk-1.html#scrolltoc), or other utilities to a terminal device. The terminal device affected is determined by searching for the first terminal in the sequence of devices associated with standard input, standard output, and standard error, respectively. With no arguments, mesg reports the current state without changing it. Processes with appropriate privileges may be able to send messages to the terminal independent of the current state.

## ps

- report process status

## Description

The ps command prints information about active processes. Without options, ps prints information about processes that have the same effective user ID and the same controlling terminal as the invoker. The output contains only the process ID, terminal identifier, cumulative execution time, and the command name. Otherwise, the information that is displayed is controlled by the options.

Some options accept lists as arguments. Items in a list can be either separated by commas or else enclosed in quotes and separated by commas or spaces. Values for *proclist* and *grplist* must be numeric.

## Options

**-a**

Lists information about all processes most frequently requested: all those except session leaders and processes not associated with a terminal.

**-A**

Lists information for all processes.Identical to -e, below.

**-e**

Lists information about every process now running.

When the -e option is specified, options -z, -t, -u, -U, -g, -G, -p, -g, -s and -a options have no effect.

**-f**

Generates a full listing.

## sar

- system activity reporter

## Description

In the first instance, the sar utility samples cumulative activity counters in the operating system at *n* intervals of *t* seconds, where *t* should be 5 or greater. If *t* is specified with more than one option, all headers are printed together and the output can be difficult to read. (If the sampling interval is less than 5, the activity of sar itself can affect the sample.) If the -o option is specified, it saves the samples in *filename* in binary format. The default value of *n* is 1.

In the second instance, no sampling interval is specified. sar extracts data from a previously recorded *filename*, either the one specified by the -f option or, by default, the standard system activity daily data file /var/adm/sa/sa*dd* for the current day *dd*. The starting and ending times of the report can be bounded using the -e and -s arguments with *time* specified in the form *hh*[:*mm*[:*ss*]]. The -i option selects records at *sec* second intervals. Otherwise, all intervals found in the data file are reported.

## Options

**-u**

Reports CPU utilization (the default):

**%usr, %sys, %wio, %idle**

portion of time running in user mode, running in system mode, idle with some process waiting for block I/O, and otherwise idle.

If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.

Ex: sar – u 1

## How do I find the port which is used by a particular application?

To make things simpler here’s an example where Tomcat application is running on my server. It is using port 8080.  
  
To check whether the Tomcat is running or not.  
  
Initially you need to understand the output presented by   
netstat -ntpl .  
In my case Tomcat is using java as their process so i grep with java. Your application might be using different name.   
#netstat -ntpl | grep java

tcp 0 0 ::ffff:127.0.0.1:8005 :::\* LISTEN 6375/java  
tcp 0 0 :::8009 :::\* LISTEN 6375/java  
tcp 0 0 :::8080 :::\* LISTEN 6375/java  
  
You should see at least one java process and you can use **ps** to identify if this is Tomcat.  
  
**# ps -ef | grep 6375**

root 6375 1 0 May18 pts/2 00:01:06 /usr/java/jdk1.6.0\_20/bin/java -Djava.util.logging.config.file=/install/apache-tomcat-5.5.29/conf/logging.properties -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djava.endorsed.dirs=/install/apache-tomcat-5.5.29/common/endorsed -classpath /install/apache-tomcat-5.5.29/bin/bootstrap.jar -Dcatalina.base=/install/apache-tomcat-5.5.29 -Dcatalina.home=/install/apache-tomcat-5.5.29 -Djava.io.tmpdir=/install/apache-tomcat-5.5.29/temp org.apache.catalina.startup.Bootstrap start  
root 9222 5091 0 18:29 pts/2 00:00:00 grep 6375

## vi / vim show line number command

To display line numbers along the left side of a window, type any one of the following:  
:set number  
or  
:set nu

To turn off line number again enter the same command:  
:set nu!  
If you need number every time you start vi/vim, append following line to your ~/.vimrc file:  
set number  
Save and close the file.

Jump to particular line number from a shell prompt, enter:  
$ vi +linenumber file.c  
$ vi +300 initlib.c